

# Panorama Essentials

Panorama Essentials  
Copyright © 2007, ProVUE Development,  
All Rights Reserved

ProVUE Development  
18685-A Main Street PMB 356  
Huntington Beach, CA 92648  
USA

[www.provue.com](http://www.provue.com)



# Table of Contents



– Click on any entry to jump to the page –

## Table of Contents 1

### Welcome to Panorama! 17

What is a Database?.....	17
A Brief History of Database Technology .....	18
What is Panorama? .....	19
Wizards.....	20
Super Fast Searching and Sorting.....	20
Phonetic Searching.....	20
Easy Set-Up.....	21
Crosstabs.....	21
Data Outlines .....	22
Data Entry Shortcuts.....	22
Smart Dates™ .....	23
Charts .....	23
Relational Links .....	23
Internet Content.....	24
Map, Phone, Email and VCard Support.....	24
Advanced Graphic Tools .....	25
Mail Merge and Mailing Labels .....	25
Database Reports.....	25
View-As-List Forms.....	25
Elastic Forms .....	26
Matrix Display .....	27
Images and Movies.....	27
High Speed Import.....	27
Flexible Text Export (including HTML).....	27
Data Transformation .....	27
Select/Remove Duplicates.....	27
Compact Storage.....	27
Formulas.....	27
Smart Program Recorder.....	28
Complete Programming Language and Development Tools.....	28
Custom Menus, Buttons, and Dialogs .....	28
Seamless Cross Platform Operation .....	28
Panorama Enterprise Server .....	29

Getting Started With Panorama For First Time Users .....	30
Upgrading to Panorama 5.5 From an Earlier Version.....	31
Deploying Applications Built With Panorama.....	32
Panorama Direct.....	32
Panorama Engine.....	32
Panorama Engine vs. Panorama Direct .....	32
<b>Files and Memory.....</b>	<b>33</b>
Files, Icons and the Desktop.....	33
Launching Panorama.....	34
Changing the Default Launch Action .....	35
Opening a Database .....	35
Databases and RAM.....	36
The Wizard Menu.....	36
The <b>Recent Databases</b> Wizard .....	37
The <b>Favorite Databases</b> Wizard .....	37
Creating a New Database.....	38
Using the <b>New Database</b> Wizard .....	38
Creating a Database with the Wizard.....	39
Creating Numeric and Date Fields .....	40
Default Values .....	40
Automatic Calculations.....	41
Adding Fields with the Add Field Dialog.....	42
Starting with a New Database Template .....	43
Creating a Database from a Text File .....	45
New Database Options (Font, Windows, Save As).....	46
Creating a New Database with the Open File dialog.....	47
Closing Panorama .....	48
Saving a Database.....	48
Revert to Saved.....	49
Auto-Save.....	50
Pitfalls of Auto-Save .....	50
Finding a Database on the Hard Disk.....	50
Working with Multiple Databases.....	51
Opening Multiple Files .....	51
File Sets.....	52
Saving Multiple Files.....	53
Importing and Exporting Data .....	54
Working with Text Files.....	54
Importing a Text File .....	54
Using the Text Import Wizard.....	55
Common Import Formulas.....	59
Import Templates .....	60
Choosing a Database to Import Into.....	61
Converting an Import Configuration into a Procedure .....	62
Importing Data from Microsoft Excel.....	62
Importing Financial Data (QIF, OFX, QFX).....	62
Importing VCard Data .....	62
Exporting a Text File.....	63
Exporting with the <b>Text Export Wizard</b> .....	<b>65</b>
Editing the Export Configuration.....	69
Common Export Formulas .....	71
Export Templates .....	71



Choosing a Database to Export From .....	72
Exporting Data to Microsoft Excel.....	72
Exporting VCard Data .....	72
Monitoring Memory Usage .....	73
Memory Usage Details .....	74
Adjusting Panorama's Database Memory Allocation .....	75
<b>Windows.....</b>	<b>77</b>
Window Components.....	77
Tool Palette.....	78
Scrolling the Tool Palette .....	78
Scroll Bars .....	79
Instant Jump to any Scroll Bar Position.....	79
Note: Most Mac OS X applications support this behaviour, including Safari, Preview, GarageBand, etc.....	79
Horizontal Scrollwheel Scrolling with Shift Key .....	79
Mighty Mouse and Trackpad Scrolling Support.....	79
Splitting a Window .....	80
Info Palette.....	81
<b>Views.....</b>	<b>83</b>
Types of Panorama Views .....	83
Data Sheet and Form Views .....	84
Other Views .....	85
Switching Between Views .....	86
Opening More Than One Window Per Database .....	87
The View Wizard.....	88
Form Modes: Data Access vs. Graphic Design .....	90
Form Operation: Individual Pages vs.View-As-List.....	91
Creating a New Form, Crosstab or Procedure.....	92
Renaming a Form, Crosstab or Procedure .....	92
Deleting a Form, Crosstab or Procedure .....	93
Changing the Order of Forms, Crosstabs or Procedures .....	93
<b>Records.....</b>	<b>95</b>
Data Organization .....	95
Tables vs. Individual Pages .....	96
Special Records.....	96
Data Records.....	96
Summary Records .....	97
Invisible Records .....	98
<b>Fields.....</b>	<b>101</b>
The Setup Menu .....	101
Add Field.....	102
Field Properties.....	103
Delete Field.....	104
Changing the Width of a Field.....	104
The Design Sheet .....	105
Database "Generations" .....	105
Typical Design Sheet Operation .....	106
Field Properties.....	108

Adding New Fields Using the Design Sheet .....	109
Removing Fields Using the Design Sheet .....	110
Re-Arranging Fields .....	110
Rules for Field Names .....	111
Repeating Fields (Line Items) .....	111
<b>Data Types .....</b>	<b>113</b>
Data Types and Memory Usage .....	114
Setting Up a Field's Data Type .....	114
Data Type Conversion Problems .....	115
Numeric Data .....	116
Money .....	117
Numeric Output Patterns .....	117
Fixed Decimal Point Patterns .....	119
Numbers with Commas, Punctuation, and Measurement Units .....	119
Scientific Notation .....	119
Special Patterns for Negative Numbers .....	120
Leading Zeros .....	120
Plural Suffixes .....	120
Displaying Numbers as Words .....	120
Dates .....	121
Entering Dates .....	121
Default Year and Century .....	121
Date Output Patterns .....	121
Date Pattern Components .....	122
Common Date Output Patterns .....	123
<b>Data Entry &amp; Editing .....</b>	<b>125</b>
Editing Records .....	125
Moving From Record to Record .....	125
Moving from Field to Field .....	127
Adding a New Record .....	128
Inserting a New Record .....	128
Deleting a Record .....	129
Deleting Multiple Records .....	130
Delete All .....	130
Duplicating a Record .....	131
Editing Data Within a Cell .....	131
The Input Box .....	132
Expanding the Input Box .....	132
Expanding a Right Justified Input Box .....	134
Editing Cells Within a Form .....	134
Data Entry Accelerators .....	135
Automatic Capitalization .....	136
Checking for Duplicate Data .....	136
Checking for Duplicates in Existing Data .....	137
Clairvoyance® .....	137
How Clairvoyance® Works .....	137
Turning Clairvoyance® On or Off .....	138
Clairvoyance® Helps Insure Data Consistency .....	138
Clairrows .....	139
Input Patterns .....	139
Entering Data with an Input Pattern .....	141

Using Input Patterns with Dates .....	141
Restricting Character Types .....	141
Custom Character Restrictions .....	143
Default Values .....	144
Default to Today's Date .....	145
"Ditto" Defaults Based on the Previous Record .....	146
Automatically Incrementing Defaults (1, 2, 3, ...) Based on the Previous Record .....	147
Creating a Unique Record Number .....	148
Automatic Calculations .....	149
Spreadsheet Mode Calculations .....	149
Procedure Mode Calculations .....	151
<b>Sorting .....</b>	<b>153</b>
Basic Sorting .....	153
Sorting By More Than One Field .....	154
Undo Sorting .....	157
Sorting Numbers and Dates .....	157
Sorting Right Justified Text .....	157
Sorting Selected Data .....	157
Sorting Within Groups .....	157
<b>Searching and Selecting .....</b>	<b>159</b>
Finding vs. Selecting .....	159
The Find/Select Dialog .....	160
Locating Dates by Month, Quarter, or Year .....	163
Find and Find Next .....	164
Select .....	165
Multiple Find/Select Criteria .....	166
Select Within .....	167
Select Additional .....	167
Select Reverse .....	167
Undo Select .....	167
Permanently Removing Unselected Data .....	167
The Live Clairvoyance™ Wizard .....	168
Basic Live Searching .....	168
Live Formula Searching .....	170
The Fields Menu .....	173
Selecting Search Results in the Original Database .....	174
The Favorite Searches Pop-up Menu .....	175
The Live Clairvoyance Preferences Dialog .....	177
Customizing Live Clairvoyance Basic Search .....	177
Customizing Live Clairvoyance Data Display .....	178
Customizing Data Display with Formulas .....	179
Customizing the Favorite Searches Pop-up Menu .....	180
Changing the Target Database .....	180
Formula Find/Select .....	181
Select Duplicates .....	181
<b>Summaries and Outlines .....</b>	<b>183</b>
3-Step Summarizing .....	183
STEP 1 - GROUP .....	189
Subgroups .....	189

Grand Total.....	189
The Group Command.....	190
Grouping by Week, Month, Quarter, or Year.....	190
Propagating Data into Summary Records.....	191
<b>STEP 2 - CALCULATE.....</b>	<b>192</b>
Total.....	192
Count.....	192
Average.....	192
Minimum.....	192
Maximum.....	192
Recalculating Summaries.....	192
Running Total.....	193
Using Running Total to Balance a Checkbook.....	193
<b>STEP 3 - OUTLINE.....</b>	<b>194</b>
The Outline Level.....	194
Collapsing vs. Selecting.....	198
Expanding and Collapsing Specific Details.....	198
Sorting by Summary Value.....	202
Sorting Within Groups.....	206
Getting Rid of Summary Records.....	206
Getting Rid of Detail.....	206
Printing Reports with Summary Information.....	207
<b>Crosstabs.....</b>	<b>209</b>
Category and Tabulation Fields.....	211
Creating and Setting Up a New Crosstab View.....	212
Crosstabs by Day, Month, Quarter or Year.....	216
Changing the Crosstab Design.....	216
Re-Calculating a Crosstab.....	217
Adjusting Crosstab Column Widths.....	217
Selecting Original Data.....	217
<b>Data Processing.....</b>	<b>221</b>
Transforming Selected Data.....	221
Filling a Field with a Fixed Value.....	222
Filling a Field with a Formula.....	223
Numeric Calculations With Formula Fill.....	223
Using Formula Fill to Transform Characters.....	225
Date Calculations with Formula Fill.....	227
The SEQ Function.....	228
Filling Empty Cells.....	230
Automatic Numbering.....	231
Propagate.....	231
UnPropagate.....	233
Using UnPropagate to Eliminate Duplicates.....	234
Change (Find and Replace).....	236
Changing with the Replace( Function.....	237
<b>Introduction to Forms.....</b>	<b>239</b>
Opening a Form.....	240
Opening A Form in a New Window.....	241
Form Modes: Data Access vs. Graphic Design.....	242



Form Operation: Individual Pages vs.View-As-List.....	243
Creating a New Form.....	243
Renaming a Form.....	244
Deleting a Form.....	244
Browsing the Database With a Form.....	244
Browsing the Database With a View-As-List Form.....	245
<b>Graphic Design.....</b>	<b>247</b>
Graphic Objects.....	247
Types of Graphic Objects.....	247
Creating a Graphic Object.....	250
Creating Perfect Squares, Circles and Lines.....	252
Customizing the Tool Palette.....	252
Using the Keyboard to Select Common Tools.....	255
SuperObjects.....	256
Modifying Objects.....	256
Selecting a Single Object.....	256
Selecting Multiple Objects at Once.....	257
The Graphic Control Strip.....	259
Rulers.....	259
Moving a Single Object.....	260
Nudging an Object (or Objects).....	261
Nudge “Auto Guides”.....	261
Viewing and Setting Exact Object Dimensions.....	262
Changing the Size of a Single Object.....	263
Nudging the Size of an Object.....	264
Nudge Size “Auto Guides”.....	264
Percentage Scaling.....	264
Resizing Without Handles.....	265
Removing Objects.....	266
Modifying Object Attributes.....	266
Solid, Outline and Hollow Objects.....	266
Line Width.....	266
Color.....	267
Copying and Pasting Colors.....	267
Font.....	268
Text Size.....	269
Text Style.....	270
The Object Properties Dialog.....	270
Working With Multiple Objects.....	271
Grouping Objects Together.....	271
Moving Multiple Objects.....	271
Resizing Multiple Objects.....	272
Cluster Resize.....	272
Aligning Objects.....	274
Adjusting Spacing Between Multiple Objects.....	275
Duplicating Objects.....	276
Duplicate.....	276
Drag Duplicating.....	276
Step and Repeat.....	277
Cut, Copy, and Paste.....	278
Copying an Entire Form.....	278
Overlapping Objects.....	279

Changing the Stacking Order .....	279
Selecting a Completely Hidden Object .....	279
Locked Objects .....	280
Ignoring Locked Objects .....	281
Alignment Grid .....	281
Form Background Colors .....	282
<b>Displaying and Editing Text .....</b>	<b>283</b>
Displaying Text .....	283
Fixed Text Objects .....	283
Editing Fixed Text .....	285
Moving and Resizing Fixed Text Objects .....	285
Text Font, Size and Style .....	286
Text Alignment .....	286
Displaying Data in Auto-Wrap Text .....	287
Data Merge Pop-Up Menu .....	288
Using Data Merge to Create Address Labels .....	288
Displaying Formulas in Auto-Wrap Text .....	291
Text Display SuperObjects™ .....	292
Creating and Modifying Text Display SuperObjects .....	292
Text Display Options .....	294
Using Formulas to Display Text .....	296
Combining Multiple Text Items Into One .....	297
Creating a Smart Formula .....	298
Eliminating Unnecessary Punctuation and Blank Areas With the Sandwich Function .....	299
Combining Numbers with Text .....	300
Displaying Dates .....	302
Editing Text .....	303
Types of Data Editing Objects .....	303
Working with Data Cell Objects .....	305
Text Editor SuperObject .....	307
Creating and Modifying Text Editor SuperObjects .....	307
Text Editor Options .....	309
Automatically Creating Rows or Columns of Data Cells or Text Editor SuperObjects .....	312
Automatic Layout Options .....	313
Word Processor SuperObject .....	316
Using the Mini Correspondence Wizard .....	317
<b>Images &amp; Movies .....</b>	<b>319</b>
Fixed Images .....	319
Flash Art™ .....	320
Creating Super Flash Art Objects .....	321
Using Flash Art to Display a Fixed Image .....	325
Displaying Images in a Different Folder (Directory) .....	326
Displaying Non PICT Images (Enhanced Image Pack) .....	329
Flash Art Image Drag and Drop .....	331
The Image Drops Example Database .....	331
Adding Drag and Drop Images to a Database .....	333
DropImagesFromFinder Options .....	337
Super Flash Art™ Options .....	338
Formula .....	338
Default .....	338
Include Pictures on Disk .....	340

Border.....	340
Scroll Bars .....	341
Align .....	341
Displaying Movies in a Form.....	347
<b>Buttons &amp; Widgets .....</b>	<b>351</b>
Push Buttons.....	351
Super Object Push Button .....	351
Push Button Styles .....	353
Button Title .....	355
Title Positioning .....	355
Standard Font/Size.....	355
3D Title .....	355
Hide Title .....	355
Click/Release .....	356
Color Options .....	356
Data Buttons .....	358
Data Button SuperObjects™ .....	358
Creating a Group of Radio Buttons .....	361
Multiple Value Button Groups.....	365
Super Data Button Options.....	369
Data.....	369
Title.....	369
Value .....	370
Allow Multiple Values .....	370
Value Separator .....	370
"Radio" button.....	371
Procedure.....	371
Sample .....	371
Sticky Push Button SuperObjects™ .....	373
Pop-Up Menus .....	375
Pop-Up Menu SuperObjects™ .....	375
The Pop-Up Menu Formula .....	378
Dividing Lines in the Menu .....	379
Pop-Up Menu Options .....	380
Data.....	380
Menu Formula .....	380
Menu Type .....	381
Display Options .....	383
Procedure.....	383
Pop-Up Menu Font, Size and Dimensions .....	383
List SuperObjects.....	383
Creating List SuperObjects™ .....	384
List Options.....	386
Data.....	387
Sep .....	388
Database .....	391
Sort Up .....	393
No Duplicates .....	394
Formula .....	394
Grow Box.....	394
Thin Scroll Bars .....	394
Procedure.....	395

Click/Release .....	395
Building the List .....	395
Maximum List Size .....	397
<b>Form Goodies .....</b>	<b>399</b>
View-As-List Forms.....	399
How View-As-List Forms Work.....	400
Creating a View-As-List Form.....	402
Working with Tiles .....	405
Adding a View-As-List Header .....	406
Editable View-As-List Forms .....	410
View-As-List Background Colors .....	415
Buttons on a View-As-List Form.....	417
Elastic Forms .....	418
Theory of Elastic Forms.....	419
Building an Elastic Form.....	420
Defining the Quadrants .....	420
Maximum Window Size .....	425
Removing the Window's Scroll Bars .....	425
Modifying an Elastic Form .....	426
Expanding Multiple Objects Proportionally .....	427
Elastic View-As-List Forms.....	428
Super Matrix Objects .....	430
<b>Charts.....</b>	<b>431</b>
<b>Printing Basics .....</b>	<b>433</b>
Printing Different Views.....	433
Printing the Data Sheet.....	433
Printing Data Sheet Headers & Footers .....	434
Printing a Form .....	437
Preparing Data For Printing .....	438
The Page Setup Dialog.....	439
The Print Dialog .....	440
Print Preview.....	441
Print One Record .....	443
Setting up Default Printers .....	443
<b>Custom Reports .....</b>	<b>445</b>
Working with Tiles.....	446
Creating Additional Tiles.....	450
Creating A New Tile By Duplicating.....	452
Tiles In Action .....	454
Data Tiles.....	455
Margins .....	462
Top Margin Tile .....	462
Left Margin Tile .....	464
Bottom Margin .....	466
Headers and Footers .....	466
Header Tile .....	467
Creating a Header Tile by Duplicating the Data Tile .....	468
Footer Tile.....	471



Page Numbers.....	472
Printing the Current Date and Time.....	473
First Page Header Tile.....	475
The QuickReport Dialog.....	477
Printing Multiple Column Reports.....	481
Controlling the Number of Columns.....	482
Printing Summary Information.....	483
Summary Tiles.....	485
Printing Data Grouped by Month, Quarter or Year.....	487
Group Headers.....	490
Keeping a Group Together on a Column or Page.....	493
Starting a Group on a New Column or Page.....	493
<b>Labels.....</b>	<b>495</b>
Label Fundamentals.....	495
The QuickLabel Dialog.....	495
Printing Labels on Sheets.....	498
Printing 3 by 10 1" Labels (Avery 5160).....	499
Aligning Labels on the Sheet.....	499
Printer Inaccuracy and Vertical Creep.....	499
<b>Formulas.....</b>	<b>501</b>
Formulas In Action.....	501
Displaying/Printing A Formula.....	502
Storing Formula Results in the Database.....	503
Using a Formula to Locate/Select Information.....	505
Formulas in Procedures.....	507
Using the Formula Wizard.....	508
Calculations with Database Fields.....	509
Changing the Active Database.....	511
The Programming Reference Wizard.....	512
Formula Components.....	513
Formula Grammar.....	513
Calculation Order and Parentheses.....	514
Functions.....	514
Multi-Parameter Functions.....	515
Zero Parameter Functions.....	515
Whitespace.....	516
Grammar Errors.....	517
Values.....	519
Constants.....	519
Build in Constants: Pi, Carriage Return and Tab.....	520
Fields.....	520
Using the Current Field.....	521
Variables.....	521
Variable Names.....	522
What's Inside A Variable?.....	522
The Life Cycle of a Variable.....	522
Creating Variables in a Procedure.....	523
Initializing Variables.....	524
Variables and Data Types.....	524
SuperObject Variables.....	524
Variable Name Conflicts.....	524

Permanent Variable Tips .....	525
Special Characters.....	525
Arithmetic Formulas .....	526
Dividing by Zero.....	527
Overflow/Underflow Problems .....	527
Basic Numeric Functions.....	528
Scientific Functions.....	529
Financial Functions.....	530
Text Formulas .....	531
Gluing Strings Together.....	531
Functions for Taking Strings Apart .....	532
Taking Strings Apart (Text Funnels).....	533
Numeric Start and End Positions .....	533
Specifying Numeric Length Instead of Position.....	534
String Testing Functions .....	534
String Modification Functions.....	535
Converting Between Numbers and Strings.....	538
ASCII Character Constant Functions .....	538
Text Arrays .....	539
Picking a Separator Character .....	539
Working With Arrays.....	540
Date Arithmetic .....	544
Today's Date.....	544
Converting Between Dates and Text.....	545
Date Functions.....	546
Time Arithmetic .....	548
Converting Between Times and Text.....	548
Time Calculations .....	549
Time Calculations with Text .....	551
True/False Formulas .....	552
Comparison Operators .....	552
A soundslike B.....	553
A match B.....	553
A matchexact B .....	554
A notmatch B.....	554
A notmatchexact B .....	554
Combining Comparisons .....	554
A and B.....	554
A or B .....	555
A xor B.....	555
not A .....	555
Equals Comparison vs. Assignment.....	555
True/False Values.....	556
The ? Function.....	557
Converting a Boolean Value to Text.....	557
Linking With Another Database .....	558
The Lookup Wizard.....	559
Type Mismatch Problems .....	562
Lookup Variations.....	563
Looking Up Rates in a Rate Table.....	564
Looking Up Multiple Fields From One Record.....	564
The GrabData Function .....	564
Looking Up Data in the Current File .....	565

Zip Code Lookup.....	565
US Post Office Abbreviation Functions .....	566
Import/Export Functions.....	566
System and Database Information Functions .....	567
System Information.....	567
Database Information .....	569
Custom Functions.....	570
<b>Procedures.....</b>	<b>571</b>
Programming Isn't Magic! .....	571
Introduction to (Panorama) Programming.....	571
Procedures .....	572
Statements.....	572
A Simple Procedure in Action .....	573
Creating a Procedure with the Recorder .....	579
Recording Mouse Clicks.....	582
Non Recordable Menus and Tools .....	582
Recording Data Entry .....	583
Writing a Procedure from Scratch.....	583
Writing Statements .....	585
Trying Out a Procedure .....	586
Checking for Mistakes .....	588
Mysterious Errors .....	589
Closing the Window When a Procedure is Finished .....	589
Re-Opening a Procedure .....	589
Font and Size .....	589
Adding a Recording to an Existing Procedure.....	590
Programming Reference Wizard.....	591
Navigation Using the Search Panel and Topic List.....	591
The Full Text Search Option.....	593
Navigation Using HyperLinks.....	593
Built In vs. Custom Statements and Functions .....	594
Using the Template Panel .....	594
Minimizing the Programming Reference Wizard .....	595
Data Flow.....	596
Assignment Statements.....	596
Variables.....	597
Creating a Variable.....	597
Assigning a Value to a Variable .....	598
Using a Variable in a Formula .....	598
The Birth and Death of a Local Variable .....	599
Long Life Variables.....	599
Variable Accessibility.....	599
Creating Variables with a SuperObject .....	600
Control Flow.....	601
True/False Formulas.....	601
Equals Comparison vs. Assignment.....	602
True/False Values .....	602
IF Statements .....	603
ELSE Statements .....	603
Nested if Statements .....	603
Error Handling with if error .....	604
CASE Statements.....	604

LOOP Statements.....	605
Stopping a Loop in the Middle.....	606
Restarting a Loop in the Middle.....	606
Subroutines.....	606
CALL Statement.....	607
Calling Procedures With Unusual Names.....	608
Passing Values to and from a Subroutine (Parameters) and Results.....	608
Terminating a Subroutine in the Middle.....	608
Mini Subroutines within a Procedure.....	609
Subroutines and Local Variables.....	610
Other Control Flow Statements.....	611
Jumping to an Another Location in the Program.....	611
Stopping the Program.....	612
Doing Nothing for a While.....	612
Program Formatting.....	613
Notes To Yourself.....	616
“Commenting Out” Statements.....	616
Suppressing Display of Text and Graphics.....	617
Updating the Display After (or Within) a NoShow Block.....	617
ShowPage.....	618
ShowLine.....	618
ShowFields field,field,...,field.....	618
ShowColumns field,field,...,field.....	619
ShowVariables var,var,...,var.....	619
ShowRecordCounter.....	619
ShowOther field,code.....	620
Checking NoShow Status.....	620
Disabling the Watch Cursor.....	620
Debugging a Procedure.....	622
The Panorama Interactive Debugger.....	622
The Debug Statement.....	622
Using the Debugger.....	622
Single Stepping.....	623
Resuming Full Speed Execution.....	625
Making Corrections to a Procedure.....	625
Watching Computations.....	625
Error Detail Wizard.....	628
Using the Error Detail Wizard.....	629
Finding the Source of the Error.....	630
Open Reference Wizard.....	632
Copy to Clipboard.....	632
Error Detail Problems.....	632
Using the View Wizard with Procedures.....	633
Searching All Procedures.....	633
50 Ways to Trigger a Procedure.....	636
The Action Menu.....	636
Action Menu Options.....	637
Setting Different Menu Item Styles (Bold, Italic, etc.).....	637
Shortcuts/Command Key Equivalents.....	637
Disabled Menu Items.....	638
Separator Lines in a Menu.....	638
Renaming the Action Menu.....	640
Dividing the Action Menu into Multiple Menus.....	641



“Unlisted” Procedures.....	642
Live Menus.....	643
Buttons.....	643
Hidden Triggers .....	644



# Welcome to Panorama!

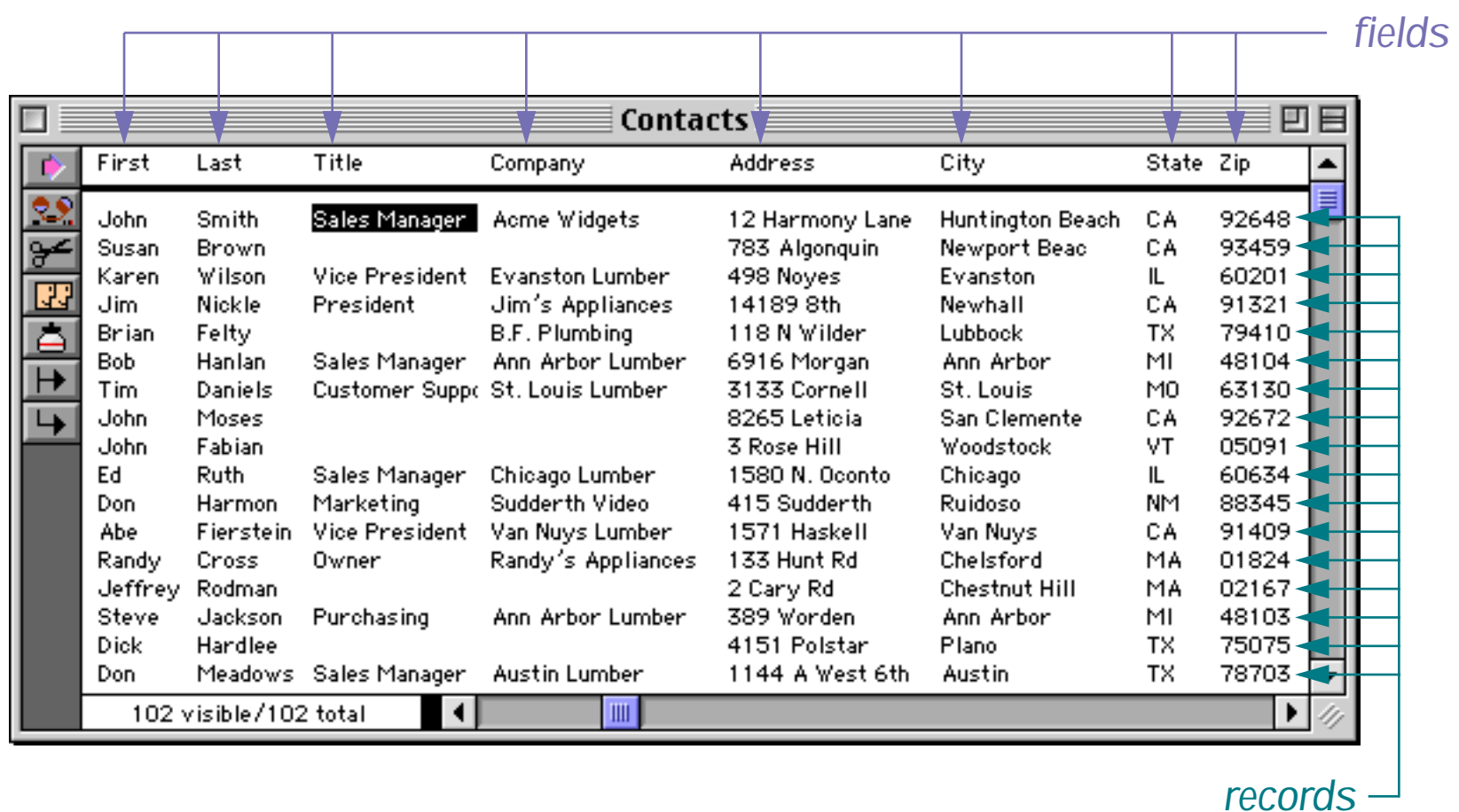


Congratulations! You are about to get acquainted with Panorama, a powerful tool for organizing and understanding information. With Panorama you can store, retrieve, categorize, summarize, chart, merge and print your information.

This book explains how to use Panorama. It assumes that you are already familiar with the basics of operating your computer. You should be familiar with pointing, clicking and dragging with the mouse, copying files, choosing commands from pull down menus, using scroll bars and editing text. If you are not familiar with these topics please review the training material that came with your computer.

## What is a Database?

The right knowledge at the right time can advance a career, a company, a civilization. Obtaining the right information at the right time is not often an easy task, especially when you are confronted with large amounts of data. Data that is arranged randomly (for example a pile of receipts) won't do you much good. When a collection of data is organized it is called a **database**. A computer program for entering and manipulating the information in a database is called a database program or a database management program.



First	Last	Title	Company	Address	City	State	Zip
John	Smith	Sales Manager	Acme Widgets	12 Harmony Lane	Huntington Beach	CA	92648
Susan	Brown			783 Algonquin	Newport Beac	CA	93459
Karen	Wilson	Vice President	Evanston Lumber	498 Noyes	Evanston	IL	60201
Jim	Nickle	President	Jim's Appliances	14189 8th	Newhall	CA	91321
Brian	Felty		B.F. Plumbing	118 N Wilder	Lubbock	TX	79410
Bob	Hanlan	Sales Manager	Ann Arbor Lumber	6916 Morgan	Ann Arbor	MI	48104
Tim	Daniels	Customer Supp	St. Louis Lumber	3133 Cornell	St. Louis	MO	63130
John	Moses			8265 Leticia	San Clemente	CA	92672
John	Fabian			3 Rose Hill	Woodstock	VT	05091
Ed	Ruth	Sales Manager	Chicago Lumber	1580 N. Oconto	Chicago	IL	60634
Don	Harmon	Marketing	Sudderth Video	415 Sudderth	Ruidoso	NM	88345
Abe	Fierstein	Vice President	Van Nuys Lumber	1571 Haskell	Van Nuys	CA	91409
Randy	Cross	Owner	Randy's Appliances	133 Hunt Rd	Chelsford	MA	01824
Jeffrey	Rodman			2 Cary Rd	Chestnut Hill	MA	02167
Steve	Jackson	Purchasing	Ann Arbor Lumber	389 Worden	Ann Arbor	MI	48103
Dick	Hardlee			4151 Polstar	Plano	TX	75075
Don	Meadows	Sales Manager	Austin Lumber	1144 A West 6th	Austin	TX	78703

Every database— whether on paper or in a computer — is composed of records. A **record** contains a collection of information about a particular person, company or entity. For example, each record in a mailing list database would contain the name and address of a particular person, while each record in an invoice database would contain all of the information collected for a single order.

Each record is divided into fields. A **field** contains a single piece of information about the subject of the record, for example a name, a street address, a phone number, etc. Fields are what make a database more than a hodgepodge of random information. Each field appears in the same place within every record in the database.

A database program like Panorama helps you design, manage and use data that is structured into records and fields. Before you begin you must tell the database program how you want the fields to be set up. As you enter new data Panorama helps make sure that everything goes in the right place and the structure remains intact. Once data is entered, Panorama can quickly scan the data to find the information you need, or categorize and summarize the information you need for a report. Panorama can also re-organize the data (for example, sorting) or even change the database field structure as your needs change — without having to start over from scratch. Virtually any job that can be done manually with a filing cabinet, card file or paper list can be done faster and more efficiently with a computerized database program like Panorama.

### A Brief History of Database Technology

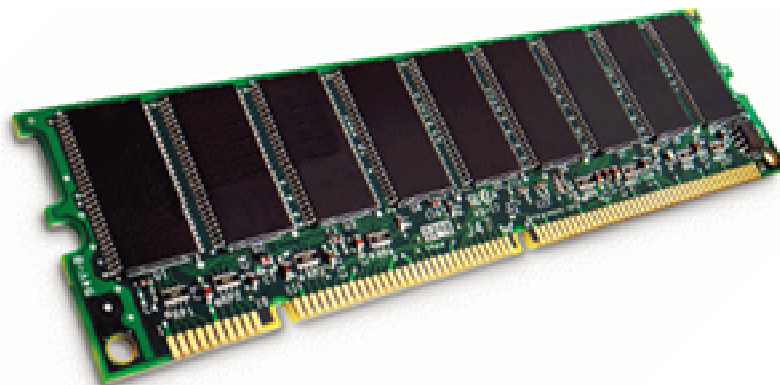
The need for data organization goes back long before computers. Before computers data was usually organized with paper forms and filing cabinets. Unfortunately, the information in a row of filing cabinets is hardly at your fingertips — at best an item might be located in a minute or two, while at worst a misfiled item might never be found at all.



In the 1960's and 1970's disk based database systems revolutionized the collection and storage of data. Instead of storing the data in a filing cabinet it is stored on a spinning magnetic disk (hard drive). Depending on how the information is filed an item can be located in as little as a second. Today's popular database programs, including Access and FileMaker, are based on this hard drive technology that was originally developed 30 to 40 years ago.



Storing information on a spinning disk is a fantastic improvement over filing cabinets, but it still relies on a mechanical system. Although disk drives have become faster and faster over the years, the speed of rotation and the movement of the head over the disk platter are still a bottleneck, just as feet, hands and fingers were a bottleneck when accessing data in a filing cabinet. Fortunately, there is an alternative. Besides the hard drive, your computer contains a large internal electronic memory bank (RAM) that allows the computer to work with large quantities of information at pure electronic speeds. In the past this electronic memory was too small for large databases, but today's computers have enough RAM for all but the largest database tasks.

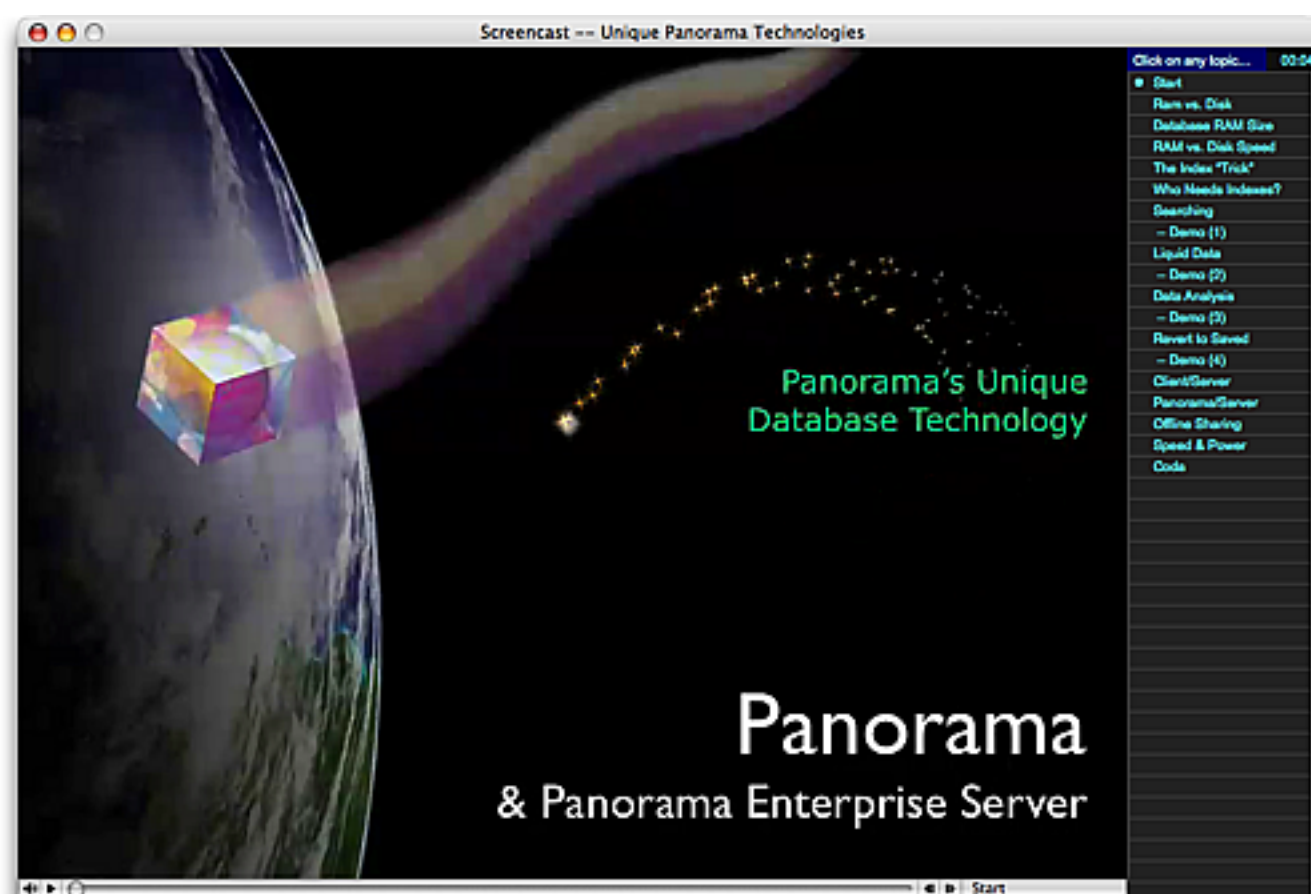


As you have probably guessed by now, Panorama is designed to leapfrog existing disk based database software by using your computers electronic memory for ultra fast operation, making Panorama faster, easier to use, and more powerful than any other database software previously available. Panorama uses the hard drive only for permanent data storage. Searching, sorting, summarizing and other data processing tasks are performed entirely in RAM.

## What is Panorama?

In addition to its blazing RAM based speed, Panorama includes a number of unique capabilities that set it apart from other database programs. Panorama is easy to learn and use, is blazingly fast, has powerful tools for analyzing financial data, and is powerful enough for even the most demanding database jobs.

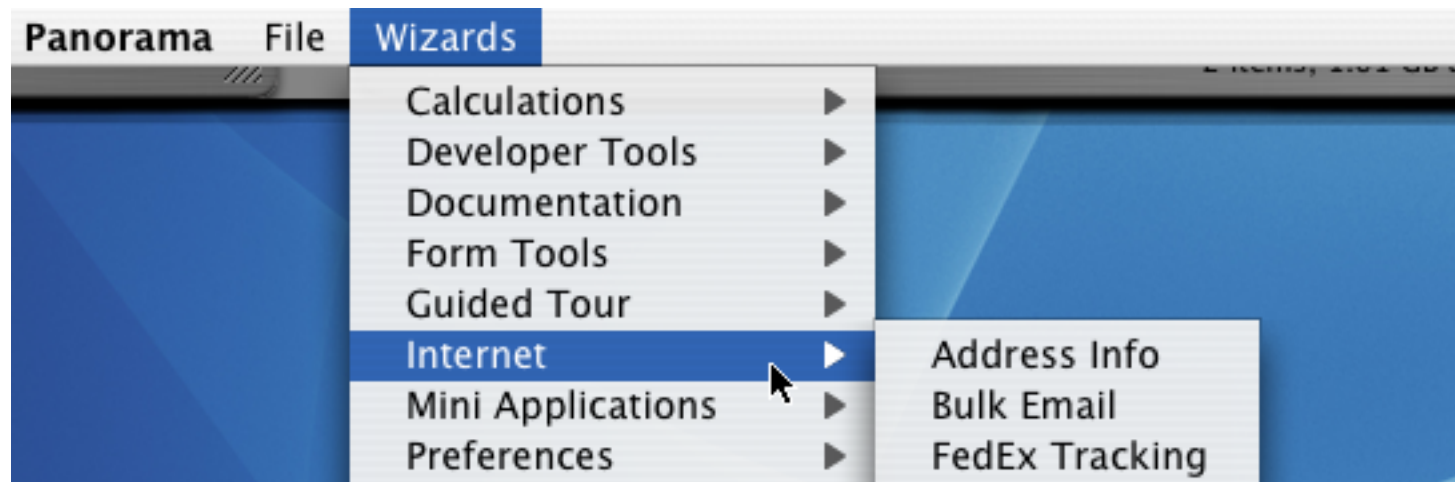
If you are just getting acquainted with Panorama, be sure to check out the guided tour screencasts (movies) included on the CD (or if you downloaded Panorama you can watch the screencasts directly from the [www.provue.com](http://www.provue.com) web site). Just sit back and relax while we show you Panorama's unique tools for organizing information. All of the movies are recorded digitally and allow you to pause, back up, or skip ahead to the topics that most interest you.





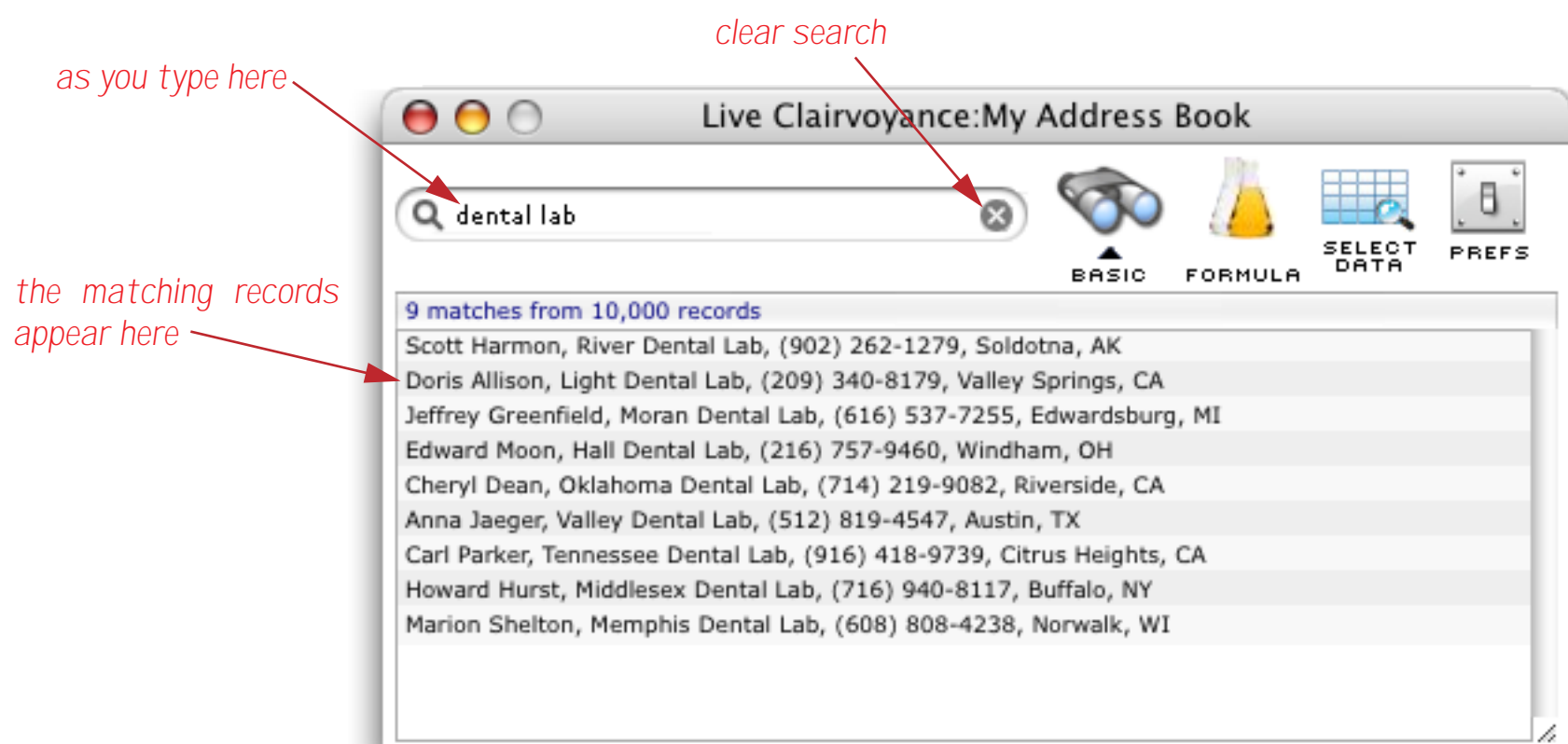
## Wizards

Panorama's **Wizard** menu contains pre-built databases for automating common tasks and enhancing productivity when using Panorama (see the separate **Wizards & Demos** PDF file). General productivity wizards include databases for organizing your contacts, calendar, correspondence and tracking your time. Panorama also includes wizards for importing and exporting data, arranging windows, locating favorite files, creating new databases and much more.



## Super Fast Searching and Sorting

Panorama's RAM based speed makes searching and sorting many times faster and more flexible than any disk based program. Searching is not limited to full word or begins with matches — you can search for data that contains a word or phrase or even perform formula based searches (for example “find all names longer than 12 characters” or “find all invoices where shipping is more than 10% of the order total”). The **Live Clairvoyance™** feature allows you to perform “live” searches on any Panorama database. The search results are updated dynamically as you type, allowing you to “hone in” on just the information you are looking for.

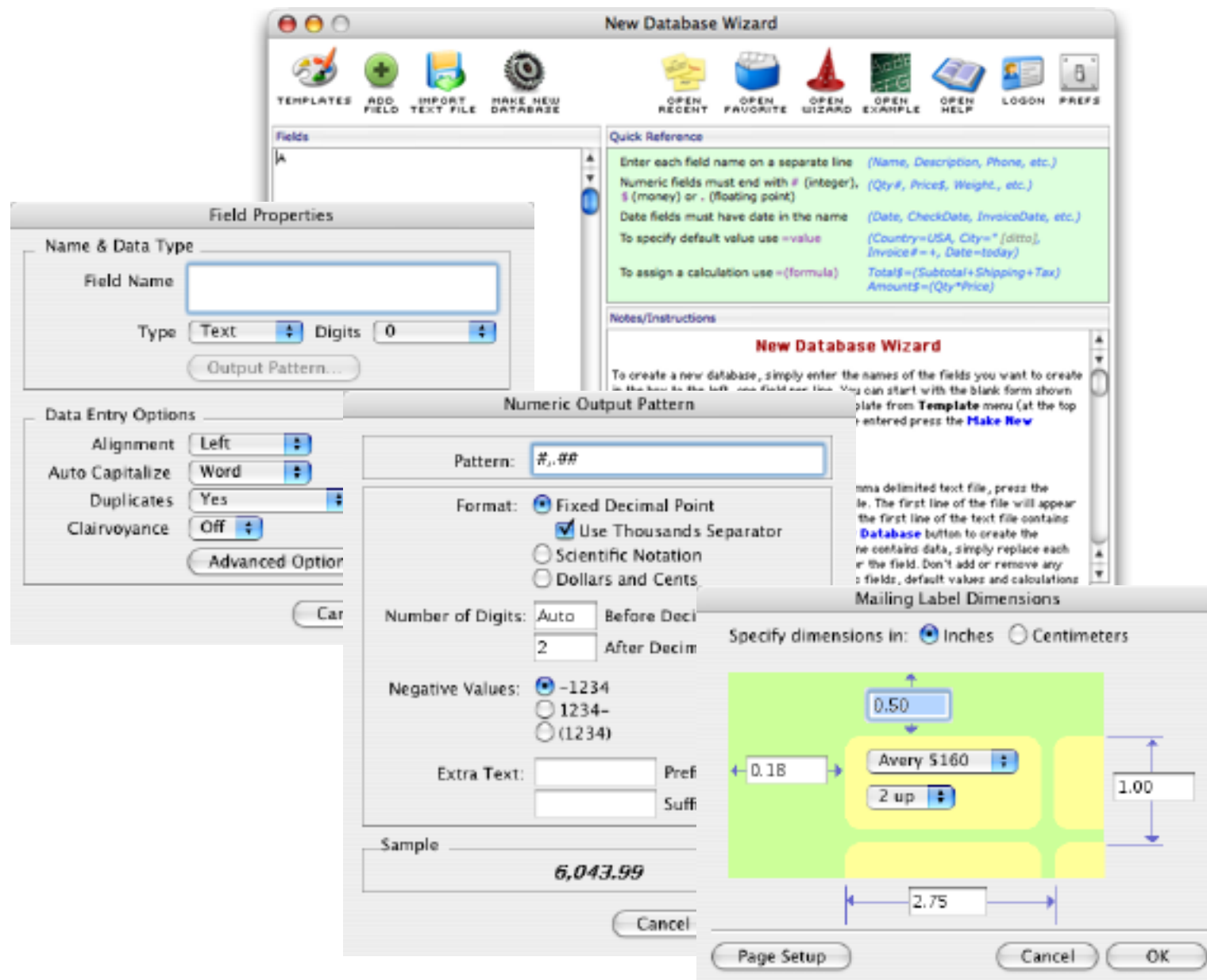


## Phonetic Searching

Panorama's “sounds like” option allows you to search for data phonetically. For example a search for “sounds like Alan” will turn up anyone named **Alan**, **Allan**, or **Allen**.

## Easy Set-Up

Simple step-by-step dialogs make it easy to define database fields, print mailing labels, and print custom reports.



## Crosstabs

Panorama's crosstab feature allows it to quickly and automatically convert raw data into a tabular summary; for instance turning raw checkbook data into a monthly budget. Crosstabs are one of the most powerful tools yet for analyzing financial data.

Corporate Checkbook:XTABS:Spending by Quarter					
xtab	1q98	2q98	3q98	4q98	TOTAL
Advertising	8,592.43	7,681.09	5,185.44	8,552.43	30,011.39
DEPOSIT	0.00	0.00	0.00	0.00	0.00
Fixed Assets	3,338.75	3,555.61	2,064.40	1,063.90	10,022.66
Insurance	3,763.50	3,763.50	3,763.50	3,763.50	15,054.00
Legal		282.44	1,186.08	1,425.11	2,893.63
Office Supplies	3,495.79	1,655.83	1,577.07	2,556.82	9,285.51
Payroll	23,408.67	22,866.11	22,440.16	23,504.18	92,219.12
Purchases	11,289.42	7,409.52	6,013.66	8,813.83	33,526.43
Rent	4,740.00	4,740.00	4,740.00	4,740.00	18,960.00
Shipping	2,067.84	3,428.15	2,890.98	2,746.63	11,133.60
Telecom	1,392.75	1,272.40	1,322.30	1,362.93	5,350.38
Utilities	694.84	684.57	610.40	600.91	2,590.72
+TOTAL	62,783.99	57,339.22	51,793.99	59,130.24	231,047.44

### Data Outlines

Panorama’s outlining feature is another powerful tool for analyzing financial data, and goes much further than ordinary subtotal calculations. You can hide and reveal different levels of subtotals, identify problems or opportunities and then zoom in on specific details. You can also perform further operations or calculations on subtotals as if they were data.

			<b>Rent</b>	<b>35,026.34</b>
02/09/99	1962	Airborne Express	Shipping	35.40
		<b>Airborne Express</b>		<b>35.40</b>
03/26/99	2018	AIRS	Shipping	138.07
		<b>AIRS</b>		<b>138.07</b>
02/09/99	1971	American Customs House	Shipping	34.00
		<b>American Customs H</b>		<b>34.00</b>
07/16/99	2191	B J D Trucking Inc.	Shipping	37.50
		<b>B J D Trucking Inc.</b>		<b>37.50</b>
09/19/99	2277	Burlington Air Express	Shipping	95.17
		<b>Burlington Air Expre</b>		<b>95.17</b>
03/16/99	2005	Consolidated Freight	Shipping	150.20
		<b>Consolidated Freight</b>		<b>150.20</b>
01/30/99	1929	Federal Express	Shipping	178.75
03/20/99	2015	Federal Express	Shipping	170.00
03/28/99	2032	Federal Express	Shipping	150.00
05/14/99	2101	Federal Express	Shipping	170.00
		<b>Federal Express</b>		<b>668.75</b>
01/31/99	1930	U P S	Shipping	52.97
02/09/99	1946	U P S	Shipping	122.60
03/20/99	2014	U P S	Shipping	25.00
03/28/99	2034	U P S	Shipping	197.42
04/17/99	2056	U P S	Shipping	25.00
05/04/99	2079	U P S	Shipping	25.00
05/08/99	2088	U P S	Shipping	25.00
07/24/99	2207	U P S	Shipping	45.80
08/21/99	2250	U P S	Shipping	155.00
09/19/99	2281	U P S	Shipping	95.32
		<b>U P S</b>		<b>769.11</b>
			<b>Shipping</b>	<b>1,928.20</b>
			<b>Taxes</b>	<b>5,152.79</b>
			<b>Telephone</b>	<b>5,690.50</b>
			<b>Utilities</b>	<b>2,054.89</b>
				<b>183,651.22</b>

### Data Entry Shortcuts

Panorama’s data entry shortcuts reduce keying errors and data entry errors. Panorama’s unique Clairvoyance® feature automatically completes data entry for you. Auto-capitalization, data entry buttons, smart defaults, and optional spelling checker and zip code lookup save even more time.

Contacts:Person

Name John Smith

Title Sales Manager

Company Acme Widgets

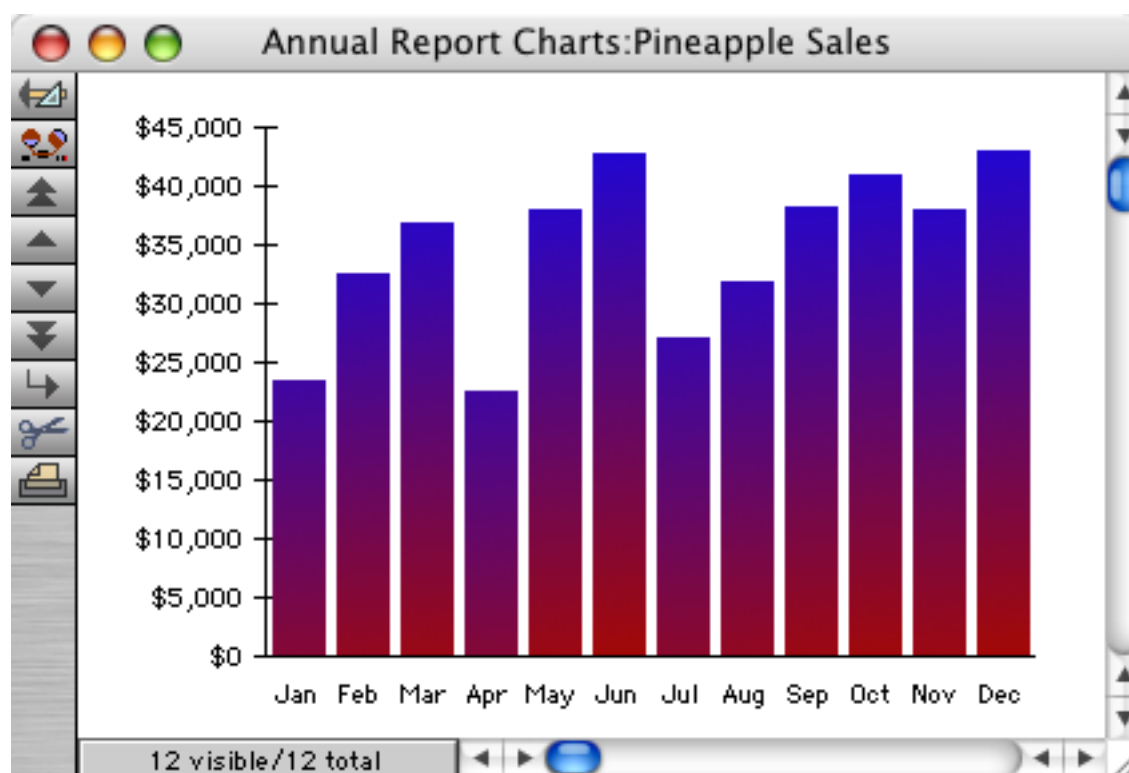
## Smart Dates™

Panorama understands dates the way you do—as part of weeks, months, quarters, or years. You can easily locate or summarize information by any of these date periods.

12/21/98	559	Fry's Electronics	Office Supplies		1,189.22	13,684.48
12/28/98	560	Valley Publication:	Advertising	Invoice 2680	963.57	17,985.81
last friday		Payroll Serv	Payroll	Payroll period fr	1,749.38	16,236.43

## Charts

Panorama's built in charts (bar, line, area, pie, and scatter) can visually reveal trends and relationships that are often hidden in a conventional report (see "[Charts](#)" on page 971).



## Relational Links

Panorama can relate the information in two or more databases, insuring consistency and simplifying complex tasks like order entry, billing, payroll, sales lead tracking, and more.

### Vendors:Detail

Organization: Valley Publications

Address: 48582 N.E. California Place  
Fontana, CA 92336

Phone: (714) 539-1824

E-Mail: mplant55@acadia.net

Ck #	Date	Amount	Memo
560	12/28/1998	963.57	Invoice 2680
546	12/14/1998	1,022.45	Invoice 2638
514	11/23/1998	1,081.34	Invoice 2542
478	10/19/1998	1,008.13	Invoice 2434

### Corporate Checkbook:Check

Date: 12/28/1998 Number: 560 Category: Advertising

Pay To: Valley Publications \$ 963.57

48582 N.E. California Place  
Fontana, CA 92336

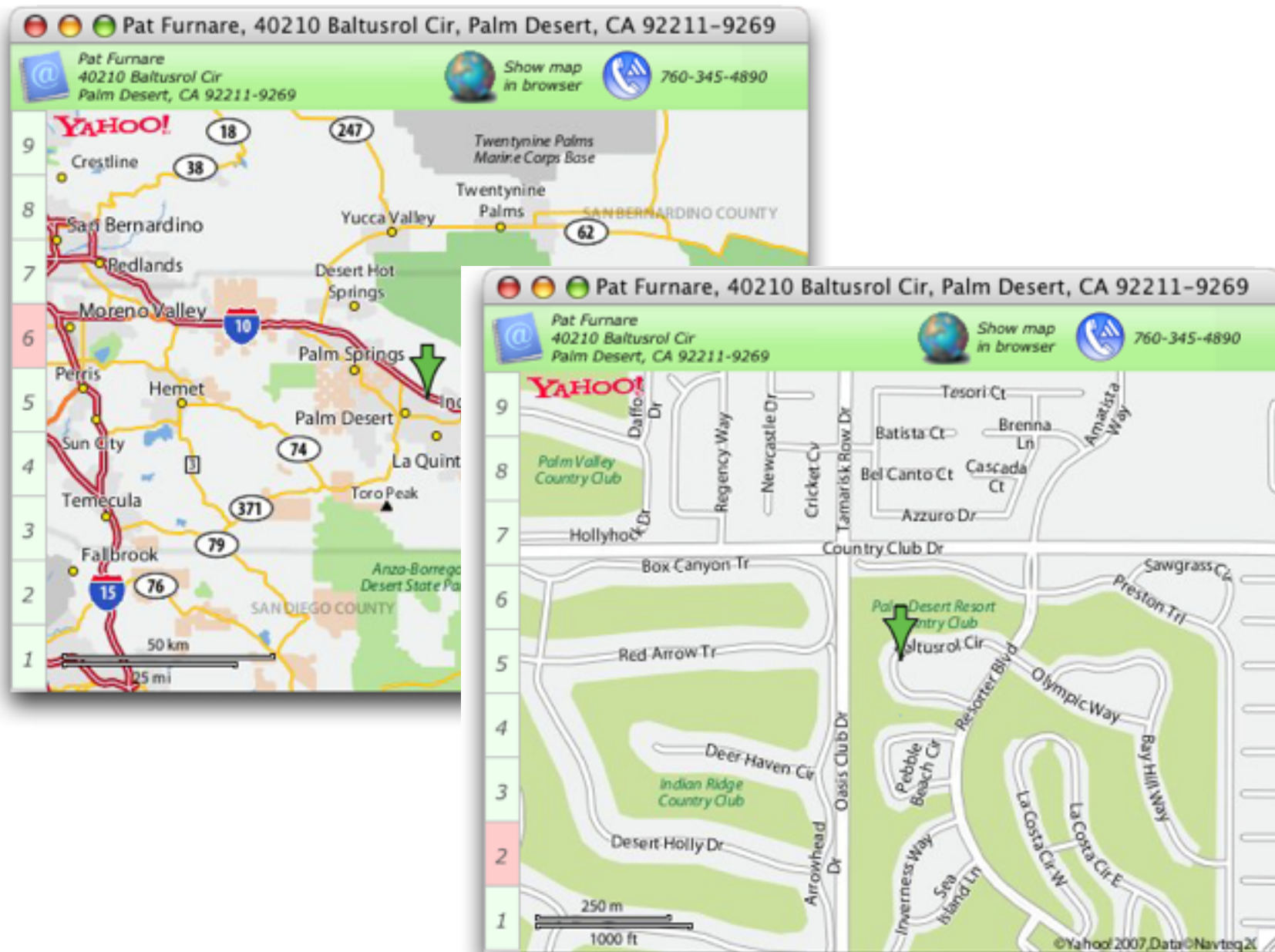
Nine hundred sixty three dollars and 57 cents

Ck #	Date	Amount	memo
560	12/28/1998	963.57	Invoice 2680
546	12/14/1998	1,022.45	
514	11/23/1998	1,081.34	
478	10/19/1998	1,008.13	
471	10/12/1998	876.14	



**Internet Content**

Panorama includes built-in access to web content, including maps, phone and zip code information, fedex tracking and more. If data is on the web, you can access it with Panorama.



Panorama can send e-mail to individual or bulk recipients. You can program e-mail transmission into your own database (for example setting up an invoice database to automatically e-mail a copy of each invoice) or use the Bulk Email wizard.

**Map, Phone, Email and VCard Support**

Based on information in your databases Panorama can automatically display maps, dial the phone, send e-mail and drag and drop between Panorama and any other VCard enabled application (like Address Book).





## Advanced Graphic Tools

Panorama's graphic capabilities build on standard Macintosh drawing features and add special tools and dialogs that are designed specifically for creating and modifying tables within forms and reports.

**HobbyShopSales:Aqua Order Form (100%)**

SEARCH DATA ADD DELETE PRINT VCARD DIA EMAIL SYNC

**Contact Information** (Jan 1, 1998) Invoice 1000

Name: Derrick Ramsey  
 Address: 35081 W. Birch Rd.  
 Walnut Creek CA 94596  
 Country: [Dropdown]  
 Phone: (925) 639-7244  
 Fax: [Field]  
 E-Mail: [Field]

**Payment Information**

Card Type: Visa Name On Card: [Field]  
 Card Number: 4062-9297-0143-2585 Expires (mm/yy): 07/03  Non-Taxable

**Super Matrix**

Qty	Description	Price	Total
-	1 C&NW 40' Stock Car	4.99	\$ 4.99
-	1 Chessie U30B	28.99	\$ 28.99
-	1 UP 40' Single Dome Tank Car	4.49	\$ 4.49
-	1 ATSF Coach	8.49	\$ 8.49

World's Greatest Hobby  
 Subtotal: \$ 146.94  
 Tax: \$ 11.39  
 Subtotal: **\$ 158.33**

## Mail Merge and Mailing Labels

Panorama's built-in mail merge can produce all the components of a direct mailing—including custom form letters, mailing labels, and postcards or envelopes. Panorama includes a built-in word processing program, reducing both the complexity and time required to put together a direct mailing.

## Database Reports

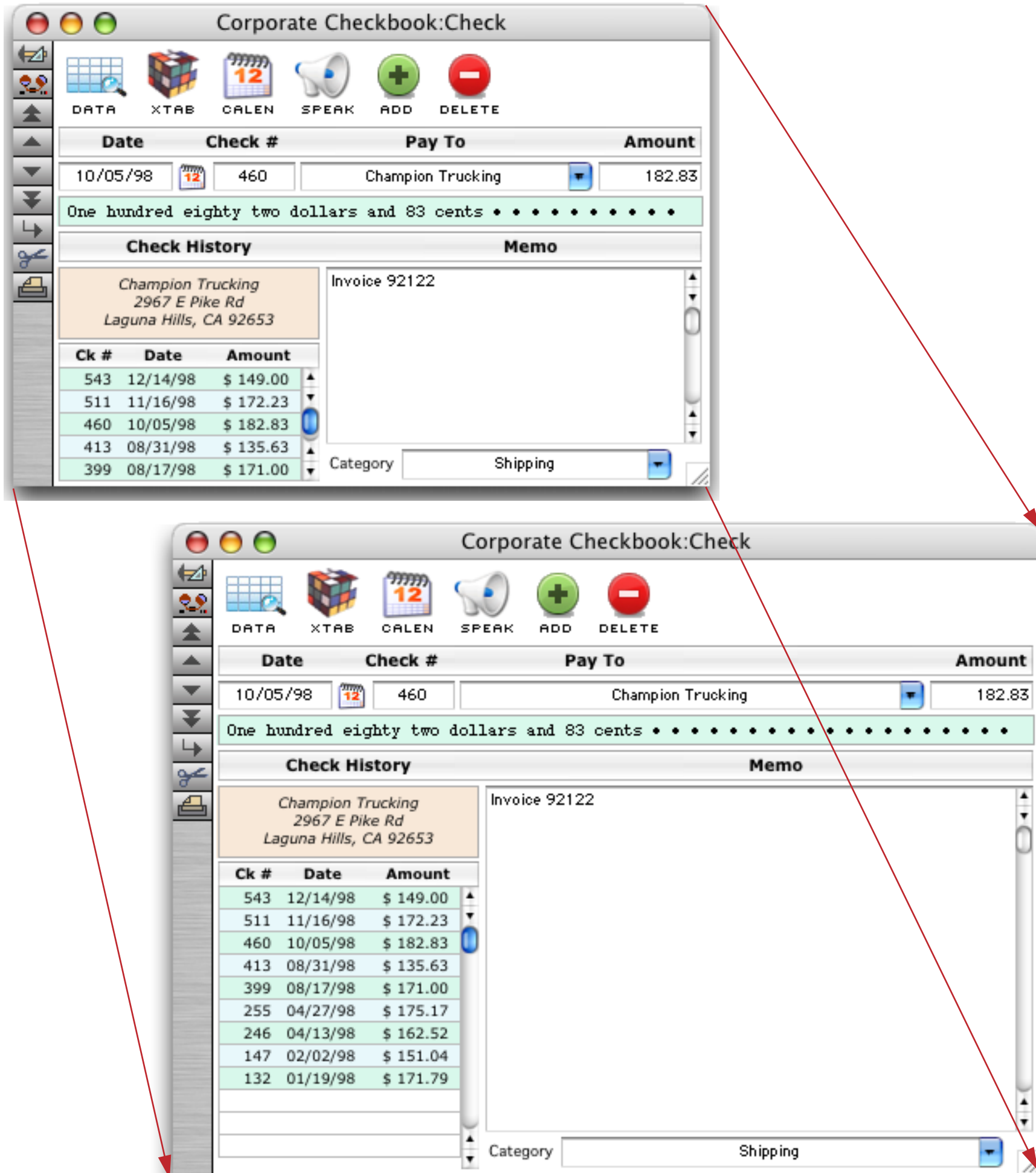
Panorama can automatically create complex database reports including catalogs, directories, bibliographies, and more. Reports can include both fixed and variable height elements (including images), and Panorama can automatically control page and column breaks to eliminate widows and orphan).

## View-As-List Forms

Panorama allows you to display forms as separate pages, or as a continuous scrolling sheet (we call this a "view-as-list" form).

### Elastic Forms

Elastic forms adjust intelligently when the window containing the form is resized or zoomed. When the form is designed, you decide how the individual elements will expand or shift as the form changes size.



## Matrix Display

Panorama matrix object makes it easy to create repeating grids including calendars, catalogs and invoices.

Sun	Mon	Tue	Wed	Thu	Fri	Sat
				1	2	3
				13 checks \$3,576.36		
4	5	6	7	8	9	10
	8 checks \$3,623.55					
11	12	13	14	15	16	17
	5 checks \$3,016.49					
18	19	20	21	22	23	24
	6 checks \$3,613.49					
25	26	27	28	29	30	31
	7 checks \$5,161.69					
						<b>Total</b>
						39 checks \$18,991.58

Ck #	Pay To	Amount
460	Champion Trucking	\$182.83
461	Cool Creek Studio	\$573.42
462	Airborne	\$87.84
463	Post Office	\$258.32
464	Precision Plastics	\$493.02
465	Kinko's	\$135.46
466	Poly Payroll Services	\$1,687.25
467	Clark Supply	\$205.41

## Images and Movies

A Panorama form can display images and movies from a wide variety of sources. An image may be fixed (for example a logo or background) or variable (changing from record to record - for example personnel photos or maps associated with individual records). Variable images may be included in the database or (more commonly) displayed directly from files on the disk. With the optional enhanced image pack Panorama can display nearly two dozen different image formats, including JPEG, TIFF, PNG, PCX and TARGA.

## High Speed Import

Panorama can import data at rates approaching 1000 records per second. Data can be imported from text files (tab or csv delimited) and from Excel spreadsheets..

## Flexible Text Export (including HTML)

Panorama can export text files in tab separated, comma separated or HTML table formats. When exporting HTML you can choose fonts, colors, column widths and titles.

## Data Transformation

Panorama can quickly transform large amounts of data. Examples include re-arranging characters or words, capitalizing, and transformations based on patterns in the data.

## Select/Remove Duplicates

Panorama can scan even the largest databases for duplicate information with the **Select Duplicates** command . Duplicates can also be removed automatically based on rules set up in advance.

## Compact Storage

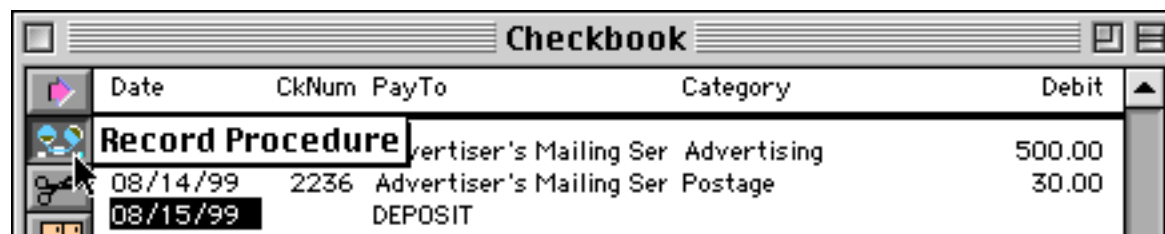
Panorama uses up to 85% less disk space than other database programs for the same data.

## Formulas

Panorama can perform simple and complex calculations on numbers, text, and dates, and includes a wizard to help you build and test formulas.

## Smart Program Recorder

Panorama's program recorder allows you to record multi-step operations and play them back later with a single click or keystroke. It's as easy to use as a cassette recorder—just start the recorder then do your work. The recorder doesn't just record mouse clicks and keystrokes as is, but automatically converts them into simple English-like commands that can be edited later if necessary.

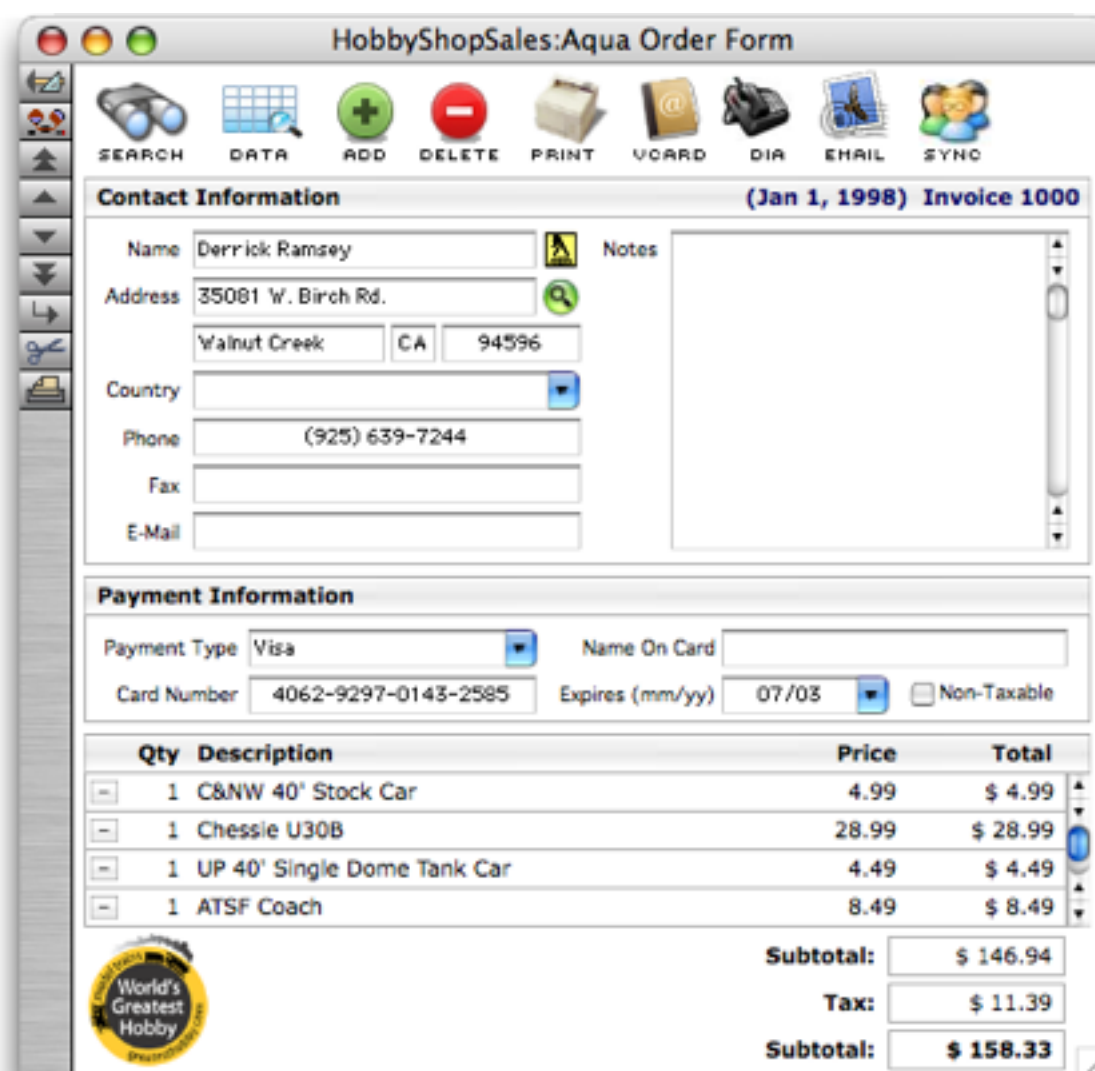


## Complete Programming Language and Development Tools

If Panorama doesn't have a feature already, you can add it yourself with Panorama's built in programming language. Panorama includes powerful programming features like variables (both local and global), if-then-else, case switching, loops, subroutines and a built in interactive debugger. You can even extend Panorama's programming language with your own custom statements and functions or embed other programming languages included AppleScript, Unix Shell Scripts, Perl, Ruby or Python.

## Custom Menus, Buttons, and Dialogs

Panorama gives you the capability to create your own custom menus, buttons and dialog, making professional quality custom applications possible.



## Seamless Cross Platform Operation

Panorama databases are 100% compatible with both the Windows or MacOS platforms and may be transferred back and forth freely between the two platforms.



## Panorama Enterprise Server

The **Panorama Enterprise Server** introduces a revolutionary new system for sharing database information. Unlike traditional client/server databases that sequester all of the data on a central server, the Panorama Enterprise system distributes complete copies of shared databases across all clients. Essentially each client caches the entire database for better performance, and lower network and server load. The Panorama Enterprise server acts like a traffic cop, managing record locking and updating clients as necessary.



**RAM Based Server.** The Panorama Enterprise Server is RAM based for blazing speed, but it also keeps a disk based transaction journal for full data recovery after any kind of system interruption (power failure, etc.). The journal is a simple sequential file with minimal impact on performance.

**RAM Based Client.** Each client keeps a full copy of each open database in RAM for fast searching, sorting, reports, etc. The server only gets involved when data is modified, managing record locking and updating clients as necessary. This increases performance two ways:

- Sorting, searching, summaries and reports are done locally in the client's RAM. You'll feel like your databases have broken the sound barrier!



- Network traffic and server load are dramatically reduced. Panorama clients don't access the network at all when browsing, sorting, etc.

**Offline Roaming.** Traditional client/server systems are completely inoperable without a network connection. Panorama gives you the option of roaming off-line while still retaining access to your databases for browsing, reports and even modification and updates. In other words, users can view and edit databases even when they don't have a network connection -- ideal for road warriors. Offline changes are automatically synchronized with the server when the client re-connects to the network. Because both client and server are RAM based this synchronization is extremely fast. If there are any synchronization conflicts (same record edited both offline and by another user) they can be resolved automatically or manually on a field-by-field basis.



**Web Database Publishing.** Panorama's WYSIWYG form design tools can automatically build HTML/CSS forms for database web access and upload them to the server. Many common web database applications can be built with no

programming at all. For fully customized applications, Panorama provides a comprehensive programming language that includes rich text and tag manipulation capabilities. Server side programs are automatically uploaded to the server and a built in simulator/debugger allows developers to debug programs on their development machine before they are uploaded.





**Managing Web Content.** Database changes made on any client (for example price list changes) immediately show up on the web server, while changes made via the web (for example incoming shopping cart orders) automatically show up on client desktops. No extra programming for database maintenance is required — authorized users work with Panorama's intuitive user interface to edit web published databases just as if they were on their local computers.

**“Cloned” Web Sites.** The Panorama Enterprise Server makes it easy to manage "cloned" web sites with identical design and programming. A typical example of this feature is managing identical test and production servers. When tests on a new feature are complete the live production server is automatically updated with the changes made on the test server (while keeping the data completely separate).



**Easy Server Administration.** All Panorama Server configuration and maintenance is managed remotely from the Panorama client software. Built in server backup automatically backs up "live" databases without shutting down the server. The server can be configured to automatically send an e-mail to the server administrator if any problems occur.

**Industrial Strength.** When publishing database content on the web the Panorama Server works as a back end to the built-in industrial strength Apache server included with OS X and OS X Server. No modifications to the Apache configuration files are required. (You can also use any other UNIX web server that supports Apache style CGI's.)



**Connect and Synchronize Mode.** Panorama Enterprise databases can be configured to operate primarily offline. This is similar to the way e-mail works - users perform data entry offline, then press "Submit" or "Connect" to submit their data and receive updates. The connection lasts only long enough to transfer the data. Before Panorama Enterprise, applications that worked like this had to be built from scratch, but the Panorama Enterprise server allows this type of operation with little or no custom programming.



**Built-in Backup.** The Panorama Enterprise Server's built-in backup system provides automatic daily backup of "live" databases without shutting down or interrupting the server.



**Remote Server Administration.** All server configuration and maintenance can be managed remotely from any Panorama client — once the server is installed you'll never need to touch it again.

This system is documented in the separate [Panorama Enterprise Handbook](#).

## Getting Started With Panorama For First Time Users

If you haven't done so already, start by installing Panorama on your system. As a newcomer to Panorama we recommend that you start with the step-by-step tutorials that walk you through the creation and use of four typical databases — a mailing list, a checkbook, an invoice and a price list.

You can either watch a movie of these tutorials and/or view them in PDF format . We've also included finished versions of these databases that you can use for learning and inspiration. As you build and use these databases you'll acquire the skills you'll need to develop custom database applications for your own business, organization, school, hobby or household.

As you become more involved with Panorama you may want to take advantage of some of the additional resources available, including technical support and various e-mail discussion groups. You may even want to attend one of the periodic conferences we hold here in Los Angeles.

## Upgrading to Panorama 5.5 From an Earlier Version

If you haven't done so already, start by installing and activating Panorama on your system. (By the way, you can leave your old version of Panorama on your hard drive if you want to — you can even use both the old and new versions at the same time (not with the same database at the same time, of course)!)

To help you quickly learn about what's has changed since the version of Panorama you are familiar with we have prepared a summary of the changes in each version, you'll find this in the [Release Notes.pdf](#) file. The summary has links to the actual topics within the body of the manual so that you can quickly focus on the new material important to you.

## Deploying Applications Built With Panorama

ProVUE has two options for affordable deployment of Panorama based applications — **Panorama Direct** and **Panorama Engine**. Panorama Direct is designed to be used as a "run-time" engine that allows the economical deployment of database applications created with the full version of Panorama. Panorama Direct prices start at \$130 and is available in economical 3, 6, 12, 25 and 50 packs for as little as \$90 per seat. The Panorama Engine allows unlimited royalty free distribution of a Panorama database for a one time fee.

### Panorama Direct

Panorama Direct is a limited, low cost version of Panorama that can run any database created with the full version of Panorama. Database files can be exchanged back and forth between full Panorama and Panorama Direct with no conversion. The Panorama Direct user can enter and modify data in the database, sort, select, print and run procedures. The user can even make minor adjustments to forms and reports. However the user will not be able to examine, create or modify procedures, create or modify SuperObjects within a form, modify cross tab set up, set up or modify charts, etc. (If desired, the database creator can lock down any database to prevent anyone from modifying forms, reports, procedures, etc., whether they are using Panorama Direct or a full copy of Panorama.)

Panorama Direct is ideal for distributing pre-built databases either in-house or around the world. Many companies save money by purchasing full copies of Panorama only for users that actually set up and design databases, while purchasing Panorama Direct for data entry positions. Panorama Direct is fully compatible with the Panorama SQL Server and may be used as a client with other copies of Panorama and/or Panorama Direct. In addition, a growing number of third-party developers use Panorama Direct to distribute commercial applications developed in Panorama. Panorama Direct makes it possible for developers to price their products aggressively while retaining all the benefits of rapid application development with Panorama.

### Panorama Engine

Panorama is normally sold on a "per-machine" basis. Once you have purchased a copy of Panorama or Panorama Direct, you are free to use it on an unlimited basis on a single machine. On that single machine you can create as many databases as you like, and change the design of any database any way you like at any time.

The **Panorama Engine** introduces a new way to purchase Panorama's outstanding capabilities. Instead of purchasing unlimited use of Panorama on a single machine, you can now purchase an unlimited "run-time" distribution license for a single database (or collection of databases). Once you have purchased this unlimited distribution license you can distribute as many copies of your database (or collection of databases) as you like, without having to purchase Panorama or Panorama Direct for each machine. The one-time license fee is very reasonable, and for shareware authors, almost zero!

### Panorama Engine vs. Panorama Direct

Panorama developers now have two ways to deploy a Panorama database: Panorama Direct or the Panorama Engine. How do you decide which to use for your project?

If you are planning to distribute your database to hundreds (or thousands) of users, then the unlimited distribution license with the Panorama Engine is the way to go. The Panorama Engine makes it possible to create low cost mass-market applications using Panorama technology.

If you are planning to distribute your database to a small group of users, Panorama Direct is a better approach. Panorama Direct is also more appropriate if you plan to change the design of your database frequently, or if your application requires features not included in the Panorama Engine (SQL, graphics mode, etc.).




# Chapter 1: Files and Memory



Panorama databases are permanently stored in files on a disk drive. (In fact, you'll often find that the words **database** and **file** are used interchangeably.) Each Panorama file contains all the components needed to use the database.

## Files, Icons and the Desktop

Before you begin to use Panorama you should be familiar with the basic operation of your computer. Whether you are using a Macintosh or a Windows based computer, files appear in a “desktop” environment that allows them to be located, moved, copied and opened. On the Macintosh this desktop environment is called the **Finder**. On Windows computers this is simply called the desktop, which you can view with **My Computer** or using the **Windows Explorer**. There are three different kinds of Panorama icons: **databases**, **file sets**, and the Panorama application itself.

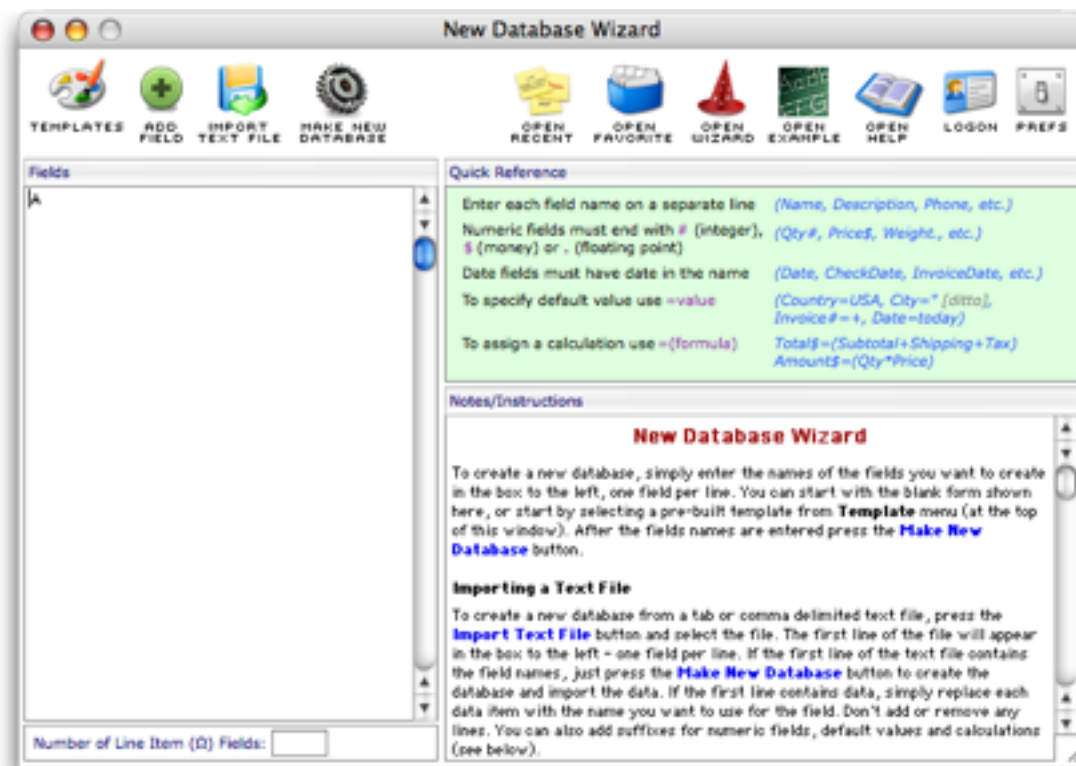
	This is the icon for a single Panorama database. On Windows machines, these files have the extension <b>.pan</b> . Double click on this icon to open the database.
	This is the icon for a set of Panorama databases (called a file set). On Windows machines, these files have the extension <b>.pnz</b> . Double click on this icon to open the entire set of databases at once.
	This is the icon for the Panorama application itself, which is usually called <b>Panorama</b> (Mac) or <b>Panorama.exe</b> (Windows). You can double click this application when you want to create a new database without opening an existing database first.

You can manipulate these icons on the desktop any way you like, just like any other files. (However, you should avoid moving or copying the Panorama application itself. Although your disk may contain many database files for different tasks, there should only be one copy of the Panorama program itself.)








## Launching Panorama



To launch Panorama you can either double click on the Panorama application itself or double click on the icon for any Panorama database or file set. If you double click on the Panorama application itself the New Database wizard will appear.



This wizard can be used to create new databases (see “[Using the New Database Wizard](#)” on page 38), but it can also be used as a handy “command post” for Panorama using the half dozen or so icons in the top right corner.

 OPEN RECENT	Click here to open the <b>Recent Databases</b> wizard. This wizard makes it easy to re-open databases that were recently opened. See “ <a href="#">The Recent Databases Wizard</a> ” on page 37 for more information.
 OPEN FAVORITE	Click here to open the <b>Favorite Databases</b> wizard. This wizard allows you to define your favorite databases and access them quickly. See “ <a href="#">The Favorite Databases Wizard</a> ” on page 37 for more information.
 OPEN WIZARD	Click here to open the <b>Open Wizard</b> wizard. This wizard allows you to open wizards via the keyboard instead of using the Wizard menu. See “ <a href="#">Open Wizard</a> ” on page 13 of the <b>Wizards &amp; Demos</b> PDF file for more information.
 OPEN EXAMPLE	Click here to open the <b>Example Launcher</b> wizard. This wizard gives you rapid access to the various example and demo files provided with Panorama. See “ <a href="#">Example Launcher</a> ” on page 20 of the <b>Wizards &amp; Demos</b> PDF file for more information.
 OPEN HELP	Click here to open the <b>Help &amp; Documentation</b> wizard. This wizard gives you rapid access to the various PDF files, on-line references and screencasts (movies) provided with Panorama. See “ <a href="#">Help &amp; Documentation</a> ” on page 40 of the <b>Wizards &amp; Demos</b> PDF file for more information.






 LOGON	Click here to log on to Panoramas security system. See the optional Panorama Security Handbook for more information (this Handbook is available for an additional charge).
 PREFS	Click here to change the preferences for this wizard, including controlling what happens when you double click on the Panorama application icon (see below).

### Changing the Default Launch Action

When you double click on the Panorama application icon Panorama normally opens the **New Database** wizard automatically. You can change this, however, by clicking on the Preferences icon in this wizard.



You have six different choices of what action Panorama will take when you double click on its icon. The most common choices are:

 <b>New Database Wizard</b>	Open the <b>New Database</b> wizard (see “ <a href="#">Using the New Database Wizard</a> ” on page 38). This is the default when Panorama is first installed.
 <b>Recent Databases Wizard</b>	Open the <b>Recent Databases</b> wizard (see “ <a href="#">The Recent Databases Wizard</a> ” on page 37).
 <b>Favorite Databases Wizard</b>	Open the <b>Favorite Databases</b> wizard (see “ <a href="#">The Favorite Databases Wizard</a> ” on page 37).

### Opening a Database

Before you can work with a database you have to open it. From the desktop the quickest way to do this is to simply double click on the file’s icon. If Panorama is already running you can use the **Open File** command (File Menu) which opens a standard Open File dialog. (This dialog has several additional buttons for importing, combining and creating new databases. For now you can simply ignore these options.)

You can also open a database using the **Recent Databases** and **Favorite Databases** wizards, described later in this chapter (see “[The Recent Databases Wizard](#)” on page 37 and “[The Favorite Databases Wizard](#)” on page 37).

## Databases and RAM

When a database is opened, Panorama copies the information from the disk into the computer's internal electronic memory (RAM). Everything you do to a file takes place in RAM; including data entry, sorting, calculating, and drawing. If you want to store your work permanently, you must save it from RAM back to the disk using the **Save** command.

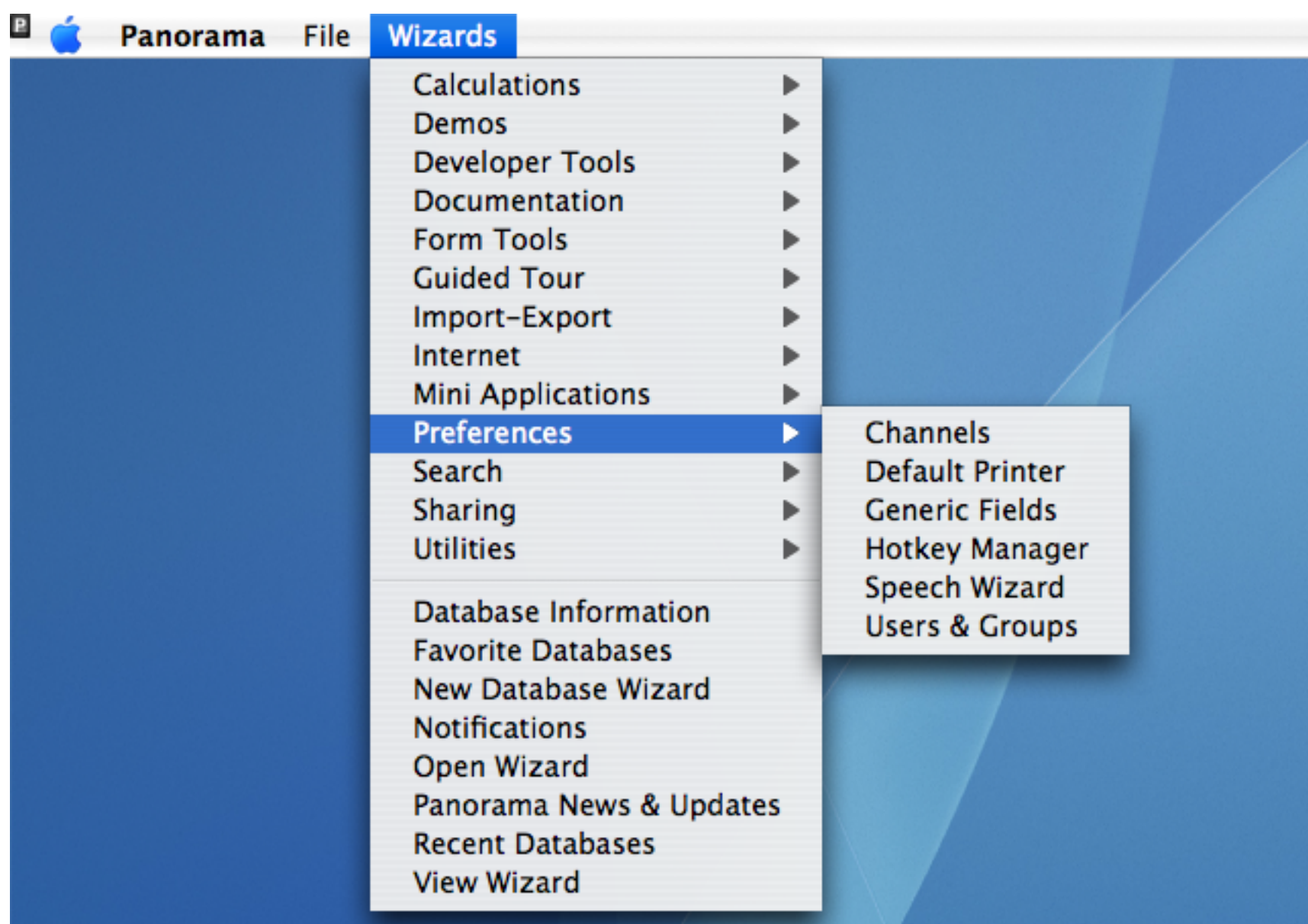
Most database programs don't take the extra step of copying the database from the disk into RAM before working with it. Since your computer can access data in RAM hundreds or even thousands of times faster than data on the disk, bringing the data into RAM makes Panorama much faster than most other database programs. If you've used other database programs you'll immediately notice how much "zipper" Panorama is compared to the programs you are used to.

If your computer has enough RAM available, you can open several Panorama databases at the same time. For more information about working with multiple files see "[Opening Multiple Files](#)" on page 51.

If your files are very large you may need to tell Panorama to use more memory. See "[Monitoring Memory Usage](#)" on page 73 for more information on this topic.

## The Wizard Menu

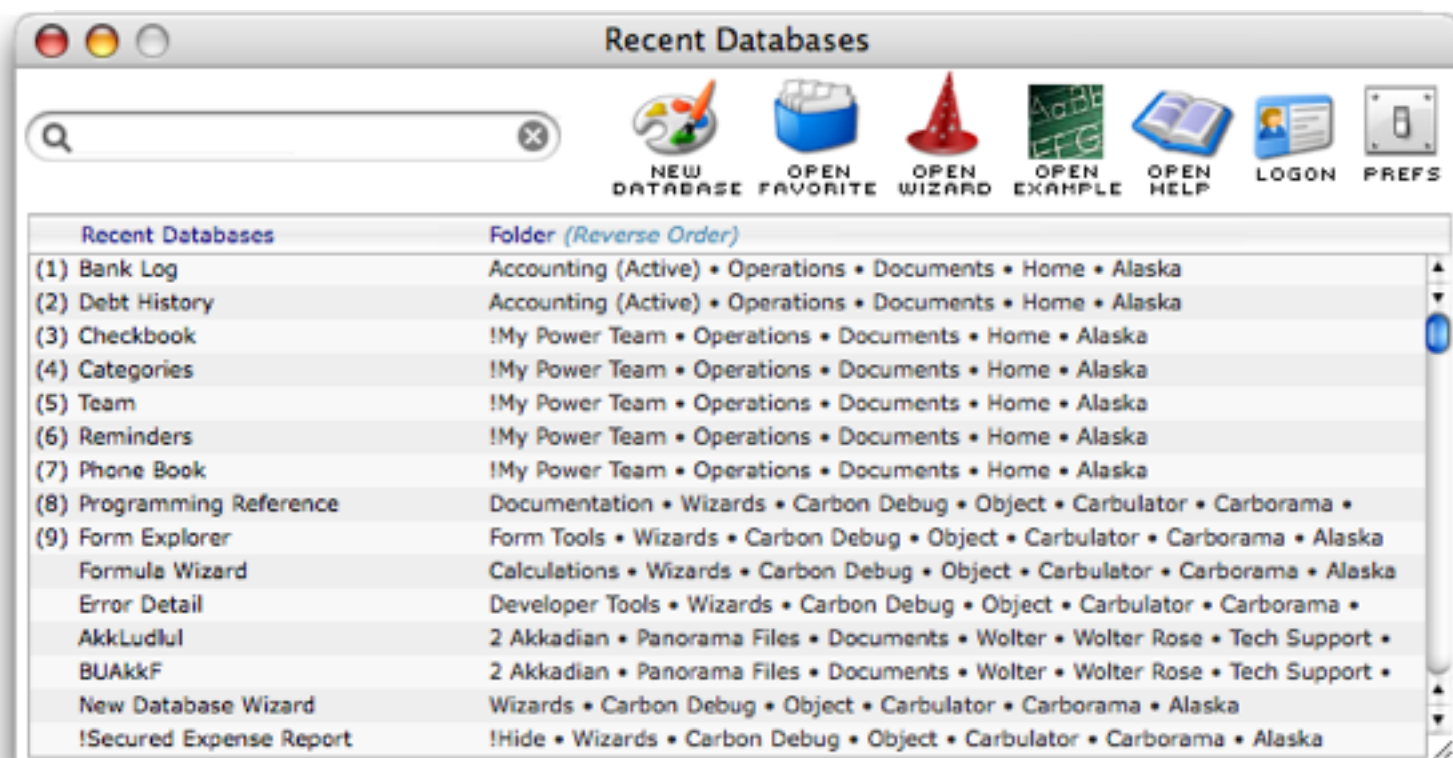
Panorama includes a number of pre-built databases that you can use as is, modify for your own purposes, or simply use as learning tools. With only a few exceptions these pre-built databases are completely accessible so that you can not only use them as is but also take them apart and see how they work. All of these databases can be opened with the **Wizards** menu and its submenus.



Many of these wizards are described throughout this manual. You can also find a complete description of every wizard in the separate **Wizards & Demos** PDF file.

## The Recent Databases Wizard

This wizard makes it easy to re-open recently opened databases. It can be opened by choosing **Recent Databases** from the **File** menu in addition to being listed in the **Wizard** menu. The wizard lists the databases that have been opened recently.



To re-open a database simply double click it's name on the list. You can also open the first nine items simply by pressing the **1** thru **9** keys on your keyboard. There's no need to press **Return**, **Enter**, or anything else, just press the number and the database will open.

To search for a particular database simply type into the search box at the top of the wizard.



At any time you can press the **1** thru **9** keys to re-open a database. For example type **boo1** to open the **Checkbook** database, **boo2** to launch the **Phone Book** database, etc. You can re-open any previously opened database with just a few keystrokes.

## The Favorite Databases Wizard

Another way to access frequently used databases is to use the **Favorite Databases** wizard, which allows you to build a list of your favorite databases (kind of like the **Bookmark** menu in a web browser). To learn more about this wizard see Chapter 1 of the **Panorama Handbook**.

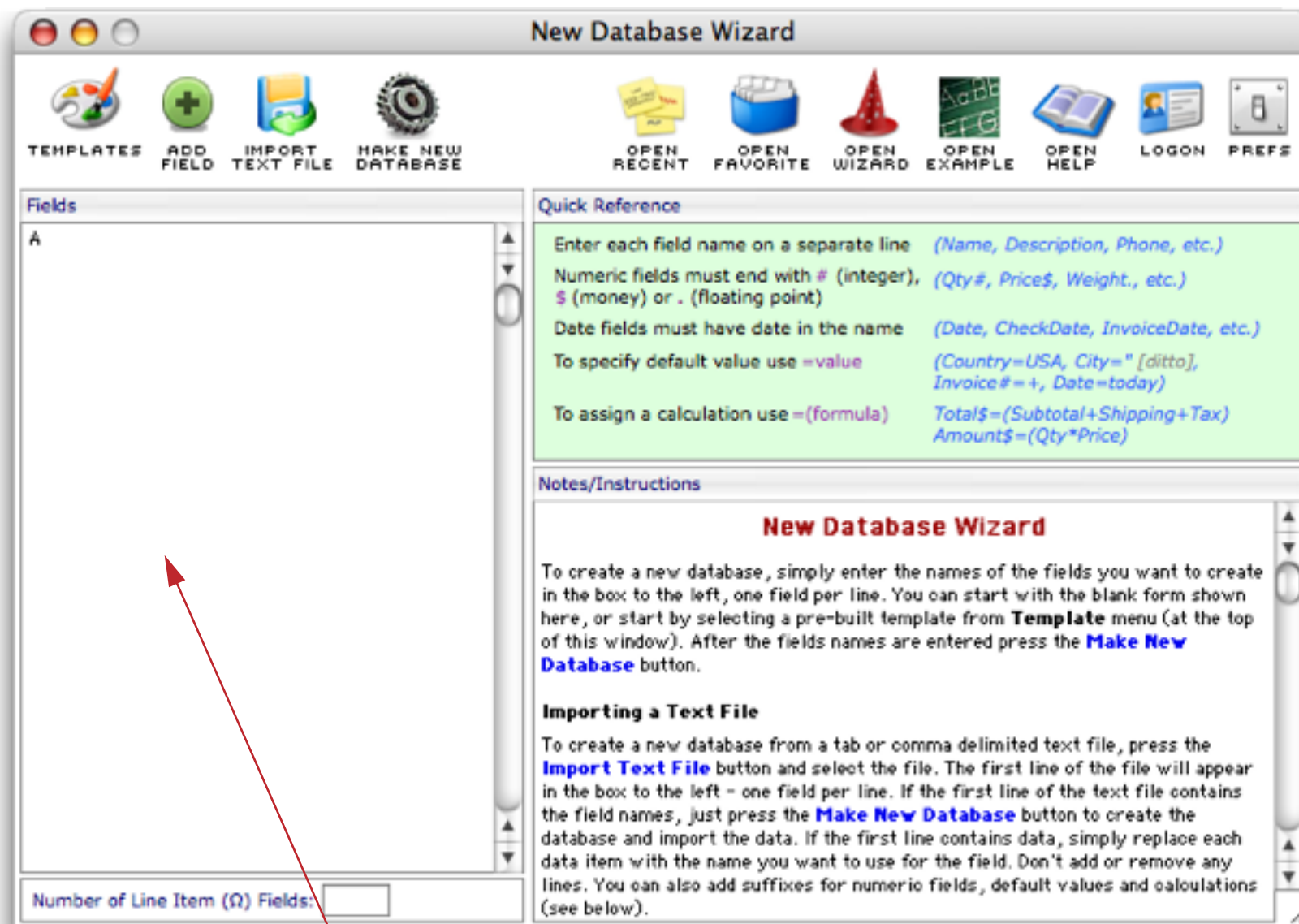


## Creating a New Database

Panorama has two ways to create a new database — using the New Database wizard and using the Open File dialog.

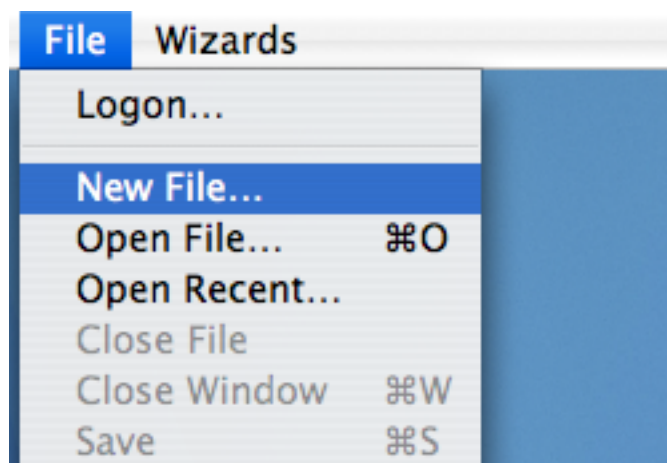
### Using the New Database Wizard

To help make creating new databases easier Panorama includes “wizard” for creating new databases.

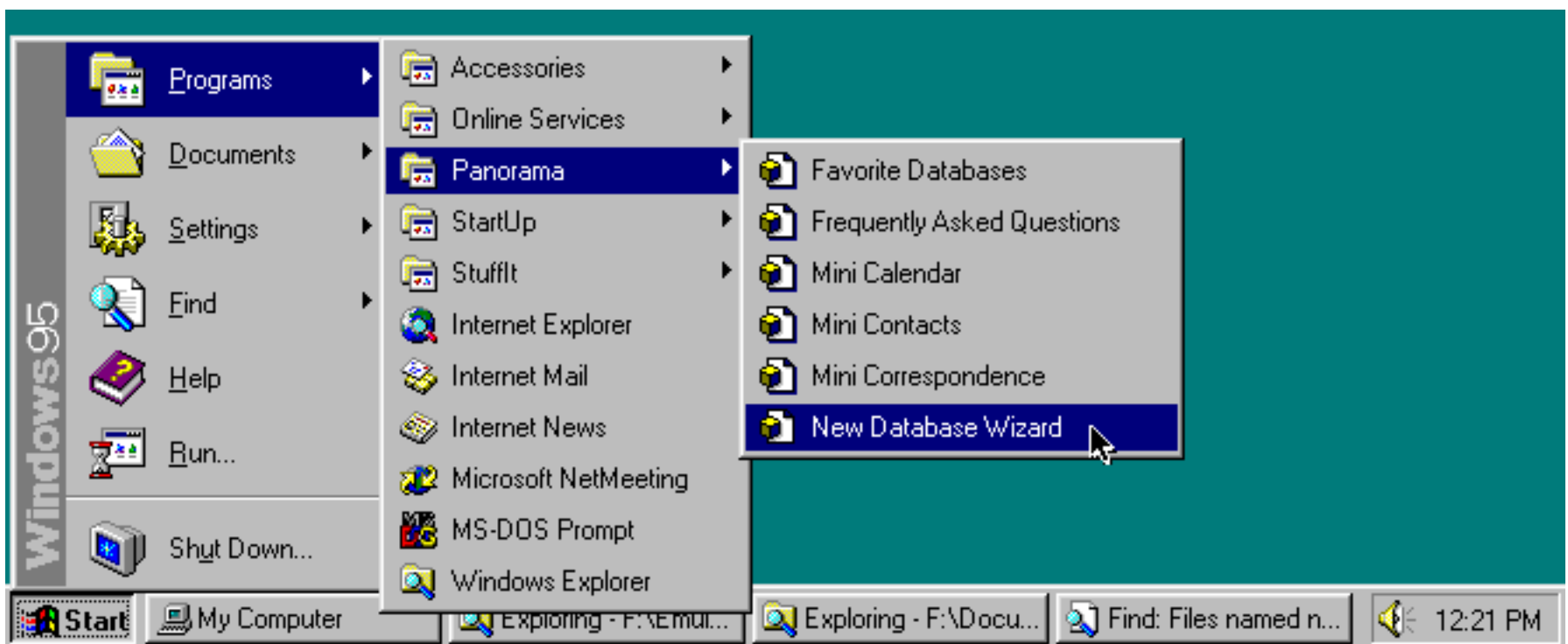


*type field names into this box*

If Panorama is already open you can choose New File from the File menu to open this wizard. (You can also use the Wizards menu to open the **New Database Wizard**.)

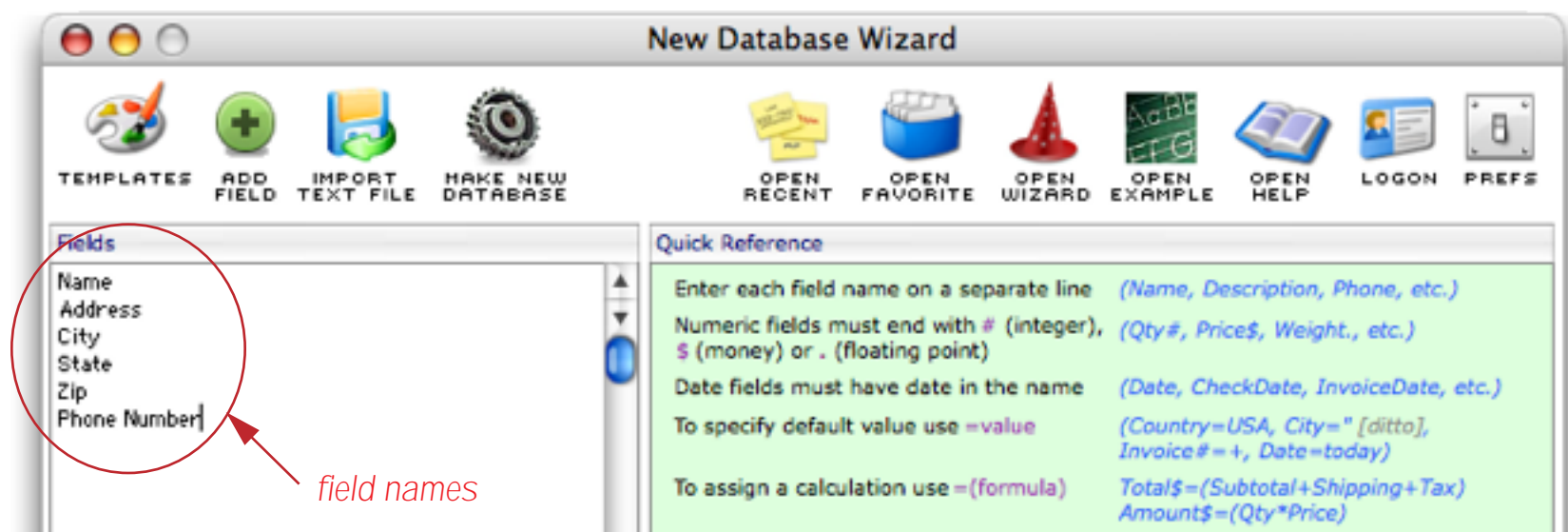


If you are working on a Windows computer you can also open the **New Database Wizard** directly from the Start Menu.

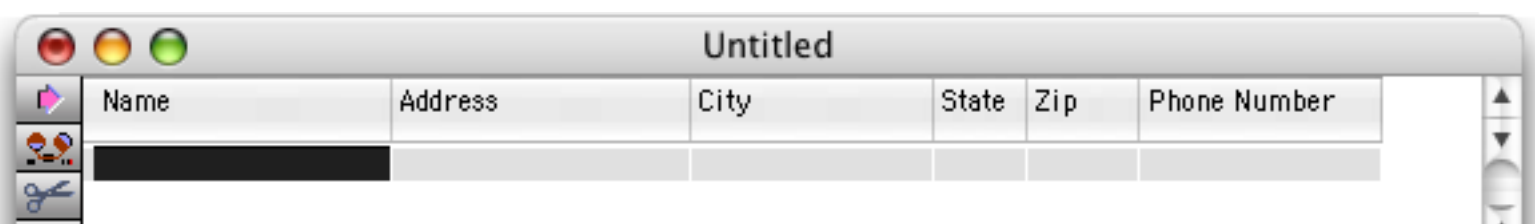


### Creating a Database with the Wizard

To create a database simply type in the name of each field, one per line.



To create the database simply press the **Make New Database** button.

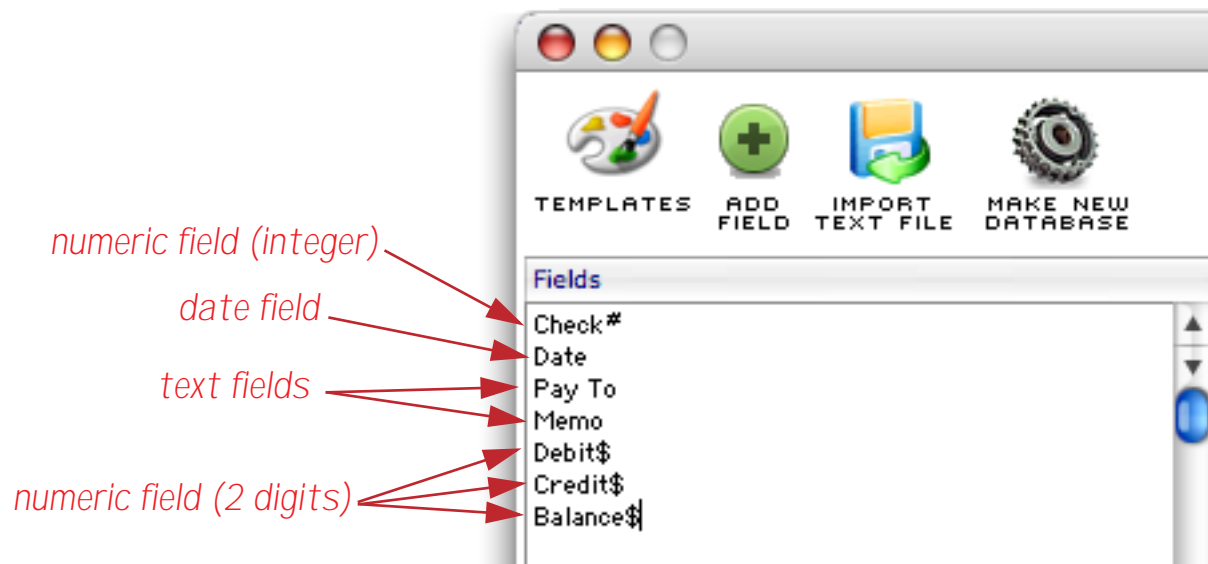


Panorama automatically sets the width of each field. See "[Changing the Width of a Field](#)" on page 217 to learn how to manually adjust the width of a field. Before you continue you should save your new database (see "[Saving a Database](#)" on page 48).



### Creating Numeric and Date Fields

In Panorama, all data is not the same. To get the most out of a database, Panorama needs to know what type of data you intend to store in each field — text, numbers or dates (see “[Data Types](#)” on page 113). The **New Database Wizard** normally creates fields designed for holding text. Adding #, \$ or . at the end of a field name tells the wizard to create a numeric field (see “[Numeric Data](#)” on page 116). A field with a name ending in # (for example **Check#**) can hold integer values (for example 1, 23 or 456). A field with a name ending in \$ (for example **Price\$**) can hold numbers with two places after the decimal point (for example 4.87 or 783.98). A field with a name ending in . (for example **Weight.**) can hold any floating point value.



Press the **Make New Database** button to create the new database. Notice that the # and \$ suffixes that were typed into the wizard are not included in the final field names. We’ve typed in one line of data to show the different types of data stored in each field.

Check	Date	Pay To	Memo	Debit	Credit	Balance
100	10/07/04	ABC Cleaner's		14.87		

Any field whose name contains the word **Date** (for example **Date**, **Start Date** or **Check Date**) will be created as a field for date values (see “[Dates](#)” on page 121). In the example above the second field (**Date**) is just such a field.

### Default Values

To assign a default value to a field (see “[Default Values](#)” on page 144) follow the field name with an = symbol followed by the default value. For example:

`Country=USA`

`ShipMethod=Federal Express`

To repeat the previous value in this field (ditto) use the " symbol.

`City="`

`Date="`

To automatically increment a numeric or date field use `+`, or plus followed by a number (`+1`, `+2`, `+5`, etc.).

```
Check Number#=#+1
```

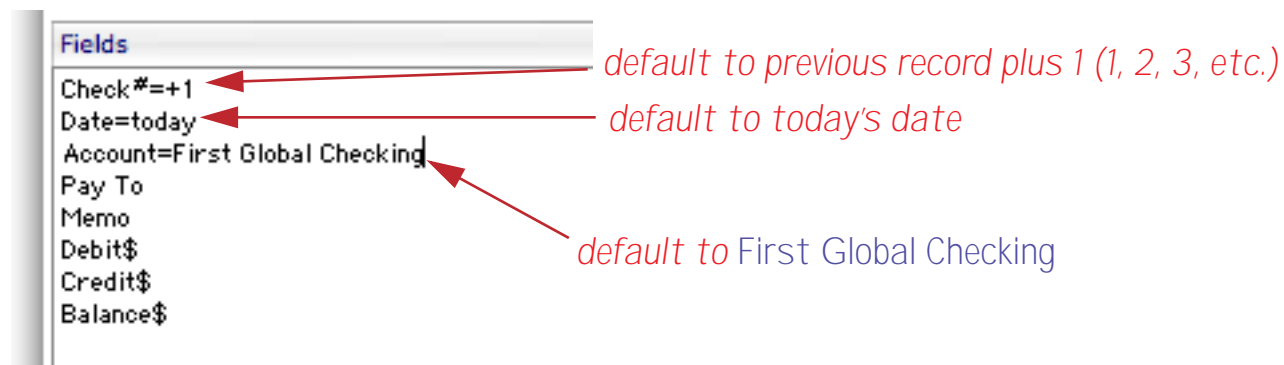
```
Invoice#=#+
```

To default to today's date use `=today`.

```
Date=today
```

```
InvoiceDate=today
```

In this example three fields have default values.



This shows the result of creating this database and adding several fields.

Check#	Date	Account	Pay To	Memo	Debit	Credit	Balance
1	07/28/04	First Global Checking					
2	07/28/04	First Global Checking					
3	07/28/04	First Global Checking					

When a new record is added Panorama will automatically fill in the **Check** number, **Date**, and **Account**.

### Automatic Calculations

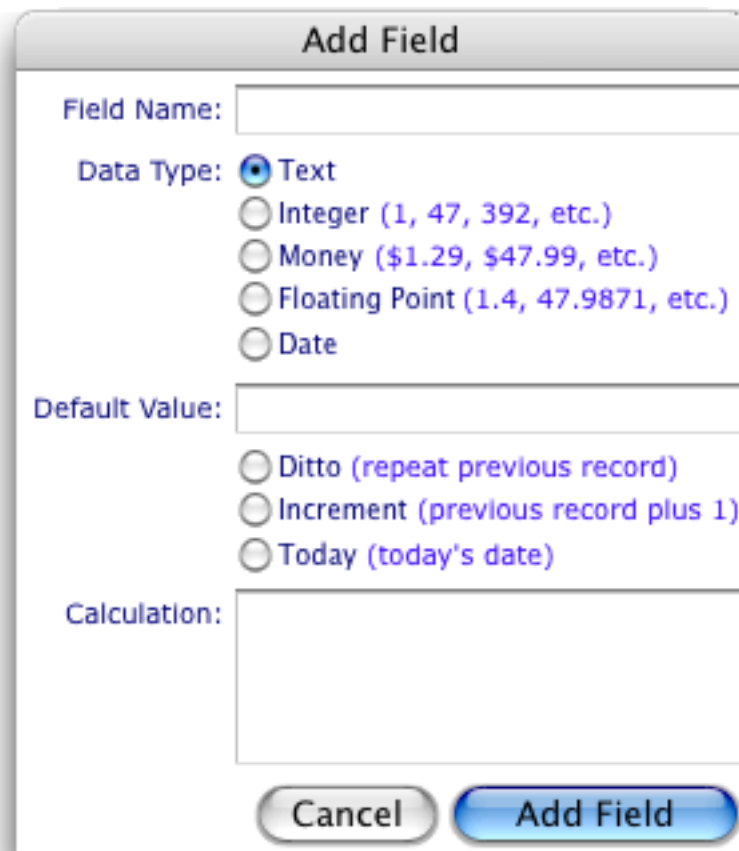
To assign an automatic calculation to a field (see “[Automatic Calculations](#)” on page 149) follow the field name with an `=` symbol followed by the formula. The formula must be surrounded by ( and ) parentheses. Using the wizard you cannot assign both a default value and a calculation to the same field. Use Panorama's Design Sheet (see “[The Design Sheet](#)” on page 105) if this is necessary. For example:

```
Total$=(Subtotal+Shipping+Tax)
```

```
Amount$=(Qty*Price)
```

### Adding Fields with the Add Field Dialog

In addition to adding fields by typing them in directly you can also add them with the assistance of a dialog. Start by pressing the **Add Field** icon (second from the left). This causes the **Add Field** dialog to appear.

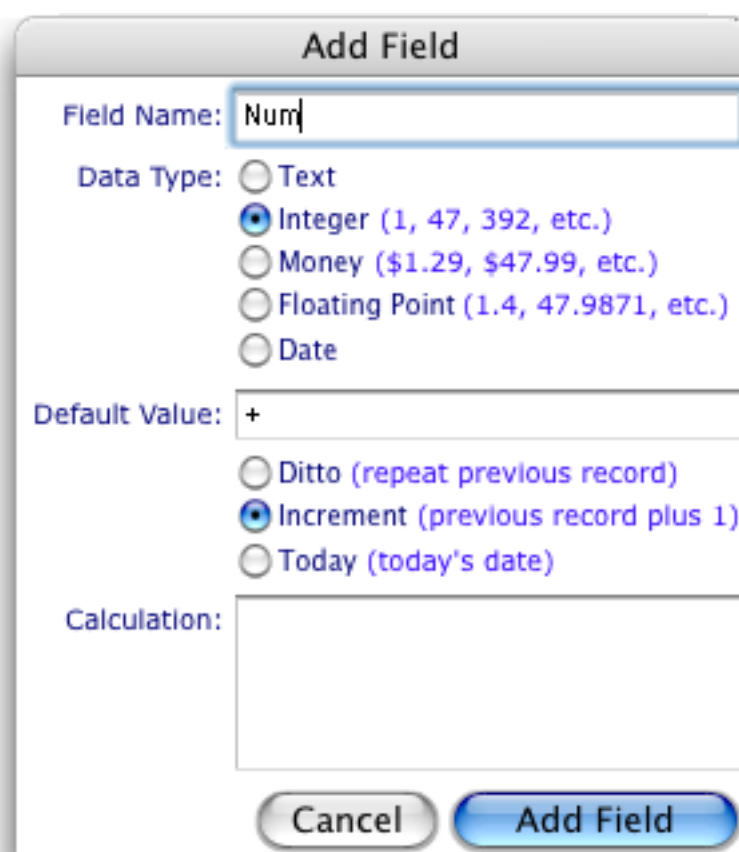


The 'Add Field' dialog box is shown with the following fields and options:

- Field Name:** An empty text input field.
- Data Type:** A group of radio buttons with the following options:
  - Text
  - Integer (1, 47, 392, etc.)
  - Money (\$1.29, \$47.99, etc.)
  - Floating Point (1.4, 47.9871, etc.)
  - Date
- Default Value:** An empty text input field.
- Default Value Options:** A group of radio buttons with the following options:
  - Ditto (repeat previous record)
  - Increment (previous record plus 1)
  - Today (today's date)
- Calculation:** An empty text input field.

At the bottom of the dialog are two buttons: **Cancel** and **Add Field**.

At a minimum you must fill in the Field Name and choose a data type. You may also fill in a default value or a calculation (but not both -- to do that you must use the design sheet).

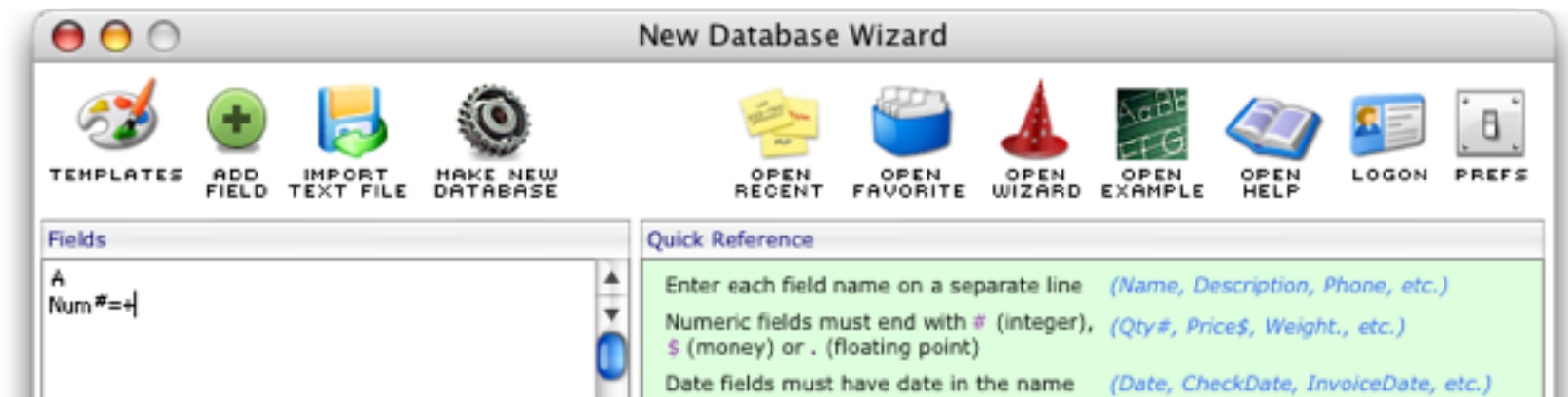


The 'Add Field' dialog box is shown with the following fields and options:

- Field Name:** A text input field containing the text "Num".
- Data Type:** A group of radio buttons with the following options:
  - Text
  - Integer (1, 47, 392, etc.)
  - Money (\$1.29, \$47.99, etc.)
  - Floating Point (1.4, 47.9871, etc.)
  - Date
- Default Value:** A text input field containing the text "+".
- Default Value Options:** A group of radio buttons with the following options:
  - Ditto (repeat previous record)
  - Increment (previous record plus 1)
  - Today (today's date)
- Calculation:** An empty text input field.

At the bottom of the dialog are two buttons: **Cancel** and **Add Field**.

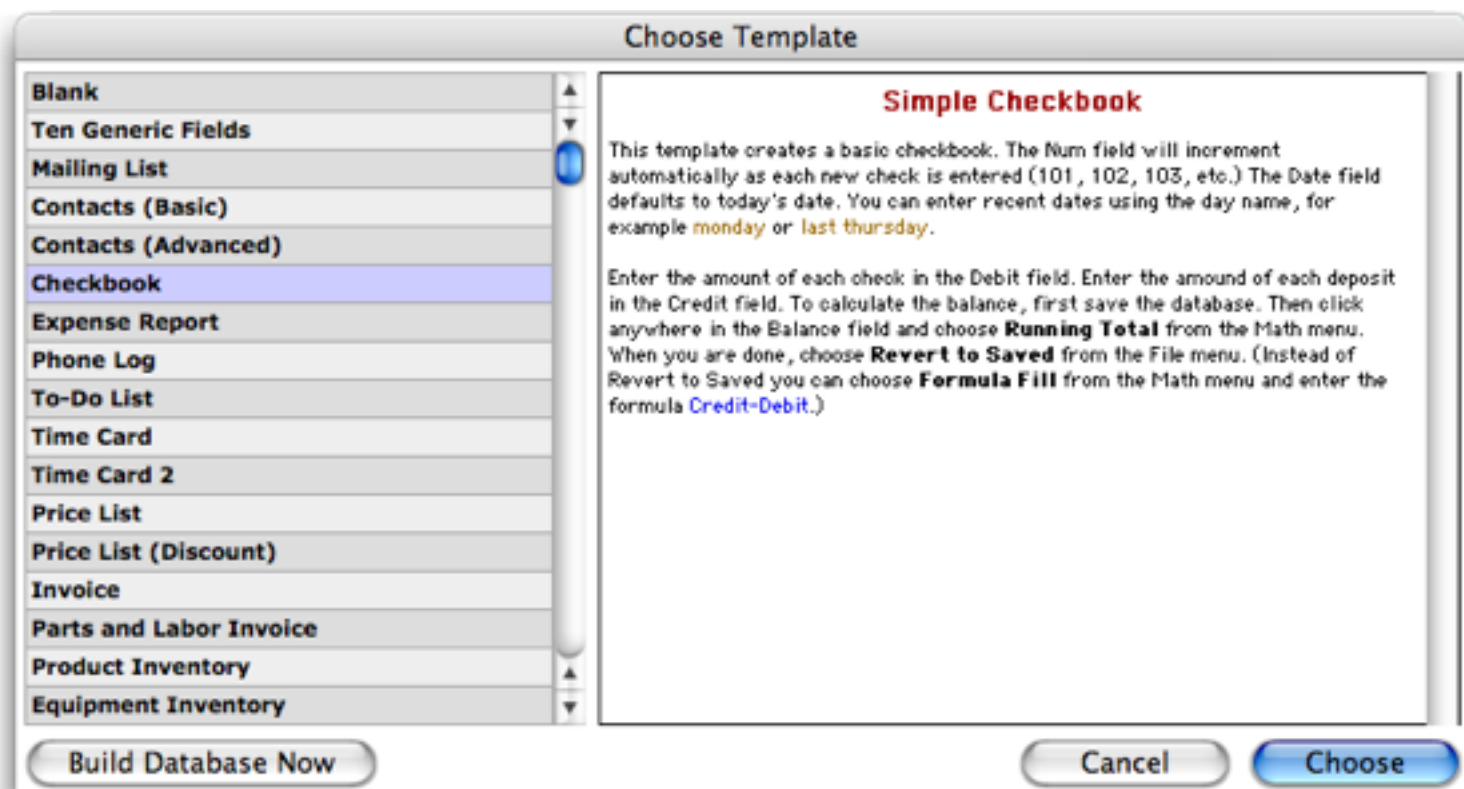
When you press **Add Field** the new field specification is added to the wizard.



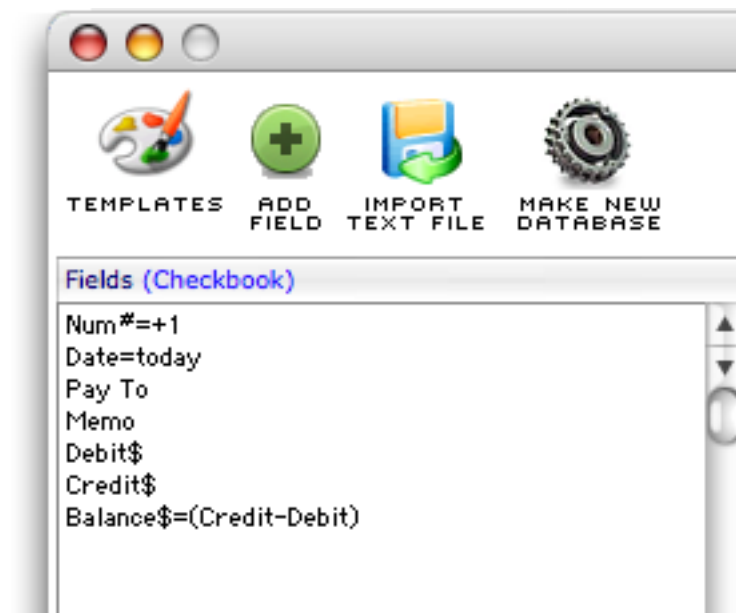
You can continue to add more fields with this dialog, and of course you can also add fields by typing them in (as well as edit fields you have already set up).

### Starting with a New Database Template

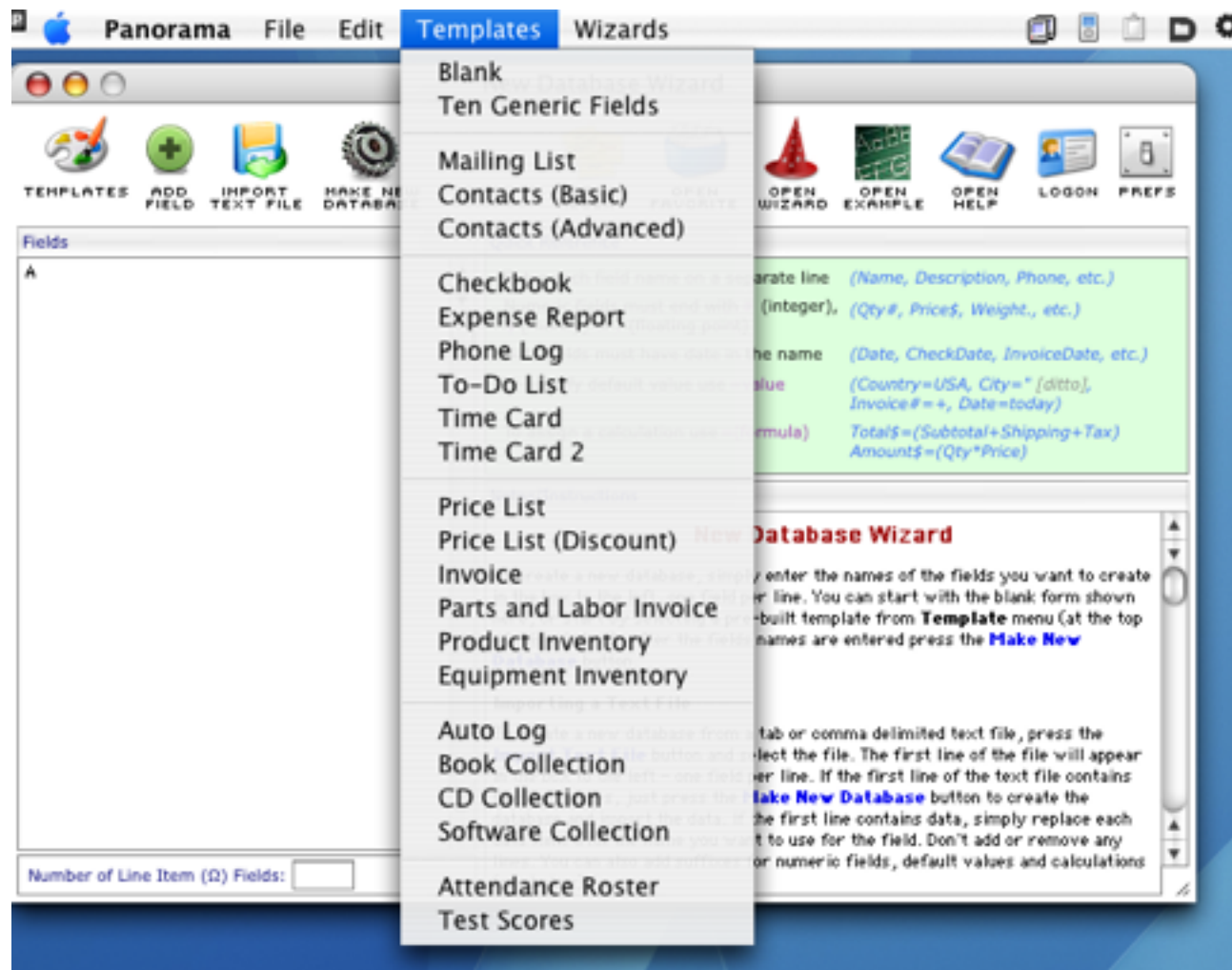
To help you get started the **New Database Wizard** includes a number of templates. To see this list press the **Templates** icon on the left. A dialog appears with a list of templates on the left hand side. Click on any template to see a description you want. Press the **Choose** button to copy this template into the **New Database wizard**, where it can be modified further if necessary. (If you want to use a template exactly as is, simply press the **Build Database Now** button.)



When you press **Choose** the template is copied into the wizard. You can modify the template if necessary. When it is ready press the **Make New Database** button.



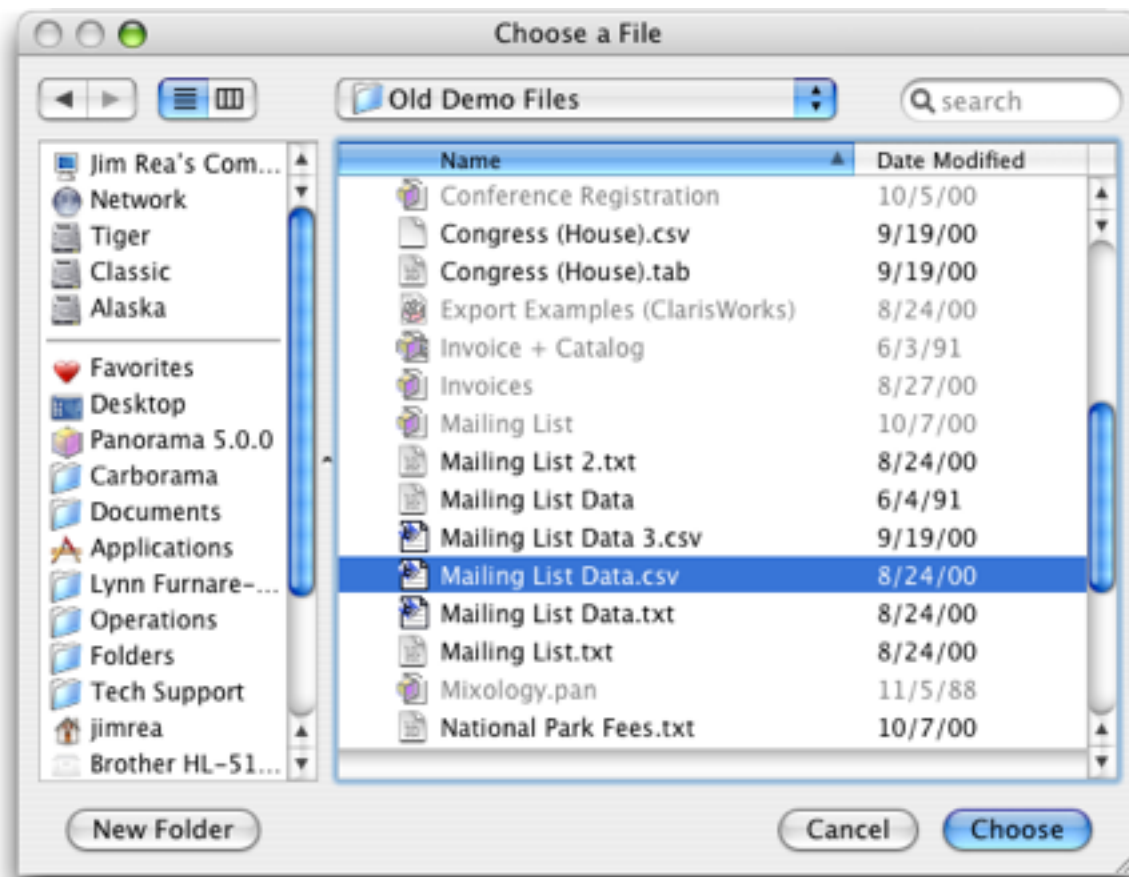
You can also choose a template from the **Templates** menu



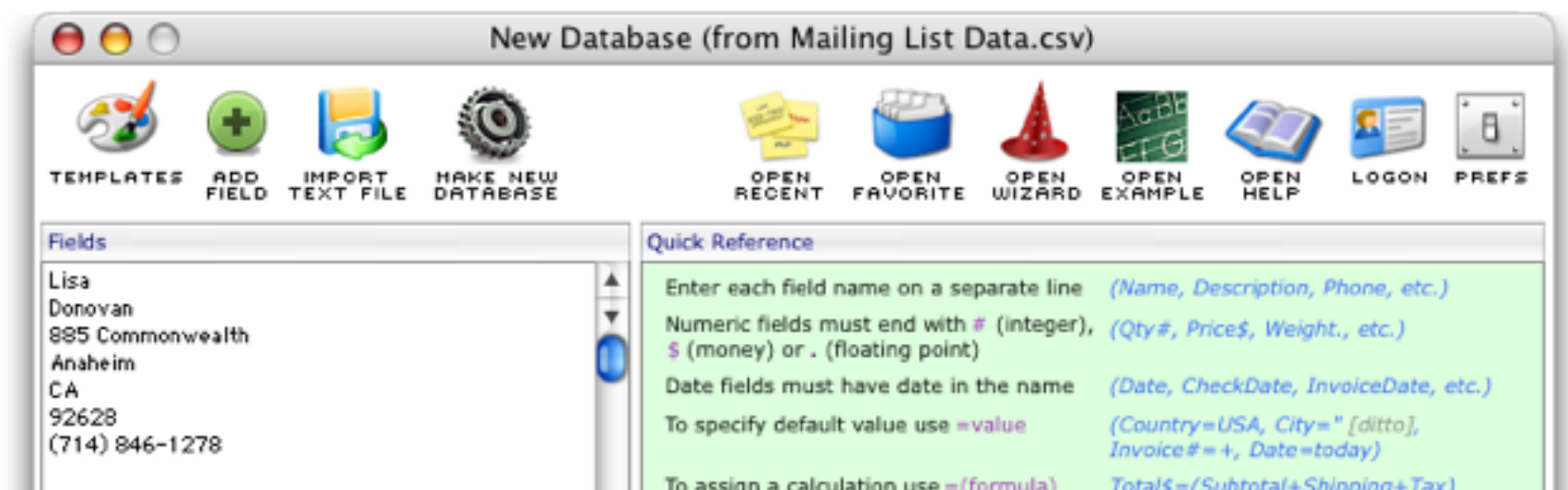


### Creating a Database from a Text File

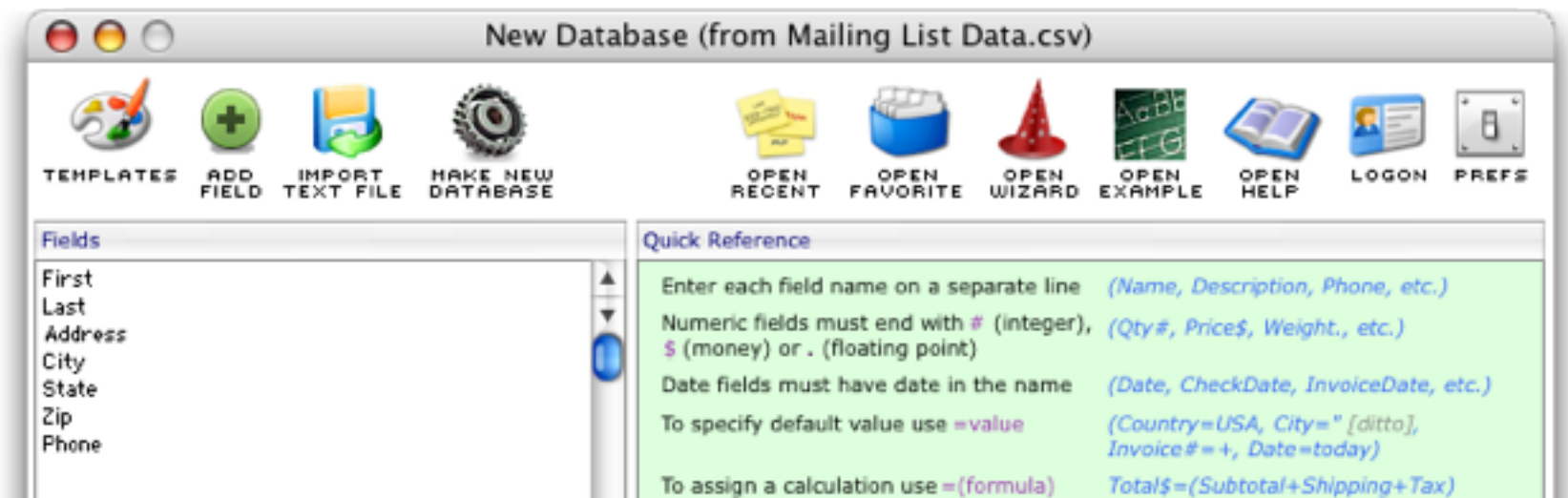
The **New Database Wizard** usually creates an empty database, but it can also create a database from a text file and automatically import the text (see “[Importing a Text File](#)” on page 54). To start this process press the **Import Text File** button at the bottom of the window. The wizard will display a dialog box asking you to select the text file you want to use.



In this case we have selected the file **Mailing List Data.csv**, a comma separated text file. When you press **Choose** the wizard displays the data from the first line of the text file.



Using the data as a template, replace the data with the field names you want to use.



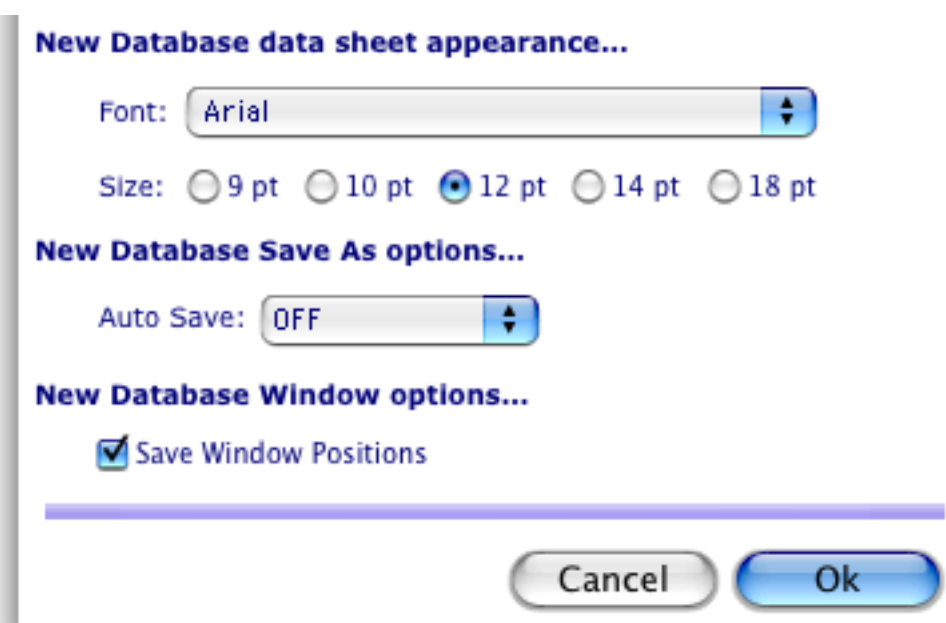
When you press the **Make New Database** button the wizard automatically creates the new database and imports the data.

First	Last	Address	City	State	Zip	Phone
Lisa	Donovan	885 Commonwealth	Anaheim	CA	92628	(714) 846-1278
Kirk	Shelby	731 Melody Lane	Brea	CA	92624	(714) 894-2489
Bob	Musco	624 Fountain Street	Costa Mesa	CA	92604	(714) 964-0922
Jim	Reynolds	1183 Brookhurst	Costa Mesa	CA	92608	(714) 894-2465
Jack	Rutan	4910 Glendale	Costa Mesa	CA	92642	(714) 895-1561
Chris	Murphy	906 Springdale	Diamond Bar	CA	92618	(714) 959-4712
Mike	Johnson	391 E. Raymond	Fullerton	CA	92625	(714) 984-3547
Russ	Greene	1099 E. Dorothy Lan	Fullerton	CA	92625	(714) 865-4871
Scott	Lutz	448 Longview	Fullerton	CA	92631	(714) 986-7448
Mary	Matthews	891 E. Graham	Huntington Bea	CA	92649	(714) 525- 8431
Joy	Scott	7780 N. Harbor	Newport Beach	CA	92640	(714) 894-5341
Fred	Claire	341 Pinecrest	Orange	CA	92450	(714) 195-1895
Cheri	Allen	399 S. Batavia	Orange	CA	92634	(714) 426-7819
Paul	Kennedy	3143 Polk	San Francisco	CA	98457	(415) 894-4679

Be sure to save the new database (see “[Saving a Database](#)” on page 48) before you continue.

#### New Database Options (Font, Windows, Save As)

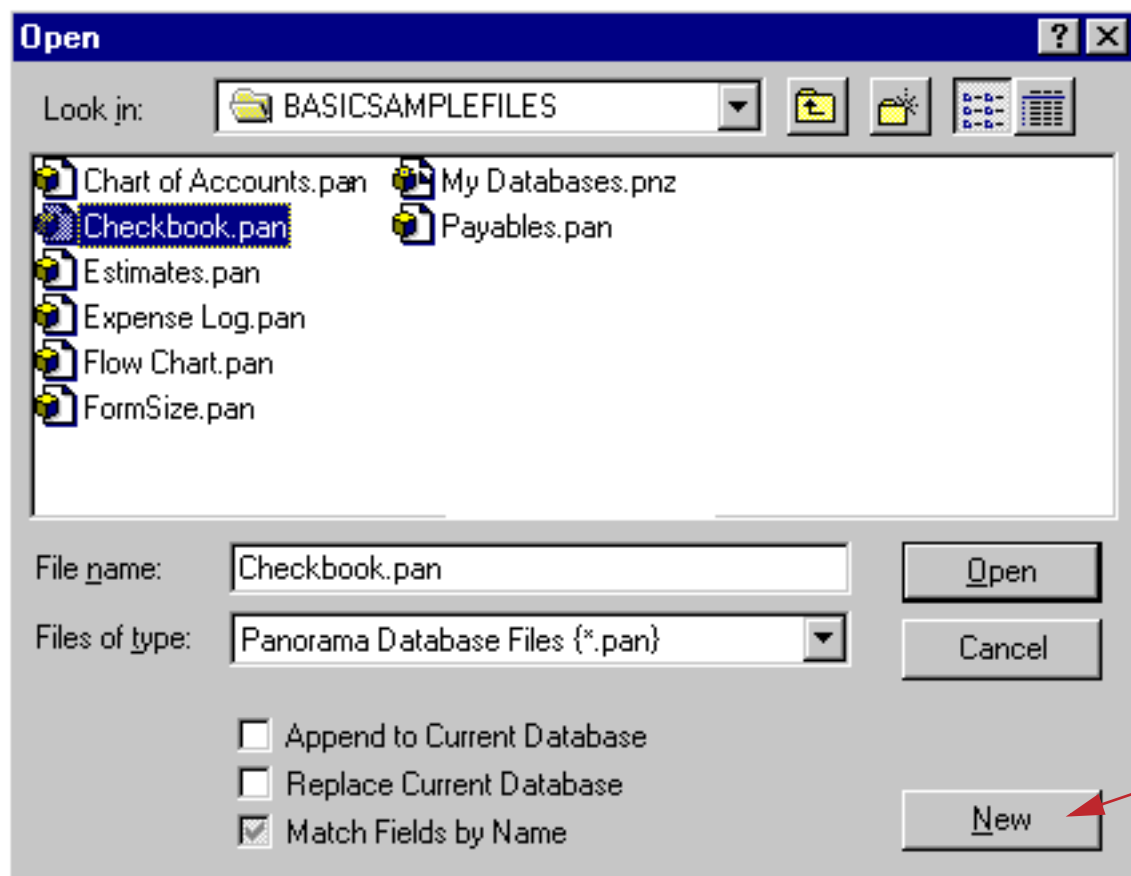
Click the Prefs icon (on the upper far right) to change the preferences for creating a new database.



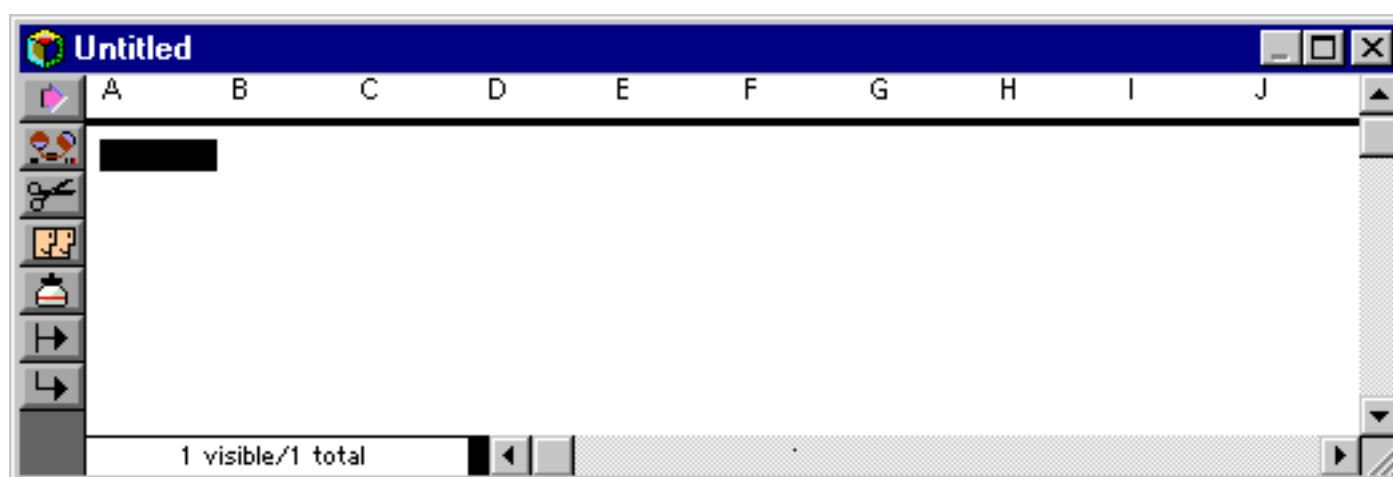
You can specify any font and size you want for your new data sheets. (Of course you can change this at any time later with the Font menu.) If you want your database to be saved automatically at regular intervals you can select this here (see “[Auto-Save](#)” on page 50 to learn how to change this later. Finally you can choose whether you want Panorama to save your window positions when you save the database.

### Creating a New Database with the Open File dialog

The Open File dialog also provides a simple method for creating a new Panorama database. Simply choose **Open File** from the File Menu, then press the **New** button.

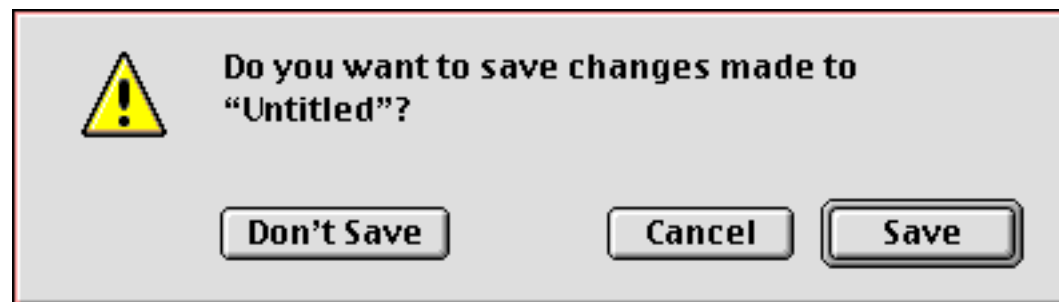


A new database created this way has ten fields and is called **Untitled**. These fields are named alphabetically: **A** through **J** (You can change the names later). You can start entering data right away or you can change the fields before you begin (see “[Fields](#)” on page 101).



## Closing Panorama

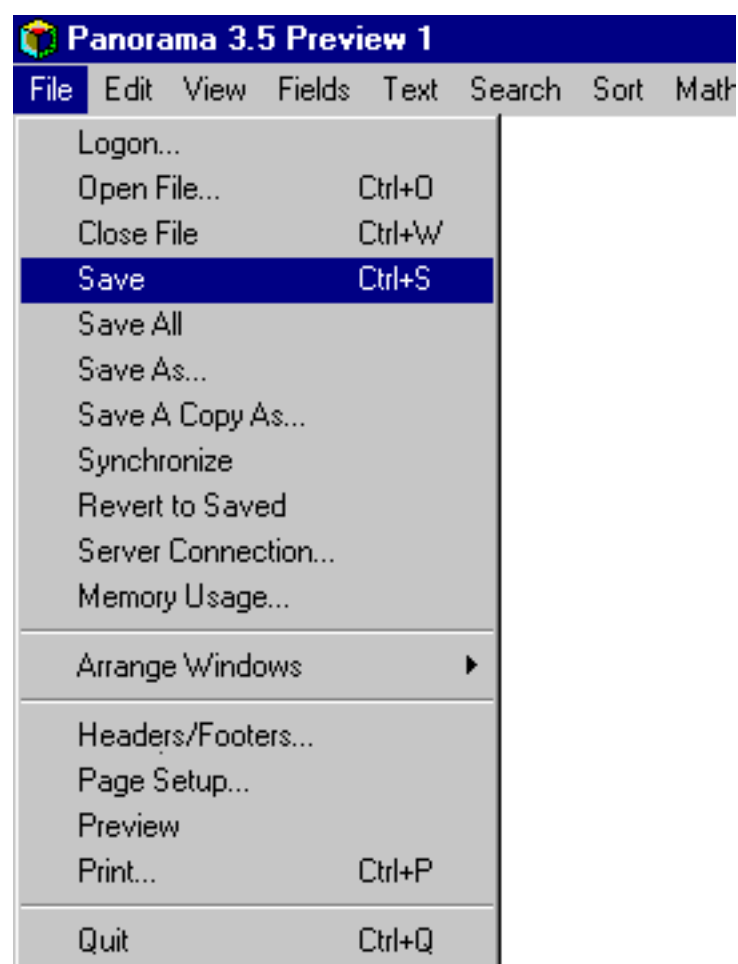
When you are finished with Panorama, use the **Quit** command (File Menu) to return to the desktop. If you have not saved your work, Panorama will display an alert and ask if you would like to save the work now.



Simply press the **Save** button to permanently save the database on the disk and return to the desktop.

## Saving a Database

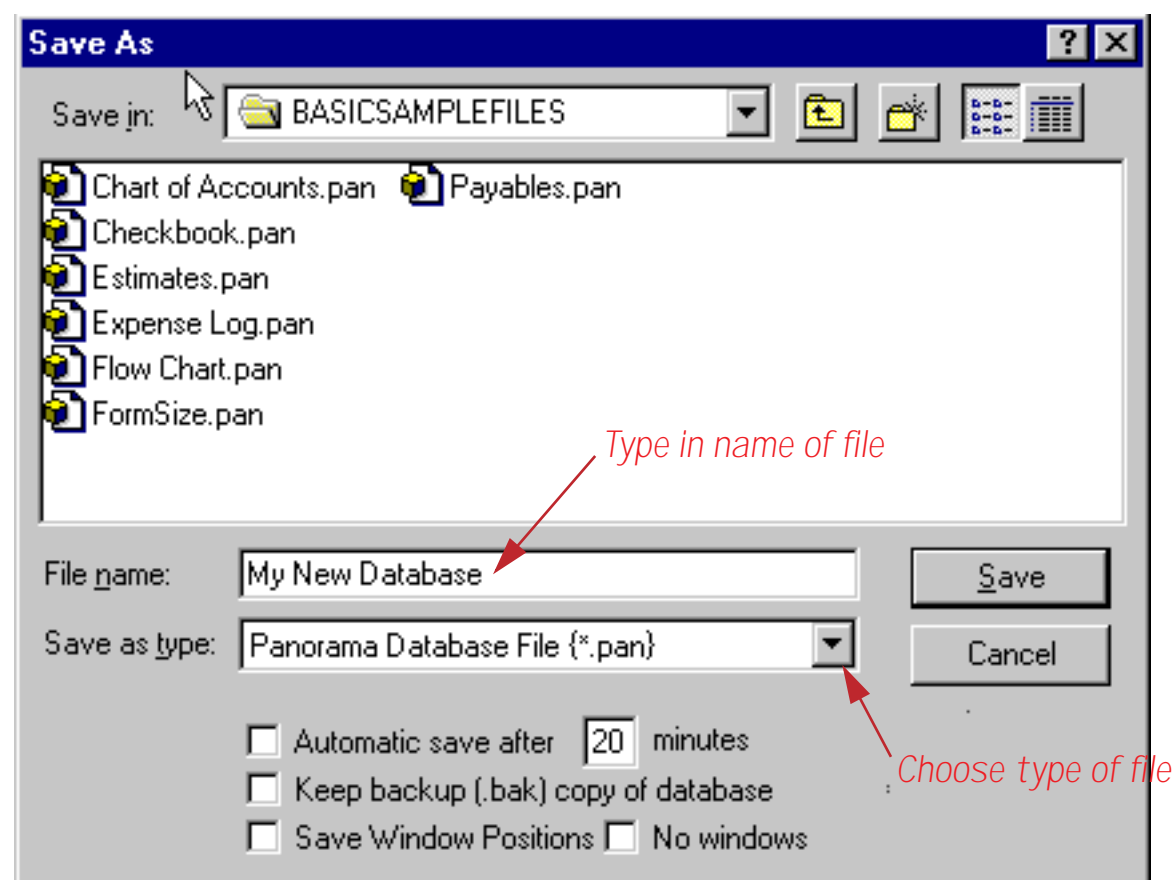
You can save your work permanently on the disk at any time with the **Save**, **Save As**, or **Save a Copy As** commands in the File menu. The **Save** command saves the database permanently on the disk, then allows you to continue with your work. You should save your work often.



The **Save A Copy As** command makes a copy of the database under a new name. It leaves the original copy in memory so you can continue to work on it. This command is like duplicating a sheet of paper and then continuing to work on the original.

The **Save As** command also makes a copy of the database under a new name. The **Save As** command, however, leaves the new copy in memory—not the original. This command is like duplicating a sheet of paper and then working on the copy while setting the original aside.

The **Save As** command allows you to choose the location where you want to save the file, the name of the new file, and several file options.

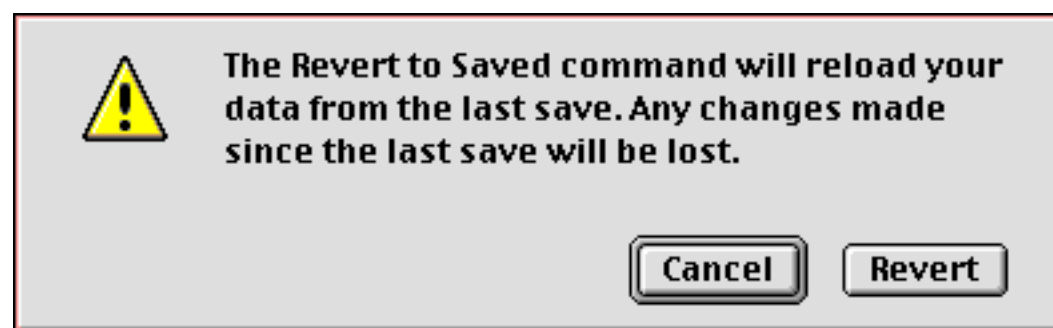


On Windows computers all Panorama database names end with **.pan**. This is called the **extension**, and it tells the system that this file is a Panorama database. You should not type the extension into the Save As dialog box—Panorama will automatically add the extension for you.

You can save the database as a regular Panorama file, as a text only file (see “[Exporting a Text File](#)” on page 63), or create a file set (see “[File Sets](#)” on page 52). You can also use the Save As command to turn on the **Save Window Positions**, **Auto-Save** and **Keep Backup Copy** options.

### Revert to Saved

The **Revert to Saved** command (File Menu) recopies the original file from the disk into RAM. This will undo all the changes made since the last time the file was saved. By all changes we mean all changes: data entry, sorting, formulas, graphic editing, creating/deleting forms, crosstabs or procedures—every single thing you’ve done to this database since the last time you saved. Before Panorama actually goes ahead with this command it asks you to verify that you really want to do this.



Be absolutely sure before you press the **Revert** button. Any work you have done since the last time you saved will be gone forever. On the other hand, this command is a great safety net that can let you recover from mistakes. Even if you accidentally delete all the data in your entire database, you can easily get it back with this command. Remember, however, that this safety net only goes back to the last time you have saved. If you accidentally delete all the records and then **Save**, you are out of luck (unless you have made some other backup).



## Auto-Save

If you wish, you can ask Panorama to automatically save your file every few minutes. To turn on this feature, choose **Save As**, then check the **Automatic Save** option and enter how often you want Panorama to save the file. When you are done, press the **Save** button. (If you are using a Macintosh computer and you have saved this file before, Panorama will ask you if you want to replace the existing file. Press **Yes**).

Once you have turned on the Automatic Save option it remains on unless you **Save As** again and turn it off.

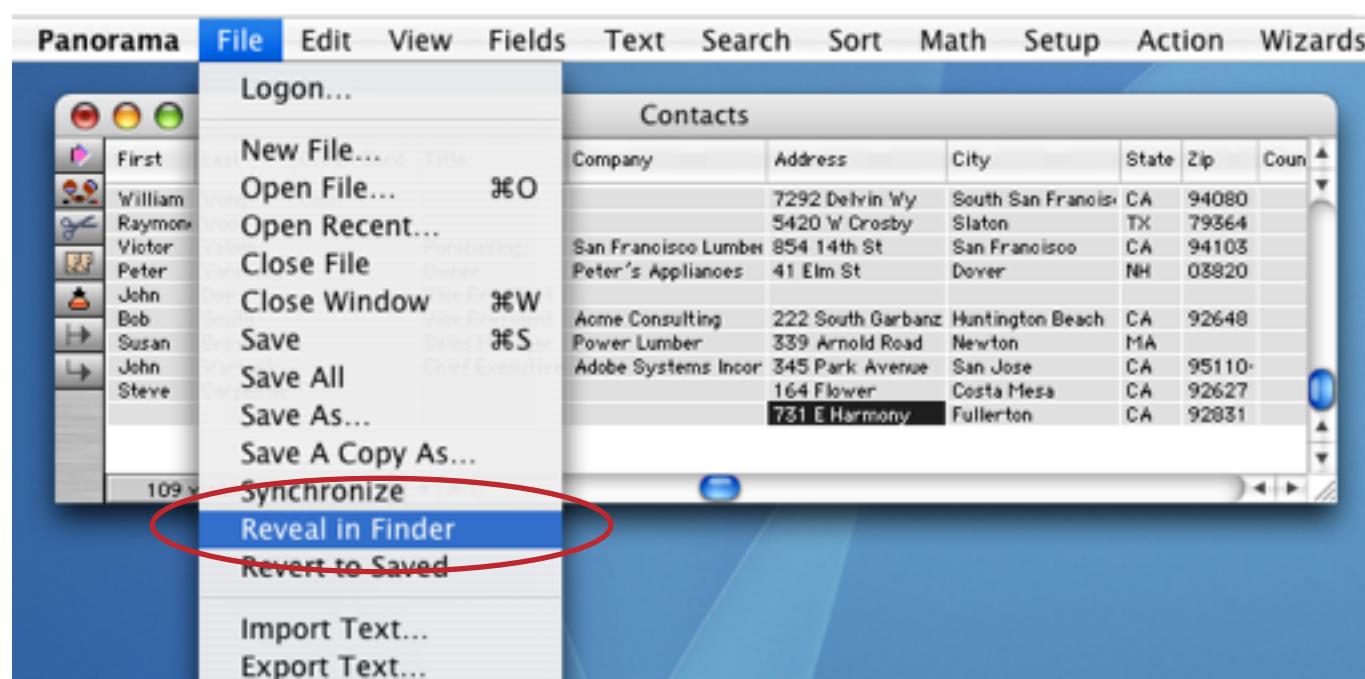
### Pitfalls of Auto-Save

If you think you might need to use **Revert to Saved**, you should not use auto-save. When you are using auto-save keep in mind that you no longer control when the file will be saved. This lack of control can cause problems if Panorama saves the file when you didn't want it to. In particular, if you have deleted items from your database you won't be able to get them back using **Revert to Saved** once the file has been automatically saved.

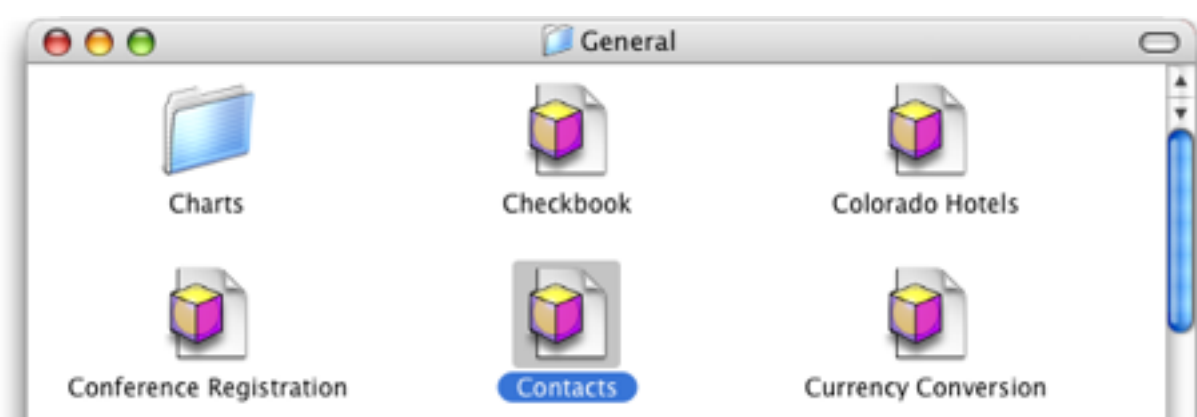
Because they can remove large amounts of data from your database, the **Remove Unselected** and **Remove Detail** commands are especially dangerous with auto-save. To help reduce this danger, these commands give you the option to temporarily suspend auto-save. Once you have suspended auto-save it will remain off until you manually save the file (with the **Save** command).

## Finding a Database on the Hard Disk

To find the location of the current database on the hard drive simply choose the **Reveal in Finder** command from the **File** menu.



This will open the folder containing the database and highlight the database file.



Note: This feature is only available in Mac OS X.

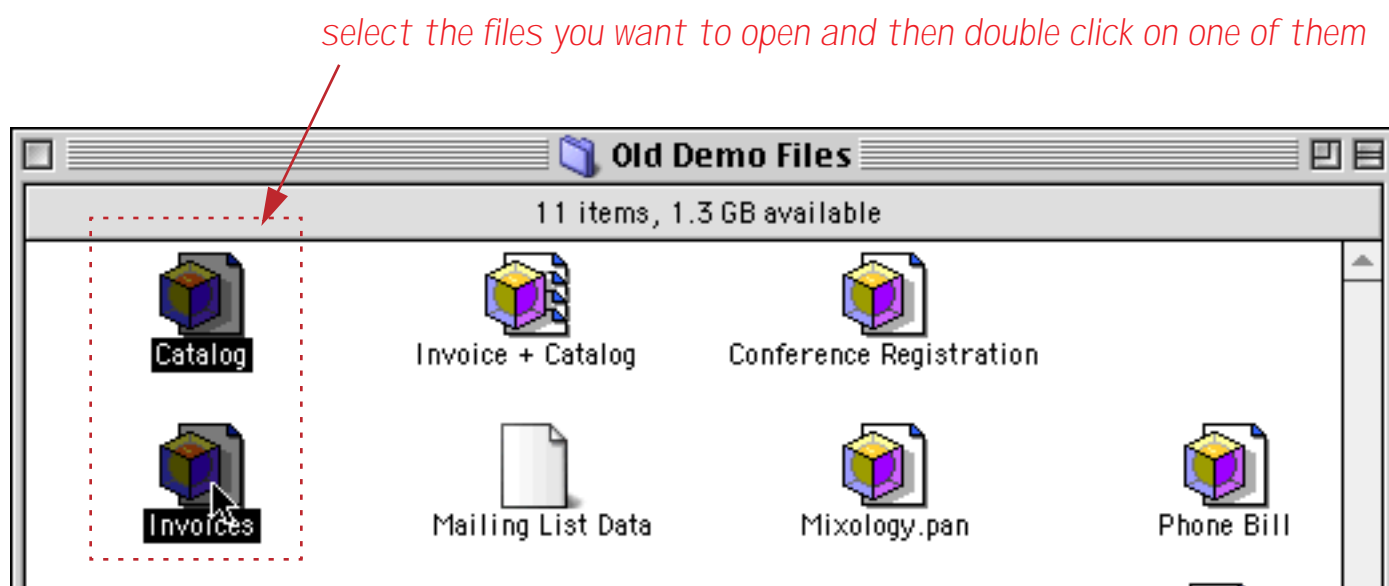
## Working with Multiple Databases

Panorama can work with several databases at the same time. The databases can be independent, or they may contain related information.

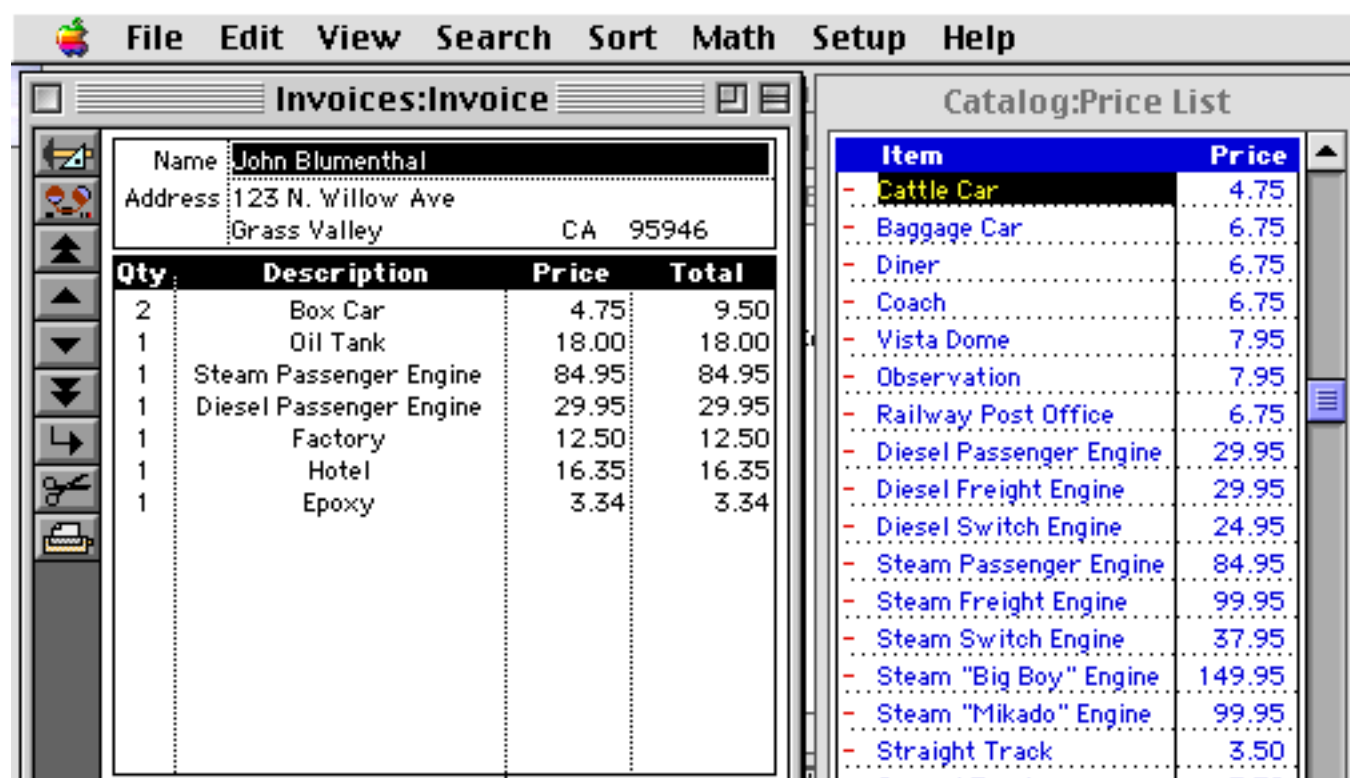
As you use Panorama, you'll probably encounter many tasks that require working with several databases at once. For example, when you are working with an invoice database, you may need to have the customer and price list databases available. As you are working with a checkbook database, you may need to access information in vendor and accounts payable databases. When you are working with a class scheduling database, you may also need the course catalog and student registration databases. Panorama can handle all these tasks.

### Opening Multiple Files

You can open multiple files all at once using the desktop (Explorer), or one at a time after Panorama is open. From the desktop, first select the files and then double click on one of them. (To select several files, either hold down the **Shift** key as you click on each file, or drag the mouse around the files.)



Panorama will open all of the databases you have selected.



To open an additional file from inside Panorama, use the **Open File** command in the File Menu. You can continue to open files until you run out of memory. (You are also limited to 64 open windows.)

## File Sets

If you plan to use a group of files together, you can create a special document that represents the entire group. This special document is called a File Set. File Sets have their own unique icon.

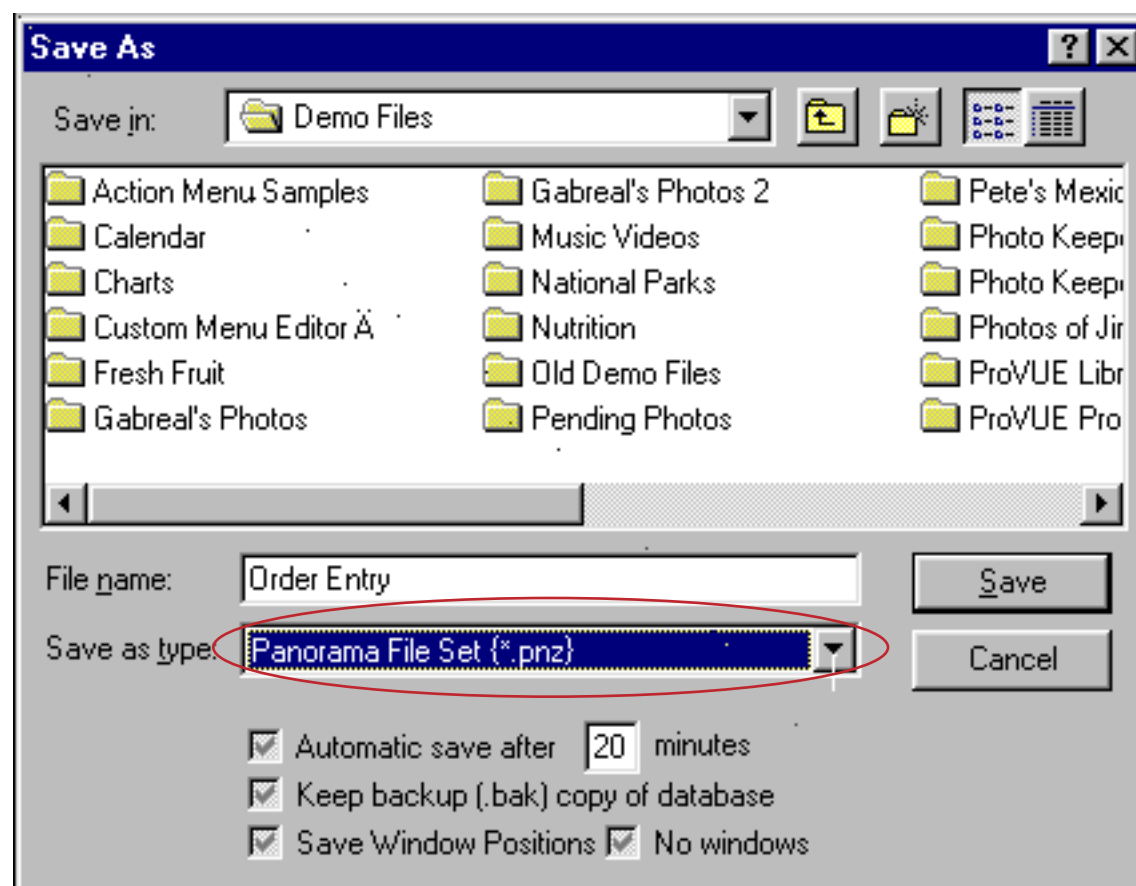


When you double click on a file set icon, Panorama will automatically open all the files included by the set. For example, you could create an **Order Entry** file set that automatically opens three databases—**Invoice**, **Customer List**, and **Price List**. A file set can also be opened with the **Open File** dialog.

To create a file set, first open the databases you want to include in the set. (Make sure that no other files that you don't want included in the set are open. Every open file will be included in the set. If you are not sure what files are currently open you can use the **Memory Usage** command to find out — see “[Monitoring Memory Usage](#)” on page 73.) Then use the **Save As** command in the File menu. Type in the name of the file set. If you are using a Macintosh computer check the **Set** radio button.



If you are using a Windows PC computer use the combo box to choose the **Panorama File Set (\*.pnz)** option



Press the **Save** button to create the file set document.

All the databases in a file set must be in the same folder as the file set itself. The files will be opened in the same order that they were originally opened when you created the set. If you want to make sure that the files open in a certain order, you should open the files manually (with the **Open File** command) in that order before saving the file set.

A file set cannot be edited or changed once it is created. To change a file set, you must use **Save As** to create it over again.

**Tip:** It's important to realize that the file set document does not contain the actual databases themselves—the databases are still in separate files. If you copy the file set to another folder or disk, you also need to copy the actual database files.

**Tip:** You cannot save the window positions associated with a file set. Instead, you must save the window positions of each individual database within the set. See "[Revert to Saved](#)" on page 49 for more information on saving window positions.

### Saving Multiple Files

The **Save** command only saves one file at a time—the file associated with the top window. To save all the open files, use the **Save All** command. (Note: Only databases that have actually been changed will be saved.)

When you use the **Quit** command, Panorama will check each open database to see if it has been modified. Panorama will ask you if you want to save your changes before shutting down Panorama.



## Importing and Exporting Data

Panorama allows you to freely exchange information between it and other applications. The common terms for these exchanges are **importing** and **exporting**.

**Importing** means to transfer information from another program or computer into a Panorama file. The data can then be manipulated using Panorama's menu commands and tools. Panorama's import capabilities allow you to take advantage of databases that have already been keyed in or databases on other computers (for instance, on minicomputers or electronic bulletin boards). See "[Importing a Text File](#)" on page 54.

**Exporting** is the exact opposite of importing. To export data from Panorama means to take data from a Panorama file and make it accessible to another "foreign" program. For example, Panorama data can be exported to Excel so that it can be included in a spreadsheet. See "[Exporting a Text File](#)" on page 63.

In addition to the manual techniques described in this chapter it is also possible to set up a procedure to automatically import or export data. To learn more about this see Chapter 25 of the **Panorama Handbook**.

### Working with Text Files



Panorama cannot directly access information in database or spreadsheet files created by other programs. Exchanging data between Panorama and another program requires an intermediate **text file**. A text file is very basic because it contains just the data—no forms, procedures, graphics, or anything else. Because text files are so simple, they provide a common interchange format for different programs. Virtually all database, spreadsheet, and word processing programs can read and write text files. This makes transferring data between Panorama and another program a two step process. Let's take Excel as an example. To transfer data from Panorama to Excel you first must export the data from Panorama as a text file. Then you go into Excel and import the text file. To transfer data from Excel to Panorama you start by exporting the data from Excel as a text file. Once the text file has been created you can go into Panorama and import the data from the text file.

On PC systems text files often have a three letter filename extension of **.txt**, for example **My Data.txt**. However, text files may use other extensions as well, such as the **.csv** (short for **comma separated values**) file shown above. Panorama can work with text files with any extension.

On Macintosh systems no extension is required. However we recommend adding **.txt** to the end of the filename anyway. This makes it easier to remember what kind of data is in the file and also improves compatibility in case the file is ever transferred to a PC system.

### Importing a Text File

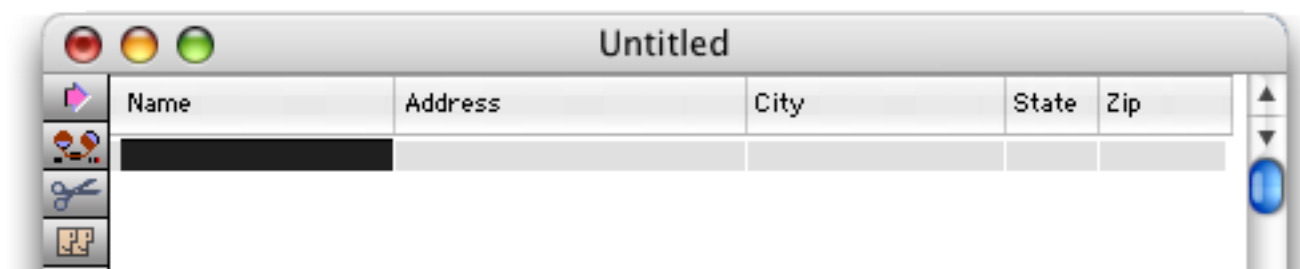
The first step in importing a text file is to prepare the text file. You can create it manually using a text editor like **SimpleText** or **Notepad**, but usually the text file is created by exporting from another database or spreadsheet program.

Macintosh	PC
 Mailing List.txt	 Mailing List.txt      Mailing List Data.csv

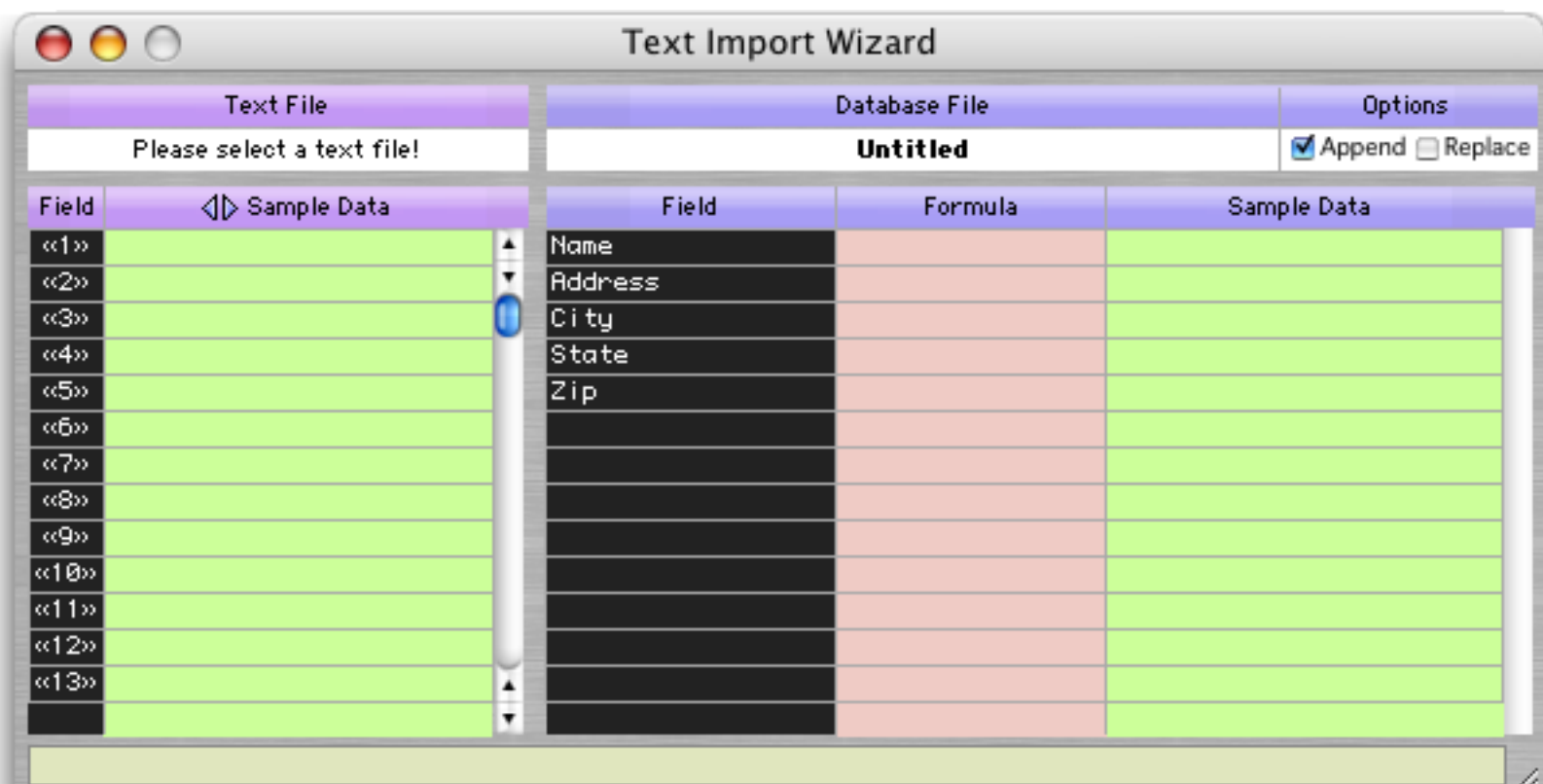


### Using the Text Import Wizard

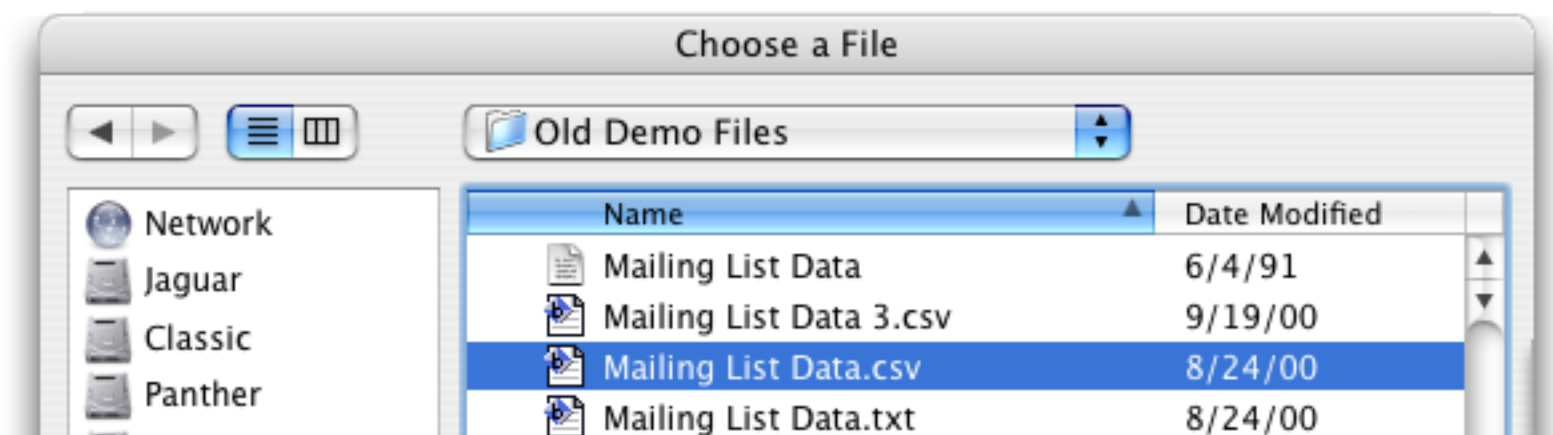
The **Text Import Wizard** makes it easy to import a text file into an existing database, even if the fields in the text file are not in the same order as the database fields. Usually you'll start the process by opening the database you want to import the data into. To illustrate this wizard we'll use a very simple mailing list file.



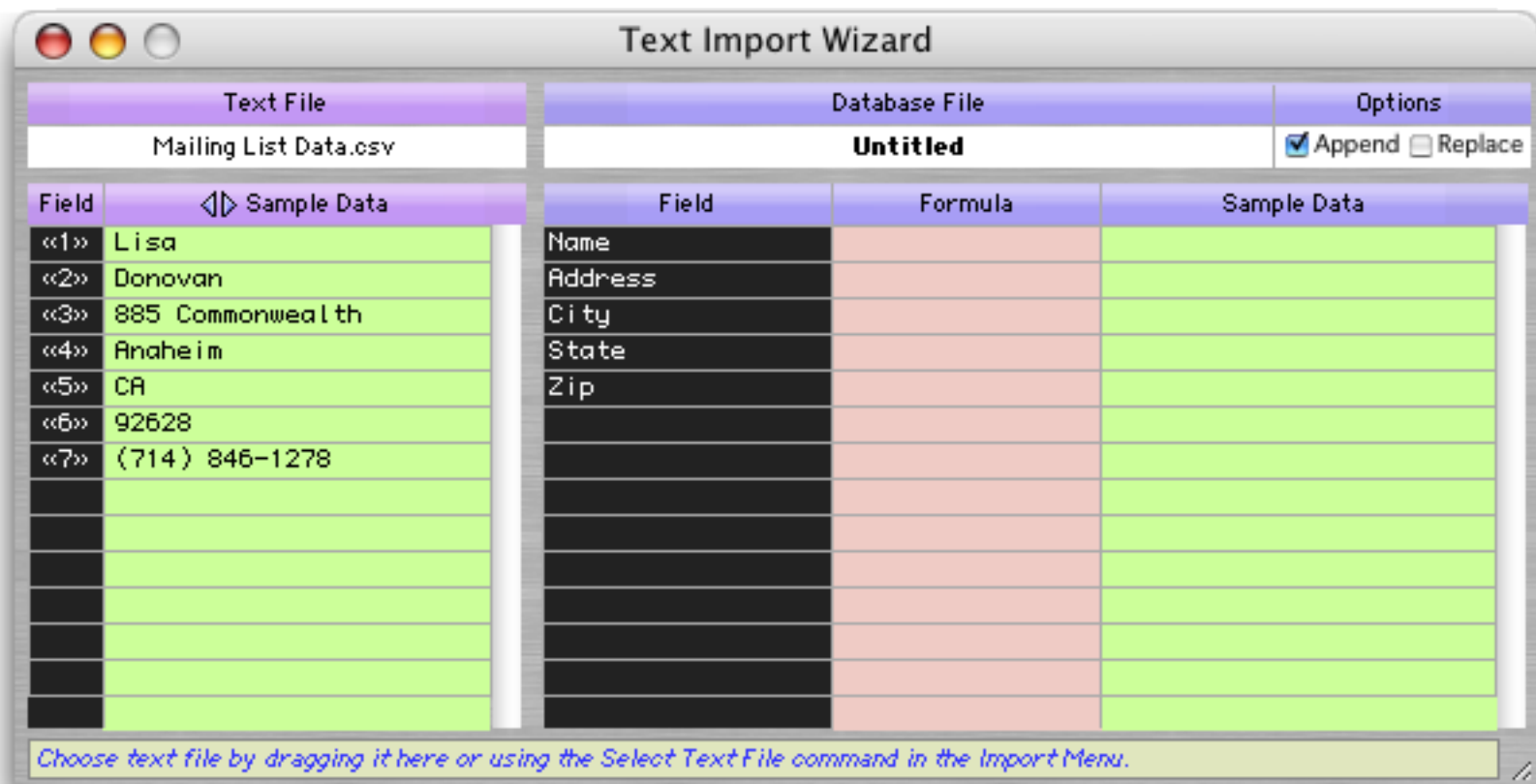
To start the import process choose **Text Import** from the **File** menu (or choose **Text Import Wizard** from **Import-Export** submenu in the **Wizard** menu). As you can see the right hand side of the window shows that we are going to import into the **Mailing List** file, and lists the five fields in this database.



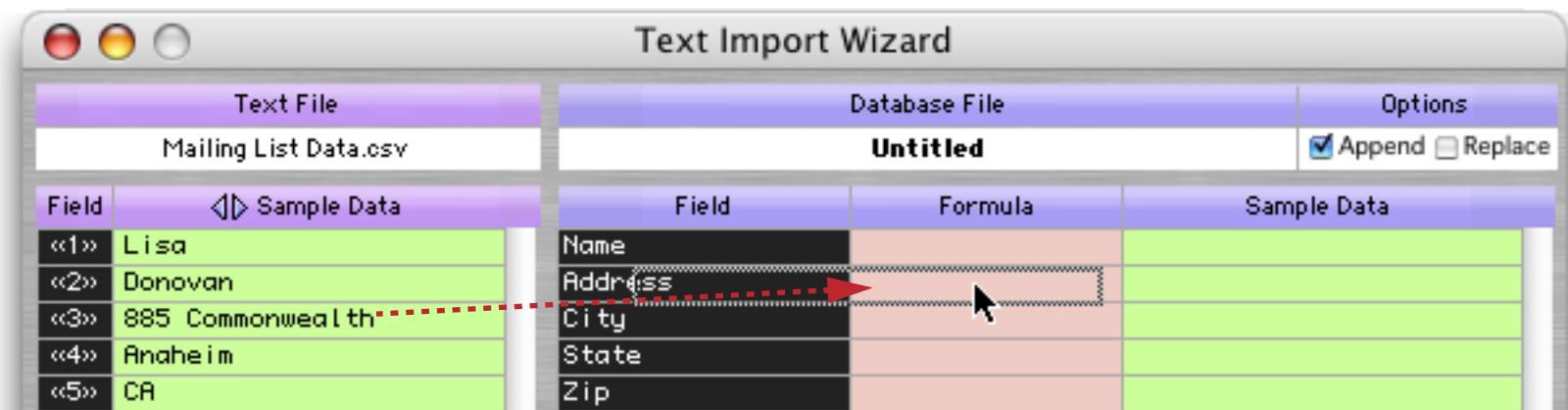
The next step is to select the text file you want to import into this database. You can simply drag the file from the **Finder** onto the wizard or you can choose **Select Text File...** from the **Import** menu and use the dialog to select the text file you want to import.



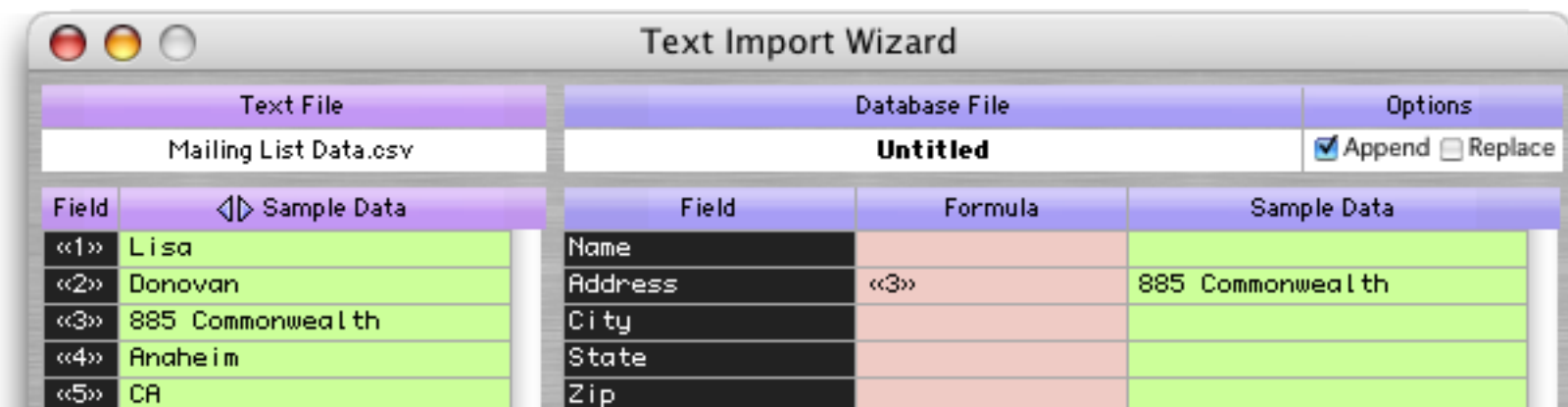
After you have selected the text file the wizard will display the name of the file (in this example [Mailing List Data.csv](#)) and contents of the first line.



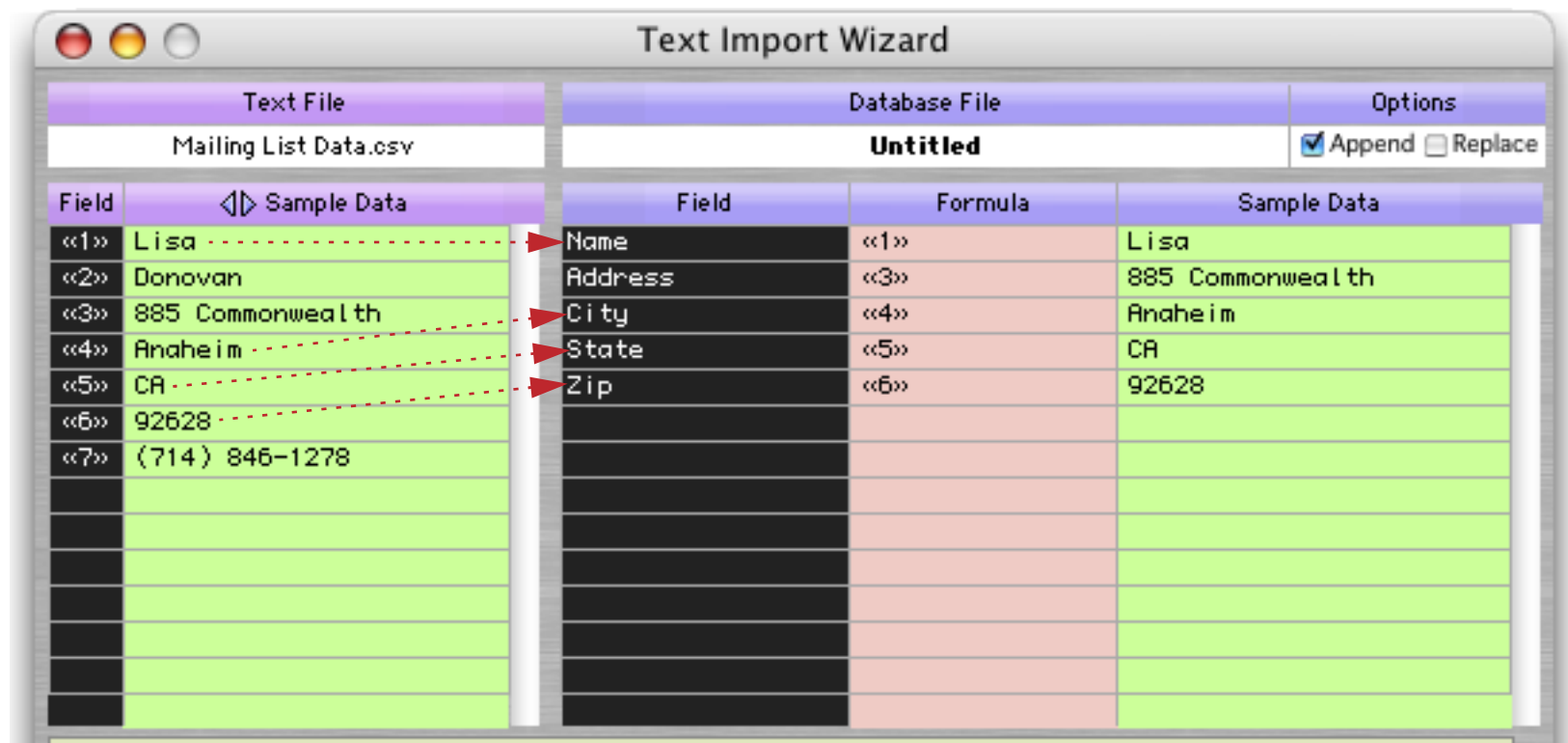
To set up the import configuration you'll need to drag from the left hand side onto the right hand side.



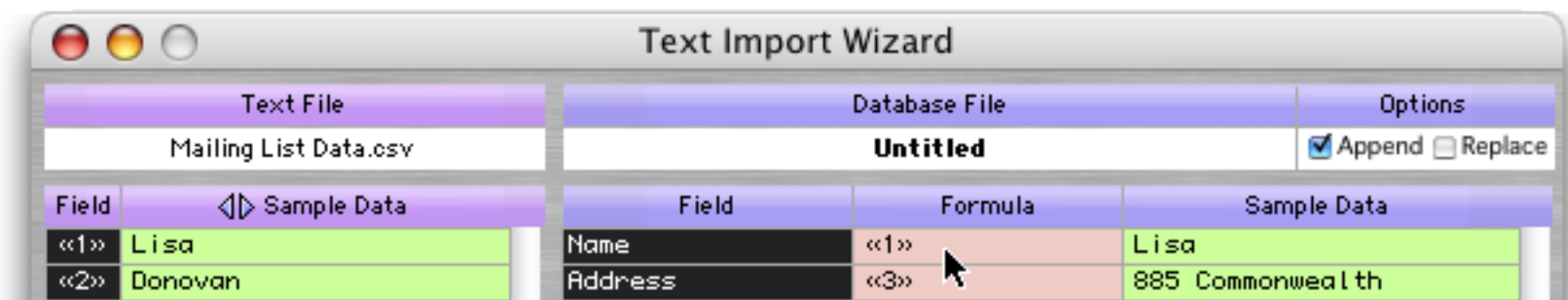
When you release the mouse the wizard will update the import configuration.



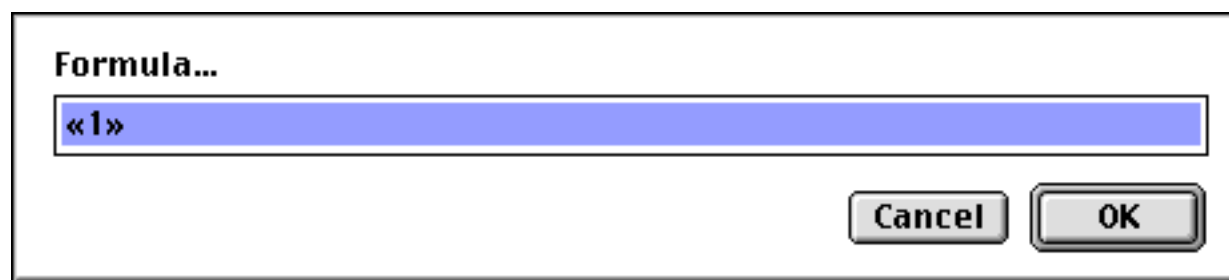
You'll need to drag each field you want to import across from the left to the right.



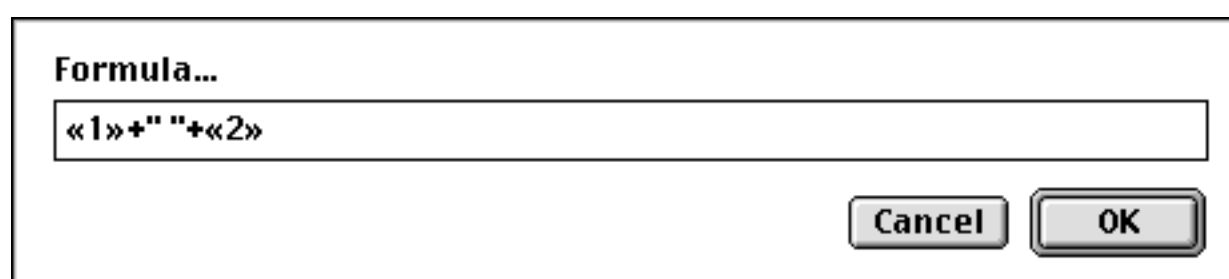
Sometimes the data you want to import doesn't match the fields in the database. In this case the database has only a single **Name** field, but the import data contains separate first and last names. Somehow these separate fields will need to be combined as the data is imported. The import wizard allows you to do this with a **formula** (see "[Formulas](#)" on page 501). To edit the formula for a field click anywhere on the field's line.



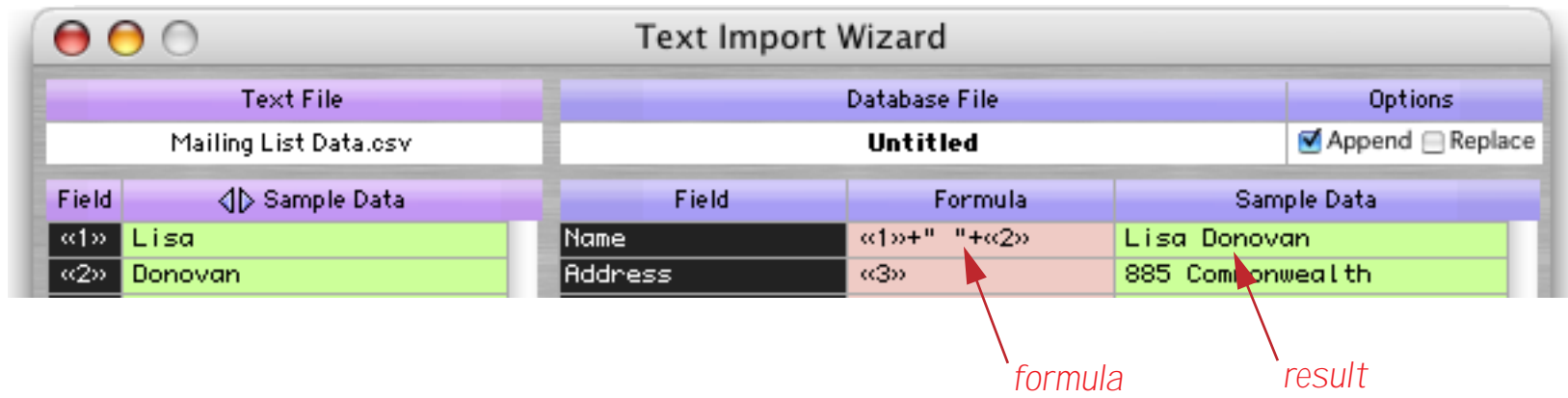
A dialog appears allowing you to edit the formula.



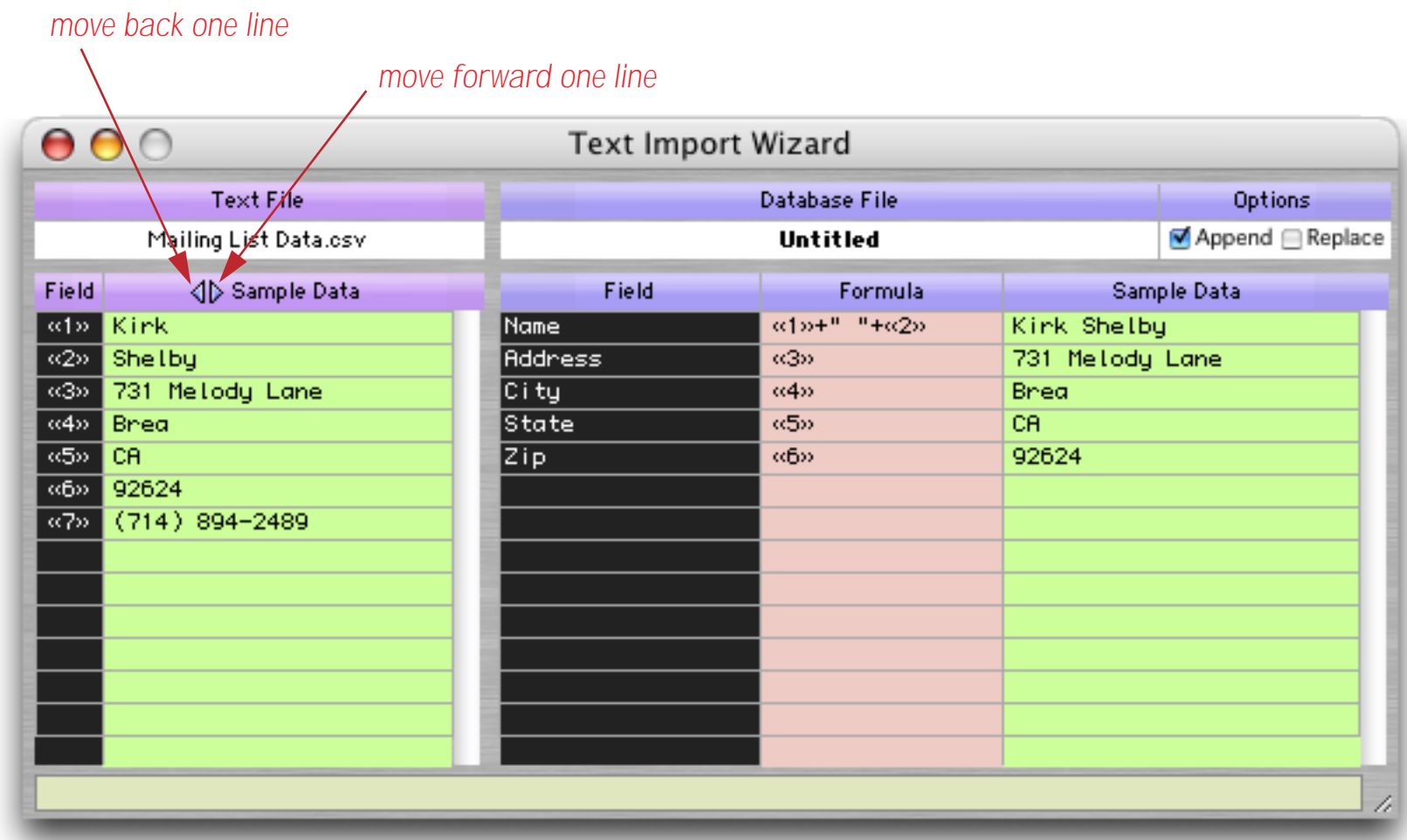
Within this formula you can include any import field by typing the field number in between « and » characters (see "[Special Characters](#)" on page 525). To "glue" two text items together you can use the + symbol. To include constant text in the formula put the text inside quotes. The illustration below shows a formula that combines the first and last name into a single field (with a space in between).



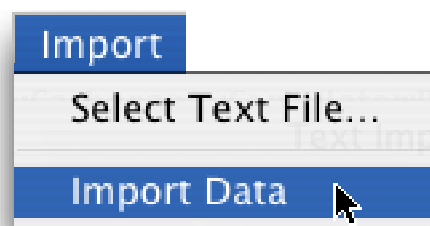
Press the **OK** button to preview the result of this formula.



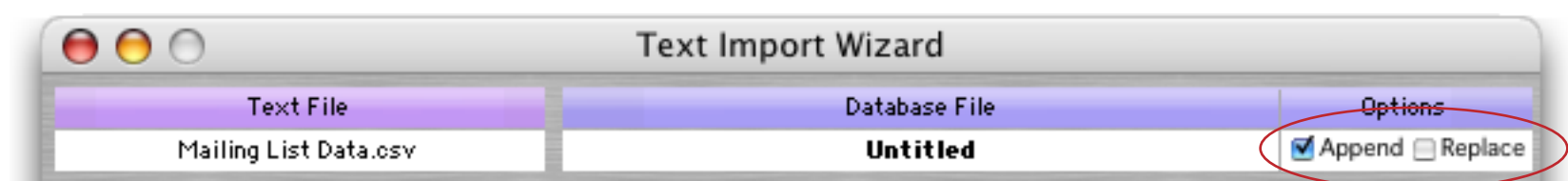
If you want to you can use the small arrows to preview additional lines in the imported data (not just the first line).



When all the fields you want to import are set up choose **Import Data** from the Import menu inside the window.



(Before you do you should make sure that the **Append** or **Replace** option you want is selected. **Append** will append the new data to whatever data is already in the database, while **Replace** will erase and replace the existing data.)



The wizard imports the data into the original database.

Name	Address	City	State	Zip
Lisa Donovan	885 Commonwealth	Anaheim	CA	92628
Kirk Shelby	731 Melody Lane	Brea	CA	92624
Bob Muscolo	624 Fountain Street	Costa Mesa	CA	92604
Jim Reynolds	1183 Brookhurst	Costa Mesa	CA	92608
Jack Rutan	4910 Glendale	Costa Mesa	CA	92642
Chris Murphy	906 Springdale	Diamond Bar	CA	92618
Mike Johnson	391 E. Raymond	Fullerton	CA	92625
Russ Greene	1099 E. Dorothy Lane	Fullerton	CA	92625
Scott Lutz	448 Longview	Fullerton	CA	92631
Mary Matthews	891 E. Graham	Huntington Beach	CA	92649
Joy Scott	7780 N. Harbor	Newport Beach	CA	92640
Fred Claire	341 Pinecrest	Orange	CA	92450

### Common Import Formulas

Using a formula you can combine import fields, split import fields, convert to upper case or lower case, and much much more. You've already learned how to combine two or more import fields together with the **+** symbol like this (see "[Gluing Strings Together](#)" on page 531).

```
<1>+" "+<2>
```

```
<2>+" , "+<1>
```

To pick out a single word (for example a first or last name) you can use the **array()** function (see "[Text Arrays](#)" on page 539). Here is a formula for picking the first name from a combined name field (assumed to be the first field, **<1>**, and using the format **First Last**, for example **John Smith**).

```
array(<1>,1," ")
```

This formula extracts the last name.

```
array(<1>,2," ")
```

To convert text to upper case use the **upper()** function (see "[String Modification Functions](#)" on page 535). This formula extracts the last name and converts it to upper case.

```
upper(array(<1>,2," "))
```

To extract only a limited number of characters use a **text funnel** (see "[Taking Strings Apart \(Text Funnels\)](#)" on page 533). This formula extracts the first five characters from a zip code.

```
<6>[1,5]
```



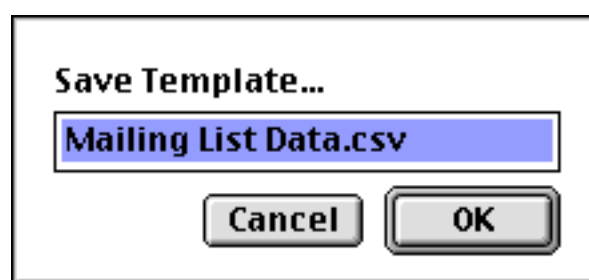
Here is an example that uses several of these techniques.



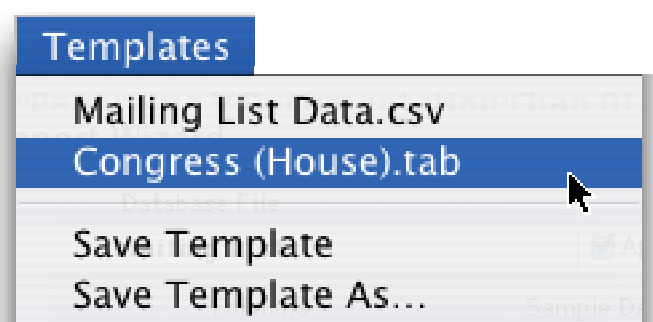
To learn more about Panorama formulas see Chapter 23 of the [Panorama Handbook](#).

### Import Templates

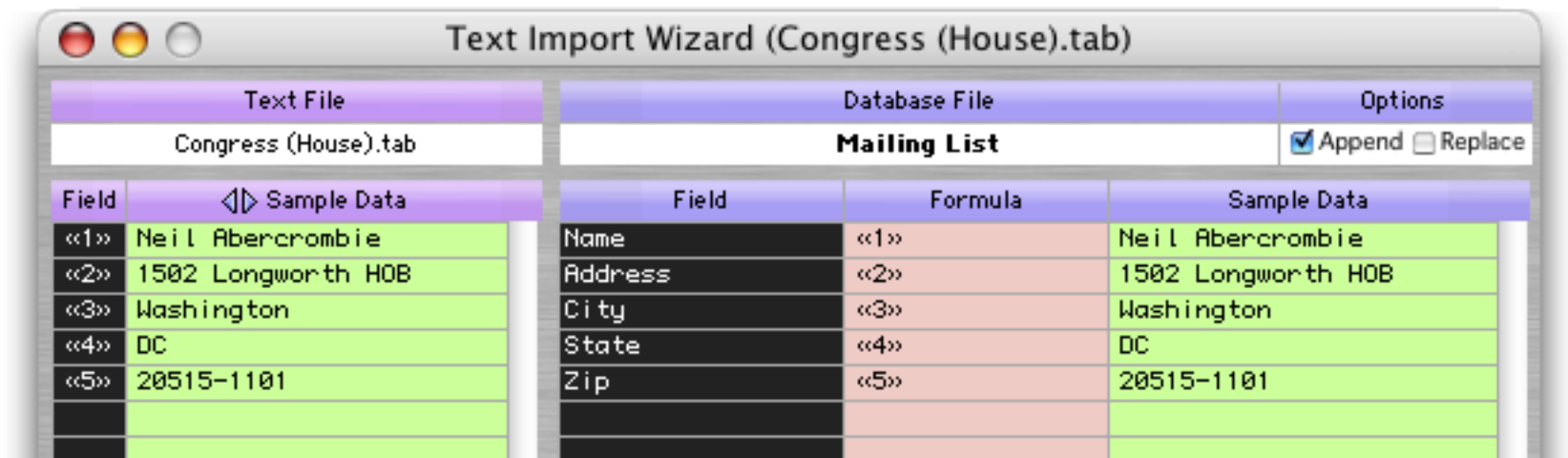
If you think you'll need to use an import configuration more than once you can save it as a **template**. The first step is to set up the configuration (as described in the previous section). Once the configuration is set up you can save it with the **Save Template** or **Save Template As...** commands in the Template menu. The wizard will prompt you to type in a name for the new template (the default is the name of the text file being imported).



Once a template has been saved you can open it again by selecting it from the **Template** menu. (Note: The template is actually stored in the database being imported into (in this case [Mailing List](#)). The template is only available when that database is being imported into. Each database may contain its own separate set of templates, which makes sense since the import configuration used with one database is not likely to work with any other database.)



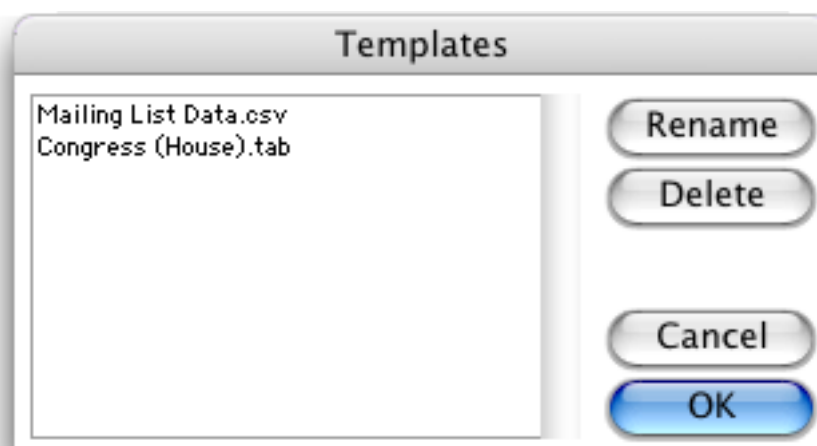
The wizard loads the entire import configuration, ready to go.



You can use the configuration as is or modify it before you actually import the data.

Name	Address	City	State	Zip
Neil Abercrombie	1502 Longworth HOB	Washington	DC	20515-1101
Gary Ackerman	2243 Rayburn HOB	Washington	DC	20515-3205
Robert Aderholt	1007 Longworth HOB	Washington	DC	20515-0104
Thomas Allen	1717 Longworth HOB	Washington	DC	20515-1901
Robert Andrews	2439 Rayburn HOB	Washington	DC	20515-3001
Bill Archer	1236 Longworth HOB	Washington	DC	20515-4307
Richard Arme	301 Cannon HOB	Washington	DC	20515-4326
Joe Baca	2300 Rayburn HOB	Washington	DC	20515-0542
Spencer Bachus	442 Cannon HOB	Washington	DC	20515-0106
Brian Baird	1721 Longworth HOB	Washington	DC	20515-4703
Richard Baker	434 Cannon HOB	Washington	DC	20515-1806
John Elias Balda	1740 Longworth HOB	Washington	DC	20515-1902

If you want to delete or rename a template choose the **Rename/Delete Templates...** command from the Template menu. This opens the dialog shown below. To rename a template first click on it and then press the **Rename** button. A dialog appears allowing you to type in a new name. To delete a template press the **Delete** button.



When you are done press the **OK** button.

#### Choosing a Database to Import Into

The **Text Import Wizard** normally imports into database that was active when the wizard was opened. However, you can use the **Database** menu to choose to import into any open database. Simply choose the database you want to import into and then set up the configuration.

### Converting an Import Configuration into a Procedure

To convert the current import configuration into a procedure choose the **Copy Procedure to Clipboard** command from the Special menu. Now create a procedure (see “[Writing a Procedure from Scratch](#)” on page 583) and **Paste** the automatically generated procedure into it.



For more information on creating and using procedures see “[Procedures](#)” on page 572.

### Importing Data from Microsoft Excel

Panorama can import data directly from Microsoft Excel. To learn more about this feature see “[Excel Wizard](#)” on page 47 of the **Wizards & Demos** PDF file.

### Importing Financial Data (QIF, OFX, QFX)

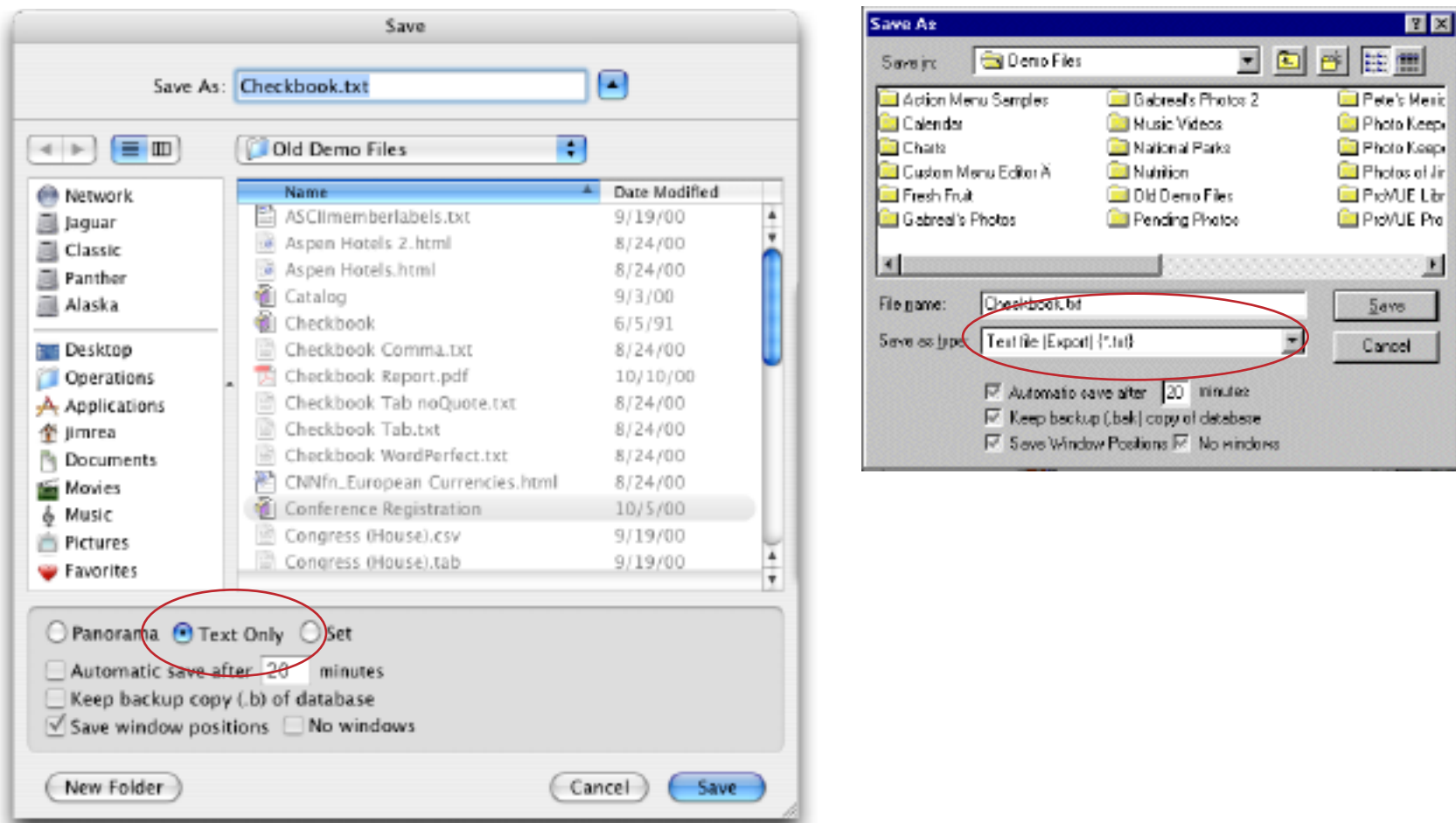
Panorama can import data directly from financial applications and from data downloaded from financial institution web sites. To learn more about this feature see “[Financial Data Wizard](#)” on page 55 of the **Wizards & Demos** PDF file.

### Importing VCard Data

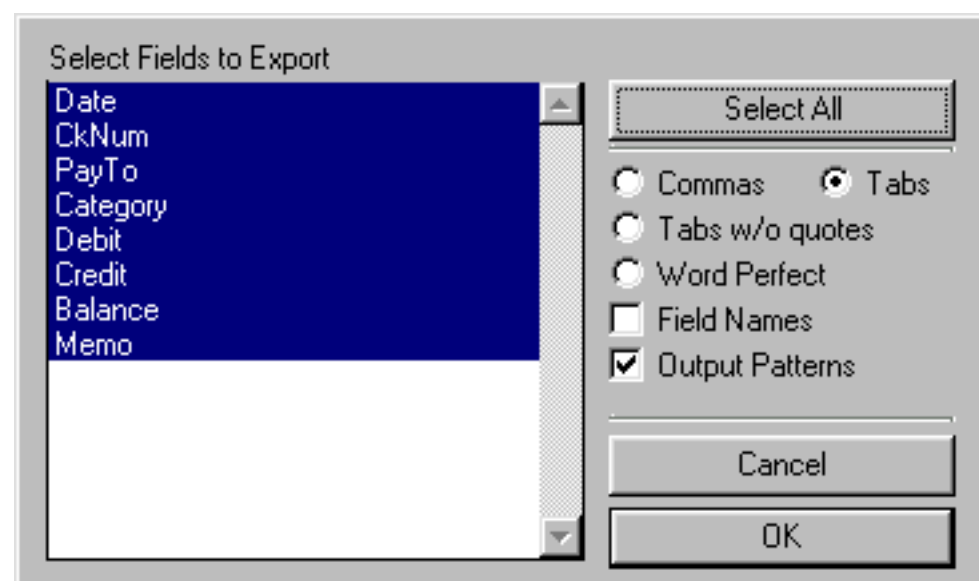
Panorama can import VCard data files from applications that support VCards. To learn more about this feature see “[Using Generic Fields with the VCard Wizard](#)” on page 242 of the **Panorama Handbook**.

## Exporting a Text File

To export the selected records (see “[Finding vs. Selecting](#)” on page 159) in the current database into a text file, use the **Save As** command in the File menu. The standard save dialog will appear on the screen. If you are using a Macintosh computer select the folder and name of the text file and click the **Text Only** radio button. If you are using a Windows PC computer use the combo box to choose the **Text file (Export) (\*.txt)** option. (The name must be different from the database name.)



Once the name and text only option are set up, click the **Save** button. This opens a second dialog that allows you to specify which fields to include in the exported data and what format to use.



Only the fields you select will be included in the text file. To select a field, click the mouse on the field name. To select additional fields, click on them also. To un-select a field, click on the field again. If you want to export all the fields in the database, you can quickly select them all with the **Select All** button.

The four radio buttons—**Commas**, **Tabs**, **Tabs w/o quotes**, and **Word Perfect**—let you pick how the text file is formatted.

Option	Description	Example
Commas	Exported file will have commas between each field and a carriage return at the end of each line. If a data cell contains a comma, it will be surrounded by quotes (").	01/08/90,1907,Northern Illinois Mold,96.05 01/08/90,1908,U S Postmaster,75.00 01/08/90,1909,"Advertiser's Mailing Service, Inc.",390.80 01/16/90,1910,"Coudert Brothers, Attomey/s At Law",223.52 01/16/90,1911,Paramount Stationers,105.84 01/17/90,1912,California Capitol,36.00 01/17/90,1913,California Capitol,28.00 01/17/90,1914,U S Postmaster,75.00 01/17/90,1915,Sacramento Bee,795.00 01/22/90,1916,Walthers,"12,463.00" 01/22/90,1917,Blue Cross Of Calif,279.03
Tabs	Exported file will have an invisible tab character between each field and a carriage return (Mac) or carriage-return/line feed (Windows) at the end of each line. If a data cell contains a comma, it will be surrounded by quotes (").	01/08/90 1907 Northern Illinois Mold 96.05 01/08/90 1908 U S Postmaster 75.00 01/08/90 1909 "Advertiser's Mailing Service, Inc." 390.80 01/16/90 1910 "Coudert Brothers, Attomey/s At Law" 223.52 01/16/90 1911 Paramount Stationers 105.84 01/17/90 1912 California Capitol 36.00 01/17/90 1913 California Capitol 28.00 01/17/90 1914 U S Postmaster 75.00 01/17/90 1915 Sacramento Bee 795.00 01/22/90 1916 Walthers "12,463.00" 01/22/90 1917 Blue Cross Of Calif 279.03
Tabs w/o quotes	Exported file will have an invisible tab character between each field and a carriage return (Mac) or carriage-return/line feed (Windows) at the end of each line. No quotes are added even if the data contains commas.	01/08/90 1907 Northern Illinois Mold 96.05 01/08/90 1908 U S Postmaster 75.00 01/08/90 1909 Advertiser's Mailing Service, Inc. 390.80 01/16/90 1910 Coudert Brothers, Attomey/s At Law 223.52 01/16/90 1911 Paramount Stationers 105.84 01/17/90 1912 California Capitol 36.00 01/17/90 1913 California Capitol 28.00 01/17/90 1914 U S Postmaster 75.00 01/17/90 1915 Sacramento Bee 795.00 01/22/90 1916 Walthers 12,463.00 01/22/90 1917 Blue Cross Of Calif 279.03
Word Perfect	Exported file will use Word Perfect's unique mail merge format, which requires a Control-R/Carriage Return between each field and a Control-E/Carriage Return between each record.	

The **Field Names** checkbox lets you choose whether you want the first record of the text file to list the field names. If this button is checked, an extra line will be added to the top of the text file. This extra line will contain the names of each field.

Date,Num,Pay To,Debit  
01/08/90,1907,Northern Illinois Mold,96.05  
01/08/90,1908,U S Postmaster,75.00  
01/08/90,1909,"Advertiser's Mailing Service, Inc.",390.80  
01/16/90,1910,"Coudert Brothers, Attomey/s At Law",223.52  
01/16/90,1911,Paramount Stationers,105.84  
01/17/90,1912,California Capitol,36.00  
01/17/90,1913,California Capitol,28.00  
01/17/90,1914,U S Postmaster,75.00  
01/17/90,1915,Sacramento Bee,795.00  
01/22/90,1916,Walthers,"12,463.00"  
01/22/90,1917,Blue Cross Of Calif,279.03

The **Output Patterns** checkbox controls whether the data is exported using the output patterns set up for each field (see "[Numeric Output Patterns](#)" on page 256 and "[Date Output Patterns](#)" on page 261), or in the standard numeric and date formats. Checking this box will cause numeric and date columns to be exported in using the output patterns you have set up for each field.

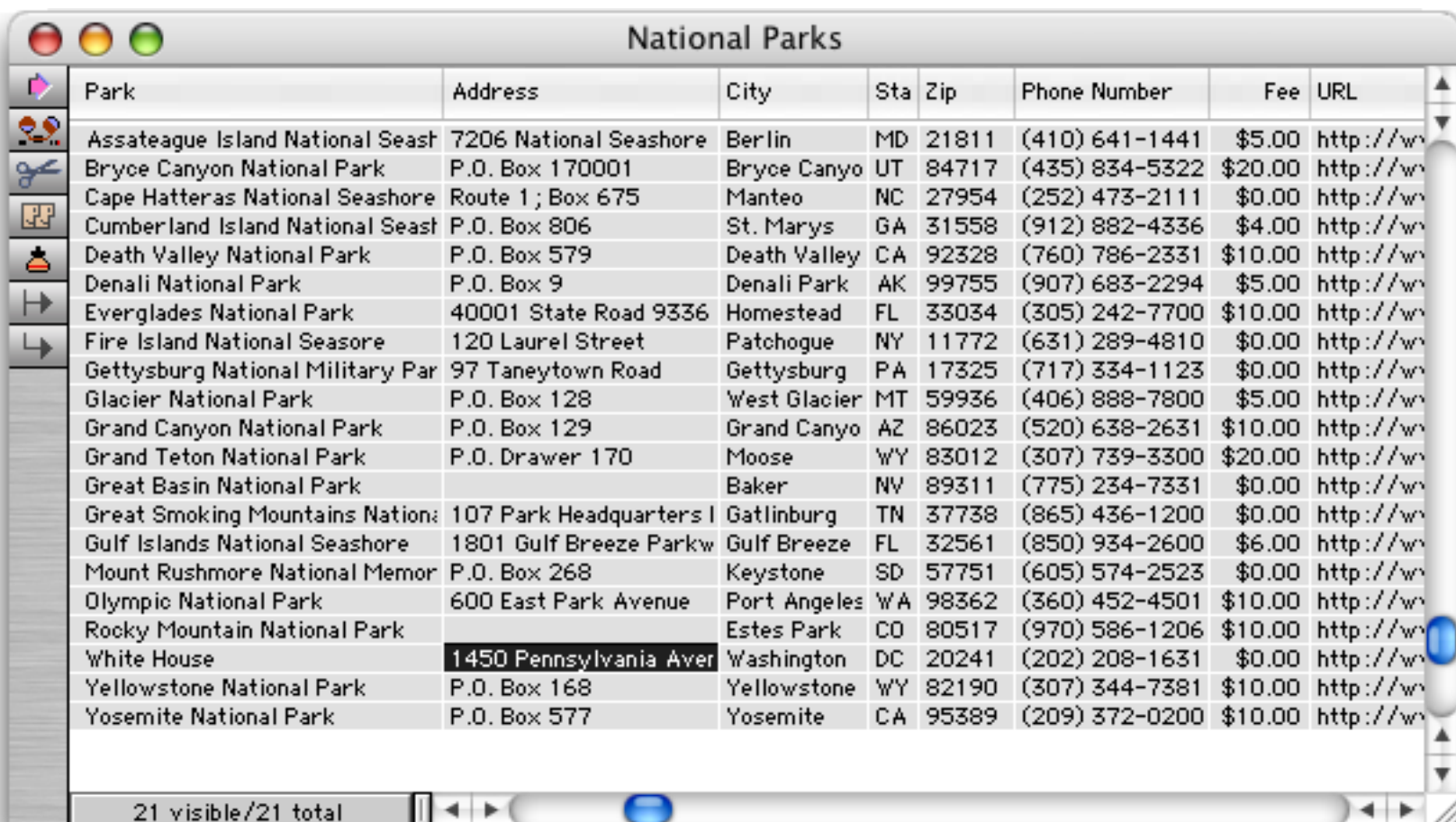


Once you have selected the fields you want exported and checked the appropriate buttons and boxes, pressing the **OK** button will export the data to a text file on the disk.

The **Save As** command exports only the selected records in the database. If you don't want to export the entire database, use the **Find/Select** command to choose just the records you want (see "[Select](#)" on page 341) before using the **Save As** command. Only the selected records will be exported.

### Exporting with the Text Export Wizard

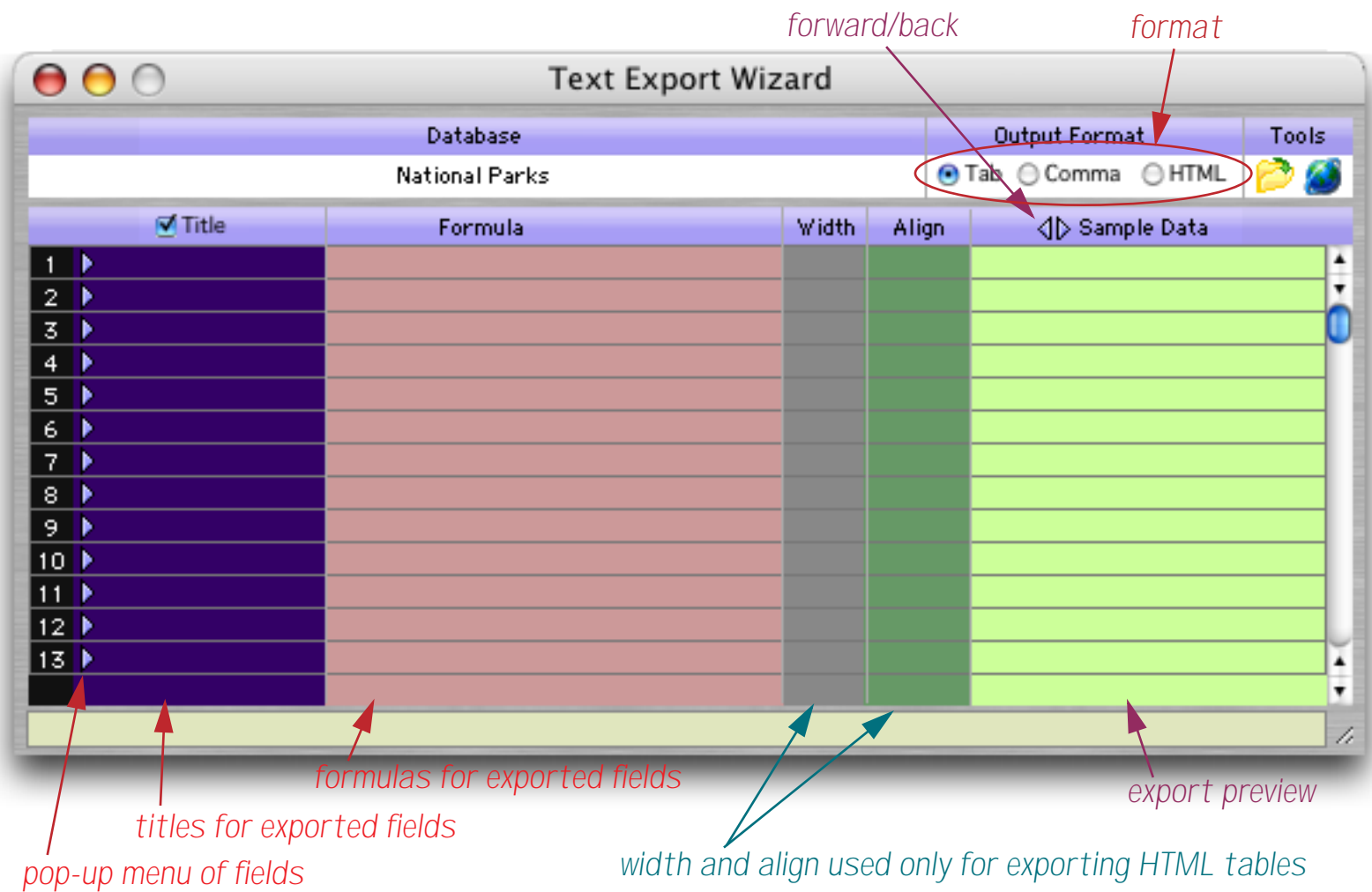
The **Save As** dialog gives you limited control over the format of the text you are exporting. For more control you can use the **Text Export Wizard**. This wizard allows you to specify the order of the fields being exported, and to manipulate the data as it is being exported (converting it to upper case, for example, or combining several database fields into one export field). The wizard can even be used to convert the database into an HTML table. To illustrate this wizard we will use this database of national parks.



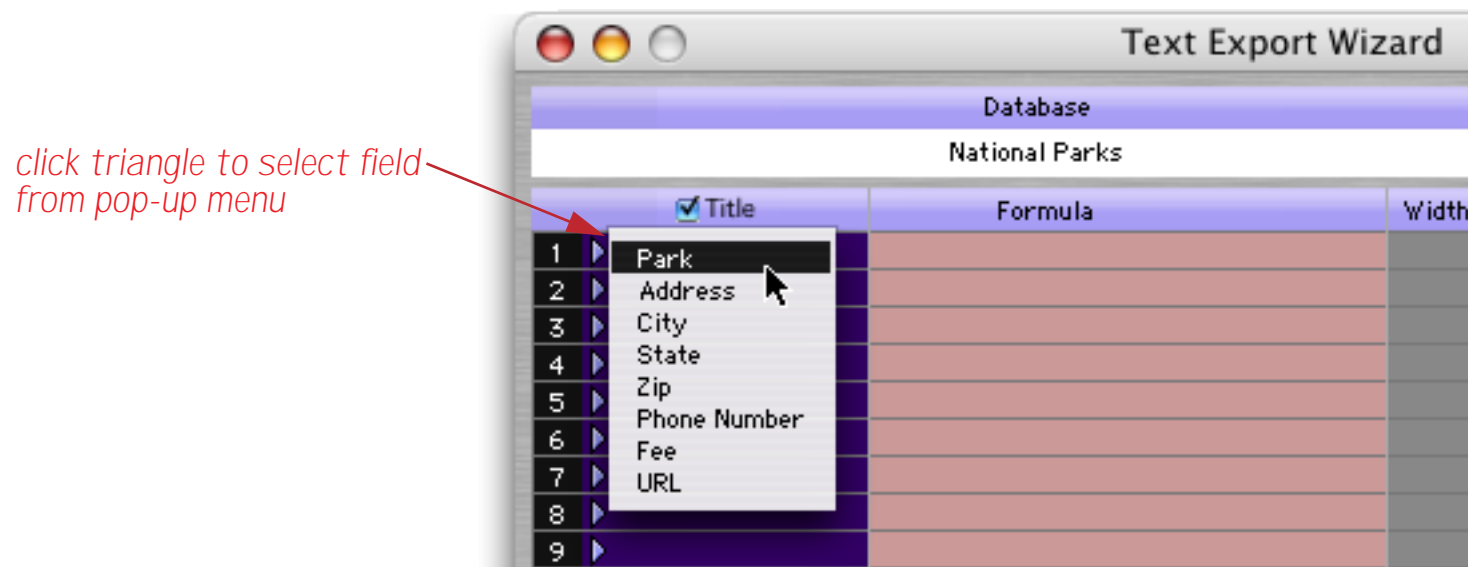
Park	Address	City	Sta	Zip	Phone Number	Fee	URL
Assateague Island National Seashore	7206 National Seashore	Berlin	MD	21811	(410) 641-1441	\$5.00	http://www...
Bryce Canyon National Park	P.O. Box 170001	Bryce Canyon	UT	84717	(435) 834-5322	\$20.00	http://www...
Cape Hatteras National Seashore	Route 1 ; Box 675	Manteo	NC	27954	(252) 473-2111	\$0.00	http://www...
Cumberland Island National Seashore	P.O. Box 806	St. Marys	GA	31558	(912) 882-4336	\$4.00	http://www...
Death Valley National Park	P.O. Box 579	Death Valley	CA	92328	(760) 786-2331	\$10.00	http://www...
Denali National Park	P.O. Box 9	Denali Park	AK	99755	(907) 683-2294	\$5.00	http://www...
Everglades National Park	40001 State Road 9336	Homestead	FL	33034	(305) 242-7700	\$10.00	http://www...
Fire Island National Seashore	120 Laurel Street	Patchogue	NY	11772	(631) 289-4810	\$0.00	http://www...
Gettysburg National Military Park	97 Taneytown Road	Gettysburg	PA	17325	(717) 334-1123	\$0.00	http://www...
Glacier National Park	P.O. Box 128	West Glacier	MT	59936	(406) 888-7800	\$5.00	http://www...
Grand Canyon National Park	P.O. Box 129	Grand Canyon	AZ	86023	(520) 638-2631	\$10.00	http://www...
Grand Teton National Park	P.O. Drawer 170	Moose	WY	83012	(307) 739-3300	\$20.00	http://www...
Great Basin National Park		Baker	NV	89311	(775) 234-7331	\$0.00	http://www...
Great Smoky Mountains National Park	107 Park Headquarters I	Gatlinburg	TN	37738	(865) 436-1200	\$0.00	http://www...
Gulf Islands National Seashore	1801 Gulf Breeze Parkway	Gulf Breeze	FL	32561	(850) 934-2600	\$6.00	http://www...
Mount Rushmore National Memorial	P.O. Box 268	Keystone	SD	57751	(605) 574-2523	\$0.00	http://www...
Olympic National Park	600 East Park Avenue	Port Angeles	WA	98362	(360) 452-4501	\$10.00	http://www...
Rocky Mountain National Park		Estes Park	CO	80517	(970) 586-1206	\$10.00	http://www...
White House	1450 Pennsylvania Avenue	Washington	DC	20241	(202) 208-1631	\$0.00	http://www...
Yellowstone National Park	P.O. Box 168	Yellowstone	WY	82190	(307) 344-7381	\$10.00	http://www...
Yosemite National Park	P.O. Box 577	Yosemite	CA	95389	(209) 372-0200	\$10.00	http://www...

21 visible / 21 total

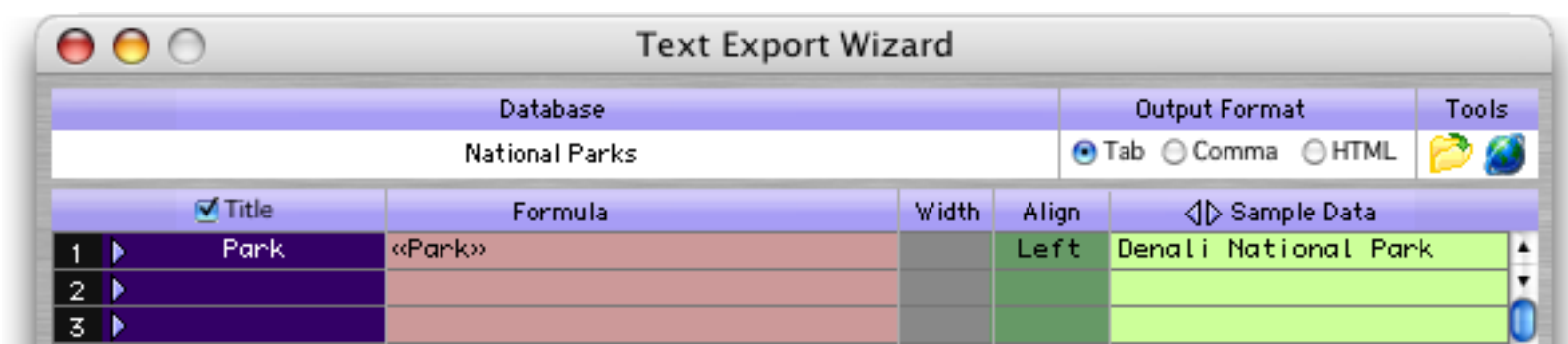
To begin the export process choose **Text Export** from the **File** menu (or choose **Text Export Wizard** from the **Import-Export** submenu of the **Wizard** menu).



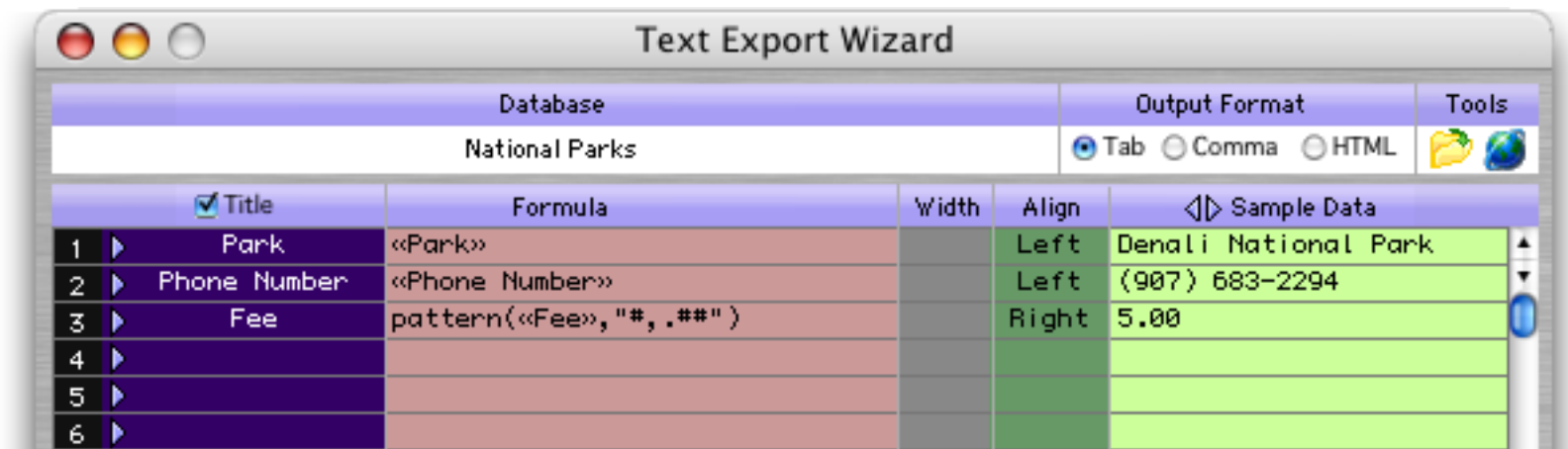
To set up a field to be exported simply choose that field using the pop-up menu




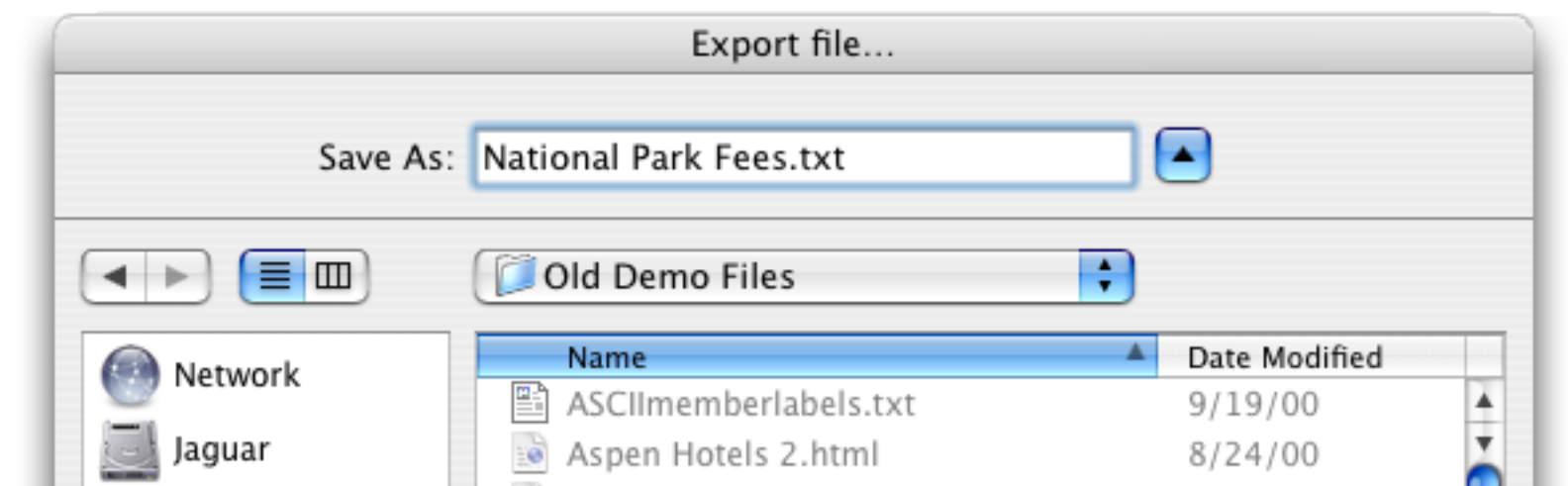
When you make your choice the wizard will fill in one line of the configuration.



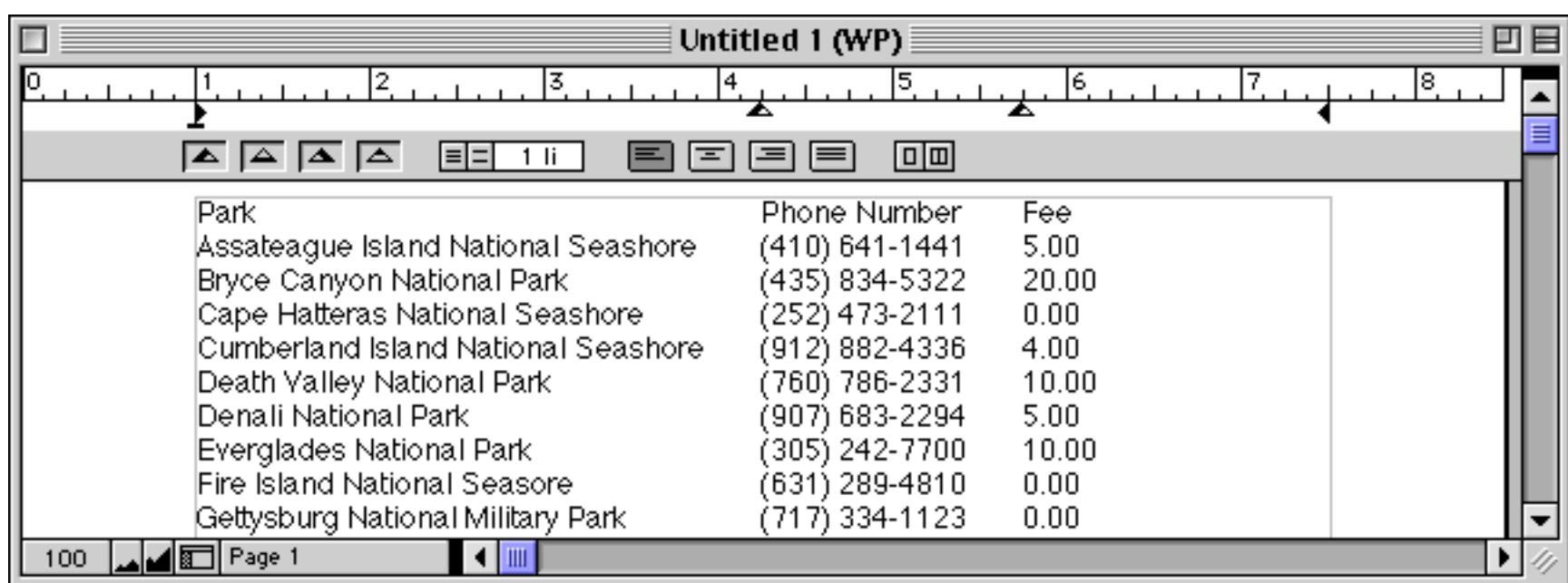
Repeat the process to select all of the fields you want to export. If a numeric field is selected the wizard will use the `pattern()` function (see “[Converting Between Numbers and Strings](#)” on page 538) to convert the number into text, as shown below. If a date field is selected the `datepattern()` function will be used to perform the conversion (see “[Converting Between Dates and Text](#)” on page 545).



Once the configuration is set up select the **Export Text File** command in the Export menu to actually export the data. (You can also click on the  export icon in the Tools box.) This command will ask you to select a folder for the exported file, and to type in a file name.



When you press the **Save** button the file will be exported. Here's what the resulting text file looks like when it is opened with a word processor (in this case ClarisWorks™). We've set the tab stops to make it easier to see how the data has been exported as separate, tab separated columns.

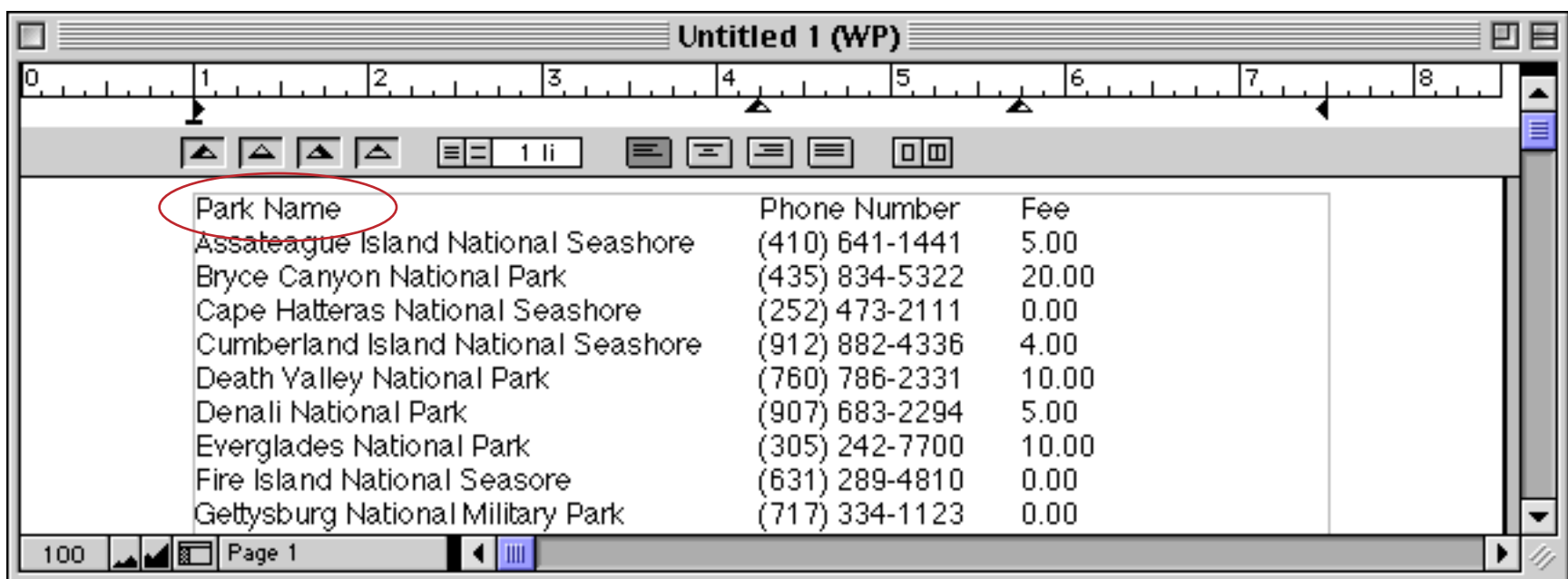


The first line of this exported file contains the title of each column (**Park**, **Phone Number**, **Fee**). You can edit a title simply by clicking on it.

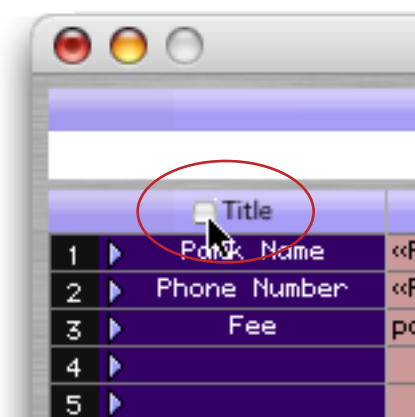


*double click any title to edit it*

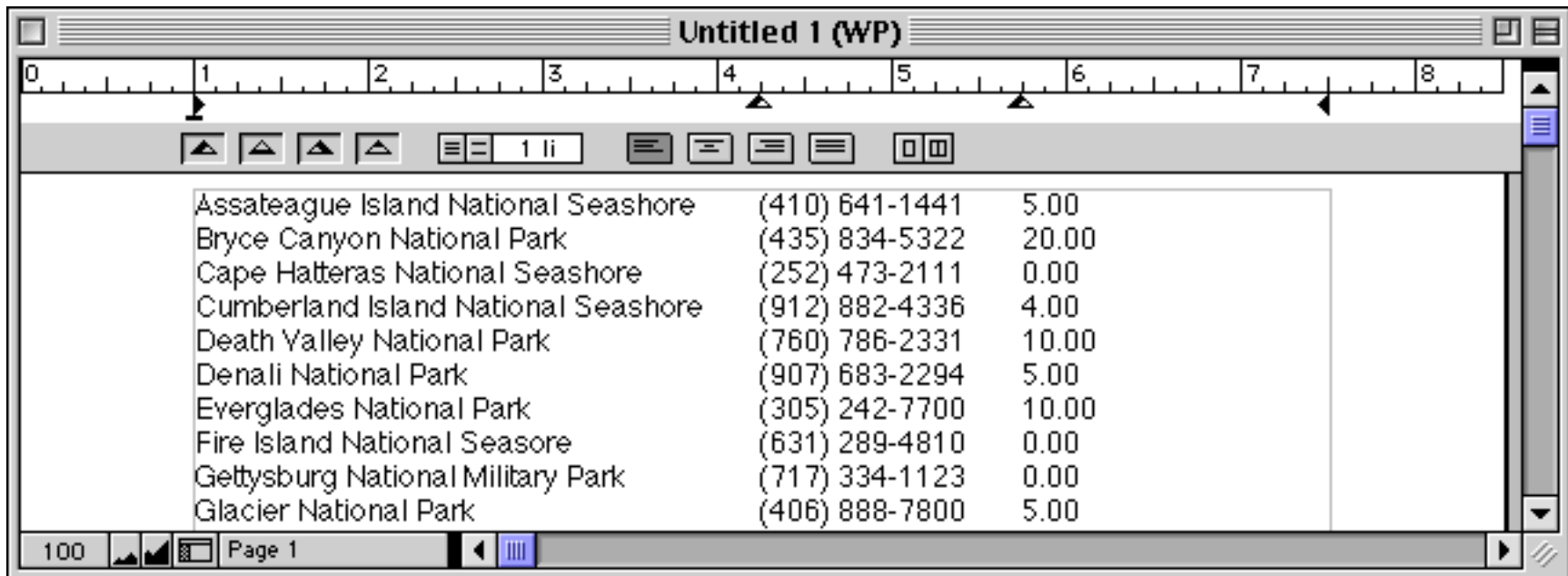
When you are done press the **Enter** key. If you re-export the data and examine it you will see the new title.



If you want to leave out the first line of titles simply un-check the **Title** option.



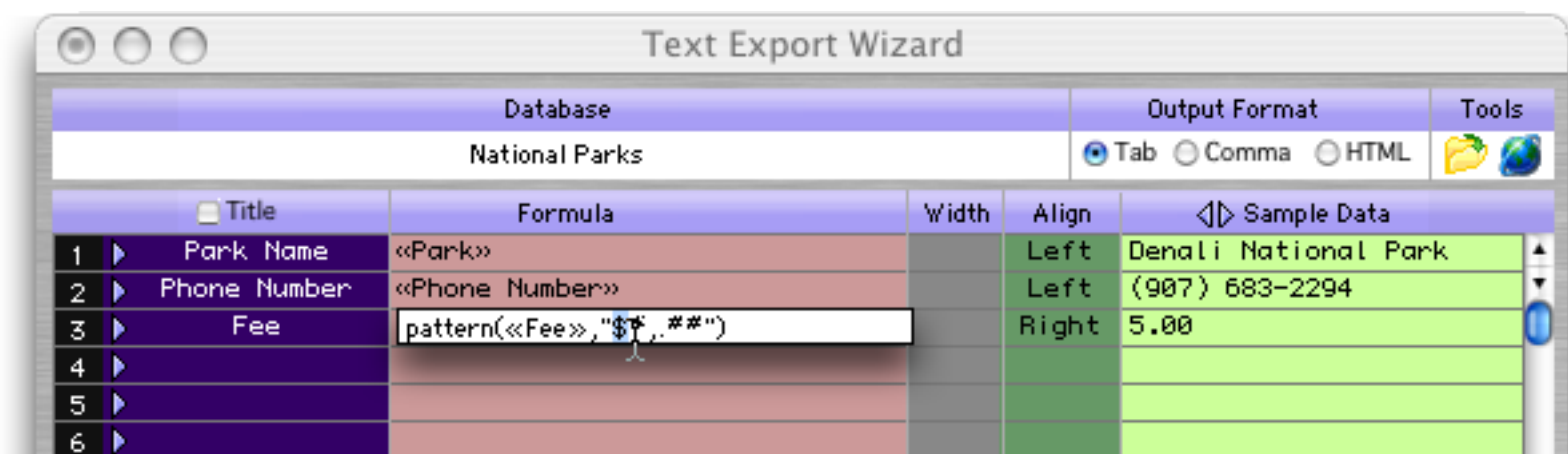
With this option turned off the exported data starts right on the first line, with no titles at all.



Assateague Island National Seashore	(410) 641-1441	5.00
Bryce Canyon National Park	(435) 834-5322	20.00
Cape Hatteras National Seashore	(252) 473-2111	0.00
Cumberland Island National Seashore	(912) 882-4336	4.00
Death Valley National Park	(760) 786-2331	10.00
Denali National Park	(907) 683-2294	5.00
Everglades National Park	(305) 242-7700	10.00
Fire Island National Seashore	(631) 289-4810	0.00
Gettysburg National Military Park	(717) 334-1123	0.00
Glacier National Park	(406) 888-7800	5.00

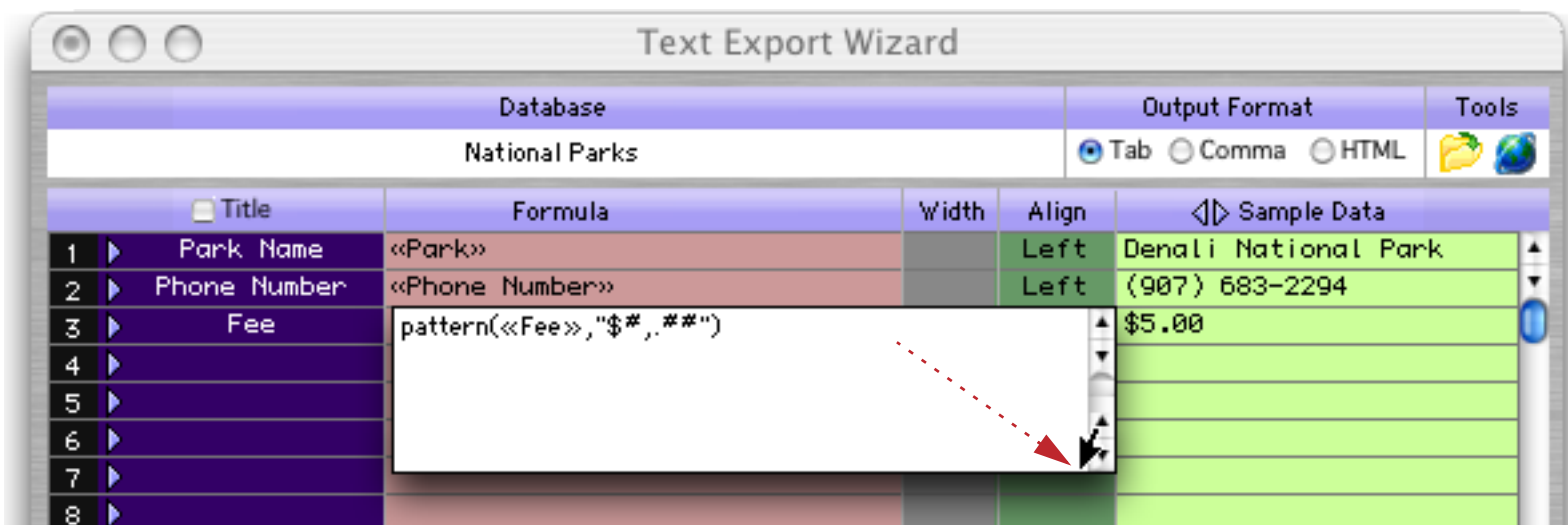
### Editing the Export Configuration

To edit the title, formula, width or alignment simply click on the item you want to edit. A pop-up editing window will appear (see “[The Input Box](#)” on page 276).



Database		Output Format		Tools
National Parks		<input checked="" type="radio"/> Tab <input type="radio"/> Comma <input type="radio"/> HTML		
Title	Formula	Width	Align	Sample Data
1 Park Name	<<Park>>		Left	Denali National Park
2 Phone Number	<<Phone Number>>		Left	(907) 683-2294
3 Fee	pattern(<<Fee>>,"\$#,##")		Right	5.00
4				
5				
6				

If necessary you can expand the input box by dragging on the bottom right hand corner (see “[Expanding the Input Box](#)” on page 132).



Database		Output Format		Tools
National Parks		<input checked="" type="radio"/> Tab <input type="radio"/> Comma <input type="radio"/> HTML		
Title	Formula	Width	Align	Sample Data
1 Park Name	<<Park>>		Left	Denali National Park
2 Phone Number	<<Phone Number>>		Left	(907) 683-2294
3 Fee	pattern(<<Fee>>,"\$#,##")			\$5.00
4				
5				
6				
7				
8				

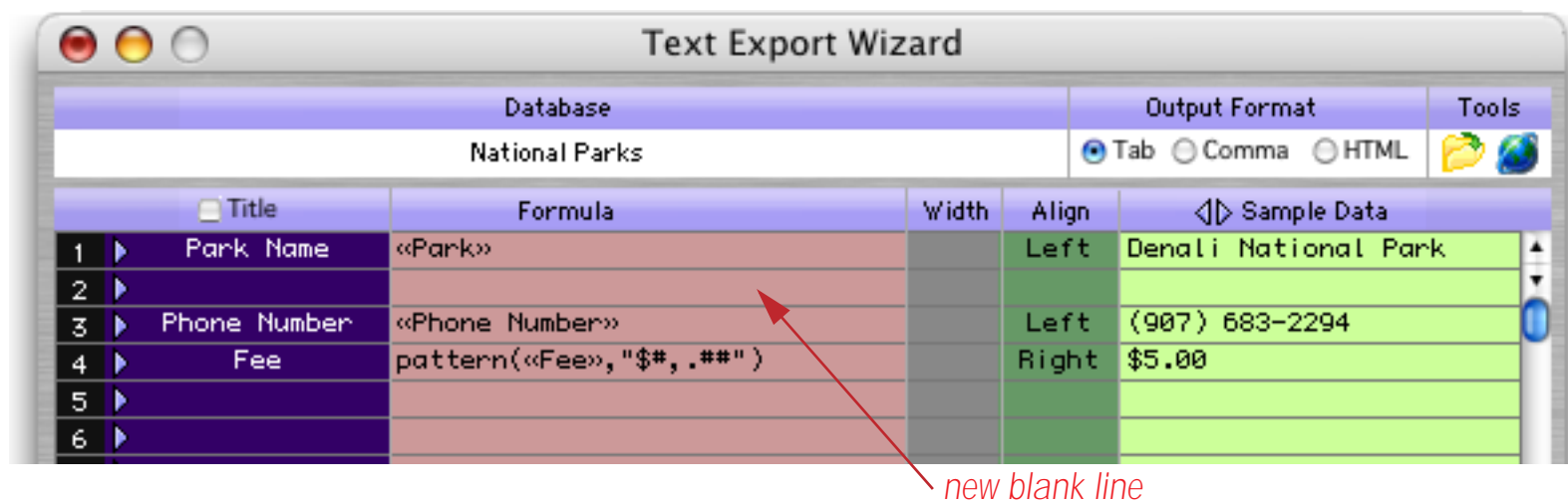
Press the **Enter** key when you have completed editing the item.



To insert a new item in the middle choose **Insert Line...** from the Edit menu. The wizard will ask you where you want the new line inserted.

A dialog box titled "Insert Before Line #:" with a text input field containing the number "2". Below the input field are two buttons: "Cancel" and "OK".

When you press **OK** the new line is inserted.



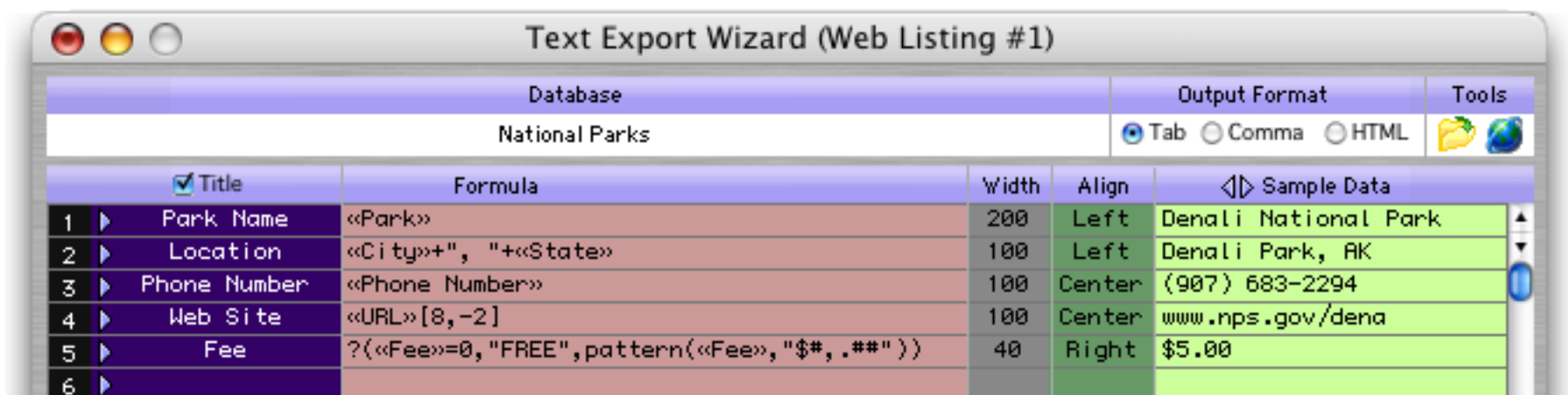
To delete an item use the **Delete Line...** command in the Edit menu. The wizard will ask you which line you want to delete.

A dialog box titled "Delete Line #:" with a text input field containing the number "2". Below the input field are two buttons: "Cancel" and "OK".

Enter the line number and press **OK** to delete the line.

### Common Export Formulas

The **Text Export Wizard** allows you to use any valid Panorama formula to create each item. The example below contains several formulas.



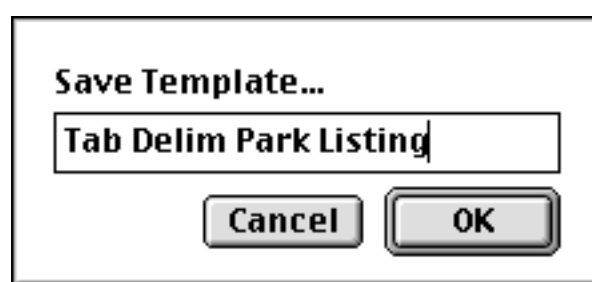
The second export field, **Location**, combines two database fields, **City** and **State** (see “[Gluing Strings Together](#)” on page 531). The fourth export field, **Web Site**, uses a text funnel to remove `http://` and `/` from each URL (see “[Taking Strings Apart \(Text Funnels\)](#)” on page 533). The final export field, **Fee**, uses the `?(` function to check for parks where the access fee is zero. If the fee is zero then the word **FREE** is exported instead of `$0.00` (see “[The ? Function](#)” on page 557). This formula also uses the `pattern(` function (see “[Converting Between Numbers and Strings](#)” on page 538).

The right hand side of the window displays a sample of the exported data for the current record in the database. This preview can be very handy for checking the result of your formulas. Using the small arrows you can move back and forth to preview other records in the database.

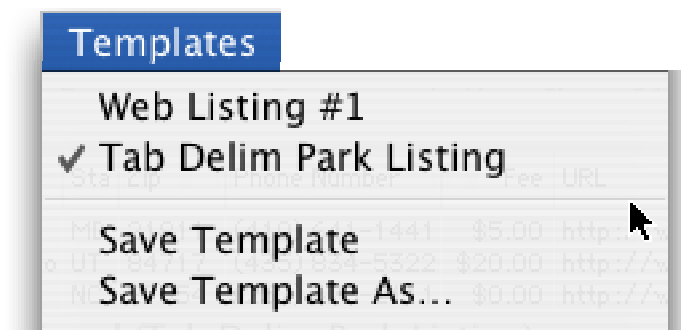


### Export Templates

If you think you’ll need to use an export configuration more than once you can save it as a **template**. The first step is to set up the configuration (as described in the previous section). Once the configuration is set up you can save it with the **Save Template** or **Save Template As...** commands in the Template menu. The wizard will prompt you to type in a name for the new template (the default is the name of the text file being imported).

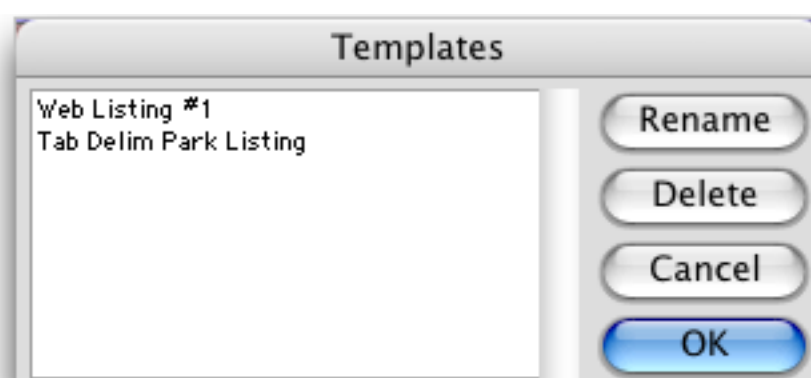


Once a template has been saved you can open it again by selecting it from the Template menu. (Note: The template is actually stored in the database being imported into (in this case [National Parks](#)). The template is only available when that database is being exported from. Each database may contain its own separate set of templates, which makes sense since the export configuration used with one database is not likely to work with any other database.)



The wizard loads the entire export configuration, ready to go. You can use the configuration as is or modify it before you actually import the data.

If you want to delete or rename a template choose the **Rename/Delete Templates...** command from the Template menu. To rename a template first click on it and then press the **Rename** button. A dialog appears allowing you to type in a new name. To delete a template press the **Delete** button.



When you are done press the **OK** button.

#### Choosing a Database to Export From

The **Text Import Wizard** normally exports from database that was active when the wizard was opened. However, you can use the **Database** menu to choose to export from any open database. Simply choose the database you want to export from and then set up the configuration.

#### Exporting Data to Microsoft Excel

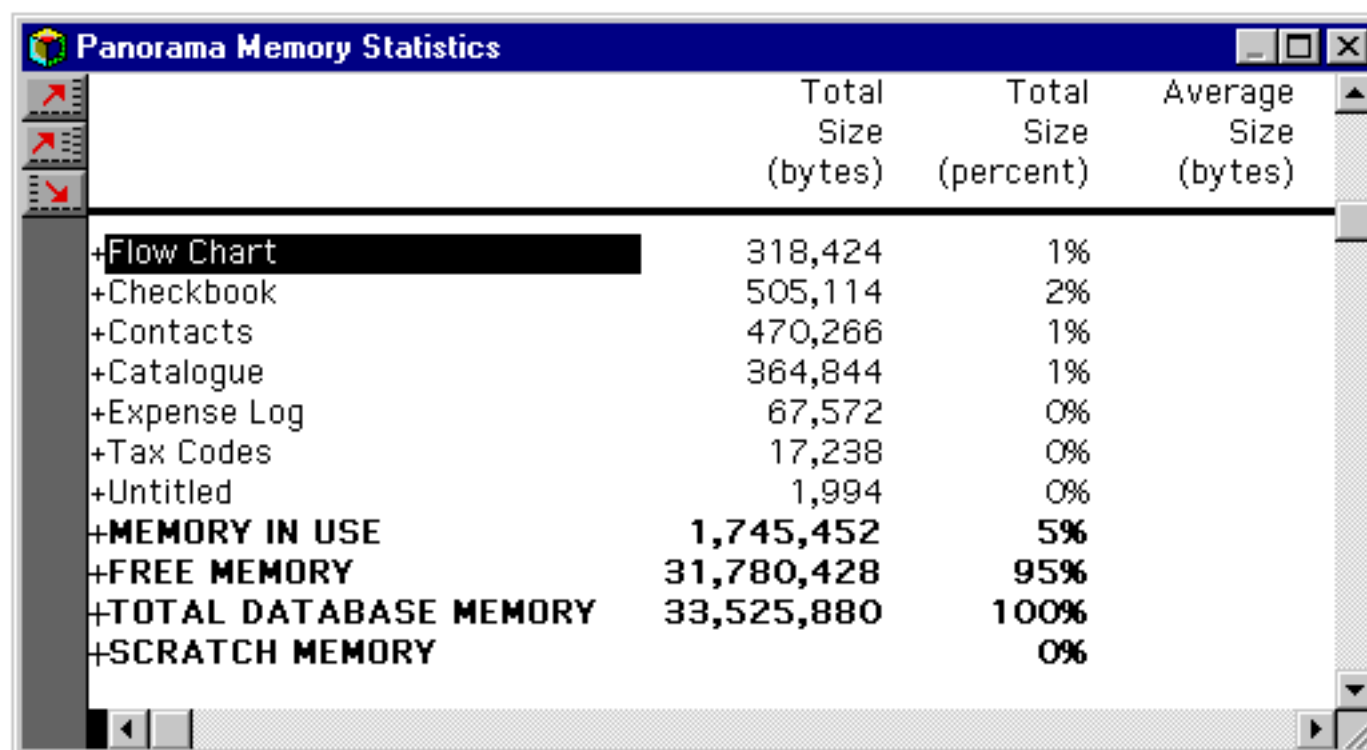
Panorama can export data directly to Microsoft Excel. To learn more about this feature see "[Excel Wizard](#)" on page 47 of the [Wizards & Demos](#) PDF file.

#### Exporting VCard Data

Panorama can export VCard data files to applications that support VCards. To learn more about this feature see "[Using Generic Fields with the VCard Wizard](#)" on page 242 of the [Panorama Handbook](#) PDF file.

## Monitoring Memory Usage

Each database you open with Panorama is copied into the RAM memory of your computer. For most typical databases you'll have plenty of RAM available. You can use the **Memory Usage** command in the File Menu to see how much memory is in use and how much is available for expansion or for opening additional databases. This command opens a statistics window that displays the current memory usage of every open database, along with overall memory usage statistics.



	Total Size (bytes)	Total Size (percent)	Average Size (bytes)
+Flow Chart	318,424	1%	
+Checkbook	505,114	2%	
+Contacts	470,266	1%	
+Catalogue	364,844	1%	
+Expense Log	67,572	0%	
+Tax Codes	17,238	0%	
+Untitled	1,994	0%	
<b>+MEMORY IN USE</b>	<b>1,745,452</b>	<b>5%</b>	
<b>+FREE MEMORY</b>	<b>31,780,428</b>	<b>95%</b>	
<b>+TOTAL DATABASE MEMORY</b>	<b>33,525,880</b>	<b>100%</b>	
<b>+SCRATCH MEMORY</b>		<b>0%</b>	

At the top of the memory statistics data sheet, each open file is listed with the amount of memory used by that file (in bytes and as a percentage of the total memory). The **Memory In Use** line shows the total amount of memory used by all of the open databases (in this case just under 2 megabytes for 7 databases). The **Free Memory** line shows the amount of free memory available for database expansion or for loading more databases (in this case over 32 megabytes). The **Total Database Memory** line shows the total amount of memory available for Panorama databases, including the memory that is already in use.

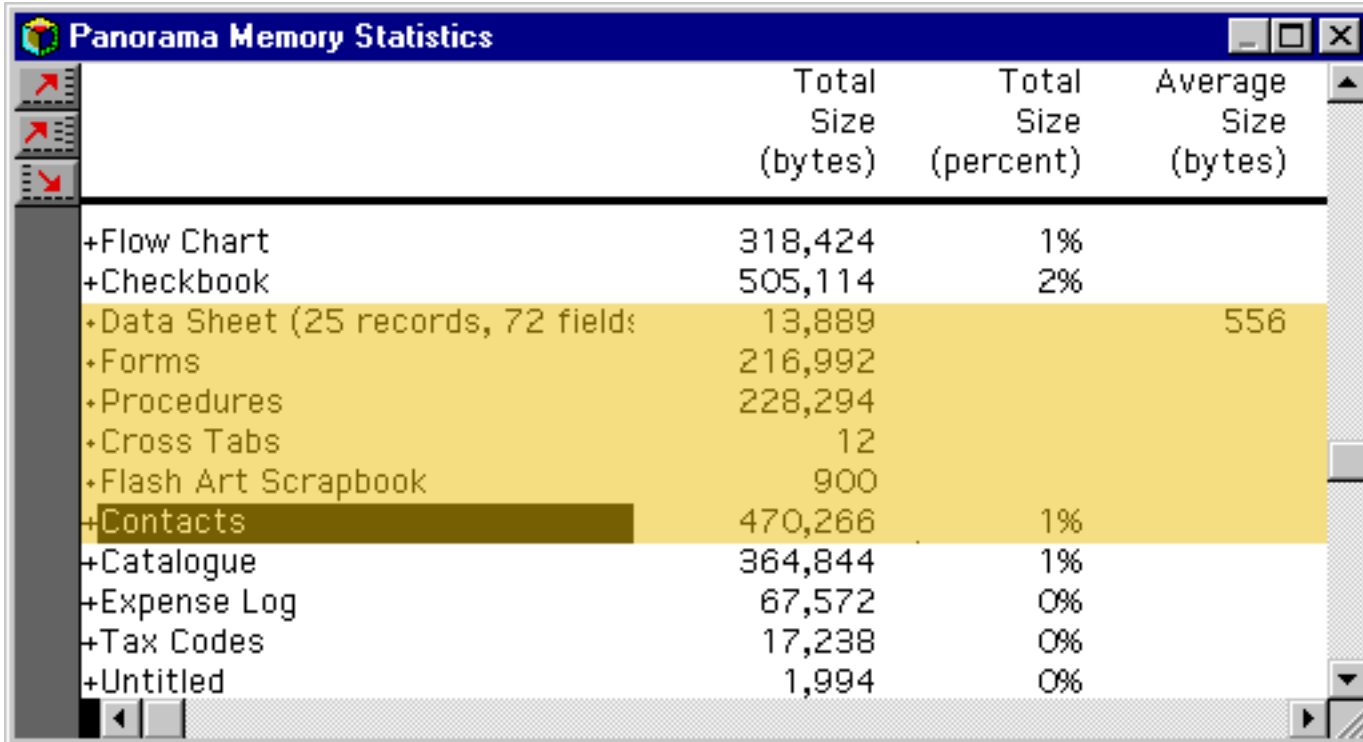
The memory statistics window has four columns. The first column shows the name of each memory area. The second column contains the actual number of bytes used by each area, while the third column displays each area's memory size as a percentage of the total database memory. The fourth column, Average Size, will be explained later.

When you are done with the memory statistics window, you can put it away by pressing the close box.

## Memory Usage Details

The memory statistics window shown above doesn't show much detail—only the overall memory usage for each file. You can use Panorama's outline tools (see "[Expanding and Collapsing Specific Details](#)" on page 198) to expand to greater levels of detail. To see more detail about a particular file, click on that file and choose the **Expand** tool.

Click on the database you want to examine and press the  **Expand** tool

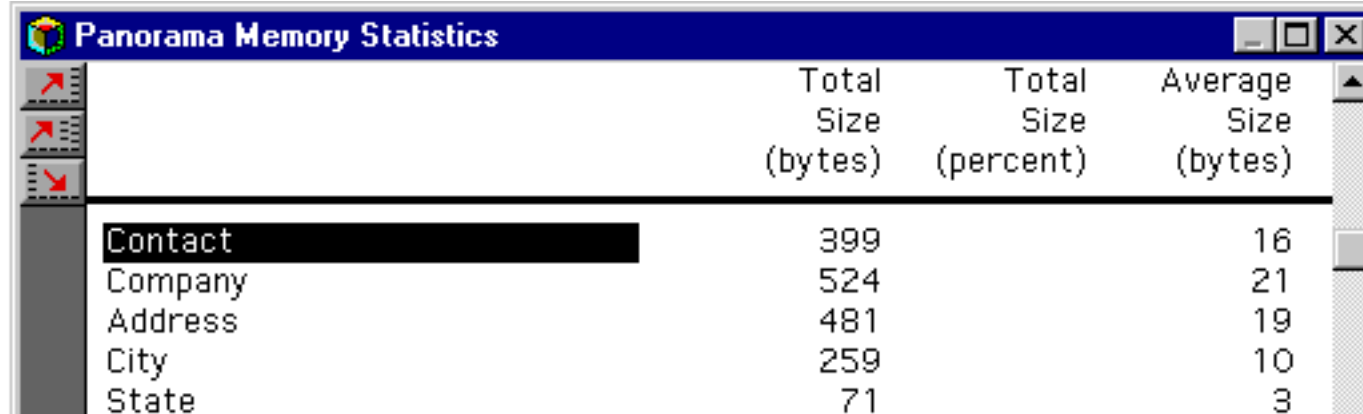


	Total Size (bytes)	Total Size (percent)	Average Size (bytes)
+Flow Chart	318,424	1%	
+Checkbook	505,114	2%	
+Data Sheet (25 records, 72 fields)	13,889		556
+Forms	216,992		
+Procedures	228,294		
+Cross Tabs	12		
+Flash Art Scrapbook	900		
+Contacts	470,266	1%	
+Catalogue	364,844	1%	
+Expense Log	67,572	0%	
+Tax Codes	17,238	0%	
+Untitled	1,994	0%	

The database statistics for this file are now divided into five subdivisions: **Data Sheet**, **Forms**, **Procedures**, **Cross Tabs**, and **Flash Art Scrapbook**. (Note: The yellow highlight in the window above is for illustration purposes only, it does not appear in the real window.)

The **Data Sheet** portion of the file contains the actual data itself. This line shows the number of records in the database, the total amount of data occupied by the data itself, and the average character length of each record in the database. In this case the database contains 25 records that are an average of 556 bytes long.

The **Data Sheet** line can be expanded further to show the amount of memory used by each field in the database, as shown below.



	Total Size (bytes)	Total Size (percent)	Average Size (bytes)
Contact	399		16
Company	524		21
Address	481		19
City	259		10
State	71		3

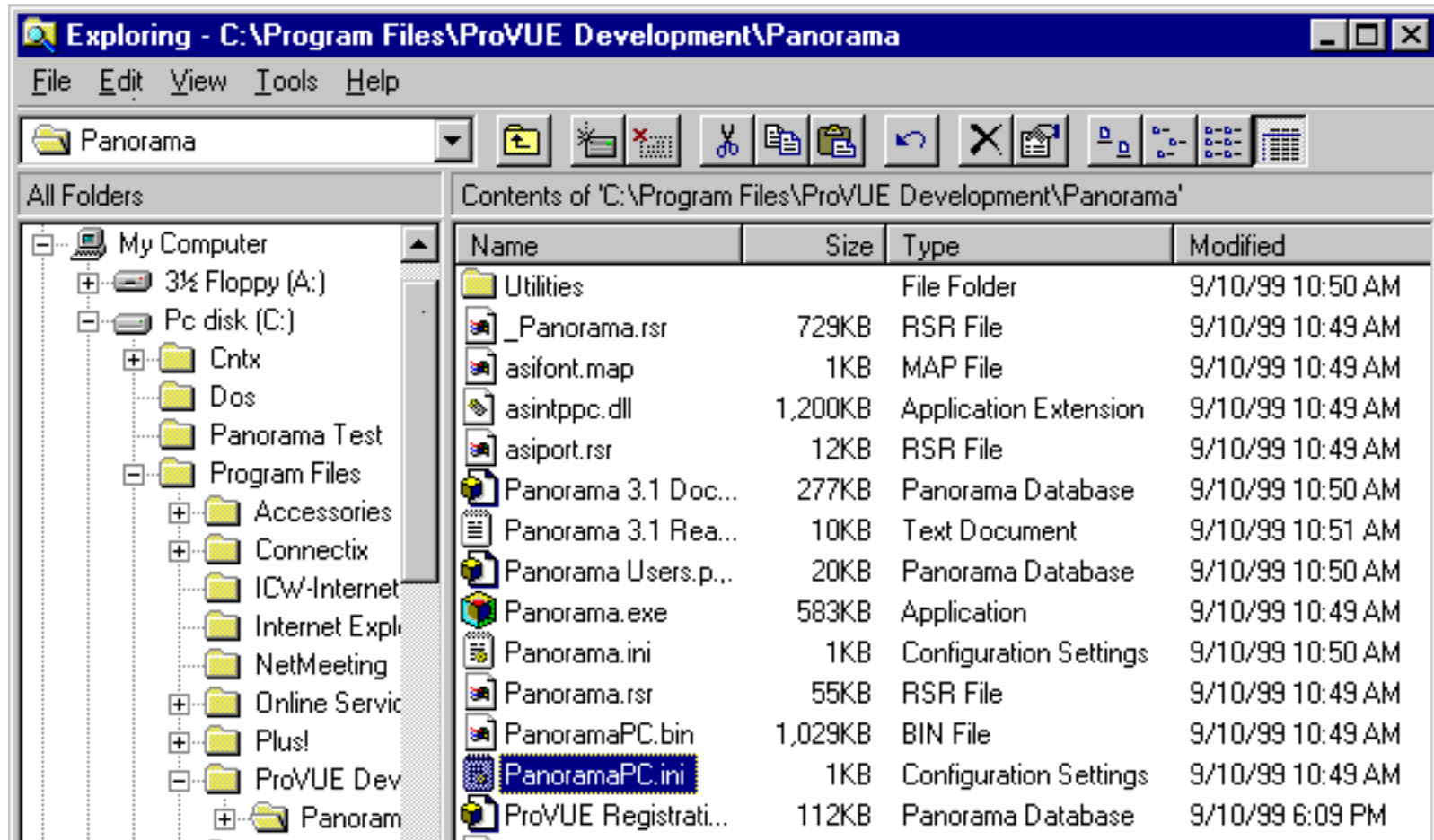
Each line shows the total amount of memory used by one field of the database, and the average size of one cell. For example, the **Contact** field (Name) is an average of 16 characters long, while the **Company** field is an average of 21 characters long.

The **Forms**, **Procedures**, and **Cross Tabs** lines can be expanded to show the actual amount of memory used by each individual form, procedure, and crosstab within the database. The **Flash Art Scrapbook** line can be expanded to show the size of each picture in the **Flash Art Scrapbook**. You usually won't need this kind of detail, but it can be useful if a file seems unusually large and you need to find out why.

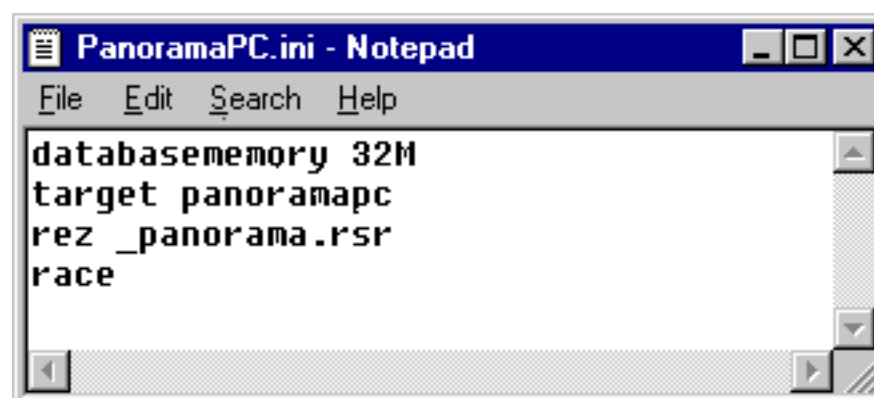


## Adjusting Panorama's Database Memory Allocation

As shipped from the factory, Panorama normally allocates 32 megabytes of memory for databases. For most applications this is more than enough. However, if you wish to use extremely large databases you may need to increase this allocation. To do this, locate the **PanoramaPC.ini** (Windows) or the **PanoramaPowerPC.ini** (Mac OS X) file. As shown in the illustration below, this file is normally located in the **C:\Program Files\ProVUE Development\Panorama** folder on Windows systems. On Mac OS X systems it is normally in the **Applications:Panorama** folder. In either case it is always in the same folder as the Panorama application itself.



Double click this file to open it in the Notepad or TextEdit. (Note: Be sure you open **PanoramaPC.ini** (Windows) or **PanoramaPowerPC.ini** (OS X) and not **Panorama.ini**. If you open the wrong file, don't worry, just close it and open the correct file.) The file should look something like this:



The **databasememory** line controls the amount of memory allocated by Panorama for databases. You may set this to any value from 3M to 999M. (Don't forget the M!). However, if you set this value to larger than the physical amount of memory available on your computer, you may reduce the amount of virtual memory available for other applications. We do not recommend opening databases that are larger than the physical memory size of your computer. Panorama will open the file and operate correctly, but its performance may be severely degraded. Once you have set the new value save and close the window, then relaunch Panorama if necessary.



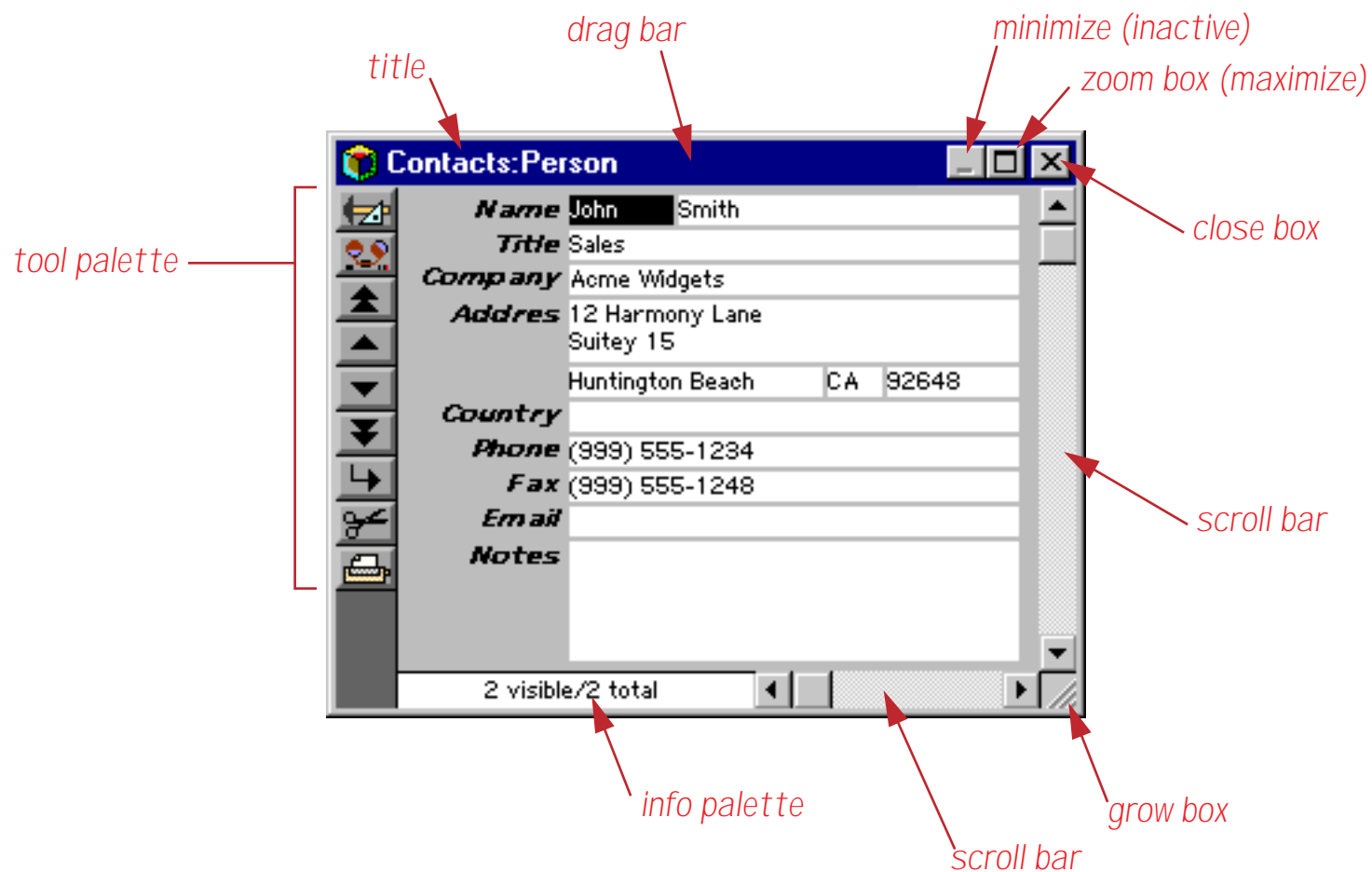
# Chapter 2: Windows



As you use Panorama most of the action takes place inside windows. Panorama uses standard windows with a few unique touches, including a tool palette with pop-up help on the left side of each window. Panorama also has an enhanced zoom box that lets you zoom a window to a specific position. This chapter covers both the standard and enhanced window features.

## Window Components

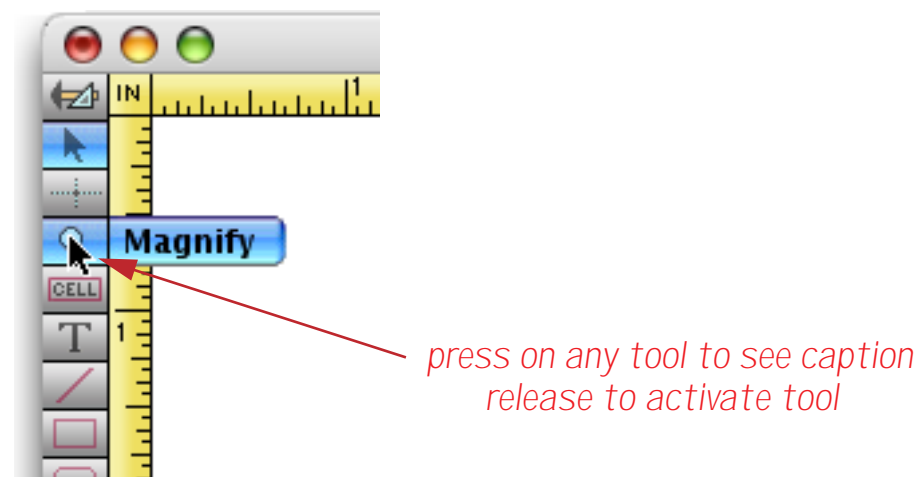
Each Panorama window has about a dozen components for configuring and controlling the window. Each of these components is activated by clicking or dragging with the mouse. Here's what a typical Panorama window looks like on a PC system.



On the Macintosh windows look very similar, except that the close box is on the left and the title is centered.

## Tool Palette

The left side of each window contains a tool palette. To help you learn and remember the function of each tool, the tool palette displays a pop-up caption whenever you click on a tool. The caption will remain visible as long as you hold the mouse down over the tool.



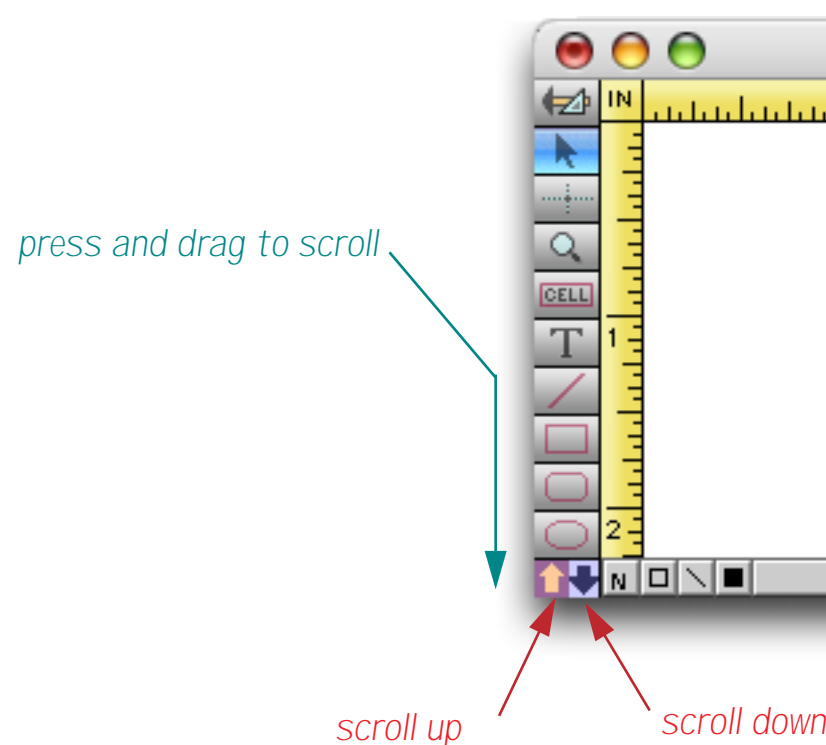
Unlike most tool palettes that perform a function when you click on a tool, Panorama's tool palette doesn't perform the function until you release the mouse. This allows you to look at the caption before you activate the tool. (Of course, you don't have to hold down the mouse and look at the caption. Once you have memorized the tool icons you can simply click on the tool you want—just like your favorite graphics or page layout program.)

You can scan through the tool captions by dragging the mouse up or down the palette (just as you would scan across the menu bar). The caption for each tool pops up as the mouse is dragged across it. You can stop at any time and release the mouse to activate a tool.

Remember, to activate a tool you must release the mouse over the tool itself—not over the pop-up caption. The pop-up caption is not a menu.

### Scrolling the Tool Palette

What if all the tools don't fit in the window? You can scroll the tools by pressing on any tool and then dragging to the top or bottom of the palette. When you reach the edge of the palette the tools will scroll. You can also scroll the tools by clicking on one of the small arrows at the bottom of the palette. Each click scrolls by one tool.



If you have a scroll wheel mouse you can also scroll the tool palette by moving the mouse over the palette and turning the scroll wheel (OS X only).

## Scroll Bars

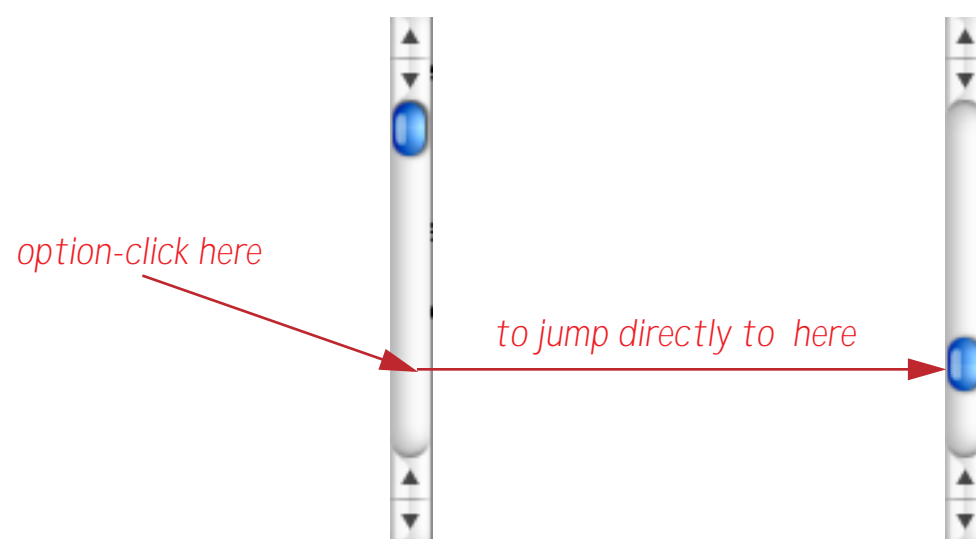
The scroll bars are used to shift the information or graphics displayed in the window. The vertical scroll bar (on the right edge of the window) shifts the display up and down, while the horizontal scroll bar (on the bottom edge of the window) shifts the display left and right. The sliding box inside each scroll bar shows the current position of the display.

If you are using OS X and a mouse with a scroll wheel, Panorama supports the scroll wheel. This wheel normally causes the window to scroll vertically when turned. If the **Shift** key is held down, the wheel will scroll the window horizontally.

When using OS X or OS 9, the **Option** key changes the way the scroll bars work. If you hold down the **Option** key and click on the up or down arrows, the window will scroll up or down an entire page. If you hold down the **Option** key and click in the main body of the scroll bar, the window will scroll directly to that spot.

### Instant Jump to any Scroll Bar Position

To jump instantly to any position within a scroll bar hold down the **Option** key and click on the location you want to jump to.



Note: Most Mac OS X applications support this behaviour, including Safari, Preview, GarageBand, etc.

### Horizontal Scrollwheel Scrolling with Shift Key

When using any scroll wheel holding down the **Shift** key while turning the wheel will cause the window to scroll horizontally. Note: Most Mac OS X applications support this behaviour, including Safari, Preview, GarageBand, etc.

### Mighty Mouse and Trackpad Scrolling Support

Panorama supports vertical scrolling with the ball on Apple's Mighty Mouse, as well as two finger scrolling on laptop trackpads. Please note, however, that horizontal and diagonal scrolling are not supported (though you can scroll diagonally using the **Shift** key as noted above).



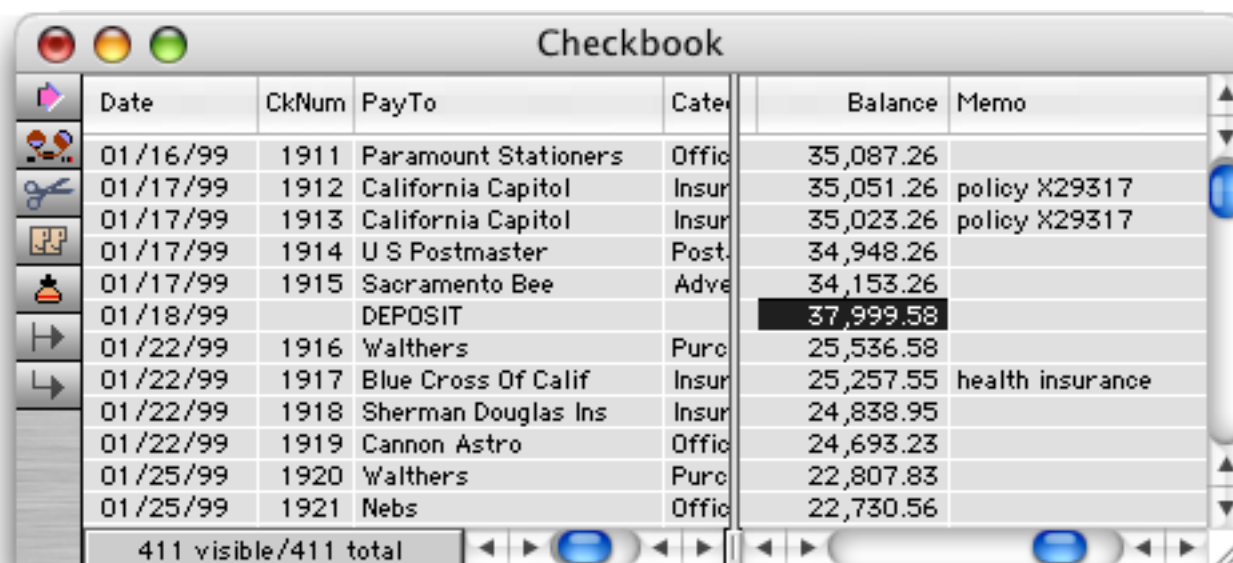
## Splitting a Window

Some Panorama windows can be split into two side by side panes. Each pane displays a different area of the database. The two panes are locked together vertically, but each pane has its own horizontal scroll bar.



*press and drag to split window*

To split a window, drag the **splitter** to the right. (The **splitter** is the small black rectangle to the left of the horizontal scroll bar.)



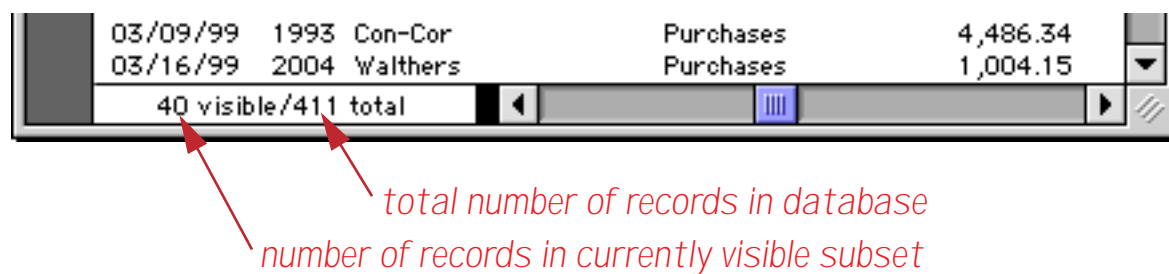
*each pane may be scrolled separately*

*drag to adjust split*

To remove the split, drag the splitter back to the left edge. To adjust the split location, drag the splitter into position.

## Info Palette

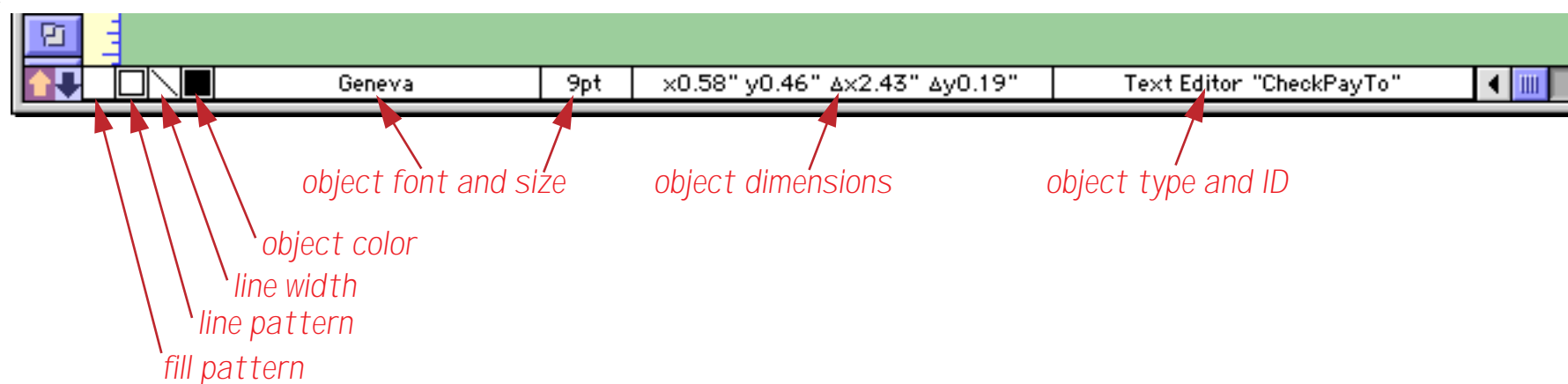
Many Panorama windows display an **Info Palette** along the bottom left corner of the window. In windows that are used for displaying and editing data (data sheet, forms, crosstabs), the info palette displays the current number of database records in the bottom left corner of each window. Panorama displays both the total number of records and the number of visible (selected) records.



If the window is less than 3 inches wide, the record count will not appear. You can also turn the record count on or off with the **Show Record Count** command in the Setup menu.

Clicking anywhere in the record count opens the **Find/Select** dialog. This dialog allows you to locate information within the database. See "[The Find/Select Dialog](#)" on page 160 for details on this dialog.

When a form window is switched into **Graphic Display Mode** the Info Palette displays a graphic control strip. This strip displays the graphic attributes of the currently selected object (or objects), and allows you to change the attributes with pop-up menus.



See "[The Graphic Control Strip](#)" on page 259, for more information on the graphic control strip.

Procedure windows also display status information along the bottom of the window. When writing a procedure Panorama displays error messages here. When single stepping through a procedure during debugging Panorama displays the results of any assignments into fields or variables in this area (see "[The Panorama Interactive Debugger](#)" on page 622).

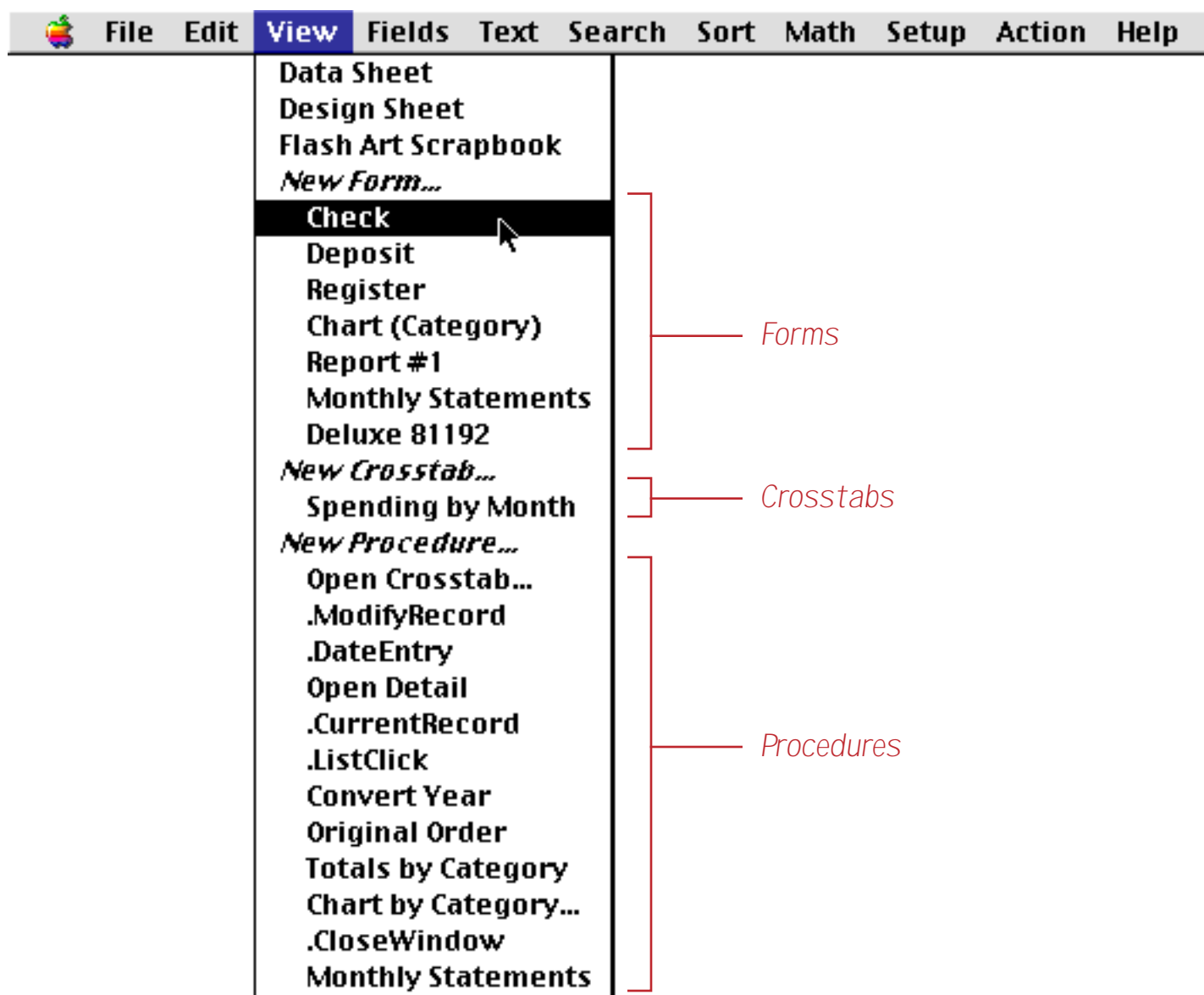


# Chapter 3: Views



A Panorama database can have up to six elements: data sheet, forms, design sheet, procedures, crosstabs, and flash art. Each window shows a view of one of these elements.

The View menu is just to the right of the Edit menu.



Use this menu to pick which view you want to see in the window. Choosing different views from this menu is like pointing a camera in different directions.

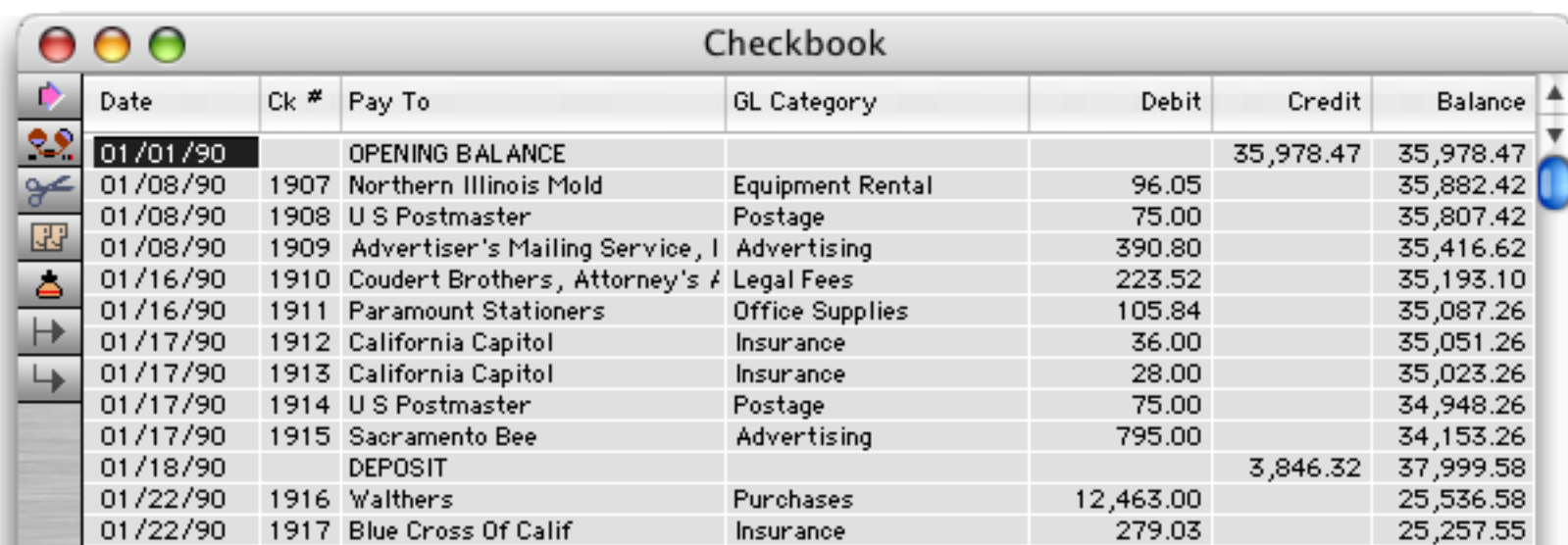
## Types of Panorama Views

Panorama has six different kinds of views. Each type of view gives you access to a different element of the database or gives you a different perspective on your data (for example, form vs. data sheet).

A new database starts with a data sheet, design sheet, and an empty flash art scrapbook. Views for forms, procedures, and crosstabs can be added if desired.

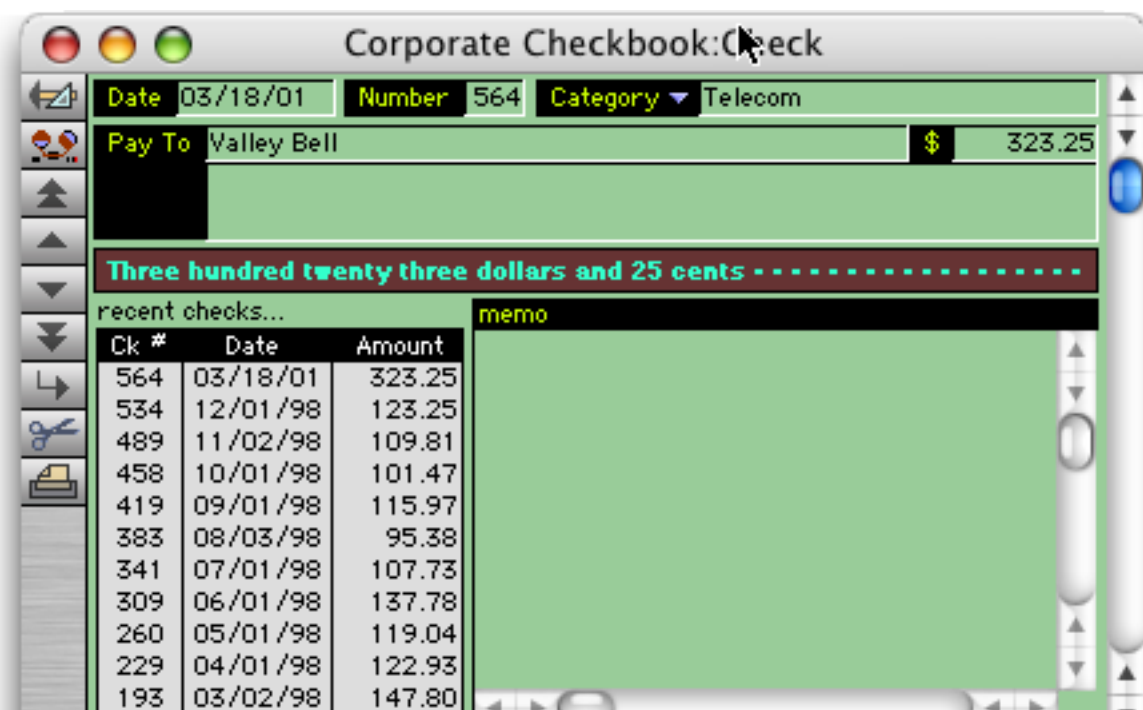
## Data Sheet and Form Views

The most important views are the data sheet and forms. These views give you access to the actual data. Use the [data sheet](#) view when you want to display the database as a sheet of rows and columns. The data sheet view has a fixed format very much like a spreadsheet, as shown below. Although you can make minor alterations like changing the font size or the width of a column, you cannot add graphics or change the overall arrangement of the data sheet view.



Date	Ck #	Pay To	GL Category	Debit	Credit	Balance
01/01/90		OPENING BALANCE			35,978.47	35,978.47
01/08/90	1907	Northern Illinois Mold	Equipment Rental	96.05		35,882.42
01/08/90	1908	U S Postmaster	Postage	75.00		35,807.42
01/08/90	1909	Advertiser's Mailing Service, I	Advertising	390.80		35,416.62
01/16/90	1910	Coudert Brothers, Attorney's #	Legal Fees	223.52		35,193.10
01/16/90	1911	Paramount Stationers	Office Supplies	105.84		35,087.26
01/17/90	1912	California Capitol	Insurance	36.00		35,051.26
01/17/90	1913	California Capitol	Insurance	28.00		35,023.26
01/17/90	1914	U S Postmaster	Postage	75.00		34,948.26
01/17/90	1915	Sacramento Bee	Advertising	795.00		34,153.26
01/18/90		DEPOSIT			3,846.32	37,999.58
01/22/90	1916	Walthers	Purchases	12,463.00		25,536.58
01/22/90	1917	Blue Cross Of Calif	Insurance	279.03		25,257.55

Use a [form](#) view when you want complete control over the arrangement and appearance of your data. Instead of appearing in a fixed row and column grid, the data can be arranged any way you want. Graphics can be added for clarity or to simulate an actual paper form. The form view is much more flexible than the data sheet view, but it is also more work to set up. Here is a typical example of a form. Notice that the window name shows the database name, [Checkbook](#), followed by the form name, [Plain Checks](#).



Corporate Checkbook: Check

Date: 03/18/01    Number: 564    Category: Telecom

Pay To: Valley Bell    \$ 323.25

Three hundred twenty three dollars and 25 cents

recent checks...

Ck #	Date	Amount	memo
564	03/18/01	323.25	
534	12/01/98	123.25	
489	11/02/98	109.81	
458	10/01/98	101.47	
419	09/01/98	115.97	
383	08/03/98	95.38	
341	07/01/98	107.73	
309	06/01/98	137.78	
260	05/01/98	119.04	
229	04/01/98	122.93	
193	03/02/98	147.80	

Every Panorama database has a single data sheet view, but can have any number of form views. A simple database might not have any form views, while a complex database may have dozens. You create as many forms as you need. Each form is identified by a unique name.

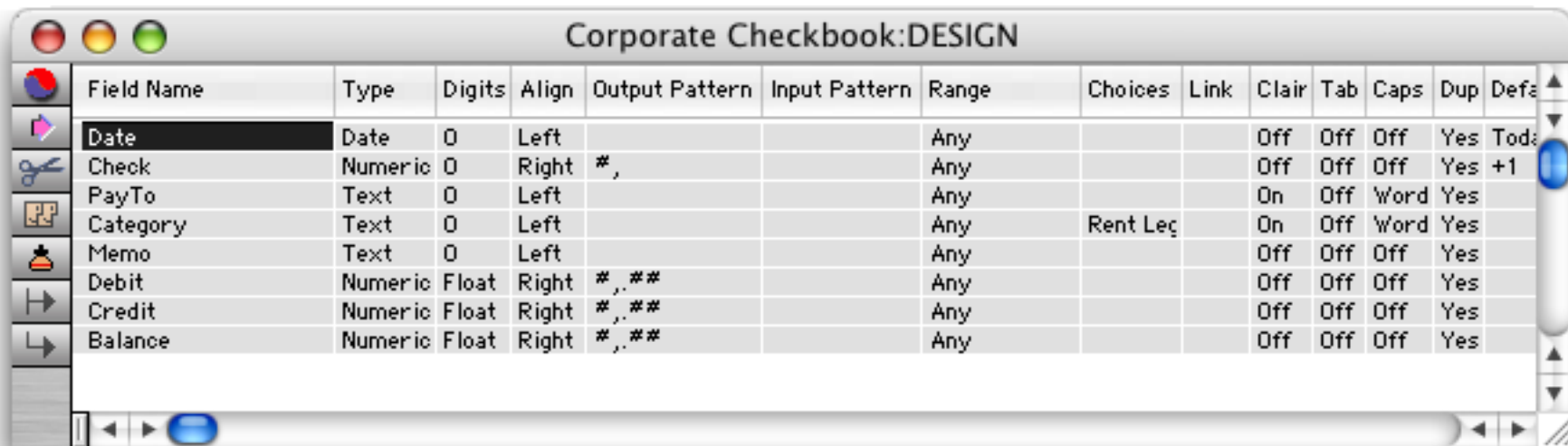
Since the data sheet and form views are based on the same underlying data, any action or change made to the data in one of these views will immediately appear in the other views. Of course you'll only notice this when several windows are open at once.



## Other Views

In addition to the data sheet and form views Panorama has four other kinds of views. Instead of accessing the actual data, these views let you access the other components of a Panorama file.

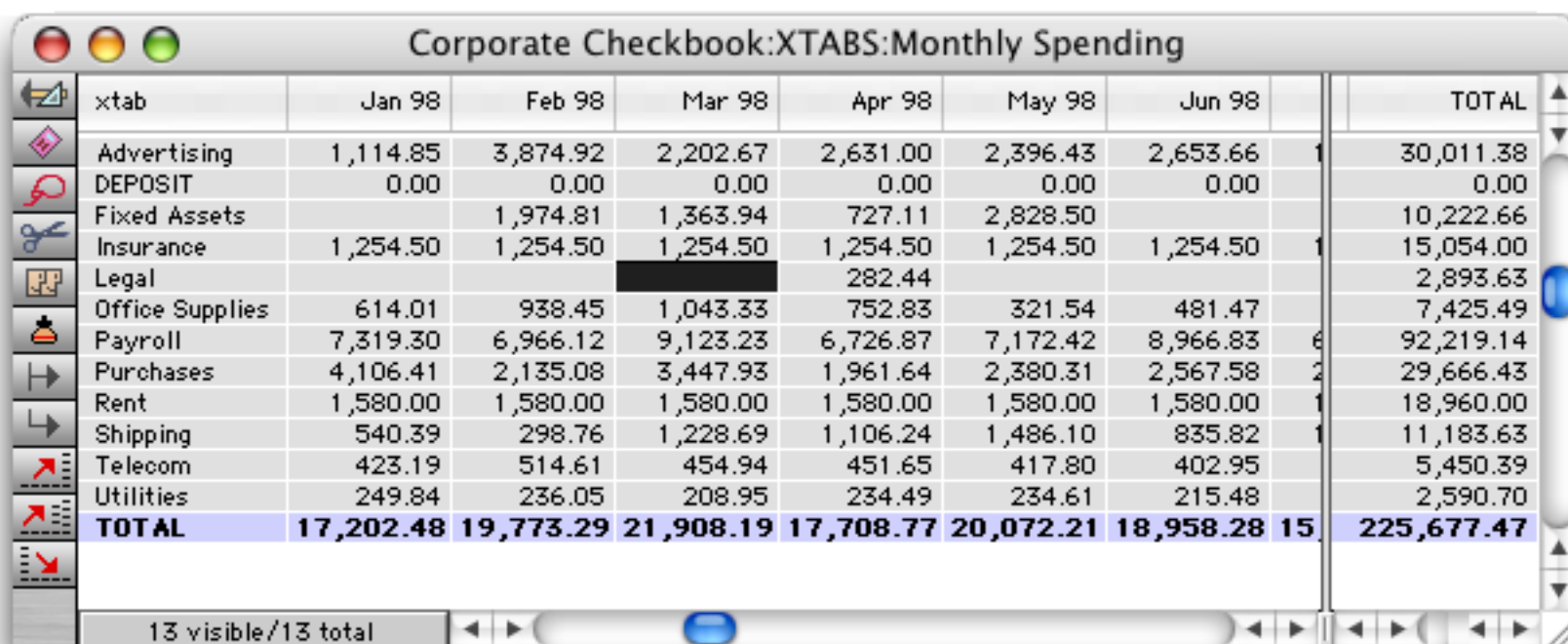
The [design sheet view](#) is the DNA or blueprint of the database. It contains the actual structure of the database fields. You can add and remove fields and make other minor modifications to the database structure without using the design sheet, but the design sheet provides the ultimate control over the structure of your database. “[The Design Sheet](#)” on page 105 to learn more.



Field Name	Type	Digits	Align	Output Pattern	Input Pattern	Range	Choices	Link	Clair	Tab	Caps	Dup	Defa
Date	Date	0	Left			Any			Off	Off	Off	Yes	Today
Check	Numeric	0	Right	#,		Any			Off	Off	Off	Yes	+1
PayTo	Text	0	Left			Any			On	Off	Word	Yes	
Category	Text	0	Left			Any	Rent Lec		On	Off	Word	Yes	
Memo	Text	0	Left			Any			Off	Off	Off	Yes	
Debit	Numeric	Float	Right	#,##		Any			Off	Off	Off	Yes	
Credit	Numeric	Float	Right	#,##		Any			Off	Off	Off	Yes	
Balance	Numeric	Float	Right	#,##		Any			Off	Off	Off	Yes	

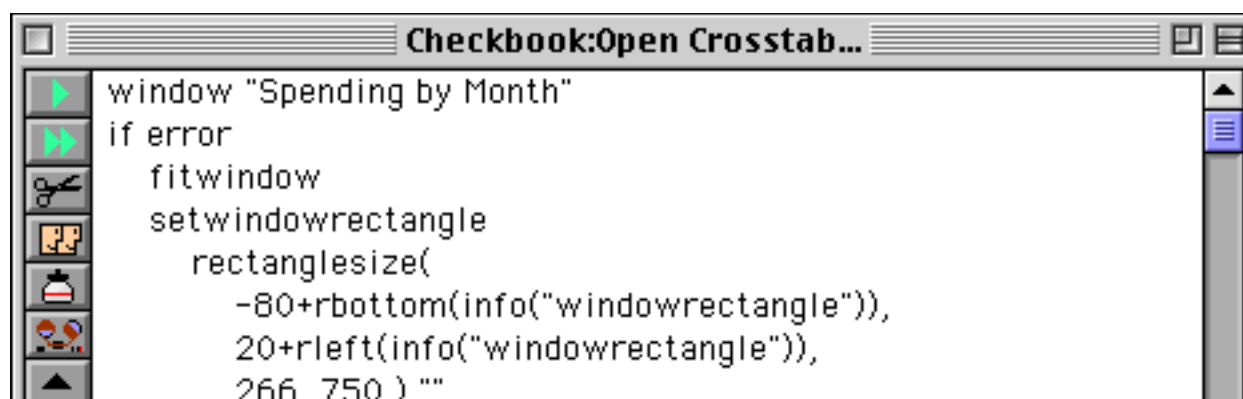
The [flash art scrapbook view](#) contains a catalog of pictures that can be displayed in a form or report. However this feature is rarely used, instead, images are usually displayed directly from the disk (see Chapter 16).

[Crosstab views](#) display a special 2-way summarization of the information in a database. Crosstabs display the relationships between data in different fields, exposing trends that might be invisible in the normal database views (data sheet and forms). Like form views, a database can contain any number of crosstab views—it may have none or several, and each crosstab has a name. The window name shows the database name (for example [Checkbook](#)), followed by XTABS, followed by the crosstab name (for example [Spending by Month](#)). To learn more about crosstabs see “[Crosstabs](#)” on page 209.



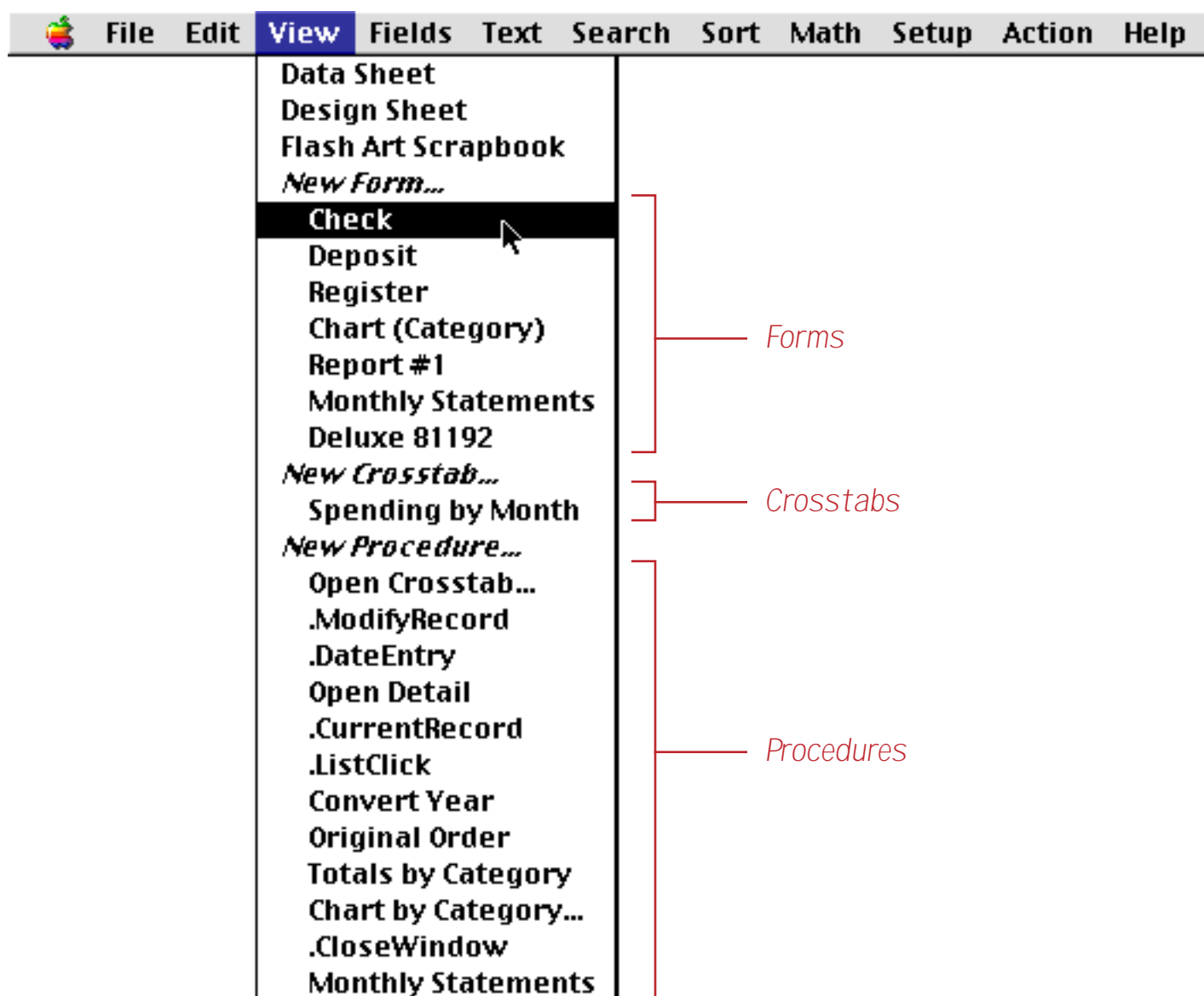
xtab	Jan 98	Feb 98	Mar 98	Apr 98	May 98	Jun 98	TOTAL
Advertising	1,114.85	3,874.92	2,202.67	2,631.00	2,396.43	2,653.66	30,011.38
DEPOSIT	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Fixed Assets		1,974.81	1,363.94	727.11	2,828.50		10,222.66
Insurance	1,254.50	1,254.50	1,254.50	1,254.50	1,254.50	1,254.50	15,054.00
Legal				282.44			2,893.63
Office Supplies	614.01	938.45	1,043.33	752.83	321.54	481.47	7,425.49
Payroll	7,319.30	6,966.12	9,123.23	6,726.87	7,172.42	8,966.83	92,219.14
Purchases	4,106.41	2,135.08	3,447.93	1,961.64	2,380.31	2,567.58	29,666.43
Rent	1,580.00	1,580.00	1,580.00	1,580.00	1,580.00	1,580.00	18,960.00
Shipping	540.39	298.76	1,228.69	1,106.24	1,486.10	835.82	11,183.63
Telecom	423.19	514.61	454.94	451.65	417.80	402.95	5,450.39
Utilities	249.84	236.05	208.95	234.49	234.61	215.48	2,590.70
<b>TOTAL</b>	<b>17,202.48</b>	<b>19,773.29</b>	<b>21,908.19</b>	<b>17,708.77</b>	<b>20,072.21</b>	<b>18,958.28</b>	<b>15,225,677.47</b>

Procedure views contain sequences of instructions for Panorama to follow. You don't have to remember each step—Panorama will remember for you. Once a procedure is set up, it can be activated several different ways. You can choose the procedure from a menu, press a button, or use a **Command** key combination (Macintosh) or **Control** key shortcut (PC). Procedures can even be activated automatically when special events occur. Like form and crosstab views, a database can contain any number of procedure views. Each procedure has its own name, which is shown in the window title. To learn more about procedures see "[Procedures](#)" on page 571.



## Switching Between Views

The **View** Menu lists all the views in a database. The pre-defined views appear at the top—data sheet, design sheet, and flash art scrapbook. Next come the views you've created—forms, crosstabs, and procedures. The View Menu also contains commands for creating your own new views—new form, new crosstab, and new procedure.

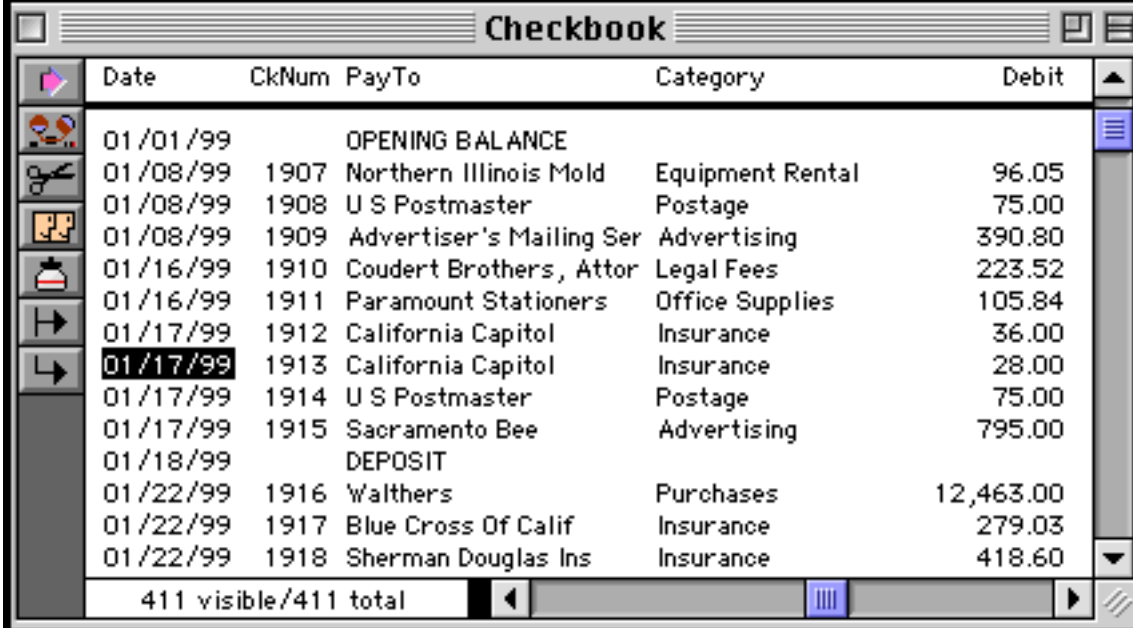


To switch to a different view within the same window, simply choose the view from the menu and release the mouse. You can flip back and forth between views at any time.

## Opening More Than One Window Per Database

You can also use the view menu to open a new window, allowing you to see two views of the database at once. To open a second window the same size as the current window, hold down the **Alt** key while you select from the **View** menu. (If you are using a Macintosh, hold down the **Control** key or the **Option** key.) The new window will appear slightly below and to the right of the original window.

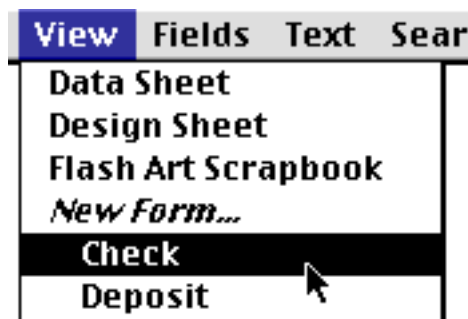
1) Start with one window



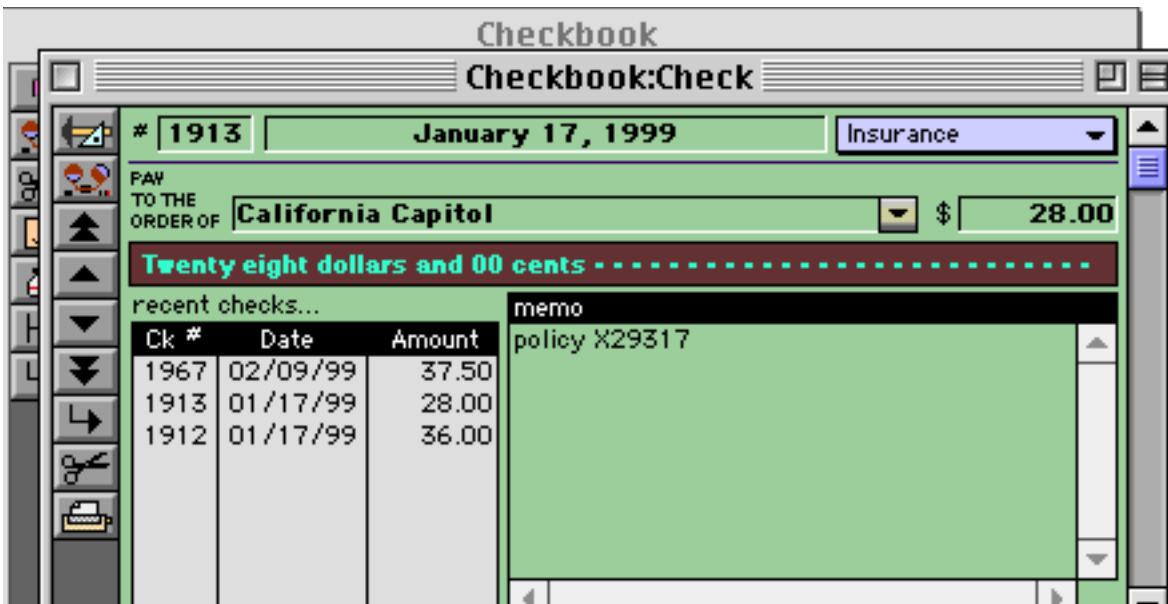
Date	CkNum	PayTo	Category	Debit
01/01/99		OPENING BALANCE		
01/08/99	1907	Northern Illinois Mold	Equipment Rental	96.05
01/08/99	1908	U S Postmaster	Postage	75.00
01/08/99	1909	Advertiser's Mailing Ser	Advertising	390.80
01/16/99	1910	Coudert Brothers, Attor	Legal Fees	223.52
01/16/99	1911	Paramount Stationers	Office Supplies	105.84
01/17/99	1912	California Capitol	Insurance	36.00
01/17/99	1913	California Capitol	Insurance	28.00
01/17/99	1914	U S Postmaster	Postage	75.00
01/17/99	1915	Sacramento Bee	Advertising	795.00
01/18/99		DEPOSIT		
01/22/99	1916	Walthers	Purchases	12,463.00
01/22/99	1917	Blue Cross Of Calif	Insurance	279.03
01/22/99	1918	Sherman Douglas Ins	Insurance	418.60

411 visible/411 total

2) While holding down the **Alt** key (PC) or the **Control** key (Mac), make a selection from the **View** menu. On the Mac you can also use the **Option** key.



3) The new window appears slightly below and to the right...



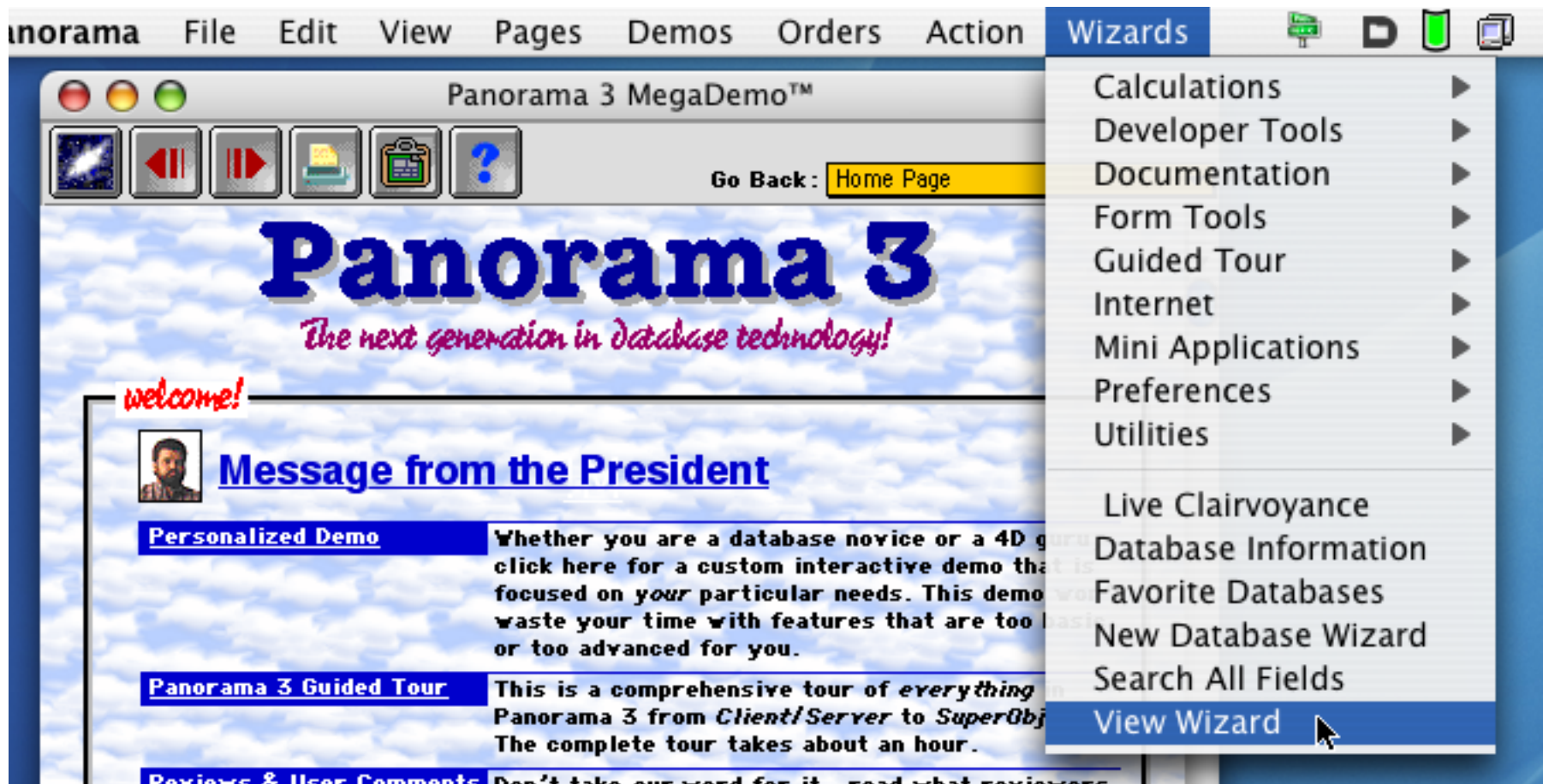
Check #	Date	Amount	Memo
1913	January 17, 1999	Insurance	
PAY TO THE ORDER OF California Capitol \$ 28.00			
Twenty eight dollars and 00 cents			
recent checks...			memo
Ck #	Date	Amount	policy X29317
1967	02/09/99	37.50	
1913	01/17/99	28.00	
1912	01/17/99	36.00	






The new window will track the original window. In fact, all windows associated with a database will track each other automatically. Any changes made in one window automatically appear in all other windows, and when any navigation is done in one window (moving up or down within the database) all of the other windows will follow along.

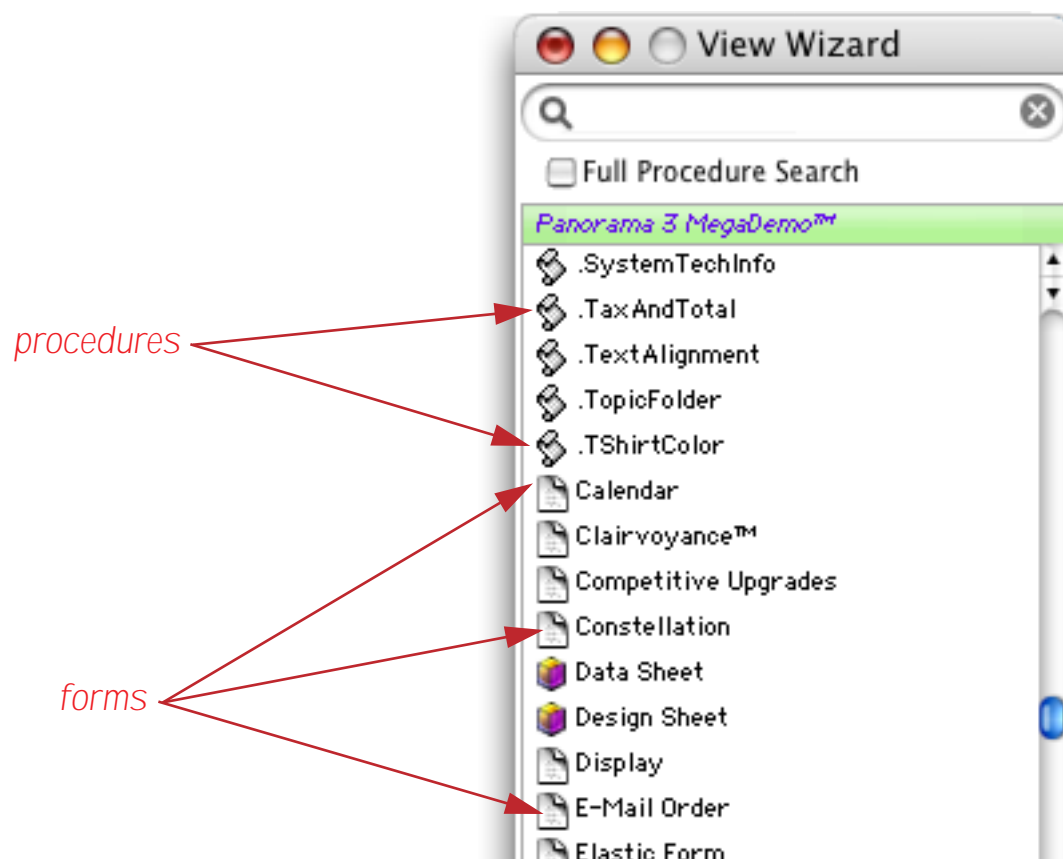
Panorama allows you to open up to 64 windows at one time.

## The View Wizard

The **View** menu works well for most databases, but when a database grows to dozens of forms and hundreds of procedures it can get a bit unwieldy. For these situations the **View Wizard** comes in handy. This is a database that comes with Panorama that can help you locate and open any view. When you first open the View Wizard database it displays a list of all the forms or procedures in the currently open Panorama database. (Whether it initially displays forms or procedures depends on what type of window was open before the wizard was activated.) For example, suppose the **Panorama 3 MegaDemo™** file is open.

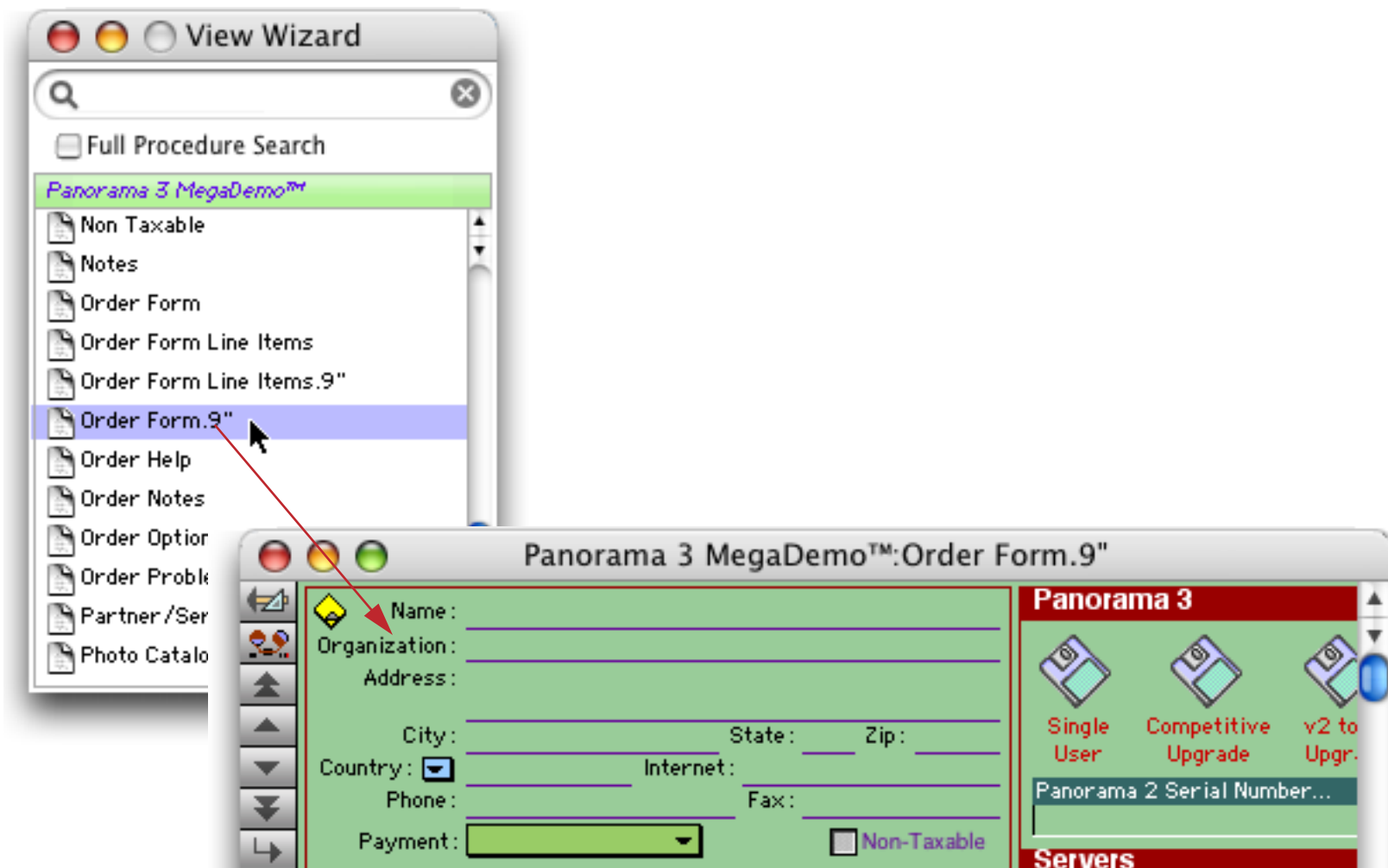


When the **View Wizard** opens it displays a list of the views in this database (**Panorama 3 MegaDemo**), including  forms,  procedures,  crosstabs, the  data sheet and the  design sheet. In the example below the procedures and forms appear to be grouped together, but that is just a coincidence — the views are listed in alphabetical order.

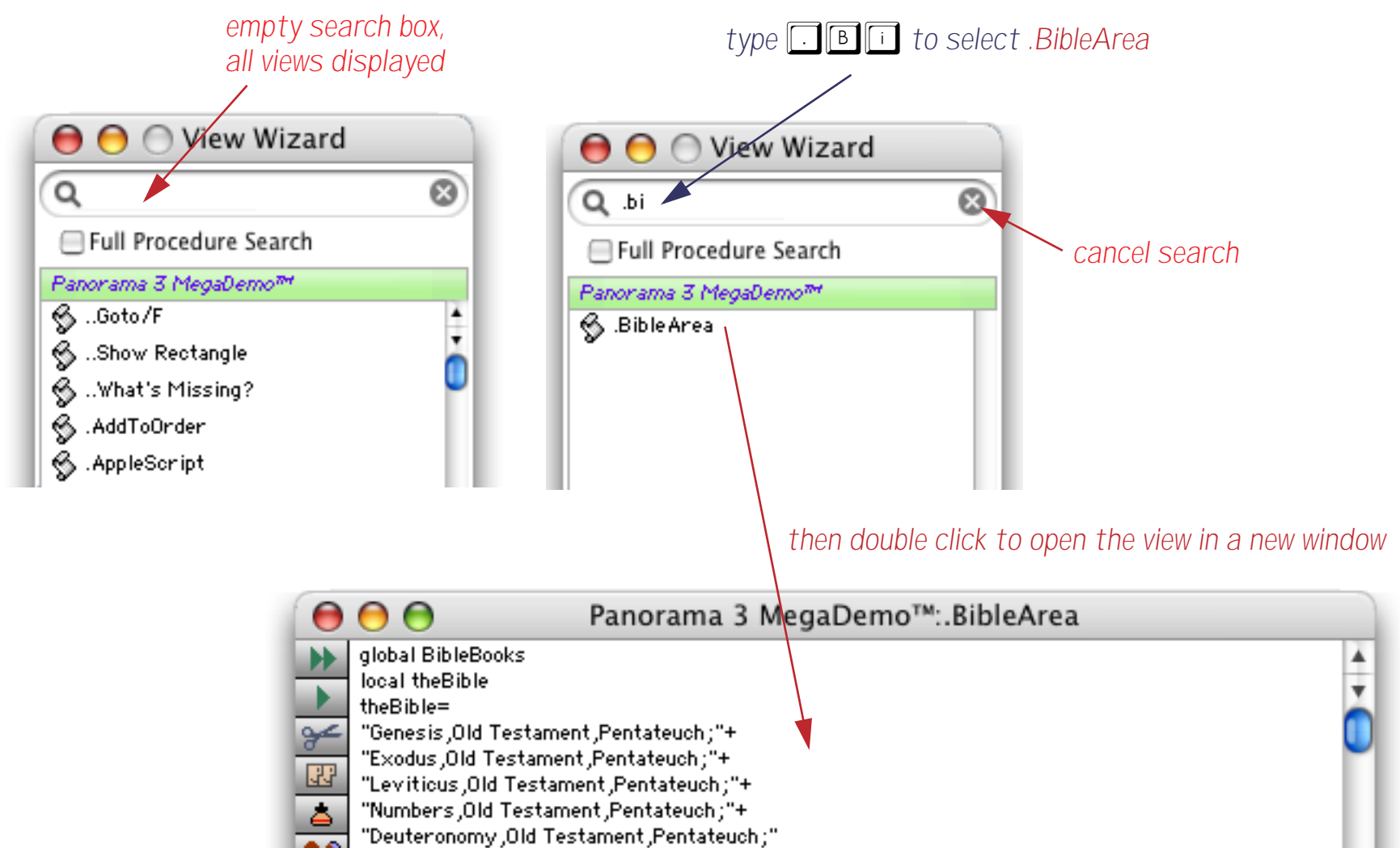




Double click to open any view.



If you know the name of the view you want to open you can find it quickly by typing in the first few letters. As you type each letter, the wizard will show you the views that match that name.

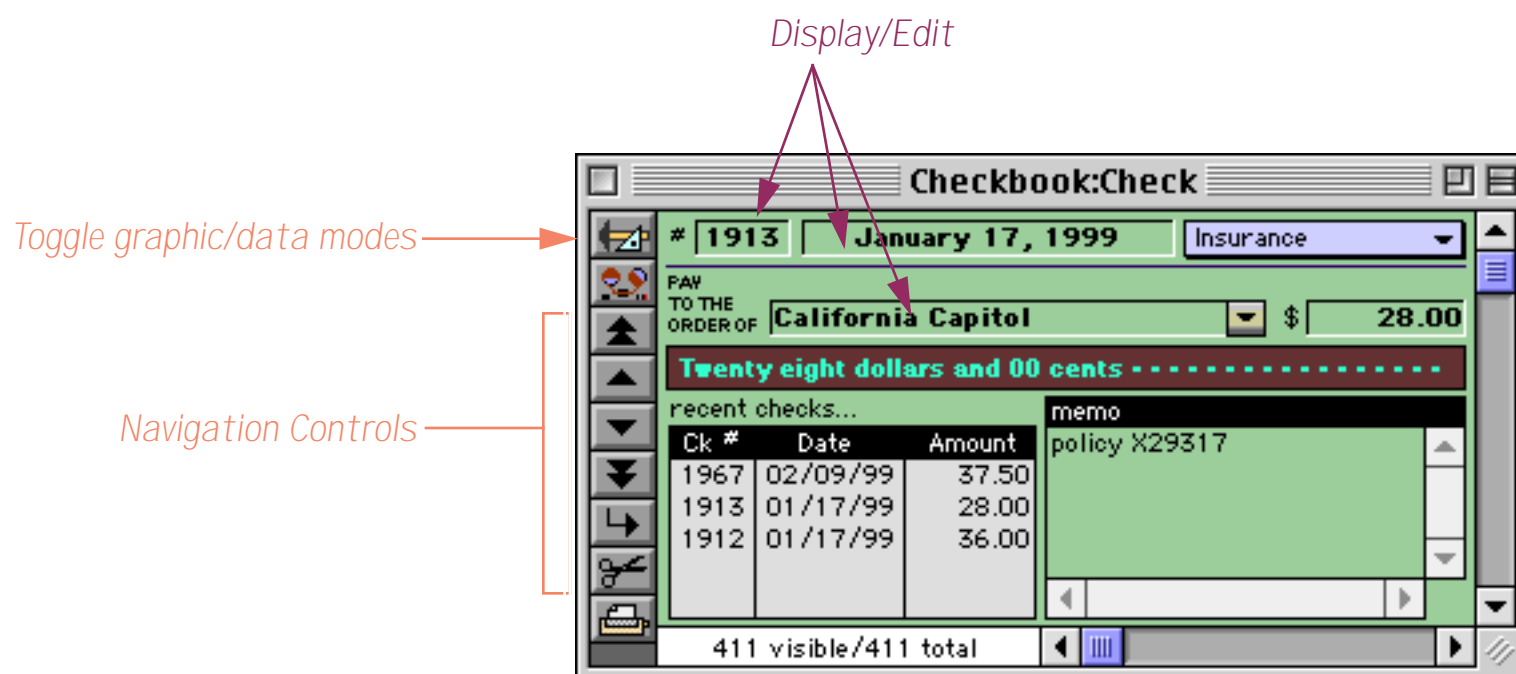


Use the Database menu to view forms, procedures or crosstabs in a different open database.

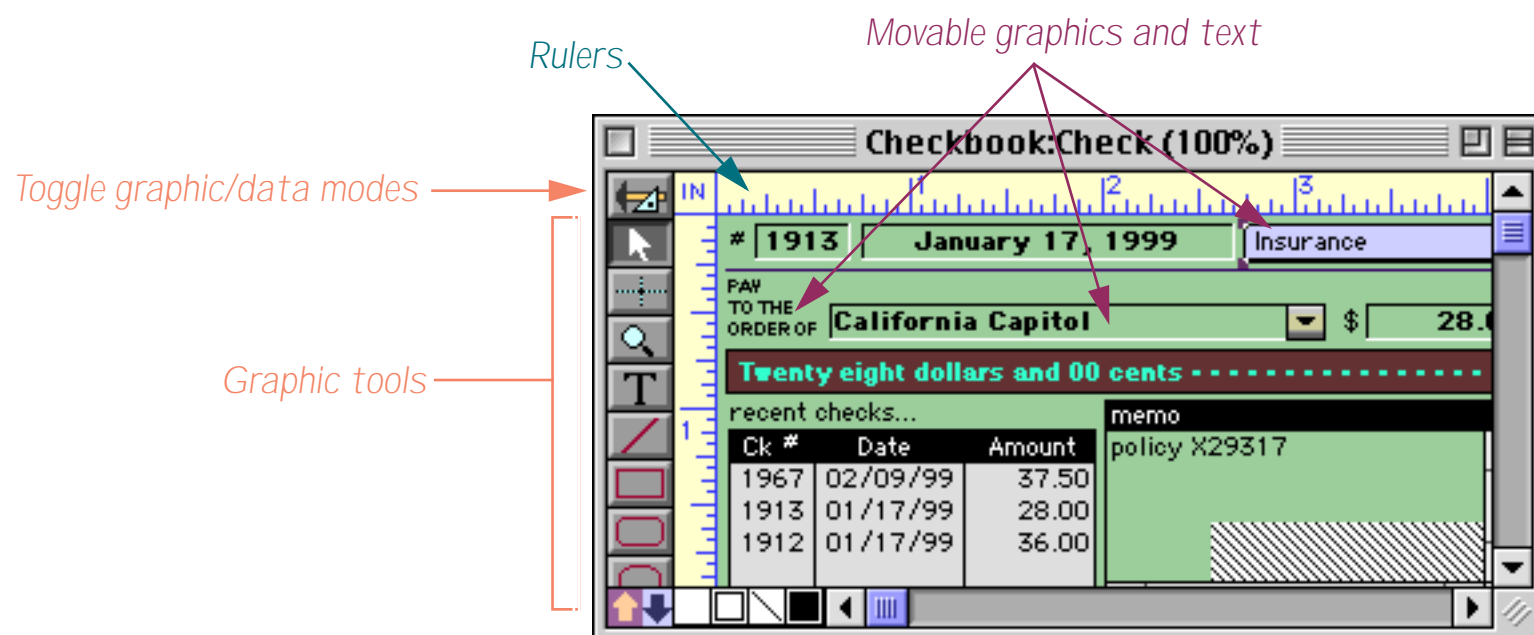



## Form Modes: Data Access vs. Graphic Design

Unlike other views, the Form View operates in two distinct modes—data access and graphic design. Data access mode (also called “data mode”) is the default mode. In this mode you can view and display data, and navigate through the database.



Graphic design mode (also called “graphics mode”) functions like an electronic drafting table. In this mode you design the form by drawing lines, boxes, and other graphic elements. This mode is very similar to many drawing and page layout programs. Graphic design mode is easily recognized by the rulers that appear at the top and left edges of the windows.



To switch between data access and graphic design modes, click on the  tool. Each click on this tool toggles the window between the two modes.

## Form Operation: Individual Pages vs. View-As-List

Panorama allows you to set up blank forms as individual pages or as a continuous sheet (**view-as-list**). When forms are set up as individual pages you see one record at a time. You can flip through the records just as you would shuffle through a stack of paper forms. All of the examples of forms you've seen so far are individual page forms.

A **view-as-list** form displays data as a continuous sheet, as shown below. Instead of flipping from record to record, you scroll up and down through the data in a manner similar to the data sheet. However, unlike the data sheet, a view-as-list form allows you to arrange the data any way you like, and even include graphics in the display. On the other hand, view-as-list forms are slower than the data sheet (because of the overhead in displaying the graphics) and they are much more work to set up.

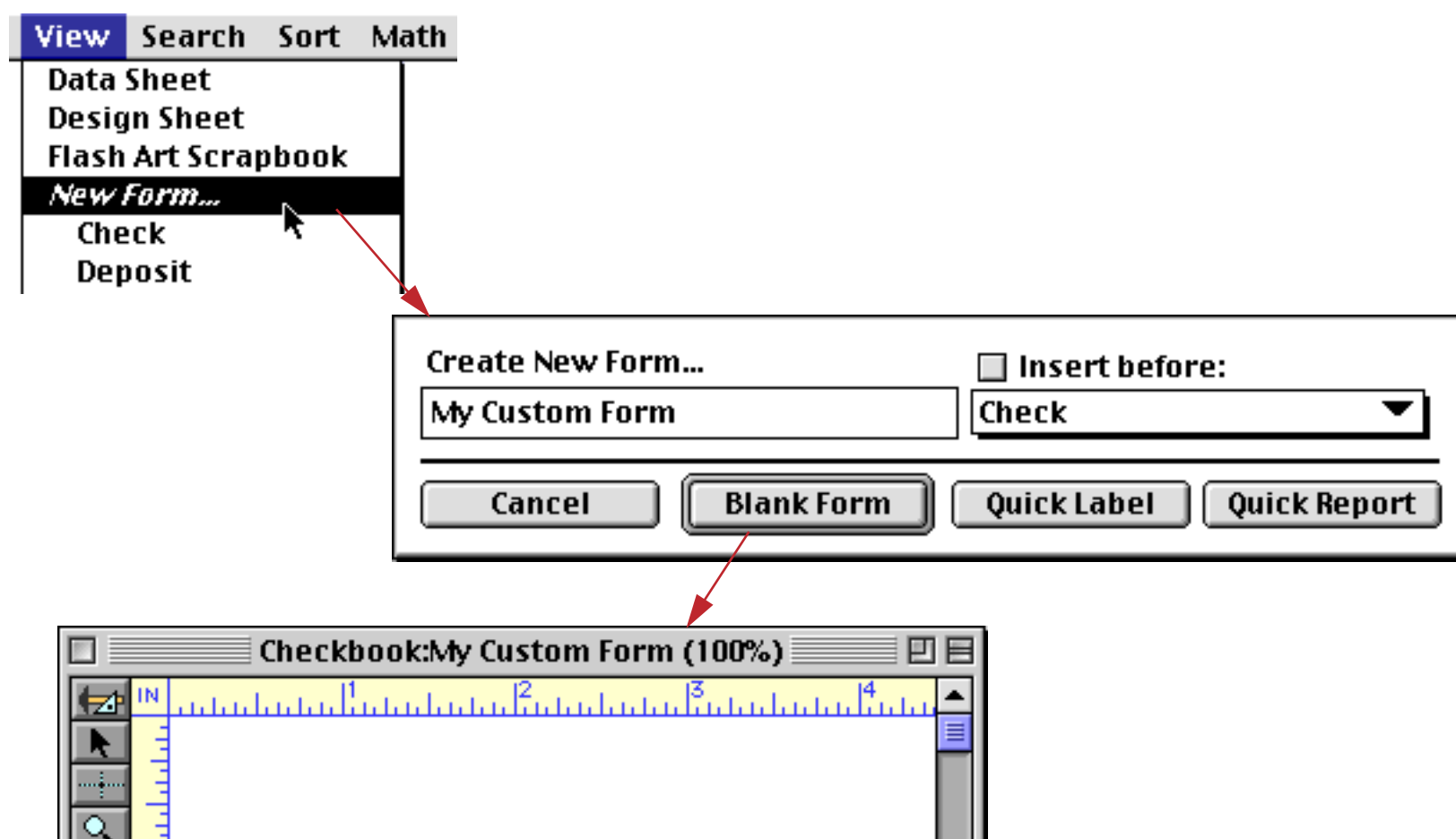
Date	Num/Pay To (Category)	Amount	Balance
01/17/99	1913 California Capitol (Insurance)	28.00	35,023.26
01/17/99	1914 U S Postmaster (Postage)	75.00	34,948.26
01/17/99	1915 Sacramento Bee (Advertising)	795.00	34,153.26
01/18/99	DEPOSIT	+3,846.32	37,999.58
01/22/99	1916 Walthers (Purchases)	12,463.00	25,536.58
01/22/99	1917 Blue Cross Of Calif (Insurance)	279.03	25,257.55
01/22/99	1918 Sherman Douglas Ins (Insurance)	418.60	24,838.95
01/22/99	1919 Cannon Astro (Office Supplies)	145.72	24,693.23
01/25/99	1920	1,885.40	22,807.83

411 visible/411 total

Unless you tell it otherwise, Panorama sets up a new form as individual pages. To convert the form to a continuous sheet you must use the **Form Preferences** command (Setup menu) to set the **View-as-List** option. You will also have to define the boundaries of the form by setting up a data tile (and optional header tile). For more information about setting up view-as-list forms see "[View-As-List Forms](#)" on page 399.

## Creating a New Form, Crosstab or Procedure

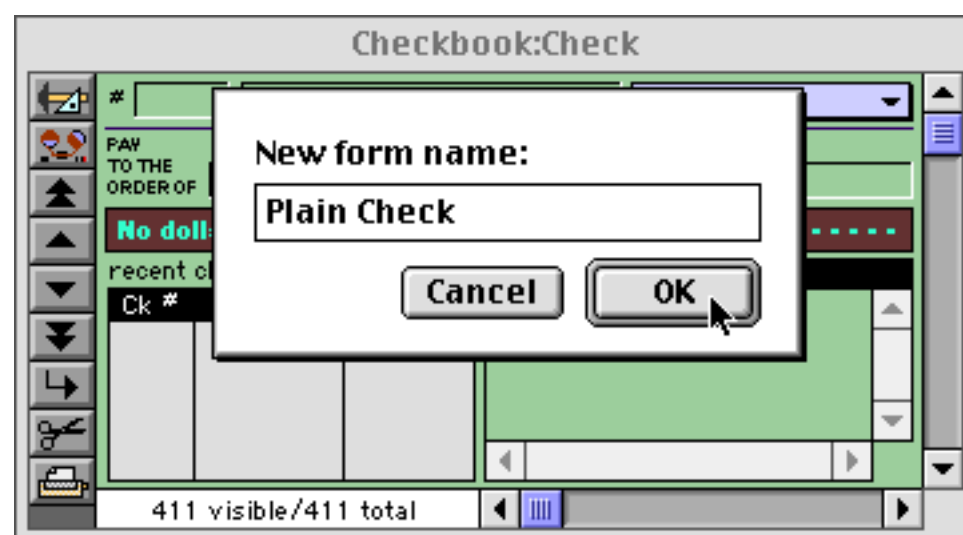
To create a new view, choose **New Form**, **New Crosstab**, or **New Procedure** from the **View Menu**. A dialog box will appear asking you to name the new view. A view name may be up to 25 characters long and can contain any letter, number or punctuation.



When you create a new view, it is usually added to the end of the appropriate section in the **View Menu**. For example, a new form usually becomes the last form in the **View Menu**. If you wish, you can insert the new view into the middle of the **View Menu**. To do this, check the **Insert before** button and use the pop-up menu directly below the **Insert before** button to specify the position of the new view. You can also re-arrange the order of the views using the **Re-Arrange** command in the **Setup menu**.

## Renaming a Form, Crosstab or Procedure

To rename the currently visible form, crosstab or procedure choose **Rename Form**, **Rename Crosstab**, or **Rename Procedure** from the **Setup Menu**. Type in the new name (limit 25 characters) and press **Ok**.



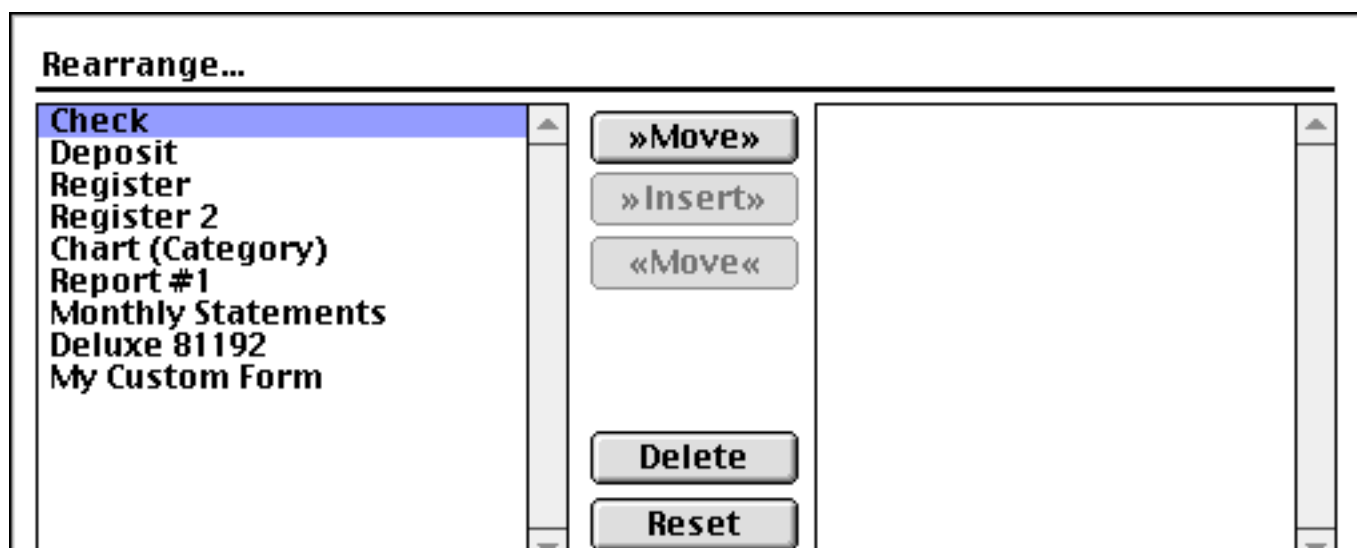
## Deleting a Form, Crosstab or Procedure

To delete a form, crosstab, or procedure choose **Delete Form**, **Delete Crosstab**, or **Delete Procedure** from the Setup Menu. Since you cannot undo after you delete a view, Panorama will ask you if you are sure before it actually deletes the view. Note: If only one view is open and you remove it, Panorama will close the entire file. To avoid this, open an extra window before you delete a view.

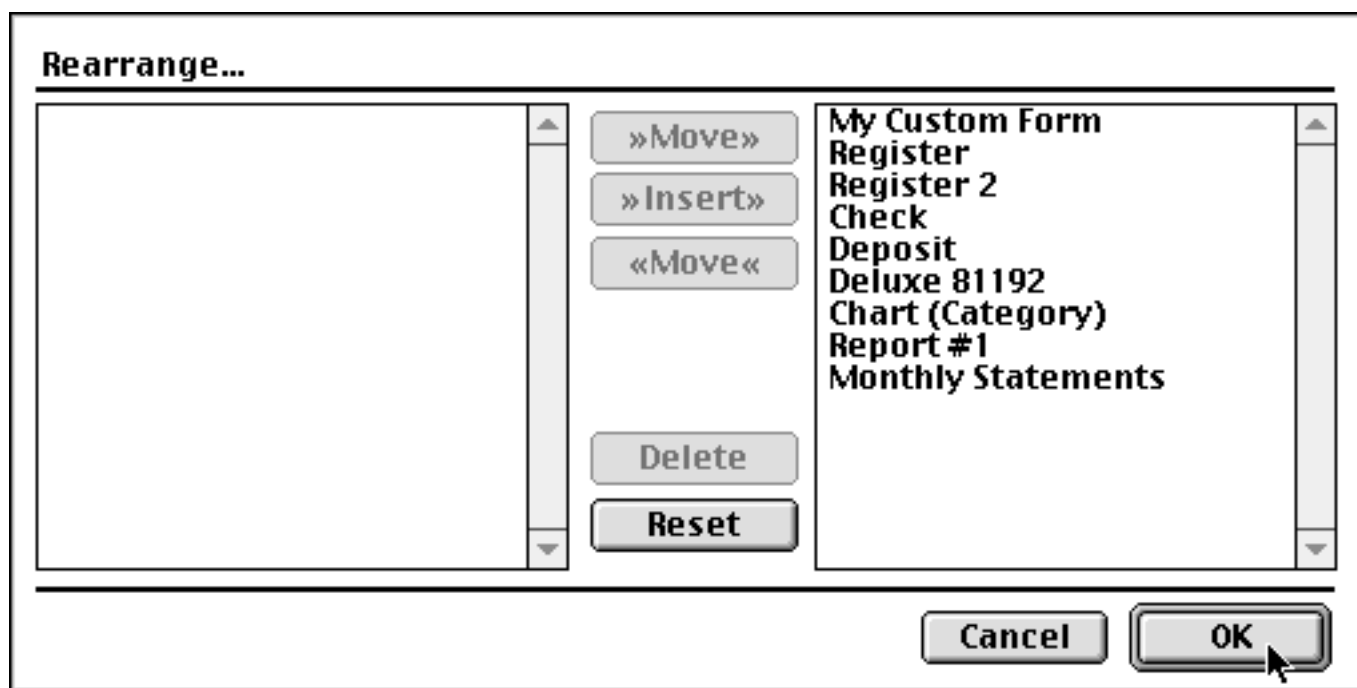
You can also delete views with the **Re-Arrange** command in the Setup menu (described in the next section). The **Re-Arrange** command is the fastest way to remove several views at once (see below).

## Changing the Order of Forms, Crosstabs or Procedures

The **Re-Arrange Forms**, **Re-Arrange Crosstabs**, and **Re-Arrange Procedures** commands in the Setup menu change the order of the items listed in the View Menu. Each of these commands displays a dialog box listing the current view order on the left and the new view order on the right.



This dialog is like a puzzle—the object is to move the views from the left to the right in the order you want. To move a view to the other side, either double-click on it or press the **>>Move>>** button. To insert a view into the middle of the list on the right, first select the spot where you want to insert and then press the **>>Insert>>** button. When all the views have been moved to the right hand side, press the **Ok** button to rearrange the views.



You can also use this dialog to delete views. To delete one or more views, press the **Delete** button instead of moving the view to the right. Warning: You cannot delete an open view using the **Re-Arrange** command. First close the view (form, crosstab or procedure) and then delete the view.





# Chapter 4: Records



The heart of a database is, naturally, the data stored in it. Since storing and organizing data is Panorama's primary task, it has special rules and procedures for handling data.

## Data Organization

Inside each Panorama database the information is organized into records and fields. A record consists of a group of related information. In a personnel database, for example, each record would contain all the information about a single employee. Most databases have anywhere from several dozen to several thousand individual records. The example database shown below has 102 records, 17 of which are currently visible in the window.

First	Last	Title	Company	Address	City	State	Zip
Henry	Hultquist		Lincoln Lumber	1197 S. 17th	Lincoln	NE	68502
Steve	Jackson	Purchasing	Ann Arbor Lumber	389 Worden	Ann Arbor	MI	48103
Glen	Knock		South Portland Lumber	909 Wescott Rd	South Portland	ME	04106
	Kovacs	Owner	Stephen's Appliances	90 Duane Lane	Demarest	NJ	07627
Mike	Kuenning	Sales Manager	Gamma Printing	3 Almy Drive	Malvern	PA	19355
Scott	Lay		Portland Lumber	1278 N.E. 136th	Portland	OR	97230
Wes	Lemarr			57 Hobart Ave	Rutherford	NJ	07070
Jerry	Levan			883 Boone Trail	Richmond	KY	40475
Tom	Long		Austin Lumber	1897 Balcones Drive	Austin	TX	78731
Tom	Love			53 Clubhouse Drive	Woodbury	CT	06798
Nadine	Lucas	Purchasing	N.L. Plumbing	759 2nd Ave	San Francisco	CA	94118
John	Maguire	Vice President	Akron Lumber	39 Beck Ave	Akron	OH	44302
James	Mahan	Owner	J.M. Plumbing	1294 W. 31st	Los Angeles	CA	90018
John	Marshall	President	John's Appliances	6110 Lowbank	Jenison	MI	49428
Don	Meadows	Sales Manager	Austin Lumber	1144 A West 6th	Austin	TX	78703
Charles	Michaels			5238 Quince	Upland	CA	91786
Steve	Miller	President	SM Printing	3894 11th Court	Jupiter	FL	33458

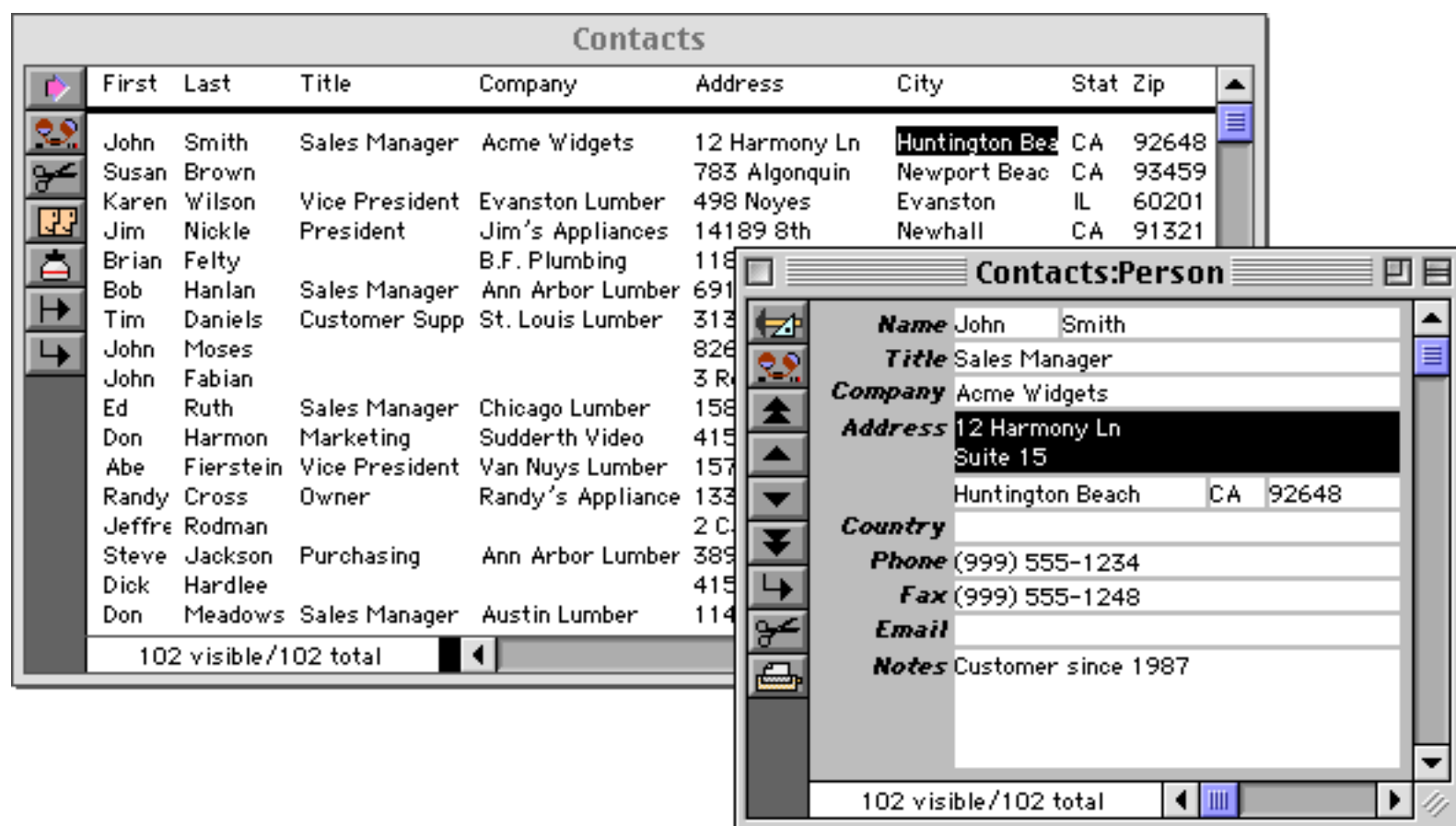
The database is also divided into fields. Each field contains a specific item of information—a street address, a phone number, a birthdate, etc. Most databases have somewhere between five and one-hundred fields (Panorama allows up to 65,000 fields per database).

Every record in a database contains exactly the same fields. If certain records don't use a particular field it can be left empty, but the field itself still exists for every record. For example, notice that in the illustration above some of the Title and Company entries are empty. (However, you can create a form that does not display all of the fields. “[Displaying and Editing Text](#)” on page 283.)

As you work with a database, you will constantly be adding new records, revising and removing old records, and rearranging (sorting, etc.) existing records. Fields can also be added, revised, and removed, but you will do this much less often. Once the fields are set up, you will usually leave them alone.

## Tables vs. Individual Pages

Panorama can display any database either as a table of records or as an individual page for each record. In the table format each line corresponds to a record, and each column corresponds to a field. Common examples include phone directories and price lists. In the individual page format each page corresponds to a single record. Each item on the page corresponds to a field. Common examples include invoices, tax returns, and report cards.



Panorama doesn't care whether the database is displayed as a table or as individual pages. As you can see, the same information is displayed and edited either way. Use the method that seems natural for the database you are working with.

## Special Records

Most database programs treat every record the same way. Panorama, however, distinguishes between three different types of records: **data records**, **summary records**, and **invisible records**. Most of the work you do will be with ordinary data records. Summary and invisible records, however, are the keys to some of Panorama's unique capabilities.

### Data Records

Ordinary records used for data storage are called **data records**. You can create new data records one at a time by keying in information, or you can add many new data records at once by importing data.

## Summary Records

**Summary records** are temporary records used for calculating totals, subtotals, and other summary information. Panorama's **Group** commands automatically create summary records for you. When viewing the database as a sheet you can identify summary records by the light blue background color, and by the fact that they are usually displayed in bold.

Date	CkNum	PayTo	Category	Debit
02/15/99	1979	U S Postmaster	Postage	125.00
02/20/99	1980	U S Postmaster	Postage	80.00
03/20/99	2012	U S Postmaster	Postage	25.00
03/20/99	2013	U S Postmaster	Postage	85.00
05/22/99	2104	U S Postmaster	Postage	115.00
07/16/99	2186	U S Postmaster	Postage	75.00
08/20/99	2239	U S Postmaster	Postage	85.00
		<b>U S Postmaster</b>		<b>850.00</b>
03/28/99	2022	U S Sprint	Telephone	42.31
06/05/99	2152	U S Sprint	Telephone	79.53
		<b>U S Sprint</b>		<b>121.84</b>
02/26/99	1982	University Copy System	Office Supplies	36.20
		<b>University Copy Sys</b>		<b>36.20</b>
02/01/99	1938	Unocal	Auto	182.59
02/09/99	1968	Unocal	Auto	57.62
03/16/99	2007	Unocal	Auto	33.32
05/24/99	2111	Unocal	Auto	119.05
07/16/99	2189	Unocal	Auto	38.11
07/24/99	2213	Unocal	Auto	34.44
08/20/99	2240	Unocal	Auto	89.91
		<b>Unocal</b>		<b>555.04</b>

520 visible/540 total

*summary records*

Summary records may appear in one of seven levels, from 1 to 7. Each higher level is used for a higher level of subtotal. Each summary level can be identified by the deeper shade of blue in the background, as shown here. This database has been grouped into three summary levels.

Date	CkNum	PayTo	Category	Debit
07/24/99	2212	City Of Caboose	Utilities	98.52
09/19/99	2290	City Of Caboose	Utilities	103.15
		<b>City Of Caboose</b>		<b>488.57</b>
02/09/99	1973	S C E	Utilities	172.03
03/29/99	2043	S C E	Utilities	89.46
05/07/99	2083	S C E	Utilities	96.26
05/24/99	2118	S C E	Utilities	97.00
06/05/99	2154	S C E	Utilities	157.31
06/14/99	2161	S C E	Utilities	56.27
08/13/99	2235	S C E	Utilities	86.53
09/19/99	2291	S C E	Utilities	81.13
		<b>S C E</b>		<b>835.99</b>
02/09/99	1972	So. Calif. Gas Co.	Utilities	136.33
03/29/99	2042	So. Calif. Gas Co.	Utilities	217.32
05/07/99	2085	So. Calif. Gas Co.	Utilities	86.74
05/24/99	2117	So. Calif. Gas Co.	Utilities	134.99
09/19/99	2292	So. Calif. Gas Co.	Utilities	154.95
		<b>So. Calif. Gas Co.</b>		<b>730.33</b>
			<b>Utilities</b>	<b>2,054.89</b>
				<b>183,651.22</b>

542 visible/543 total

*level 1 (company)*

*level 2 (category)*

*level 3 (grand total)*

Panorama allows you to treat summary records as a collapsible outline. You can use the **Outline Level** command to collapse the outline to show only high level summaries, then use the **Expand**, **Expand All**, and **Collapse** tools to expose the detail you need to see. For more about Panorama's outline capability, see "[Summaries and Outlines](#)" on page 371.

The screenshot shows a window titled "Checkbook" with a table of transactions. The table has columns for Date, CkNum, PayTo, Category, and Debit. Summary rows are shown in bold. On the left side of the table, there are three red arrows pointing to the expand/collapse icons, labeled "Expand", "Expand All", and "Collapse".

Date	CkNum	PayTo	Category	Debit
			<b>Printing</b>	<b>188.96</b>
			<b>Purchases</b>	<b>66,217.17</b>
			<b>Rent</b>	<b>35,026.34</b>
		<b>Airborne Express</b>		<b>35.40</b>
		<b>AIRS</b>		<b>138.07</b>
		<b>American Customs H</b>		<b>34.00</b>
		<b>B J D Trucking Inc.</b>		<b>37.50</b>
		<b>Burlington Air Expre</b>		<b>95.17</b>
		<b>Consolidated Freight</b>		<b>150.20</b>
01/30/99	1929	Federal Express	Shipping	178.75
03/20/99	2015	Federal Express	Shipping	170.00
03/28/99	2032	Federal Express	Shipping	150.00
05/14/99	2101	Federal Express	Shipping	170.00
		<b>Federal Express</b>		<b>668.75</b>
		<b>U P S</b>		<b>769.11</b>
			<b>Shipping</b>	<b>1,928.20</b>
			<b>Taxes</b>	<b>5,152.79</b>
			<b>Telephone</b>	<b>5,690.50</b>
			<b>Utilities</b>	<b>2,054.89</b>
				<b>183,651.22</b>

29 visible/563 total

Summary records are designed to have a very short lifetime—usually only a few minutes. When you want to calculate subtotals or other summaries you'll create new summary records. After you've examined (and possibly printed) the summaries, you'll use the **Remove Summaries** command to remove them so you can get back to regular work with your database. For more information about summary records, see "[3-Step Summarizing](#)" on page 183.

### Invisible Records

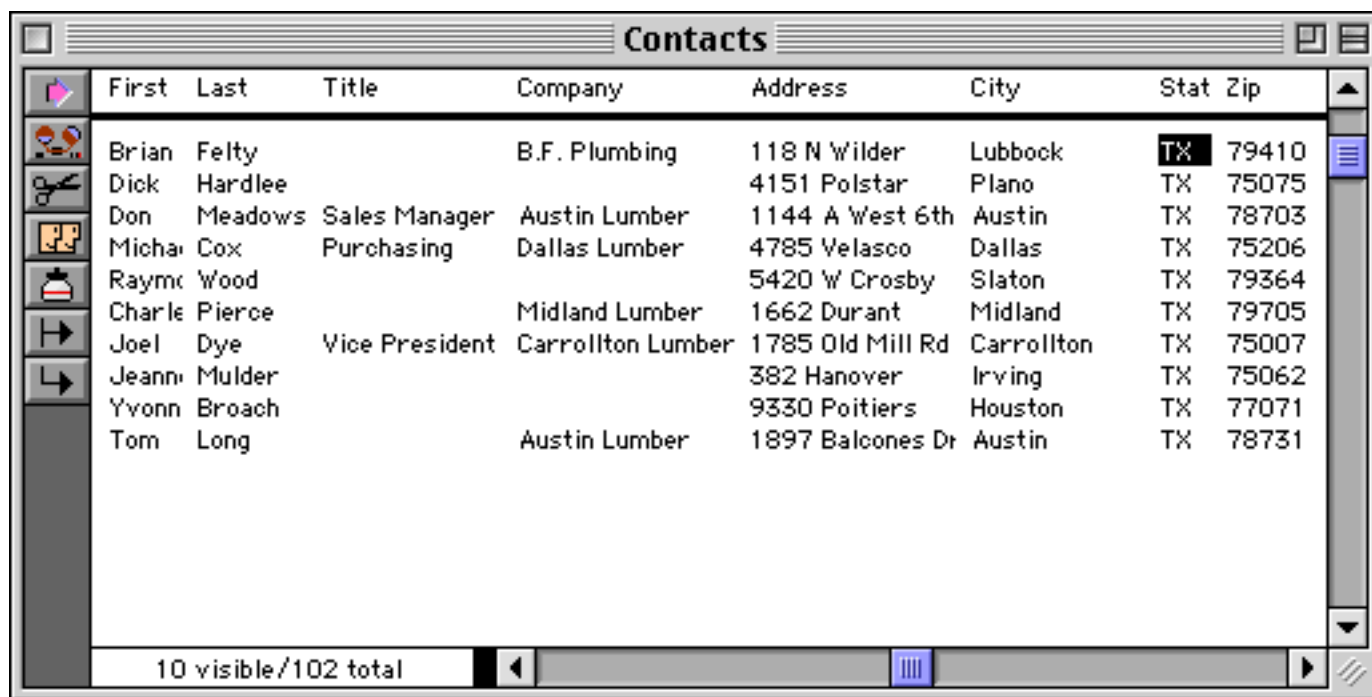
**Invisible records** are ordinary data or summary records that have been made temporarily invisible. For instance right now you might be interested only in sales made in California, transactions over \$250,000, or invoices over 45 days old. Panorama's **Find/Select** commands allow you to choose the data you want to see and make the rest temporarily invisible. This allows you to see just the information you are interested in without the other data getting in the way.

The following illustrations show a typical example. The original database started out with 102 records. We can use the **Find/Select** command to make 92 of these records temporarily invisible.

The screenshot shows a dialog box with the following elements:

- Buttons: Find ⌘S, Select ⌘F, Select Within ⌘L, Select Additional ⌘M, Cancel
- Field 1: State
- Field 2: Contains
- Field 3: TX

When the **Select** button is pressed, only records in Texas remain visible. All other records become temporarily invisible.



First	Last	Title	Company	Address	City	Stat	Zip
Brian	Felty		B.F. Plumbing	118 N Wilder	Lubbock	TX	79410
Dick	Hardlee			4151 Polstar	Plano	TX	75075
Don	Meadows	Sales Manager	Austin Lumber	1144 A West 6th	Austin	TX	78703
Michael	Cox	Purchasing	Dallas Lumber	4785 Velasco	Dallas	TX	75206
Raymond	Wood			5420 W Crosby	Slaton	TX	79364
Charles	Pierce		Midland Lumber	1662 Durant	Midland	TX	79705
Joel	Dye	Vice President	Carrollton Lumber	1785 Old Mill Rd	Carrollton	TX	75007
Jeannette	Mulder			382 Hanover	Irving	TX	75062
Yvonne	Broach			9330 Poitiers	Houston	TX	77071
Tom	Long		Austin Lumber	1897 Balcones Dr	Austin	TX	78731

At any time you can make all records visible again with the **Select All** command. You can also make a new selection at any time. For more information about invisible records, see [“Finding vs. Selecting”](#) on page 159.





# Chapter 5: Fields



The information stored in a database is organized into records and fields. Each field contains a specific category of information—names, phone numbers, birthdates, etc. Your first task after creating a new database is to decide how many and what fields are needed to get the job done. It's somewhat like designing a house—you have to decide what the best configuration is. How many bedrooms you need? How many baths? Will you need an office? And just like architecture, there are sometimes trade-offs that have to be made in designing a database.

For some database jobs it is extremely obvious how the fields should be set up. But usually you have more flexibility than you might think. Take a simple name and address list, probably one of the most basic database applications. How should the name be stored? All in one field? Or should there be separate fields for first and last names? What about Mr./Ms./Mrs., should that be in a separate field? There are no hard and fast answers—it depends on how you want to use the database. For this example (names), data entry will probably be easier if you use a single field. On the other hand, you'll have more options for organizing and formatting if you split the name into several fields. The choice is up to you.

One thing to keep in mind is that if you make a mistake, it is much easier to combine two fields together than it is to split a single field into two. For example, it is quite easy to take separate first and last name fields and combine them, but if you type the names into a single field it could be quite difficult to later split them apart. If you have any doubt, it is better to err on the side of separating the data into more fields.

You can add new fields, remove fields, or change the properties of fields at any time, even after you've filled the database with data. Once you start entering data, however, it can be more work to re-arrange the fields. Some extra planning before you start entering data can pay big dividends in the long run.

## The Setup Menu

The simple way to add or remove fields is to use the **Add Field** and **Delete Field** commands in the Setup Menu. You can also change most of the properties of a field (name, type of data, display format etc.) with the **Field Properties** command. These commands are available at any time when you are using the data sheet, and are also available when you are designing a form (graphic design mode).



In the data sheet you can also open the **Field Properties** dialog by double clicking on a field name.

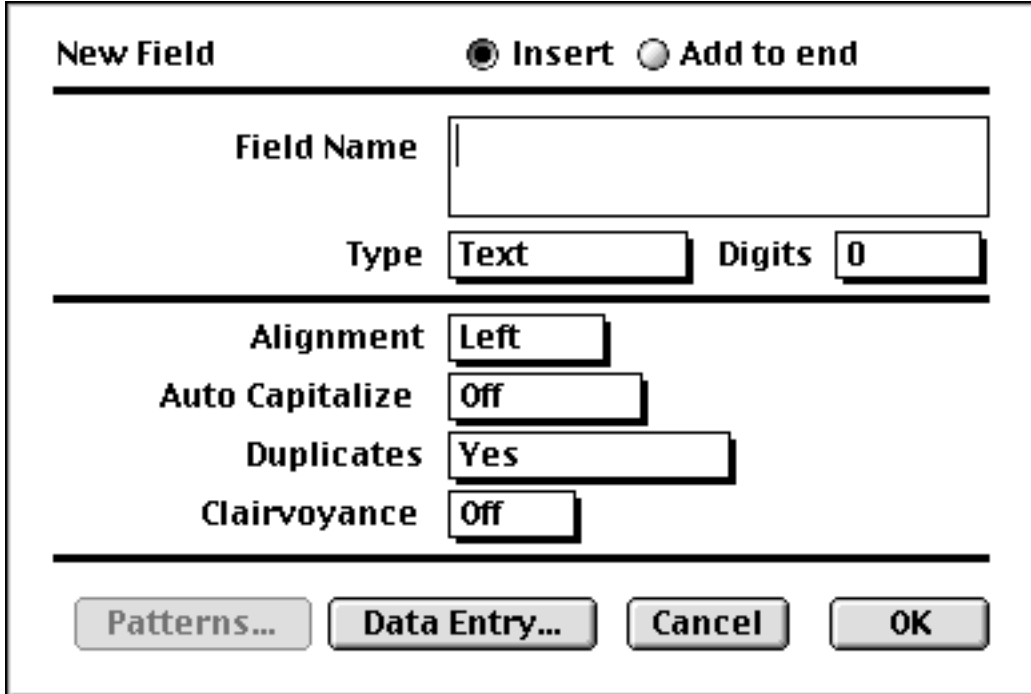
*double click column title to open field properties dialog*



First	Last	Title	Company	Address	City	Stat	Zip
John	Smith	Sales Manager	Acme Widgets	12 Harmony Ln	Huntington Bea	CA	92648
Susan	Brown			783 Algonquin	Newport Beac	CA	93459
Karen	Wilson	Vice President	Evanston Lumber	498 Noyes	Evanston	IL	60201
Jim	Nickle	President	Jim's Appliances	14189 8th	Newhall	CA	91321
Brian	Feltv		B.F. Plumbina	118 N Wilder	Lubbock	TX	79410

## Add Field

To add a new field, use the **Add Field** command in the Setup Menu.



**New Field**       **Insert**    **Add to end**

---

**Field Name**

**Type**    **Digits**

---

**Alignment**

**Auto Capitalize**

**Duplicates**

**Clairvoyance**

---

You can add the new field at the end (the right edge of the data sheet) or you can insert the new field in the middle of the database. In addition to the field name, you can use pop-up menus to set up the properties of the new field. The **Type** pop-up tells Panorama what kind of data you expect to store in the field—**text**, **numeric**, etc. See “[Setting Up a Field’s Data Type](#)” on page 114 for a full explanation of data types and the number of digits. The **Alignment** pop-up controls how the field is aligned in the data sheet—**left**, **center**, or **right** flush. The **Auto Capitalize**, **Duplicates**, and **Clairvoyance** pop-up menus turn on various data entry options—see “[Data Entry Accelerators](#)” on page 135 for more information.

Use the **Patterns** dialog to set up the display format for numbers and dates. Output patterns are discussed in detail in “[Numeric Output Patterns](#)” on page 117 and “[Date Output Patterns](#)” on page 121.

The dialog box is titled "Numeric Output Pattern:" and contains the following elements:

- A text field for the pattern, containing "#,".
- A "Sample:" text field containing "6,044".
- A section titled "Number of Digits:" with two sub-sections:
  - "Before Decimal Point" with a dropdown menu showing "Auto".
  - "After Decimal Point" with a text field containing "0".
- Radio button options for the number format:
  - Fixed Decimal Point
  - Use Thousands Separator
  - Scientific Notation
  - Dollars and Cents
- Text fields for "Prefix" and "Suffix", both currently empty.
- A section titled "Negative Values:" with three radio button options:
  - 1234
  - 1234-
  - (1234)
- "Cancel" and "OK" buttons at the bottom right.

Use the **Data Entry** dialog to set up the **Input Pattern**, **Default Value**, **Range**, and **Space Bar Tab**. These options are described in detail in Chapter 7, Data Entry & Editing.

The dialog box is titled "Data Entry Options" and contains the following elements:

- An "Input Pattern:" dropdown menu.
- A "Default Data:" text field.
- Radio button options for default data:
  - Automatic increment by one
  - Today's Date
  - Ditto
- "Character Range:" section with radio button options:
  - Any
  - Alphabetic
  - Numeric
  - AlphaNumeric
- A "Custom:" section with a list box containing the characters A, B, C, D, and E.
- "Space Bar:" section with two checkbox options:
  - Same as Tab
  - Same as Tab when pressed twice
- "Cancel" and "OK" buttons at the bottom right.

## Field Properties

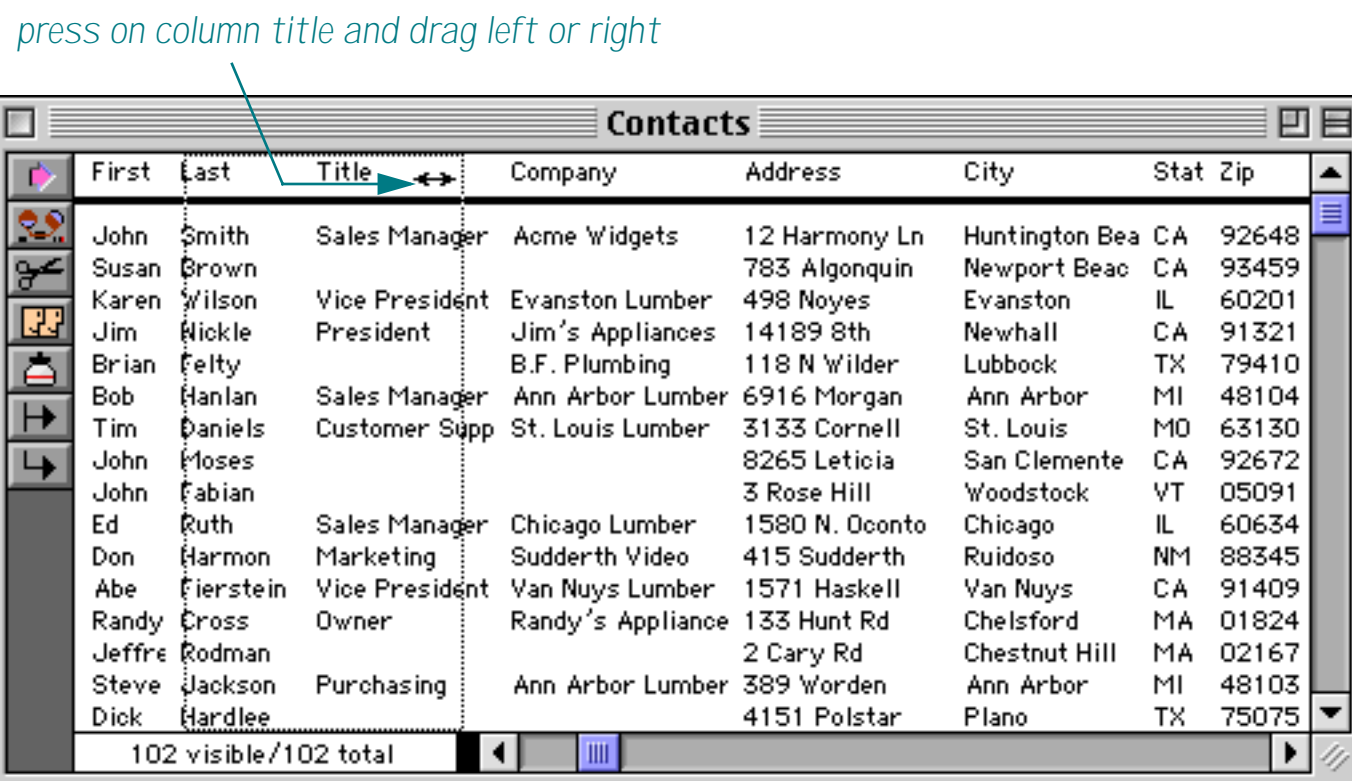
To change the properties of an existing field first select a cell in that field and then choose **Field Properties** from the Setup Menu. (If you are using the data sheet, you can also open the dialog by double clicking on a column name.) This dialog is almost identical to the **Add Field** dialog, and allows you to change the field name, data type, etc.

## Delete Field

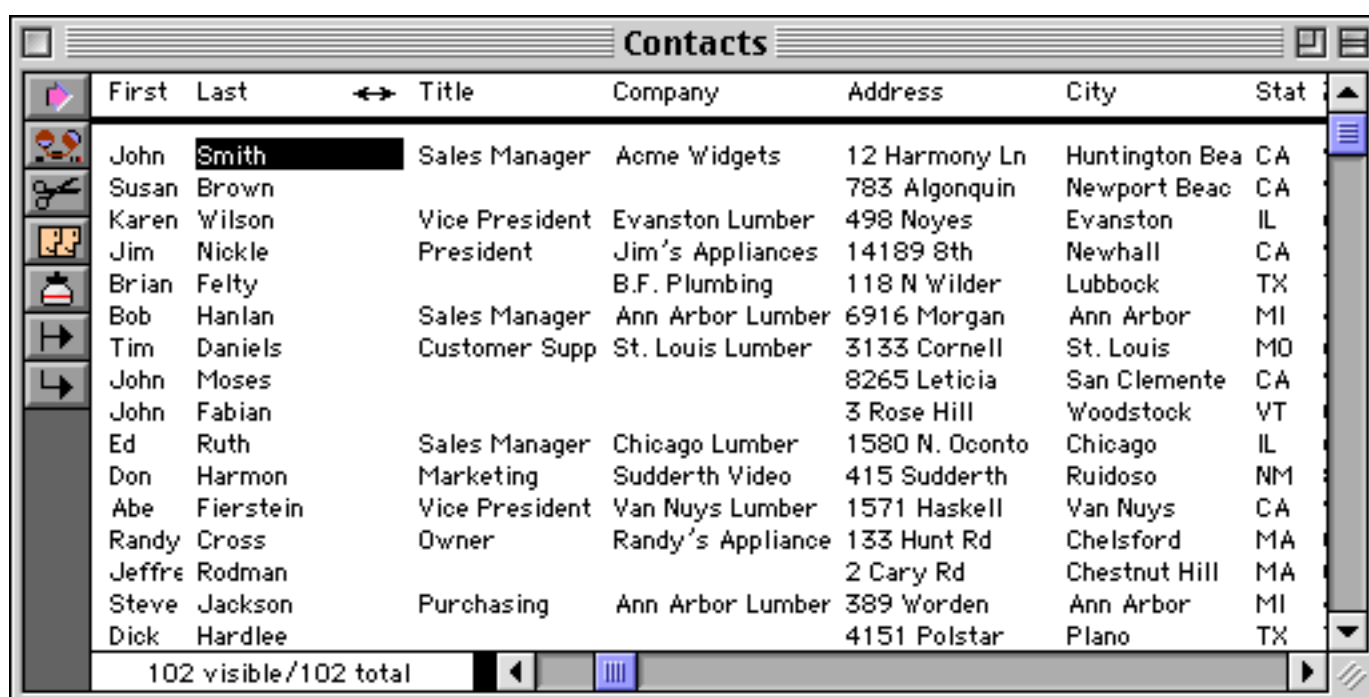
To delete a field from the database, select a cell in the field and choose **Delete Field** from the Setup Menu. (If you are using the data sheet, you can also press **Command-Delete** (Mac) or **Control-Delete** (Windows) to delete a field.) This not only deletes the field, it also deletes any data in the field. If the field contains data, Panorama will warn you that it is about to delete the field. You must confirm that you really want to delete the field before Panorama will proceed. You cannot **Undo** this operation, so be careful!

## Changing the Width of a Field

To change the width of a field in the data sheet, move the mouse over the column name. When you move the mouse over the column name, the cursor will turn into a double headed arrow. When you press the mouse, a gray box appears around the column. Drag the mouse left or right to expand or shrink the column width, then release the mouse when the field is the correct width.



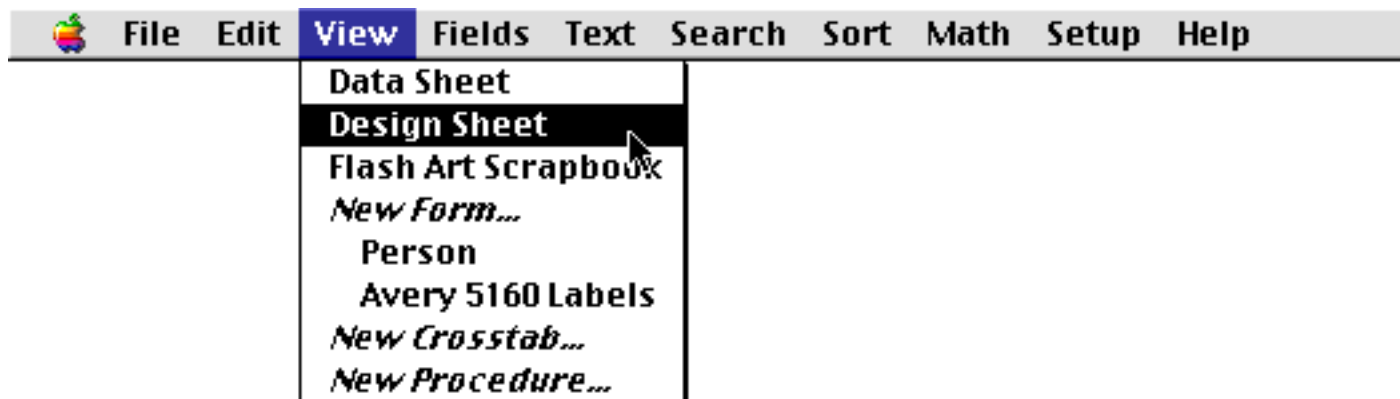
When the mouse is released the column width is adjusted.





## The Design Sheet

The Setup Menu provides an easy way to add or remove a few fields at a time. The disadvantage of the Setup Menu is that it only allows you to see a little piece of the database structure at a time. To get a more comprehensive view you'll need to open the actual blueprints of the database—the **design sheet**. Like other parts of the database, the design sheet is accessible from the View Menu.



The design sheet shows all the fields and their properties. Like DNA, the design sheet contains all the information about how the database is organized.

Field Name	Type	Di	Align	Out	Inp	Range	Choi	Link	Clair	Tab	Cap	Dup	Def	Equ	Reac	Writ	Wid	Notes
Date	Date	0	Left			Any		Off	Off	Off	Yes	tod.		0	0	7		
CkNum	Num	0	Right			Any		Off	Off	Off	Yes	+		0	0	4		
PayTo	Text	0	Left			Any		On	Off	Wor	Yes			0	0	20		
Category	Text	0	Left			Any		On	Off	Wor	Yes			0	0	10		
Debit	Num	2	Right	#,.		Any		Off	Off	Off	Yes			0	0	10		
Credit	Num	2	Right	#,.		Any		Off	Off	Off	Yes			0	0	9		
Balance	Num	2	Right	#,.		Any		Off	Off	Off	Yes			0	0	9		
Memo	Text	0	Left			Any		Off	Off	Off	Yes			0	0	22		

### Database “Generations”

When DNA mutates, the change doesn't take effect until the next generation. Panorama's design sheet works the same way. The changes you make to the design sheet don't immediately change the database. Instead, Panorama waits for you to tell it to create a “new generation” of the database. This allows you to make multiple changes to the design sheet, check the changes for accuracy, and then apply all of the changes at once to the actual database structure.

There are three ways to tell Panorama to create a new generation.

1. Click the New Generation tool.



2. Switch the window to a different view (using the View Menu).

3. Close the design sheet window.

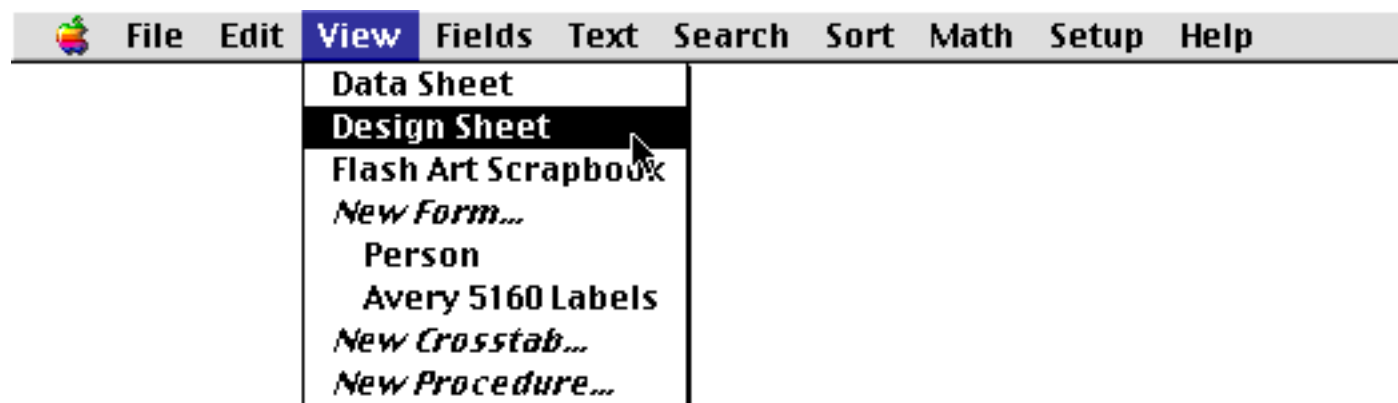
Notice that you don't have to close the design sheet window to create a new generation. If you wish, you can even leave the design sheet open as you work with the database. This allows you to make a change to the design sheet, then quickly test the change and make further changes if necessary.

Once you have created a new generation, you cannot go back to the old generation with the **Undo** command. However, you can go back to the last generation saved on the disk with the **Revert to Saved** command (See [“Revert to Saved”](#) on page 87).

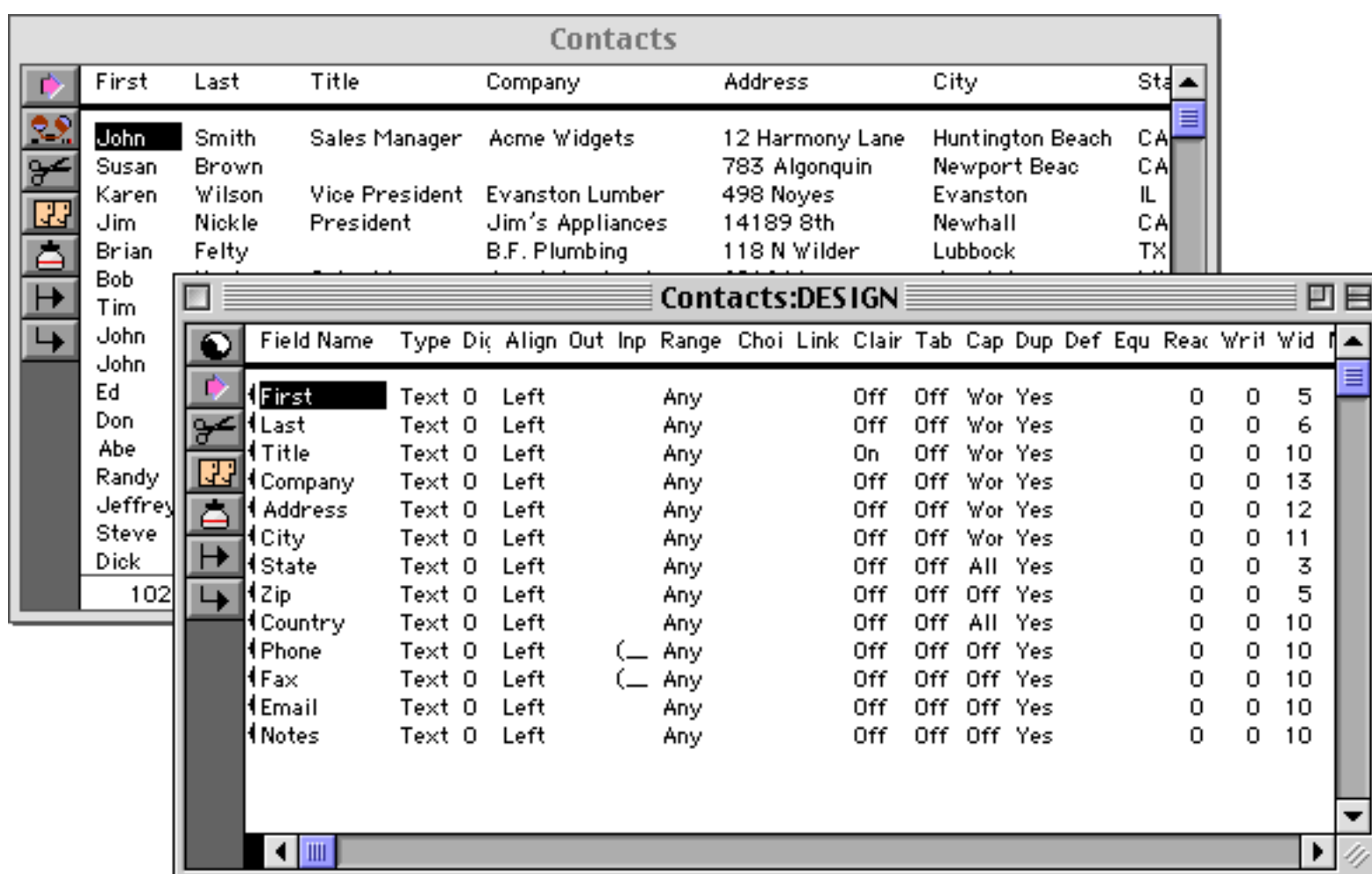
### Typical Design Sheet Operation

Once you get the hang of it, the data sheet is very simple to use. As an example, we'll show how to add two new fields to a **Contacts** database using the design sheet. We'll add fields for a prefix (**Mr./Ms./Mrs.**) and for a middle name.

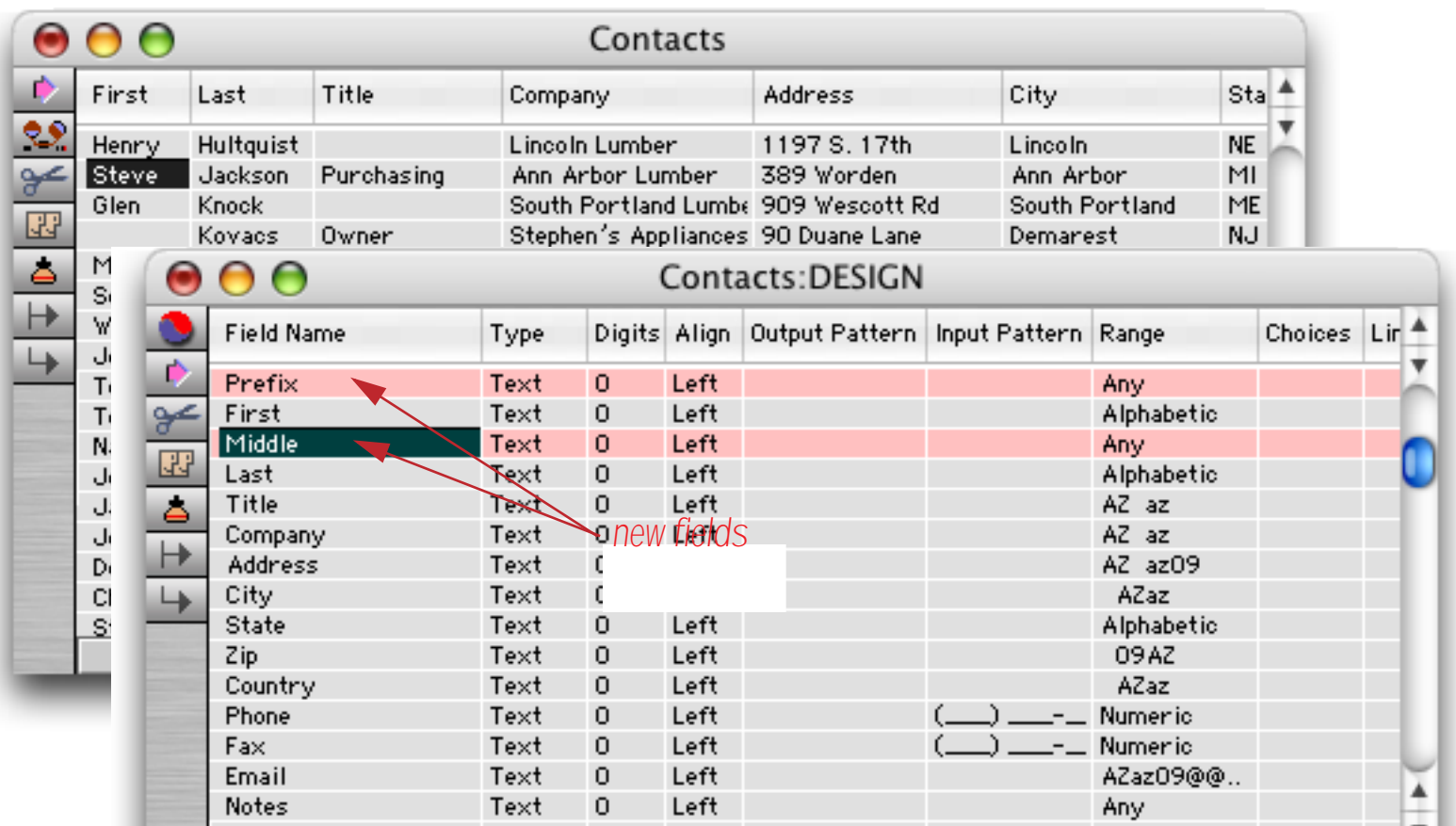
The first step is to open the design sheet using the View Menu.



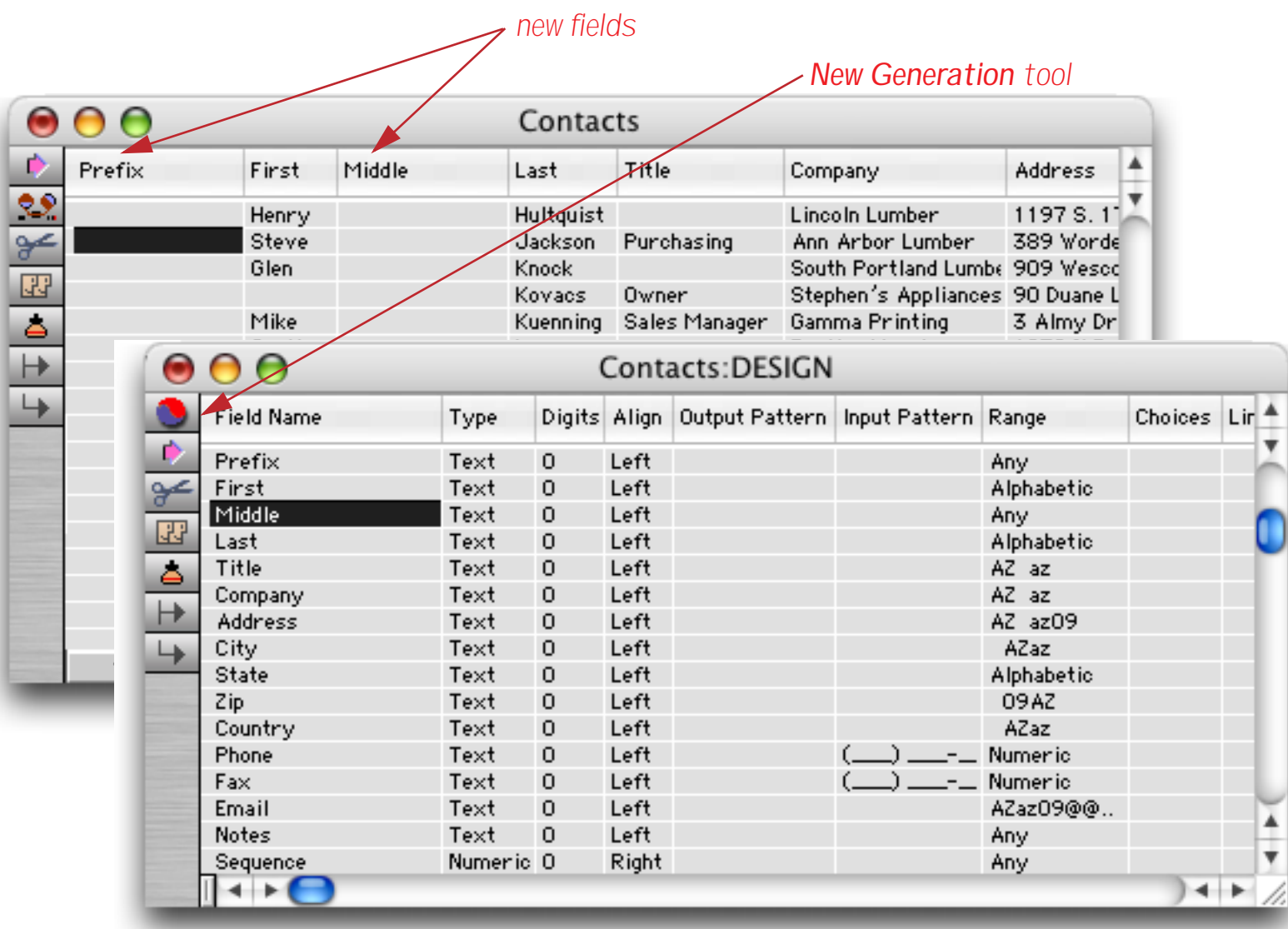
If you hold down the **Control** key (Mac) or **Alt** key (Windows) the design sheet will open in a separate window, allowing you to see both the data sheet and the design sheet at the same time (see [“Opening More Than One Window Per Database”](#) on page 87). In the illustration below we've dragged the design sheet window down and to the right so that we can see both windows at once.



Now we'll add the two new fields to the design sheet. We use the **Insert New Record** tool to insert the new lines, then type in the field names. The new fields are displayed with a pink background to show that they have not been added to the database yet.



Once the fields are entered, click on the **New Generation** tool to update the database itself.



The new fields are ready to use—you can bring the data sheet forward and start entering data in them right away. It's not necessary to close the design sheet first—you can leave it open in case you need to make further modifications to the fields. It's usually a good idea, however, to close the design sheet when you are not going to be using it again for a while.

Wait just one minute! There's one final step you don't want to forget. When you're sure you've got the structure you want, make sure you **Save** the database to make your changes permanent!

## Field Properties

Each row in the design sheet contains all the properties for a single field in the database. The design sheet contains 19 columns—one column for each field property.

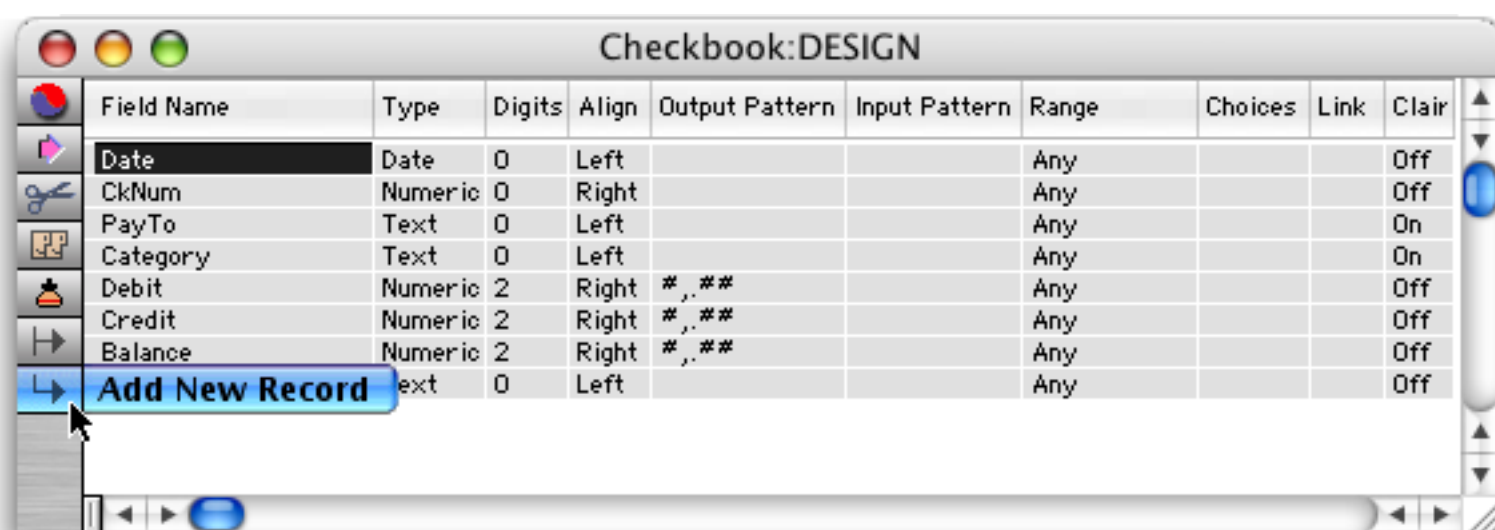
Field Name	The field name identifies the field. It helps you remember what is in the field, and is also used to identify the field in formulas. Panorama does not place any restrictions on your choice of field names, but there are some ramifications to using an unusual name (see <a href="#">“Rules for Field Names”</a> on page 111).
Type	This column specifies the type of data stored in each field: text, numeric, date, choices, or picture. See <a href="#">“Data Types”</a> on page 113 for more information.
Digits	This column specifies the number of digits to be used after the decimal point with numeric data: <b>0, 1, 2, 3, 4, Float</b> or <b>Money</b> . See <a href="#">“Numeric Data”</a> on page 116 for more information.
Align	This column specifies how the field should be aligned in the data sheet: left, center, or right flush. Usually left flush is used for everything except numbers, which are displayed right flush.
Output Pattern	The output pattern allows you to specify the display format for numbers or dates. <a href="#">“Numeric Output Patterns”</a> on page 117 and see <a href="#">“Date Output Patterns”</a> on page 121 for more information.
Input Pattern	The input pattern forces data into a specific pattern as it is entered, for example phone numbers or social security numbers. See <a href="#">“Input Patterns”</a> on page 139 for more information.
Range	This column allows you to restrict the characters that can be entered in a field. For example, you can restrict a field to only allow alphabetic or numeric entry. See <a href="#">“Restricting Character Types”</a> on page 141 for more information.
Choices	This column allows you to specify a list of choices that are valid for this field; for instance <b>Yes/No</b> , <b>Gold/Silver/Bronze</b> or <b>Regular/Unleaded</b> . The list of choices is used by the Choices data type and is also used by the Choice Palette.
Link	This column allows you to link this field with a field in another database. Only Clairvoyance <sup>®</sup> is affected by this link. You can set up this field with the Clairvoyance Link command. For more information see <a href="#">“Clairvoyance<sup>®</sup> Across Multiple Files”</a> on page 289 of the Panorama Handbook.
Clairvoyance <sup>®</sup>	This column controls Panorama's Clairvoyance feature. This feature tries to anticipate what you are about to type, then types it for you. See <a href="#">“Clairvoyance<sup>®</sup>”</a> on page 137 for the straight scoop.
Tabs	This column controls the Space Bar Tab feature. This feature makes the <b>Space Bar</b> work just like the <b>Tab</b> key, saving wear and tear on your left pinky.
Caps	Use this column to tell Panorama to automatically capitalize data entry in a field. Panorama can automatically capitalize everything, or just the first letter of each word or sentence. See <a href="#">“Automatic Capitalization”</a> on page 136.
Dups	This column specifies whether or not you want to allow duplicate entries in this field. You can also specify that you want to require duplicate entries (no unique values). See <a href="#">“Checking for Duplicate Data”</a> on page 136.
Default Value	This column allows you to specify a default value for the field when a new record is created. See <a href="#">“Default Values”</a> on page 144.

Equation	This column allows you to specify one or more calculations to be performed whenever the information in this field changes or is confirmed. For example, an invoice can be set up so that all totals are calculated whenever a quantity or price is entered or changed. See “ <a href="#">Automatic Calculations</a> ” on page 149.
Read	This attribute is used to control the security level for displaying (reading) the data in this field. The value in this field may be from 0 (anyone can see this data) to 255 (only users with the highest possible security level can see this data). For more information on security levels see the Panorama Security Handbook, available separately.
Write	This attribute is used to control the security level for modifying (writing) the data in this field. The value in this field may be from 0 (anyone can modify this data) to 255 (only users with the highest possible security level can modify this data). For more information on security levels see the Panorama Security Handbook, available separately.
Width	This contains the approximate width (in characters) of the field in the data sheet. For example, a width of 20 means that the column is about 20 characters wide (the actual width depends on the font and size). Although you can use the design sheet to set the column width, it is easier and more exact to set the width by dragging on the column name (See “ <a href="#">Changing the Width of a Field</a> ” on page 104).
Notes	You can use this field to keep notes about the field. If a database contains dozens or hundreds of fields, it may be difficult to remember what each field is for. You can use this field to store reminders to yourself about the purpose and use of each field. Panorama ignores the contents of this column.

### Adding New Fields Using the Design Sheet

In addition to changing the properties of existing fields, the design sheet can be used to add new fields or to delete existing fields. The design sheet is especially handy when you need to add a lot of new fields at once.

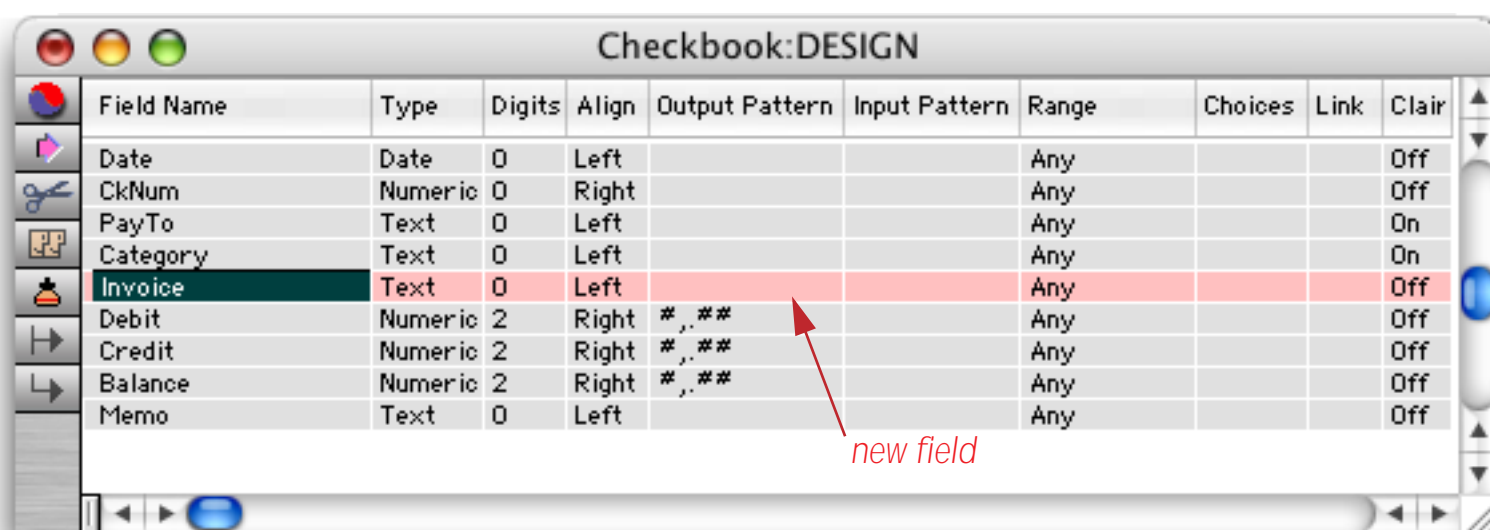
Since each line in the design sheet corresponds to a field, you can add new fields simply by adding new lines to the design sheet. The design sheet works the same way as the data sheet—you can add new lines using the **Add New Record** tool, the **Insert New Record** tool, or by pressing the **Return** key. (Keep in mind that each **record** in the design sheet corresponds to a **field** in the database, so adding or inserting a record here is equivalent to adding or inserting a field in the database itself.)



Remember that adding a line to the design sheet does not immediately create the new field. When you are ready to actually add the new field(s) to the database, tell Panorama to create a new generation (See “[Database “Generations”](#)” on page 105).



When you add a new line to the design sheet, you may notice that your new line has a pink background. This pink background serves as a reminder of the new fields you have created.



As soon as you create a new generation, the background of any new fields will change to gray. On the other hand, if you close the design sheet without creating a new generation, the new fields will not be created.

### Removing Fields Using the Design Sheet

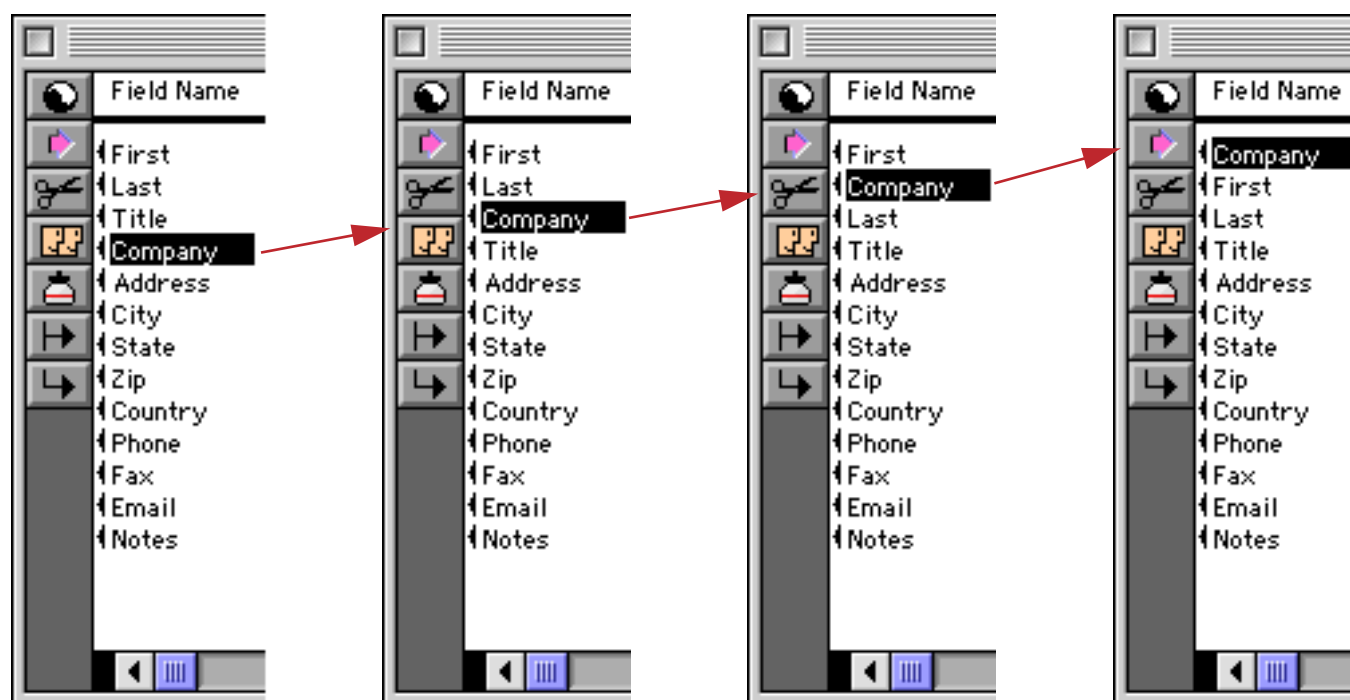
To remove a field from the database simply remove the corresponding line from the design sheet and then tell Panorama to create a new generation. You can delete a field using the **Cut Field** tool or by pressing the **Delete** key (Mac) or the **Backspace** key (PC).

When you remove a field, all the data in that field is lost forever (unless of course, you have another copy on the disk). Remember, you cannot go back to an old generation with the **Undo** command. Therefore, you'll want to be very sure that you really don't need a field anymore before you remove it.

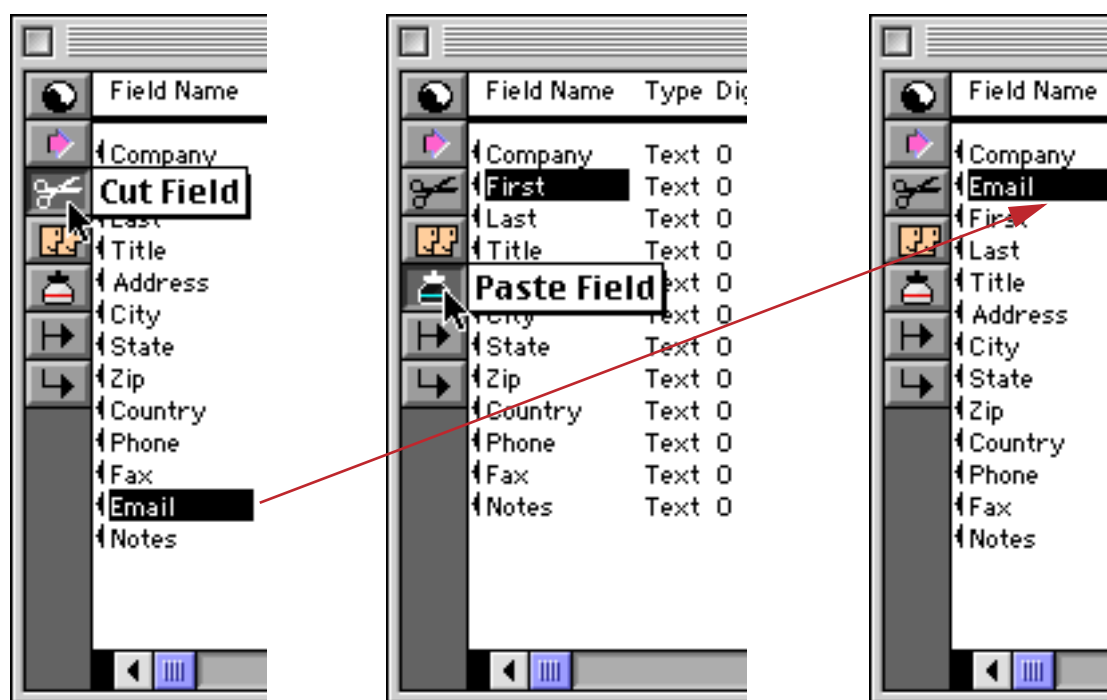
### Re-Arranging Fields

The design sheet can also be used to re-arrange the order of the fields. It's a two step process—first re-arrange the lines in the design sheet, then create a new generation.

To move a field up one line in the design sheet, press **Command-Up Arrow** (Mac) or **Control-Up Arrow** (PC). To move a field down one line in the design sheet, press **Command-Down Arrow** (Mac) or **Control-Down Arrow** (PC). Keep pressing until the field reaches the desired position.



Another way to move a field is with the **Cut Field** and **Paste Field** tools. This method may be faster if you need to move a field a long distance.



When the fields are in the correct order, tell Panorama to create a new generation.

## Rules for Field Names

Each field in a Panorama database is identified by a field name. Field names serve several purposes: they remind you what the field is for (i.e. the **Dates** field probably contains dates, the **Name** field probably contains names, etc.), they appear at the top of each column in the data sheet, and they are used to identify fields in formulas and procedures (for example **Amount=Qty\*Price**).

Panorama doesn't place any restrictions on the field names you choose. Field names may be as long as you want, and they may contain any character that can be typed from the keyboard. Field names may be split over two or more lines (see below). You can even have two or more fields with the same name (but we recommend that you avoid this, see the next paragraph).

However, if you are planning to use a field in a formula or procedure, you may want to avoid some of these unusual possibilities. If you have two or more fields with the same name, only the first field will be accessible to a formula. Field names containing blanks or punctuation (for instance **P/E Ratio**) are more difficult to use in a formula. To use such a field in a formula, you must surround the field name with « and » (for example **«P/E Ratio»**. See See "**Fields**" on page 520). (If you left out the «», Panorama would think you were trying to divide **P** by **E**, with **Ratio** left over.) You may want to avoid field names like **Date**, **Seconds**, **And**, **Or**, and **Sum**. These names can be confusing when used in a formula because Panorama has functions with the same names.

## Repeating Fields (Line Items)

Some databases contain several similar fields repeating within each record. For example, an invoice usually contains several quantities, product descriptions, product prices, etc. These fields are often called **Line Items** because they repeat for each line on the invoice. To learn how to set up this type of field see Chapter 5 of the **Panorama Handbook**.



# Chapter 6: Data Types



In Panorama, all data is not the same. Just as Eskimos distinguish between 16 types of snow, Panorama distinguishes between five types of data—**text**, **numeric**, **date**, **choices**, and **pictures**. To get the most out of a database, Panorama needs to know what type of data you intend to store in each field. This lets Panorama store the data efficiently and check for data entry errors. It also tells Panorama how to compare different values (numbers, text, and dates are all compared differently) which is important for sorting and selecting data. The data type also tells Panorama how to format some kinds of data (numbers and dates).

As mentioned above, Panorama databases can contain five different types of data. When you create the database, you specify what type of data will be stored in each field.

Data Type	Uses	Examples
<b>Text</b>	Names, Addresses, Descriptions, etc.	John, 234 Peach Avenue
<b>Numeric</b>	Prices, Quantities, etc.	4, 78.23, 4.9e-2
<b>Date</b>	Dates.	9/18/2002
<b>Choices</b>	Multiple Choice Options	Yes/No, Gold/Silver/Bronze
<b>Picture</b> (Obsolete)	Photographs, Drawings	n/a

The **text** data type is used for storing ordinary text—names, addresses, descriptions, etc. Panorama cannot perform mathematical calculations (add, subtract, etc.) on data that is stored as text.

The **numeric** data type is used for storing numbers—prices, quantities, etc. Use the numeric data type for any field you want to use in a calculation. The numeric data type has several variations that are discussed later in this chapter.

It's not always necessary to store numbers in numeric fields. For example, zip codes and phone numbers are usually stored in text fields, not numeric fields. This allows the use of nine digit zip codes (for example **92867-3482**) and foreign postal codes and phone numbers. In general, use a numeric field if you want to perform numeric calculations (addition, multiplication, etc.) and/or if you want to select or sort the information in numeric order (1, 2, 3, ... 10, 20, 30, ... etc.)

The **date** data type is self explanatory—it is used for storing dates (for instance March 1, 2009). Panorama understands the properties of dates—it knows that May 1st follows April 30th and that there are six days between May 28th and June 3rd. Panorama can handle dates from 100 A.D. to well past the year 20,000 A.D.

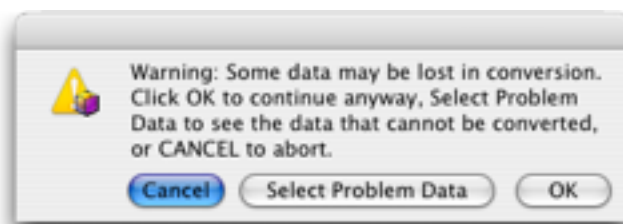
The **choices** data type is used for storing data that has only a few possible values—for instance **yes/no**, **gold/silver/bronze**, or **coach/first class**. The choices data type saves space by storing a special code instead of the entire text. When Panorama was introduced in 1988 these memory savings were important, but now this option is rarely used. The procedure for setting up a field using the choices data type is described in Chapter 6 of the **Panorama Handbook**.





## Data Type Conversion Problems

When you change the data type of an existing field it is possible that you may lose some of the data in that field. For example, if you convert a field from text to numeric, any letters or punctuation will be lost in the conversion. Panorama will warn you if this situation occurs.



The alert gives you three options:

**OK** — Pressing this button tells Panorama to complete the conversion. Any data that cannot be converted will be removed. For example, if you are converting a field from text to numeric any non-numeric data will be lost in the conversion.

**Cancel** — Pressing this button cancels the conversion

**Select Problem Data** Pressing this button tells Panorama to cancel the conversion and show you the data that is causing the problem. This option lets you look at the data that is causing the problem and decide what to do next. If necessary, you can make manual adjustments to the data, or you may decide that you don't want to change the data type after all.

For example, suppose you wanted to change the Zip code field in the database below to numeric.

	Address	City	State	Zip	Country
	12 Harmony Lane	Huntington Beach	CA	92648	
	783 Algonquin	Newport Beac	CA	93459	
	498 Noyes	Evanston	IL	60201	
	14189 8th	Newhall	CA	91321	
	398 N. Churchill	Barrie	ONT	V5A 7B2	CANADA
	118 N Wilder	Lubbock	TX	79410	

When you attempted to make the conversion, Panorama would display the conversion warning alert. Press **Select Problem Data** to see what is causing the problem.

	Address	City	State	Zip	Country
	398 N. Churchill	Barrie	ONT	V5A 7B2	CANADA
	898 Lark	Prince Rupert	BC	S3D 9H4	CANADA

Aha! The problem is the Canadian postal codes, which have letters instead of numbers. At this point you would probably want to rethink the idea of converting the Zip field to numeric.

If you did decide to go ahead with the conversion, Panorama would strip the letters from the Canadian postal codes.

	Address	City	State	Zip	Country
	398 N. Churchill	Barrie	ONT	572	CANADA
	898 Lark	Prince Rupert	BC	394	CANADA

When you are done looking at the problem data, choose the **Select All** command (Search Menu) to make all the data visible again.

## Numeric Data

Numeric data can be stored in either **fixed point** or **floating point** format. If you choose fixed point you have a choice of 0, 1, 2, 3, or 4 digits after the decimal point.

Number of Digits After Decimal Point	Example	Largest Value	Smallest Value	Typical Uses
0	93842	2,100,000,000	1	Quantities, Part Numbers
1	73.1	210,000,000	0.1	Rarely Used
2	253.22	21,000,000	0.01	Money (Dollars, Pounds, etc.)
3	0.447	2,100,000	0.001	Rarely Used
4	929.1123	210,000	0.0001	Rarely Used
Float	1.46e-12	$1.7 \cdot 10^{308}$	$2.3 \cdot 10^{-308}$	Scientific Data

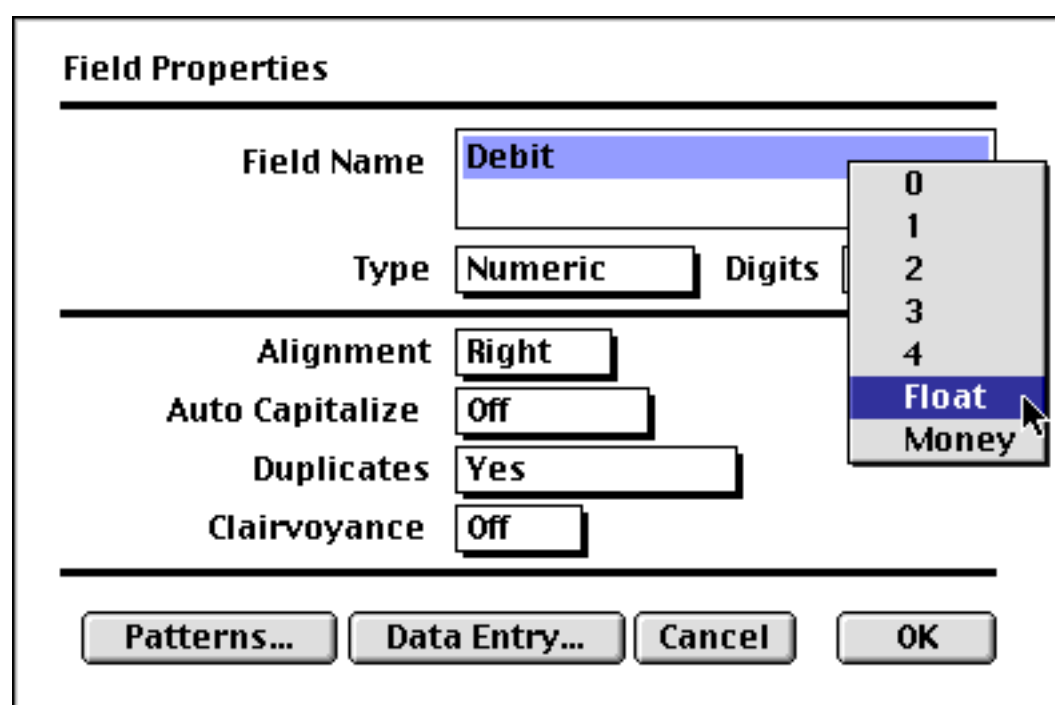
You may wonder why there are so many choices for storing numeric data. After all, a number is a number—right? Not quite. By choosing different numeric storage formats you are making a trade-off between space, speed, accuracy, and range.

Storing numbers using floating point gives you the most accuracy and numeric range. Floating point allows you to store extremely large or small values with up to 16 digits of accuracy. If you are in doubt, go ahead and pick floating point format.

Fixed point storage is more limited. The accuracy is only about 9 digits. The largest number that can be stored is about 2 billion ( $2 \cdot 10^9$ ) while the smallest fixed point number is 0.0001 ( $10^{-4}$ ). Trying to store larger or smaller values using fixed point storage will result in errors.

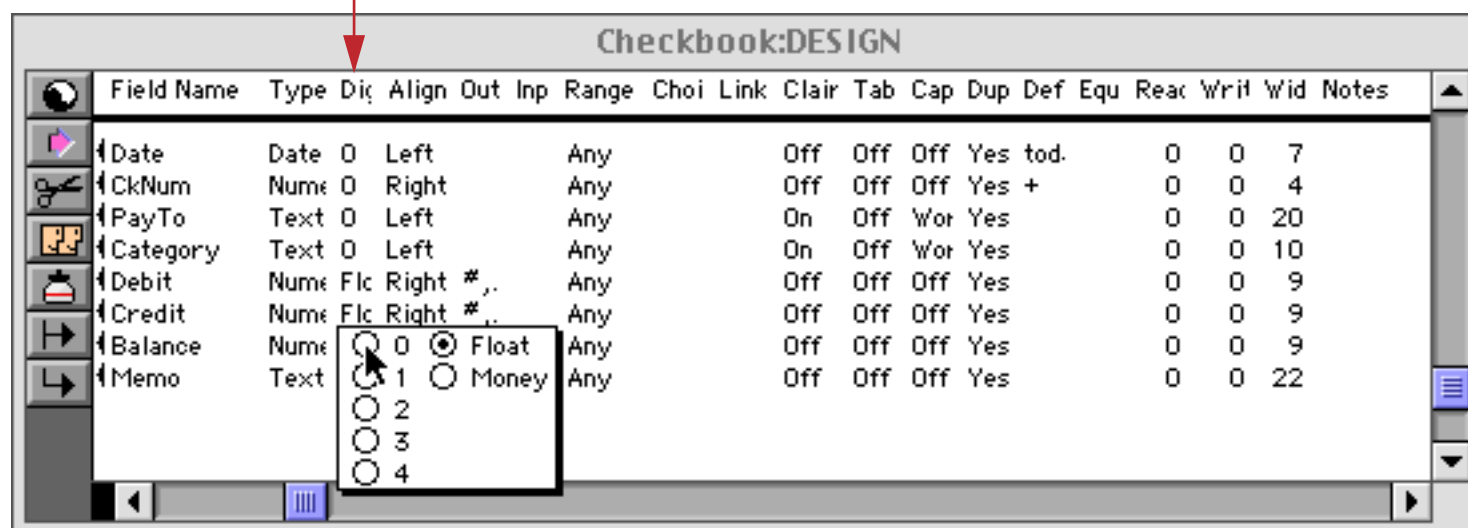
On the other hand the space required for fixed point storage is up to 8 times smaller than floating point for the same number, and Panorama can perform fixed point arithmetic much faster than floating point. You should use fixed point numeric storage whenever possible. Check the table above to see if the numbers you will be using fit in one of the fixed point numeric ranges.

You can set the number of digits via a pop-up menu in the **Field Properties** dialog box.



You can also set the number of digits using the **Digits** column in the design sheet.

*double click on any cell in this column to change the number of digits*



## Money

Usually the best way to store monetary values is using either 2-digit fixed point or Panorama's special Money format. The money format is the same as 2-digit fixed point but automatically enters the decimal point for you during data entry. This table below shows how Panorama interprets data you enter into a money field.

When you enter...	it becomes
87204	872.04
3267	32.67
14	0.14
2	0.02
42.	42.00
15.4	15.40
156.78	156.78

Both the 2-digit and money formats allow you to store monetary values up to 21 million dollars, pounds, francs, etc. (If your business deals with values greater than 21 million you should use floating point numeric storage.)

## Numeric Output Patterns

**Output patterns** allow you to control the way a number is displayed. In the design sheet, output patterns can be used to control how numbers are displayed in the data sheet. Output patterns can also be used in a formula.

Below are some ways the same number may be displayed using different output patterns. Remember, the way a number is displayed does not change its internal value. All the numbers listed below have the value 2654.

2654  
 2,654  
 \$2,654.00  
 002654  
 2.654e+3  
 26-54  
 Two thousand six hundred fifty four

Numeric output patterns consist of a string of characters containing one or more # symbols. The # symbol tells Panorama how and where to print the number.

The overall output pattern for a field can be set using the **Pattern** button in the **Field Properties** dialog box or the **Output Pattern** column in the design sheet.

When you press the **Pattern** button a new dialog appears. You can either type the pattern into the box at the top or you can select the display options you want and let Panorama create the output pattern for you.

If you are using the design sheet to set up the pattern, simply type the pattern into the **Output Pattern** column.

*type the pattern into the Output Pattern column*

Field Name	Type	Diç	Align	Out	Inp	Range	Choi	Link	Clair	Tab	Cap	Dup	Def	Equ	Reac	Writ	Wid	Notes
Date	Date	0	Left			Any			Off	Off	Off	Yes	tod.		0	0	7	
Date Clear	Date	0	Left			Any			Off	Off	Off	Yes	tod.		0	0	7	
CkNum	Num	0	Right			Any			Off	Off	Off	Yes	+		0	0	4	
PayTo	Text	0	Left			Any			On	Off	Wor	Yes			0	0	20	
Category	Text	0	Left			Any			On	Off	Wor	Yes			0	0	10	
Debit	Num	2	Right	#,###		Any			Off	Off	Off	Yes			0	0	9	
Credit	Num	2	Right	#,.		Any			Off	Off	Off	Yes			0	0	9	
Balance	Num	2	Right	#,.		Any			Off	Off	Off	Yes			0	0	9	
Memo	Text	0	Left			Any			Off	Off	Off	Yes			0	0	22	

### Fixed Decimal Point Patterns

The table below shows how the output pattern can be used to display numbers with a specified number of digits after the decimal point. These output patterns force a fixed point display even if you are displaying floating point numbers. They also allow you to override the natural display of fixed point numbers. For example, a money field can be set up to display only dollars, while still keeping track of cents for calculation purposes.

Number	Pattern	Display
1234.56		1234.56
1234.56	#	1235
1234.56	#. #	1234.6
1234.56	#. ##	1234.56
1234.56	#. ####	1234.5600

Notice that if the number of # symbols after the decimal point is less than the number of digits in the number, Panorama will round the number rather than truncating it.

### Numbers with Commas, Punctuation, and Measurement Units

If a comma is added to the pattern, the number will be printed with a comma every third digit. Other characters can also be added to the beginning or end and will be displayed unchanged. For example, you can add a currency symbol or measurement unit to the pattern, as shown below:

Number	Pattern	Display
1234.56	#, .##	1,234.56
1234.56	\$#, .##	\$1,234.56
1234.56	#, .## kg	1,234.56 kg

### Scientific Notation

If an **E** or **e** is added at the end of the output pattern, the number will be displayed using scientific notation. Any number may be displayed in scientific notation, including fixed point numbers.

Number	Pattern	Display
1234.56	#e	1e+3
1234.56	#. #E	1.2e+3
1234.56	#. ##e	1.23e+3
1234.56	#. ###E	1.235e+3
1234.56	#. ####E	1.2346e+3
1234.56	#. #####E	1.23456e+3
1234.56	#. #####E	1.234560e+3
1234.56	#. #E kg	1.2e+3 kg



### Special Patterns for Negative Numbers

Negative numbers are usually displayed with a minus sign in the front of the number. This can be changed to a trailing minus sign or to enclosing parentheses.

Number	Pattern	Display
-1234.56	#.##	-1234.56
1234.56	#.##	1234.56
-1234.56	#.##-	1234.56-
1234.56	#.##-	1234.56
-1234.56	(#.##)	(1234.56)
1234.56	(#.##)	1234.56

### Leading Zeros

You can use an output pattern to force Panorama to display leading zeros. To do this, put several # symbols in a row without a decimal point.

Number	Pattern	Display
123	#####	00123
1234	#####	01234
12345	#####	12345

Tip: If you are storing US Zip codes in a numeric field, use ##### as the output pattern. This pattern makes sure that all 5 zip code digits are displayed, even if the first digit is zero.

If your database also contains Canadian postal codes, the zip codes must be stored in a text field. In that case no output pattern is necessary.

### Plural Suffixes

If a pattern contains measurement units you may want to properly pluralize the units depending on the value being displayed. Use the ~ symbol to do this.

Number	Pattern	Display
1	# mile~	1 mile
5	# mile~	5 miles

### Displaying Numbers as Words

If you wish, numbers can be displayed as words instead of digits. To do this, use the **§** symbol instead of the # symbol. Only one **§** symbol should be used. To make the **§** symbol on a Macintosh, press **Option-6**. On the PC, press **Alt-0167**. Only the integer part of the number will be displayed—any fractional part will be ignored.

If you are displaying money, you'll probably want to display the fractional part (cents) as well as the integer part. You can do this with the ¢ (cents) symbol. On the Macintosh, press **Option-4** to create the ¢ symbol. On the PC, press **Alt-0162**. Use one ¢ for each digit you want display (usually 2).

Number	Pattern	Display
312	§	Three hundred twelve
42.29	§ dollar~ and ¢¢/100	Forty two dollars and 29/100

## Dates

Panorama has a special data type for storing dates. When you use the date type to store your dates, Panorama can sort your dates in the correct order, check your dates for validity as they are entered, and calculate the number of days between two dates. Dates are quite compact; almost any date in the 20th or 21st century will take only two bytes of storage.

### Entering Dates

Panorama is very flexible about how you type dates. We call this feature “smart dates.” You can enter dates numerically (for instance **04/09/02** or **4/9/2**) or you can spell out the date (for instance **April 9th, 1997** or **Apr 9 97**). You can use any character as a separator between numeric dates, for example **4-9-01** or even **4.9.01**.

To enter today’s date, simply type **today**. You can also enter **yesterday** or **tomorrow**. Panorama will automatically convert these entries to the correct month, day and year.

If the date is in the current week, you can simply type in the name of the day, for example **saturday** or **tue**. To specify a day in the previous or upcoming weeks add the words last or next, for example **next tuesday** or **last saturday**.

When a date is edited, Panorama normally displays the date in the format **mm/dd/yy**. However, if you have set up an output pattern that Panorama understands for data entry, it will use that pattern instead. Patterns that can be used for data entry include **Month dd, yyyy**, **Mon dd yy**, and **mm/dd/yyyy**.

If you are using an international system and you enter the date as numbers you must use the format **dd/mm/yy**. Panorama does not understand the format **7-Aug-1998**. (However you may use any delimiter character you want, for example **7/8/98** or **7-8-98** or even **7.8.98**.)

### Default Year and Century

When you enter a date, you can leave the year off and let Panorama figure it out for you. Panorama will automatically round the date to the nearest year. For instance, if today’s date is **3/9/02** and you enter the date **4/1** then Panorama will assume you mean **4/1/02**. But if you enter **12/1** (or **December 1**) Panorama will assume you mean **12/1/01**, not **12/1/02**, because **12/1/01** is closer to **3/9/02** than **12/1/02** is.

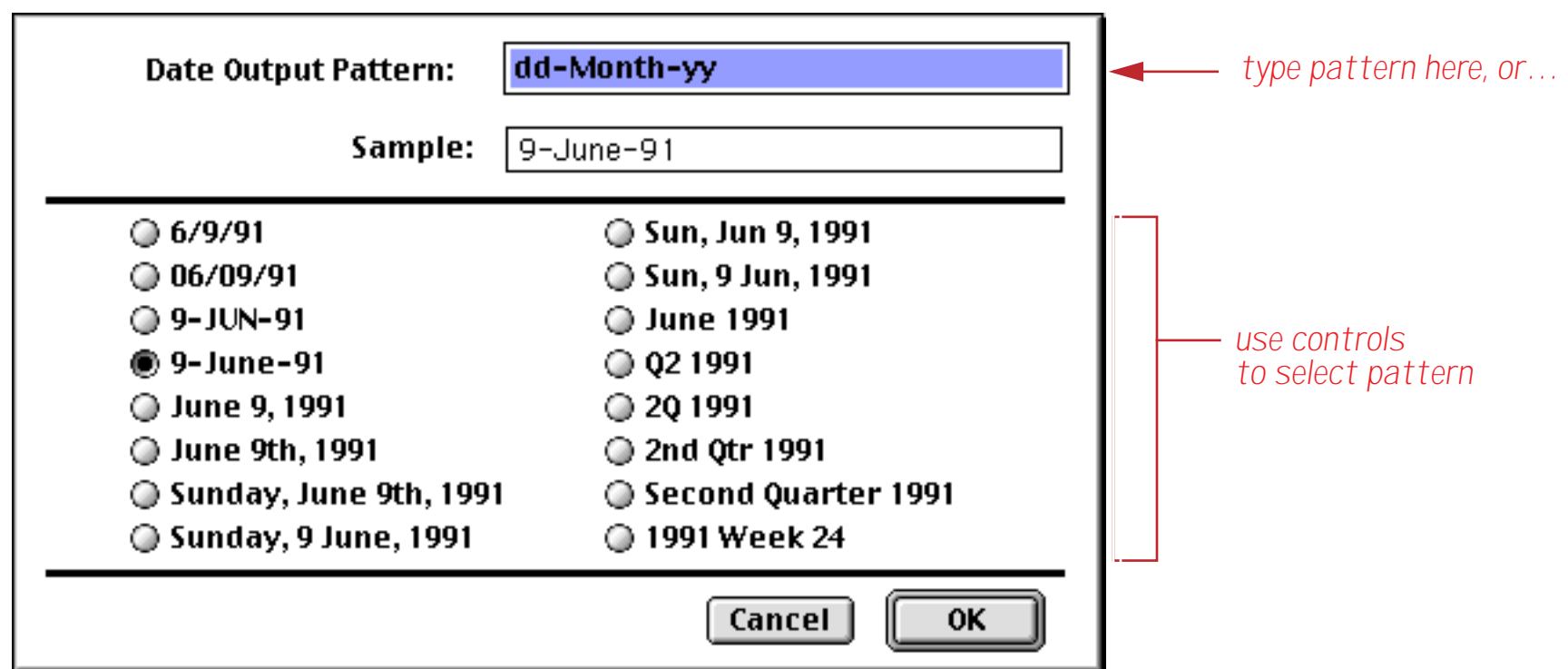
Panorama also rounds dates to the current century. If the current year is **2002** (or even **1992**) and you enter the date **7/2/23** Panorama will assume you mean **7/2/2023**. If you want to enter a date more than 50 years from the current date you must enter the full date, for example **7/2/1923**

### Date Output Patterns

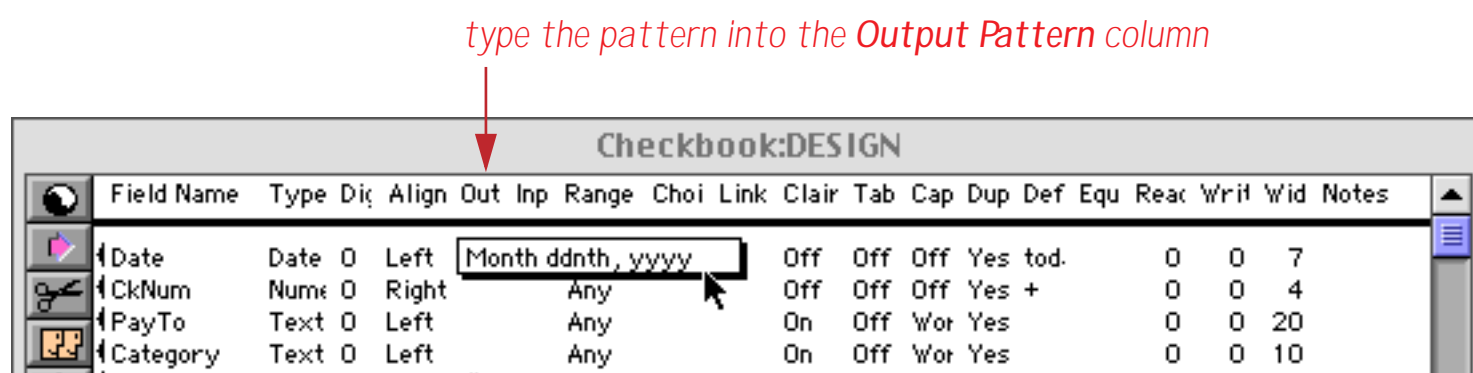
**Output patterns** allow you to control the format Panorama uses to display dates. A date output pattern consists of a number of individual components (month, day, year, etc.) that are strung together. For example, the pattern **mm/dd/yy** contains three components and will display in the format **3/11/04**.

The overall output pattern for a field can be set using the **Pattern** button in the **Field Properties** dialog box or using the **Output Pattern** column in the design sheet.

When you press the **Pattern** button a new dialog appears. You can either type the pattern into the box at the top or you can select the pattern you want using the radio buttons.



If you are using the design sheet to set up the pattern, simply type the pattern into the **Output Pattern** column. See “[The Design Sheet](#)” on page 105 for more information on opening and using the design sheet.



### Date Pattern Components

There are 15 different basic components that can be used as part of a date pattern. A date pattern is built up by combining these components together with punctuation to build a complete pattern. (See “[Common Date Output Patterns](#)” on page 123 for examples of complete, ready-to-use date patterns.)

Component	Example	Description
yy	02	Year (within century)
yyyy	2002	Year (including century)
qq	2	Quarter (numeric)
qtr	2nd	Quarter (abbreviated)
quarter	second	Quarter (spelled out)
mm	9	Month (numeric)
MM	09	Month (with leading zero)
mon	sep	Month (abbreviated)
month	september	Month (spelled out)

Component	Example	Description
ww	46	Week (within year)
dd	5	Day (numeric)
DD	05	Day (with leading zero)
day	tue	Day Of Week (abbreviated)
dayofweek	tuesday	Day Of Week (spelled out)
dow	3	Day Of Week (0[sun]-6[sat])

Some of these components (`qtr`, `quarter`, `mon`, `month`, `day`, and `dayofweek`) can be either upper or lower case. The table below shows how a component can be displayed in all lower case, initial caps, or all upper case.

Pattern	Display
month	september
Month	September
MONTH	SEPTEMBER
dayofweek	friday
DayOfWeek	Friday
DAYOFWEEK	FRIDAY

### Common Date Output Patterns

A date output pattern is assembled from the basic components listed in the previous section along with any punctuation or text that is needed between the components. The table below lists several common date patterns.

Date	Pattern	Display
3/9/2002	mm/dd/yy	3/9/02
3/9/2002	MM/DD/YY	03/09/02
3/9/2002	mm-dd-yyyy	3-9-2002
3/9/2002	dd-MON-yy	9-MAR-02
3/9/2002	dd-Month-yy	9-March-02
3/9/2002	Month dd, yyyy	March 9, 2002
3/9/2002	Month ddnth, yyyy	March 9th, 2002
3/9/2002	DayOfWeek, Month ddnth, yyyy	Saturday, March 9th, 2002
3/9/2002	qqqyy	1q02
3/9/2002	Week ww of yyyy	Week 11 of 2002
5/23/2002	Quarter "Quarter" yyyy	Second Quarter 2002
7/11/2004	Qtr "Qtr" yyyy	3rd Qtr 2004

Date	Pattern	Display
3/9/2002	"Day" dd, "Month" mm	Day 9, Month 3
3/1/2002	ddnth "day of" Month, yyyy	1st day of March, 2002
3/2/2002	ddnth "day of" Month, yyyy	2nd day of March, 2002
3/9/2002	ddnth "day of" Month, yyyy	9th day of March, 2002
3/1/1867	mmnth "month of" yyyy	3rd month of 1867
3/9/1978	wwnth week of yyyy	3rd week of 1978

If you need to include the words `qtr`, `quarter`, `mon`, `month`, or `day` in your date, you must quote them so that they are not treated as components, as shown in several of the examples in the table above. The quote key is just to the left of the **Return** key. Be sure to use regular quotes, not smart quotes (" not " ").

As shown in several of the examples, you can add the suffix `nth` to the `mm`, `ww`, or `dd` components, Panorama automatically adds the correct suffix depending on the number displayed.



# Chapter 7: Data Entry & Editing



Before you can organize, analyze, or report anything, you have to get your data into the computer. Usually this means using the keyboard to type it in. Panorama has been designed to help make this task fast and accurate.

Of course, if the information is already stored on a computer, you may be able to access it without re-typing. Panorama can exchange data with most Macintosh software packages, and with many PC and mainframe packages as well. See [“Importing a Text File”](#) on page 54.

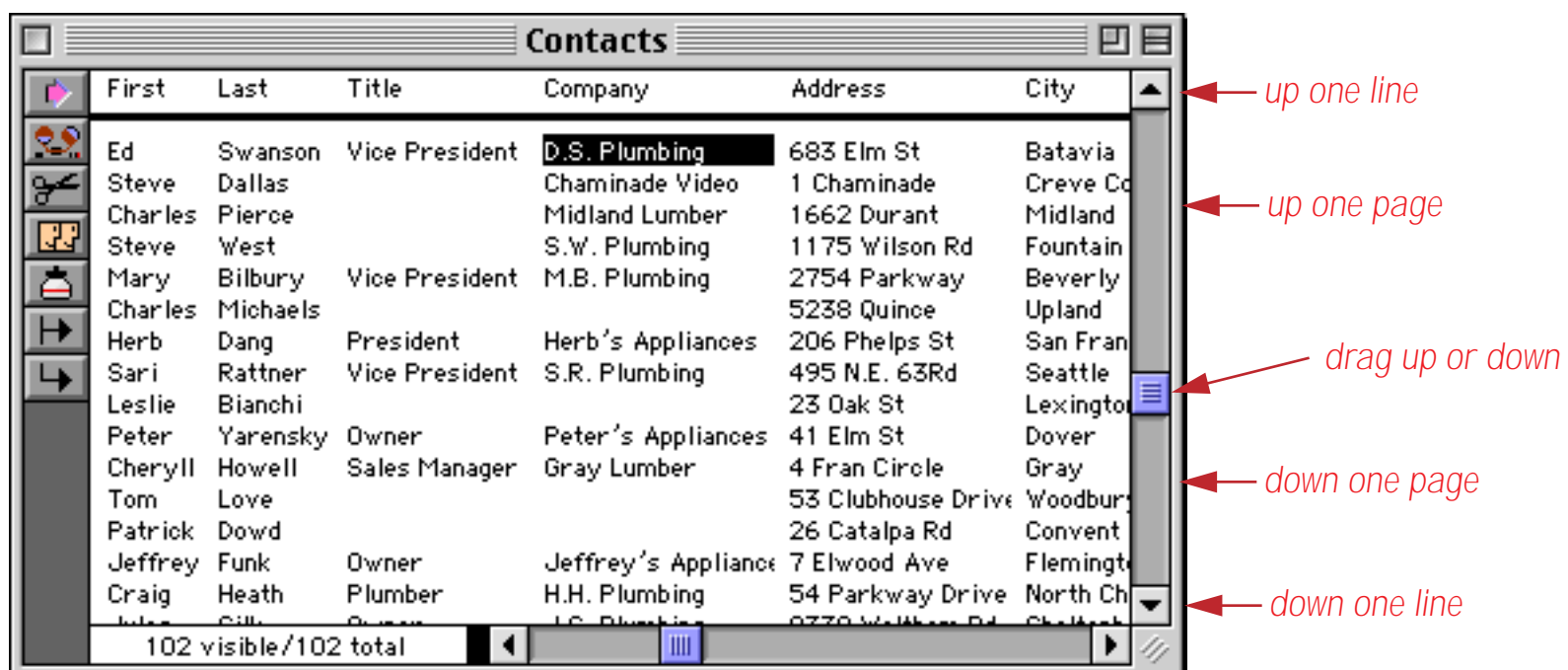
## Editing Records

A new database starts out with just one lonely record. Over its lifetime hundreds or even thousands of records will be added to a typical database. In addition, many records will become obsolete and be deleted. Both inserting and deleting records are easy tasks with Panorama.

## Moving From Record to Record

Unless your database is very small, only a small part of it will be visible at a time. You can shift the data within the window to work with different parts of the database. To shift the data you either scroll (data sheet) or flip from page to page (form).

Use the vertical scroll bar to scroll a data sheet to any record in the database. Click on the scroll bar arrows to move up or down one record at a time. Click in the scroll bar's gray area to move up or down one window at a time. Drag the scroll bar thumb to move directly to any position in the database.



When you are using a form window, the vertical scroll bar works differently. Instead of scrolling to another record, the scroll bar shifts the position of the form within the window. This allows you to get at every corner of a large form even if you are using a small window. To move from record to record, use the **First Record**, **Previous Record**, **Next Record**, and **Last Record** tools in the tool palette. The **First Record** tool brings you to the very first visible record in the database—the top line of the data sheet. The **Last Record** tool takes you to the end of the database—the bottom line of the data sheet. The **Previous Record** and **Next Record** tools move one record at a time. You can also move up or down one record at a time by pressing the **Up Arrow** or **Down Arrow** keys.

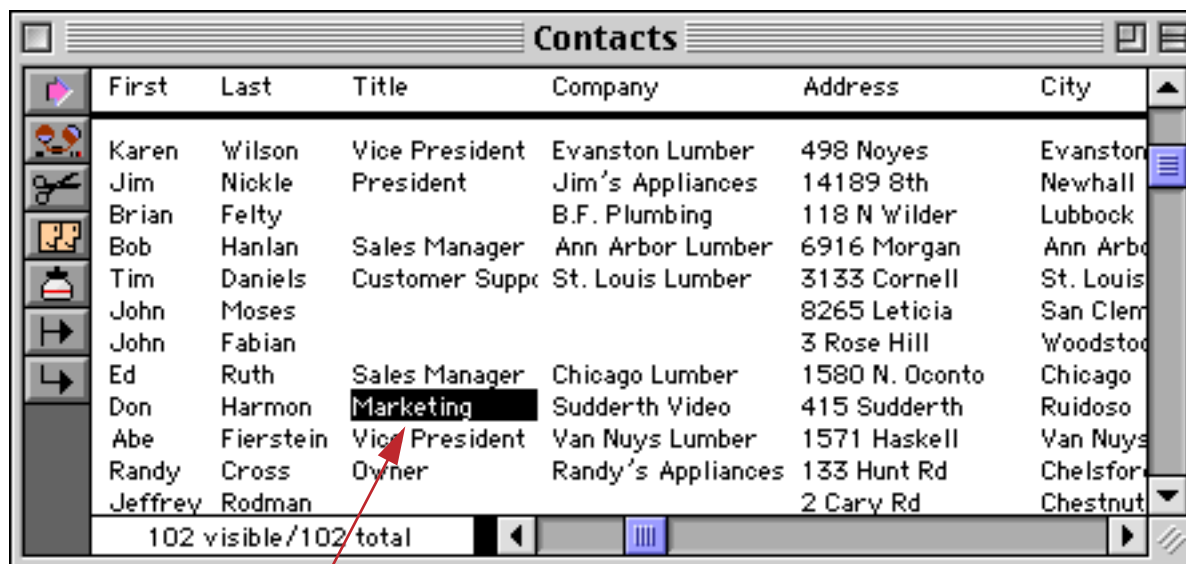


If the database has more than two open windows (for example a form and a data sheet), the position of both windows will always remain synchronized. In other words, if you scroll the data sheet the form will follow and vice versa.

In addition to manually moving from record to record, you can let Panorama search for the information you want to look at or modify. See Chapter 9 for more information about searching for and selecting data.

## Moving from Field to Field

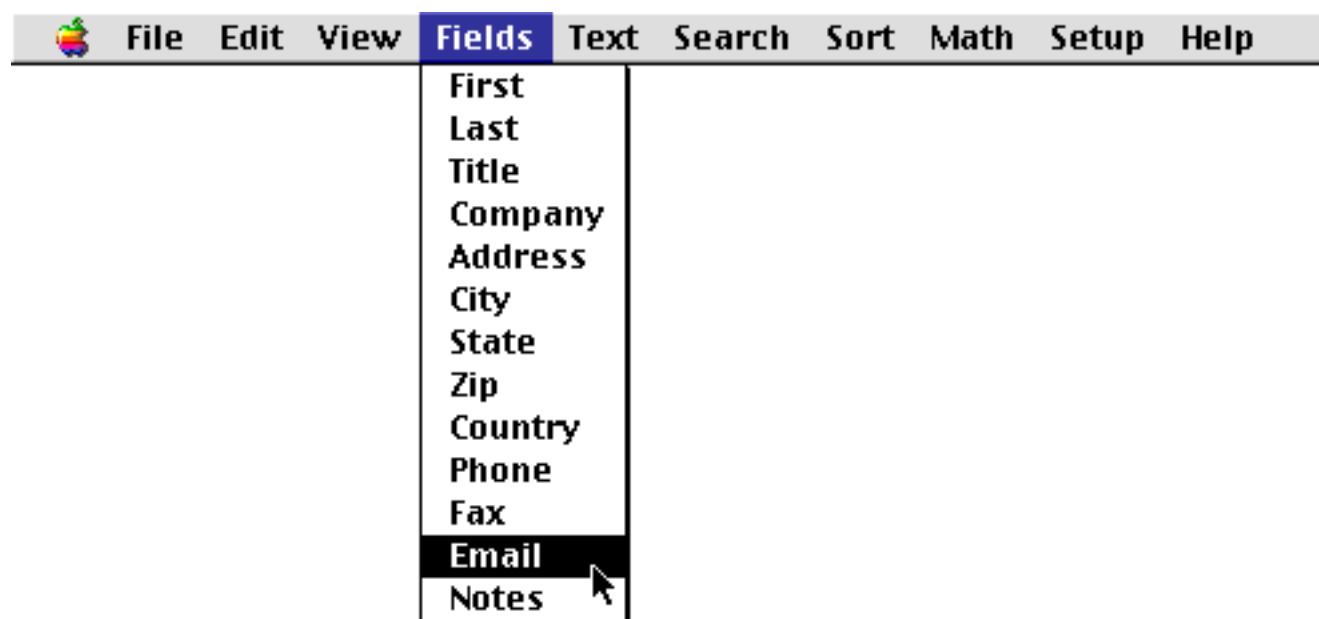
Within a data sheet, you can move to another field by clicking anywhere the field's column (if it is visible) or by clicking on the horizontal scroll bar. (“[Splitting a Window](#)” on page 80 for information on how to split a window into two separately scrollable “panes.”)



*click anywhere in a column to select a field*

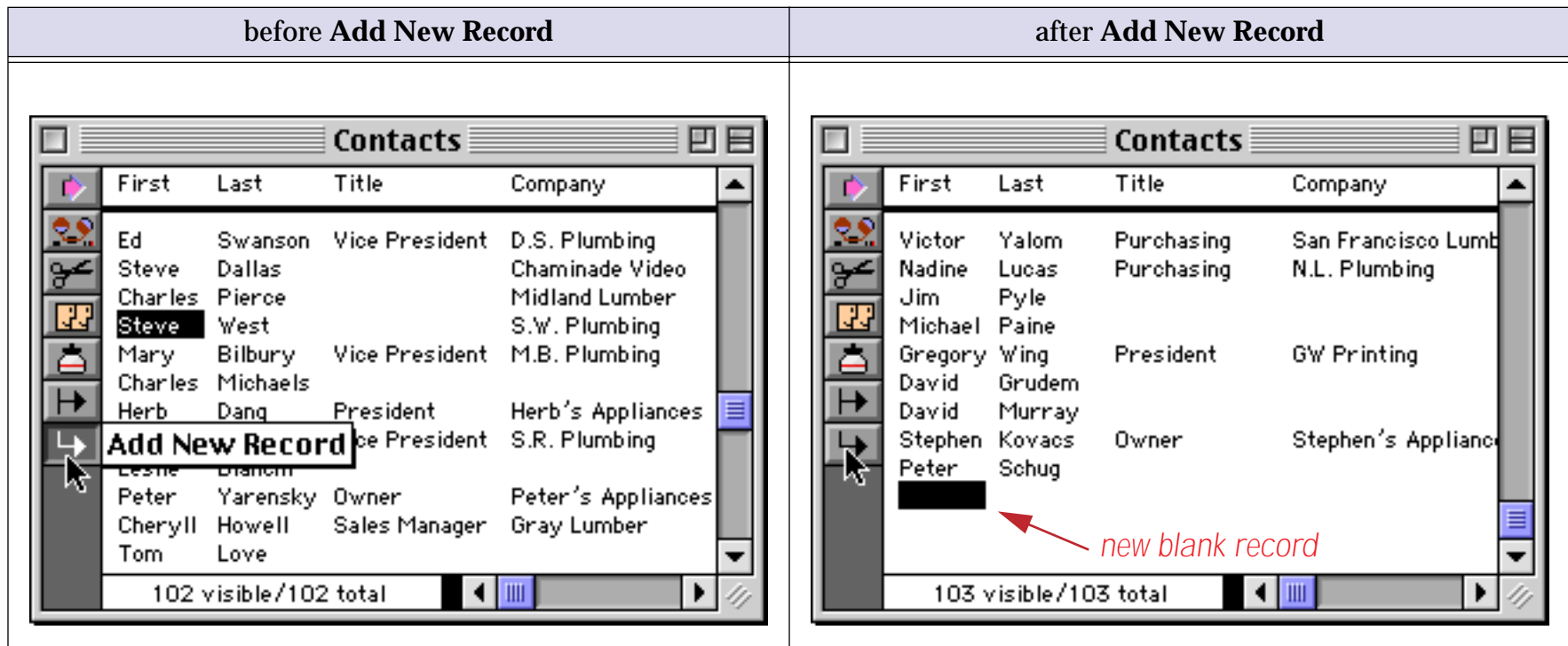
*or use the scroll bar*

You can also use the **Fields** menu to move to a specific field. This menu contains a list of all of the fields in the current database. The fields are listed in the order that they appear in the data sheet. To move to a field, simply select it from the menu. The **Fields** menu makes it easier to select a specific field when the database has dozens (or even hundreds) of fields.



### Adding a New Record

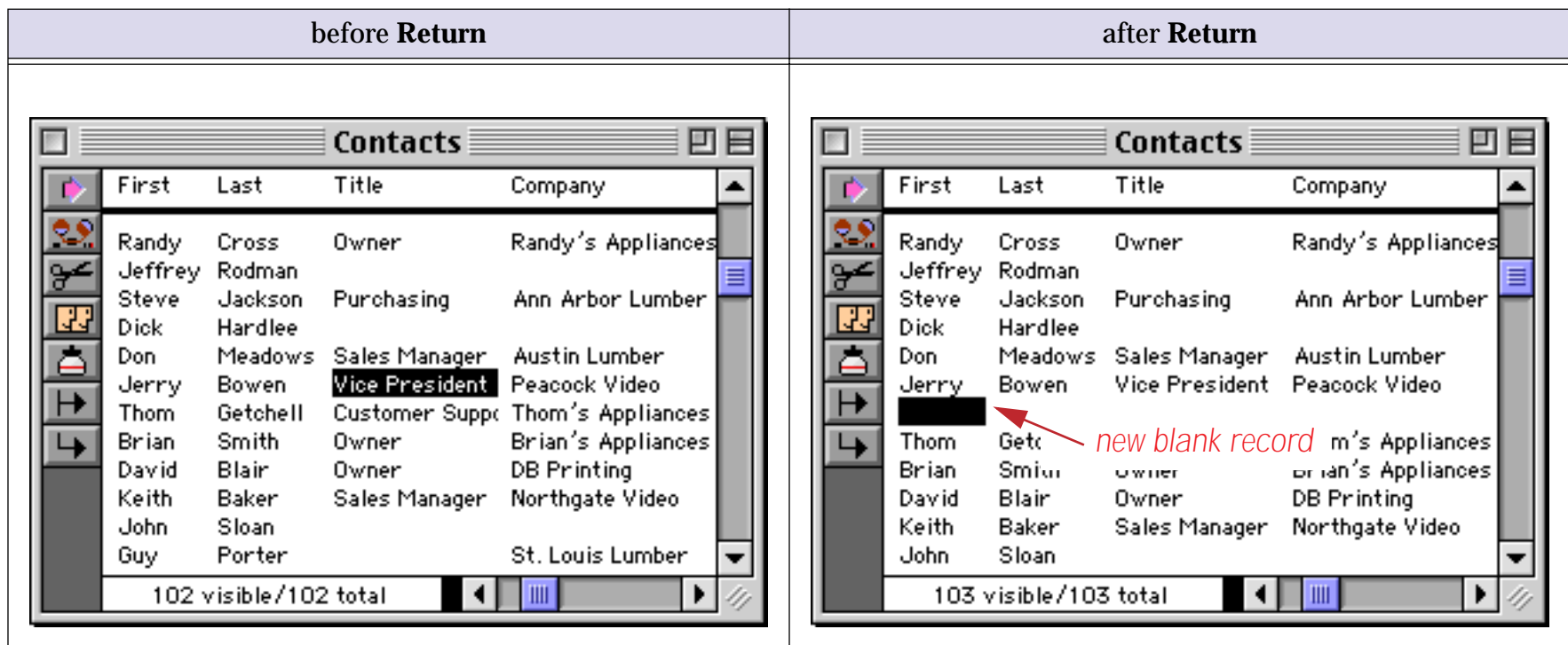
To add a new record to the end (bottom) of the database, either click on the **Add New Record** tool or choose **Add New Record** from the Edit Menu.



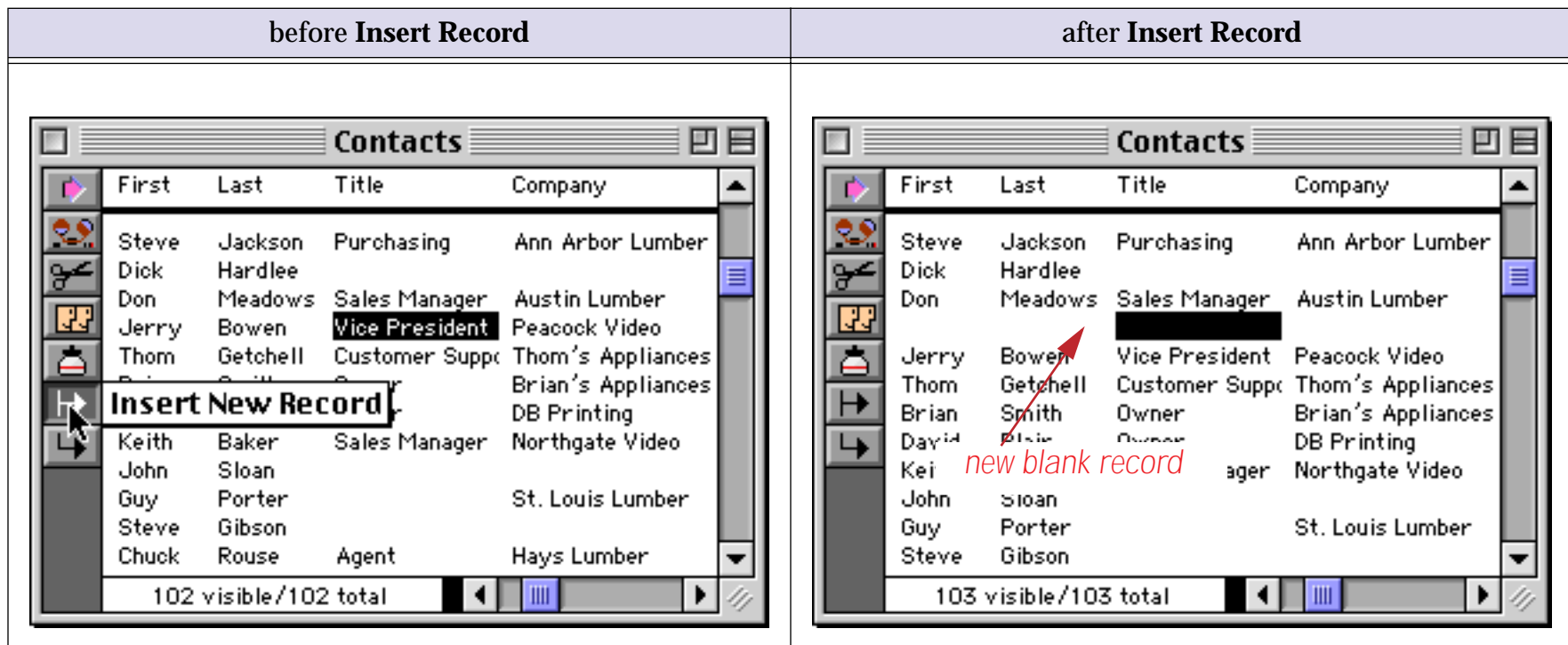
You can also add a new record by tabbing from the end of the bottom line of the data sheet.

### Inserting a New Record

When you are working in the data sheet, you can insert a new record either above or below the current record. (In a form you can only add new records at the end of the database.) To insert a new record after the current record press the **Return** key.



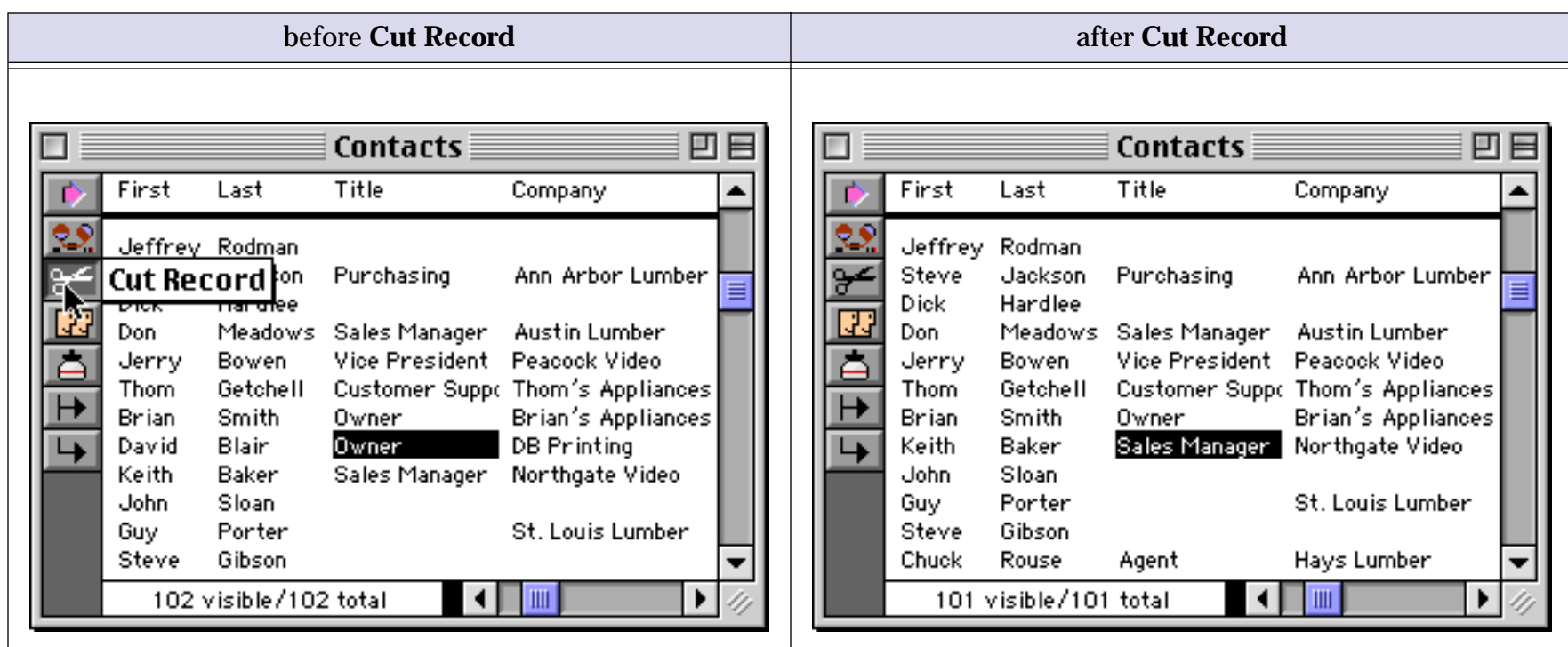
To insert a new record before the current record, click on the **Insert Record** tool.



Usually new records are completely blank, ready for your input (as shown in the examples above). You can, however, ask Panorama to automatically fill in one or more cells whenever a new record is created. A field can default to a fixed value (like **yes** or **no**, or **taxable**, or today's date), an automatically incrementing number (1, 2, 3, ...), or a copy of the data in a previous record. See "[Default Values](#)" on page 144 for more information.

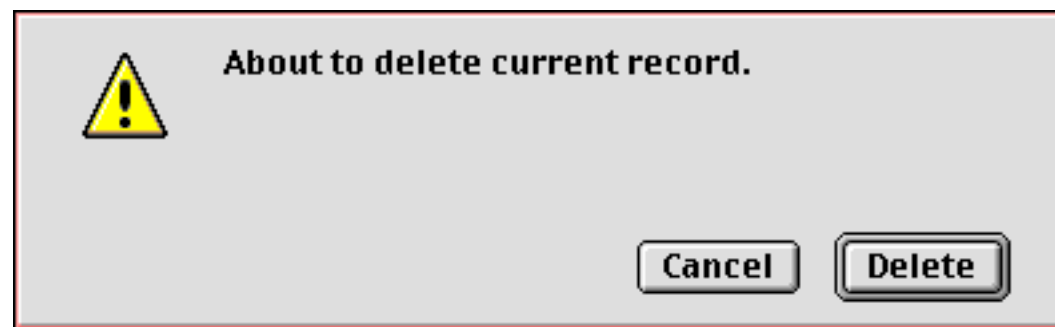
### Deleting a Record

To delete an entire record, click on the **Cut Record** tool.





If you are using the data sheet, you can also delete a record by pressing the **Delete** or **Backspace** key (upper right hand corner of the keyboard, above **Return**). Panorama will display an alert asking you to confirm that you really want to delete the record. (Mac only tip: If you want to skip the alert, hold down the **Option** key while you delete the record.)



If you delete a line accidentally, you can use **Undo** to restore the line. You can also get the line back with the **Paste Record** tool.

Tip: A Panorama database must have at least one visible record—it cannot have zero records. If your database has only one record, Panorama will not allow you to delete it.

### Deleting Multiple Records

Sometimes you may need to systematically delete large numbers of records. For example, you might need to delete all invoices previous to 1987 or all students with below passing grades. Instead of deleting these records one-by-one you can let Panorama do most of the work for you. First use the **Find/Select** command to select the records you want to keep. Then use the **Remove Unselected** command to delete everything else. Remember, however, that you cannot delete every record—you must leave at least one record in the database at all times. See Chapter 9 for more information on selecting and deleting multiple records..

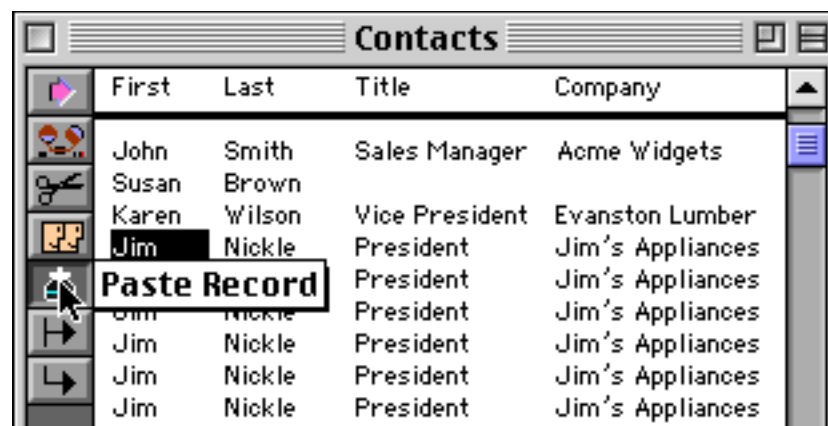
### Delete All

To delete all the data in the database, use the **Delete All** command in the Edit Menu. This command deletes all the data, leaving just one blank record. Before it performs this dastardly deed, Panorama asks you to confirm that you really know what you are doing. Keep in mind that there is no **Undo** after **Delete All**. (However, if you saved a copy of your database, you can **Revert to Saved**.)

You can use the **Delete All** command to set up a clone of an existing database. First open the original database, then use **Save As** to save it under a new name. Finally use **Delete All** to empty the new database. The new “cloned” database will contain all of the forms, crosstabs and procedures of the original database, but no data.

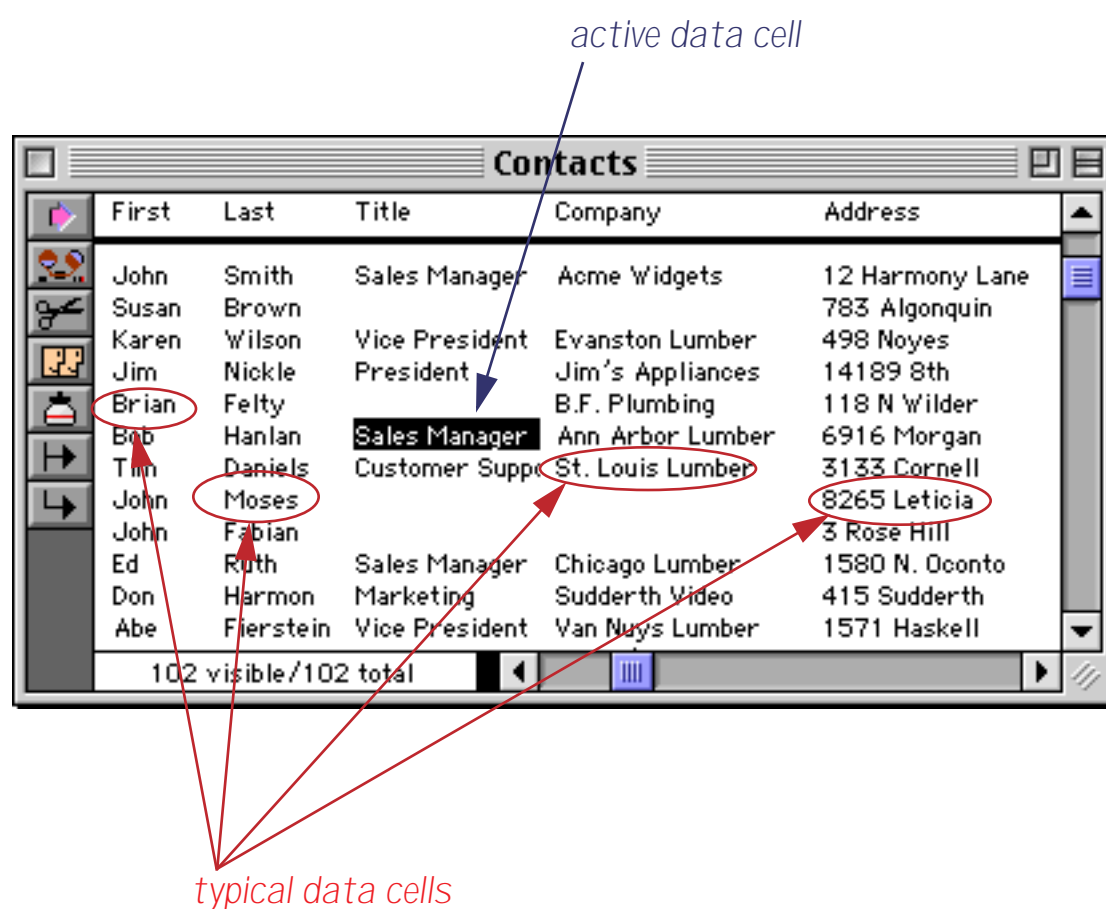
## Duplicating a Record

In the data sheet you can easily make one or more copies of a record. Use the **Copy Record** tool to copy the current line into the clipboard, then use the **Paste Record** tool to paste the line back into the database. The **Paste Record** tool inserts a new line just above the current line and then pastes the contents of the clipboard into the new line. You can paste the record back into the database as many times as you like. In this illustration the **Paste Record** tool has already made six copies of the line, and is about to create a seventh.



## Editing Data Within a Cell

Data cells are the smallest unit of information handled by Panorama. Each data cell contains a single piece of information—a person's name, a phone number, an account balance.

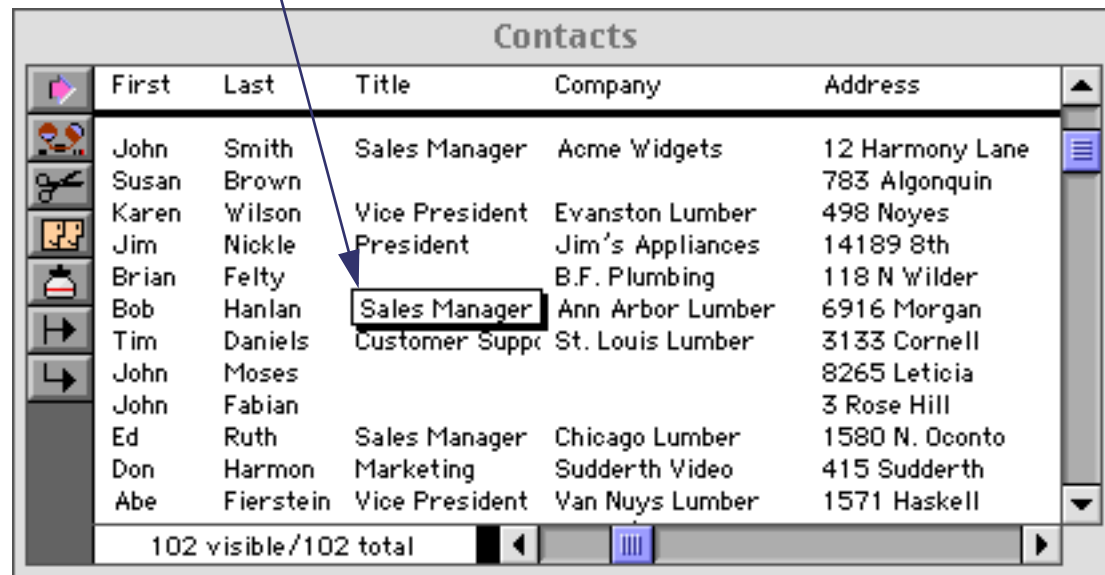


The currently selected cell is called the active cell. Only one cell can be active at a time. You can activate a cell by clicking on it, or by scrolling to it with the scroll bars. You can also move the active cell with the arrow keys.

## The Input Box

Every data cell has a pop-up **Input Box** that is used to edit the text within the cell. The Input Box acts like a temporary window that pops up on top of the data cell for data entry and editing.

*double click cell to open **Input Box** for editing contents of cell*



	First	Last	Title	Company	Address
	John	Smith	Sales Manager	Acme Widgets	12 Harmony Lane
	Susan	Brown			783 Algonquin
	Karen	Wilson	Vice President	Evanston Lumber	498 Noyes
	Jim	Nickle	President	Jim's Appliances	14189 8th
	Brian	Felty		B.F. Plumbing	118 N Wilder
	Bob	Hanlan	Sales Manager	Ann Arbor Lumber	6916 Morgan
	Tim	Daniels	Customer Suppt	St. Louis Lumber	3133 Cornell
	John	Moses			8265 Leticia
	John	Fabian			3 Rose Hill
	Ed	Ruth	Sales Manager	Chicago Lumber	1580 N. Oconto
	Don	Harmon	Marketing	Sudderth Video	415 Sudderth
	Abe	Fierstein	Vice President	Van Nuys Lumber	1571 Haskell

You can open the Input Box by double clicking on the cell, or by making the cell active and starting to type. Once the Input Box is open, you can edit the text within the cell using the usual mouse editing (word processing) techniques. Specifically, you can click the mouse to select an insertion point, drag the mouse to select a range of characters, cut, copy or paste selected text using the clipboard, or use the keyboard to type characters at the insertion point. (If you are not familiar with these techniques you should review the operating system documentation that came with your computer.)

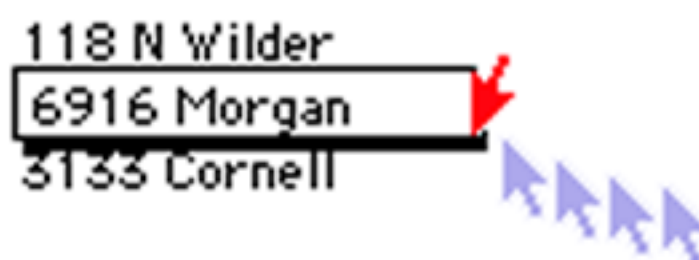
When you have finished editing the text within the cell, press the **Enter** key. This closes the Input Box and updates the data cell with your changes. The Input Box closes automatically (and updates the data) if you click on any other data cell or window. Tip: Clicking anywhere outside of the Input Box is the same as pressing the **Enter** key.

If the Input Box is only one line high pressing the **Return** key will close the Input Box and update the data cell. If the Input Box is more than one line high the **Return** key adds a new line to the data cell (see the next section).

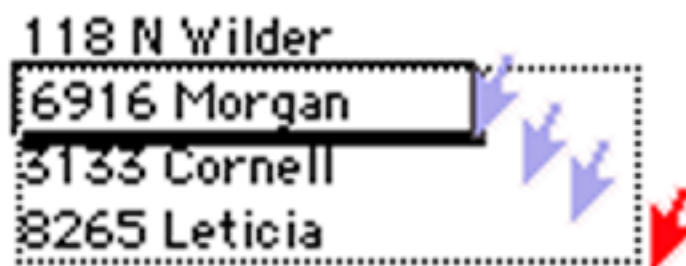
If you would like to close the Input Box without updating the data cell, press **Command-Period** (Mac) or **Control-Period** (Windows). Pressing the **Esc** key also closes the Input Box without updating the data cell. Or you can use the **Undo** command to restore the original text, then press the **Enter** key.

## Expanding the Input Box

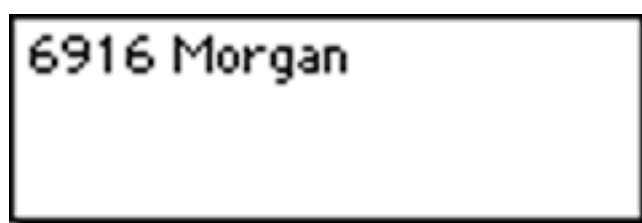
One of the most powerful features of the Input Box is that it can be expanded to accommodate large amounts of data. To expand the Input Box, move the mouse to the lower right corner of the box. When you reach the corner, the arrow will flip over. In the illustration below, you see the normal arrow cursor as you approach the corner. The arrow flips over when it reaches the corner of the Input Box. (The arrow doesn't actually change color as shown here, the color is simply to make the illustration more clear.)



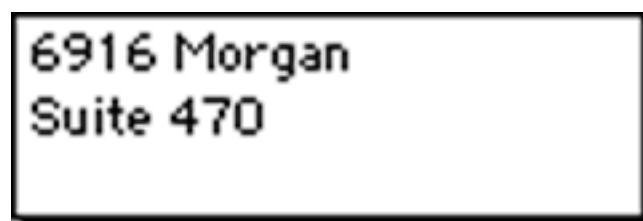
Once you see the upside down arrow, press the mouse and drag the corner of the box to its new location.



When you release the mouse Panorama will change the size of the box.

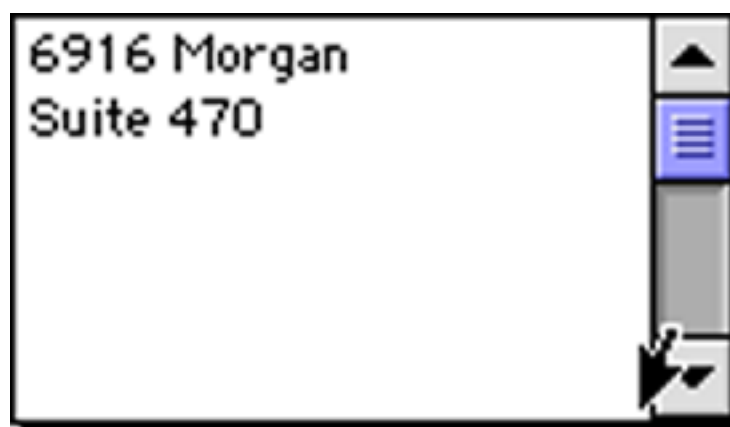


Now you can type additional lines into the Input Box. Press **Return** to start a new line.



Once you change the size of the Input Box, Panorama remembers the size permanently. In the data sheet, the Input Box size is remembered separately for each column, while in a form the size is remembered for each individual data cell object. (Of course Panorama will forget the sizes if you **Close** or **Quit** without saving the database.)

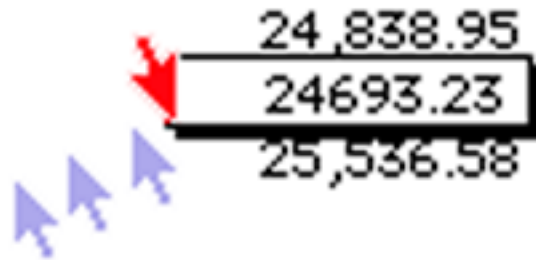
If you make the Input Box more than one inch high, a scroll bar is added on the right hand side of the box. The scroll bar allows you to enter and edit up to 32,767 characters per data cell.



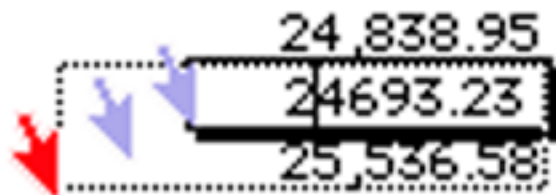
**Tip:** The scroll bar is actually outside the Input Box. To change the size of the Input Box you must click in the bottom corner of the box itself, just to the left of the scroll bar (as shown above). Do not try to drag the corner of the scroll bar.

### Expanding a Right Justified Input Box

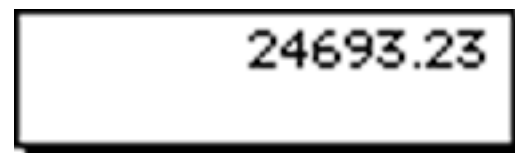
If you are editing a right flush data cell, move the mouse to the lower left corner of the box instead of the lower right hand corner.



Once the mouse is over the lower left hand corner you can expand down and/or to the left.



Now you can edit the text in the expanded input Box. Press **Enter** when you are finished.



### Editing Cells Within a Form

All of the previous examples have shown editing cells within the data sheet. However, a form may contain data cells also. Unlike the data sheet, the cells on a form may be arranged any way you like. They can also be more than one line high.

A screenshot of a software window titled "Contacts:Person". The window contains a form with several fields: "Name" (John Smith), "Title" (Sales Manager), "Company" (Acme Widgets), "Address" (12 Harmony Lane, Suite 15, Huntington Beach, CA 92648), "Country", "Phone" ((999) 555-1234), "Fax" ((999) 555-1248), "Email", and "Notes" (Customer since 1987). The "Name" field is currently selected. On the left side of the form, there is a vertical toolbar with icons for zooming, scrolling, and other navigation functions. At the bottom of the window, a status bar shows "102 visible/102 total".



Just as in the data sheet, you double click on a cell to edit it.



You can also move the mouse over the corner of the cell and drag to expand the Input Box.



As an alternative to data cells, a form may be designed with **Text Editor SuperObjects**. Text Editor SuperObjects allow you to edit text right in the form window—no double click is required. You can simply click or drag on the text to begin editing. Press **Enter** when you are finished. The illustration below shows the effect of double clicking on the word **Harmony**. As you can see, instead of opening an Input Box this selects the word for editing.



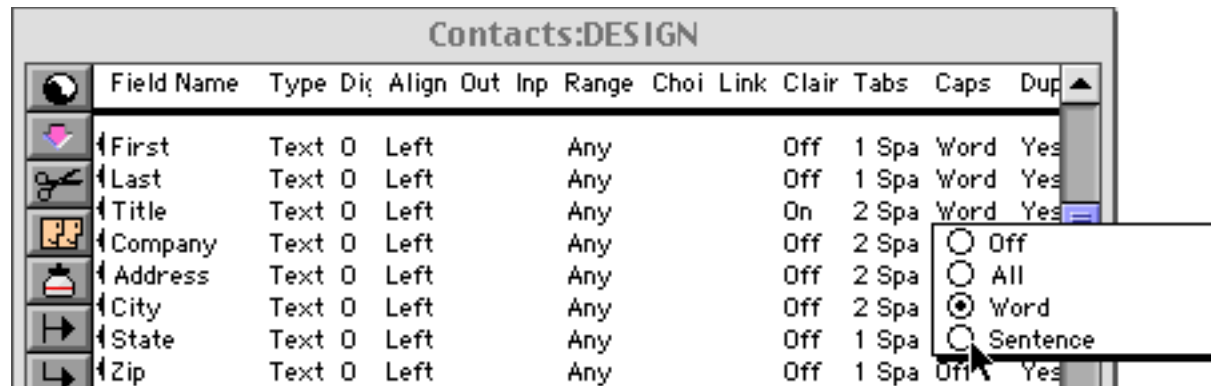
Since the Text Editor SuperObject doesn't use an Input Box, you cannot expand the size of the editing area "on-the-fly" the same way you can with data cells. The editing area must be defined in advance. On the other hand, the Text Editor SuperObject doesn't require the extra double click, and works more like other standard applications you may be used to. See "[Text Editor SuperObject](#)" on page 307 to learn how to create a form with Text Editor SuperObjects.

## Data Entry Accelerators

Data entry is probably the most tedious task you'll face while using your computer. Let's face it, no one enjoys keying in data. The best way is to get someone else to do it! But since that usually isn't possible, Panorama includes a number of features that help accelerate data entry. You don't have to use any of these features, but when you do, you'll find that ugly data entry tasks are finished faster and more accurately.

## Automatic Capitalization

Panorama has four options for automatic capitalization: **off**, **all**, **word**, and **sentence**. You can set up automatic capitalization with the **Field Properties** dialog or with the **Caps** column in the design sheet.



The **All** option tells Panorama to capitalize every letter entered into the field. Use this option for fields containing abbreviations—CA, NY, etc.

The **Word** option tells Panorama to capitalize the first letter of each word. Use this option for fields containing names, addresses, etc.

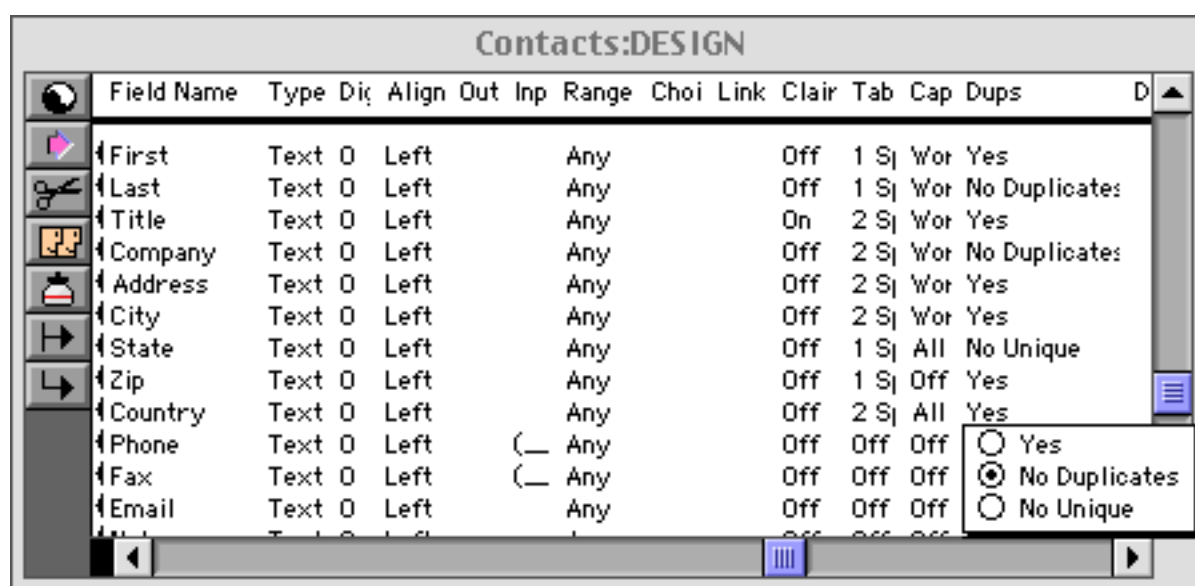
Sometimes you may need to override Panorama's automatic word capitalization. If you need an extra capital letter simply press the **Shift** key (for instance **McDonald**). If you want the first letter of a word to be lower case type the letter twice (**Bank Oof America**), then delete the upper case letter (**Bank of America**).

The **Sentence** option tells Panorama to capitalize the first letter of each sentence. Use this option for fields containing paragraphs of text—for instance catalog descriptions or correspondence.

## Checking for Duplicate Data

Panorama usually does not care if you enter duplicate information into a database. However, if you wish you can ask Panorama to check for duplicate data every time you enter or edit a data cell in a given field.

Panorama has three options for checking duplicate data—**Yes**, **No Duplicates** and **No Unique**. You can set up duplicate checking with the **Field Properties** dialog or with the **Dups** column in the design sheet.



The **Yes** option simply tells Panorama to allow duplicates. This is the default.

Use the **No Duplicates** option to make sure that a value is not entered more than once. For instance, a check-book database should never have duplicate check numbers.

The **No Unique** option tells Panorama to warn you if you attempt to enter a value that is not already in the database. For instance if a field contains only **Yes/No** values, this option would warn you if you attempted to enter **True** or **False**.

When Panorama encounters a duplicate or unique value (depending on the option), it warns you. However, it does not prevent you from entering the value. You are given the option of entering the data even though it conflicts with the existing data—it's up to you.



### Checking for Duplicates in Existing Data

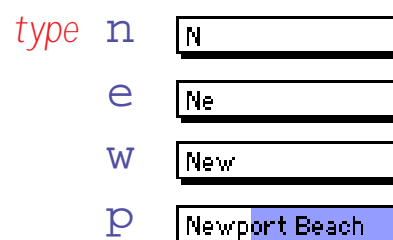
Checking for duplicates only happens when new data is typed into the database. Panorama does not check data that has already been entered, and it does not check data that is imported or pasted into the database. There are, however, several techniques for checking for duplicates in existing data. See “[Select Duplicates](#)” on page 366 of the **Panorama Handbook** to learn how to use the **Select Duplicates** command. Another method is to sort the data and then use the **UnPropagate** command to identify the duplicates (by searching for blank cells). See “[Using UnPropagate to Eliminate Duplicates](#)” on page 448 of the **Panorama Handbook** for details.

### Clairvoyance®

Many databases contain fields where the same information is repeated over and over. For instance, a checkbook will contain the same bills month after month—rent, phone, utilities, charge cards. Another example is an inventory database that contains many items from each vendor, with the vendor name repeated over and over. Panorama’s Clairvoyance feature anticipates when you are about to enter data that has been entered before, and completes the entry for you. This can save you a lot of typing, and helps improve consistency as well.

### How Clairvoyance® Works

How can Panorama anticipate what you are about to type? The secret lies in Panorama’s ability to scan the database in a fraction of a second. When you are using Clairvoyance, Panorama scans the entire database each time you enter a character. As it scans the database, it checks the characters you have typed against the data already in the database. When there is only one possible match, Clairvoyance guesses that you are about to repeat yourself and completes the word or phrase for you.

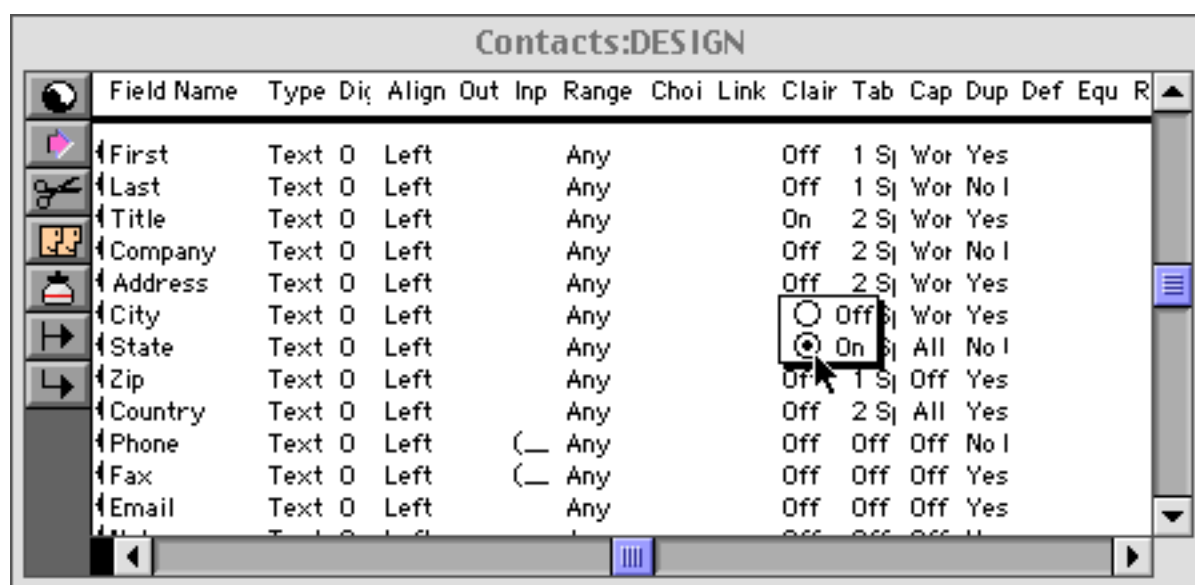


Of course, Clairvoyance can only be helpful when you are repeating a word or phrase that is already in the database. If you are entering a new word or phrase, Clairvoyance cannot help you—but it won't get in your way, either. As you type in a new word or phrase Clairvoyance may guess that you are entering an old word or phrase. Just keep typing, and Clairvoyance will automatically erase its guess when it no longer matches what you have typed.

type	n	N
	e	Ne
	w	New
	p	Newport Beach
	o	Newport Beach
	r	Newport Beach
	t	Newport Beach
space		Newport Beach
	n	Newport N
	e	Newport Ne
	w	Newport New
	s	Newport News

#### Turning Clairvoyance® On or Off

Clairvoyance can be turned on or off with the **Field Properties** dialog box (Setup Menu) or with the **Clairvoyance** column in the design sheet.



#### Clairvoyance® Helps Insure Data Consistency

One problem when building large databases is making sure that information always gets entered the same way, especially when more than one person is keying in the data. For example, a single company could be entered in your inventory database many ways—

```
Fuji
Fuji, Inc
Fuji USA
Fuji Photo, Inc
Fuji Photo Film USA
Fuji USA, Inc.
```

Clairvoyance helps solve this problem by accurately repeating the information time after time. You may find that Clairvoyance's ability to insure data consistency is more important than the keystroke savings.

### Clairrows

When you hold down the **Command** key (Mac) or **Control** key (Windows), the up and down arrows on the keyboard become clairvoyant arrows, or “clairrows.” With the key held down you can use the arrows to scan through the values that are already in the database. Each time you press **Command/Control-Down Arrow** the next value appears, while each time you press **Command/Control-Up Arrow** the previous value appears. You can scan through the values until you find the information you are looking for, then press the **Enter** key to enter the value.

To give the clairrows a head start you can type in the first few letters of the information you are looking for. For example, suppose that you are looking through a travel database for a particular Best Western Hotel. Start by typing **Best**, then press **Command/Control-Down Arrow**. The first hotel with a name beginning with **Best** will appear. Each time you press **Command/Control-Down Arrow** the name of next hotel (alphabetically) will appear—for example **Best Western Aspenalt Lodge**, **Best Western Bar X Motel**, **Best Western Boulder Inn**, etc. Press **Command/Control-Up Arrow** to move backwards through the hotel names. Continue until the hotel you are looking for appears, then press **Enter**.

type	b	<input type="text" value="B"/>
	e	<input type="text" value="Be"/>
	s	<input type="text" value="Bes"/>
	t	<input type="text" value="Best"/>
Cmd/Ctl-Down	Arrow	<input type="text" value="Best Western Aspenalt Lodge"/>
Cmd/Ctl-Down	Arrow	<input type="text" value="Best Western Bar X Motel"/>
Cmd/Ctl-Down	Arrow	<input type="text" value="Best Western Boulder Inn"/>
Cmd/Ctl-Down	Arrow	<input type="text" value="Best Western Caravan Motel"/>
Cmd/Ctl-Up	Arrow	<input type="text" value="Best Western Boulder Inn"/>
Enter		<input type="text" value="Best Western Boulder Inn"/>

### Input Patterns

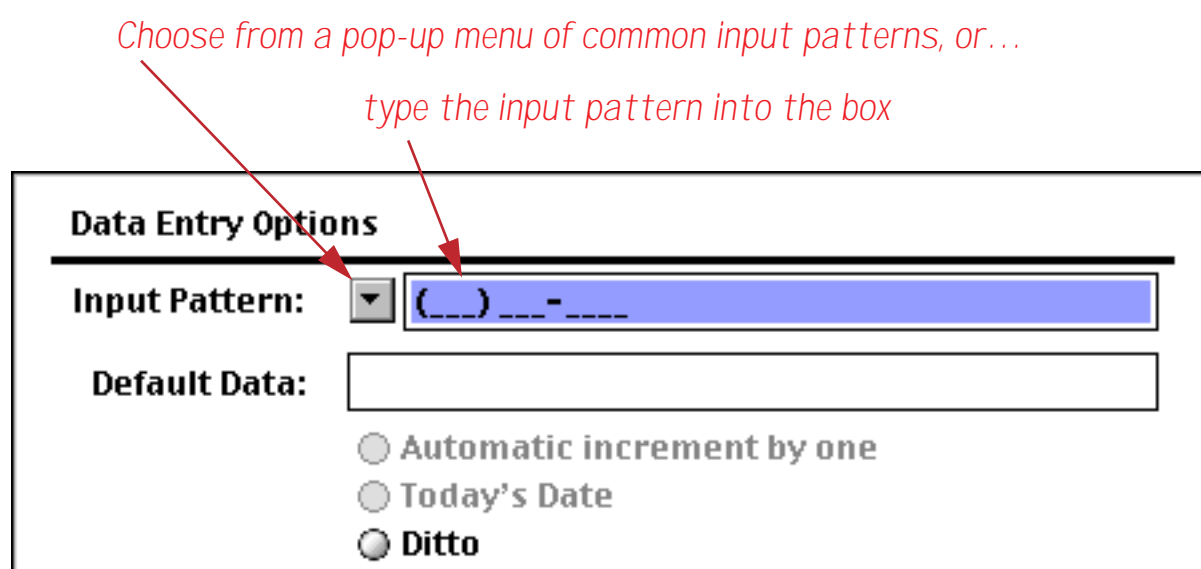
Sometimes you may wish to force the data being entered into a specific pattern. For instance in the United States and Canada long distance phone numbers almost always use the pattern (999) 999-9999. Panorama's **Input Pattern** can take care of entering the pattern for you. Once the pattern is set up, you only type the actual data (in this case the digits of the phone number). Panorama combines the data you enter with the pattern to produce the actual data. For example, combining the input pattern `( _ _ _ ) _ _ _ - _ _ _ _` with **3124562468** produces the data **(312) 456-2468**.



An input pattern consists of a string of characters with an underscore in each spot where actual data will be entered. The input pattern is just like fill in the blanks, but instead of filling in the blanks you fill in the underscores. (Press Shift-Dash to enter the underscore character. The dash key is in the top row of the keyboard, just to the right of the 0 key.) The table below lists some common input patterns.

Type of Data	Input Pattern	Example
Phone Number	( _ _ _ ) _ _ _ - _ _ _ _	(312) 456-2469
Social Security Number	_ _ _ - _ _ _ - _ _ _ _	234-54-5476
License Plate	_ _ _ _ / _ _ _ _	AGB 287
Date	_ _ / _ _ / _ _	03/24/05
Time	_ _ : _ _ : _ _ _ _	11:24:36 PM

You can set up the input pattern with the Field Properties Data Entry Options sub-dialog (open the **Field Properties** dialog and press the **Data Entry** button).



You can also set up the pattern with the **Input Pattern** column of the design sheet. (See “[The Design Sheet](#)” on page 105 if you are not already familiar with using the design sheet.)

Field Name	Type	Dig	Align	Out	Input Pattern	Range	Choi	Link	Clair	Tab	Cap
First	Text	0	Left			Any		Off	1	Sj	Wor
Last	Text	0	Left			Any		Off	1	Sj	Wor
Title	Text	0	Left			Any		On	2	Sj	Wor
Company	Text	0	Left			Any		Off	2	Sj	Wor
Address	Text	0	Left			Any		Off	2	Sj	Wor
City	Text	0	Left			Any		On	2	Sj	Wor
State	Text	0	Left			Any		Off	1	Sj	All
Zip	Text	0	Left			Any		Off	1	Sj	Off
Country	Text	0	Left			Any		Off	2	Sj	All
Phone	Text	0	Left		( _ _ _ ) _ _ _ - _ _ _ _	Any		Off	Off	Off	
Fax	Text	0	Left		( _ _ ) _ _ _ - _ _ _ _	Any		Off	Off	Off	
Email	Text	0	Left		_ _ @ _ . _ _ _ . _ _ _	Any		Off	Off	Off	

**Tip:** Input patterns should not be used with numeric fields. If you want to add a pattern to a numeric value, you should use an output pattern (see “[Numeric Output Patterns](#)” on page 117).

### Entering Data with an Input Pattern

This illustration shows an example of entering a phone number with an input pattern.

type 3	<input type="text" value="(3"/>
1	<input type="text" value="(31"/>
2	<input type="text" value="(312"/>
4	<input type="text" value="(312) 4"/>
5	<input type="text" value="(312) 45"/>
6	<input type="text" value="(312) 456-"/>
2	<input type="text" value="(312) 456-2"/>
4	<input type="text" value="(312) 456-24"/>
6	<input type="text" value="(312) 456-246"/>
8	<input type="text" value="(312) 456-2468"/>

The input pattern is only active when you are adding new characters at the end of the text. It does not adjust the data when you are inserting text in the middle of the cell. For example, it does not prevent you from creating a four digit area code, like this:

<i>click in middle of text</i>	<input type="text" value="(312) 456-2468"/>
type 4	<input type="text" value="(3142) 456-2468"/>

### Using Input Patterns with Dates

The purpose of input patterns is to save keystrokes in data entry by inserting constantly occurring dashes, colons, parentheses, or other punctuation. When it comes to date fields, you must decide how you like to enter dates. Using the input pattern `__ / __ / __` removes the need to type `/`'s (or some other separator) between the month, day and year. However, using the pattern requires that you type leading zeros in front of single digit months and days. For instance, to enter **January 1st** you must type **0101**. Without the pattern you can enter a single digit, for example **1/1**. (Keep in mind that Panorama allows any non-numeric character as the separator, so you could also type **1.1** on a numeric keypad—very fast.) It's up to you which method you prefer.

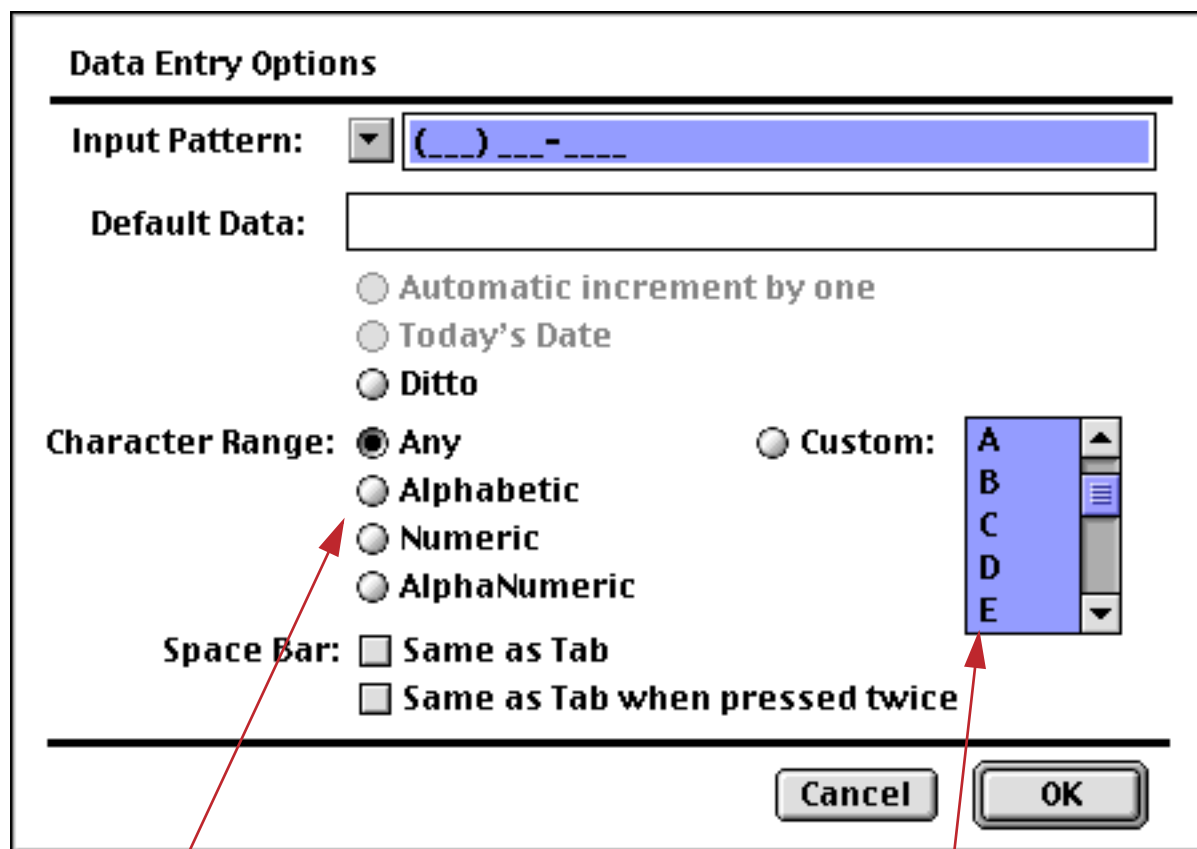
One other point to keep in mind—the input pattern can interfere with Panorama's Smart Date feature ("[Entering Dates](#)" on page 260). For example, with a pattern if you attempt to enter **yesterday** you will get **ye/st/erday**, and if you enter **tuesday** you will get **tu/es/day**. You can go back and edit out the `/`'s, but if you are going to use Smart Dates frequently you might want to forego the input pattern.

An input pattern can be used to override Panorama's century rounding feature. If you want to enter all dates in the 20th century you can use the pattern `__ / __ / 19__`. If you are using this pattern then Panorama will treat **030423** as **3/4/1923**, not **3/4/2023**. (Remember, Panorama normally rounds the year to the nearest century (within 50 years) if you do not specify all four digits of the year.)

### Restricting Character Types

Panorama normally allows you to enter any character that can be typed from the keyboard. If necessary you can restrict the kinds of characters that can be entered into each field. Panorama has five different character restriction levels—**Any**, **Alphabetic**, **Numeric**, **AlphaNumeric**, and **Custom**.

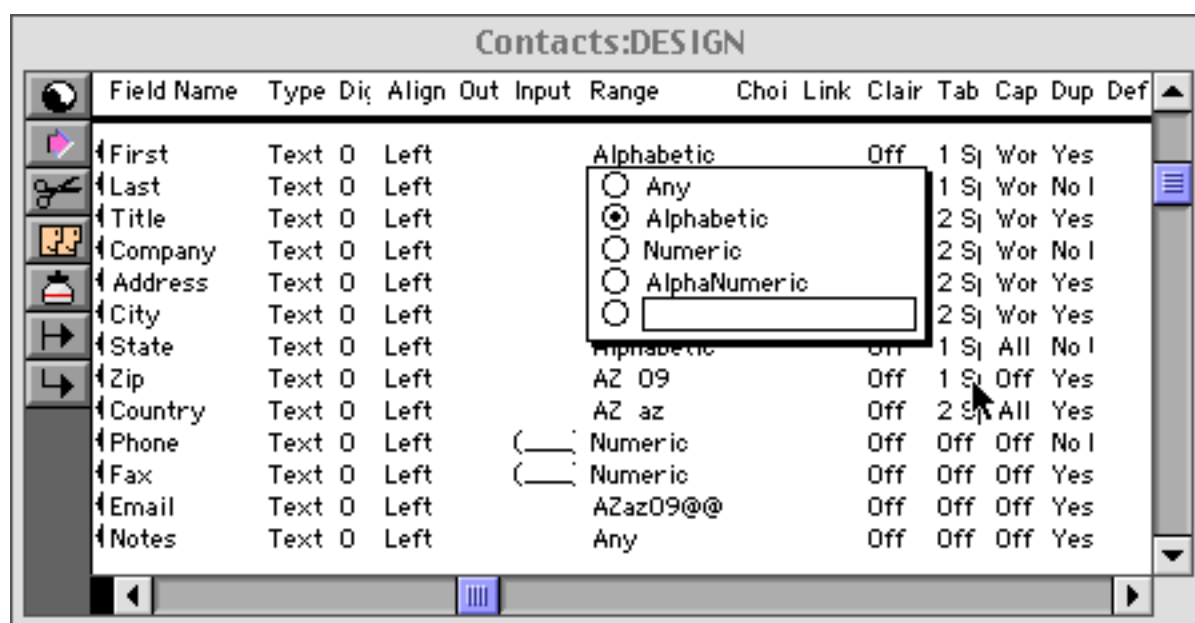
You can set up character entry restrictions with the Field Properties Data Entry Options sub-dialog (open the Field Properties dialog and press the **Data Entry** button).



*Choose a standard range, or...*

*design your own custom range (see text)*

You can also set up the character range with the **Range** column of the design sheet. (See [“The Design Sheet”](#) on page 105 if you are not already familiar with using the design sheet.)



The **Any** “restriction” really isn’t a restriction at all—it allows any kind of text—letters, numbers, spaces, or punctuation. Panorama lets you type in anything you want with no restrictions. This is the default option.

The **Alphabetic** restriction allows only letters (A-Z and a-z) and spaces. The letters may be either upper or lower case. If you attempt to type in a non-alphabetic character (a number, for example) Panorama will beep and ignore the character.

The **Numeric** restriction allows only digits (0...9), periods, minus sign, and the letter E (for scientific notation).

The **AlphaNumeric** restriction allows both letters and numbers, as well as spaces.

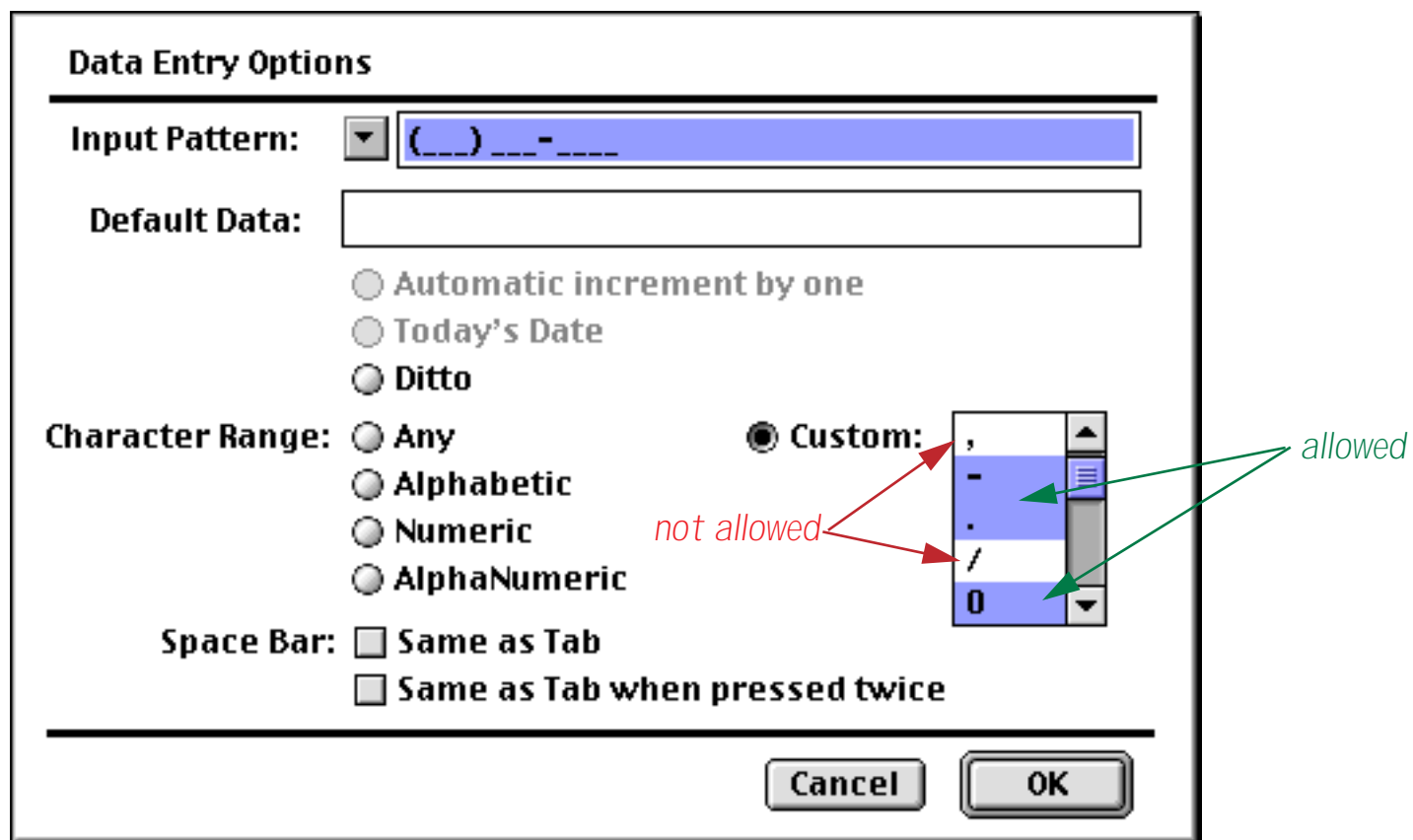
### Custom Character Restrictions

The **Custom** option allows you to exactly specify the characters you want to allow and disallow in this field. The actual characters allowed are defined by one or more pairs of characters. Each pair specifies a range of characters that are allowed. For example the pair **09** would allow all characters in the range 0...9, while the pair **az** would allow all lower case letters. You can combine several pairs to create a more complex range, for example **az09** for all lower case letters or numbers. A pair may specify a single character as both the beginning and end of the range, for instance **%%** (only the percent symbol allowed), or **09%%** (numerals and the percent symbol, but not the decimal point). If you wish to allow spaces, one of the pairs should be a pair of spaces, for instance **AZ az09**. To preview the effect of a character range you can use the ASCII Chart wizard.

The table below shows some common examples of custom character restrictions. For each range a sample of Ok data and bad data is shown, with the disallowed characters shown in **red**.

Custom Range	Ok Data	Bad Data	Comments
<b>09</b>	936	923.77	Only digits, no decimal point, spaces or other punctuation allowed.
<b>09..</b>	156.23	1,294.48	Basic fixed point numbers
<b>09..%%</b>	67.82%	67.82 %	Percentages (no spaces)
<b>09//</b>	5/23/02	March 1st	Numeric format dates
<b>09::</b>	1:24:83	1:24 PM	Time
<b>09 : :AAaaMMmmPPpp</b>	5:32 PM	5:32 DL	Time (am/pm)
<b>AZ</b>	SEATTLE	Seattle	Upper case letters only—no punctuation or lower case letters
<b>Azaz</b>	John	John Smith	Letters but no spaces
<b>AZaz@..</b>	sue@my.net	sue\$my.net	Handy for email addresses
<b>09 (( ))--</b>	(213) 444-1234	342-3982 ext 12	Basic US phone numbers
<b>!~</b>	Check#	El Niño	Everything ok except spaces, international characters and special symbols
<b>!Û</b>	Niño	El Niño	Everything ok except spaces

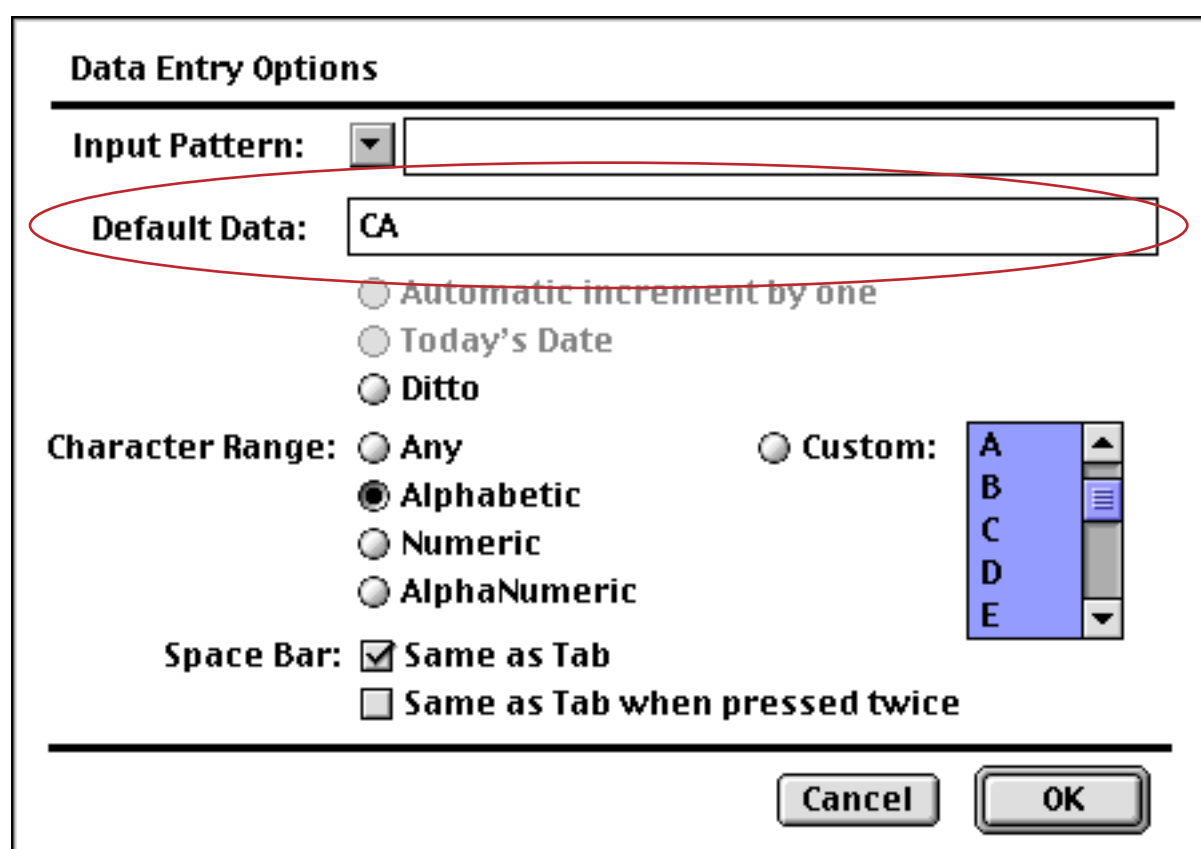
If you are using the **Field Properties Data Entry Options** sub-dialog, you can set up a custom restriction by selecting characters from the scrolling list. Only selected characters will be allowed. This dialog will automatically set up the character pairs as specified in the last paragraph.



When you press the **OK** button Panorama will convert the list into a series of character ranges as described in the previous section. If you want to see what the custom range you have created looks like, open the design sheet.

### Default Values

When a new record is added to a database, it is usually completely empty. You can, however, set up a default value for each field. One way to set up default values is with the **Field Properties Data Entry Options** sub-dialog (open the **Field Properties** dialog and press the **Data Entry** button). The dialog below is for a **State** field which defaults to **CA** (California).





You can also set up the default with the **Value** column of the design sheet. (See “[The Design Sheet](#)” on page 105 if you are not already familiar with using the design sheet.)

Field Name	Type	Di	Align	Out	Inp	Range	Chc	Link	Clair	Tab	Cap	Dup	Default	Equ
First	Text	0	Left			Alpha			Off	1	Sj	Wor	Yes	
Last	Text	0	Left			Alpha			Off	1	Sj	Wor	No	I
Title	Text	0	Left			AZ a			On	2	Sj	Wor	Yes	
Company	Text	0	Left			AZ a			Off	2	Sj	Wor	No	I
Address	Text	0	Left			AZ a			Off	2	Sj	Wor	Yes	
City	Text	0	Left			AZ a			On	2	Sj	Wor	Yes	
State	Text	0	Left			Alpha			Off	1	Sj	All	No	I
Zip	Text	0	Left			AZ 0			Off	1	Sj	Off	Yes	
Country	Text	0	Left			AZ a			Off	2	Sj	All	Yes	USA
Phone	Text	0	Left			( _ Nume			Off	Off	Off	No	I	
Fax	Text	0	Left			( _ Nume			Off	Off	Off	Yes		
Email	Text	0	Left			AZazl			Off	Off	Off	Yes		
Notes	Text	0	Left			Any			Off	Off	Off	Yes		

The simplest default is a fixed value, as shown in the example above. For example you might want the Country field to default to your home country, a shipping field to default to your preferred shipper. Once defaults are set up, they are automatically entered whenever a new record is created.

Contacts:Person (2 Column)

Name [Redacted] Phone [Redacted]

Title [Redacted] Fax [Redacted]

Company [Redacted] Email [Redacted]

Address [Redacted] Notes [Redacted]

Country USA

103 visible/103 total

*new record with default values*

### Default to Today's Date

To default to today's date use the default value **today**.

Field Name	Type	Di	Align	Out	Inp	Range	Choi	Link	Clair	Tab	Cap	Dup	Default	Equ	Reac	Writ	Wid	Notes
Date	Date	0	Left			Any			Off	Off	Off	Yes	today		0	0	7	
CkNum	Num	0	Right			Any			Off	Off	Off	Yes	+		0	0	4	
PayTo	Text	0	Left			Any			On	Off	Wor	Yes			0	0	16	
Category	Text	0	Left			Any			On	Off	Wor	Yes			0	0	12	
Debit	Num	2	Right	#,.		Any			Off	Off	Off	Yes			0	0	9	
Credit	Num	2	Right	#,.		Any			Off	Off	Off	Yes			0	0	9	
Balance	Num	2	Right	#,.		Any			Off	Off	Off	Yes			0	0	9	
Memo	Text	0	Left			Any			Off	Off	Off	Yes			0	0	22	

If you are using the Data Entry Options dialog, you can simply click on the **Today's Date** radio button.

**Data Entry Options**

Input Pattern:

Default Data: **Today**

Automatic increment by one  
 **Today's Date**  
 Ditto

Note: The **today** default only works for fields that use the date data type. If you use the default value **today** with a text field you will simply get the word today. See “[Dates](#)” on page 121 for more information on the date data type.

#### “Ditto” Defaults Based on the Previous Record

Instead of being fixed, a default value can be based on the data in the previous record. You can produce this type of “**ditto**” default by using the default value `"`. This is the quote character, which is produced by holding down the **Shift** key and pressing the `"` key (just to the right of the **semicolon** key). (Some people mistakenly call this the double-quote character.)

Field Name	Type	Dir	Align	Out	Inp	Range	Choi	Link	Clair	Tab	Cap	Dup	Defar	Equ
First	Text	0	Left			Alphat			Off	1 Sj	Wor	Yes		
Last	Text	0	Left			Alphat			Off	1 Sj	Wor	No	I	
Title	Text	0	Left			AZ az			On	2 Sj	Wor	Yes		
Company	Text	0	Left			AZ az			Off	2 Sj	Wor	No	I	
Address	Text	0	Left			AZ az			Off	2 Sj	Wor	Yes		
City	Text	0	Left			AZ az			On	2 Sj	Wor	Yes	"	
State	Text	0	Left			Alphat			Off	1 Sj	All	No	I	CA
Zip	Text	0	Left			AZ 09			Off	1 Sj	Off	Yes	"	
Country	Text	0	Left			AZ az			Off	2 Sj	All	Yes	USA	
Phone	Text	0	Left			( _ Numer			Off	Off	Off	No	I	
Fax	Text	0	Left			( _ Numer			Off	Off	Off	Yes		
Email	Text	0	Left			AZaz0			Off	Off	Off	Yes		
Notes	Text	0	Left			Any			Off	Off	Off	Yes		

When you create a new record, the fields using the ditto default will contain the same values as the previous record. In this illustration a new record has been added with four default values, two fixed and two “ditto.”

Company	Address	City	State	Zip	Country
	7292 Delvin Wy	South San Francis	CA	94080	
San Francisco Lumber	854 14th St	San Francisco	CA	94103	
N.L. Plumbing	759 2Nd Ave	San Francisco	CA	94118	
	7265 Lakeland Drive	Roseville	CA	95661	
	625 S.E. High	Pullman	WA	99163	
GW Printing	779 Arnold Rd	Newton Centre	MA	02159	
	15 Lownds Drive	Windsor Locks	CT	06096	
	31 Cross Highway	Westport	CT	06880	
Stephen's Appliances	90 Duane Lane	Demarest	NJ	07627	
	7718 Odell St	Bronx	NY	10462	
		Bronx	CA	10462	USA

*"ditto" defaults for City and Zip*

*fixed defaults for State and Country*

#### Automatically Incrementing Defaults (1, 2, 3, ...) Based on the Previous Record

For a numeric field you can specify a default that is created by adding to the previous value in the field. To do this, use a default of **+nn**, where **nn** is the amount to add to the previous value. For example **+1** causes the value to increment by one for each new record.

Field Name	Type	Dig	Align	Out	Inp	Range	Choi	Link	Clair	Tab	Cap	Dup	Def	Equ	Reac	Writ	Wid	Notes
Date	Date	0	Left			Any		Off	Off	Off	Yes	tod.		0	0	7		
CkNum	Num	0	Right			Any		Off	Off	Off	Yes		+1	0	0	4		
PayTo	Text	0	Left			Any		On	Off	Wor	Yes			0	0	16		
Category	Text	0	Left			Any		On	Off	Wor	Yes			0	0	12		
Debit	Num	2	Right	#,.		Any		Off	Off	Off	Yes			0	0	9		
Credit	Num	2	Right	#,.		Any		Off	Off	Off	Yes			0	0	9		
Balance	Num	2	Right	#,.		Any		Off	Off	Off	Yes			0	0	9		
Memo	Text	0	Left			Any		Off	Off	Off	Yes			0	0	22		

You can use any number, even a negative number like **+5**. This default would cause Panorama to add negative 5 (same as subtracting 5) to the value each time a new record is created. If the numeric type allows it, you can even use non-integer values like **2.5** or **0.1**.

As new records are added to the database, they are numbered automatically, like this.

Date	CkNum	PayTo	Category	Debit	Credit	Balance
09/21/99	2295	Advertiser's Mailing Ser	Postage	67.00		60,155.4
09/26/99	2296	TesLabe	Fixed Assets	2,465.00		57,690.4
09/26/99	2297	AC Label Company	Advertising	205.97		57,484.5
09/28/99	2298	Graphic Depot	Advertising	344.00		57,140.5
09/28/99	2299	Advertiser's Mailing Ser	Advertising	167.00		56,973.5
03/08/00	2300					
03/08/00	2301					
03/08/00	2302					
03/08/00	2303					

*automatically incrementing check numbers*  
*this field set up to default to today's date*

Be sure to keep in mind that an incrementing default like **+1** is based on the previous record, not on the largest value in the entire database. So if you insert a record in the middle of the database, the incremented value will be based on the value just above it, not on the value at the end of the database.

Date	CkNum	PayTo	Category	Debit	Credit	Balance
09/21/99	2295	Advertiser's Mailing Ser	Postage	67.00		60,155.4
09/26/99	2296	TesLabe	Fixed Assets	2,465.00		57,690.4
09/26/99	2297	AC Label Company	Advertising	205.97		57,484.5
03/08/00	2298					
03/08/00	2299					
03/08/00	2300					
09/28/99	2298	Graphic Depot	Advertising	344.00		57,140.5
09/28/99	2299	Advertiser's Mailing Ser	Advertising	167.00		56,973.5
03/08/00	2300					
03/08/00	2301					
03/08/00	2302					

*records inserted in the middle have incorrect numbers*

If you want to generate a unique incrementing number for use as a record ID (for instance an invoice number or check number), use the technique described in the next section.

### Creating a Unique Record Number

Many databases applications require that each record contain a unique number that can be used to identify the record. Common examples include invoice numbers, batch ID's, employee numbers, etc. Panorama can automatically assign a unique number to each new record as it is created, even if several people are using the database simultaneously over a network.

The field containing the record number must be a numeric field. To specify that this field should contain a unique record number, the default should be + . Do not specify any increment value, just use a single + character.

Field Name	Type	Diç	Align	Out	Inp	Range	Choi	Link	Clair	Tab	Cap	Dup	Def	Equ	Reac	Writ	Wid	Notes
Date	Date	0	Left			Any			Off	Off	Off	Yes	tod.		0	0	7	
CkNum	Num	0	Right			Any			Off	Off	Off	Yes	+		0	0	4	
PayTo	Text	0	Left			Any			On	Off	Wor	Yes			0	0	16	
Category	Text	0	Left			Any			On	Off	Wor	Yes			0	0	12	
Debit	Num	2	Right	#,.		Any			Off	Off	Off	Yes			0	0	9	
Credit	Num	2	Right	#,.		Any			Off	Off	Off	Yes			0	0	9	
Balance	Num	2	Right	#,.		Any			Off	Off	Off	Yes			0	0	9	
Memo	Text	0	Left			Any			Off	Off	Off	Yes			0	0	22	

Each database contains a counter for keeping track of the next record number. Every time a new record is created the counter is incremented by one. Even if the record is later deleted, the number will never be re-used (unless you Quit Panorama or close the database without saving your changes, or unless you reset the counter manually as described below).

If necessary you can manually change the record counter dialog. See “[The Privilege Dialog](#)” on page 204 of the [Panorama Handbook](#) for details.

### Automatic Calculations

The **Equation** column in the design sheet allows you to set up a formula for calculating the value of a field based on the values in other fields.

### Spreadsheet Mode Calculations

If you’ve ever used a spreadsheet you’ll be very comfortable with Panorama’s default **Spreadsheet Mode** for automatic calculations (see “[Procedure Mode Calculations](#)” on page 151 of the [Panorama Handbook](#) for an alternative calculation mode). In this mode you simply place the formula for calculating a field into the Equation column of the design sheet for that field. To illustrate this we’ll use this simple database with four numeric fields for data entry (**A**, **B**, **C** and **D**) and two fields that we will be calculated (**Total** and **Avg**).

City	A	B	C	D	Total	Avg
Anaheim	3.25	7.31	14.82	1.93	27.31	6.83
Bakersfield	2.29	8.26	12.91	3.02	26.48	6.62
Camarillo	2.83	9.19	15.11	2.54	29.67	7.42
Fullerton	2.59	8.25	13.48	1.77	26.09	6.52
Laguna Beach	3.06	9.45	12.5	3.01	28.02	7.00
Newport Beach	3.18	7.22	12.32	2.38	25.10	6.27
Whittier	3.67	5.23	14.24	3.27	26.41	6.60



To automatically calculate the total and average simply enter the formulas in the appropriate rows of the **Equation** column in the design sheet.

Field Name	Type	Dir	Align	Out	Inp	Range	Choi	Link	Clair	Tab	Cap	Dup	Def	Equation	Reac	Writ	Wid	Notes
City	Text	O	Left			Any		Off	Off	Off	Yes				0	0	11	
A	Nume	Fic	Right			Any		Off	Off	Off	Yes				0	0	4	
B	Nume	Fic	Right			Any		Off	Off	Off	Yes				0	0	4	
C	Nume	Fic	Right			Any		Off	Off	Off	Yes				0	0	4	
D	Nume	Fic	Right			Any		Off	Off	Off	Yes				0	0	4	
Total	Nume	Fic	Right	#,.	#,.	Any		Off	Off	Off	Yes			A+B+C+D	0	0	4	
Avg	Nume	Fic	Right	#,.	#,.	Any		Off	Off	Off	Yes			(A+B+C+D)/4	0	0	4	

*formula to calculate total*  
*formula to calculate average*

To activate these formulas you need to create a new generation for this database (see “[Database “Generations”](#)” on page 105). Once you’ve done this you can start entering or updating information. In this illustration a new record has been added (**Diamond Bar**) and the first number typed in (but not entered into the database yet).

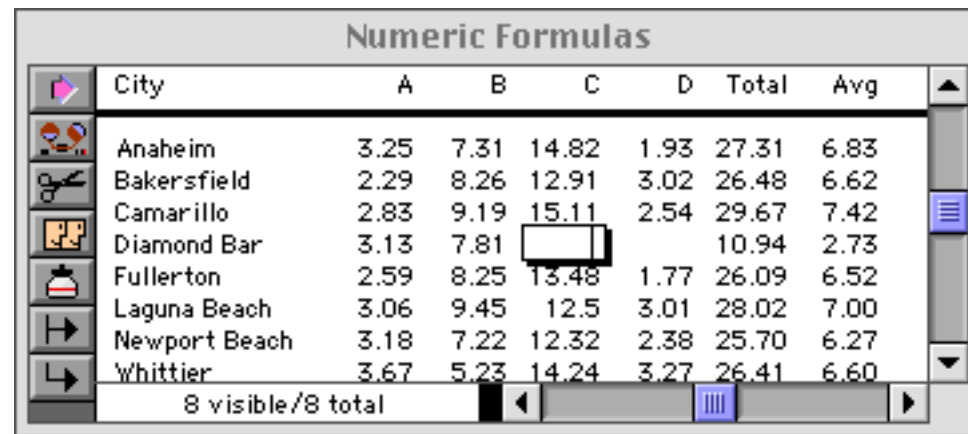
City	A	B	C	D	Total	Avg
Anaheim	3.25	7.31	14.82	1.93	27.31	6.83
Bakersfield	2.29	8.26	12.91	3.02	26.48	6.62
Camarillo	2.83	9.19	15.11	2.54	29.67	7.42
Diamond Bar	3.13					
Fullerton	2.59	8.25	13.48	1.77	26.09	6.52
Laguna Beach	3.06	9.45	12.5	3.01	28.02	7.00
Newport Beach	3.18	7.22	12.32	2.38	25.70	6.27
Whittier	3.67	5.23	14.24	3.27	26.41	6.60

As soon as the data is entered by pressing the **Tab** (or **Enter**) keys the formulas update the **Total** and **Avg** fields.

City	A	B	C	D	Total	Avg
Anaheim	3.25	7.31	14.82	1.93	27.31	6.83
Bakersfield	2.29	8.26	12.91	3.02	26.48	6.62
Camarillo	2.83	9.19	15.11	2.54	29.67	7.42
Diamond Bar	3.13				3.13	0.78
Fullerton	2.59	8.25	13.48	1.77	26.09	6.52
Laguna Beach	3.06	9.45	12.5	3.01	28.02	7.00
Newport Beach	3.18	7.22	12.32	2.38	25.70	6.27
Whittier	3.67	5.23	14.24	3.27	26.41	6.60

*formulas in design sheet update fields as data is entered*

As more data is entered the **Total** and **Avg** fields are updated instantaneously.



City	A	B	C	D	Total	Avg
Anaheim	3.25	7.31	14.82	1.93	27.31	6.83
Bakersfield	2.29	8.26	12.91	3.02	26.48	6.62
Camarillo	2.83	9.19	15.11	2.54	29.67	7.42
Diamond Bar	3.13	7.81			10.94	2.73
Fullerton	2.59	8.25	13.48	1.77	26.09	6.52
Laguna Beach	3.06	9.45	12.5	3.01	28.02	7.00
Newport Beach	3.18	7.22	12.32	2.38	25.70	6.27
Whittier	3.67	5.23	14.24	3.27	26.41	6.60

The **Total** and **Avg** fields will be updated any time the **A**, **B**, **C** or **D** fields are modified.

#### Procedure Mode Calculations

An alternate advanced mode for performing calculations is called **Procedure Mode**. In this mode little “mini-procedures” are triggered when data is entered into a cell. Each mini-procedure can contain one or more assignment statements. A mini-procedure can also trigger a real procedure. To learn how to use Procedure Mode see Chapter 7 of the **Panorama Handbook**.



# Chapter 8: Sorting



Most data is more useful when it is in some kind of order. Panorama's **Sort Menu** has several commands that can quickly sort your data into order.

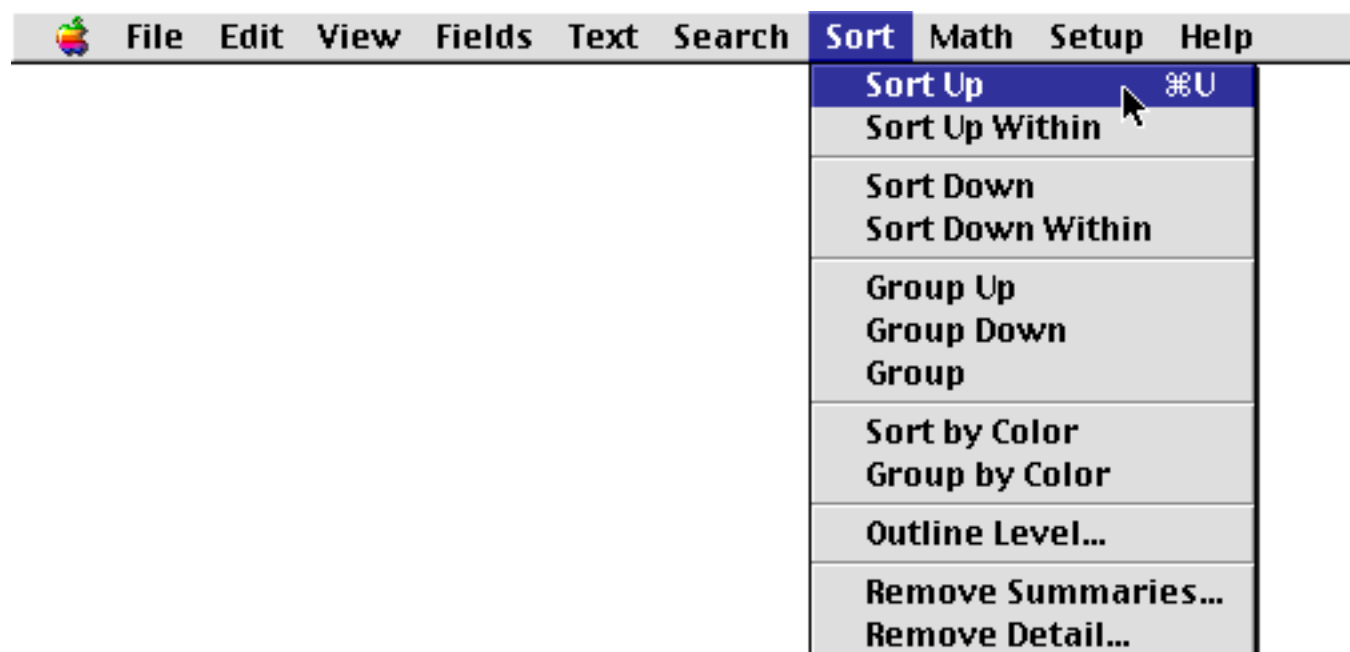
## Basic Sorting

Before sorting, you must choose the field to be sorted. For example, you might want to sort a mailing list database by name, city, state, or zip code. To choose the field you want to sort, just click on it. In a data sheet or view-as-list window, you can click on any cell in the column. If you are using a regular form (individual pages), click on the field you want to sort. (You must click on a data cell, not on an auto-wrap text object containing the field.)

First	Last	Title	Company	Address	City
Abe	Fierstein	Vice President	Van Nuys Lumber	1571 Haskell	Van Nuys
Randy	Cross	Owner	Randy's Appliances	133 Hunt Rd	Chelsford
Jeffrey	Rodman			2 Cary Rd	Chestnut
Steve	Jackson	Purchasing	Ann Arbor Lumber	389 Worden	Ann Arbor
Dick	Hardlee			4151 Polstar	Plano
Don	Meadows	Sales Manager	Austin Lumber	1144 A West 6th	Austin
Jerry	Bowen	Vice President	Peacock Video	2847 Peacock	Highland
Thom	Getchell	Customer Supp	Thom's Appliances	543 Laurel	Menlo Park
Brian	Smith	Owner	Brian's Appliances	1844 Tiburon	Hollister
David	Blair	Owner	DB Printing	869 W. Temple	Lenox
Keith	Baker	Sales Manager	Northgate Video	552 Northgate	Lindenhu
John	Sloan			79 Danube Way	Olympia
Guy	Porter		St. Louis Lumber	8702 Pershing	St. Louis
Steve	Gibson			57 Sunnyview	St. Peter
Chuck	Rouse	Agent	Hays Lumber	625 West 15th	Hays
Peter	Silvers	Customer Supp	P.S. Plumbing	9382 Hampson	New Orle
Michael	Cox	Purchasing	Dallas Lumber	4785 Velasco	Dallas
James	Mahan	Owner	J.M. Plumbing	1294 W. 31St	Los Ange
Gary	Gintz	President	Gary's Appliances	7436 35th S.W	Seattle

102 visible/102 total

To sort in ascending order (A to Z) use the **Sort Up** command. To sort in descending order (Z to A) use the **Sort Down** command. The **Sort Down** command is especially useful with numeric data if you want to rank the numbers from largest to smallest.



When the database is sorted, all the records are re-arranged. However the current record remains in the same spot in the window. For example, if you click on the name **Zabriskie** and then sort, the record containing **Zabriskie** will remain on the screen. The record will not be in the same spot in the file, however, because it is now near the end of the file with the other Z's. In this case the current record was **Keith Baker**, which is still the current record. However the database is now sorted from A to Z within the **Last** name field.

First	Last	Title	Company	Address	City
Keith	Baker	Sales Manager	Northgate Video	552 Northgate	Lindenhu
Nabil	Basir		Armonk Lumber	12 Upland Lane	Armonk
John	Bath	President	J.B. Plumbing	8864 Ave	Mendota
Jack	Beardsley	Sales Manager	Toledo Lumber	4964 Pelham	Toledo
Carl	Berg	Owner	C.B. Plumbing	161 Norton St	New Hav
Leslie	Bianchi			23 Oak St	Lexingto
Mary	Bilbury	Vice President	M.B. Plumbing	2754 Parkway	Beverly
Joseph	Bizzarri	Owner	JB Printing	7045 Mandel	Westche
David	Blair	Owner	DB Printing	869 W. Temple	Lenox
Al	Bodner			93 Valencia Lane	Clifton F
Jerry	Boone			6125 Park Drive	Travers
Jerry	Bowen	Vice President	Peacock Video	2847 Peacock	Highland
Yvonne	Broach			9330 Poitiers	Houston
Susan	Brown			783 Algonquin	Newport
Tom	Cane			8820 Sierra Court	Dublin
David	Cohn			307 Ronnie Drive	Buffalo
Michael	Cox	Purchasing	Dallas Lumber	4785 Velasco	Dallas
Anne	Crane			Grosse Pointe Shores	Grosse P
Randy	Cross	Owner	Randy's Appliances	133 Hunt Rd	Chelsfor

### Sorting By More Than One Field

It is often necessary to sort by more than one field at a time. For example, you may need to sort both first and last names, or cities and states. After you have sorted the database once, you can use the **Sort Up Within** (or **Sort Down Within**) commands to sort by additional fields.



For example to sort by city and state, first sort by the state.

Company	Address	City	State	Zip
M.B. Plumbing	2754 Parkway	Beverly Hills	CA	90210
Peacock Video	2847 Peacock	Highland	CA	92346
	783 Algonquin	Newport Beach	CA	93459
	8820 Sierra Court	Dublin	CA	94568
Herb's Appliances	206 Phelps St	San Francisco	CA	94124
M.D. Plumbing	518 Arneill Rd	Camarillo	CA	93010
	519 Leahy	Redwood City	CA	94061
Van Nuys Lumber	1571 Haskell	Van Nuys	CA	91409
Thom's Appliances	543 Laurel	Menlo Park	CA	94025
N.L. Plumbing	759 2Nd Ave	San Francisco	CA	94118
J.M. Plumbing	1294 W. 31St	Los Angeles	CA	90018
	5238 Quince	Upland	CA	91786
	8265 Leticia	San Clemente	CA	92672
Jim's Appliances	14189 8th	Newhall	CA	91321
Palo Alto Lumber	1828 Amaranta	Palo Alto	CA	94306
D.P. Plumbing	191 Treg Lane	Concord	CA	94518
	7265 Lakeland Drive	Roseville	CA	95661
Riverside Lumber	1901 Red Oak Drive	Riverside	CA	92509
Acme Widgets	12 Harmony Lane	Huntington Beach	CA	92648
Brian's Appliances	1844 Tiburon	Hollister	CA	95023
Latham Video	4792 Latham	Mountain View	CA	94041
P.T. Plumbing	1009 Secret Bay	Davis	CA	95616
	7292 Delvin Wy	South San Francisco	CA	94080
San Francisco Lumber	854 14th St	San Francisco	CA	94103
S.W. Plumbing	1175 Wilson Rd	Fountain	CO	80817
C.B. Plumbing	161 Norton St	New Haven	CT	06511
	15 Lownds Drive	Windsor Locks	CT	06096
	53 Clubhouse Drive	Woodbury	CT	06798
	31 Cross Highway	Westport	CT	06880
West Palm Beach Lumber	8206 13th Way	West Palm Beach	FL	33407
SM Printing	3894 11th Court	Jupiter	FL	33458
DB Printing	869 W. Temple	Lenox	IA	50851
Northgate Video	552 Northgate	Lindenhurst	IL	60046

As you can see, the records are now sorted by state. However, the cities are not sorted within each state. For example, **Davis, CA** should appear before **San Francisco, CA**, but it doesn't. In fact, within each state the records aren't sorted at all, they are still in the order they were in before they were sorted.

To complete our two column sort, click on the **City** field and use **Sort Up Within** to sort the cities. The **Sort Up Within** command leaves the states in order but sorts the cities within each state.

Company	Address	City	State	Zip
M.B. Plumbing	2754 Parkway	Beverly Hills	CA	90210
M.D. Plumbing	518 Arneill Rd	Camarillo	CA	93010
D.P. Plumbing	191 Treg Lane	Concord	CA	94518
P.T. Plumbing	1009 Secret Bay	Davis	CA	95616
	8820 Sierra Court	Dublin	CA	94568
Peacock Video	2847 Peacock	Highland	CA	92346
Brian's Appliances	1844 Tiburon	Hollister	CA	95023
Acme Widgets	12 Harmony Lane	Huntington Beach	CA	92648
J.M. Plumbing	1294 W. 31St	Los Angeles	CA	90018
Thom's Appliances	543 Laurel	Menlo Park	CA	94025
Latham Video	4792 Latham	Mountain View	CA	94041
Jim's Appliances	14189 8th	Newhall	CA	91321
	783 Algonquin	Newport Beach	CA	93459
Palo Alto Lumber	1828 Amaranta	Palo Alto	CA	94306
	519 Leahy	Redwood City	CA	94061
Riverside Lumber	1901 Red Oak Driv	Riverside	CA	92509
	7265 Lakeland Dri	Roseville	CA	95661
	8265 Leticia	San Clemente	CA	92672
Herb's Appliances	206 Phelps St	San Francisco	CA	94124
N.L. Plumbing	759 2Nd Ave	San Francisco	CA	94118
San Francisco Lumber	854 14th St	San Francisco	CA	94103
	7292 Delvin Wy	South San Francis	CA	94080
	5238 Quince	Upland	CA	91786
Van Nuys Lumber	1571 Haskell	Van Nuys	CA	91409
S.W. Plumbing	1175 Wilson Rd	Fountain	CO	80817
C.B. Plumbing	161 Norton St	New Haven	CT	06511
	31 Cross Highway	Westport	CT	06880
	15 Lownds Drive	Windsor Locks	CT	06096
	53 Clubhouse Drive	Woodbury	CT	06798
SM Printing	3894 11th Court	Jupiter	FL	33458
West Palm Beach Lurr	8206 13th Way	West Palm Beach	FL	33407
DB Printing	869 W. Temple	Lenox	IA	50851
D.S. Plumbing	683 Elm St	Batavia	IL	60510

102 visible/102 total

As you can see, the cities are now sorted within **California** and **Connecticut** (and within all of the other states with more than one record as well).

The **Sort Within** commands can be used over and over again to sort 3, 4, or more fields within each other. Always start with the regular **Sort Up** or **Sort Down** commands, then use **Sort Up Within** or **Sort Down Within** to sort each of the additional fields. For example, if you look closely you will notice that the three **San Francisco** records in the window above are not sorted by zip code. If you wanted them to be, you could simply click on the **Zip** field and then choose **Sort Up Within** again. The records will still be sorted by state and by city within state, but now they will also be sorted by zip code within city as well.

Riverside Lumber	1901 Red Oak Driv	Riverside	CA	92509
	7265 Lakeland Dri	Roseville	CA	95661
	8265 Leticia	San Clemente	CA	92672
San Francisco Lumber	854 14th St	San Francisco	CA	94103
N.L. Plumbing	759 2Nd Ave	San Francisco	CA	94118
Herb's Appliances	206 Phelps St	San Francisco	CA	94124
	7292 Delvin Wy	South San Francis	CA	94080
	5238 Quince	Upland	CA	91786
Van Nuys Lumber	1571 Haskell	Van Nuys	CA	91409
S.W. Plumbing	1175 Wilson Rd	Fountain	CO	80817
C.B. Plumbing	161 Norton St	New Haven	CT	06511

Note: There is an alternate way to sort multiple fields. This alternate method does not use the **Sort Within** command. Instead of sorting within, sort the fields in reverse order using the regular **Sort Up** or **Sort Down** commands. For example to sort by city within state, first click on the **City** field, then **Sort Up**, then click on the **State** field, and finally **Sort Up** again.

### Undo Sorting

The **Undo** command will undo the effect of the last **Sort** or **Sort Within** command, putting the database back into the original order. Remember that only the last sort can be undone. Once you sort again the original order cannot be restored (unless you have saved the file on the disk in the original order, then you can use **Revert to Saved** to restore the order.)

### Sorting Numbers and Dates

It is important to store numbers using the numeric data type, and dates using the date type. If you store numbers and dates using the text data type they will not sort correctly, as shown in this table.

Stored as Numeric (correct)	Stored as Text (incorrect)
9	6000
80	700
700	80
6000	9

If your numbers or dates are not sorting correctly, make sure they are stored using the correct data type. See “[Numeric Data](#)” on page 255 for more information on the numeric data type. See “[Dates](#)” on page 260 for more information on the date data type.

### Sorting Right Justified Text

If your text is right justified, it will sort like a numeric field. In other words, **2** will sort before **10**, and **B** will sort before **AA**. The actual sorting rules for right justified text are—1) short data sorts before longer data (**B** before **AA**) and 2) if two data items are the same length, they will be sorted in alphabetical order (**AA** before **BA**).

### Sorting Selected Data

Sorting is not affected by the **Find/Select** command. The sort commands always sort the entire database—not just the selected records. When the invisible data is selected again you will see that it is sorted properly within the rest of the data.

### Sorting Within Groups

Later you’ll learn how Panorama can organize a database into groups, with summaries for each group (see Chapter 10). If you attempt to sort your database after it has been grouped, Panorama will automatically sort the data within the groups instead of sorting the entire database. If you want to sort the entire database you must remove the groups with the **Remove Summaries** command (Sort Menu).



# Chapter 9: Searching and Selecting



A Panorama database may contain dozens, hundreds, or even thousands of records. Finding a particular piece of information could be like locating a needle in a haystack. Fortunately Panorama can easily locate information for you.

## Finding vs. Selecting

Panorama has two ways of locating information, finding and selecting. Finding is much like looking up a name in a phone book—Panorama points out the location of the information you are looking for. For example, you might ask Panorama to find a phone number or a price in a catalog. Panorama will locate the information, and position the database to that spot. In this example we've asked Panorama to find [Blue Cross](#).

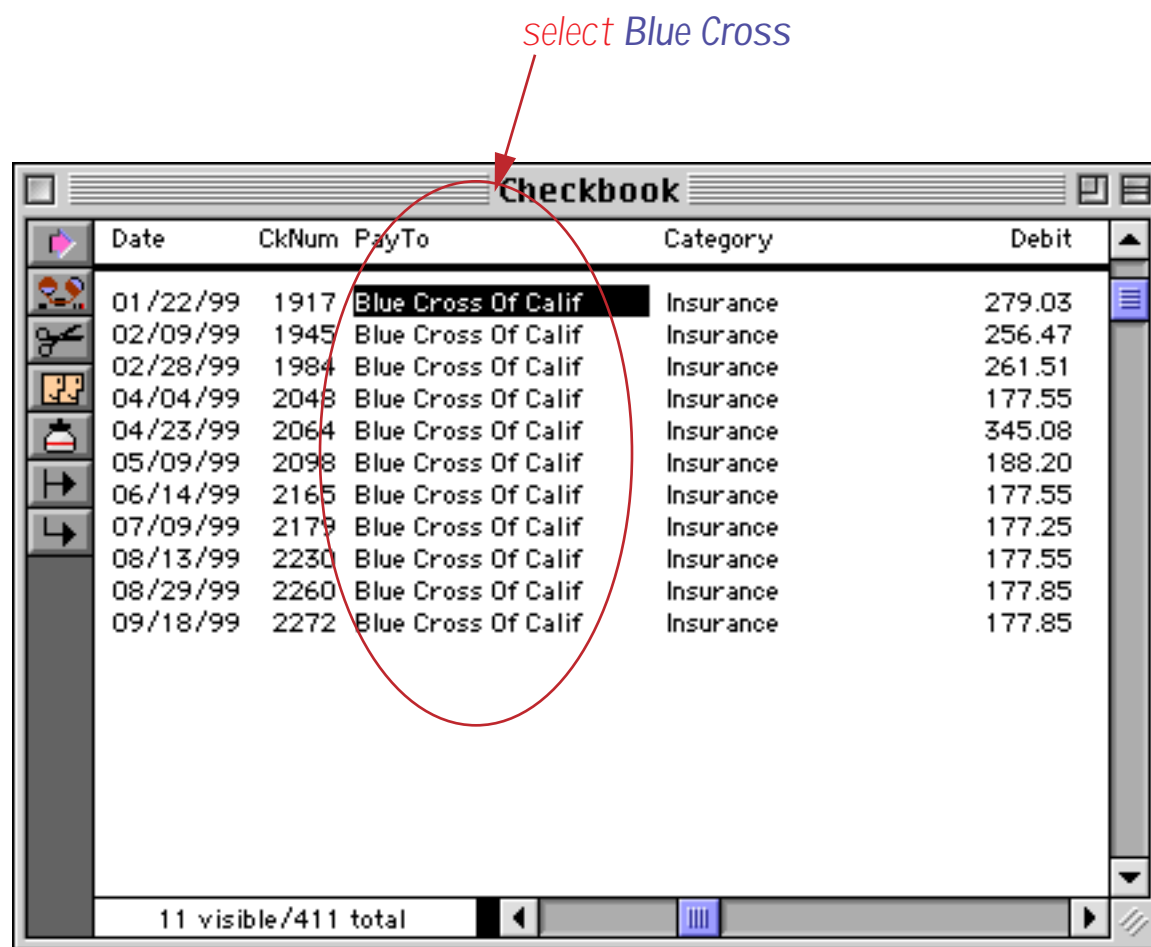
*find Blue Cross*

Date	CkNum	PayTo	Category	Debit
01/01/99		OPENING BALANCE		
01/08/99	1907	Northern Illinois Mold	Equipment Rental	96.05
01/08/99	1908	U S Postmaster	Postage	75.00
01/08/99	1909	Advertiser's Mailing Ser	Advertising	390.80
01/16/99	1910	Coudert Brothers, Attor	Legal Fees	223.52
01/16/99	1911	Paramount Stationers	Office Supplies	105.84
01/17/99	1912	California Capitol	Insurance	36.00
01/17/99	1913	California Capitol	Insurance	28.00
01/17/99	1914	U S Postmaster	Postage	75.00
01/17/99	1915	Sacramento Bee	Advertising	795.00
01/18/99		DEPOSIT		
01/22/99	1916	Walthers	Purchases	12,463.00
01/22/99	1917	Blue Cross Of Calif	Insurance	279.03
01/22/99	1918	Sherman Douglas Ins	Insurance	418.60
01/22/99	1919	Cannon Astro	Office Supplies	145.72
01/25/99	1920	Walthers	Purchases	1,885.40
01/25/99	1921	Nebs	Office Supplies	77.27
01/25/99	1922	Ramona Drinking Water	Office Supplies	98.10
01/25/99	1923	Pacific Partners	Rent	4,070.83

411 visible/411 total



Selecting is like creating a whole new phone book containing only the information you are looking for. All the selected data remains visible, while everything else temporarily vanishes. For example, you might ask Panorama to select all customers that have purchased from you in the last six months, or all transactions over \$250,000.



Deciding whether to find or select is your choice. Usually find is used when you want to locate a specific item like an address or price, while select is used when you want to locate a set of information. You can also combine the two techniques—for example, first select a subset of the database, then find a specific item within that subset.

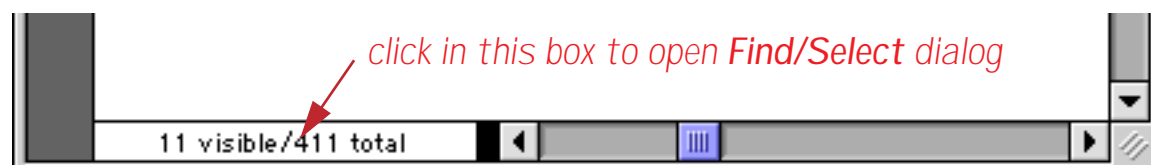
Note: Some other database programs have a “Find” command that actually does the same thing as Panorama’s Select command (for example FileMaker). These programs do not have a true find capability like Panorama.

### The Find/Select Dialog

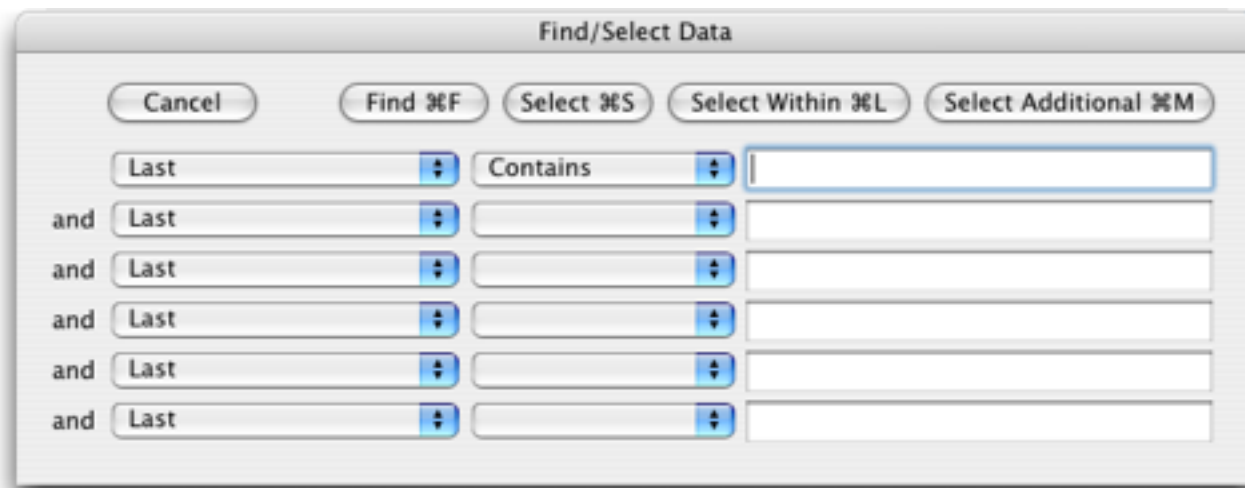
Panorama locates information by scanning through the entire database looking for data that matches your criteria. The Find/Select dialog (Search Menu) allows you to specify three criteria for locating information—the field containing the data, the kind of match you want (exact match, greater than, etc.) and the match value (the data you are looking for).



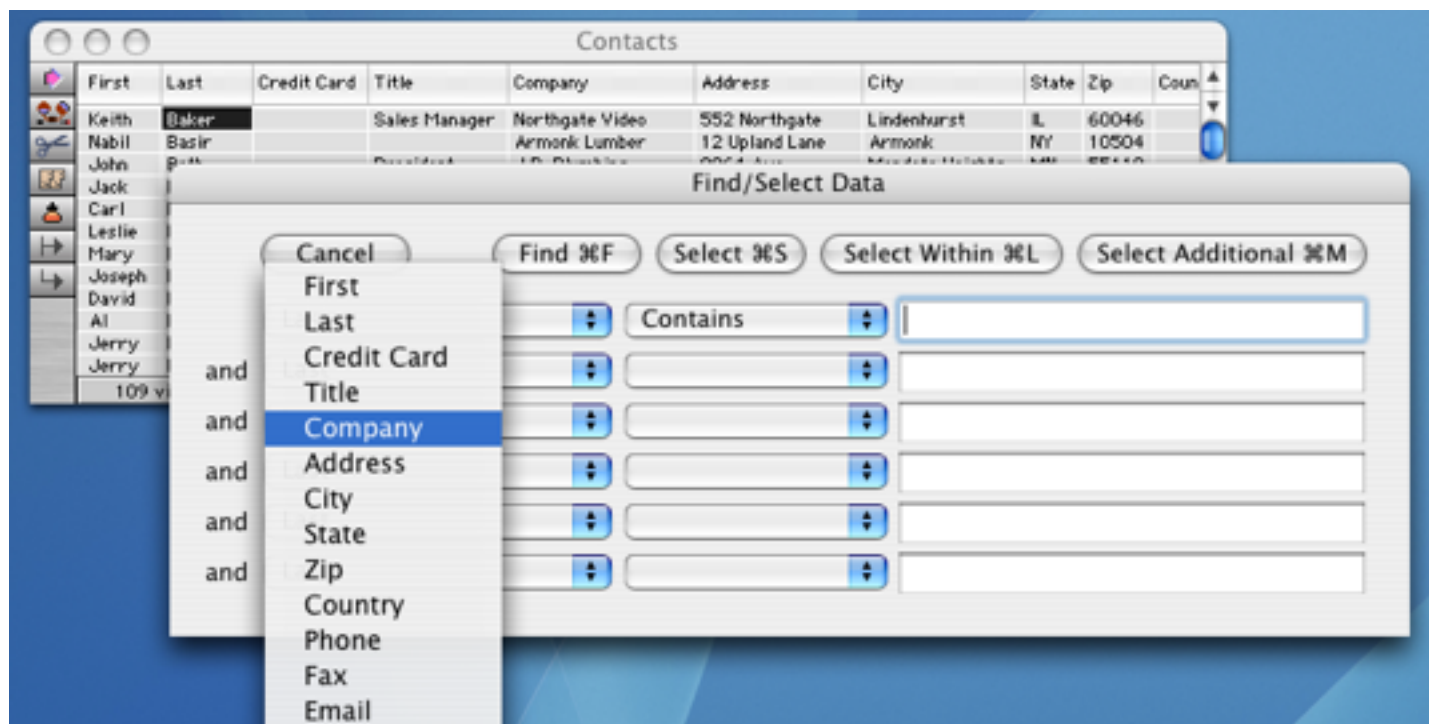
Tip: You can also open the Find/Select dialog by clicking on the record count displayed in the lower left hand corner of the window.



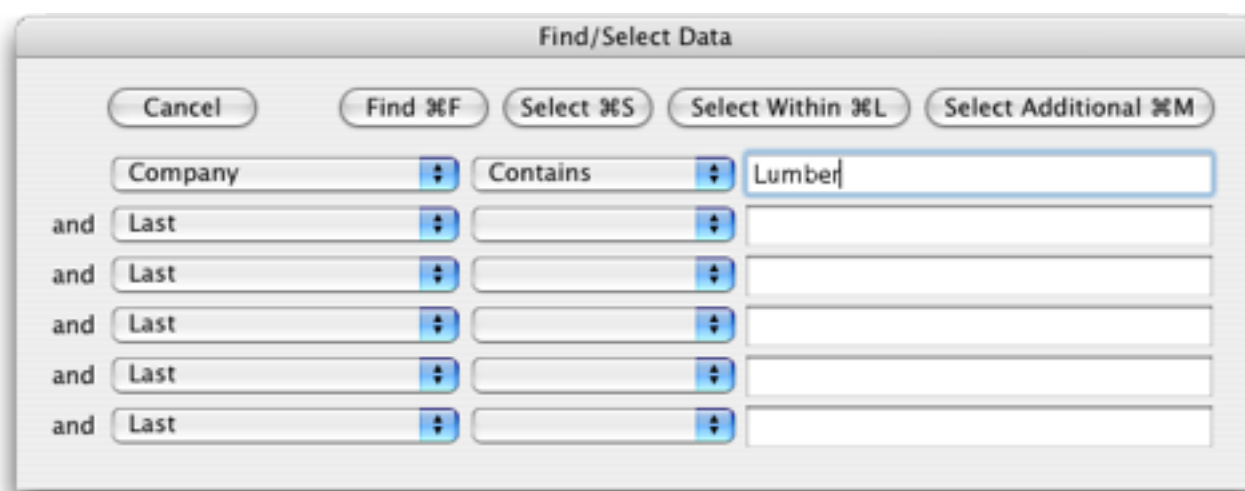
Here is what the Find/Select dialog looks like when you first open it.



The pop-up menus on the left side of the dialog tells Panorama what fields contains the data you are looking for. For example, if you are looking for a company, you would tell Panorama to look in the [Company](#) field. The Find/Select dialog initially assumes that you want to locate information in the current column.



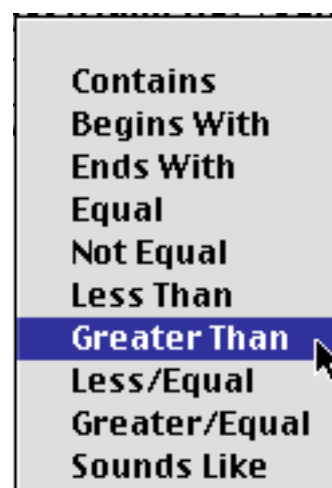
Type the data you are looking for into the box on the right side of the dialog. This data is called the match value. For example if you are looking for a person, that person's name would be the match value. If you are looking for transactions over \$250,000 the match value would be [250000](#).



If you press the **Select** button (or the **Enter** key) Panorama will select all records with a company name containing **Lumber**.



The pop-up menu in the middle of the dialog controls how Panorama looks for a match. There are ten different choices for identifying a match.



**Contains** — Any data cell that contains the match value will be identified as a match. For example, if you ask Panorama to locate cities containing an, it will locate cities like Anaheim, Lansing, Los Angeles, and San Jose since they all contain an. Notice that capitalization is ignored, so an, An, AN, and aN are all acceptable matches.

**Not Contains** — Any data cell that does not contain the match value will be identified as a match. For example, if you ask Panorama to locate phone numbers not containing (714) it will locate phone numbers in other area codes. Capitalization is ignored, so an, An, AN, and aN are all equivalent as far as this option is concerned.

**Begins With** — Any data cell that begins with the match value will be identified as a match. For example, if you ask Panorama to locate states beginning with co, it will locate states like Colorado and Connecticut. Capitalization is ignored, so co, Co, CO, and cO are all acceptable matches.

**Ends With** — Any data cell that ends with the match value will be identified as a match. For example, if you ask Panorama to locate baseball teams ending with sox it will locate both Red Sox and White Sox. Capitalization is ignored, so sox, Sox, SOX, and sOx are all acceptable matches.

**Equal (=)** — Any data cell that exactly matches the match value will be identified as a match. An exact match means just that. The spelling, punctuation, and capitalization must be exactly the same—for example red will not match RED or Red.

**Not Equal** ( $\neq$ ) — Any data cell that does not exactly match the match value will be identified as a match.

**Less Than** ( $<$ ) — Any data cell that is less than the value in the box on the right will be identified as a match.

**Greater Than** ( $>$ ) — Any data cell that is greater than the match value will be identified as a match.

**Less Than or Equal** ( $\leq$ ) — Any data cell that is less than or equal to the match value will be identified as a match.

**Greater Than or Equal** ( $\geq$ ) — Any data cell that is greater than or equal to the match value will be identified as a match.

**Sounds Like** — Any data cell that “sounds like” the match value will be identified as a match. Panorama uses a special algorithm to determine which values sound like the match value. This algorithm is not perfect, but it does work pretty well. For example, if you are looking for someone named Luboviski but you are not sure if it is spelled with an i, ie, or y, the sounds like match will save the day.

The sounds like match can be used with more than one word at a time. For example, if you are searching through a video rental database for the movie [Escape from New York](#), the sounds like algorithm will find it even if it is misspelled [Escapade from New York](#). If any word in the match value sounds like any word in the data cell, the data will be identified as a match.

Note: If two words do not start with the same letter, the sounds like algorithm will not think they sound alike. For example, sounds like does not think that [Chris](#) and [Kris](#) sound alike.

**Match** and **MatchExact** — These options allow you to create a “wildcard” pattern for comparing data. See Chapter 9 of the [Panorama Handbook](#) for details.

### Locating Dates by Month, Quarter, or Year

The **Find/Select** dialog allows you to specify dates by day, month, quarter, or year. These options only work if the dates are stored using the date data type, not text (see “[Dates](#)” on page 121).

To specify an individual day, simply type the date in the format [mm/dd/yy](#) or [mm/dd](#). If you leave off the year, Panorama will round the date to the nearest year.

A screenshot of the Find/Select dialog box. It features a dropdown menu on the left with a small icon, currently set to 'Date'. To its right is a text input field containing 'Date'. Further right is another dropdown menu set to 'Equal'. The final part of the dialog is a large text input field containing the date '3/17/00'.

To specify a month, you must enter the month in the format [MONTH YY](#) or [MON YY](#) (for example [September 1998](#) or [Aug 01](#)). You cannot specify a month using the format [mm/yy](#), because Panorama cannot distinguish this from [mm/dd](#).

A screenshot of the Find/Select dialog box. It features a dropdown menu on the left with a small icon, currently set to 'Date'. To its right is a text input field containing 'Date'. Further right is another dropdown menu set to 'Equal'. The final part of the dialog is a large text input field containing the text 'Sep 2004'.

To specify a quarter, use the format [QqYY](#) (for example [1q99](#) or [3Q2006](#)).

A screenshot of the Find/Select dialog box. It features a dropdown menu on the left with a small icon, currently set to 'Date'. To its right is a text input field containing 'Date'. Further right is another dropdown menu set to 'Equal'. The final part of the dialog is a large text input field containing the text '3q2006'.

To specify an entire year, use the format [YY](#) or [YYYY](#) (for example [90](#) or [1994](#)).

A screenshot of the Find/Select dialog box. It features a dropdown menu on the left with a small icon, currently set to 'Date'. To its right is a text input field containing 'Date'. Further right is another dropdown menu set to 'Equal'. The final part of the dialog is a large text input field containing the year '2001'.

If you specify a month, quarter, or year, you must use the **Equals** (=) type of match.

## Find and Find Next

Once you have specified what you are looking for, you can either find or select the information (see “[Finding vs. Selecting](#)” on page 159). Press the **Find** button to find the information.

When you press the **Find** button, Panorama will go to the top of the database and start scanning. When Panorama finds a data cell that matches what you are looking for, it stops scanning and displays the information it has found.

To resume scanning for additional matches, use the **Find Next** command. If there aren't any more matches, Panorama will beep.

*find Office Supplies*

Date	CkNum	PayTo	Category	Debit
01/01/99		OPENING BALANCE		
01/08/99	1907	Northern Illinois Mold	Equipment Rental	96.05
01/08/99	1908	U S Postmaster	Postage	75.00
01/08/99	1909	Advertiser's Mailing Ser	Advertising	390.80
01/16/99	1910	Coudert Brothers, Attor	Legal Fees	223.52
01/16/99	1911	Paramount Stationers	Office Supplies	105.84
01/17/99	1912	California Capitol	Insurance	36.00
01/17/99	1913	California Capitol	Insurance	28.00
01/17/99	1914	U S Postmaster	Postage	75.00
01/17/99	1915	Sacramento Bee	Advertising	795.00
01/18/99		DEPOSIT		
01/22/99	1916	Walthers	Purchases	12,463.00
01/22/99	1917	Blue Cross Of Calif	Insurance	279.03
01/22/99	1918	Sherman Douglas Ins	Insurance	418.60
01/22/99	1919	Cannon Astro	Office Supplies	145.72
01/25/99	1920	Walthers	Purchases	1,885.40
01/25/99	1921	Nebs	Office Supplies	77.27
01/25/99	1922	Ramona Drinking Water	Office Supplies	98.10
01/25/99	1923	Pacific Partners	Rent	4,070.83

*find next*

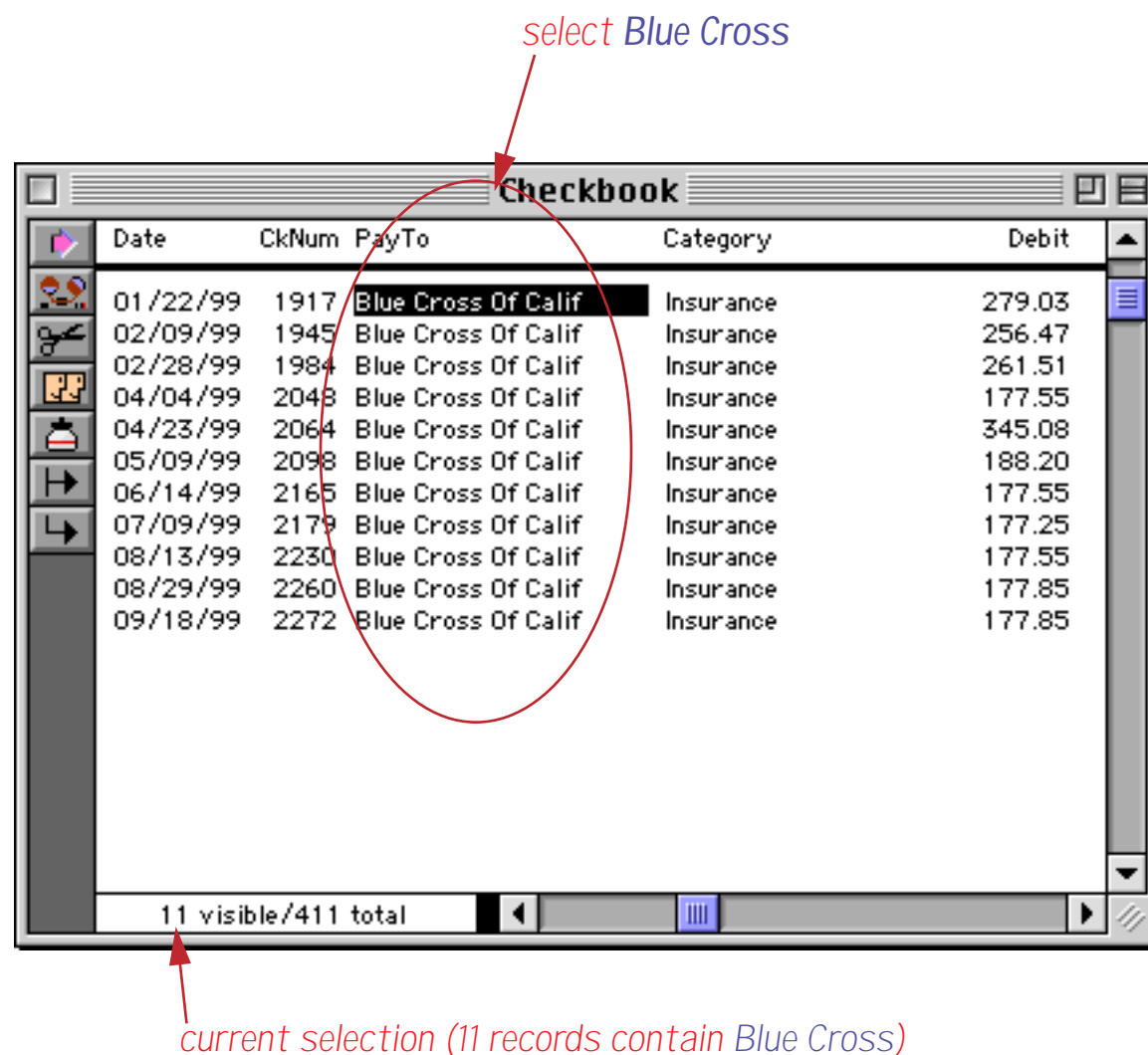
411 visible/411 total

You can use the **Undo** command to move backwards through the items you have already found. You are not limited to just one **Undo**. You can keep undoing until you reach the first matching item in the file.



## Select

To select a subset of the database, press the **Select** button. Panorama will scan through the entire database and select the records that match the criteria you have specified. The selected records remain visible, while the records that do not match temporarily vanish. Panorama displays the number of selected records in the lower left hand corner of the window.



To restore the invisible records, choose the **Select All** command from the Search Menu.

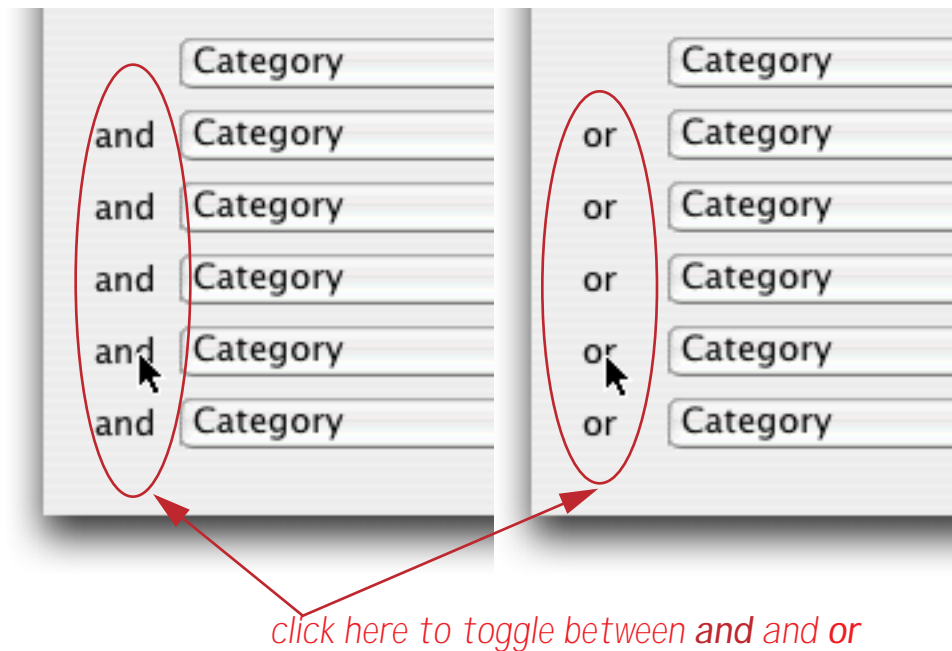
To make a different selection, simply use **Find/Select** again. The original selection will vanish and the new selection will become visible. (You do not need to choose **Select All** before selecting another subset.)

Note: If your database is large and you have selected only a few records, you may find that Panorama seems sluggish. Remember, Panorama may be skipping over hundreds of invisible records that are between the visible records on the screen. When you use **Select All**, Panorama's normal blazing speed will return.

Note: After you have selected a subset of the data, you may find that you cannot move the data sheet scroll bar to the very top or very bottom. This will happen if the first or last record is not one of the selected (visible) records.

## Multiple Find/Select Criteria

The Find/Select dialog can be expanded to allow up to six concurrent match criteria. For example, you can select all names in **Oregon** or **Idaho** or **Montana**, or you could select all transactions that are greater than **\$10,000** and less than **\$25,000**. (However you cannot combine both **and** and **or** at the same time with the Find/Select dialog, to do that you must use the **Formula Find/Select** dialog — see “**Formula Find/Select**” on page 181). The six separate match criteria can be combined with **and** or **or**. To toggle between **and** and **or**, click on the word **and** or **or** on the left edge of the dialog.



When the criteria are combined with **and**, everything must match. For example, the dialog shown below could be used to select all presidents of lumber companies within California.

Find/Select Data

Cancel Find ⌘F Select ⌘S Select Within ⌘L Select Additional ⌘M

	Title	Equal	President
and	Company	Contains	Lumber
and	State	Equal	CA
and	Last		

When the criteria are combined with **or**, Panorama will declare a winner if anything matches. For example, the dialog shown below tells Panorama to locate anyone with the words President, Manager, or Agent in his or her title.

Find/Select Data

Cancel Find ⌘F Select ⌘S Select Within ⌘L Select Additional ⌘M

	Title	Contains	President
and	Title	Contains	Manager
and	Title	Contains	Agent
and	Title		

The **Find/Select** dialog does not let you mix **and** and **or** within a single selection. See “[Formula Find/Select](#)” on page 181 if you need to mix **and** and **or** within a single selection.

### Select Within

The **Select Within** button allows you to select a subset of the currently selected records (The normal **Select** button selects a subset of the entire database.) For example, if you have already selected transactions over \$5,000, you can use **Select Within** to select only those transactions in California. The result would be the subset containing all California transactions over \$5,000.

Of course, you could obtain the same effect by combining multiple criteria (**and**) in the first place (see the previous section). But you don't always know in advance exactly what you are looking for. The **Select Within** button allows you to whittle your data down gradually until you've extracted just the nugget of information you really need.

### Select Additional

The **Select Additional** button allows you to select a superset of the currently selected records. For example, if you have already selected names in Ohio, you could use **Select Additional** to also select names in Illinois. The result would be the subset containing all names in either Ohio or Illinois.

Of course you could obtain the same effect by combining multiple criteria (**or**) in the first place (see above). But you don't always know in advance exactly what you are looking for. The **Select Additional** button lets you assemble the pieces of information you need piece by piece.

### Select Reverse

The **Select Reverse** command reverses selected and deselected records. For instance, if you have selected all transactions greater than \$600, the **Select Reverse** command will select all transactions less than or equal to \$600.

### Undo Select

The **Undo** command can reverse the effects of the last 16 select operations, including **Select**, **Select Within**, **Select Additional**, and **Select Reverse**. As long as you do not use any other commands or tools, you can use the **Undo** command up to 16 times in a row.

The quickest way to select the entire database is the **Select All** command.

### Permanently Removing Unselected Data

Unselected data is hidden, but it is still part of the database. You can restore the invisible data with the **Select All** command. Sometimes, however, you may wish to free the memory occupied by the hidden records. The **Remove Unselected** command in the Search Menu permanently removes the unselected records from the database.

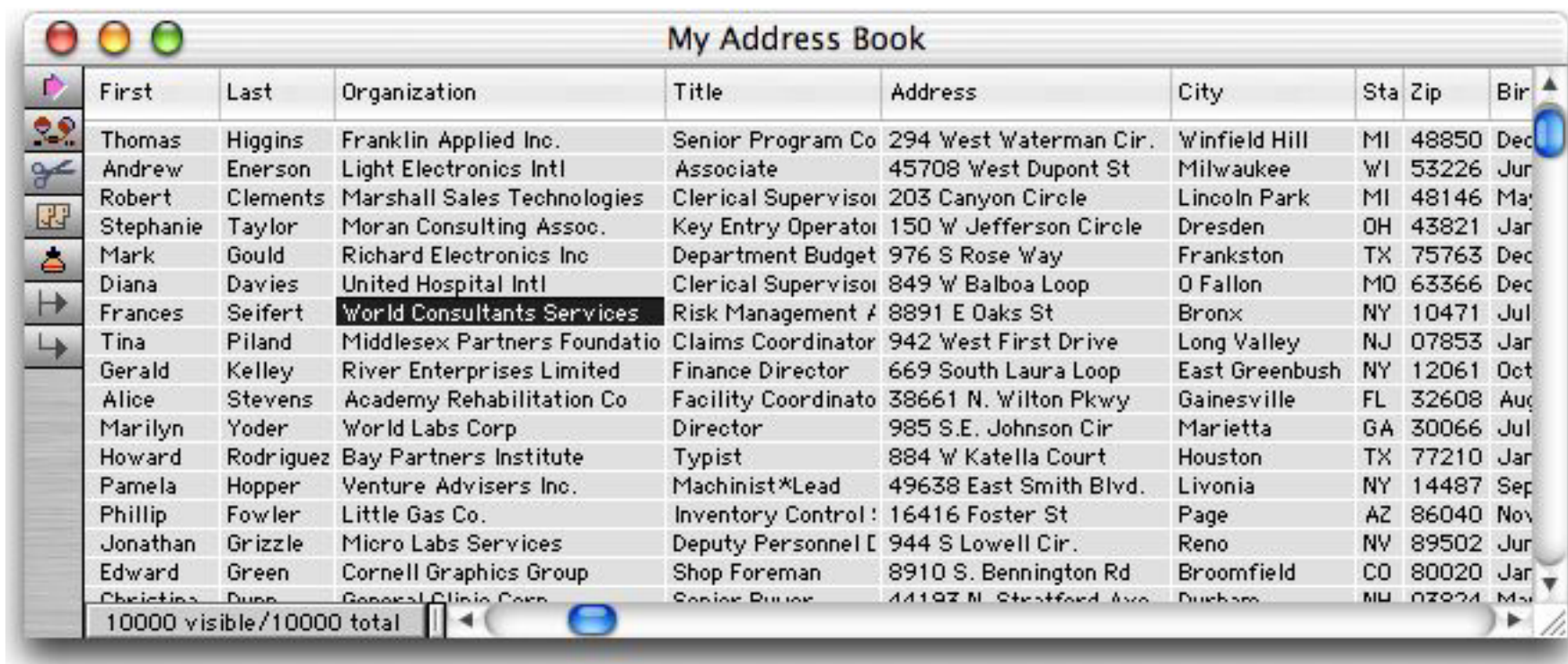
Before **Remove Unselected** actually erases the unselected data, it asks you to confirm that you really want to proceed. Panorama doesn't want you to accidentally delete hundreds or thousands of records. Be careful, because the **Remove Unselected** command cannot be reversed with the **Undo** command. However, if you have saved a copy of the data on disk prior to using **Remove Unselected**, you can still recover the data with the **Revert to Saved** command (until you save again).

## The Live Clairvoyance™ Wizard

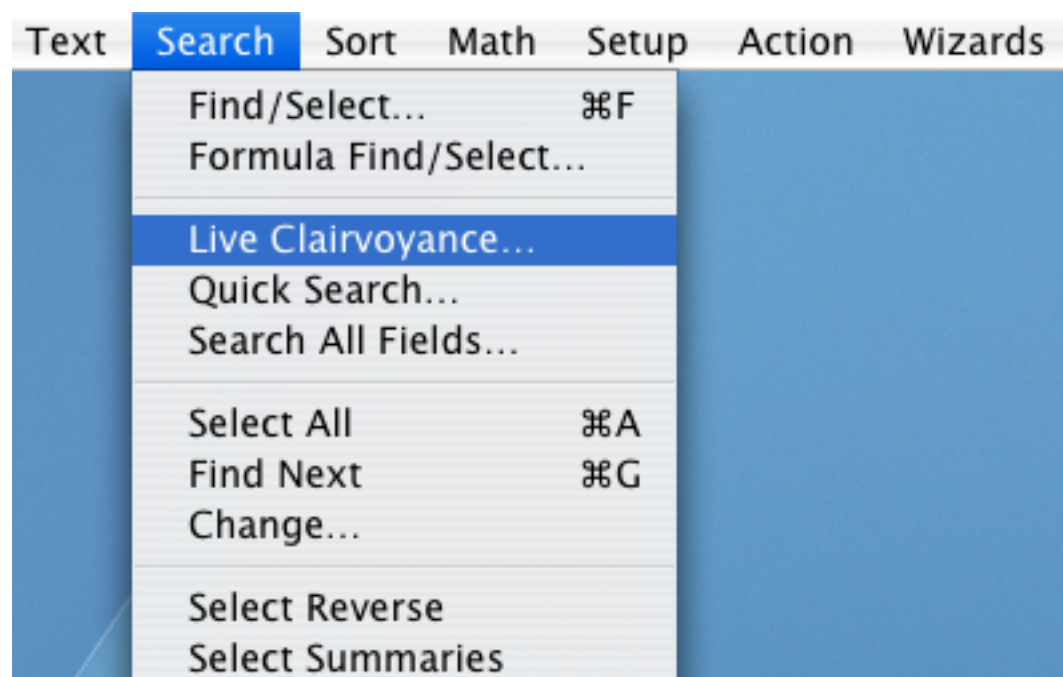
The Live Clairvoyance™ wizard allows you to perform “live” searches on any Panorama database. The search results are updated dynamically as you type, allowing you to “hone in” on just the information you are looking for. The search may include multiple fields or even all fields in the database being searched, and you can also search based on a boolean formula. (If you've used the search box in iTunes you'll find the operation of this wizard familiar.) You can save your favorite searches for later re-use. Using the Live Clairvoyance wizard doesn't require you to do any programming or make any modifications to your existing databases.

### Basic Live Searching

The easiest way to understand the Live Clairvoyance is to see it in action. For example, suppose you wanted to search the database [My Address Book](#), which contains 10,000 names, address, phone numbers, birthdays, etc.

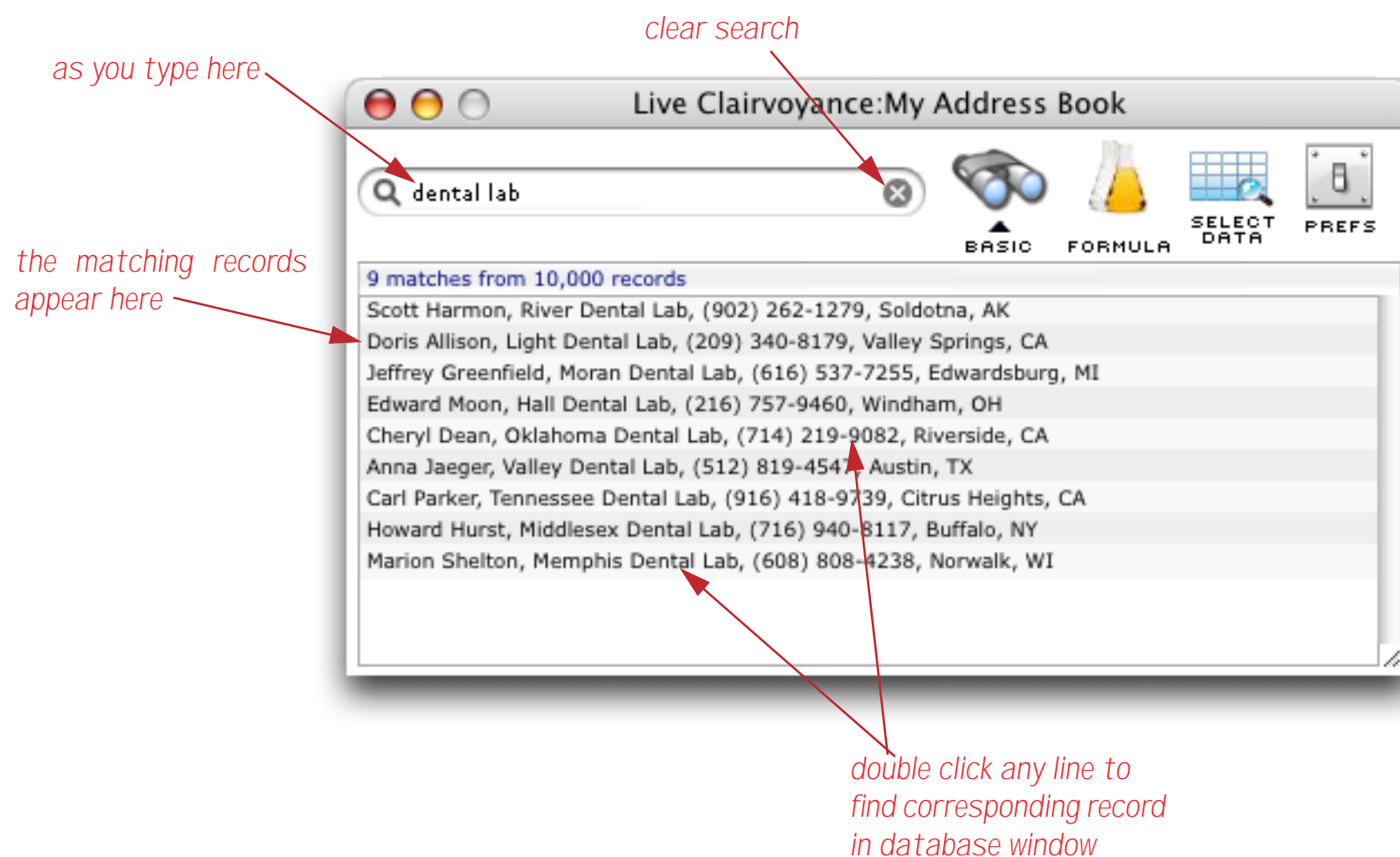



To search this database, choose Live Clairvoyance from the Search menu (or from the Search submenu of the Wizard menu). On MacOS systems you can also press **Control-F** to open this wizard (this key setting can be changed with the Hotkey Manager in the Preferences submenu of the Wizard menu).





The database that is active when you choose this command becomes the target database (the database that will be searched). Once the wizard is open simply type the word or phrase you want to search for into the search box at the top of the wizard window. As you type each character the list will update to show the records that match the word or phrase you have typed so far.



To clear the search and start over, press the  button.

To jump to any record you have found in the original database, double click on that record in the list. Panorama will bring the target database forward and position the current record to the corresponding location.

As it searches the database the wizard displays the status of the search, for example **9 matches from 10,000 records**. If there is a + symbol after the number of matches (for example **9+ matches from 10,000 records**) the wizard has not completed the search yet, and more matching records may be found. The + symbol will disappear when the entire database has been searched.

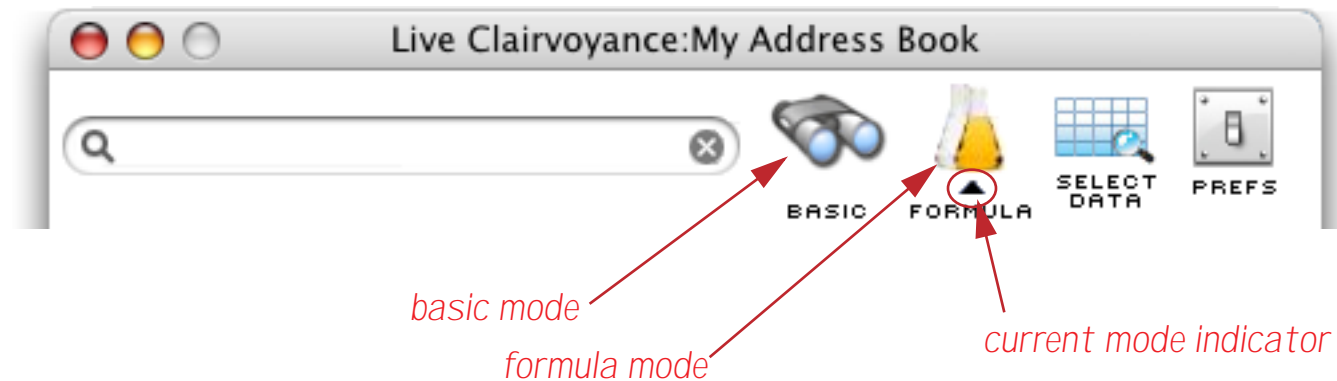
(Note: Prior to Panorama 5.5, this wizard would search only selected records in the target database. Now it searches all records.)



## Live Formula Searching

The Live Clairvoyance wizard has two modes: **basic** searching (described in the previous section) and **formula** searching. The formula search mode allows you to search for records based on a boolean formula. You can make this formula as simple or complex as you want, using parentheses, **and/or**, and any operator of function in Panorama's repertoire (see "[True/False Formulas](#)" on page 552 for a detailed explanation of **true-false logic**).

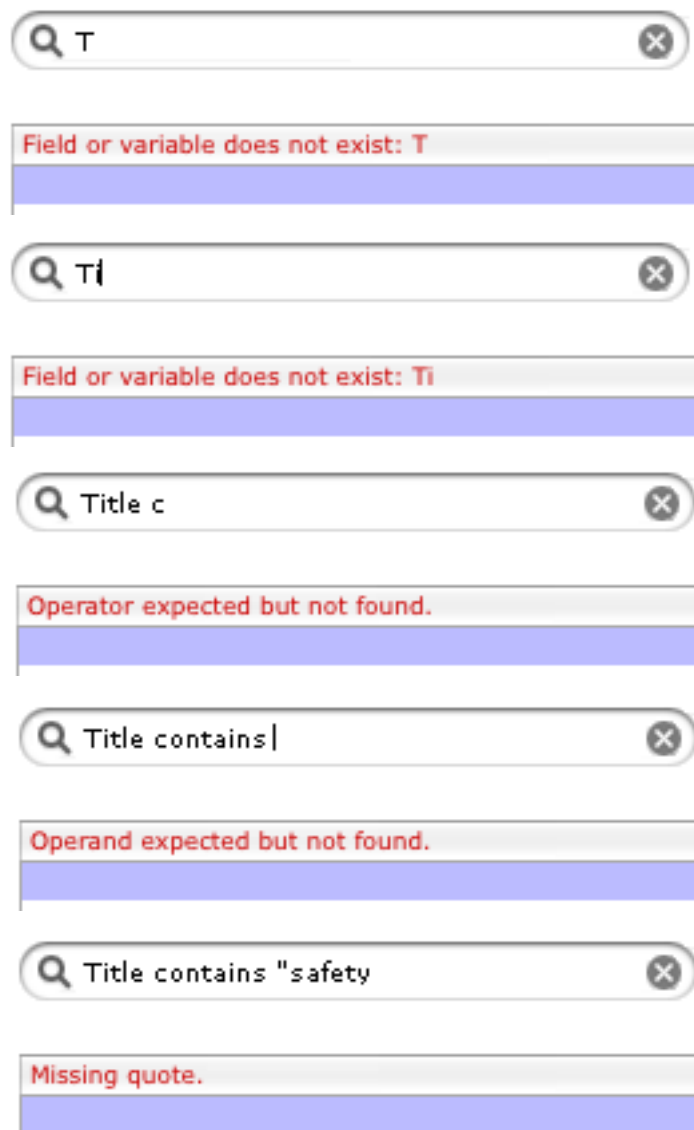
To switch between **basic** and **formula** mode click the appropriate icon in the toolbar. The small triangle shows which mode is currently selected.



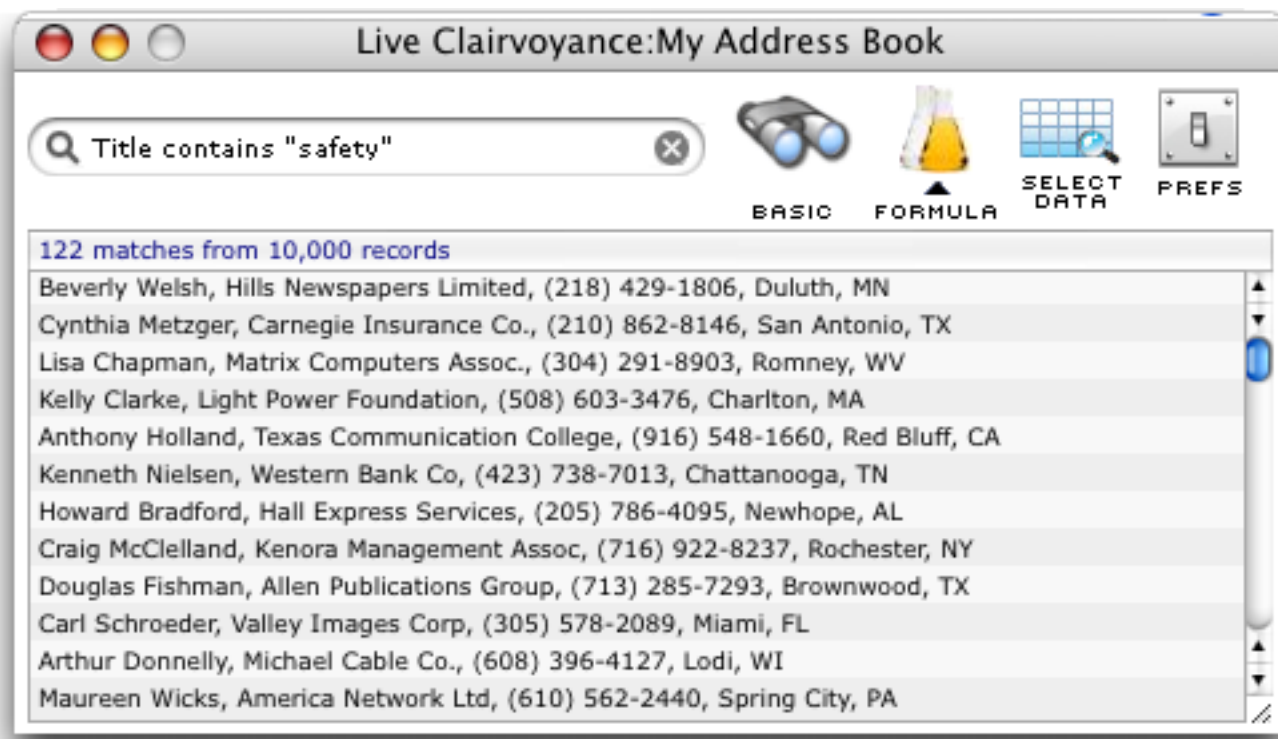
Once the formula mode is selected the next step is to type a valid formula into the wizard. Suppose you want to locate all safety related personnel in California from this database, you would use a formula like this:

```
Title contains "safety" and State="CA"
```

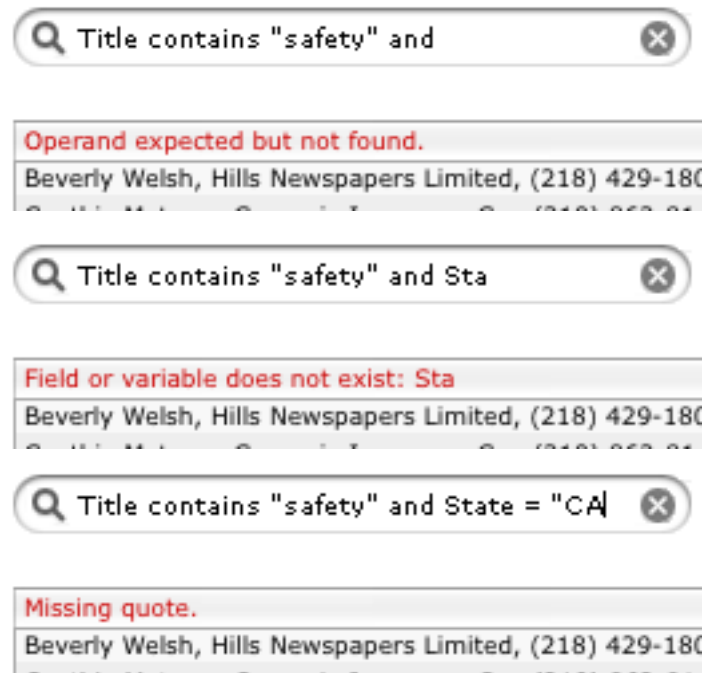
As you type each key the wizard will check to see if you have entered a valid formula. If the formula is not valid the search status will display the error message. You'll usually ignore this message as you type in the formula.



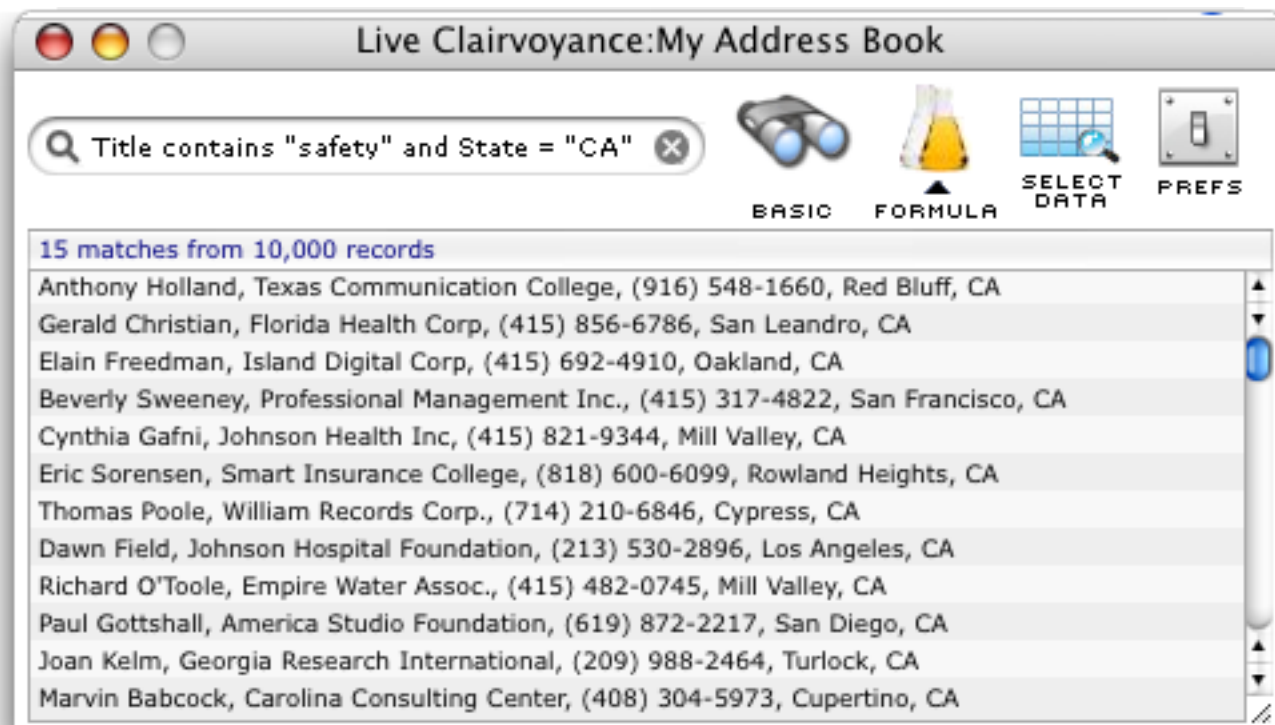
When you complete a valid formula the wizard will search the database for any matches. In this case there are 122 people in this database that have safety in their job title.



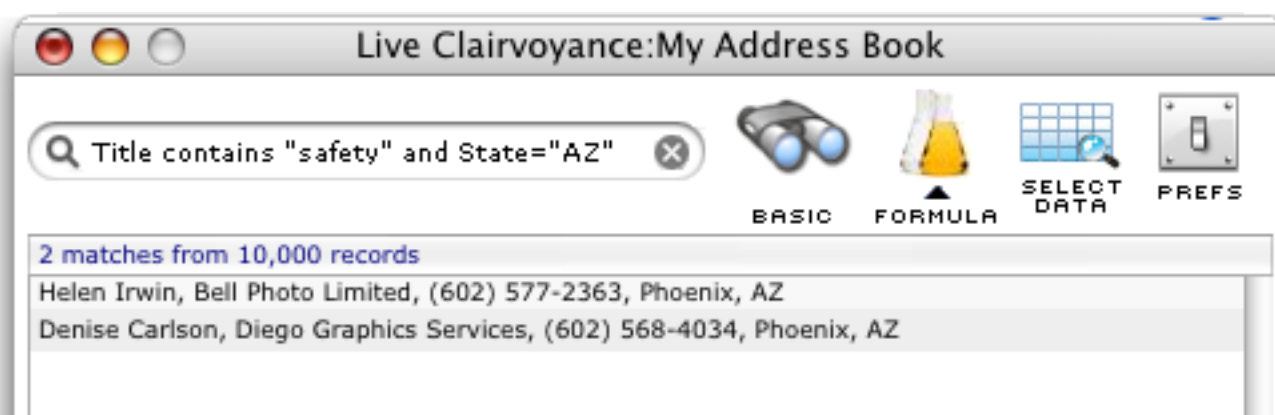
However we were only interested in safety personnel in California, so we need to type in the rest of the formula.



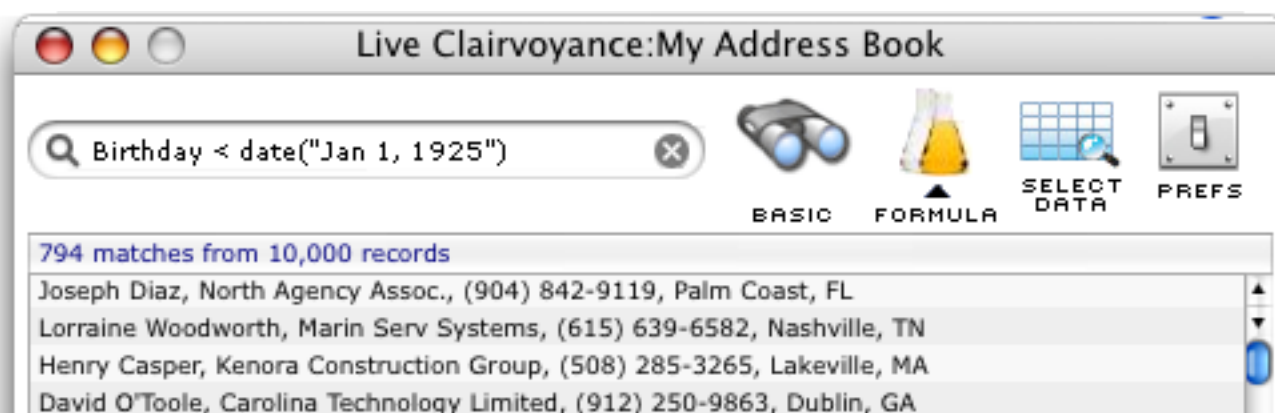
When the final quote key is pressed the search results appear.



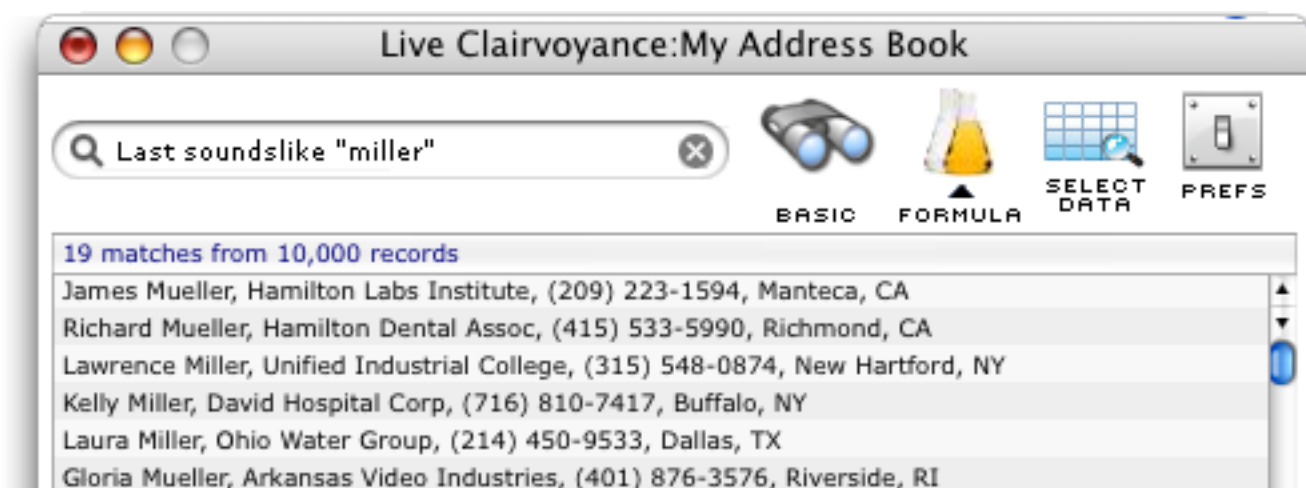
Once you've set the formula up you can easily tinker with it, for example changing **CA** to **AZ**. The search results will instantaneously update as you modify the formula.



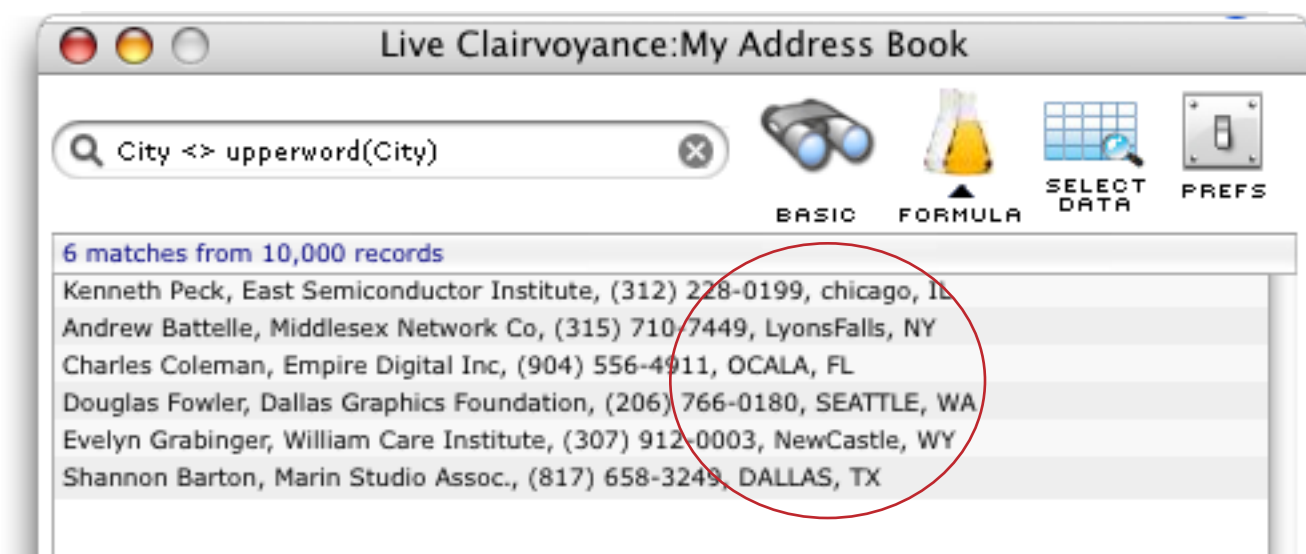
Panorama has a wide variety of operators and functions that can be used in a formula. You should be able to select data just about any way you want. For example here is a formula that selects everyone in the database born before 1925.



Here is a search that searches the database phonetically. Notice that searching phonetically for **Miller** will also find **Mueller** and other similar spellings.



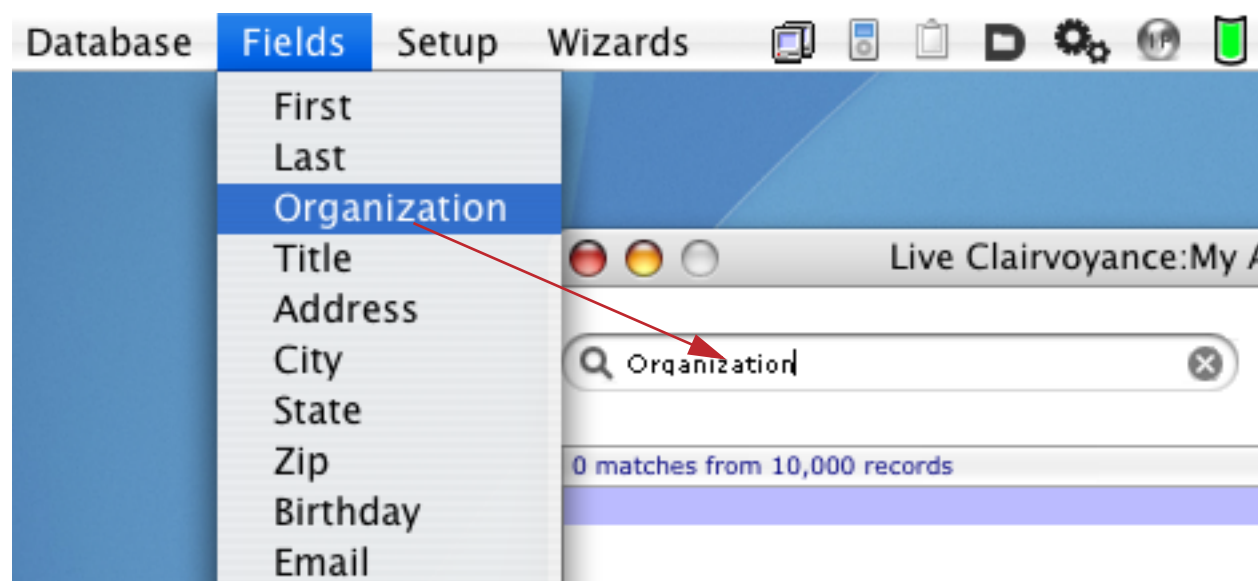
This search checks city names to make sure they are capitalized correctly (first letter upper case, rest lower case). It turns out there are six records with improper capitalization. Imagine how long it would take to find these manually!



By the way, if you don't have enough room for your formula you can expand the width of the wizard.

### The Fields Menu

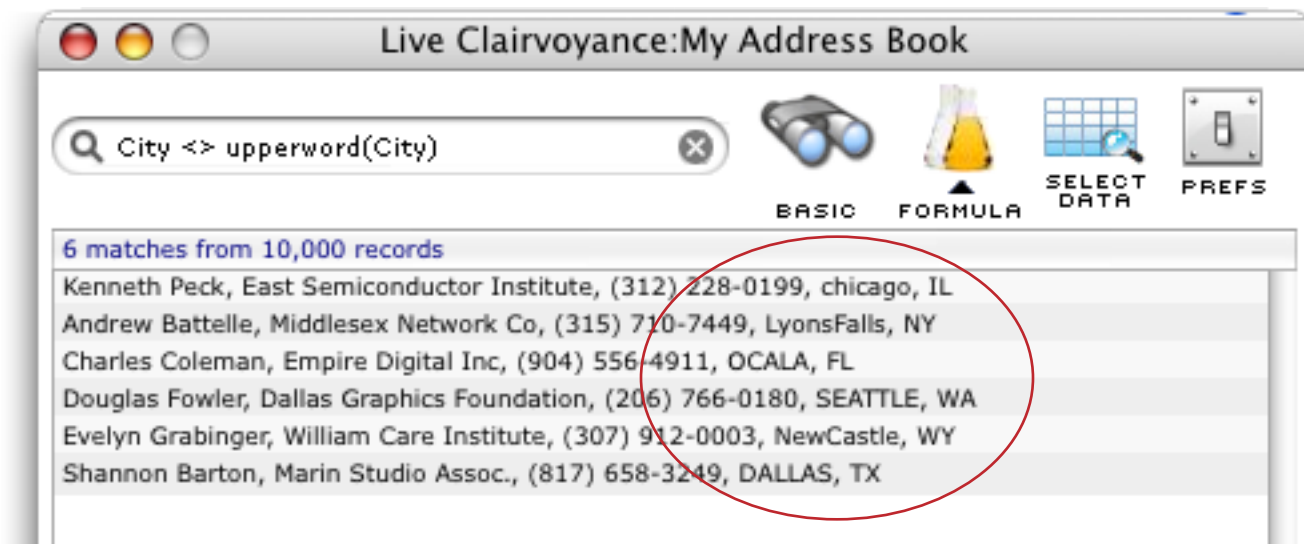
When the wizard is in formula mode an extra **Fields** menu appears in the menu bar. This menu lists all of the fields in the target database. Simply select a field from the menu and it will be typed into the formula for you.



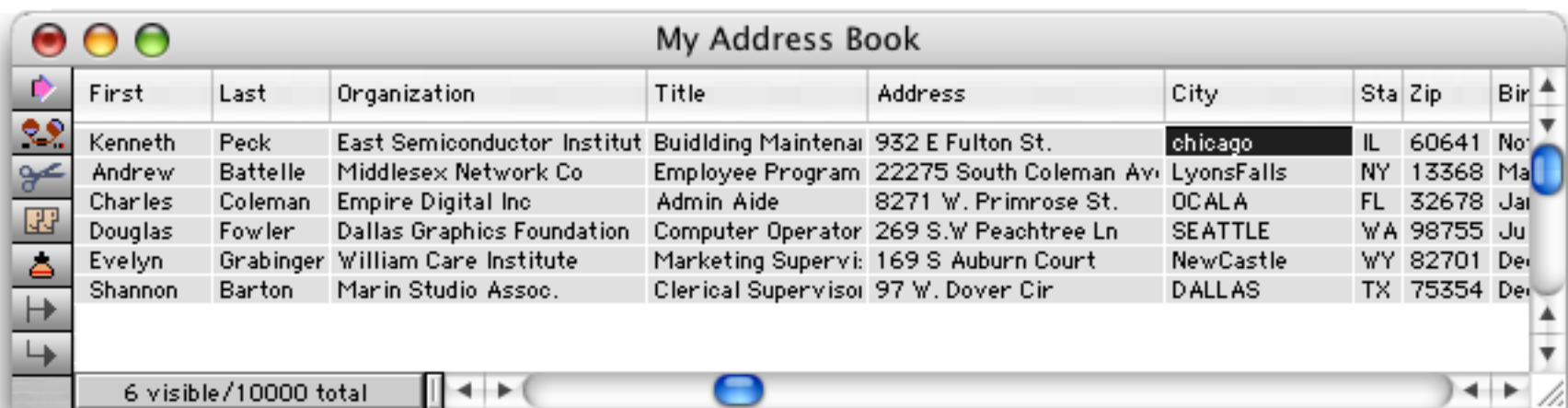
If the field has special characters or punctuation in it the wizard will automatically add « and » chevrons as needed (see "[Whitespace](#)" on page 516).

## Selecting Search Results in the Original Database

Once you've located a set of records in the Live Clairvoyance you may want to work with that set of records in the original database. For example consider this selection from the previous section (which selects all cities that are not capitalized correctly).




To see this data in the original database press the **SELECT DATA** tool.



Now you can work with this subset and make any changes that are necessary.



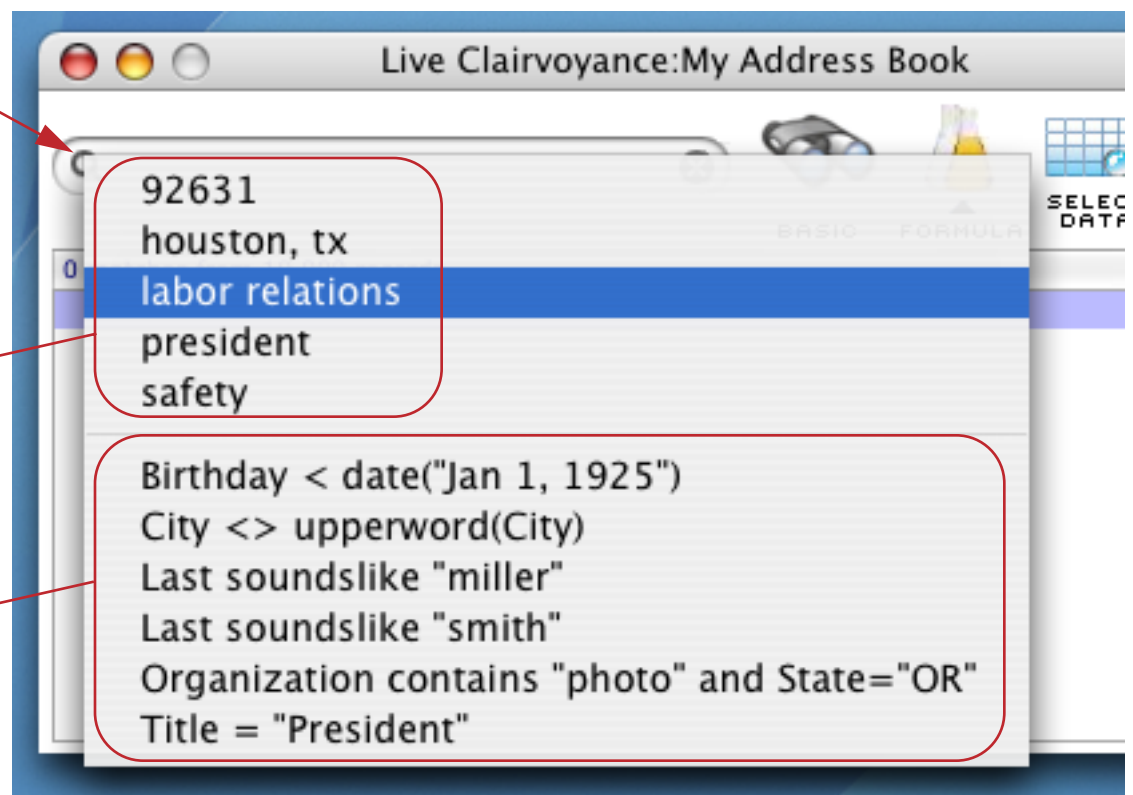
## The Favorite Searches Pop-up Menu

The wizard allows you to keep a pop-up menu of your favorite searches. Click the  icon to see the pop-up menu.

*click for pop-up favorite searches*

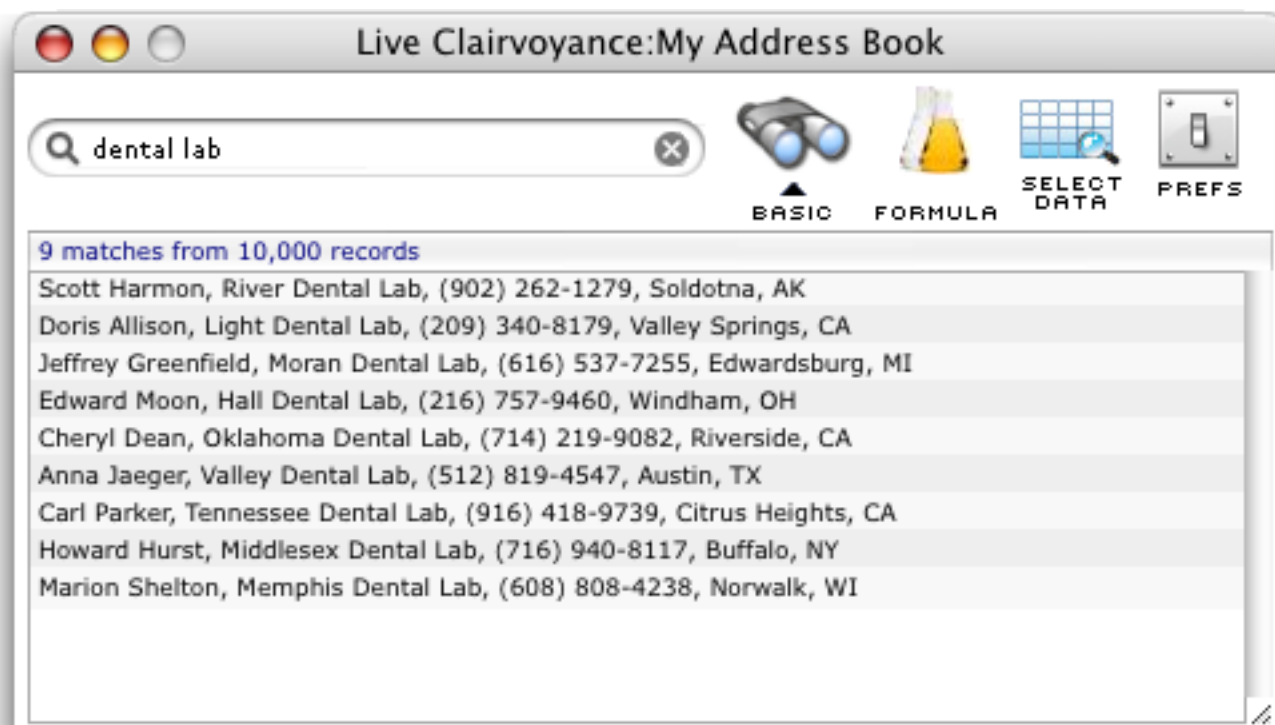
*basic searches*


*formula searches*

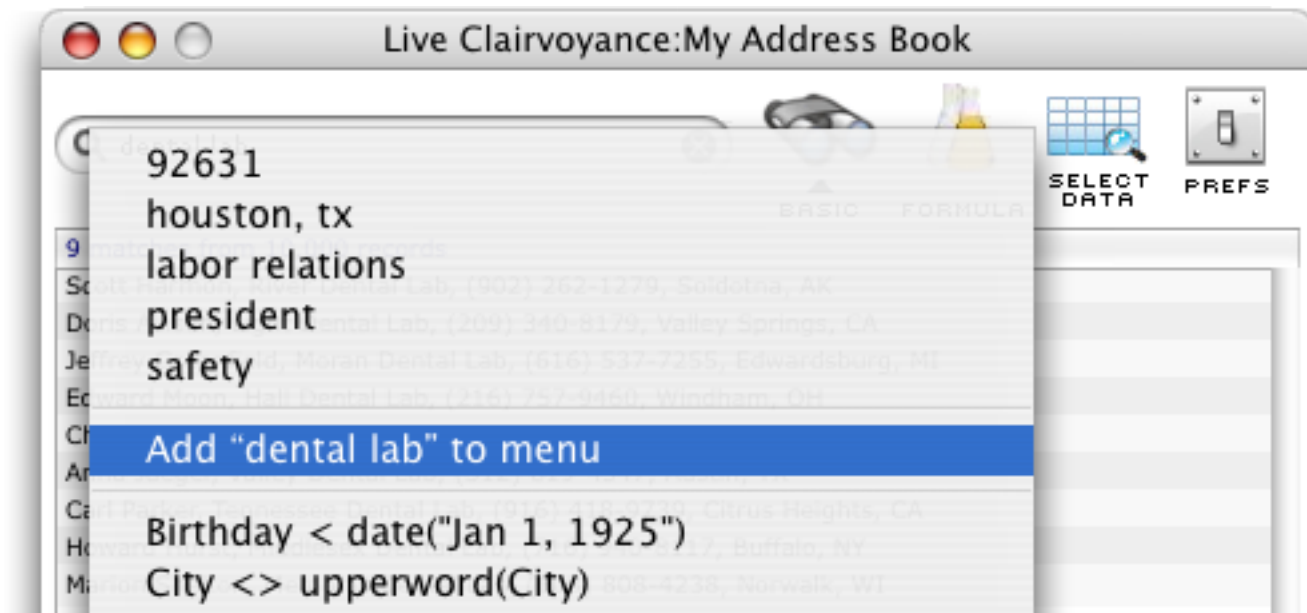


The top section of the menu lists basic searches you have saved, while the bottom section lists formula searches. To perform a search simply select it from the menu. (The wizard will automatically switch to the correct search mode.)

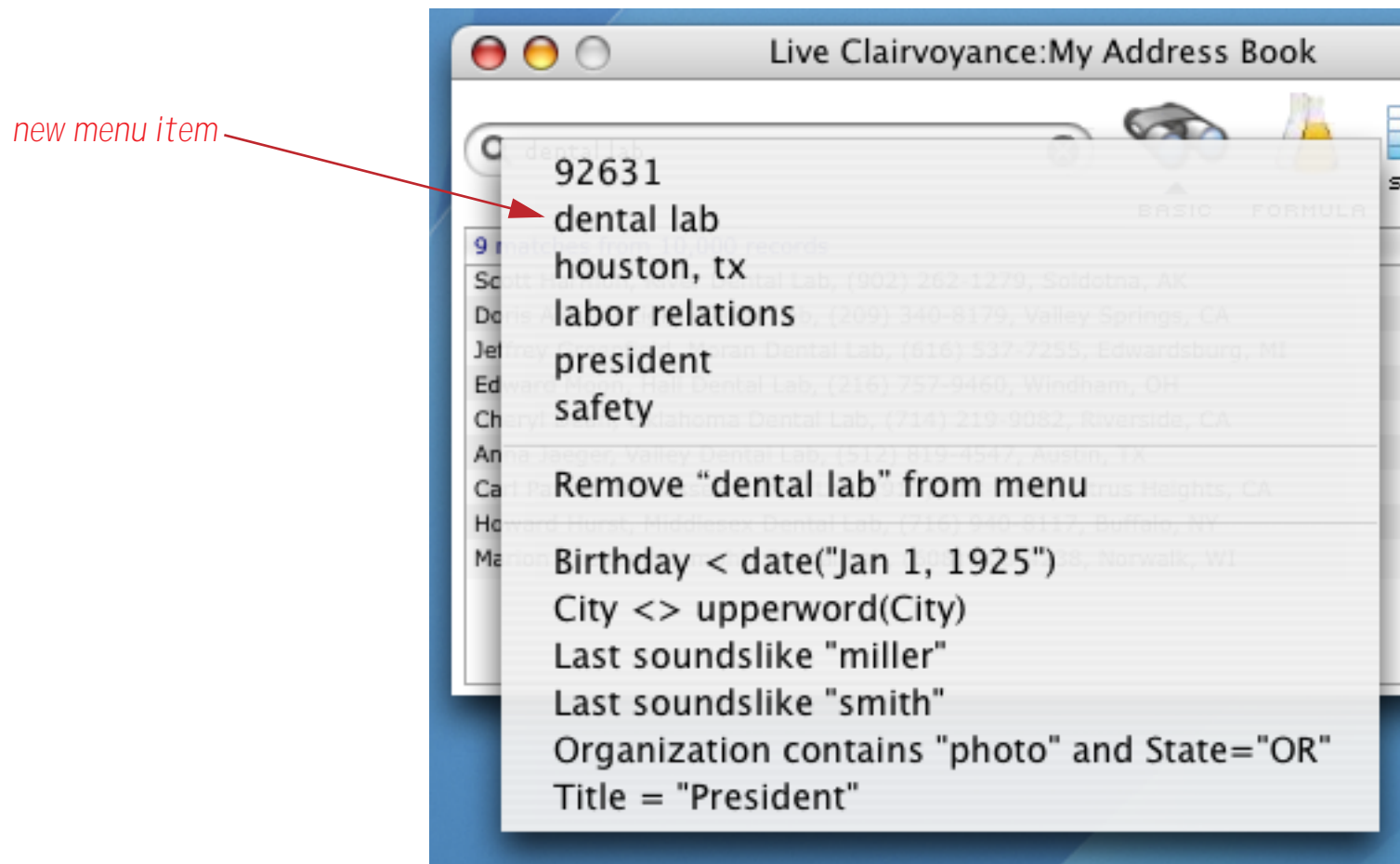
To add a search to the menu start by typing the search into the box at the top. You may use either a basic or a formula search.



Now click the  icon. You'll see a choice for adding your search to the menu, simply choose this option.



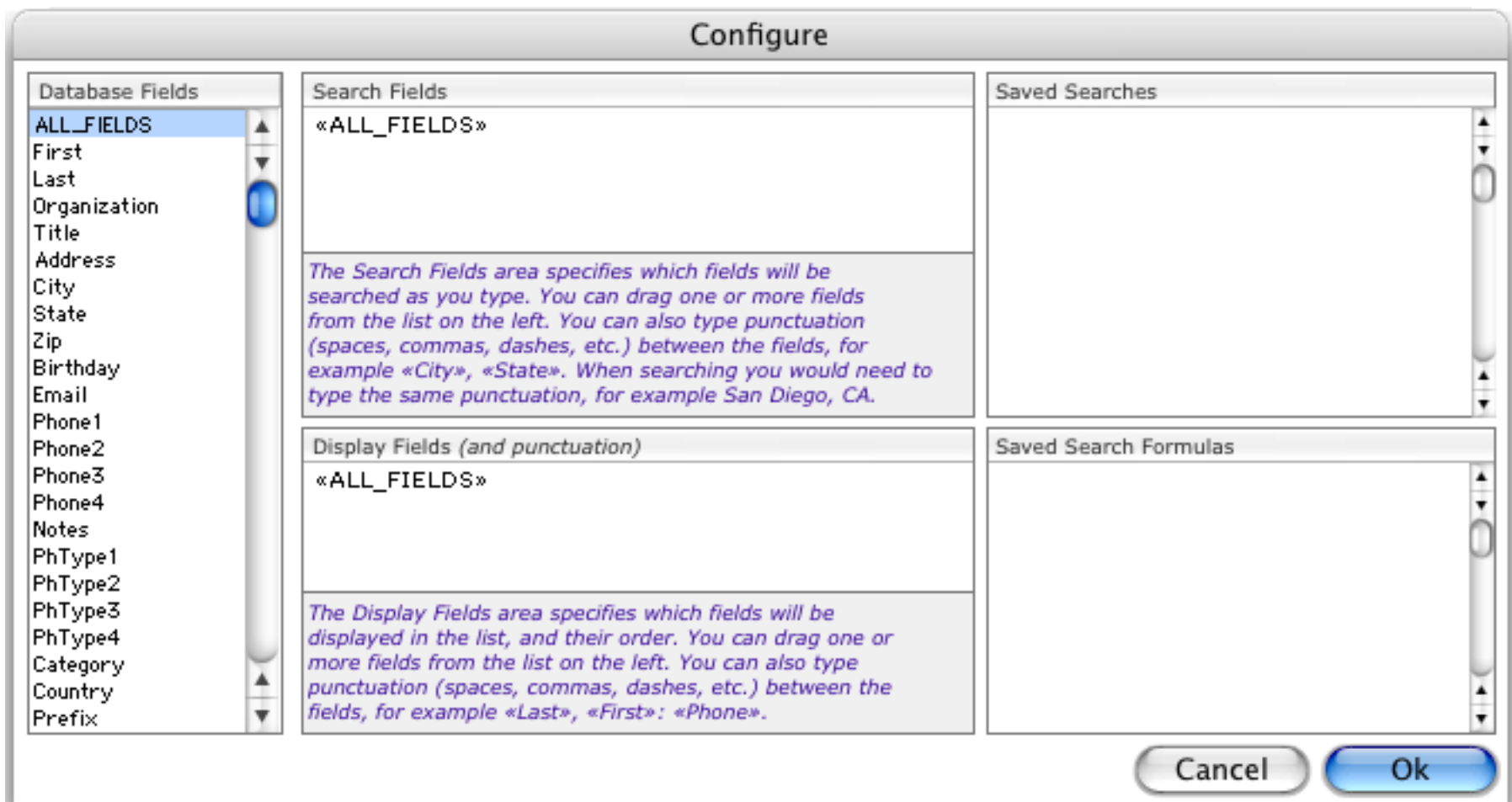
The wizard will automatically add your search to the proper spot on the menu.



You can also add and remove items from the menu with the Preferences dialog, described in the next section.

## The Live Clairvoyance Preferences Dialog

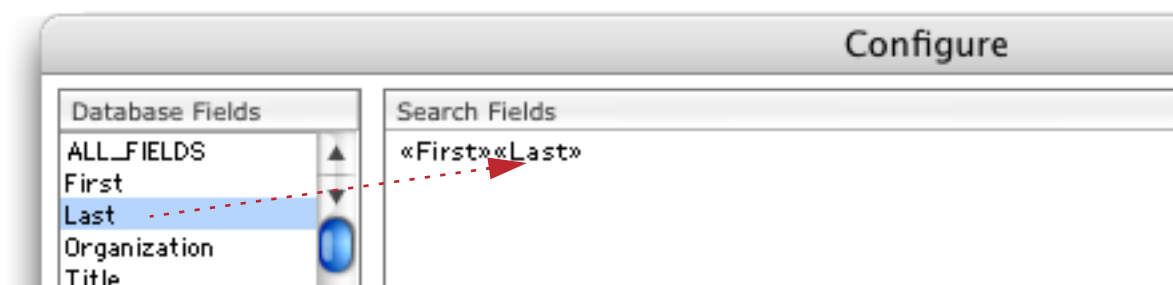
The preferences dialog allows you to customize the wizard to search and/or display only specific fields, and to customize the pop-up favorite search menu. To open this dialog choose **Configuration** from the Setup menu or press the **PREFS** tool. Here's what the dialog looks like for a database that has never been customized before.



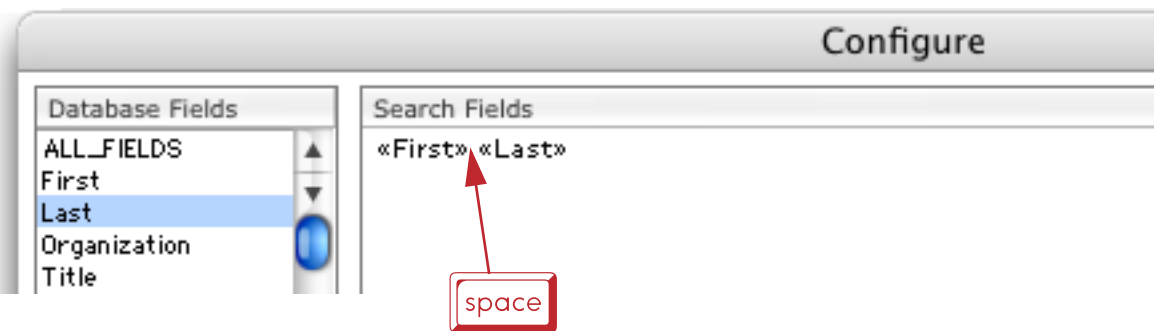
### Customizing Live Clairvoyance Basic Search

Customizing the basic search fields is especially helpful when using very large databases because searching a large number of fields can adversely affect the search speed. The left hand side of the dialog contains a list of all the fields in the target database. You can drag fields from this list into the two boxes on the right. The first item on the list is always **ALL\_FIELDS**. Use this if you want to search or display all of the fields in the database.

To set up one or more search fields, drag the fields from the list on the left to the text editing area in the upper right. You'll probably want to type some punctuation in between the fields (for example spaces and/or commas). For example, suppose you had separate fields for first and last names. Now suppose you set up the search fields like this, without any punctuation between the fields.



Now suppose you want to search for **Bob Smith**. With the search configuration set up as shown above, you would have to type **BobSmith** into the search box. A more normal way to set up this configuration would be like this:



With this configuration you could search for **Bob Smith** by typing **Bob Smith**.

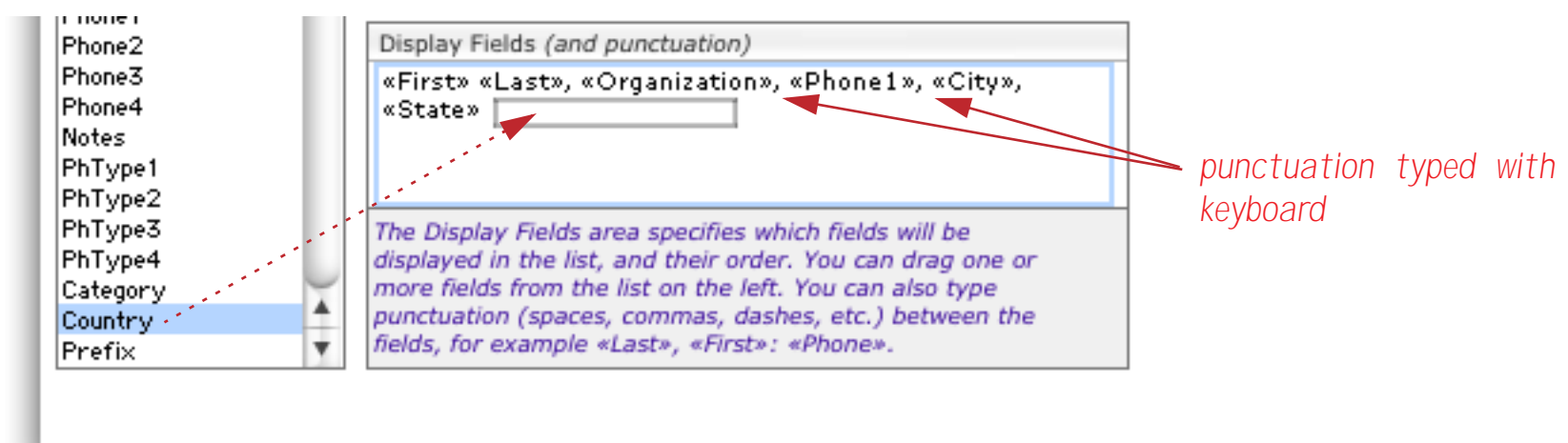
You are not limited to spaces as punctuation. Another way to set these fields up would be like this:



Now you could search for Bob by typing in **Bob, Smith** or **Smith, Bob**. Searching for **Bob Smith**, however, would not find this person. There is no right way to configure the search fields. It's basically up to you to decide what style is most comfortable for you.

#### Customizing Live Clairvoyance Data Display

Customizing the display fields can help make the display easier to understand and more useful. To customize the way records are displayed in the Live Clairvoyance list, drag fields from the list on the left to the text editing box on the bottom right. As with the search fields you can type in any punctuation you like between fields.





Once you press the **Ok** button the wizard will start using your new configuration. The illustration below shows the same database configured with the standard configuration (all fields) and with a custom configuration.

*ALL FIELDS (standard configuration)*

122 matches from 10,000 records	
Beverly, Welsh, Hills Newspapers Limited, Safety Administrator, 17671 Sandhill Dr, Duluth, MN,	
Cynthia, Metzger, Carnegie Insurance Co., Safety Administrator, 19635 S. Madison Blvd, San	
Lisa, Chapman, Matrix Computers Assoc., Safety Analyst, 363 N.W. Concord Court, Romney, WV,	
Kelly, Clarke, Light Power Foundation, Safety Analyst, 234 South Howard Pl, Charlton, MA, 01507,	
Anthony, Holland, Texas Communication College, Safety Analyst, 14510 W. Stevens Ave, Red Bluff,	
Kenneth, Nielsen, Western Bank Co, Safety Administrator, 383 E Sequoia Trail, Chattanooga, TN,	
Howard, Bradford, Hall Express Services, Safety Analyst, 835 E. Whitney Cir, Newhope, AL, 35760,	
Craig, McClelland, Kenora Management Assoc, Safety Analyst, 694 S.W Eisenhower Ter, Rochester,	
Douglas, Fishman, Allen Publications Group, Safety Analyst, 269 East Bristol Circle, Brownwood,	
Carl, Schroeder, Valley Images Corp, Safety Administrator, 1047 N. Fisher St., Miami, FL, 33169,	
Arthur, Donnelly, Michael Cable Co., Safety Analyst, 287 S Clay Ct, Lodi, WI, 53555, 12/16/26,	
Maureen, Wicks, America Network Ltd, Safety Administrator, 857 N.E. Overbrook Ter, Spring City,	

*Custom Configuration: «First» «Last», «Organization», «Phone1», «City», «State»*

122 matches from 10,000 records	
Beverly Welsh, Hills Newspapers Limited, (218) 429-1806, Duluth, MN	
Cynthia Metzger, Carnegie Insurance Co., (210) 862-8146, San Antonio, TX	
Lisa Chapman, Matrix Computers Assoc., (304) 291-8903, Romney, WV	
Kelly Clarke, Light Power Foundation, (508) 603-3476, Charlton, MA	
Anthony Holland, Texas Communication College, (916) 548-1660, Red Bluff, CA	
Kenneth Nielsen, Western Bank Co, (423) 738-7013, Chattanooga, TN	
Howard Bradford, Hall Express Services, (205) 786-4095, Newhope, AL	
Craig McClelland, Kenora Management Assoc, (716) 922-8237, Rochester, NY	
Douglas Fishman, Allen Publications Group, (713) 285-7293, Brownwood, TX	
Carl Schroeder, Valley Images Corp, (305) 578-2089, Miami, FL	
Arthur Donnelly, Michael Cable Co., (608) 396-4127, Lodi, WI	
Maureen Wicks, America Network Ltd, (610) 562-2440, Spring City, PA	

The new configuration will also be stored in a permanent variable in the target database. The next time you save the target database the Live Clairvoyance configuration for that database will be saved along with the data itself. The bottom line is that Panorama keeps a separate configuration for each of your databases.

### Customizing Data Display with Formulas

If you are a more advanced Panorama user you can insert a formula into either the search or display configuration by using curly braces { and }. Here is an example that will display only the first initial of the first name, with the last name capitalized.

```
Display Fields (and punctuation)
{First[1,1]+"."+upper(Last)}, «Organization», «Phone1»,
«City», «State»
```

Here is the resulting display.

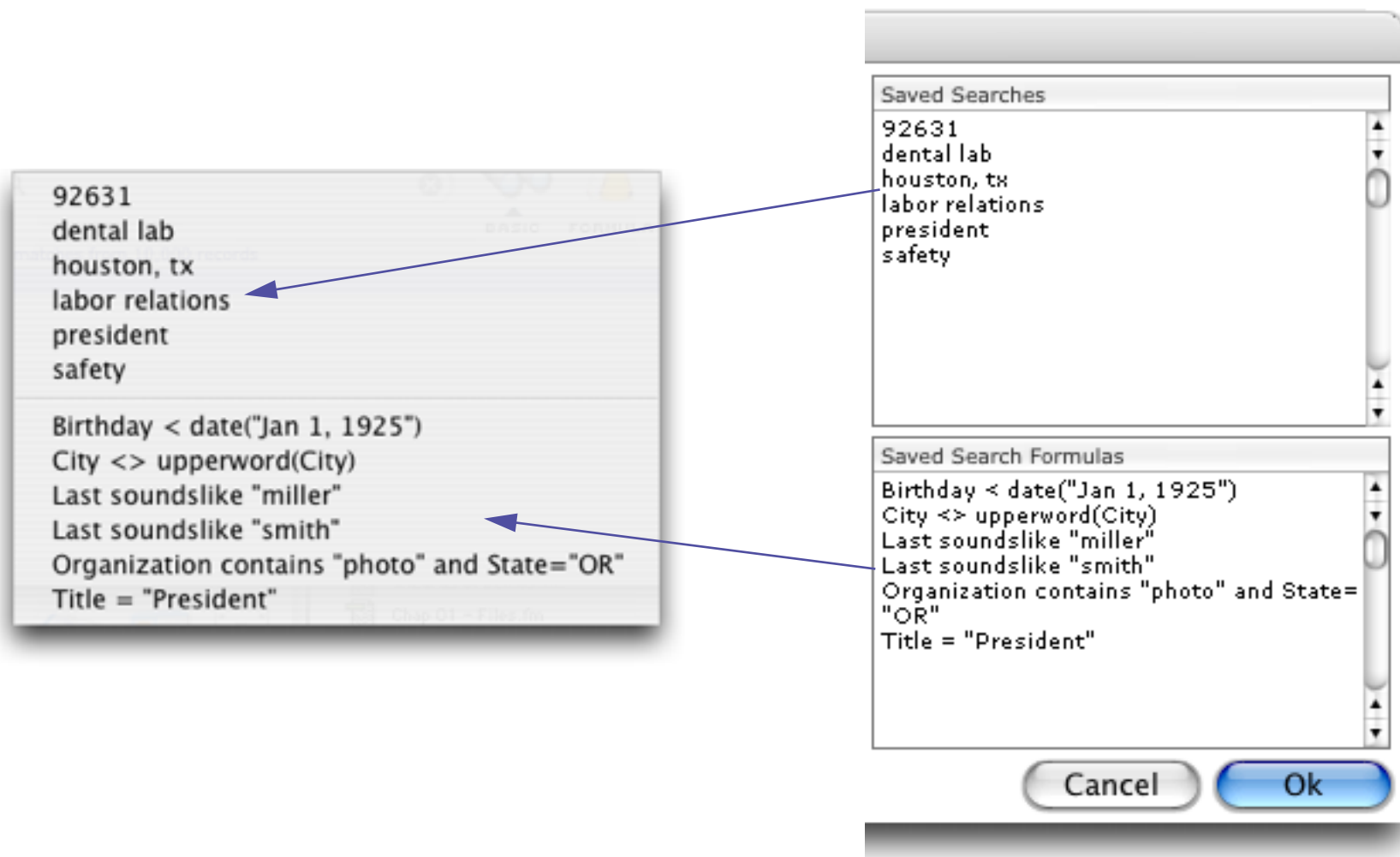
40 matches from 10,000 records	
H.CASPER, Kenora Construction Group, (508) 285-3265, Lakeville, MA	
G.ORDEN, General Construction Center, (303) 211-5528, Boulder, CO	
C.CAMERON, Rock Construction Foundation, (616) 976-1709, Holton, MI	
T.MULLANEY, Oklahoma Construction Foundation, (804) 519-6652, Midlothian, VA	



You can use any Panorama formula operator or function that you want. However, one thing you should NOT do is use the « and » characters within the formula. For example, this means that the field names must be specified as First and Last, not «First» and «Last». If a field name contains spaces or other punctuation you won't be able to use it in a formula. Note: If your formula contains an error, the Live Clairvoyance wizard performs only very limited error checking. You'll get an error message, but the message may not be very helpful in determining the problem. We recommend first debugging the formula with the **Formula Wizard**, then using **Copy** and **Paste** to transfer the formula to the **Live Clairvoyance** configuration dialog.

### Customizing the Favorite Searches Pop-up Menu

The right hand side of the Preferences dialog allows you to edit the contents of the Favorite Searches pop-up menu. You can type in new entries and edit or delete existing entries.



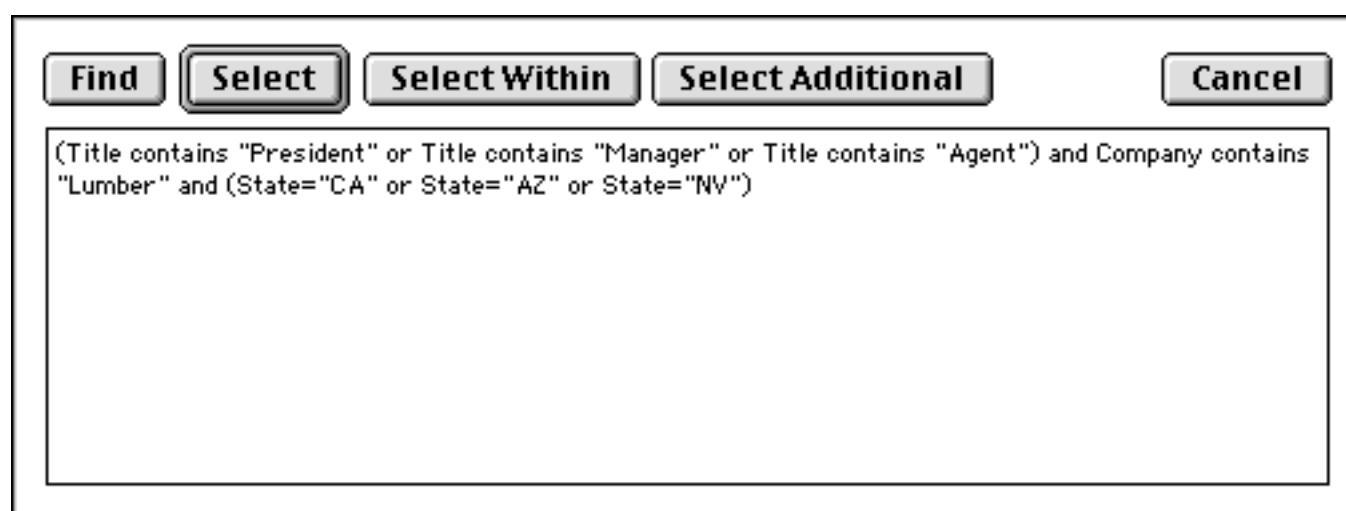
### Changing the Target Database

When you open the **Live Clairvoyance** wizard the currently active database automatically becomes the “target” database (the database that is being searched). There are two ways to change the target database:

- 1) Click on another database, then choose Live Clairvoyance from the Wizard menu again. Even if the wizard is already open, this will change the target database to the newly selected database.
- 2) With the Live Clairvoyance window open, choose the target database from the Database menu.

## Formula Find/Select

The **Find/Select** dialog is a quick and powerful way to locate data, but it does have limitations—data can only be located based on six fixed values. Another restriction is that **and** and **or** cannot be mixed. The **Formula Find/Select** dialog removes these restrictions by using a formula to locate information.



Although it is more work to set up, the **Formula Find/Select** dialog allows you to locate virtually anything that can be described by Panorama's powerful formulas.

The **Formula Find/Select** dialog relies on the ability of a formula to make comparisons and true-false decisions. See "[True/False Formulas](#)" on page 552 for a detailed explanation of **true-false logic**.

If you locate information with the **Find/Select** dialog and then open the **Formula Find/Select** dialog, the dialog will contain a formula equivalent to the criteria specified with the **Find/Select** dialog. This can be a good way to learn how to build your own formulas.

One advantage of the **Formula Find/Select** dialog is that it allows you to mix **and** and **or** operations. For example, the formula

```
(State="CA" or State="NY") and Amount>100
```

allows you to locate all transactions over \$100 in either California or New York. The parentheses allow you to force the correct order for combining the comparisons.

Another advantage of the **Formula Find/Select** dialog is that it allows you to select data based on calculations or comparisons between two fields. For example, the formula

```
Price/Cost > 2
```

allows you to quickly locate items with high profit margins.

The formula can also manipulate the data before using it. For example, this formula uses text funnels to strip off the first three characters of the name, allowing us to quickly locate all doctors in the database.

```
Name[1,3] = "Dr."
```

## Select Duplicates

The **Select Duplicates** command (in the Search Menu) provides a fast and easy way to locate duplicate information in a database. The **Select Duplicates** command does not remove the duplicates, it simply selects them so you can examine them. You can then decide what to do about each duplicate on a case-by-case basis. You may select duplicates based on a single field (for example, all duplicate company names), on multiple fields (for example, all records with duplicate address, city, and state), or on a formula that may combine fields or use partial fields (for example, all records containing duplicate area codes).

To select duplicates based on a single field, start by using the **Sort Up** command to sort the database by that field. If the database is not sorted, the **Select Duplicates** command will warn you. For example, here is a conference registration database that may contain duplicate company information. It has been sorted into alphabetical order by company.

Company Name	Street Address	Suite Box	City	State	Zip Code
Alameda Escrow	1000 Roche Bly		Santa Ana	CA	92705
Alexander Escrow	1004 Oban Dr.		San Rafael	CA	94903
Alliance Escrow	1524 Charlemaç		Marina del Rey	CA	90291
Alpha Pic	174 Bellevue A		Palo Alto	CA	94306
Alvarado, Johnson, & Wr	3542 Roadside I		Berkeley	CA	94720
American Paint	81 Norwood Av		West Chester	PA	19380
Andover Designs	425 Westerly S		New York	NY	10003
Arlington Associates	573 Dundee Rd.		Waldwick	NJ	07463
Arrow Dev.	411 Pacific		Cambridge	MA	02140
Arrow, Inc.	390 Davis St.		Los Angeles	CA	90067
Art Supplies	125 Shoreway F Suite		Los Gatos	CA	95030
Bath Co.	2994 Garcia Av		Burbank	CA	91505
Bayshore Typesetting	1221 Main Stre		San Rafael	CA	94903
Bayshore Typesetting	653 Hoover Roa		Washington	DC	20036
Belmont Printing	1094 Shady Tra Suite		Wellesley	MA	02181
Birch Catering	136 Harvey		San Francisco	CA	94104

After the database is sorted, choose the **Select Duplicates** command from the Search Menu (Make sure you have clicked on the field you want to check for duplicates before selecting the command.) This command opens a dialog box. Leave the dialog box empty and press the **OK** button. Panorama will select the records that contain duplicate information (if any), making everything else invisible.

Company Name	Street Address	Suite Box	City	State	Zip Code
Bayshore Typesetting	1221 Main Stre		San Rafael	CA	94903
Bayshore Typesetting	653 Hoover Roa		Washington	DC	20036
State Of Texas, Dept Of E	Knott Building		Austin	TX	78746
State Of Texas, Dept Of E	Knott Building		Austin	TX	78746

As you can see, there are two possible duplicates in this database.

# Chapter 10: Summaries and Outlines



It is very difficult to look at a database containing thousands of records and make much sense of it. There's simply too much information to cope with. To make the information more understandable, it needs to be summarized. Panorama can rapidly summarize a database according to the criteria you specify.

Panorama has two methods for summarizing a database. The first method, which is discussed in this chapter, is to create an outline using summary records. See "[Crosstabs](#)" on page 209 to learn about the second method, **crosstabs**.

## **3-Step Summarizing**

Summarizing a database is a three step process—**group**, **calculate**, and **outline**. Before diving into all the details and options, let's take a look at a basic overview of these three steps.

The first step is always the raw data. For this example we'll use a checkbook database, which we will summarize by category. Notice that the categories start out in more or less random order.

Date	CkNum	PayTo	Category	Debit
01/08/99	1907	Northern Illinois Mold	Equipment Rental	96.05
01/08/99	1908	U S Postmaster	Postage	75.00
01/08/99	1909	Advertiser's Mailing Ser	Advertising	390.80
01/16/99	1910	Coudert Brothers, Attor	Legal Fees	223.52
01/16/99	1911	Paramount Stationers	Office Supplies	105.84
01/17/99	1912	California Capitol	Insurance	36.00
01/17/99	1913	California Capitol	Insurance	28.00
01/17/99	1914	U S Postmaster	Postage	75.00
01/17/99	1915	Sacramento Bee	Advertising	795.00
01/22/99	1916	Walthers	Purchases	12,463.00
01/22/99	1917	Blue Cross Of Calif	Insurance	279.03
01/22/99	1918	Sherman Douglas Ins	Insurance	418.60
01/22/99	1919	Cannon Astro	Office Supplies	145.72
01/25/99	1920	Walthers	Purchases	1,885.40
01/25/99	1921	Nebs	Office Supplies	77.27
01/25/99	1922	Ramona Drinking Water	Office Supplies	98.10
01/25/99	1923	Pacific Partners	Rent	4,070.83
01/29/99	1924	Athearn	Purchases	1,906.32
01/29/99	1925	Advertiser's Mailing Ser	Advertising	860.22
01/29/99	1926	PacTel Cellular	Telephone	141.09
01/30/99	1927	State Board Of Equalizat	Taxes	549.00
01/30/99	1928	Walthers	Purchases	828.70
01/30/99	1929	Federal Express	Shipping	178.75
01/31/99	1930	U P S	Shipping	52.97
01/31/99	1931	Sacramento Bee	Advertising	795.00
02/07/99	1932	Cable & Wireless	Telephone	99.25
02/01/99	1933	Ed Burnett Consultants	Advertising	327.64
02/01/99	1934	US Sprint	Telephone	17.14
02/01/99	1935	Copierland	Office Supplies	137.04
02/01/99	1936	Alpha Graphics	Advertising	195.74
02/01/99	1937	Walthers	Purchases	536.00
02/01/99	1938	Unocal	Auto	182.59
02/01/99	1939	FastFacts	Office Supplies	108.25
02/01/99	1940	Yuba Register	Advertising	316.66
02/07/99	1941	City Of Caboose	Utilities	84.78
02/07/99	1942	U S Postmaster	Postage	75.00
02/07/99	1943	U S Postmaster	Postage	35.00
02/09/99	1944	Page One	Advertising	23.68
02/09/99	1945	Blue Cross Of Calif	Insurance	256.47
02/09/99	1946	U P S	Shipping	122.60
02/09/99	1947	Pacific Partners	Rent	3,862.63
02/09/99	1948	California Secretary Of :	Legal Fees	5.00
02/09/99	1949	City Of Caboose	Legal Fees	90.00
02/09/99	1950	Pitney Bowes	Equipment Rental	73.14

411 visible/411 total



Step 1 is **group**, so we'll start with the **Group Up** command (Sort menu) to divide the database into groups by spending category (advertising, purchases, rent, etc.). As you can see, the database is now sorted in order by category. In addition, Panorama has added several new records to the database. These are called **summary records** and can be identified by their blue background color and by the fact that they are displayed in bold.

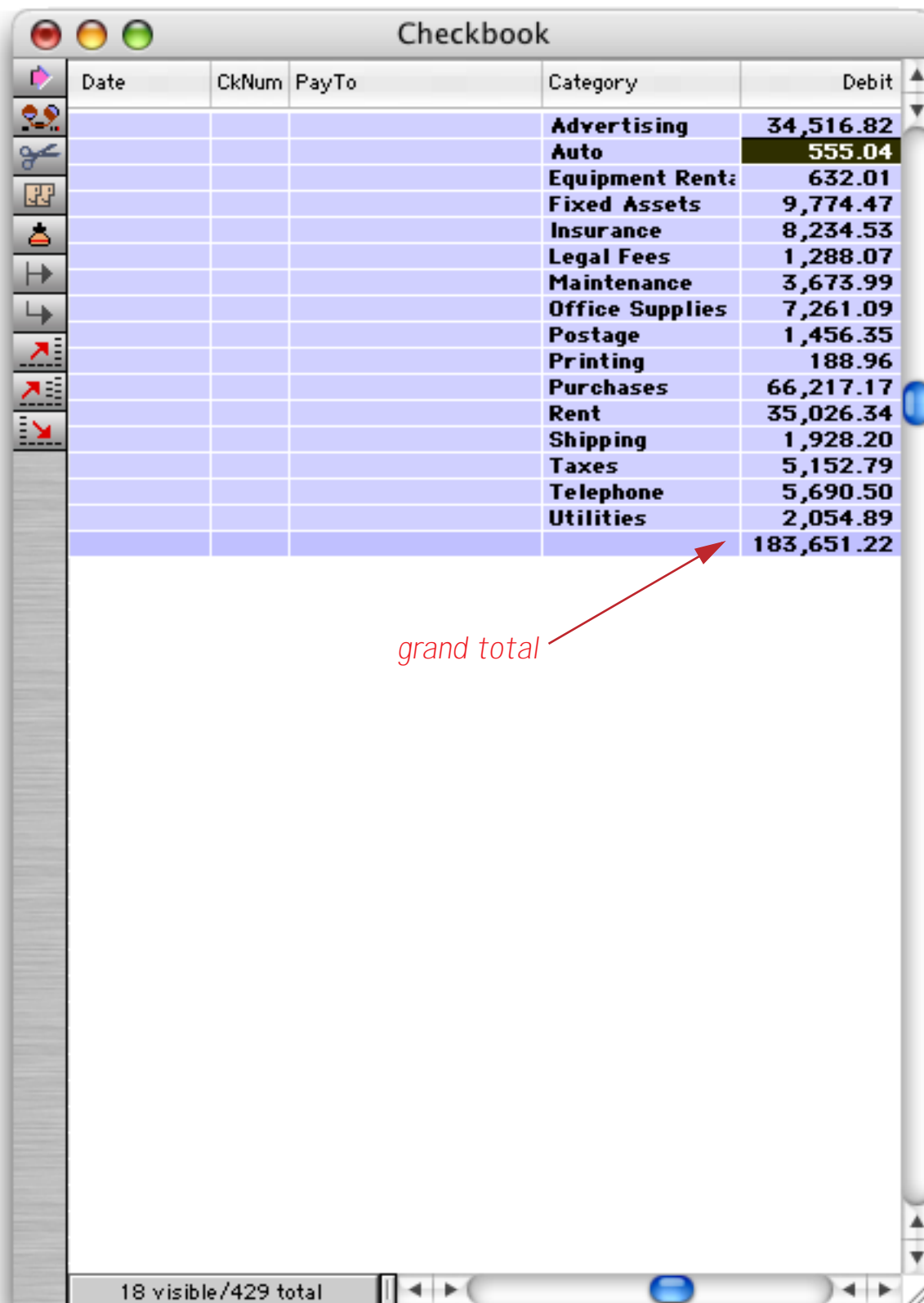
The screenshot shows the 'Checkbook' application window with a list of transactions. The transactions are sorted by category. Summary records are highlighted in blue and bold. Red arrows point from the text 'summary records' to these records.

Date	CkNum	PayTo	Category	Debit
07/16/99	2185	Railroad Model Craftsma	Advertising	453.42
07/18/99	2199	New Direction	Advertising	112.48
07/19/99	2203	Model Railroader	Advertising	110.00
07/20/99	2205	Advertiser's Mailing Ser	Advertising	27.00
07/24/99	2206	Sir Speedy	Advertising	142.40
07/24/99	2209	Advertiser's Mailing Ser	Advertising	500.00
07/24/99	2211	Railroad Model Craftsma	Advertising	453.42
08/13/99	2227	Page One	Advertising	92.05
08/14/99	2237	Advertiser's Mailing Ser	Advertising	500.00
08/29/99	2257	Advertiser's Mailing Ser	Advertising	425.00
09/06/99	2266	Advertiser's Mailing Ser	Advertising	495.41
09/18/99	2271	Railroad Model Craftsma	Advertising	453.42
09/19/99	2283	Caboose Gazette	Advertising	1,990.10
09/26/99	2297	AC Label Company	Advertising	205.97
09/28/99	2298	Graphic Depot	Advertising	344.00
09/28/99	2299	Advertiser's Mailing Ser	Advertising	167.00
			<b>Advertising</b>	
02/01/99	1938	Unocal	Auto	182.59
02/09/99	1968	Unocal	Auto	57.62
03/16/99	2007	Unocal	Auto	33.32
05/24/99	2111	Unocal	Auto	119.05
07/16/99	2189	Unocal	Auto	38.11
07/24/99	2213	Unocal	Auto	34.44
08/20/99	2240	Unocal	Auto	89.91
			<b>Auto</b>	
01/08/99	1907	Northern Illinois Mold	Equipment Rental	96.05
02/09/99	1950	Pitney Bowes	Equipment Rental	73.14
04/23/99	2063	Pitney Bowes	Equipment Rental	79.69
05/24/99	2137	Pitney Bowes	Equipment Rental	25.75
05/24/99	2141	Pitney Bowes	Equipment Rental	79.69
08/21/99	2251	Pitney Bowes	Equipment Rental	198.00
08/21/99	2253	Pitney Bowes	Equipment Rental	79.69
			<b>Equipment Renta</b>	
02/09/99	1952	GECC	Fixed Assets	428.39
05/02/99	2072	GECC	Fixed Assets	704.00
05/24/99	2112	GECC	Fixed Assets	74.00
06/14/99	2158	C M S	Fixed Assets	1,168.75
07/03/99	2175	GECC	Fixed Assets	250.00
07/18/99	2200	SSG LaserWorks	Fixed Assets	793.00
08/21/99	2243	GECC	Fixed Assets	725.00
09/18/99	2275	T.W. Bender Group	Fixed Assets	2,814.33
09/19/99	2280	GECC	Fixed Assets	352.00
09/26/99	2296	TesLabe	Fixed Assets	2,465.00
			<b>Fixed Assets</b>	

Step 2 is **calculate**, which we'll do with the **Total** command (Math menu). This command scans the database and calculates the subtotal for each category, as well as an overall total at the bottom of the database (not visible in this illustration).

Date	CkNum	PayTo	Category	Debit
07/16/99	2185	Railroad Model Craftsma	Advertising	453.42
07/18/99	2199	New Direction	Advertising	112.48
07/19/99	2203	Model Railroader	Advertising	110.00
07/20/99	2205	Advertiser's Mailing Ser	Advertising	27.00
07/24/99	2206	Sir Speedy	Advertising	142.40
07/24/99	2209	Advertiser's Mailing Ser	Advertising	500.00
07/24/99	2211	Railroad Model Craftsma	Advertising	453.42
08/13/99	2227	Page One	Advertising	92.05
08/14/99	2237	Advertiser's Mailing Ser	Advertising	500.00
08/29/99	2257	Advertiser's Mailing Ser	Advertising	425.00
09/06/99	2266	Advertiser's Mailing Ser	Advertising	495.41
09/18/99	2271	Railroad Model Craftsma	Advertising	453.42
09/19/99	2283	Caboose Gazette	Advertising	1,990.10
09/26/99	2297	AC Label Company	Advertising	205.97
09/28/99	2298	Graphic Depot	Advertising	344.00
09/28/99	2299	Advertiser's Mailing Ser	Advertising	167.00
			<b>Advertising</b>	<b>34,516.82</b>
02/01/99	1938	Unocal	Auto	182.59
02/09/99	1968	Unocal	Auto	57.62
03/16/99	2007	Unocal	Auto	33.32
05/24/99	2111	Unocal	Auto	119.05
07/16/99	2189	Unocal	Auto	38.11
07/24/99	2213	Unocal	Auto	34.44
08/20/99	2240	Unocal	Auto	89.91
			<b>Auto</b>	<b>555.04</b>
01/08/99	1907	Northern Illinois Mold	Equipment Rental	96.05
02/09/99	1950	Pitney Bowes	Equipment Rental	73.14
04/23/99	2063	Pitney Bowes	Equipment Rental	79.69
05/24/99	2137	Pitney Bowes	Equipment Rental	25.75
05/24/99	2141	Pitney Bowes	Equipment Rental	79.69
08/21/99	2251	Pitney Bowes	Equipment Rental	198.00
08/21/99	2253	Pitney Bowes	Equipment Rental	79.69
			<b>Equipment Rent:</b>	<b>632.01</b>
02/09/99	1952	GECC	Fixed Assets	428.39
05/02/99	2072	GECC	Fixed Assets	704.00
05/24/99	2112	GECC	Fixed Assets	74.00
06/14/99	2158	C M S	Fixed Assets	1,168.75
07/03/99	2175	GECC	Fixed Assets	250.00
07/18/99	2200	SSG LaserWorks	Fixed Assets	793.00
08/21/99	2243	GECC	Fixed Assets	725.00
09/18/99	2275	T.W. Bender Group	Fixed Assets	2,814.33
09/19/99	2280	GECC	Fixed Assets	352.00
09/26/99	2296	TesLabe	Fixed Assets	2,465.00
			<b>Fixed Assets</b>	<b>9,774.47</b>

Step 3 is **outline**, which allows us to hide unnecessary detail so that we can focus on just the numbers that are important to us. For our example we'll use the **Outline Level** dialog (Sort menu) to collapse the database so that only the summary information is visible. In addition to the subtotals you can also now see the grand total at the bottom. Notice that the background blue color for the grand total is slightly darker than the other summary records.



Date	CkNum	PayTo	Category	Debit
			<b>Advertising</b>	<b>34,516.82</b>
			<b>Auto</b>	<b>555.04</b>
			<b>Equipment Rentals</b>	<b>632.01</b>
			<b>Fixed Assets</b>	<b>9,774.47</b>
			<b>Insurance</b>	<b>8,234.53</b>
			<b>Legal Fees</b>	<b>1,288.07</b>
			<b>Maintenance</b>	<b>3,673.99</b>
			<b>Office Supplies</b>	<b>7,261.09</b>
			<b>Postage</b>	<b>1,456.35</b>
			<b>Printing</b>	<b>188.96</b>
			<b>Purchases</b>	<b>66,217.17</b>
			<b>Rent</b>	<b>35,026.34</b>
			<b>Shipping</b>	<b>1,928.20</b>
			<b>Taxes</b>	<b>5,152.79</b>
			<b>Telephone</b>	<b>5,690.50</b>
			<b>Utilities</b>	<b>2,054.89</b>
				<b>183,651.22</b>

18 visible/429 total

Using the **Expand** and **Collapse** tools (in the tool palette) the summaries can be expanded or collapsed to show more or less detail. Here we've used the **Expand** tool to examine the maintenance detail.

Date	CkNum	PayTo	Category	Debit
			<b>Advertising</b>	<b>34,516.82</b>
			<b>Auto</b>	<b>555.04</b>
			<b>Equipment Rent:</b>	<b>632.01</b>
			<b>Fixed Assets</b>	<b>9,774.47</b>
			<b>Insurance</b>	<b>8,234.53</b>
			<b>Legal Fees</b>	<b>1,288.07</b>
02/09/99	1964	Copierland	Maintenance	310.00
03/01/99	1986	Boyer & Ambrose Carpe	Maintenance	87.50
03/12/99	2002	Boyer & Ambrose Carpe	Maintenance	156.35
03/29/99	2037	Priority One Computers	Maintenance	496.40
03/29/99	2046	Executive Surveillance	Maintenance	132.00
04/24/99	2067	Boyer & Ambrose Carpe	Maintenance	132.00
05/08/99	2090	ServiceWorld	Maintenance	265.63
05/24/99	2114	E T S	Maintenance	49.00
05/24/99	2116	Sun Computers	Maintenance	282.00
05/24/99	2139	Pitney Bowes	Maintenance	140.00
05/29/99	2149	Sun Computers	Maintenance	276.00
06/05/99	2157	Sun Computers	Maintenance	101.25
06/21/99	2167	Dial One	Maintenance	267.13
07/16/99	2190	Executive Surveillance	Maintenance	168.00
07/24/99	2214	J & M Fire Extinguisher	Maintenance	38.19
08/08/99	2220	Newport Buidling & Mai	Maintenance	120.00
08/21/99	2252	Vint Pest Control	Maintenance	120.00
09/18/99	2270	Computek Computer	Maintenance	100.00
09/18/99	2274	Boyer & Ambrose Carpe	Maintenance	432.54
			<b>Maintenance</b>	<b>3,673.99</b>
			<b>Office Supplies</b>	<b>7,261.09</b>
			<b>Postage</b>	<b>1,456.35</b>
			<b>Printing</b>	<b>188.96</b>
			<b>Purchases</b>	<b>66,217.17</b>
			<b>Rent</b>	<b>35,026.34</b>
			<b>Shipping</b>	<b>1,928.20</b>
			<b>Taxes</b>	<b>5,152.79</b>
			<b>Telephone</b>	<b>5,690.50</b>
			<b>Utilities</b>	<b>2,054.89</b>
				<b>183,651.22</b>

When you are finished with the summary, use the **Remove Summaries** command (Sort menu) to remove the subtotals and totals, leaving only the original data. You can then continue with normal operations on your database (data entry, sorting, searching, etc.).

Tip: The 3-Step summary process is an ideal candidate for automation with procedures. See "[Procedures](#)" on page 571 for information on recording and using a procedure. Panorama also comes with a special wizard to help automate the summary process, the **Summary & Outline** wizard. See Chapter 10 of the **Panorama Handbook** to learn more about this wizard.

Now that you've seen the basic three step summary process, let's look at each step in detail.

## STEP 1 - GROUP

The first step in summarizing a database is to divide the database into groups or categories. For example, a checkbook database could be arranged into groups by month (Jan, Feb, Mar, etc.), by budget category (rent, food, transportation, etc.), or by payee (Evian Apartments, Lakeman's Market, Unocal, etc.).

To divide a database into groups, first click anywhere in the field you want to group. Then use either the **Group Up** or **Group Down** command (Sort menu) to divide the database. The **Group Up** command arranges the data into ascending order—A's first, Z's last. The **Group Down** command arranges the data in descending order, Z to A.

The **Group Up** and **Group Down** commands add a special summary record at the end of each group. Summary records are temporary records used for calculating and displaying summary information. On the data sheet you can easily identify summary records by their blue background color, and by the fact that they are usually displayed in bold.

09/19/99	2291	S C E	Utilities	81.13
09/19/99	2292	So. Calif. Gas Co.	Utilities	154.95
			<b>Utilities</b>	<b>2,054.89</b>
				<b>183,651.22</b>

*darkness of blue background indicates summary level*

### Subgroups

Groups can be sub-divided into even smaller subgroups. For example if you had arranged a mailing list into groups by state, you could further divide each state into subgroups by city. You can continue subdividing the groups up to six times (up to seven levels of groups within groups).

To subdivide groups into smaller groups, first click on the field you want to sub-group, then use the **Group Up** or **Group Down** command again.

When you are looking at the data sheet, you can identify the subgroup level by the size of the plus sign to the left of the summary record. The lowest level subgroups have the smallest plus signs; the higher level groups have larger plus signs. The grand-total record has the largest plus sign (see illustration above).

### Grand Total

When you arrange a database into groups, Panorama automatically creates an additional summary record at the bottom of the database. This summary record is for the largest group of all, the entire database. When totals or other summary calculations are performed (see "[STEP 2 - CALCULATE](#)" on page 192), this summary record holds the overall grand total (or average, count, etc.) for all of the selected records in the entire database.



If all you need is a grand total (or average, count, etc.), you can skip Step 1 and go directly to Step 2, Calculate. When a summary calculation is performed on a database that doesn't have any summary records, Panorama automatically appends a single summary record to the end of the database. It then calculates the grand total (or average, count, whatever) in this summary record. The illustration below shows the result after the **Total** command has been used without first grouping the database.

09/19/99	2292	So. Calif. Gas Co.	Utilities	154.95
09/19/99	2293	Pace Club	Office Supplies	168.00
09/21/99	2294	Advertiser's Mailing Ser	Postage	167.00
09/21/99	2295	Advertiser's Mailing Ser	Postage	67.00
09/26/99	2296	TesLabe	Fixed Assets	2,465.00
09/26/99	2297	AC Label Company	Advertising	205.97
09/28/99	2298	Graphic Depot	Advertising	344.00
09/28/99	2299	Advertiser's Mailing Ser	Advertising	167.00
<b>08/01/04</b>	<b>2311</b>			<b>183,651.22</b>

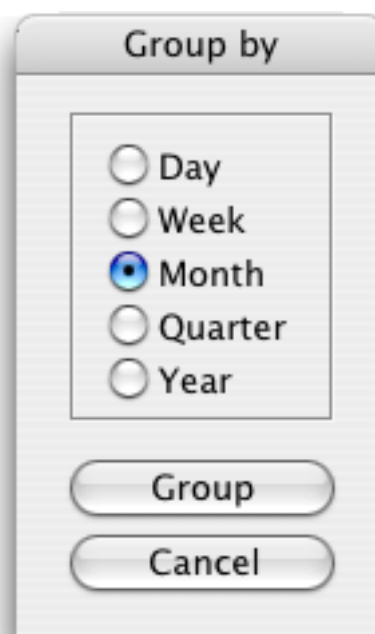
412 visible / 412 total

### The Group Command

Unlike **Group Up** and **Group Down**, the **Group** command does not sort the database. The **Group** command is useful when you want to add summary records to a database that is already arranged in the proper order.

### Grouping by Week, Month, Quarter, or Year

When a group command is used on a field containing dates, Panorama will ask you how long each group should be—a week, month, quarter, or year.



Select the period and press **Group** to arrange the data into groups. Note: The dates must be stored using the date data type, not text.

You can group dates more than once—for instance first by year, then by month. This produces subgroups (in this case by month) within the larger groups (by year).

Grouping a database by month, quarter, or year does not change the way the dates are displayed. You may want to change the output pattern for the date field so that only the month, quarter, or year is displayed, instead of the entire date. The output pattern is specified as part of the design sheet. To display only the month and year, use an output pattern like Mon-yy or mm-yy. To display only the quarter and year, the pattern could be **qqyy** or **Qtr "Qtr" yy**. To display the year only, the pattern could simply be **yy** or **yyyy**. See “[Date Output Patterns](#)” on page 121 for more information about date output patterns and how to set them up.

If you print a database that is grouped by month, quarter, or year, you can use different summary tiles to format the dates properly. See “[Printing Data Grouped by Month, Quarter or Year](#)” on page 487.

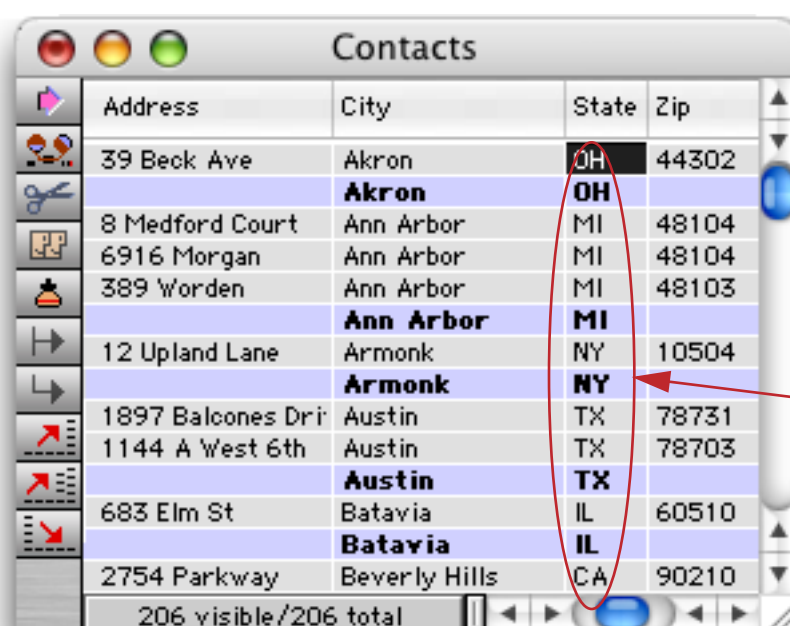
### Propagating Data into Summary Records

The Group commands create summary records but leaves most fields blank. The **Propagate** command can be used to copy additional information into the newly created summary records. In the illustration below the database has been Grouped by **City**.



Address	City	State	Zip
39 Beck Ave	Akron	OH	44302
	<b>Akron</b>		
8 Medford Court	Ann Arbor	MI	48104
6916 Morgan	Ann Arbor	MI	48104
389 Worden	Ann Arbor	MI	48103
	<b>Ann Arbor</b>		
12 Upland Lane	Armonk	NY	10504
	<b>Armonk</b>		
1897 Balcones Dri	Austin	TX	78731
1144 A West 6th	Austin	TX	78703
	<b>Austin</b>		
683 Elm St	Batavia	IL	60510
	<b>Batavia</b>		
2754 Parkway	Beverly Hills	CA	90210

Next we'll fill in the **State** field for each summary record by choosing **Propagate** from the Math menu.

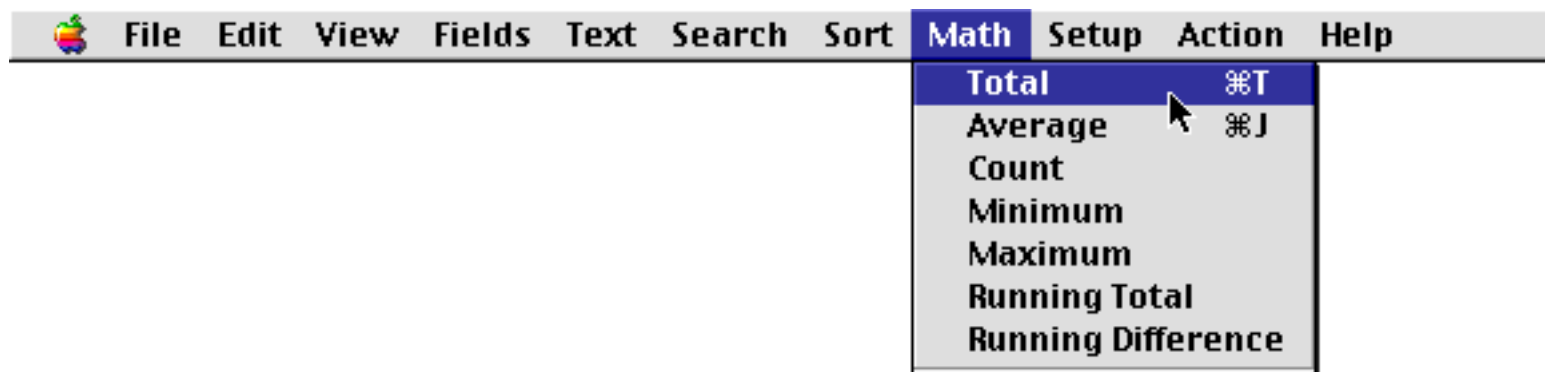


Address	City	State	Zip
39 Beck Ave	Akron	OH	44302
	<b>Akron</b>	<b>OH</b>	
8 Medford Court	Ann Arbor	MI	48104
6916 Morgan	Ann Arbor	MI	48104
389 Worden	Ann Arbor	MI	48103
	<b>Ann Arbor</b>	<b>MI</b>	
12 Upland Lane	Armonk	NY	10504
	<b>Armonk</b>	<b>NY</b>	
1897 Balcones Dri	Austin	TX	78731
1144 A West 6th	Austin	TX	78703
	<b>Austin</b>	<b>TX</b>	
683 Elm St	Batavia	IL	60510
	<b>Batavia</b>	<b>IL</b>	
2754 Parkway	Beverly Hills	CA	90210

*summaries filled in with Propagate*

## STEP 2 - CALCULATE

Once the database has been arranged into groups, the next step is to calculate the summary information. Panorama's Math menu has 7 different kinds of summary calculations).



To perform a calculation, first pick the field you want to calculate by clicking on it. Then choose the command (**Total**, **Average**, etc.) from the Math menu. Panorama will calculate the summaries for each sub-group, group, and for the entire database.

### Total

The **Total** command adds up the data in the current field. It calculates subtotals for each group and the grand total for the entire database. The **Total** command can only be used with numeric fields. If you attempt to total a text, date, or choice field, Panorama will display a warning message.

### Count

The **Count** command counts the number of non-empty data cells in the current field. If the database is arranged into groups, it will also count the number of non-empty data cells in each group. Empty data cells will not be counted. You can count any field containing either text or numbers, but dates cannot be counted. (Date fields cannot be counted because Panorama would be unable to correctly display the result.)

### Average

The **Average** command averages the data in the current field, calculating sub averages for each group and the overall average for the entire database. Averages can only be computed for numeric and date fields. If you attempt to average a non-numeric field Panorama will display a warning message.

### Minimum

The **Minimum** command finds the smallest value in the current field. If the database is arranged into groups, it will find the smallest value in each group and sub-group. The **Minimum** command can be used with text, numeric or date fields.

### Maximum

The **Maximum** command finds the largest value in the current field. If the database is arranged into groups, it will find the largest value in each group and sub-group. The **Maximum** command can be used with text, numeric or date fields.

### Recalculating Summaries

Summaries are not re-calculated automatically when the database changes. If the information in the database changes, you must go back and use the Math menu to re-calculate. If the summary records have been deleted, or if the categories have changed, you must remove the summary records and re-group the database before you can re-calculate the new summary values.

## Running Total

**Running Total** is a special computation. Unlike the other summary calculations, **Running Total** modifies every data cell in the active field, not just the summary records. Like the **Total** computation, **Running Total** starts at the top of the database and adds up each data cell as it moves down the column. **Running Total**, however, replaces each data cell with the current total. The result is a field which contains the cumulative total at each point in the database. This is very useful for computing checking account balances, sales year to date, and other cumulative statistics.

If you use the **Running Total** command on your raw data, the raw data will be destroyed in the process of calculating the running total. We recommend that you avoid this problem by creating an extra field to hold the running total. You can use the **Formula Fill** command to copy the data into the field, and then use the **Running Total** command without disturbing the original raw data.

### Using Running Total to Balance a Checkbook

Let's take a look at how to balance a checkbook using the **Running Total** command. Start with an empty **Balance** field.

Date	CkNum	PayTo	Category	Debit	Credit	Balance
01/01/99		OPENING BALANCE			35,978.47	
01/08/99	1907	Northern Illinois Mold	Equipment Rental	96.05		
01/08/99	1908	U S Postmaster	Postage	75.00		
01/08/99	1909	Advertiser's Mailing Ser	Advertising	390.80		
01/16/99	1910	Coudert Brothers, Attor	Legal Fees	223.52		
01/16/99	1911	Paramount Stationers	Office Supplies	105.84		
01/17/99	1912	California Capitol	Insurance	36.00		
01/17/99	1913	California Capitol	Insurance	28.00		
01/17/99	1914	U S Postmaster	Postage	75.00		
01/17/99	1915	Sacramento Bee	Advertising	795.00		
01/18/99		DEPOSIT			3,846.32	
01/22/99	1916	Walthers	Purchases	12,463.00		

Use the **Formula Fill** command to calculate the how much the balance changes for each line. The formula is **Credit-Debit**.

Enter the formula:

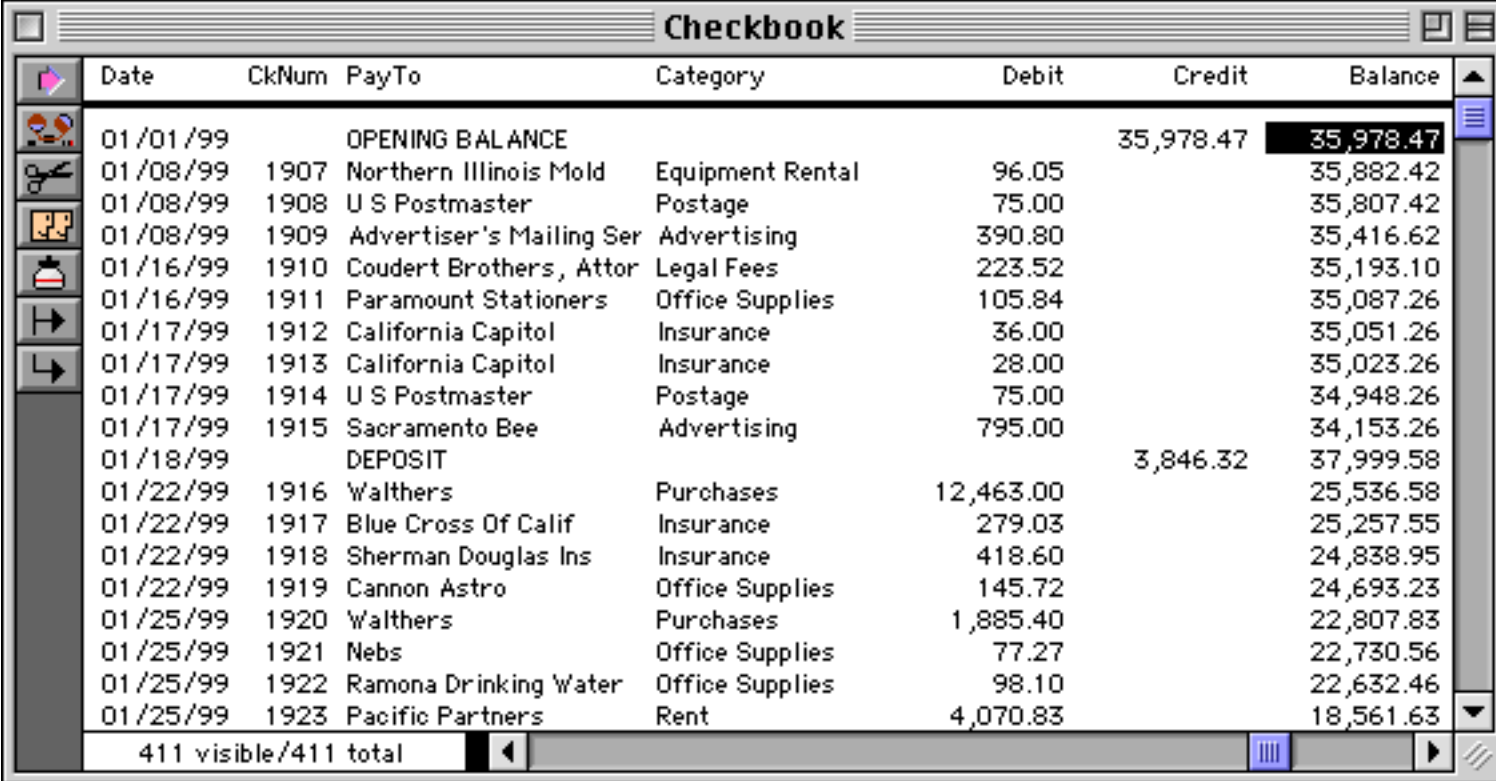
Credit-Debit

Cancel OK

Press the **OK** button to calculate the formula.

Date	CkNum	PayTo	Category	Debit	Credit	Balance
01/01/99		OPENING BALANCE			35,978.47	35,978.47
01/08/99	1907	Northern Illinois Mold	Equipment Rental	96.05		-96.05
01/08/99	1908	U S Postmaster	Postage	75.00		-171.05
01/08/99	1909	Advertiser's Mailing Ser	Advertising	390.80		-561.85
01/16/99	1910	Coudert Brothers, Attor	Legal Fees	223.52		-785.37
01/16/99	1911	Paramount Stationers	Office Supplies	105.84		-891.21
01/17/99	1912	California Capitol	Insurance	36.00		-927.21
01/17/99	1913	California Capitol	Insurance	28.00		-955.21
01/17/99	1914	U S Postmaster	Postage	75.00		-1,030.21
01/17/99	1915	Sacramento Bee	Advertising	795.00		-1,825.21
01/18/99		DEPOSIT			3,846.32	1,021.11
01/22/99	1916	Walthers	Purchases	12,463.00		-11,441.89

Now choose the **Running Total** command from the Math menu to calculate the balance after each transaction.



Date	CkNum	PayTo	Category	Debit	Credit	Balance
01/01/99		OPENING BALANCE			35,978.47	35,978.47
01/08/99	1907	Northern Illinois Mold	Equipment Rental	96.05		35,882.42
01/08/99	1908	U S Postmaster	Postage	75.00		35,807.42
01/08/99	1909	Advertiser's Mailing Ser	Advertising	390.80		35,416.62
01/16/99	1910	Coudert Brothers, Attor	Legal Fees	223.52		35,193.10
01/16/99	1911	Paramount Stationers	Office Supplies	105.84		35,087.26
01/17/99	1912	California Capitol	Insurance	36.00		35,051.26
01/17/99	1913	California Capitol	Insurance	28.00		35,023.26
01/17/99	1914	U S Postmaster	Postage	75.00		34,948.26
01/17/99	1915	Sacramento Bee	Advertising	795.00		34,153.26
01/18/99		DEPOSIT			3,846.32	37,999.58
01/22/99	1916	Walthers	Purchases	12,463.00		25,536.58
01/22/99	1917	Blue Cross Of Calif	Insurance	279.03		25,257.55
01/22/99	1918	Sherman Douglas Ins	Insurance	418.60		24,838.95
01/22/99	1919	Cannon Astro	Office Supplies	145.72		24,693.23
01/25/99	1920	Walthers	Purchases	1,885.40		22,807.83
01/25/99	1921	Nebs	Office Supplies	77.27		22,730.56
01/25/99	1922	Ramona Drinking Water	Office Supplies	98.10		22,632.46
01/25/99	1923	Pacific Partners	Rent	4,070.83		18,561.63

You'll need to repeat these steps when new transactions are added, or existing transactions changed. This process can be automated with a procedure.

```
field Balance
formulafill Credit-Debit
runningtotal
```

See "[Procedures](#)" on page 571 for more information on creating procedures.

### STEP 3 - OUTLINE

Outlines are a way of organizing information into groups within groups. Panorama's group commands rearrange your database into an outline structure. Tip: Unlike most outlines which start from the top, Panorama's outline is upside-down...it starts from the bottom.

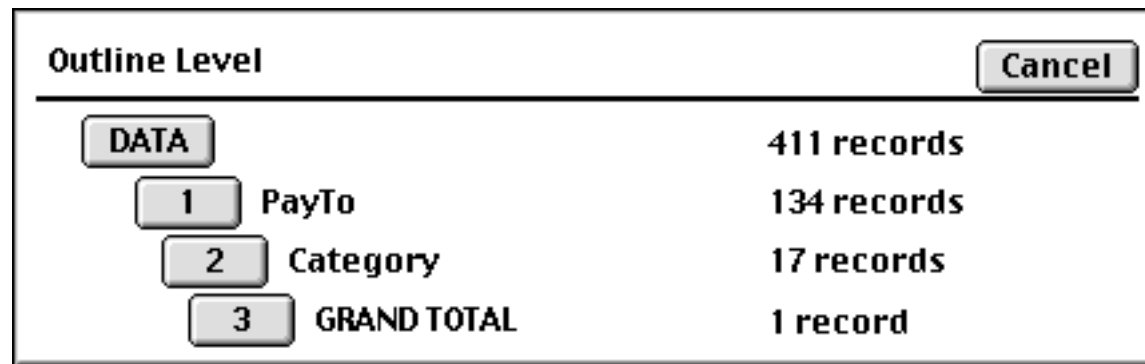
Unlike a paper outline, a Panorama outline can be expanded or collapsed to show more or less detail. This makes it easy to spot overall trends in your data, and then zero in on the details behind those trends.

#### The Outline Level

The **Outline Level** dialog (in the Sort menu) can be used to hide the database detail, leaving only the summary information visible. The **Outline Level** dialog displays a diagram of the database outline structure as it is currently grouped. The diagram shows each level of the database, where it came from, and how much data it contains.



Like the Find/Select command (see “[The Find/Select Dialog](#)” on page 160), the Outline Level dialog makes part of the database temporarily vanish. The Outline Level dialog allows you to hide all data below a certain summary level, leaving only the higher summary levels. The numbered buttons on the left side of the dialog allow you to choose how much detail you want to see. Low numbers display more detail, higher numbers display less detail. Press the **Data** button to make everything visible.

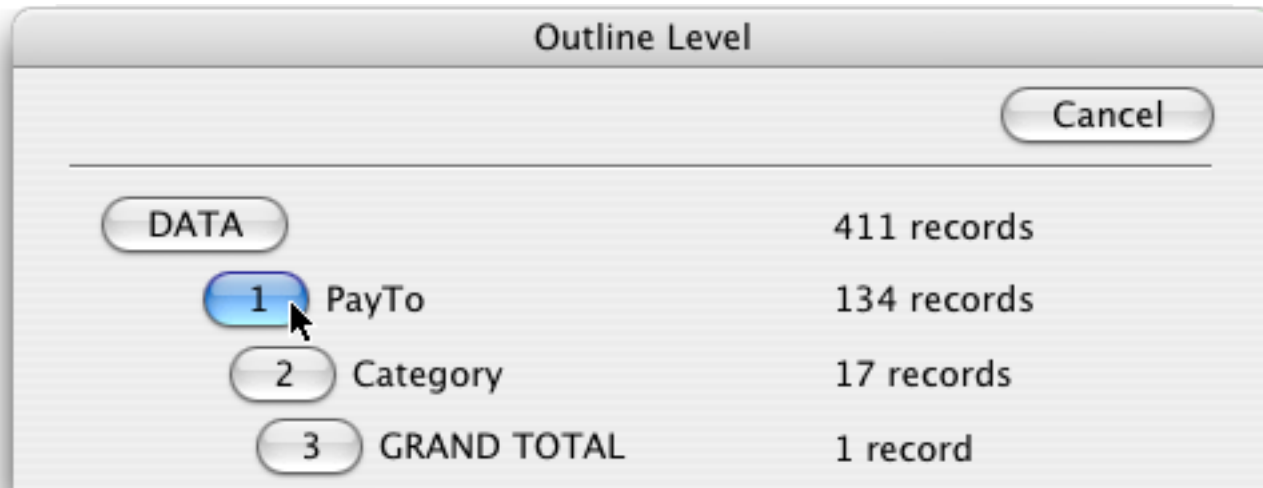


Some examples will help make the operation of this dialog clear. We'll start with a checkbook database that has been grouped by **Category** and by **PayTo** name.

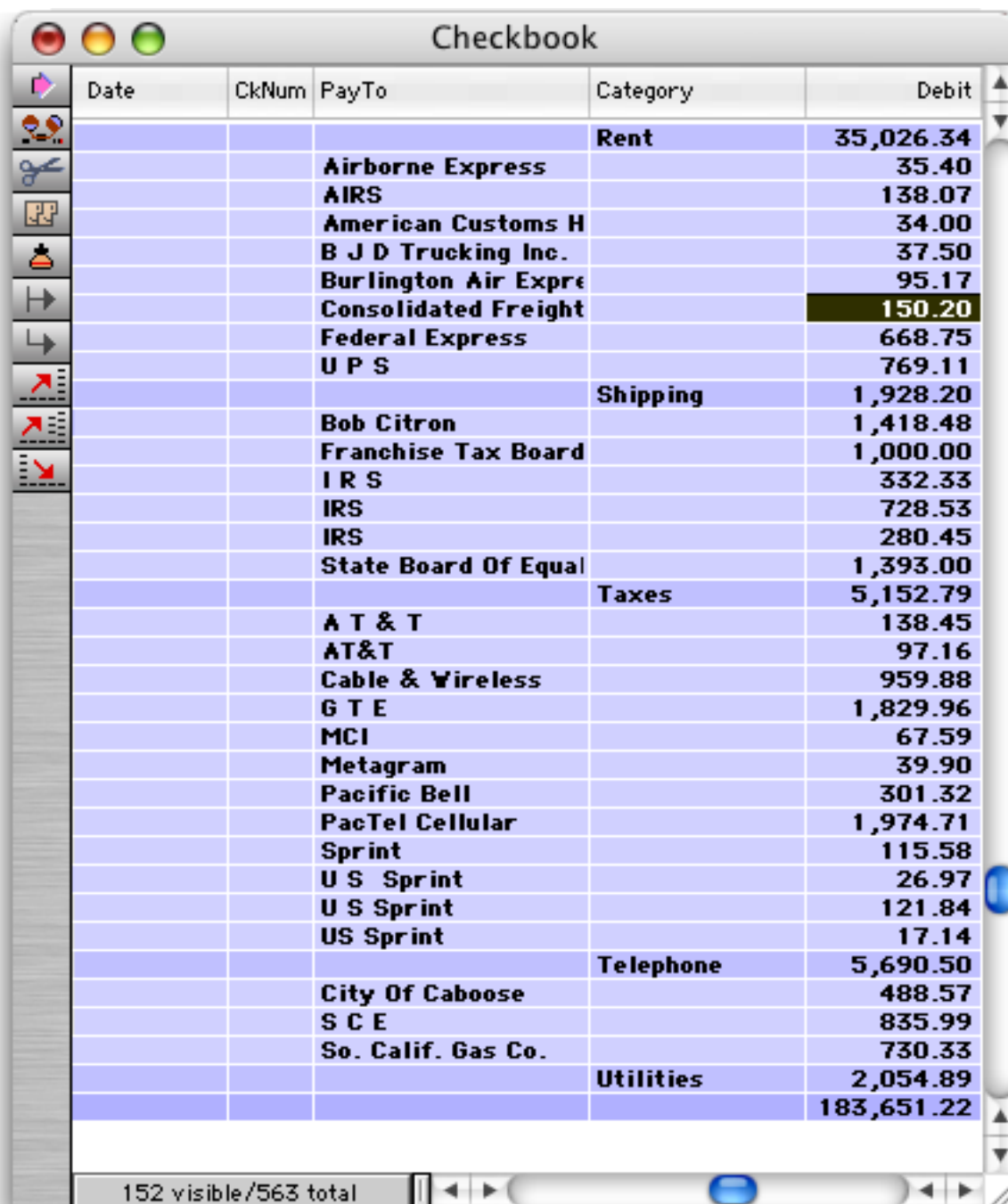
Date	CkNum	PayTo	Category	Debit
05/24/99	2126	Sprint	Telephone	51.02
08/21/99	2248	Sprint	Telephone	26.84
		<b>Sprint</b>		<b>115.58</b>
05/07/99	2082	U S Sprint	Telephone	26.97
		<b>U S Sprint</b>		<b>26.97</b>
03/28/99	2022	U S Sprint	Telephone	42.31
06/05/99	2152	U S Sprint	Telephone	79.53
		<b>U S Sprint</b>		<b>121.84</b>
02/01/99	1934	US Sprint	Telephone	17.14
		<b>US Sprint</b>		<b>17.14</b>
			<b>Telephone</b>	<b>5,690.50</b>
02/07/99	1941	City Of Caboose	Utilities	84.78
02/09/99	1953	City Of Caboose	Utilities	9.64
03/29/99	2041	City Of Caboose	Utilities	77.71
05/24/99	2119	City Of Caboose	Utilities	114.77
07/24/99	2212	City Of Caboose	Utilities	98.52
09/19/99	2290	City Of Caboose	Utilities	103.15
		<b>City Of Caboose</b>		<b>488.57</b>
02/09/99	1973	S C E	Utilities	172.03
03/29/99	2043	S C E	Utilities	89.46
05/07/99	2083	S C E	Utilities	96.26
05/24/99	2118	S C E	Utilities	97.00
06/05/99	2154	S C E	Utilities	157.31
06/14/99	2161	S C E	Utilities	56.27
08/13/99	2235	S C E	Utilities	86.53
09/19/99	2291	S C E	Utilities	81.13
		<b>S C E</b>		<b>835.99</b>
02/09/99	1972	So. Calif. Gas Co.	Utilities	136.33
03/29/99	2042	So. Calif. Gas Co.	Utilities	217.32
05/07/99	2085	So. Calif. Gas Co.	Utilities	86.74
05/24/99	2117	So. Calif. Gas Co.	Utilities	134.99
09/19/99	2292	So. Calif. Gas Co.	Utilities	154.95
		<b>So. Calif. Gas Co.</b>		<b>730.33</b>
			<b>Utilities</b>	<b>2,054.89</b>
				<b>183,651.22</b>

563 visible/563 total

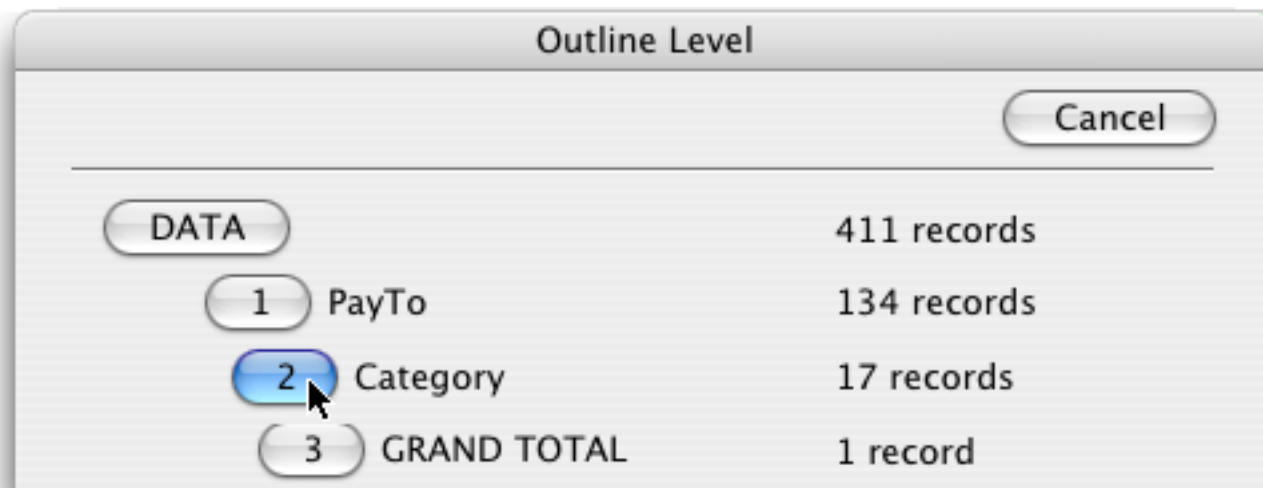
Now open the Outline Level dialog...



Press 1 to collapse the database so that none of the raw data is visible, only summary records.



Now open the Outline Level dialog again to collapse further...



Press **2** to collapse the database so that only the category subtotals and grand totals are visible.

Date	CkNum	PayTo	Category	Debit
			<b>Advertising</b>	<b>34,516.82</b>
			<b>Auto</b>	<b>555.04</b>
			<b>Equipment Rentals</b>	<b>632.01</b>
			<b>Fixed Assets</b>	<b>9,774.47</b>
			<b>Insurance</b>	<b>8,234.53</b>
			<b>Legal Fees</b>	<b>1,288.07</b>
			<b>Maintenance</b>	<b>3,673.99</b>
			<b>Office Supplies</b>	<b>7,261.09</b>
			<b>Postage</b>	<b>1,456.35</b>
			<b>Printing</b>	<b>188.96</b>
			<b>Purchases</b>	<b>66,217.17</b>
			<b>Rent</b>	<b>35,026.34</b>
			<b>Shipping</b>	<b>1,928.20</b>
			<b>Taxes</b>	<b>5,152.79</b>
			<b>Telephone</b>	<b>5,690.50</b>
			<b>Utilities</b>	<b>2,054.89</b>
				<b>183,651.22</b>

18 visible/563 total

If you want to collapse the database completely so that only the Grand Total is visible, press **3**. Or you can go back to showing all of the data by pressing the **Data** button.

### Collapsing vs. Selecting

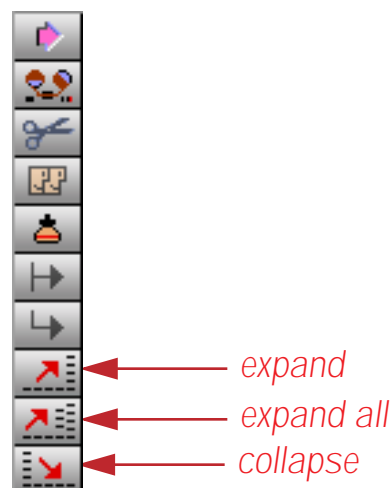
Collapsing and selecting are very similar in one way—both hide data from view. In spite of this similarity, collapsing and selecting are independent of each other. Data that is selected can be invisible because it is collapsed, and data that is expanded can be invisible because it is not selected. Data is only visible when it is both selected and expanded.

This distinction can sometimes be confusing, since you can't always tell at a glance why data is invisible. Suppose you select a subset of the database, and then collapse the database outline. Some of the data is invisible because it is de-selected. Some of the data is invisible because it is collapsed.

To make sure that all the data is visible, you must use both the **Outline Level** command to expand all the data and the **Select All** command to select all the data.

### Expanding and Collapsing Specific Details

The **Outline Level** dialog expands and collapses the entire database. You can use the **Expand**, **Expand All**, and **Collapse** tools to expand or collapse a specific area of the database. These tools automatically appear when summary records are added to your database, and disappear when they are removed.



To expand a specific summary, click on the summary record you want to expand and then press the **Expand** tool. This tool expands the next level of detail for that summary record only. To illustrate this we'll use our checkbook database which has been grouped by **Category** and **PayTo** (see "[STEP 1 - GROUP](#)" on page 189), and collapsed to show only the **Category** subtotals (see "[The Outline Level](#)" on page 194). To expand a specific category, click on the line and press the **Expand** tool.

Date	CkNum	PayTo	Category	Debit
			<b>Advertising</b>	<b>34,516.82</b>
			<b>Auto</b>	<b>555.04</b>
			<b>Equipment Rentals</b>	<b>632.01</b>
			<b>Fixed Assets</b>	<b>9,774.47</b>
			<b>Insurance</b>	<b>8,234.53</b>
			<b>Legal Fees</b>	<b>1,288.07</b>
			<b>Maintenance</b>	<b>3,673.99</b>
			<b>Office Supplies</b>	<b>7,261.09</b>
			<b>Postage</b>	<b>1,456.35</b>
			<b>Printing</b>	<b>188.96</b>
			<b>Purchases</b>	<b>66,217.17</b>
			<b>Rent</b>	<b>35,026.34</b>
			<b>Shipping</b>	<b>1,928.20</b>
			<b>Taxes</b>	<b>5,152.79</b>
			<b>Telephone</b>	<b>5,690.50</b>
			<b>Utilities</b>	<b>2,054.89</b>
				<b>183,651.22</b>

Now you can see all of the subtotals for each **PayTo** within the **Purchases** category.

Date	CkNum	PayTo	Category	Debit
			<b>Advertising</b>	<b>34,516.82</b>
			<b>Auto</b>	<b>555.04</b>
			<b>Equipment Rent:</b>	<b>632.01</b>
			<b>Fixed Assets</b>	<b>9,774.47</b>
			<b>Insurance</b>	<b>8,234.53</b>
			<b>Legal Fees</b>	<b>1,288.07</b>
			<b>Maintenance</b>	<b>3,673.99</b>
			<b>Office Supplies</b>	<b>7,261.09</b>
			<b>Postage</b>	<b>1,456.35</b>
			<b>Printing</b>	<b>188.96</b>
		<b>Athearn</b>		<b>5,458.23</b>
		<b>Atlas</b>		<b>167.61</b>
		<b>Con-Cor</b>		<b>15,609.34</b>
		<b>CTex</b>		<b>1,023.98</b>
		<b>Detail Associates</b>		<b>104.10</b>
		<b>El Mar Corporation</b>		<b>257.00</b>
		<b>El Mar Plastics</b>		<b>1,757.00</b>
		<b>Life-Like</b>		<b>500.93</b>
		<b>Mantua</b>		<b>277.76</b>
		<b>MDC</b>		<b>4,164.83</b>
		<b>Potter Manufacturin</b>		<b>1,115.43</b>
		<b>ProLitho</b>		<b>8,630.74</b>
		<b>Telon Productions</b>		<b>5,146.82</b>
		<b>Walthers</b>		<b>22,003.40</b>
			<b>Purchases</b>	<b>66,217.17</b>
			<b>Rent</b>	<b>35,026.34</b>
			<b>Shipping</b>	<b>1,928.20</b>
			<b>Taxes</b>	<b>5,152.79</b>
			<b>Telephone</b>	<b>5,690.50</b>
			<b>Utilities</b>	<b>2,054.89</b>
				<b>183,651.22</b>

32 visible / 563 total



In this example you can expand further. You can always expand until you get to the raw data. Simply click on the subtotal line and press the **Expand** tool.

Date	CkNum	PayTo	Category	Debit
			<b>Advertising</b>	<b>34,516.82</b>
			<b>Auto</b>	<b>555.04</b>
			<b>Equipment Rent:</b>	<b>632.01</b>
			<b>Fixed Assets</b>	<b>9,774.47</b>
			<b>Insurance</b>	<b>8,234.53</b>
			<b>Legal Fees</b>	<b>1,288.07</b>
			<b>Maintenance</b>	<b>3,673.99</b>
			<b>Office Supplies</b>	<b>7,261.09</b>
			<b>Postage</b>	<b>1,456.35</b>
			<b>Printing</b>	<b>188.96</b>
		<b>Athearn</b>		<b>5,458.23</b>
		<b>Atlas</b>		<b>167.61</b>
		<b>Con-Cor</b>		<b>15,609.34</b>
		<b>CTex</b>		<b>1,023.98</b>
		<b>Detail Associates</b>		<b>104.10</b>
		<b>El Mar Corporation</b>		<b>257.00</b>
		<b>El Mar Plastics</b>		<b>1,757.00</b>
		<b>Life-Like</b>		<b>500.93</b>
		<b>Mantua</b>		<b>277.76</b>
		<b>MDC</b>		<b>4,164.83</b>
		<b>Potter Manufacturin</b>		<b>1,115.43</b>
		<b>ProLitho</b>		<b>8,630.74</b>
02/09/99	1957	Telon Productions	Purchases	1,901.09
04/27/99	2069	Telon Productions	Purchases	965.73
07/03/99	2176	Telon Productions	Purchases	1,330.00
08/13/99	2229	Telon Productions	Purchases	950.00
		<b>Telon Productions</b>		<b>5,146.82</b>
		<b>Walthers</b>		<b>22,003.40</b>
			<b>Purchases</b>	<b>66,217.17</b>
			<b>Rent</b>	<b>35,026.34</b>
			<b>Shipping</b>	<b>1,928.20</b>
			<b>Taxes</b>	<b>5,152.79</b>
			<b>Telephone</b>	<b>5,690.50</b>
			<b>Utilities</b>	<b>2,054.89</b>
				<b>183,651.22</b>

36 visible/563 total

The **Expand All** tool expands a single summary record all the way to the raw data. It shows you all of the data that went into calculating that summary. In our checkbook example, using **Expand All** on **Shipping** would reveal every check written for shipping, no matter what carrier was used.

Date	CkNum	PayTo	Category	Debit
			<b>Purchases</b>	<b>66,217.17</b>
			<b>Rent</b>	<b>35,026.34</b>
02/09/99	1962	Airborne Express	Shipping	35.40
		<b>Airborne Express</b>		<b>35.40</b>
03/26/99	2018	AIRS	Shipping	138.07
		<b>AIRS</b>		<b>138.07</b>
02/09/99	1971	American Customs House	Shipping	34.00
		<b>American Customs H</b>		<b>34.00</b>
07/16/99	2191	B J D Trucking Inc.	Shipping	37.50
		<b>B J D Trucking Inc.</b>		<b>37.50</b>
09/19/99	2277	Burlington Air Express	Shipping	95.17
		<b>Burlington Air Expre</b>		<b>95.17</b>
03/16/99	2005	Consolidated Freight	Shipping	150.20
		<b>Consolidated Freight</b>		<b>150.20</b>
01/30/99	1929	Federal Express	Shipping	178.75
03/20/99	2015	Federal Express	Shipping	170.00
03/28/99	2032	Federal Express	Shipping	150.00
05/14/99	2101	Federal Express	Shipping	170.00
		<b>Federal Express</b>		<b>668.75</b>
01/31/99	1930	U P S	Shipping	52.97
02/09/99	1946	U P S	Shipping	122.60
03/20/99	2014	U P S	Shipping	25.00
03/28/99	2034	U P S	Shipping	197.42
04/17/99	2056	U P S	Shipping	25.00
05/04/99	2079	U P S	Shipping	25.00
05/08/99	2088	U P S	Shipping	25.00
07/24/99	2207	U P S	Shipping	45.80
08/21/99	2250	U P S	Shipping	155.00
09/19/99	2281	U P S	Shipping	95.32
		<b>U P S</b>		<b>769.11</b>
			<b>Shipping</b>	<b>1,928.20</b>
			<b>Taxes</b>	<b>5,152.79</b>
			<b>Telephone</b>	<b>5,690.50</b>
			<b>Utilities</b>	<b>2,054.89</b>
				<b>183,651.22</b>

46 visible/563 total

The **Collapse** tool hides the detail associated with a specific summary record. For example, if you collapsed **Federal Express**, the four checks written to Federal Express would disappear.



Date	CkNum	PayTo	Category	Debit
			<b>Purchases</b>	<b>66,217.17</b>
			<b>Rent</b>	<b>35,026.34</b>
02/09/99	1962	Airborne Express	Shipping	35.40
		<b>Airborne Express</b>		<b>35.40</b>
03/26/99	2018	AIRS	Shipping	138.07
		<b>AIRS</b>		<b>138.07</b>
02/09/99	1971	American Customs House	Shipping	34.00
		<b>American Customs H</b>		<b>34.00</b>
07/16/99	2191	B J D Trucking Inc.	Shipping	37.50
		<b>B J D Trucking Inc.</b>		<b>37.50</b>
09/19/99	2277	Burlington Air Express	Shipping	95.17
		<b>Burlington Air Expre</b>		<b>95.17</b>
03/16/99	2005	Consolidated Freight	Shipping	150.20
		<b>Consolidated Freight</b>		<b>150.20</b>
		<b>Federal Express</b>		<b>668.75</b>
01/31/99	1930	UP S	Shipping	52.97
02/09/99	1946	UP S	Shipping	122.60
03/20/99	2014	UP S	Shipping	25.00
03/28/99	2034	UP S	Shipping	197.42
04/17/99	2056	UP S	Shipping	25.00
05/04/99	2079	UP S	Shipping	25.00
05/08/99	2088	UP S	Shipping	25.00
07/24/99	2207	UP S	Shipping	45.80
08/21/99	2250	UP S	Shipping	155.00
09/19/99	2281	UP S	Shipping	95.32
		<b>UP S</b>		<b>769.11</b>
			<b>Shipping</b>	<b>1,928.20</b>
			<b>Taxes</b>	<b>5,152.79</b>
			<b>Telephone</b>	<b>5,690.50</b>
			<b>Utilities</b>	<b>2,054.89</b>
				<b>183,651.22</b>

### Sorting by Summary Value

Once summary information has been calculated, you can sort the database so that the data is ranked in order by the summary values. For example, the information in an invoice database can be summarized by customer and then rearranged to rank each customer by sales volume. Once the data is ranked you can see who your top customers are at a glance. The ability to rank summary information is a unique feature of Panorama.

The first step in sorting summary information is to calculate the summary information. To do this use the group (see "[STEP 1 - GROUP](#)" on page 189) and summary calculation (see "[STEP 2 - CALCULATE](#)" on page 192) commands already covered in this chapter.

The second step is to collapse the outline. Use the **Outline Level** command to collapse the database to the level you want to sort (see "[The Outline Level](#)" on page 194).

Once the database is collapsed, the final step is to sort the database by summary value. Click on the field containing the summary values and then choose **Sort Up** or **Sort Down**. **Sort Down** is the usual choice because it ranks the summaries from highest to lowest.

To illustrate this technique, we started by grouping and totalling the checkbook database by category.

Date	CkNum	PayTo	Category	Debit
			<b>Advertising</b>	<b>34,516.82</b>
			<b>Auto</b>	<b>555.04</b>
			<b>Equipment Rent:</b>	<b>632.01</b>
			<b>Fixed Assets</b>	<b>9,774.47</b>
			<b>Insurance</b>	<b>8,234.53</b>
			<b>Legal Fees</b>	<b>1,288.07</b>
			<b>Maintenance</b>	<b>3,673.99</b>
			<b>Office Supplies</b>	<b>7,261.09</b>
			<b>Postage</b>	<b>1,456.35</b>
			<b>Printing</b>	<b>188.96</b>
			<b>Purchases</b>	<b>66,217.17</b>
			<b>Rent</b>	<b>35,026.34</b>
			<b>Shipping</b>	<b>1,928.20</b>
			<b>Taxes</b>	<b>5,152.79</b>
			<b>Telephone</b>	<b>5,690.50</b>
			<b>Utilities</b>	<b>2,054.89</b>
				<b>183,651.22</b>

18 visible/429 total

As you can see, the subtotals are listed in alphabetical order. Use the **Sort Down** command to sort the subtotals so that they are listed in order from highest to lowest amount.

Date	CkNum	PayTo	Category	Debit
			<b>Purchases</b>	<b>66,217.17</b>
			<b>Rent</b>	<b>35,026.34</b>
			<b>Advertising</b>	<b>34,516.82</b>
			<b>Fixed Assets</b>	<b>9,774.47</b>
			<b>Insurance</b>	<b>8,234.53</b>
			<b>Office Supplies</b>	<b>7,261.09</b>
			<b>Telephone</b>	<b>5,690.50</b>
			<b>Taxes</b>	<b>5,152.79</b>
			<b>Maintenance</b>	<b>3,673.99</b>
			<b>Utilities</b>	<b>2,054.89</b>
			<b>Shipping</b>	<b>1,928.20</b>
			<b>Postage</b>	<b>1,456.35</b>
			<b>Legal Fees</b>	<b>1,288.07</b>
			<b>Equipment Rent:</b>	<b>632.01</b>
			<b>Auto</b>	<b>555.04</b>
			<b>Printing</b>	<b>188.96</b>
				<b>0.00</b>
				<b>183,651.22</b>

18 visible/429 total

After the database has been sorted, you can re-expand some or all of the outline if you wish. The detail information for each summary follows the summary as it is sorted. (In other words, when the database is grouped the sort commands sort the groups, not the individual records.)

Date	CkNum	PayTo	Category	Debit
			<b>Purchases</b>	<b>66,217.17</b>
			<b>Rent</b>	<b>35,026.34</b>
			<b>Advertising</b>	<b>34,516.82</b>
05/24/99	2141	Pitney Bowes	Equipment Rental	79.69
08/21/99	2253	Pitney Bowes	Equipment Rental	79.69
01/08/99	1907	Northern Illinois Mold	Equipment Rental	96.05
02/09/99	1950	Pitney Bowes	Equipment Rental	73.14
04/23/99	2063	Pitney Bowes	Equipment Rental	79.69
05/24/99	2137	Pitney Bowes	Equipment Rental	25.75
09/19/99	2280	GECC	Fixed Assets	352.00
09/18/99	2275	T.W. Bender Group	Fixed Assets	2,814.33
09/26/99	2296	TesLabe	Fixed Assets	2,465.00
08/21/99	2243	GECC	Fixed Assets	725.00
07/18/99	2200	SSG LaserWorks	Fixed Assets	793.00
			<b>Fixed Assets</b>	<b>9,774.47</b>
			<b>Insurance</b>	<b>8,234.53</b>
			<b>Office Supplies</b>	<b>7,261.09</b>
			<b>Telephone</b>	<b>5,690.50</b>
			<b>Taxes</b>	<b>5,152.79</b>

29 visible/429 total



The real power of this technique appears when the database is grouped by multiple levels. For example, suppose the checkbook database is grouped by month and by category, then collapsed to show the monthly totals for each category. This is shown in the window on the left. On the right is the same database, but now the **Sort Down** command has been used to sort the subtotals. The subtotals have been sorted within each month. Now we can easily see that **Purchases** was the top spending category in January, March, and April, but **Advertising** was tops in February (with **Purchases** dropping to number 3).

*before Sort Down...*

Date	Category	Debit
01/31/99	Advertising	2,841.02
	Equipment Rent:	96.05
	Insurance	761.63
	Legal Fees	223.52
	Office Supplies	426.93
	Postage	150.00
	Purchases	17,083.42
	Rent	4,070.83
	Shipping	231.72
	Taxes	549.00
	Telephone	141.09
	<b>26,575.21</b>	
02/28/99	Advertising	8,134.13
	Auto	240.21
	Equipment Rent:	73.14
	Fixed Assets	428.39
	Insurance	601.48
	Legal Fees	95.00
	Maintenance	310.00
	Office Supplies	915.50
	Postage	315.00
	Purchases	3,386.78
	Rent	7,742.19
Shipping	192.00	
Telephone	736.66	
Utilities	402.78	
	<b>23,573.26</b>	
03/30/99	Advertising	7,501.90
	Auto	33.32
	Insurance	1,539.14
	Maintenance	872.25
	Office Supplies	1,579.74
	Postage	110.00
	Purchases	19,557.48
	Rent	3,566.30
	Shipping	830.69
	Taxes	2,008.98
	Telephone	1,027.01
Utilities	384.49	
	<b>39,011.30</b>	
04/27/99	Advertising	1,024.52
	Equipment Rent:	79.69
	Insurance	522.63
	Maintenance	132.00
	Office Supplies	339.46
	Postage	156.35
	Purchases	2,268.19
	Rent	265.00
	Shipping	25.00
	Taxes	734.33
	Telephone	305.56
	<b>5,852.73</b>	

*after Sort Down*

Date	Category	Debit
01/31/99	Purchases	17,083.42
	Rent	4,070.83
	Advertising	2,841.02
	Insurance	761.63
	Taxes	549.00
	Office Supplies	426.93
	Shipping	231.72
	Legal Fees	223.52
	Postage	150.00
	Telephone	141.09
	Equipment Rent:	96.05
	<b>26,575.21</b>	
02/28/99	Advertising	8,134.13
	Rent	7,742.19
	Purchases	3,386.78
	Office Supplies	915.50
	Telephone	736.66
	Insurance	601.48
	Fixed Assets	428.39
	Utilities	402.78
	Postage	315.00
	Maintenance	310.00
	Auto	240.21
Shipping	192.00	
Legal Fees	95.00	
Equipment Rent:	73.14	
	<b>23,573.26</b>	
03/30/99	Purchases	19,557.48
	Advertising	7,501.90
	Rent	3,566.30
	Taxes	2,008.98
	Office Supplies	1,579.74
	Insurance	1,539.14
	Telephone	1,027.01
	Maintenance	872.25
	Shipping	830.69
	Utilities	384.49
	Postage	110.00
Auto	33.32	
	<b>39,011.30</b>	
04/27/99	Purchases	2,268.19
	Advertising	1,024.52
	Taxes	734.33
	Insurance	522.63
	Office Supplies	339.46
	Telephone	305.56
	Rent	265.00
	Postage	156.35
	Maintenance	132.00
	Equipment Rent:	79.69
	Shipping	25.00
	<b>5,852.73</b>	

This technique is very powerful any time you need to rank summary information. You can quickly answer questions like “Who are our top customers?” “What products had the most service problems last year?” or “Which SKU’s are the best sellers in different seasons of the year?”

## Sorting Within Groups

If you sort your database without collapsing it, Panorama will sort the data within each group instead of sorting the entire database. If you want to sort the summary values themselves, you must use the **Outline Level** dialog to collapse the outline, as described in the previous section.

## Getting Rid of Summary Records

When you are done with summary records, you can get rid of them with the **Remove Summaries** dialog (Sort menu). This command can either remove all the summaries, or it can selectively remove certain summary levels.

The **Remove Summaries** dialog displays a diagram of the database outline structure. The diagram shows each summary level, how it was created, and the number of records in the level. If you want to get rid of all the summary records, press the button with the highest number or the **Remove All Summaries** button.



If you press one of the lower numbered buttons, only some of the summaries will be removed. For example if you press **1**, only the lowest summary level will be removed (the smallest subgroups). The remaining summary records will each drop down to the next lowest level.

You can also remove individual summary records with the **Cut Record** tool or the **Delete** or **Backspace** key (see “[Deleting a Record](#)” on page 273).

## Getting Rid of Detail

Occasionally you may want to completely remove the raw data, leaving only the summary information. The **Remove Detail** dialog (Sort menu) does this for you. You can remove just the data, or you can remove the data along with some of the lower levels of summary information. **Warning:** This command is extremely dangerous, since it effectively deletes most of your database. Handle with care!

The **Remove Detail** dialog displays a diagram of the database outline structure. The diagram shows each summary level, how it was created, and the number of records in the level. Press one of the buttons on the left to choose the level of detail to remove. If you press the **Data** button, only the raw data will be removed. The level 1 summary records will be converted into data records, and all other summary records will drop down to the next lower level.



If you press one of the numbered buttons (**1** through **6**) Panorama will remove the summary records up to and including that level as well as the data records. The remaining summary records will be adjusted so that the lowest remaining summary level becomes the new data records, etc.

### Printing Reports with Summary Information

You'll often want to print reports with the summary information you have generated. In Panorama reports are printed using forms and report tiles (see "[Custom Reports](#)" on page 445). In addition to the standard report tools Panorama also has special features for printing summary information. You can print special headers and footers for each group, and can control how groups break across columns and pages. See "[Printing Summary Information](#)" on page 483 to learn more about how to use these features.



# Chapter 11: Crosstabs



This chapter describes Panorama's most powerful tool for analyzing and summarizing data—**crosstabs**. A crosstab is simply a table with categories across the top and down the left, with numbers in the middle and totals across the bottom and down the right.

xtab	Jan 99	Feb 99	Mar 99	Apr 99	May 99	Jun 99	TOTAL
Advertising	2,841.02	8,134.13	7,501.90	1,024.52	7,541.18	828.00	34,516.82
Auto		240.21	33.32		119.05		555.04
Equipment Rental	96.05	73.14		79.69	105.44		632.01
Fixed Assets		428.39			778.00	1,168.75	9,774.47
Insurance	761.63	601.48	1,539.14	522.63	1,220.45	461.99	8,234.53
Legal Fees	223.52	95.00			799.55	15.00	1,288.07
Maintenance		310.00	872.25	132.00	1,012.63	368.38	3,673.99
Office Supplies	426.93	915.50	1,579.74	339.46	1,265.34	281.52	7,261.09
Postage	150.00	315.00	110.00	156.35	115.00		1,456.35
Printing					96.68		188.96
Purchases	17,083.42	3,386.78	19,557.48	2,268.19	4,796.82	5,172.74	66,217.17
Rent	4,070.83	7,742.19	3,566.30	265.00	7,630.56	3,874.00	35,026.34
Shipping	231.72	192.00	830.69	25.00	220.00		1,928.20
Taxes	549.00		2,008.98	734.33	513.51	155.76	5,152.79
Telephone	141.09	736.66	1,027.01	305.56	915.96	202.31	5,690.50
Utilities		402.78	384.49		529.76	213.58	2,054.89
<b>*TOTAL</b>	<b>26,575.21</b>	<b>23,573.26</b>	<b>39,011.30</b>	<b>5,852.73</b>	<b>27,659.93</b>	<b>12,742.03</b>	<b>183,651.22</b>

The word crosstab is short for cross tabulation, referring to the criss-cross way that totals are tabulated both across and down. Probably the most common example of a crosstab is a budget, with months or years across the top, and spending categories down the left.

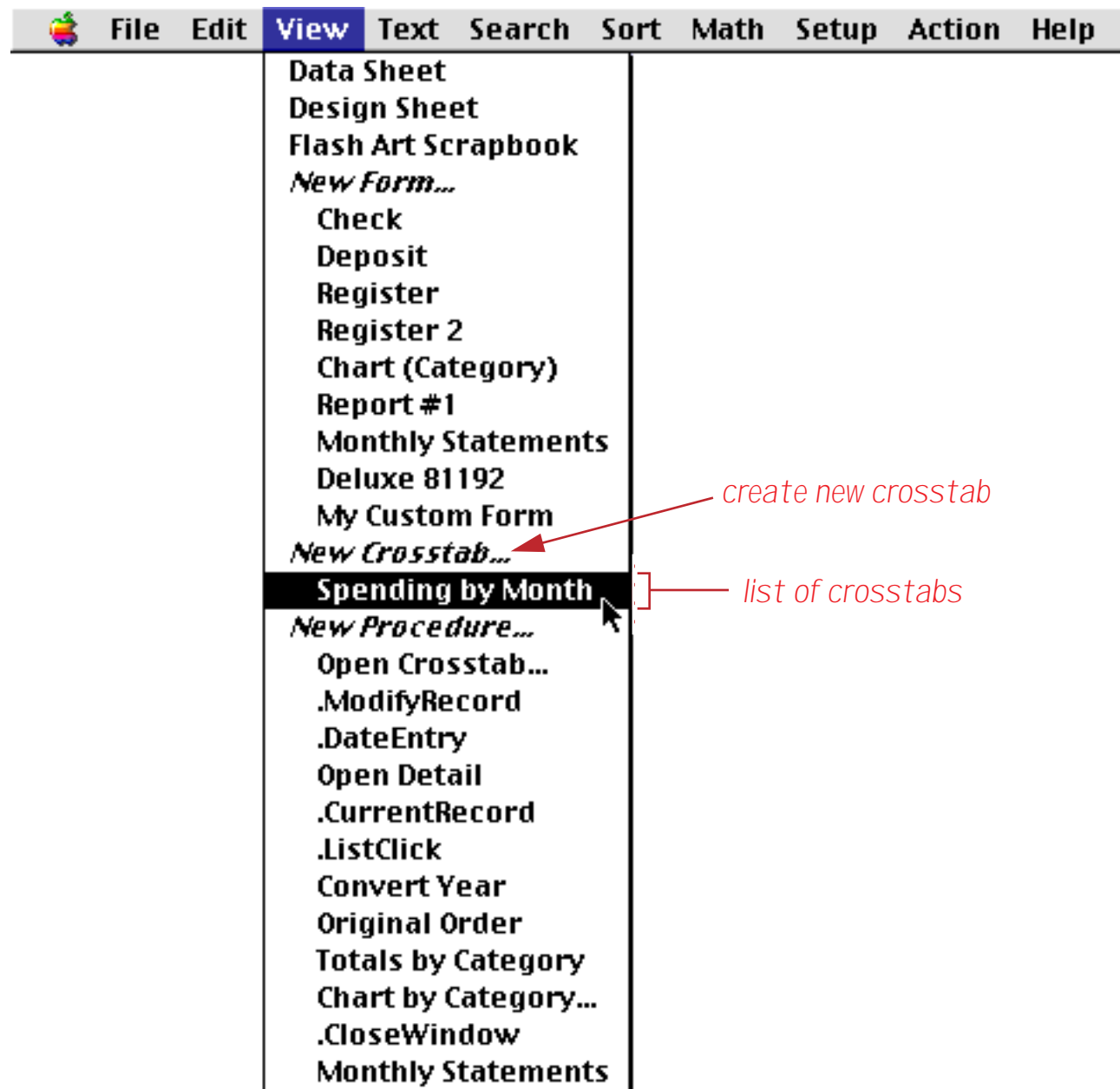
Before Panorama became available, crosstabs were usually created using a spreadsheet. Spreadsheets are perfect for totalling the rows and columns in the crosstab table. Unfortunately, spreadsheets cannot help with the really tedious part of creating a crosstab table—taking the raw data, categorizing it, and converting it into the crosstab table format. With a spreadsheet, this tedious number crunching must be done by hand.

Panorama automates the entire crosstab process from start to finish. Starting with raw data (a checkbook database, for example), Panorama divides the data into categories and automatically creates and calculates the entire crosstab table. When the raw data changes, the entire process can be repeated with a single mouse click. A simple dialog sets up the whole process.

Panorama can also work a crosstab backwards, allowing you to locate the raw data associated with any crosstab value. For example, if the crosstab table shows that July's advertising expenditures seem a bit high, simply click on that value and press the **Select Original Data** tool. The individual data records for July advertising will appear.



Panorama does not limit you to one crosstab table per database. Each crosstab table appears in its own window, and you can have as many different crosstabs as you need. Crosstab tables are created and opened with the **View** menu.



Although each crosstab table gets its raw data from the main database, it is otherwise independent. Setting up and calculating a crosstab table does not change the main database in any way.

## Category and Tabulation Fields

A crosstab is based on three fields in the main database. Two of these fields are called **category fields**, and the third is called the **tabulation field**. The two category fields are the fields that criss-cross across the top and left sides of the crosstab table. The tabulation field holds the raw data that is counted or totalled in the center of the crosstab table. In the example crosstab shown below, **Date** and **Category** are the category fields while **Debit** is the tabulation field.

**Checkbook**

Date	CkNum	PayTo	Category	Debit
01/22/99	1916	Walthers	Purchases	12,463.00
01/22/99	1917	Blue Cross Of Calif	Insurance	279.03
01/22/99	1918	Sherman Douglas Ins	Insurance	418.60
01/22/99	1919	Cannon Astro	Office Supplies	145.72
01/25/99	1920	Walthers	Purchases	1,885.40
01/25/99	1921	Nebs	Office Supplies	77.27
01/25/99	1922	Ramona Drinking Water	Office Supplies	98.10
01/25/99	1923	Pacific Partners	Rent	4,070.83
01/29/99	1924	Athearn	Purchases	1,906.32
01/29/99	1925	Advertiser's Mailing Ser	Advertising	860.22
01/29/99	1926	PacTel Cellular	Telephone	141.09
01/30/99	1927	State Board Of Equalizat	Taxes	549.00

4 1 visible/411 total

**Checkbook:XTABS:Spending by Month**

xtab	Jan 99	Feb 99	Mar 99	Apr 99	May 99	Jun 99	TOTAL
Advertising	2,841.02	8,134.13	7,501.90	1,024.52	7,541.18	828.00	34,516.82
Auto		240.21	33.32		119.05		555.04
Equipment Rental	96.05	73.14		79.69	105.44		632.01
Fixed Assets		428.39			778.00	1,168.75	9,774.47
Insurance	761.63	601.48	1,539.14	522.63	1,220.45	461.99	8,234.53
Legal Fees	223.52	95.00			799.55	15.00	1,288.07
Maintenance		310.00	872.25	132.00	1,012.63	368.38	3,673.99
Office Supplies	426.93	915.50	1,579.74	339.46	1,265.34	281.52	7,261.09
Postage	150.00	315.00	110.00	156.35	115.00		1,456.35
Printing					96.68		188.96
Purchases	17,083.42	3,386.78	19,557.48	2,268.19	4,796.82	5,172.74	66,217.17
Rent	4,070.83	7,742.19	3,566.30	265.00	7,630.56	3,874.00	35,026.34
Shipping	231.72	192.00	830.69	25.00	220.00		1,928.20
Taxes	549.00		2,008.98	734.33	513.51	155.76	5,152.79
Telephone	141.09	736.66	1,027.01	305.56	915.96	202.31	5,690.50
Utilities		402.78	384.49		529.76	213.58	2,054.89
<b>+TOTAL</b>	<b>26,575.21</b>	<b>23,573.26</b>	<b>39,011.30</b>	<b>5,852.73</b>	<b>27,659.93</b>	<b>12,742.03</b>	<b>183,651.2:</b>

17 visible/17 total

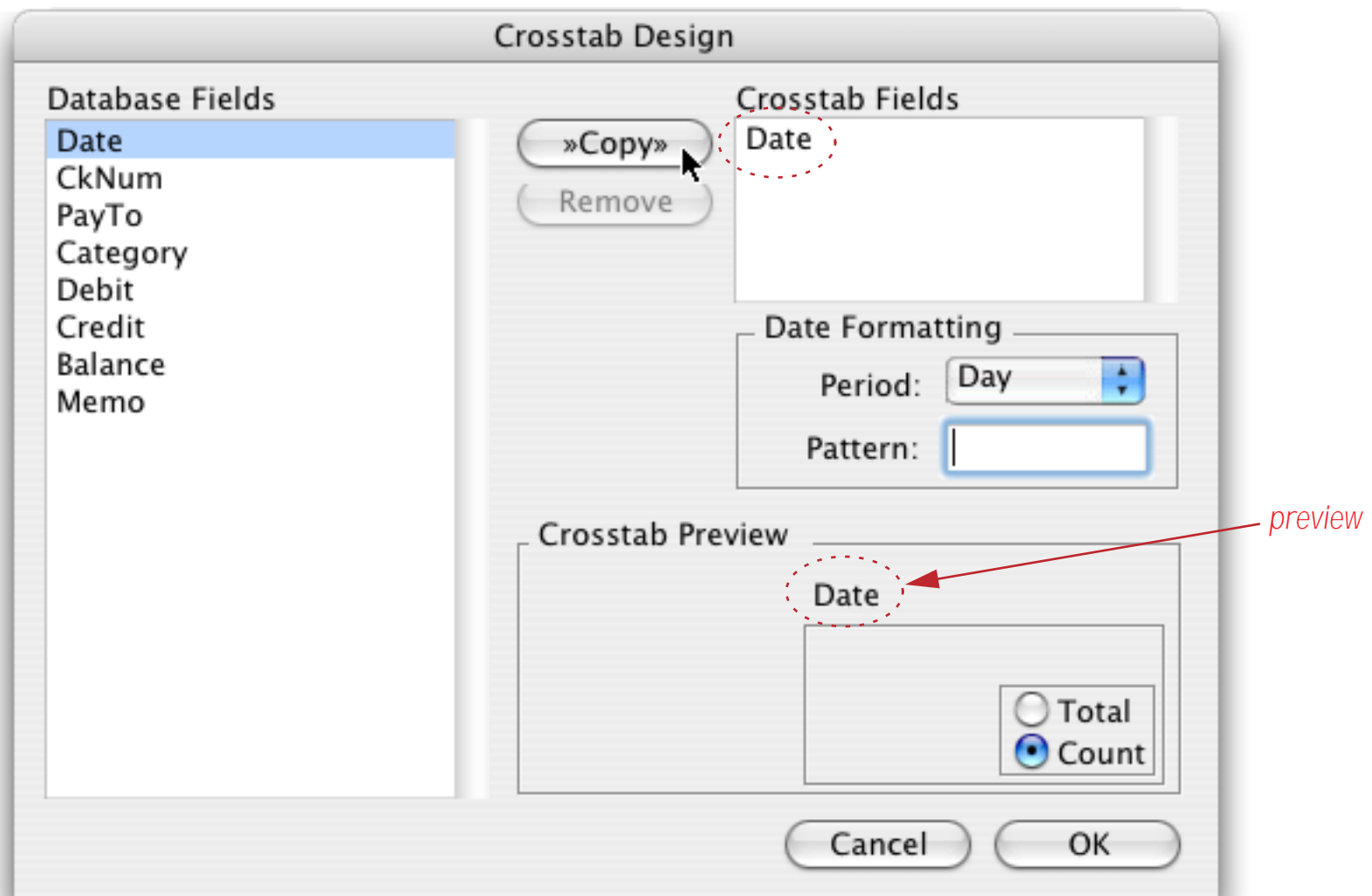
## Creating and Setting Up a New Crosstab View

New crosstab views are created using the View menu. Choose **New Crosstab** from the menu. Then you must give the new crosstab view a name (up to 25 characters) and press **Ok** to create the new view.

When a new crosstab view is created, the **Crosstab Design** dialog box automatically appears. This dialog allows you to specify the category and tabulation fields (see previous section) and to specify what type of calculation (total or count) to use.

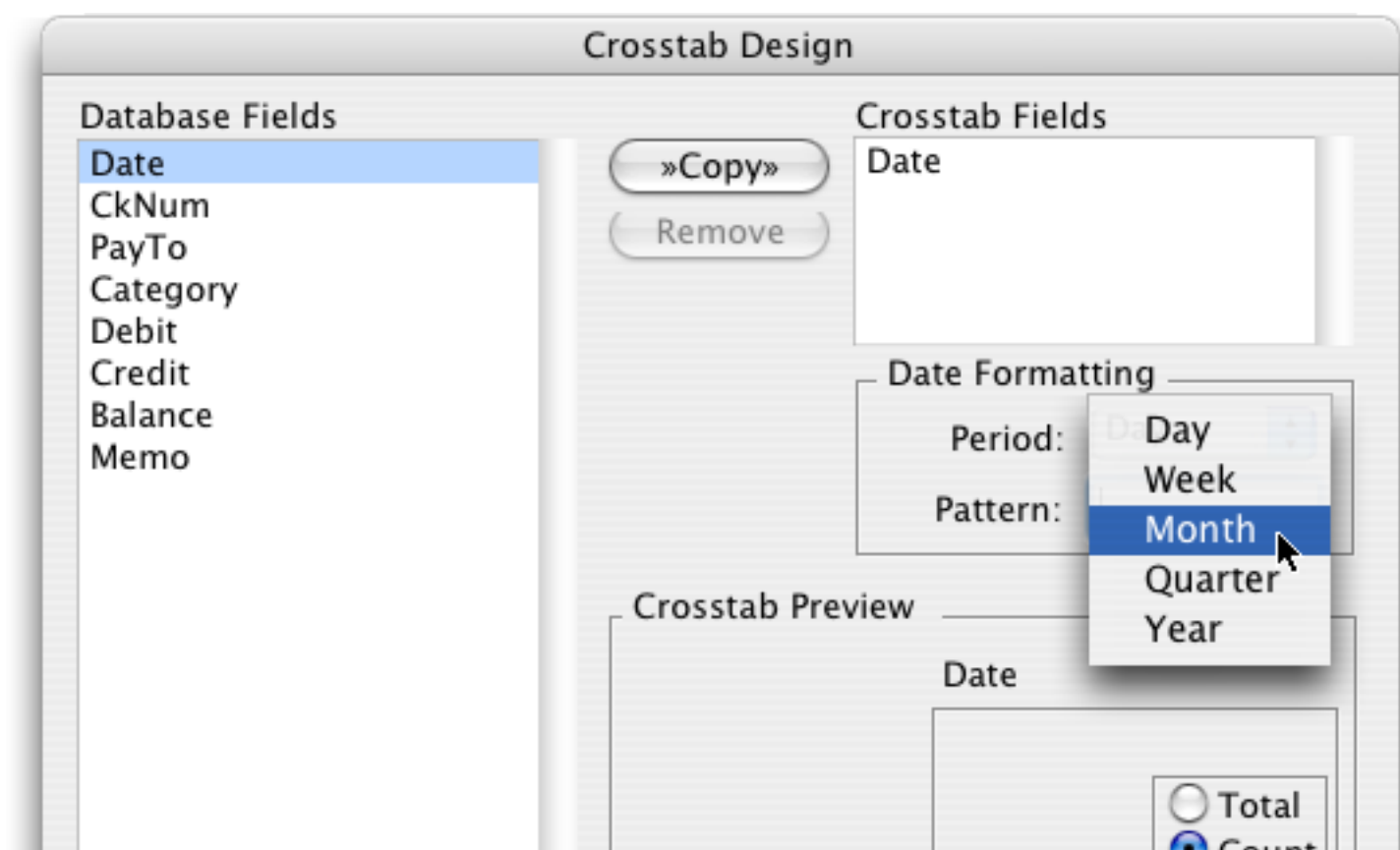
To use the Crosstab Design dialog you copy fields from the list on the left over to the list on the right. The left side lists all the fields in the database. The right side lists the fields in the crosstab. The first two fields copied to the right become the two category fields. The third field becomes the tabulation field. As you build the crosstab, a miniature schematic diagram of the crosstab appears in the lower right hand corner of the dialog.

Let's walk through the creation of a crosstab like the one shown at the beginning of this chapter. First, click on the **Date** field and copy it to the right. (Hint: To copy the field either press the **»Copy»** button or double click on the field name.)

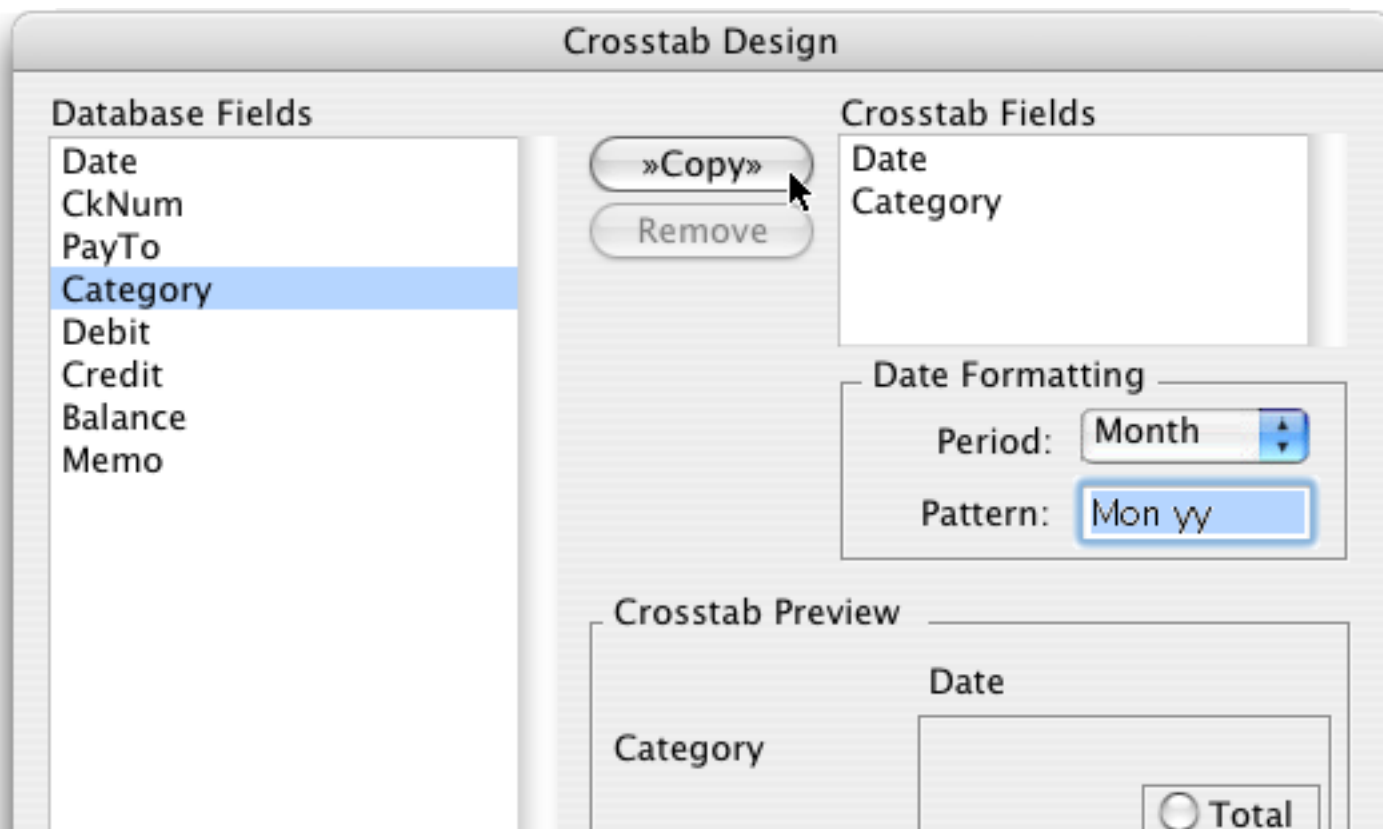


As you can see, the **Date** field now appears in the list of Crosstab fields. It also appears in the preview of the final crosstab, showing that the **Date** will appear across the top of the crosstab.

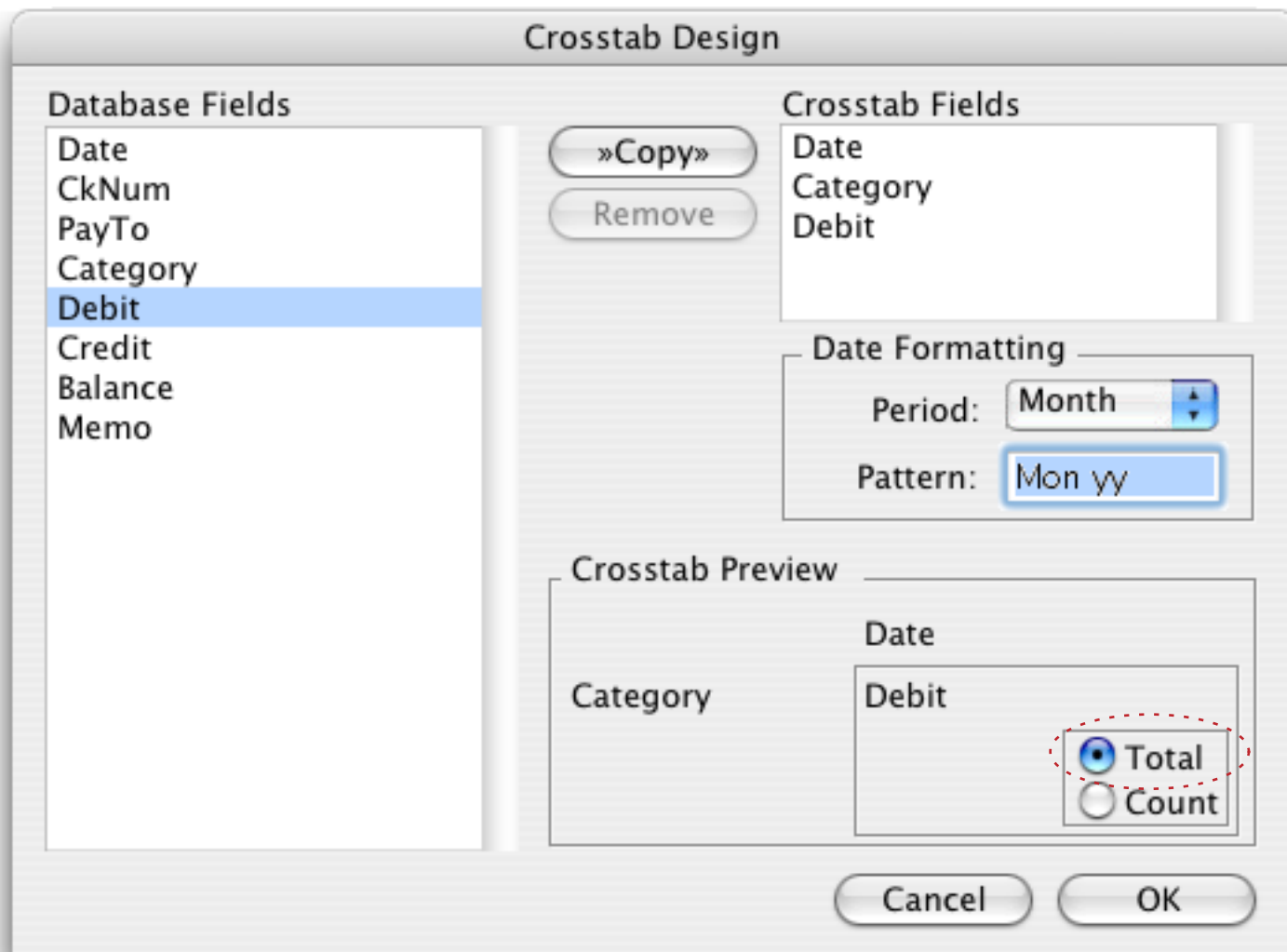
Since we want the date to be grouped by month, select **Month** from the **Date Period** pop-up menu. (If you forget to do this now you can always go back and change it later.)



Next, copy the **Category** field into the crosstab. It will appear in the list and on the left side of the mini-diagram, showing that the category will appear on the left side of the crosstab.

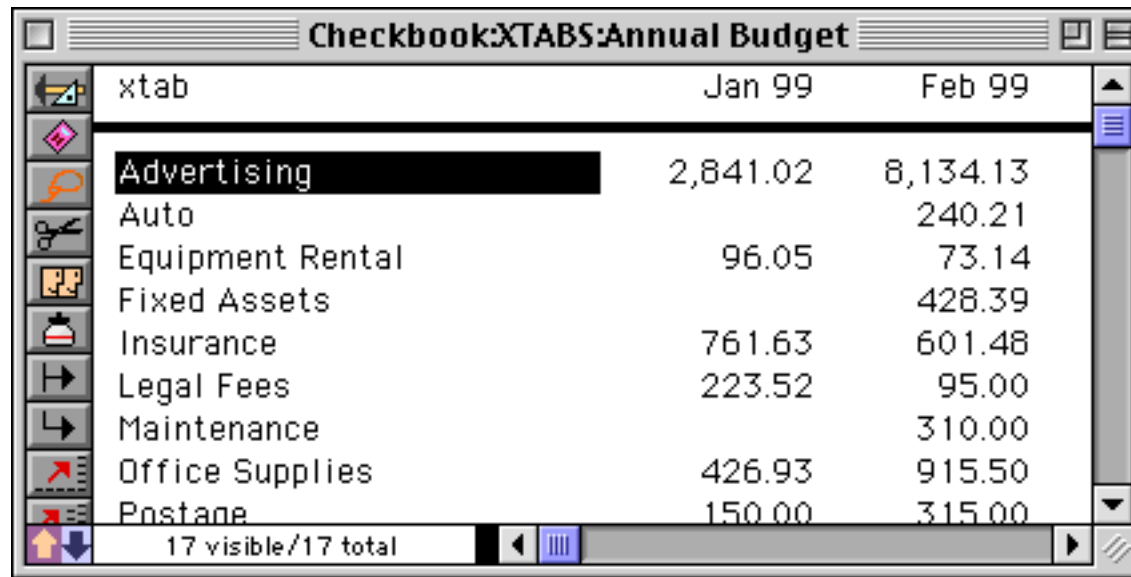


Now copy the **Debit** field into the crosstab. You'll also need to click on the **Total** radio button, since we want to calculate sums of the checks, not counts of the checks.





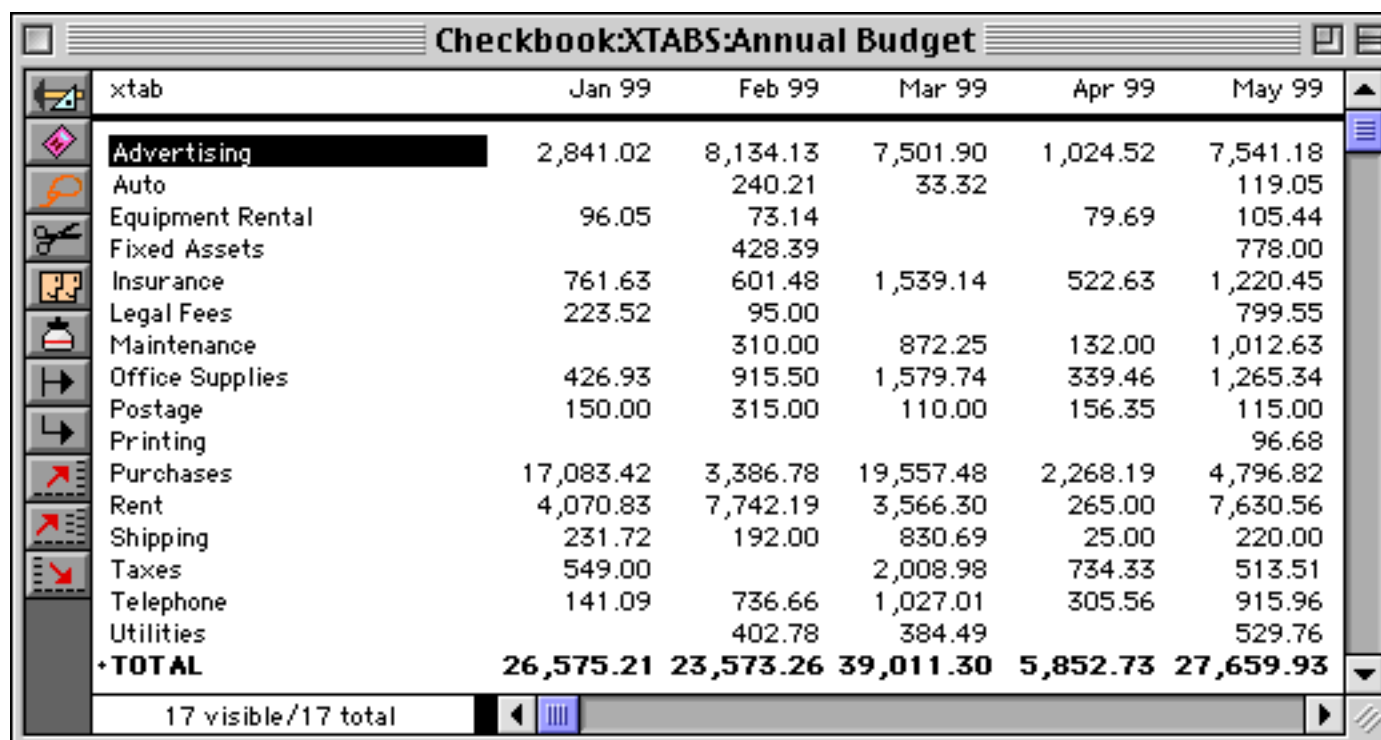
Once you've specified the category and tabulation fields, press **Ok** to actually calculate the crosstab. Depending on the complexity of the database, there may be a delay of seconds or even minutes as the crosstab is calculated. When the calculations are finished, the new crosstab table will appear. It will look something like this.



xtab	Jan 99	Feb 99
Advertising	2,841.02	8,134.13
Auto		240.21
Equipment Rental	96.05	73.14
Fixed Assets		428.39
Insurance	761.63	601.48
Legal Fees	223.52	95.00
Maintenance		310.00
Office Supplies	426.93	915.50
Postage	150.00	315.00

17 visible/17 total

You can now adjust the window size, font, and column widths as you like.

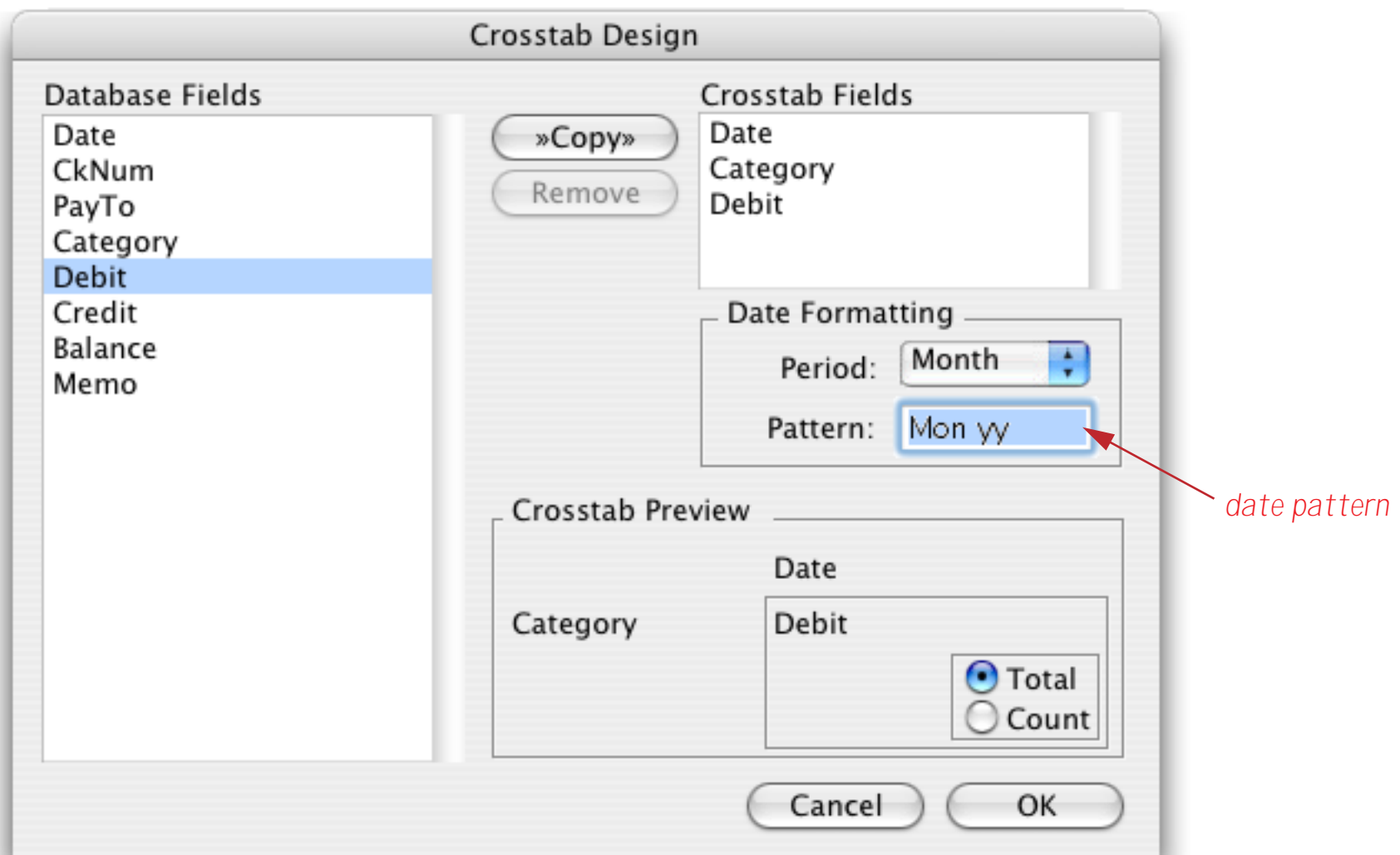


xtab	Jan 99	Feb 99	Mar 99	Apr 99	May 99
Advertising	2,841.02	8,134.13	7,501.90	1,024.52	7,541.18
Auto		240.21	33.32		119.05
Equipment Rental	96.05	73.14		79.69	105.44
Fixed Assets		428.39			778.00
Insurance	761.63	601.48	1,539.14	522.63	1,220.45
Legal Fees	223.52	95.00			799.55
Maintenance		310.00	872.25	132.00	1,012.63
Office Supplies	426.93	915.50	1,579.74	339.46	1,265.34
Postage	150.00	315.00	110.00	156.35	115.00
Printing					96.68
Purchases	17,083.42	3,386.78	19,557.48	2,268.19	4,796.82
Rent	4,070.83	7,742.19	3,566.30	265.00	7,630.56
Shipping	231.72	192.00	830.69	25.00	220.00
Taxes	549.00		2,008.98	734.33	513.51
Telephone	141.09	736.66	1,027.01	305.56	915.96
Utilities		402.78	384.49		529.76
<b>+TOTAL</b>	<b>26,575.21</b>	<b>23,573.26</b>	<b>39,011.30</b>	<b>5,852.73</b>	<b>27,659.93</b>

17 visible/17 total

## Crosstabs by Day, Month, Quarter or Year

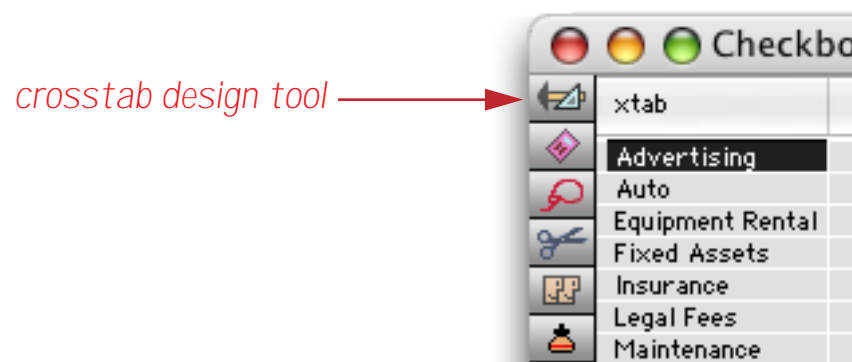
If one of the category fields contains dates, you must tell Panorama what period to group by. The **Date Period** pop-up menu has five choices: **day**, **week**, **month**, **quarter**, and **year**. You can also specify a pattern for displaying the date in the crosstab. For example, months can be displayed as **1-04**, **Jan-04**, or **January 2004**.



Panorama supplies a default pattern when you choose from the **Date Period** pop-up menu. You can use this predefined pattern, or you can type in any pattern you want. See [“Date Output Patterns”](#) on page 261 for more information about date patterns.

## Changing the Crosstab Design

The crosstab design can be changed at any time by pressing the **Crosstab Design** tool. This brings up the same dialog box you used to originally set up the crosstab.

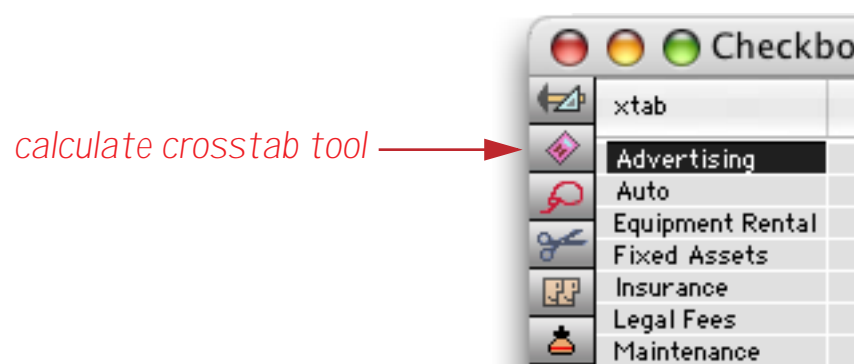


To erase the entire crosstab design and start over, select all the fields in the CrossTab Fields list and press the **Remove** button. (You can select all the fields by dragging the mouse over the list.)

If you want to change just one field, select both the old field from the CrossTab Fields list and the new field from the Database Fields list. Then press the **»Copy»** button to change the field.

## Re-Calculating a Crosstab

Crosstabs do not automatically update when the main database changes. This is because of the time it takes to recalculate the crosstab. If you change the main database and want to re-calculate a crosstab, press the **Calculate Crosstab** tool.



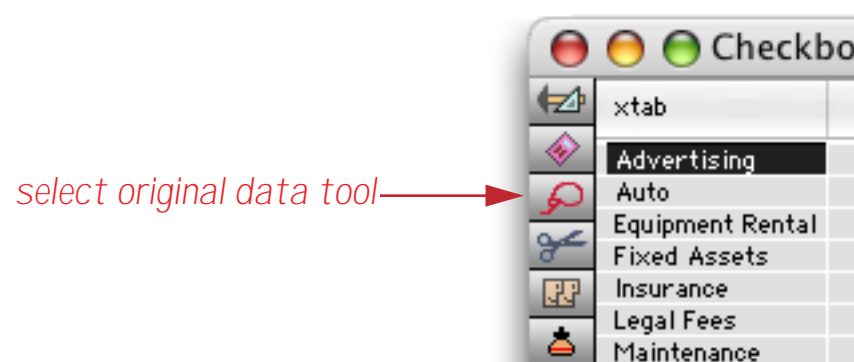
## Adjusting Crosstab Column Widths

When a new crosstab is created, Panorama tries to assign an appropriate width for each column. You can adjust these column widths the same way you would adjust the column widths in the data sheet. Move the cursor over the column titles and drag left or right to adjust the width.

Whenever the crosstab is recalculated, Panorama automatically resets the width of every column except for the first two. The third, fourth, fifth and all additional columns are all set to the same width as the second column.

## Selecting Original Data

Using the **Select Original Data** tool, Panorama can locate the raw data behind any value in the crosstab table. To do this you must have a regular database window open in addition to the crosstab window, usually the data sheet window.



To select the original data, first click on the crosstab value you are interested in. Then click on the **Select Original Data** tool, The original data is selected and will appear in the data sheet or other database window. For example, you could click on the **Mar 99 Office Supplies** cell, then choose **Select Original Data**.

xtab	Jan 99	Feb 99	Mar 99	Apr 99	May 99	Jun 99
	0.00	0.00	0.00	0.00	0.00	0.00
<b>Select Original Data</b>	1.02	8,134.13	7,501.90	1,024.52	7,541.18	828.00
Auto		240.21	33.32		119.05	
Equipment Rental	96.05	73.14		79.69	105.44	
Fixed Assets		428.39			778.00	1,168.75
Insurance	761.63	601.48	1,539.14	522.63	1,220.45	461.99
Legal Fees	223.52	95.00			799.55	15.00
Maintenance		310.00	872.25	132.00	1,012.63	368.38
Office Supplies	426.93	915.50	<b>1,579.74</b>	339.46	1,265.34	281.52
Postage	150.00	315.00	110.00	156.35	115.00	
Printing					96.68	
Purchases	17,083.42	3,386.78	19,557.48	2,268.19	4,796.82	5,172.74
Rent	4,070.83	7,742.19	3,566.30	265.00	7,630.56	3,874.00
Shipping	231.72	192.00	830.69	25.00	220.00	
Taxes	549.00		2,008.98	734.33	513.51	155.76
Telephone	141.09	736.66	1,027.01	305.56	915.96	202.31
Utilities		402.78	384.49		529.76	213.58
<b>*TOTAL</b>	<b>26,575.21</b>	<b>23,573.26</b>	<b>39,011.30</b>	<b>5,852.73</b>	<b>27,659.93</b>	<b>12,742.03</b>

In the data sheet the raw data backing up this crosstab cell will be selected - in this case the seven checks written for **Office Supplies** in March of 1999.

Date	CkNum	PayTo	Category	Debit
03/12/99	1999	Pitney Bowes	Office Supplies	53.94
03/12/99	2001	Paper Mart	Office Supplies	572.78
03/12/99	2003	Pace Club	Office Supplies	267.47
03/28/99	2028	Pitney Bowes	Office Supplies	247.00
03/28/99	2030	NEBS	Office Supplies	67.20
03/28/99	2031	Ramona Drinking Water	Office Supplies	51.55
03/29/99	2036	Jeffco	Office Supplies	319.80

You can select the original data for any value in the crosstab. If you click on a cell in the first or last column of the crosstab, the **Select Original Data** tool will select all the data associated with the entire row. For example, you could click on the **Fixed Assets** cell.

xtab	Jan 99	Feb 99	Mar 99	Apr 99	May 99	Jun 99
	0.00	0.00	0.00	0.00	0.00	0.00
<b>Select Original Data</b>	1.02	8,134.13	7,501.90	1,024.52	7,541.18	828.00
Auto		240.21	33.32		119.05	
Equipment Rental	96.05	73.14		79.69	105.44	
<b>Fixed Assets</b>		428.39			778.00	1,168.75
Insurance	761.63	601.48	1,539.14	522.63	1,220.45	461.99
Legal Fees	223.52	95.00			799.55	15.00
Maintenance		310.00	872.25	132.00	1,012.63	368.38
Office Supplies	426.93	915.50	1,579.74	339.46	1,265.34	281.52
Postage	150.00	315.00	110.00	156.35	115.00	
Printing					96.68	
Purchases	17,083.42	3,386.78	19,557.48	2,268.19	4,796.82	5,172.74
Rent	4,070.83	7,742.19	3,566.30	265.00	7,630.56	3,874.00
Shipping	231.72	192.00	830.69	25.00	220.00	
Taxes	549.00		2,008.98	734.33	513.51	155.76
Telephone	141.09	736.66	1,027.01	305.56	915.96	202.31
Utilities		402.78	384.49		529.76	213.58
<b>*TOTAL</b>	<b>26,575.21</b>	<b>23,573.26</b>	<b>39,011.30</b>	<b>5,852.73</b>	<b>27,659.93</b>	<b>12,742.03</b>

When you choose the **Select Original Data** tool, the data sheet will show all of the checks written for **Fixed Assets** in every month.

Date	CkNum	PayTo	Category	Debit
02/09/99	1952	GECC	Fixed Assets	428.39
05/02/99	2072	GECC	Fixed Assets	704.00
05/24/99	2112	GECC	Fixed Assets	74.00
06/14/99	2158	C M S	Fixed Assets	1,168.75
07/03/99	2175	GECC	Fixed Assets	250.00
07/18/99	2200	SSG LaserWorks	Fixed Assets	793.00
08/21/99	2243	GECC	Fixed Assets	725.00
09/18/99	2275	T.W. Bender Group	Fixed Assets	2,814.33
09/19/99	2280	GECC	Fixed Assets	352.00
09/26/99	2296	TesLabe	Fixed Assets	2,465.00



If you click on a cell in the bottom row of the crosstab, the **Select Original Data** tool will select all of the data for the entire column. For example, you could click on the total for [April 99](#).

xtab	Jan 99	Feb 99	Mar 99	Apr 99	May 99	Jun 99
	0.00	0.00	0.00	0.00	0.00	0.00
<b>Select Original Data</b>	11.02	8,134.13	7,501.90	1,024.52	7,541.18	828.00
Photo		240.21	33.32		119.05	
Equipment Rental	96.05	73.14		79.69	105.44	
Fixed Assets		428.39			778.00	1,168.75
Insurance	761.63	601.48	1,539.14	522.63	1,220.45	461.99
Legal Fees	223.52	95.00			799.55	15.00
Maintenance		310.00	872.25	132.00	1,012.63	368.38
Office Supplies	426.93	915.50	1,579.74	339.46	1,265.34	281.52
Postage	150.00	315.00	110.00	156.35	115.00	
Printing					96.68	
Purchases	17,083.42	3,386.78	19,557.48	2,268.19	4,796.82	5,172.74
Rent	4,070.83	7,742.19	3,566.30	265.00	7,630.56	3,874.00
Shipping	231.72	192.00	830.69	25.00	220.00	
Taxes	549.00		2,008.98	734.33	513.51	155.76
Telephone	141.09	736.66	1,027.01	305.56	915.96	202.31
Utilities		402.78	384.49		529.76	213.58
<b>*TOTAL</b>	<b>26,575.21</b>	<b>23,573.26</b>	<b>39,011.30</b>	<b>5,852.73</b>	<b>27,659.93</b>	<b>12,742.03</b>

When you choose the **Select Original Data** tool, the data sheet will show all of the checks written for [April 99](#) in every category.

Date	CkNum	PayTo	Category	Debit
04/04/99	2048	Blue Cross Of Calif	Insurance	177.55
04/04/99	2049	El Mar Plastics	Purchases	750.00
04/04/99	2050	Sir Speedy	Advertising	240.86
04/04/99	2051	Detail Associates	Purchases	104.10
04/06/99	2052	Sir Speedy	Advertising	186.65
04/16/99	2053	Advertiser's Mailing Ser	Postage	156.35
04/17/99	2054	Paper Mart	Office Supplies	339.46
04/17/99	2055	Public Storage	Rent	95.00
04/17/99	2056	U P S	Shipping	25.00
04/17/99	2057	PacTel Cellular	Telephone	187.58
04/17/99	2058	The Mail Secretary	Rent	75.00
04/17/99	2059	Public Storage	Rent	95.00
04/17/99	2060	Cap-Cor	Purchases	448.36

If you click on the grand total value in the lower right hand corner of the crosstab, the **Select Original Data** tool will select the entire database.

To re-select the entire original database, activate a data sheet or form window, then choose the **Select All** command from the Search Menu. Be sure to re-select the entire database before you re-calculate the crosstab.

**Warning:** If the database has been edited since the crosstab was calculated, the **Select Original Data** tool may not be able to locate the original data. If the database contained invisible (unselected) data when the crosstab was calculated, the **Select Original Data** tool may select this unselected data. If this is a problem, use the **Remove Unselected** command before calculating the crosstab (see "[Permanently Removing Unselected Data](#)" on page 167). Make sure you have a backup copy of your data on disk before you use this command.

# Chapter 12: Data Processing



This chapter describes some of the most powerful commands in Panorama. These commands allow you to automatically transform and modify large amounts of existing data. Many different kinds of transformations are possible, including mathematical calculations, re-arranging characters or words, transforming individual characters (for example converting from lower to upper case), and transformations based on patterns in the data.

The commands described in this chapter are very powerful. In a few seconds you may be able to make changes to your data that would otherwise require tedious hours of manual data entry. Like any power tool, these commands should be treated with respect. For insurance, you should **Save** your database before you begin trying to transform it. If you mangle your data, you can always get it back with the **Revert to Saved** command.

Most of the commands in this chapter are found in the Math Menu. However, this doesn't mean that only numbers can be transformed. Unless specified otherwise, these commands can transform all kinds of data, including text, numeric, dates, and choices.

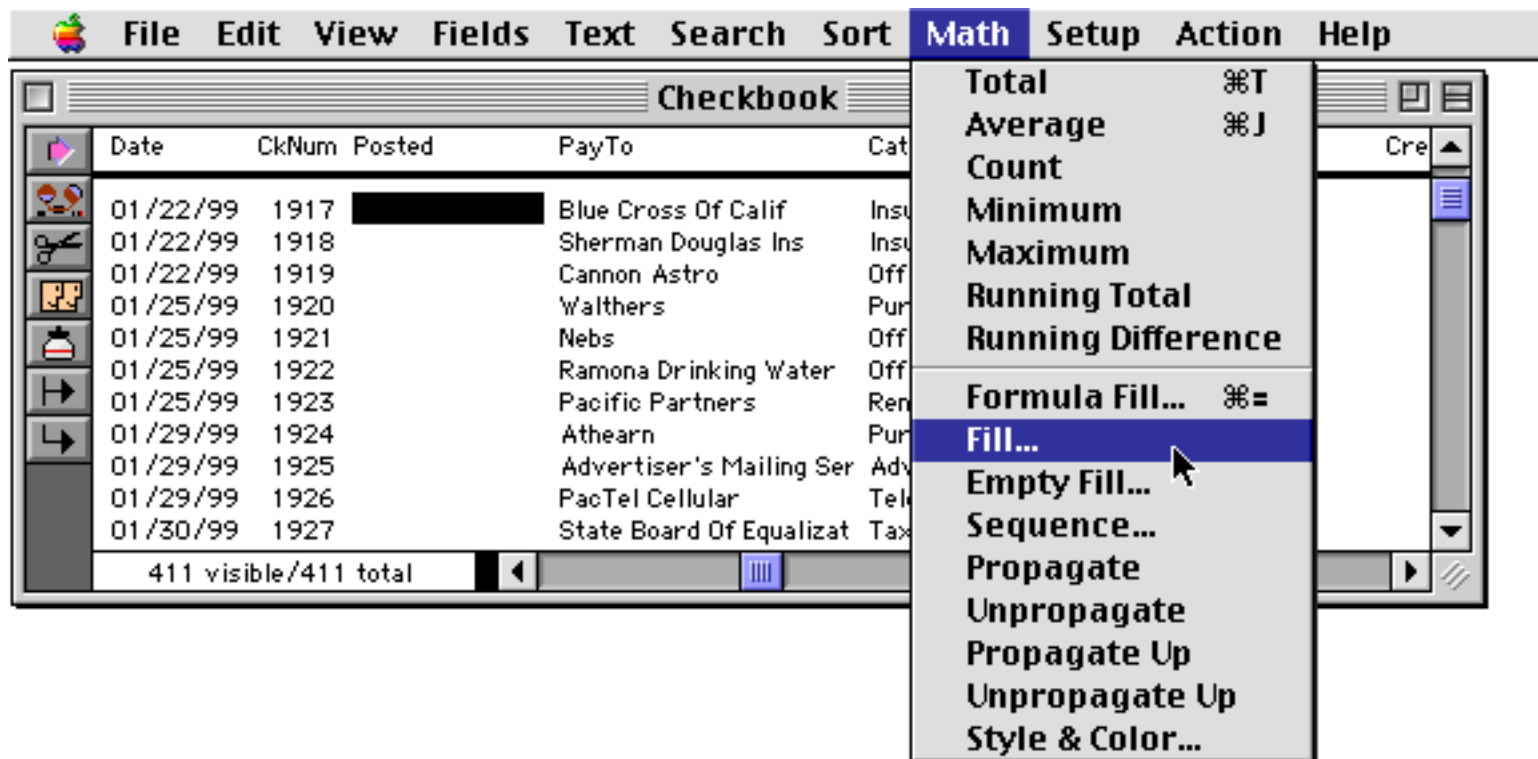
## Transforming Selected Data

The transformation commands described in this chapter may be used on an entire database, or on a selected subset. The **Find/Select** (or **Formula Find/Select**) command is used to select the data you want to transform, then the commands described in this chapter are used to transform the data. Only the selected data will be transformed—the invisible data will be left untouched. See “[The Find/Select Dialog](#)” on page 160 for more information on selecting a subset of the database.

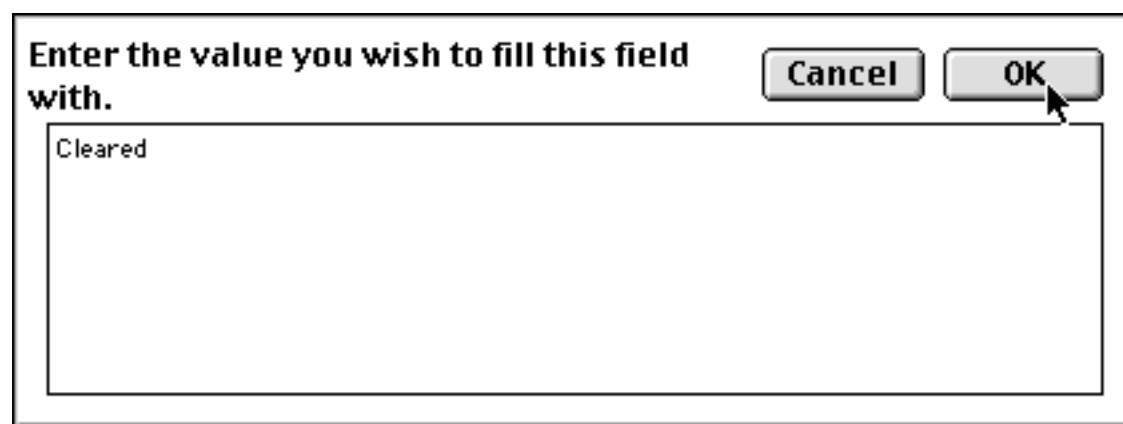
The same rules apply to data that has been collapsed with the outline tools. If data is invisible because it has been collapsed, it will not be transformed. Only data that is both selected and expanded will be transformed. See “[Summaries and Outlines](#)” on page 183 for more information on outlines.

## Filling a Field with a Fixed Value

The **Fill** command fills all the selected cells in the current field with a single value. Any data already in the field is destroyed (although you can get it back with **Undo**). To use the **Fill** command, just click on the field you want to fill and choose **Fill** from the Math Menu. Type in the value you want to fill the field with and then press the **Ok** button. For example, suppose you wanted to fill every cell in the **Posted** column in the database below with the text **Cleared**. Start by clicking on the **Posted** column and choosing the **Fill** command.



Now type **Cleared** into the dialog box.



When you press OK, every selected cell in this field will be replaced with **Cleared**. In this case all of the cells are empty, but they will be replaced whether they are empty or not. (See “[Filling Empty Cells](#)” on page 230 if you don’t want to disturb cells that already have data in them.)

Date	CkNum	Posted	PayTo	Category	Debit	Cre
01/22/99	1917	Cleared	Blue Cross Of Calif	Insurance	279.03	
01/22/99	1918	Cleared	Sherman Douglas Ins	Insurance	418.60	
01/22/99	1919	Cleared	Cannon Astro	Office Supplies	145.72	
01/25/99	1920	Cleared	Walthers	Purchases	1,885.40	
01/25/99	1921	Cleared	Nebs	Office Supplies	77.27	
01/25/99	1922	Cleared	Ramona Drinking Water	Office Supplies	98.10	
01/25/99	1923	Cleared	Pacific Partners	Rent	4,070.83	
01/29/99	1924	Cleared	Athearn	Purchases	1,906.32	
01/29/99	1925	Cleared	Advertiser's Mailing Ser	Advertising	860.22	
01/29/99	1926	Cleared	PacTel Cellular	Telephone	141.09	
01/30/99	1927	Cleared	State Board Of Equalizat	Taxes	549.00	

411 visible/411 total

The new data must be compatible with the field that is being filled. For example, you cannot fill a numeric field with **n/a** because **n/a** is not a numeric value. Panorama will warn you if you attempt to fill a field with an incompatible value.

### Filling a Field with a Formula

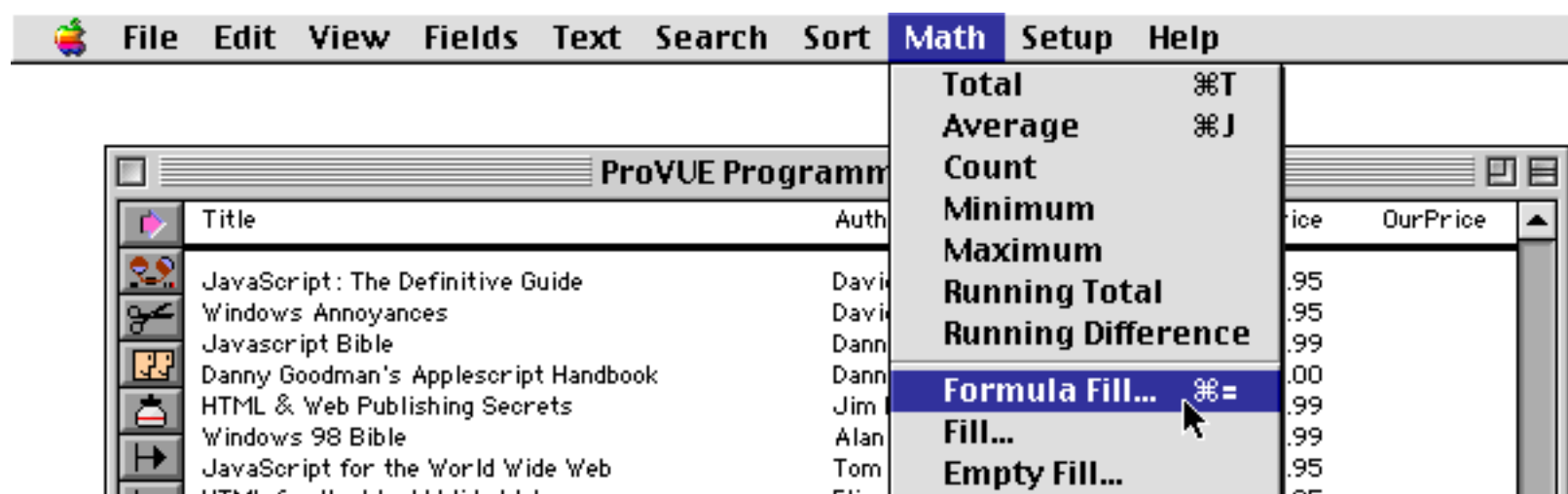
The **Formula Fill** command fills all the selected cells in the current field with the result of a formula. Any data already in the field is destroyed (although you can get it back with **Undo**). See “[Formulas](#)” on page 501 for more information on formulas.

The formula must match the data type of the field being filled. For example, a numeric formula can only be used if the current field contains numeric data. Panorama will display an error message if you use a formula that results in an incorrect data type.

Use the **Formula Fill** command when you need to perform a calculation on every selected record in the database. See “[Automatic Calculations](#)” on page 149 if you need to calculate a value immediately when data is entered.

### Numeric Calculations With Formula Fill

Use **Formula Fill** to perform calculations using the existing data. Use this command to calculate totals within a record, discounts, percentages, etc. For our example we’ll use a database with two price fields: **ListPrice** and **OurPrice**. We’ll use the **Formula Fill** command to calculate **OurPrice** based on a 25% discount from the list price. Start by clicking on the **OurPrice** field and choosing **Formula Fill** from the Math menu.





Now enter the formula to calculate the discount.



When you press **OK** Panorama will calculate the prices.

The screenshot shows the "ProVUE Programming Library" window. It contains a table with the following columns: Title, Authors, ListPrice, and OurPrice. The data is as follows:

Title	Authors	ListPrice	OurPrice
JavaScript: The Definitive Guide	David Flanagan, Dan Shafer	39.95	29.96
Windows Annoyances	David A. Karp	29.95	22.46
Javascript Bible	Danny Goodman	49.99	37.49
Danny Goodman's Applescript Handbook	Danny Goodman	40.00	30.00
HTML & Web Publishing Secrets	Jim Heid	49.99	37.49
Windows 98 Bible	Alan Simpson	39.99	29.99
JavaScript for the World Wide Web	Tom Negrino, Dori Smith	17.95	13.46
HTML for the World Wide Web	Elizabeth Castro	17.95	13.46
About Face: The Essentials of User Interface Design	Alan Cooper	29.99	22.49
Algorithms in C	Robert Sedgewick	59.95	44.96
C: A Reference Manual	Samuel P. Harbison, Guy L. Steele	42.95	32.21
Teach Yourself C in 21 Days	Peter G. Aitken	29.99	22.49
Teach Yourself Advanced C in 21 Days	Bradley L. Jones, Gregory L. Stein	34.95	26.21
HTML Quick Reference	Robert Mullen	20.00	15.00
The Design of Everyday Things	Donald A. Norman	15.95	11.96

At the bottom of the window, it says "23 visible/23 total".

As a further example, suppose that through a special contract you are able to offer a 40% discount on books published by O'Reilly & Associates. To change the discount for these books, start by using the Find/Select command to select only the books for this publisher.

The screenshot shows the "ProVUE Programming Library" window with a filtered view. The columns are ISBN, Binding, Pages, Publisher, Edition, ListPrice, and OurPrice. Only books published by "O'Reilly & Associates" are shown:

ISBN	Binding	Pages	Publisher	Edition	ListPrice	OurPrice
1565922352	Paperback	552	O'Reilly & Associates	2nd edition (Ma	32.95	24.71
1565921658	Paperback	180	O'Reilly & Associates	May 1996	24.95	18.71
1565923839	Paperback	560	O'Reilly & Associates	March 1998	34.95	26.21
1565921542	Paperback	410	O'Reilly & Associates	June 1996	23.96	17.97
1565922301	Paperback	180	O'Reilly & Associates	November 1996	39.95	29.96
1565923928	Paperback	776	O'Reilly & Associates	June 1998 (Th	39.95	29.96
1565922662	Paperback	280	O'Reilly & Associates	June 1997	29.95	22.46

Now click on the **OurPrice** field, and use the Formula Fill command again to calculate the new discount for these books. The new formula is **ListPrice \* 0.6**.

The screenshot shows the "ProVUE Programming Library" window with the same filtered view as the previous image. The "OurPrice" column has been updated with new values based on the formula "ListPrice \* 0.6":

ISBN	Binding	Pages	Publisher	Edition	ListPrice	OurPrice
1565922352	Paperback	552	O'Reilly & Associates	2nd edition (Ma	32.95	19.77
1565921658	Paperback	180	O'Reilly & Associates	May 1996	24.95	14.97
1565923839	Paperback	560	O'Reilly & Associates	March 1998	34.95	20.97
1565921542	Paperback	410	O'Reilly & Associates	June 1996	23.96	14.36
1565922301	Paperback	180	O'Reilly & Associates	November 1996	39.95	23.97
1565923928	Paperback	776	O'Reilly & Associates	June 1998 (Th	39.95	23.97
1565922662	Paperback	280	O'Reilly & Associates	June 1997	29.95	17.97



When you **Select All** you'll see that all the other books have kept their 25% discount. **Formula Fill** (and in fact all of the commands described in this chapter) does not touch unselected records.

ISBN	Binding	Pages	Publisher	Edition	ListPrice	OurPrice
1565923928	Paperback	776	O'Reilly & Associates	June 1998 (Th	39.95	23.97
1565922662	Paperback	280	O'Reilly & Associates	June 1997	29.95	17.97
0764531883	Paperback	607	IDG Books Worldwide	March 1998 (3	49.99	37.49
0679758062	Paperback	554	Random House	1994	40.00	30.00
0764540033	Paperback	626	IDG Books Worldwide	May 1997	49.99	37.49
0764531921	Paperback	1112	IDG Books Worldwide	June 1998	39.99	29.99
0201696487	Paperback	208	Peachpit Press	January 1998	17.95	13.46
020168862X	Paperback	255	Peachpit Press	February 1997	17.95	13.46
1568843224	Paperback	400	IDG Books Worldwide	August 1995	29.99	22.49
0201514257	Hardcover	657	Addison-Wesley	April 1992	59.95	44.96
0133262243	Paperback	455	Prentice Hall Computer Boc	December 1994	42.95	32.21
0672310694	Paperback	736	Sams Publishing	August 1997 (4	29.99	22.49
0672304716	Paperback	878	Sams Publishing	April 1994	34.95	26.21
0789708671	Paperback	222	Que	1996	20.00	15.00
0385267746	Paperback	257	Currency/DoubleDay	March 1990	15.95	11.96

Remember that if you use **Formula Fill** to calculate numeric values, you must be filling a numeric field. (If you need to **Formula Fill** a text field with numeric values, use the `str()` or `pattern()` functions to convert the numbers to text. For example, the formula `str(Price-Cost)` can be used to fill a text field.) See “[Converting Between Numbers and Strings](#)” on page 538 for more information.

### Using Formula Fill to Transform Characters

Using the **Formula Fill** command you can combine multiple fields, split a field apart, re-arrange words or phrases, and translate characters (for example, converting uppercase to lower case). Use the + operator to join (concatenate) text fields together.

For example the formula `First+" "+Last` will combine the first and last names (with a space in between) into a single field. We'll start by inserting a blank **Name** field into our Contacts database (see “[Add Field](#)” on page 102).

First	Last	Name	Title	Company	Address
John	Smith		Sales Manager	Acme Widgets	12 Harmo
Susan	Brown				783 Algor
Karen	Wilson		Vice President	Evanston Lumber	498 Noye
Jim	Nickle		President	Jim's Appliances	14189 8t
Brian	Felty			B.F. Plumbing	118 N Wil
Bob	Hanlan		Sales Manager	Ann Arbor Lumber	6916 Mor
Tim	Daniels		Customer Supp	St. Louis Lumber	3133 Cor
John	Moses				8265 Leti
John	Fabian				3 Rose Hi

Now select the **Formula Fill** command from the Math Menu and type in the formula `First+" "+Last`.

Enter the formula:

First+" "+Last

Cancel OK

When you press **OK** the Name field will be filled in.

First	Last	Name	Title	Company	Address
John	Smith	John Smith	Sales Manager	Acme Widgets	12 Harmo
Susan	Brown	Susan Brown			783 Algor
Karen	Wilson	Karen Wilson	Vice President	Evanston Lumber	498 Noyes
Jim	Nickle	Jim Nickle	President	Jim's Appliances	14189 8th
Brian	Felty	Brian Felty		B.F. Plumbing	118 N Wil
Bob	Hanlan	Bob Hanlan	Sales Manager	Ann Arbor Lumber	6916 Mor
Tim	Daniels	Tim Daniels	Customer Supp	St. Louis Lumber	3133 Cor
John	Moses	John Moses			8265 Leti
John	Fabian	John Fabian			3 Rose Hi

102 visible/102 total

By using a different formula we can change the results. For example, if you wanted the last name first, you would use the formula `Last+", "+First`. Here's the result.

First	Last	Name	Title	Company	Address
John	Smith	Smith, John	Sales Manager	Acme Widgets	12 Harmo
Susan	Brown	Brown, Susan			783 Algor
Karen	Wilson	Wilson, Karen	Vice President	Evanston Lumber	498 Noyes
Jim	Nickle	Nickle, Jim	President	Jim's Appliances	14189 8th
Brian	Felty	Felty, Brian		B.F. Plumbing	118 N Wil
Bob	Hanlan	Hanlan, Bob	Sales Manager	Ann Arbor Lumber	6916 Mor
Tim	Daniels	Daniels, Tim	Customer Supp	St. Louis Lumber	3133 Cor
John	Moses	Moses, John			8265 Leti
John	Fabian	Fabian, John			3 Rose Hi

102 visible/102 total

Use the `upper()`, `lower()`, and `upperword()` functions to convert text between upper and lower case (see "[String Modification Functions](#)" on page 535). For example, if you wanted the last names in all upper case the formula `FirstName+" "+upper(LastName)` would be used. Here is the result.

First	Last	Name	Title	Company	Address
John	Smith	John SMITH	Sales Manager	Acme Widgets	12 Harmo
Susan	Brown	Susan BROWN			783 Algor
Karen	Wilson	Karen WILSON	Vice President	Evanston Lumber	498 Noyes
Jim	Nickle	Jim NICKLE	President	Jim's Appliances	14189 8th
Brian	Felty	Brian FELTY		B.F. Plumbing	118 N Wil
Bob	Hanlan	Bob HANLAN	Sales Manager	Ann Arbor Lumber	6916 Mor
Tim	Daniels	Tim DANIELS	Customer Supp	St. Louis Lumber	3133 Cor
John	Moses	John MOSES			8265 Leti
John	Fabian	John FABIAN			3 Rose Hi

102 visible/102 total

Use text funnels to split a field apart or to re-arrange words or phrases. Text funnels allow a formula to extract part of a cell based on a fixed character position within the cell, or based on patterns and context within the cell. In this example we'll use the formula `FirstName[1,1]+". "+LastName` to fill the field with the first initial and the last name. Here is the result.

First	Last	Name	Title	Company	Address
John	Smith	J. Smith	Sales Manager	Acme Widgets	12 Harmo
Susan	Brown	S. Brown			783 Algor
Karen	Wilson	K. Wilson	Vice President	Evanston Lumber	498 Noyes
Jim	Nickle	J. Nickle	President	Jim's Appliances	14189 8th
Brian	Felty	B. Felty		B.F. Plumbing	118 N Wil
Bob	Hanlan	B. Hanlan	Sales Manager	Ann Arbor Lumber	6916 Mor
Tim	Daniels	T. Daniels	Customer Supp	St. Louis Lumber	3133 Cor
John	Moses	J. Moses			8265 Leti
John	Fabian	J. Fabian			3 Rose Hi

### Date Calculations with Formula Fill

Use the **Formula Fill** command to calculate the difference between dates, or to adjust dates. See "[Date Arithmetic](#)" on page 544 for details on performing calculations with dates.

A typical use for date arithmetic is aging of an accounts receivable database.

Invoice #	Subtotal	Tax	Total	Ship Date	Age
160	27.81	1.39	29.20	10/09/99	
161	40.35	2.02	42.37	10/10/99	
162	119.68	5.98	125.66	10/11/99	

To calculate the age of an invoice based on the current date, use the **Formula Fill** command with the formula shown here:

Enter the formula:

today()-«Ship Date»

Cancel OK

(Notice the chevrons (« and ») around the field name. These are necessary because of the space in the field name. You can simply let Panorama type the field name in for you by choosing from the **Field** menu.)

Press **OK** to calculate the age of each invoice.

Invoice #	Subtotal	Tax	Total	Ship Date	Age
160	27.81	1.39	29.20	10/09/99	23
161	40.35	2.02	42.37	10/10/99	22
162	119.68	5.98	125.66	10/11/99	21
163	189.70	9.49	199.19	10/14/99	18
164	84.85	4.24	89.09	10/15/99	17
165	59.90	3.00	62.90	10/16/99	16
166	30.60	1.53	32.13	10/16/99	16
167	34.82	1.74	36.56	10/17/99	15
168	34.32	1.72	36.04	10/17/99	15
169	9.50	0.48	9.98	10/17/99	15
170	15.54	0.78	16.32	10/18/99	14
171	20.60	1.03	21.63	10/18/99	14
172	70.32	3.52	73.84	10/19/99	13

If you want to calculate ages rounded to the nearest 30 day interval use the formula below instead.

Enter the formula: Cancel OK

round((today()-«Ship Date»-15),30)

Press **OK** to calculate the age of each invoice rounded to the nearest 30 days.

Invoice #	Subtotal	Tax	Total	Ship Date	Age
101	174.59	8.73	183.32	08/08/99	60
102	130.53	6.53	137.06	08/10/99	60
105	169.65	8.48	178.13	08/14/99	60
106	177.70	8.89	186.59	08/17/99	60
110	144.72	7.24	151.96	08/20/99	60
113	191.38	9.57	200.95	08/24/99	60
115	293.63	14.68	308.31	08/29/99	60
119	122.89	6.14	129.03	09/01/99	60
122	115.65	5.78	121.43	09/04/99	30
123	125.45	6.27	131.72	09/06/99	30
124	155.08	7.75	162.83	09/08/99	30
126	368.73	18.44	387.17	09/14/99	30
130	126.20	6.31	132.51	09/17/99	30
141	104.41	5.22	109.63	09/23/99	30
144	131.70	6.59	138.29	09/25/99	30
146	152.10	7.61	159.71	09/28/99	30
153	141.15	7.06	148.21	10/05/99	0
162	119.68	5.98	125.66	10/11/99	0
163	189.70	9.49	199.19	10/14/99	0
173	107.14	5.36	112.50	10/20/99	0
175	109.20	5.46	114.66	10/23/99	0

21 visible / 74 total

### The SEQ Function

The **seq()** function is a special function for use with the **Formula Fill** command. This function returns a unique number for each selected record, starting with **1** at the top of the database. Use this function if you need a unique record number in a formula. Here is an example that fills a column with the words **One**, **Two**, **Three**, **Four**, etc.

Enter the formula: Cancel OK

pattern(seq(),"S")

When you press **OK** the field is filled in.

Name	Time	Place	Medal	Behind
Steven Martin	18.17	One		
Joshua Trask	18.19	Two		
Jim Lederman	18.63	Three		
Michael Williams	19.21	Four		
John Wilcox	19.23	Five		
Stan Damone	20.05	Six		
Keith Dawson	20.34	Seven		
Ben Longworth	20.93	Eight		

Here is another example that uses the `seq()` function to assign medals to the first three finishers in the race.

Enter the formula:

Cancel OK

`=(seq())<4,array("Gold/Silver/Bronze",seq(),"/"),""`

The first three finishers are assigned gold, silver, and bronze medals, with all of the other records left blank.

Name	Time	Place	Medal	Behind
Steven Martin	18.17	One	Gold	
Joshua Trask	18.19	Two	Silver	
Jim Lederman	18.63	Three	Bronze	
Michael Williams	19.21	Four		
John Wilcox	19.23	Five		
Stan Damone	20.05	Six		
Keith Dawson	20.34	Seven		
Ben Longworth	20.93	Eight		

See Chapter 23 for more information on the `array()` function used in this example.



## Filling Empty Cells

The **Empty Fill** command is very similar to the **Fill** command (see “[Filling a Field with a Fixed Value](#)” on page 222). However, the **Empty Fill** command will not destroy the data already in the field. In fact, **Empty Fill** will only fill cells that are completely empty. Here is a database where some of the name prefixes have been left blank.

T	First Name	Last Name	Company Name	Street Address	Suite Box	City	State
	Jim	Abrahms	International Transportat	329 North State		Alameda	CA
Ms.	Isabel	Alston	Leader Systems	10463 N. Blaney		San Rafael	CA
	Anthony	Campbell	Drake Inc.	3452 Van Ness		Chicago	IL
Dr.	Robert	Churchill	Smiley & Sons Developme	445 Hanilton		Manasquan	NJ
Ms.	Bea	Clairmont	Arrow Dev.	411 Pacific		Cambridge	MA
Mrs.	Dottie	Davidson	Van Ness Marketing	181 Taintor Dr.		Boston	MA
Mrs.	Barbara	Elliott	Foltene	10440 Torre A		Cambridge	MA
	Lew	Farrell	Coast Label & Supply	33095 Lexingto		San Rafael	CA
Ms.	Kris	Frazee	Mariani Publishing	18550 Lark Ave		San Francisco	CA
	Tim	Hill	Alameda Escrow	1000 Roche Blv		Santa Ana	CA
	Keith	Jacobs	Bath Co.	2994 Garcia Av		Burbank	CA
	Joe	Jones	Professionals Inc.	2 Owen St.		Provo	UT
	Al	Keizer	Certified Labs	1184 Lincoln Av	Suite	Westlake Village	CA
Ms.	Robin	Knight	Fico Appliance Service	2155 S. Bascom	Suite	Tallahassee	FL
	Mark	Lauing	Sherman-Davis	490 Broadway		Naples	FL
Dr.	Bill	Lieber	Arlington Associates	573 Dundee Rd.		Waldwick	NJ
	Russ	Malone	Northridge Bakeries	2210 Wilshire E	Suite	Chicago	IL
Ms.	Jan	Morgan	McCormick-Ridder	4044 Civic Terr		San Diego	CA
	Frank	Pearce	Taylor & Associates	9344 Kashiwa E		San Diego	CA
Mrs.	Dotty	Prenner	Cook & Sons	1900 S. Eads St	Suite	White Plains	NY
	Mike	Reynolds	Birch Catering	136 Harvey		San Francisco	CA

Using the **Empty Fill** command these empty cells can quickly be filled with **Mr.**

Enter the value you wish to fill this field with.

Mr.

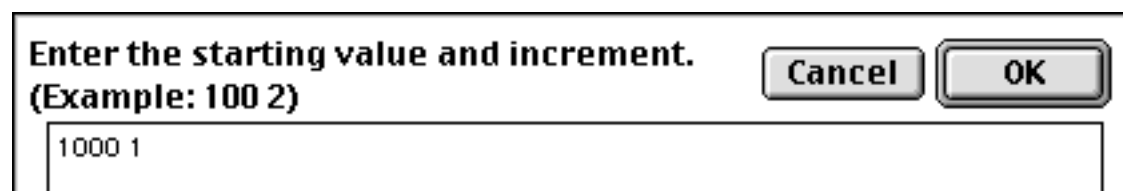
Here's the finished result.

T	First Name	Last Name	Company Name	Street Address	Suite Box	City	State
Mr.	Jim	Abrahms	International Transportat	329 North State		Alameda	CA
Ms.	Isabel	Alston	Leader Systems	10463 N. Blaney		San Rafael	CA
Mr.	Anthony	Campbell	Drake Inc.	3452 Van Ness		Chicago	IL
Dr.	Robert	Churchill	Smiley & Sons Developme	445 Hanilton		Manasquan	NJ
Ms.	Bea	Clairmont	Arrow Dev.	411 Pacific		Cambridge	MA
Mrs.	Dottie	Davidson	Van Ness Marketing	181 Taintor Dr.		Boston	MA
Mrs.	Barbara	Elliott	Foltene	10440 Torre A		Cambridge	MA
Mr.	Lew	Farrell	Coast Label & Supply	33095 Lexingto		San Rafael	CA
Ms.	Kris	Frazee	Mariani Publishing	18550 Lark Ave		San Francisco	CA
Mr.	Tim	Hill	Alameda Escrow	1000 Roche Blv		Santa Ana	CA
Mr.	Keith	Jacobs	Bath Co.	2994 Garcia Av		Burbank	CA
Mr.	Joe	Jones	Professionals Inc.	2 Owen St.		Provo	UT
Mr.	Al	Keizer	Certified Labs	1184 Lincoln Av	Suite	Westlake Village	CA
Ms.	Robin	Knight	Fico Appliance Service	2155 S. Bascom	Suite	Tallahassee	FL
Mr.	Mark	Lauing	Sherman-Davis	490 Broadway		Naples	FL
Dr.	Bill	Lieber	Arlington Associates	573 Dundee Rd.		Waldwick	NJ
Mr.	Russ	Malone	Northridge Bakeries	2210 Wilshire E	Suite	Chicago	IL
Ms.	Jan	Morgan	McCormick-Ridder	4044 Civic Terr		San Diego	CA
Mr.	Frank	Pearce	Taylor & Associates	9344 Kashiwa E		San Diego	CA
Mrs.	Dotty	Prenner	Cook & Sons	1900 S. Eads St	Suite	White Plains	NY
Mr.	Mike	Reynolds	Birch Catering	136 Harvey		San Francisco	CA

## Automatic Numbering

The **Sequence** command fills the current field with a numeric sequence (for example 1, 2, 3 or 100, 110, 120). The **Sequence** command only works with numeric fields, you cannot sequence a text, date, or choice field.

To use the **Sequence** command, first click on the field you want to fill, then choose the **Sequence** command from the Math Menu. Type the first value in the sequence, then a space, then the increment value.



For example, type **1000 10** if you want to fill the field with the sequence **1000, 1010, 1020**, etc. Press **OK** to actually fill the field. In this example the sequence number starts at 1000 and increments by one.

Reg #	T	First Name	Last Name	Company	Street Address	Sui Box
1000	Mr.	Jim	Abrahms	International Transportat	329 North State	
1001	Ms.	Isabel	Alston	Leader Systems	10463 N. Blaney	
1002	Mr.	Anthony	Campbell	Drake Inc.	3452 Van Ness	
1003	Dr.	Robert	Churchill	Smiley & Sons Developme	445 Hamilton	
1004	Ms.	Bea	Clairmont	Arrow Dev.	411 Pacific	
1005	Mrs.	Dottie	Davidson	Van Ness Marketing	181 Taintor Dr.	
1006	Mrs.	Barbara	Elliott	Foltene	10440 Torre Av	
1007	Mr.	Lew	Farrell	Coast Label & Supply	33095 Lexington	
1008	Ms.	Kris	Frazee	Mariani Publishing	18550 Lark Ave	
1009	Mr.	Tim	Hill	Alameda Escrow	1000 Roche Bly	
1010	Mr.	Keith	Jacobs	Bath Co.	2994 Garcia Av	
1011	Mr.	Joe	Jones	Professionals Inc.	2 Owen St.	
1012	Mr.	Al	Keizer	Certified Labs	1184 Lincoln Av Sui	
1013	Ms.	Robin	Knight	Fico Appliance Service	2155 S. Bascom Sui	

The sequence can start with any number and increase by any value, including non-integer values or negative values. The table below shows four examples of starting and increment values.

1 1	5 5	1 0.1	100 -1
1	5	1.0	100
2	10	1.1	99
3	15	1.2	98
4	20	1.3	97
5	25	1.4	96

If the database contains summary records, the sequence count will reset to one after each summary record. If you want to sequence the current field without restarting at summary records, use the **Formula Fill** command with the formula `seq()`. See “[Summaries and Outlines](#)” on page 371 for more information on summary records. See “[Filling a Field with a Formula](#)” on page 223 for more information on the **Formula Fill** command.

## Propagate

Like **Empty Fill**, the **Propagate** command fills all the empty cells in the current field. However, instead of filling the empty cells with a fixed value, the **Propagate** command propagates filled data cells into the empty data cells (if any) below them.

To illustrate, here is a database where the date was only entered for the first check written each day. For example, checks 1907, 1908 and 1909 were all written on January 8th, but the date has only been filled in for check 1907.

Date	CkNum	PayTo
01/01/99		OPENING BALANCE
01/08/99	1907	Northern Illinois Mold
	1908	U S Postmaster
	1909	Advertiser's Mailing Ser
01/16/99	1910	Coudert Brothers, Attor
	1911	Paramount Stationers
01/17/99	1912	California Capitol
	1913	California Capitol
	1914	U S Postmaster
	1915	Sacramento Bee
01/18/99		DEPOSIT
01/22/99	1916	Walthers
	1917	Blue Cross Of Calif
	1918	Sherman Douglas Ins
	1919	Cannon Astro
01/25/99	1920	Walthers
	1921	Nebs
	1922	Ramona Drinking Water
	1923	Pacific Partners
01/29/99	1924	Athearn

The Propagate command will fill in the empty cells, as shown by the arrows in this illustration.

Date	CkNum	PayTo
01/01/99		OPENING BALANCE
01/08/99	1907	Northern Illinois Mold
	1908	U S Postmaster
	1909	Advertiser's Mailing Ser
01/16/99	1910	Coudert Brothers, Attor
	1911	Paramount Stationers
01/17/99	1912	California Capitol
	1913	California Capitol
	1914	U S Postmaster
	1915	Sacramento Bee

Here is the actual result after the Propagate has completed.

Date	CkNum	PayTo
01/01/99		OPENING BALANCE
01/08/99	1907	Northern Illinois Mold
01/08/99	1908	U S Postmaster
01/08/99	1909	Advertiser's Mailing Ser
01/16/99	1910	Coudert Brothers, Attor
01/16/99	1911	Paramount Stationers
01/17/99	1912	California Capitol
01/17/99	1913	California Capitol
01/17/99	1914	U S Postmaster
01/17/99	1915	Sacramento Bee
01/18/99		DEPOSIT
01/22/99	1916	Walthers
01/22/99	1917	Blue Cross Of Calif
01/22/99	1918	Sherman Douglas Ins
01/22/99	1919	Cannon Astro
01/25/99	1920	Walthers
01/25/99	1921	Nebs

The **Propagate Up** command performs the same operation upside down, propagating filled data cells into the empty data cells above them.

## UnPropagate

This command performs the exact inverse of the **Propagate** command. If the same value appears in two or more consecutive data cells, the **Unpropagate** command empties the second and subsequent data cells. Here is a database that has been sorted by city.

City	State	Zip Code
San Diego	CA	92126
San Diego	CA	92108
San Diego	CA	92109
San Francisco	CA	94114
San Francisco	CA	94104
San Francisco	CA	94104
San Francisco	CA	94107
San Mateo	CA	94404
San Mateo	CA	94404
San Mateo	CA	94404
San Mateo	CA	94404
San Rafael	CA	94912
San Rafael	CA	94901
San Rafael	CA	94903
San Rafael	CA	94903
San Rafael	CA	94901
San Rafael	CA	94903
San Rafael	CA	94903
San Ramon	CA	94583

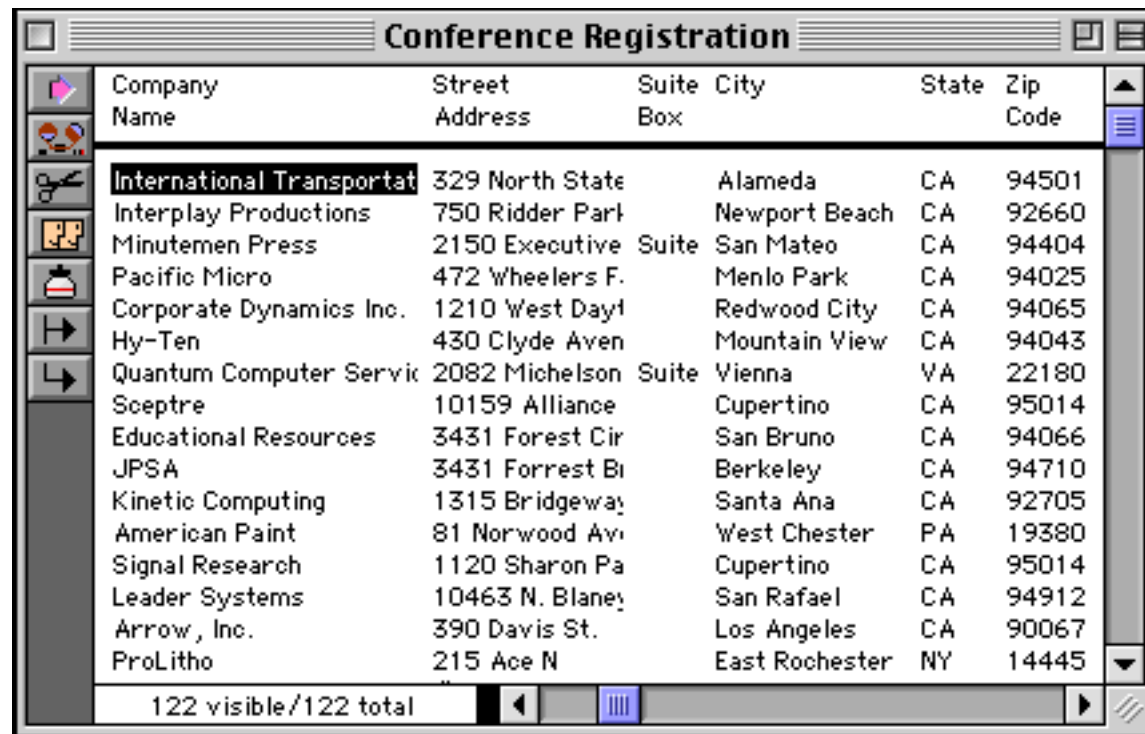
The **Unpropagate** command eliminates all but the first entry for each city.

City	State	Zip Code
San Diego	CA	92126
	CA	92108
	CA	92109
San Francisco	CA	94114
	CA	94104
	CA	94104
	CA	94107
San Mateo	CA	94404
	CA	94404
	CA	94404
	CA	94404
San Rafael	CA	94912
	CA	94901
	CA	94903
	CA	94903
	CA	94901
	CA	94903
	CA	94903
San Ramon	CA	94583
Santa Ana	CA	92705
	CA	92704
	CA	92705
Southport	CT	06490

The **Unpropagate Up** command performs the same operation upside down, leaving the last of several duplicate values while clearing the others.

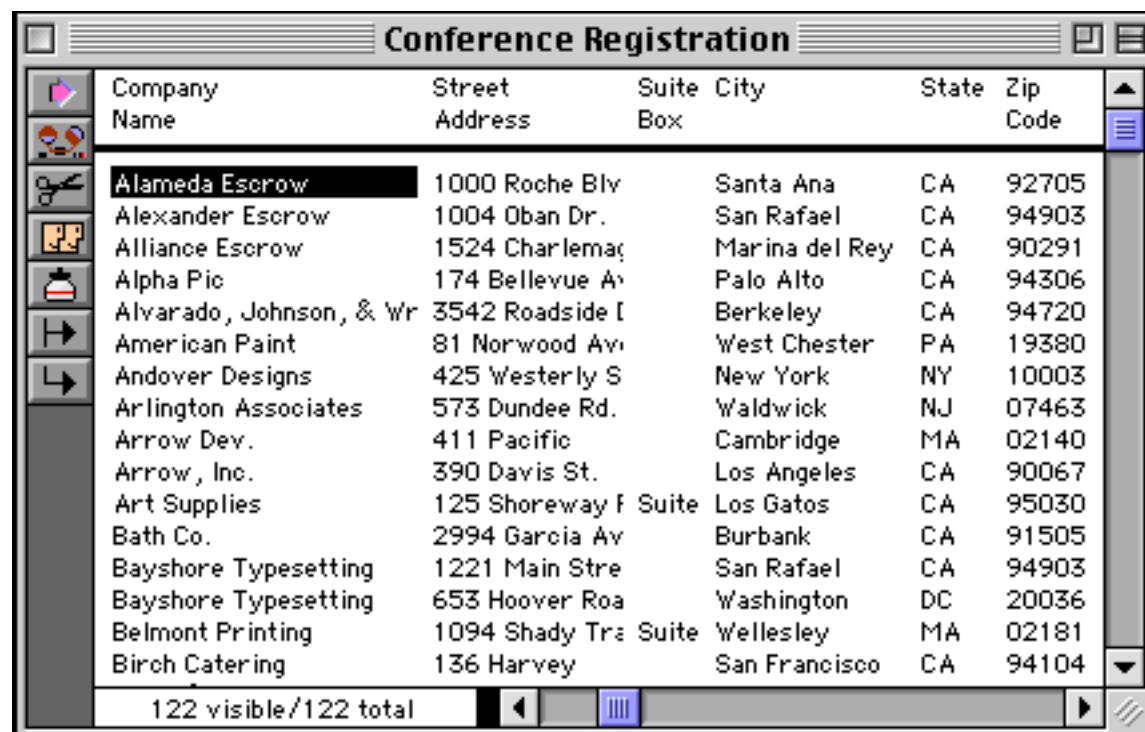
## Using UnPropagate to Eliminate Duplicates

The **UnPropagate** command can be used to eliminate duplicate values in a database. The first step is to click on the field that contains the potentially duplicate values, for example **Name** or **Company**. If you want to eliminate duplicates over multiple fields (for example an entire address) you must create a new field and use the **Formula Fill** command to combine the data into a single field.



Company Name	Street Address	Suite Box	City	State	Zip Code
International Transportat	329 North State		Alameda	CA	94501
Interplay Productions	750 Ridder Park		Newport Beach	CA	92660
Minutemen Press	2150 Executive	Suite	San Mateo	CA	94404
Pacific Micro	472 Wheelers F.		Menlo Park	CA	94025
Corporate Dynamics Inc.	1210 West Dayl		Redwood City	CA	94065
Hy-Ten	430 Clyde Aven		Mountain View	CA	94043
Quantum Computer Servic	2082 Michelson	Suite	Vienna	VA	22180
Sceptre	10159 Alliance		Cupertino	CA	95014
Educational Resources	3431 Forest Cir		San Bruno	CA	94066
JPSA	3431 Forrest Bl		Berkeley	CA	94710
Kinetic Computing	1315 Bridgeway		Santa Ana	CA	92705
American Paint	81 Norwood Av		West Chester	PA	19380
Signal Research	1120 Sharon Pa		Cupertino	CA	95014
Leader Systems	10463 N. Blaney		San Rafael	CA	94912
Arrow, Inc.	390 Davis St.		Los Angeles	CA	90067
ProLitho	215 Ace N		East Rochester	NY	14445

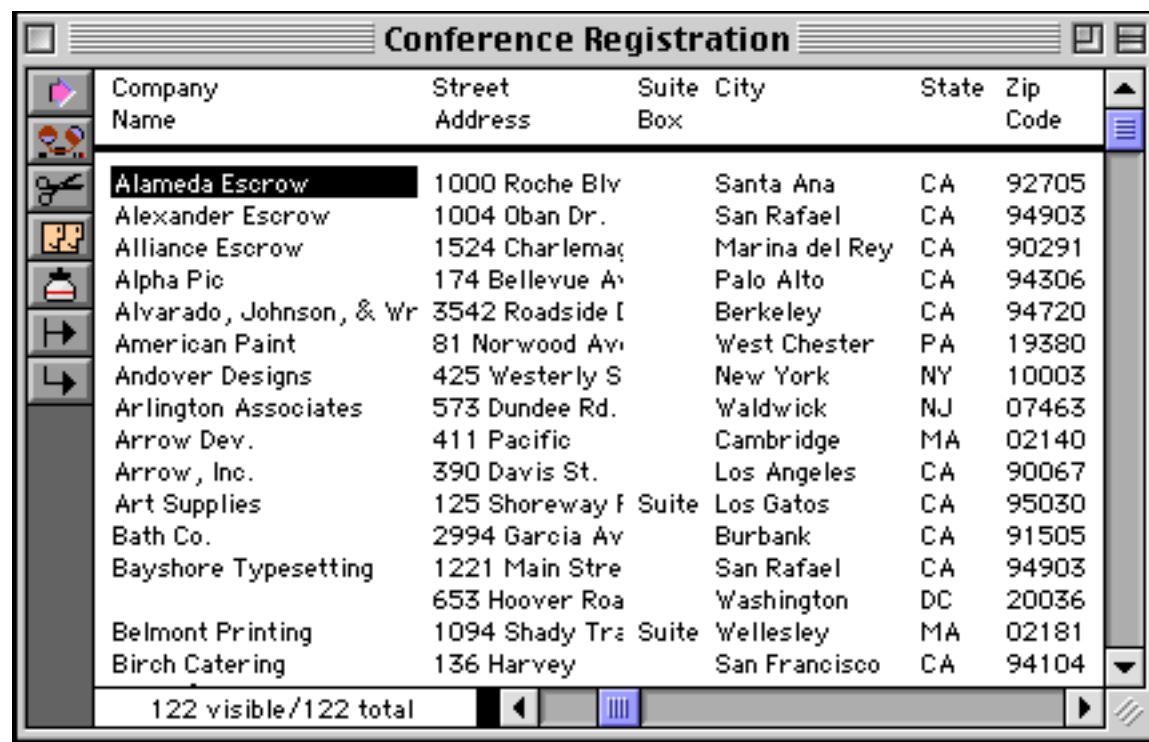
The next step is **SortUp** the database. This brings all the duplicate values together. For example, there are two **Bayshore Typesetting** entries in this database.



Company Name	Street Address	Suite Box	City	State	Zip Code
Alameda Escrow	1000 Roche Bly		Santa Ana	CA	92705
Alexander Escrow	1004 Oban Dr.		San Rafael	CA	94903
Alliance Escrow	1524 Charlemaç		Marina del Rey	CA	90291
Alpha Pic	174 Bellevue Av		Palo Alto	CA	94306
Alvarado, Johnson, & Wr	3542 Roadside [		Berkeley	CA	94720
American Paint	81 Norwood Av		West Chester	PA	19380
Andover Designs	425 Westerly S		New York	NY	10003
Arlington Associates	573 Dundee Rd.		Waldwick	NJ	07463
Arrow Dev.	411 Pacific		Cambridge	MA	02140
Arrow, Inc.	390 Davis St.		Los Angeles	CA	90067
Art Supplies	125 Shoreway F Suite		Los Gatos	CA	95030
Bath Co.	2994 Garcia Av		Burbank	CA	91505
Bayshore Typesetting	1221 Main Stre		San Rafael	CA	94903
Bayshore Typesetting	653 Hoover Roa		Washington	DC	20036
Belmont Printing	1094 Shady Tra Suite		Wellesley	MA	02181
Birch Catering	136 Harvey		San Francisco	CA	94104



The next step is the **UnPropagate** command. Wherever a duplicate value appears in the data cell, the **UnPropagate** command clears the cell.



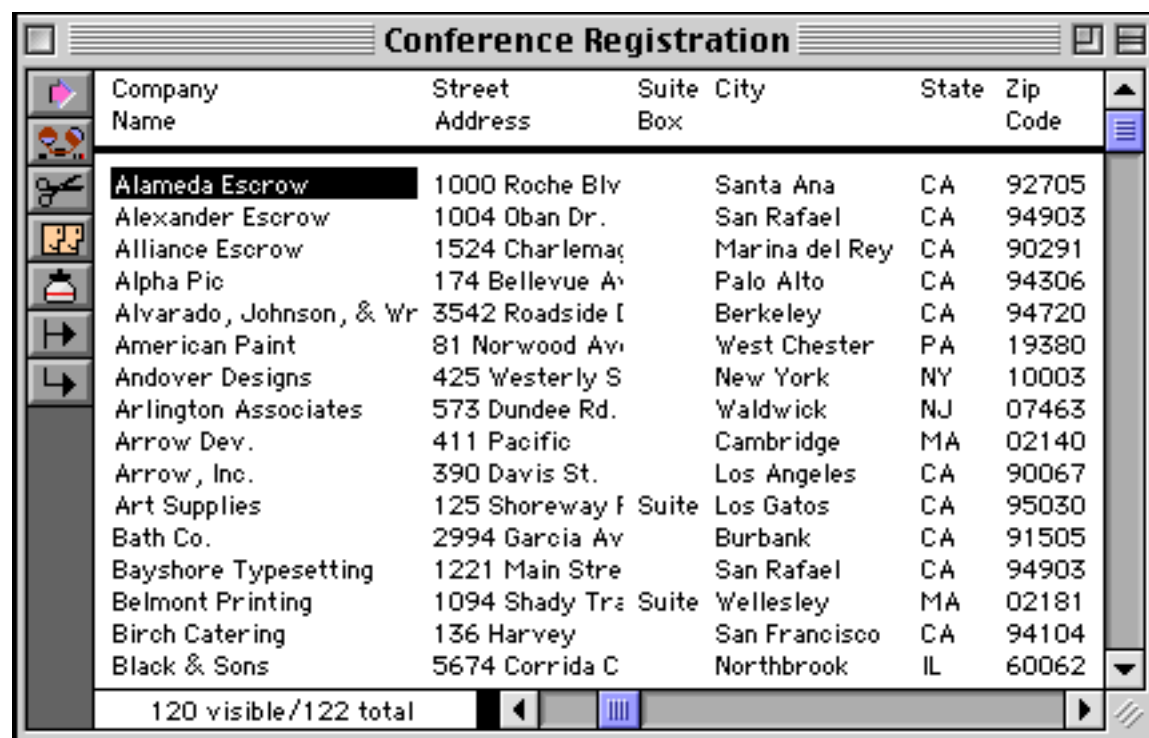
Company Name	Street Address	Suite Box	City	State	Zip Code
Alameda Escrow	1000 Roche Blv		Santa Ana	CA	92705
Alexander Escrow	1004 Oban Dr.		San Rafael	CA	94903
Alliance Escrow	1524 Charlemaç		Marina del Rey	CA	90291
Alpha Pic	174 Bellevue Av		Palo Alto	CA	94306
Alvarado, Johnson, & Wr	3542 Roadside I		Berkeley	CA	94720
American Paint	81 Norwood Av		West Chester	PA	19380
Andover Designs	425 Westerly S		New York	NY	10003
Arlington Associates	573 Dundee Rd.		Waldwick	NJ	07463
Arrow Dev.	411 Pacific		Cambridge	MA	02140
Arrow, Inc.	390 Davis St.		Los Angeles	CA	90067
Art Supplies	125 Shoreway F Suite		Los Gatos	CA	95030
Bath Co.	2994 Garcia Av		Burbank	CA	91505
Bayshore Typesetting	1221 Main Stre		San Rafael	CA	94903
	653 Hoover Roa		Washington	DC	20036
Belmont Printing	1094 Shady Tra Suite		Wellesley	MA	02181
Birch Catering	136 Harvey		San Francisco	CA	94104

122 visible/122 total

Now use the **Find/Select** command to select the non-empty data cells. Just pick **Not Equal** from the pop-up menu.



All of the duplicate records will disappear when you press the **Select** button. In this database there were two duplicate companies, so there are now 120 selected (non-duplicate) records.



Company Name	Street Address	Suite Box	City	State	Zip Code
Alameda Escrow	1000 Roche Blv		Santa Ana	CA	92705
Alexander Escrow	1004 Oban Dr.		San Rafael	CA	94903
Alliance Escrow	1524 Charlemaç		Marina del Rey	CA	90291
Alpha Pic	174 Bellevue Av		Palo Alto	CA	94306
Alvarado, Johnson, & Wr	3542 Roadside I		Berkeley	CA	94720
American Paint	81 Norwood Av		West Chester	PA	19380
Andover Designs	425 Westerly S		New York	NY	10003
Arlington Associates	573 Dundee Rd.		Waldwick	NJ	07463
Arrow Dev.	411 Pacific		Cambridge	MA	02140
Arrow, Inc.	390 Davis St.		Los Angeles	CA	90067
Art Supplies	125 Shoreway F Suite		Los Gatos	CA	95030
Bath Co.	2994 Garcia Av		Burbank	CA	91505
Bayshore Typesetting	1221 Main Stre		San Rafael	CA	94903
Belmont Printing	1094 Shady Tra Suite		Wellesley	MA	02181
Birch Catering	136 Harvey		San Francisco	CA	94104
Black & Sons	5674 Corrida C		Northbrook	IL	60062

120 visible/122 total

The final step is to permanently remove the duplicate records with the **Remove Unselected** command.

## Change (Find and Replace)

The **Change** command (in the Search menu) finds and replaces a word or phrase in the current field. For example, you can use the **Change** command to replace every occurrence of **Inc.** to **Incorporated**, or every occurrence of **Purchase Order** to **P.O.**

The **Change** dialog allows you to specify the original (From) and the new (To) word or phrase.

The screenshot shows a dialog box titled "Change". It has two input fields: "From:" containing "Inc." and "To:" containing "Incorporated". Below these fields are two checkboxes: "Adjust Capitalization" and "Replace Entire Words Only", both of which are unchecked. At the bottom right of the dialog are two buttons: "Cancel" and "Change".

The **Adjust Capitalization** option allows you to specify whether you want capitalization to be adjusted as the word or phrase is replaced. If you check this option, Panorama will automatically adjust the capitalization of the new word or phrase as it is inserted into the database. If you leave this option off, capitalization is not adjusted. In fact, if the **Adjust Capitalization** option is off, only words or phrases that exactly match the capitalization typed into the dialog will be replaced. The table below shows the result of replacing **Inc.** with **Incorporated** with **Adjust Capitalization** both off and on.

Original	Adjust Capitalization OFF	Adjust Capitalization ON
Inc.	Incorporated	Incorporated
INC.	INC.	INCORPORATED
inc.	inc.	incorporated

The **Replace Entire Words Only** option tells Panorama to replace only entire words, not sections of words. For example, if you ask Panorama to change **is** to **was**, it will also change **this** to **thwas**. This is, of course, wrong. To prevent this, just check the **Replace Entire Words Only** option.

## Changing with the Replace( Function

The **Change** command is not the only way to replace words or phrases. You can also use the **Formula Fill** command and the `replace()` or `replacemultiple()` functions (see “[String Modification Functions](#)” on page 535). This technique is especially handy if you need to replace several words or phrases at once. For example, consider the addresses in the database below.

Company Name	Street Address	Suite Box	City
International Transportat	329 North State St.		Alameda
Pacific Micro	472 Wheelers Farms Rd.		Menlo Park
Interplay Productions	750 Ridder Park Dr.		Newport B
Minutemen Press	2150 Executive Dr.	Suite	San Mateo
Quantum Computer Servik	2082 Michelson Dr.	Suite	Vienna
Sceptre	10159 Alliance Rd.		Cupertino
Corporate Dynamics Inc.	1210 West Dayton St.		Redwood C
Hy-Ten	430 Clyde Avenue		Mountain W
Educational Resources	3431 Forest Circle		San Bruno
Kinetic Computing	1315 Bridgeway		Santa Ana
JPSA	3431 Forrest Bridge		Berkeley
American Paint	81 Norwood Ave.		West Ches
Signal Research	1120 Sharon Park Dr.		Cupertino
Leader Systems	10463 N. Blaney Ave.		San Rafael

Suppose you wanted to expand the abbreviations in these addresses: **St.** to **Street**, **Dr.** to **Drive**, etc. You could do this by using the **Change** command over and over again. Or you can simply use the `replacemultiple()` function to replace all of the abbreviations in one fell swoop.

**Enter the formula:** Cancel OK

```
replacemultiple(«Street
Address», "Rd./St./Dr./Ln./Ave.", "Road/Street/Drive/Lane/Avenue", "/")
```

Press **OK** to replace all of the abbreviations at once:

Company Name	Street Address	Suite Box	City
International Transportat	329 North State Street		Alameda
Pacific Micro	472 Wheelers Farms Road		Menlo Park
Interplay Productions	750 Ridder Park Drive		Newport B
Minutemen Press	2150 Executive Drive	Suite	San Mateo
Quantum Computer Servik	2082 Michelson Drive	Suite	Vienna
Sceptre	10159 Alliance Road		Cupertino
Corporate Dynamics Inc.	1210 West Dayton Street		Redwood C
Hy-Ten	430 Clyde Avenue		Mountain W
Educational Resources	3431 Forest Circle		San Bruno
Kinetic Computing	1315 Bridgeway		Santa Ana
JPSA	3431 Forrest Bridge		Berkeley
American Paint	81 Norwood Avenue		West Ches
Signal Research	1120 Sharon Park Drive		Cupertino
Leader Systems	10463 N. Blaney Avenue		San Rafael
Arrow, Inc.	390 Davis Street		Los Angele
ProLitho	215 Ace N		East Roche

122 visible / 122 total



# Chapter 13: Introduction to Forms



Panorama has two interfaces for displaying and editing data — the data sheet and forms. So far most of this manual has concentrated on using the data sheet. Starting with this chapter we'll introduce a much more flexible way to display and edit data: the form.

A single database only has one data sheet but it may contain many forms. You can design each form for a specific purpose, for example entering data, printing a mailing label, or printing a report.

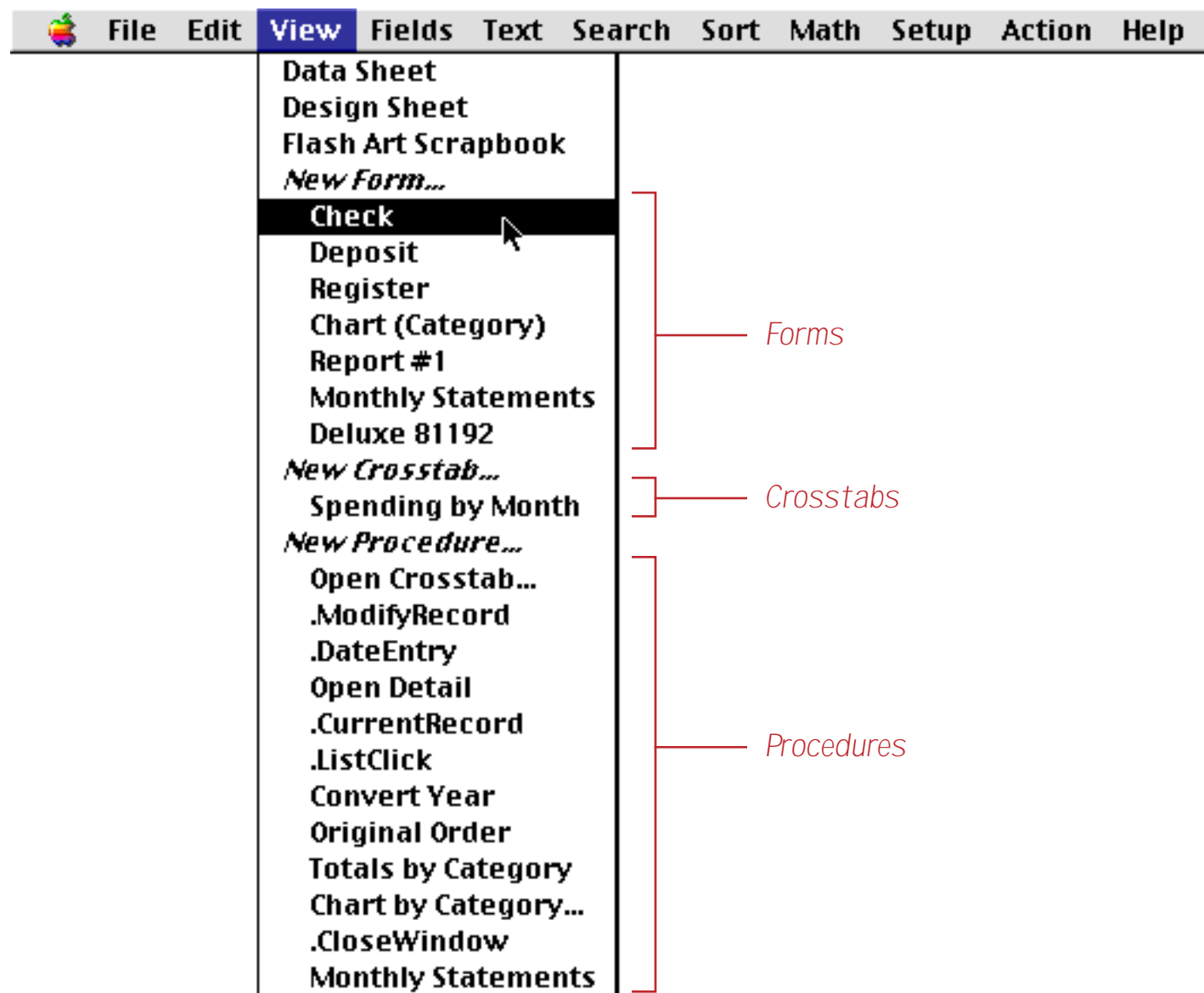
The data sheet displays a fixed format of rows and columns. You can change the text font and the width of the columns, but beyond that you don't have any control over the data sheet's appearance. Each form, on the other hand, is completely customizable. You can (and in fact must) set up the placement of each item on the form, including data, text and artwork. The form view is much more flexible than the data sheet view, but it is also more work to set up. Here is a typical example of a form. Notice that the window name shows the database name, [Checkbook](#), followed by the form name, [Plain Checks](#).

Ck #	Date	Amount
2239	08/20/90	85.00
2186	07/16/90	75.00
2104	05/22/90	115.00
2013	03/20/90	85.00
2012	03/20/90	25.00
1980	02/20/90	80.00
1979	02/15/90	125.00
1943	02/07/90	35.00
1942	02/07/90	75.00
1914	01/17/90	75.00
1908	01/08/90	75.00



## Opening a Form

The **View** Menu lists all the views in a database, including forms. The pre-defined views appear at the top—data sheet, design sheet, and flash art scrapbook. Next come the views you've created—forms, crosstabs, and procedures. The **View** Menu also contains commands for creating your own new views—new form, new crosstab, and new procedures.

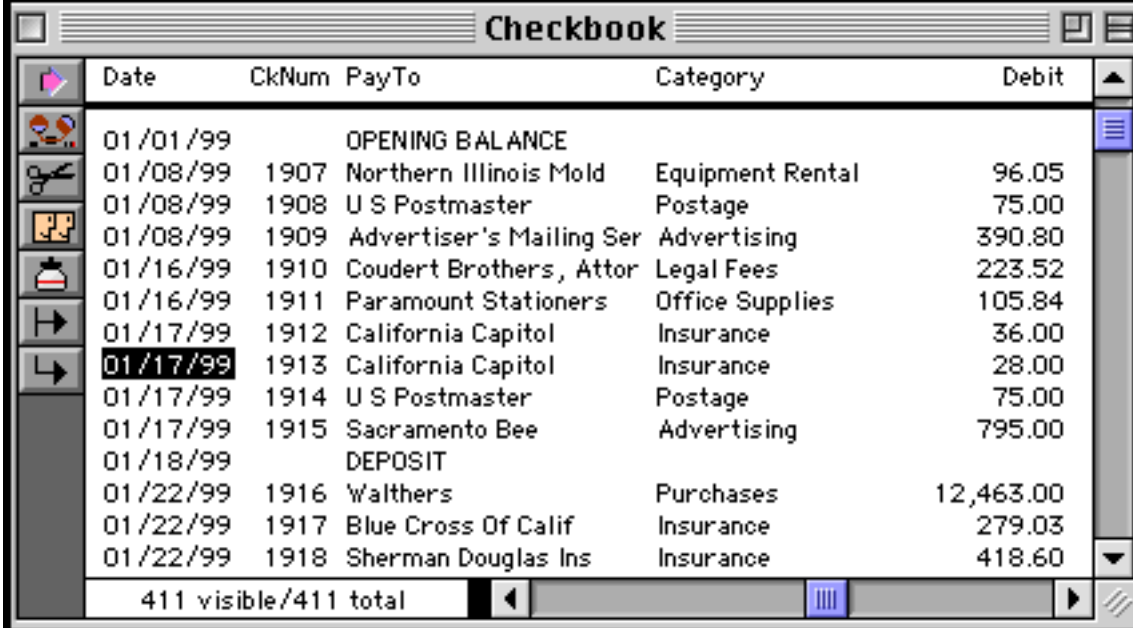


To open a form within the current window, simply choose the form from the menu. You can flip back and forth between different forms (or other views, like the data sheet) at any time.

## Opening A Form in a New Window

If you wish, you can open a form in a new window, allowing you to see two different views of the database at once. To open a form in a second window the same size as the current window, hold down the **Alt** key while you select from the View Menu. (If you are using a Macintosh, hold down the **Control** key.) The new window will appear slightly below and to the right of the original window.

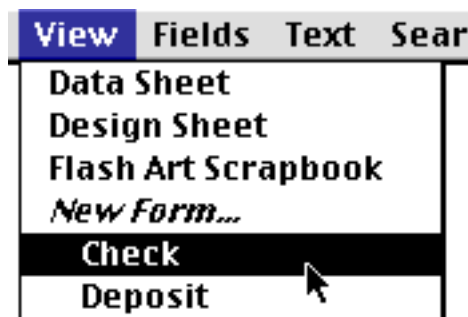
1) Start with one window



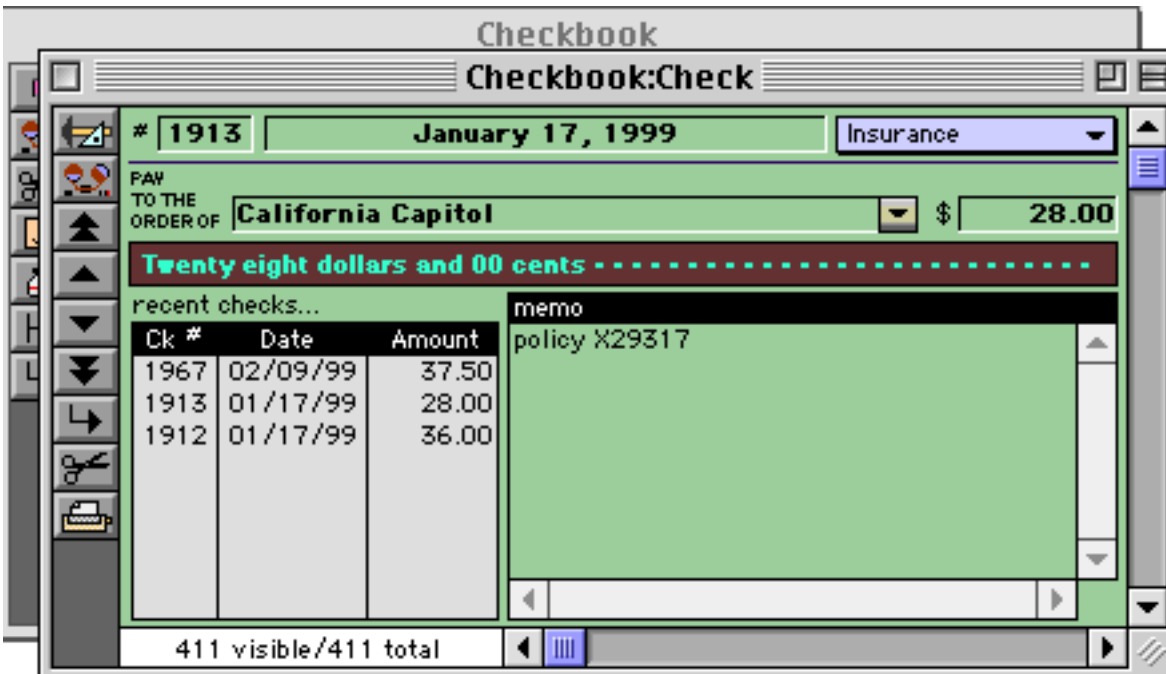
Date	CkNum	PayTo	Category	Debit
01/01/99		OPENING BALANCE		
01/08/99	1907	Northern Illinois Mold	Equipment Rental	96.05
01/08/99	1908	U S Postmaster	Postage	75.00
01/08/99	1909	Advertiser's Mailing Ser	Advertising	390.80
01/16/99	1910	Coudert Brothers, Attor	Legal Fees	223.52
01/16/99	1911	Paramount Stationers	Office Supplies	105.84
01/17/99	1912	California Capitol	Insurance	36.00
01/17/99	1913	California Capitol	Insurance	28.00
01/17/99	1914	U S Postmaster	Postage	75.00
01/17/99	1915	Sacramento Bee	Advertising	795.00
01/18/99		DEPOSIT		
01/22/99	1916	Walthers	Purchases	12,463.00
01/22/99	1917	Blue Cross Of Calif	Insurance	279.03
01/22/99	1918	Sherman Douglas Ins	Insurance	418.60

411 visible/411 total

2) While holding down the **Alt** key (PC) or the **Control** key (Mac), make a selection from the View menu.



3) The new window appears slightly below and to the right...



Checkbook:Check

# 1913      January 17, 1999      Insurance

PAY TO THE ORDER OF California Capitol      \$ 28.00

**Twenty eight dollars and 00 cents**

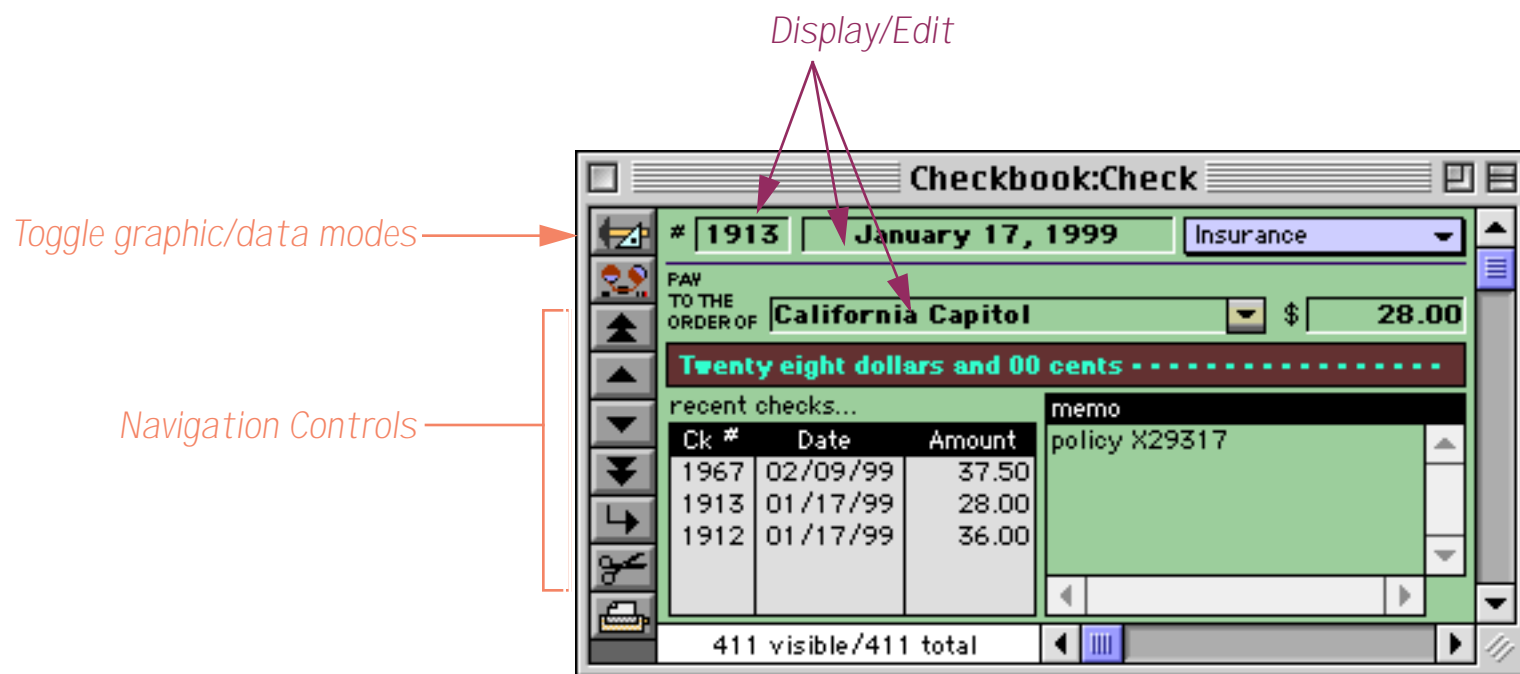
recent checks...	memo		
Ck #	Date	Amount	
1967	02/09/99	37.50	policy X29317
1913	01/17/99	28.00	
1912	01/17/99	36.00	

411 visible/411 total

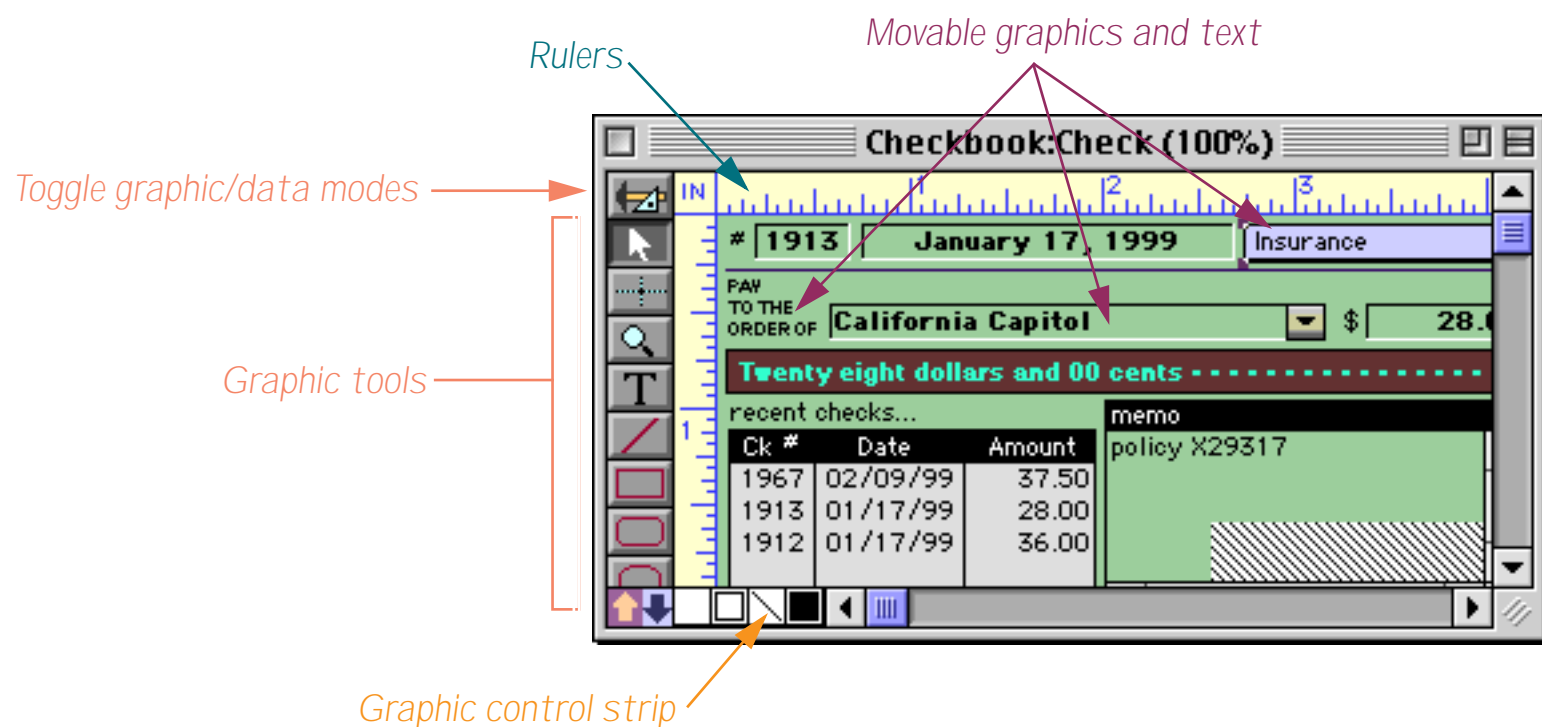
The new window will track the original window. Any changes made in this new window automatically appear in all other windows, and when any navigation is done in one window (moving up or down within the database) all of the other windows will follow along.


## Form Modes: Data Access vs. Graphic Design

Unlike other views, the Form View operates in two distinct modes—data access and graphic design. Data access mode (also called “data mode”) is the default mode. In this mode you can view and display data, and navigate through the database.



Graphic design mode (also called “graphics mode”) functions like an electronic drafting table. In this mode you design the form by drawing lines, boxes, and other graphic elements. This mode is very similar to many drawing and page layout programs. Graphic design mode is easily recognized by the rulers that appear at the top and left edges of the windows.

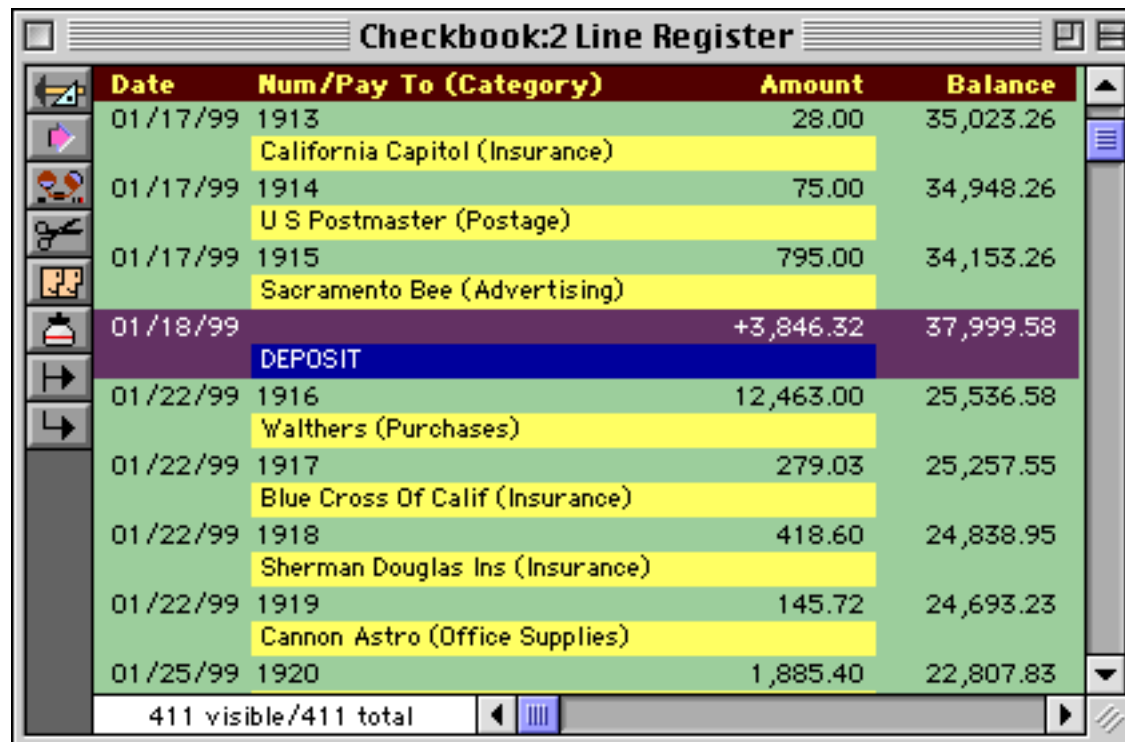


To switch between data access and graphic design modes, click on the  tool. Each click on this tool toggles the window between the two modes.

## Form Operation: Individual Pages vs. View-As-List

Panorama allows you to set up blank forms as individual pages or as a continuous sheet (*view-as-list*). When forms are set up as individual pages you see one record at a time. You can flip through the records just as you would shuffle through a stack of paper forms. All of the examples of forms you've seen so far are individual page forms.

A *view-as-list* form displays data as a continuous sheet, as shown below. Instead of flipping from record to record, you scroll up and down through the data in a manner similar to the data sheet. However, unlike the data sheet, a *view-as-list* form allows you to arrange the data any way you like, and even include graphics in the display. On the other hand, *view-as-list* forms are slower than the data sheet (because of the overhead in displaying the graphics) and they are much more work to set up.



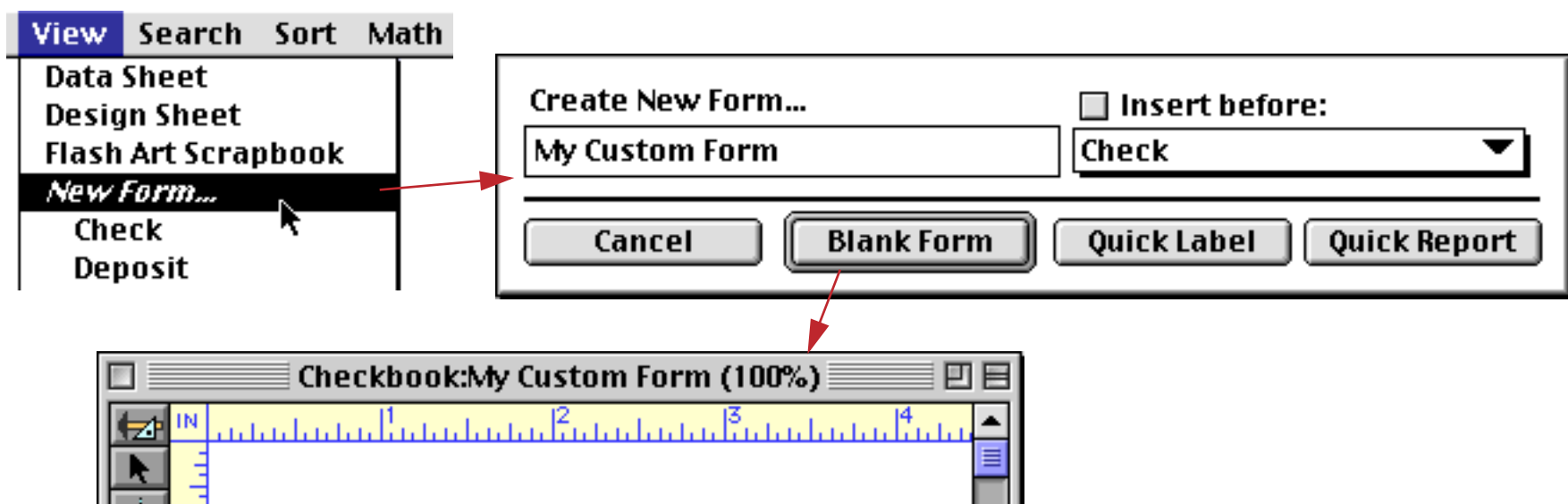
Date	Num/Pay To (Category)	Amount	Balance
01/17/99	1913 California Capitol (Insurance)	28.00	35,023.26
01/17/99	1914 U S Postmaster (Postage)	75.00	34,948.26
01/17/99	1915 Sacramento Bee (Advertising)	795.00	34,153.26
01/18/99	DEPOSIT	+3,846.32	37,999.58
01/22/99	1916 Walthers (Purchases)	12,463.00	25,536.58
01/22/99	1917 Blue Cross Of Calif (Insurance)	279.03	25,257.55
01/22/99	1918 Sherman Douglas Ins (Insurance)	418.60	24,838.95
01/22/99	1919 Cannon Astro (Office Supplies)	145.72	24,693.23
01/25/99	1920	1,885.40	22,807.83

411 visible/411 total

Unless you tell it otherwise, Panorama sets up a new form as individual pages. To convert the form to a continuous sheet you must use the **Form Preferences** command (Setup menu) to set the *View-as-List* option. You will also have to define the boundaries of the form by setting up a data tile (and optional header tile, see [“Adding a View-As-List Header”](#) on page 406). For more information about setting up *view-as-list* forms see [“Creating a View-As-List Form”](#) on page 402.

## Creating a New Form

To create a new view, choose **New Form** from the View Menu. A dialog box will appear asking you to name the new form. A form name may be up to 25 characters long and can contain any letter, number or punctuation.



When you create a new form, it usually becomes the last form in the **View Menu**. If you wish, you can insert the new view into the middle of the **View Menu**. To do this, check the **Insert before** button and use the pop-up menu directly below the **Insert before** button to specify the position of the new view. You can also rearrange the order of the forms using the **Re-Arrange Forms** command in the **Setup menu**. “[Changing the Order of Forms, Crosstabs or Procedures](#)” on page 93 for more information on this process.

## Renaming a Form

To rename the currently visible form choose **Rename Form** from the **Setup Menu**.

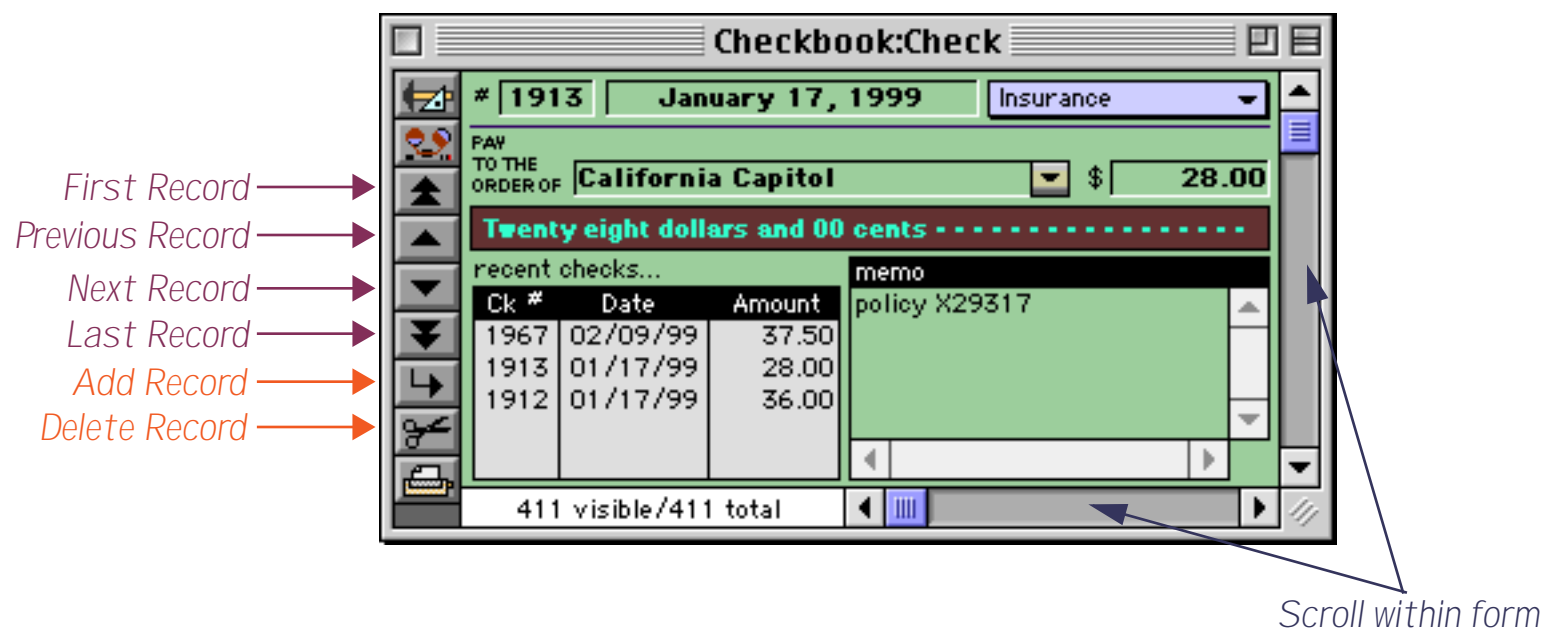
## Deleting a Form

To delete a form choose **Delete Form** from the **Setup Menu**. Since you cannot undo after you delete a view, Panorama will ask you if you are sure before it actually deletes the form. Note: If the form is the only window open for this database, Panorama will close the entire file when it deletes the form. To avoid this, open an additional window (the data sheet or another form) before you delete the form.

You can also delete forms with the **Re-Arrange Forms** command in the **Setup menu**. This is the fastest way to remove several forms at once.

## Browsing the Database With a Form

When working with a normal form (as opposed to a view-as-list form) Panorama displays one record at a time. You can navigate from record to record using the VCR style buttons in the tool palette.

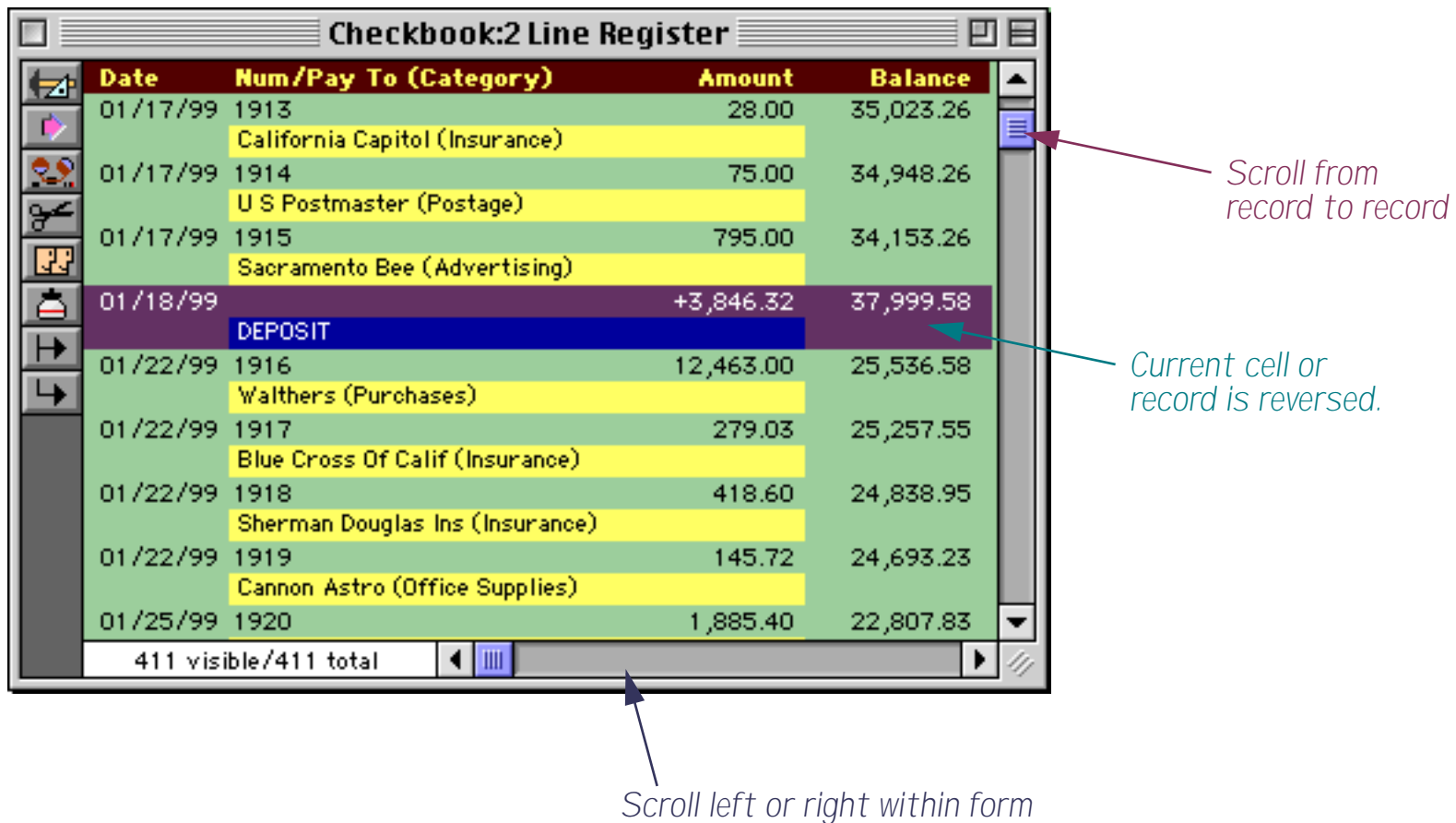


If the form is taller and/or wider than the current window, you can use the scroll bars on the bottom and right to slide the form around within the window. The form can also slide automatically as you tab from cell to cell within the form.



## Browsing the Database With a View-As-List Form

When working with a view-as-list form, Panorama display several records at a time — one row per record. You can click on any visible record to make it active, or use the vertical scroll bar to move to any record within the database (just like the vertical scroll bar in the design sheet). You can also use the up and down arrow keys to move up or down one record at a time.



If the form is wider than the window, you can use the horizontal scroll bar at the bottom of the window to slide the form left or right within the window. See “[View-As-List Forms](#)” on page 399 to learn how to create view-as-list forms.



# Chapter 14: Graphic Design








Panorama has a built in graphic editor for creating and modifying the layout of forms and reports. If you've used an object oriented graphic editor before you will find many familiar features.

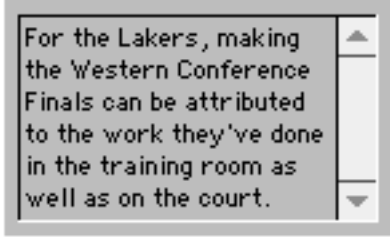











## Graphic Objects














Panorama forms are built with **graphic objects** (also called simply **objects**). Each object is treated as a unit (rather than as a collection of dots), and each object has a specific shape, position, size, and color. You can easily modify an object without disturbing the other objects—for instance sliding a rectangle to a new position or changing the diameter of a circle. Most of the next few chapters deals with techniques and shortcuts for arranging graphic objects on the surface of the form.

## Types of Graphic Objects

Panorama has over two dozen different types of graphic objects. Objects fall into five classes: Shapes, Text, Multi-Media, Buttons, and Layout. Each type of object has its own characteristics and appearance, as shown in the following table.

Class	Samples	Object	Tool	Description
<b>Shapes</b>		Line		Simple line at any angle.
		Rectangle		Simple rectangle, may be filled or transparent.
		Round Rectangle		Rectangle with curved corners.
		Oval		Oval (or circle), may be filled or transparent.

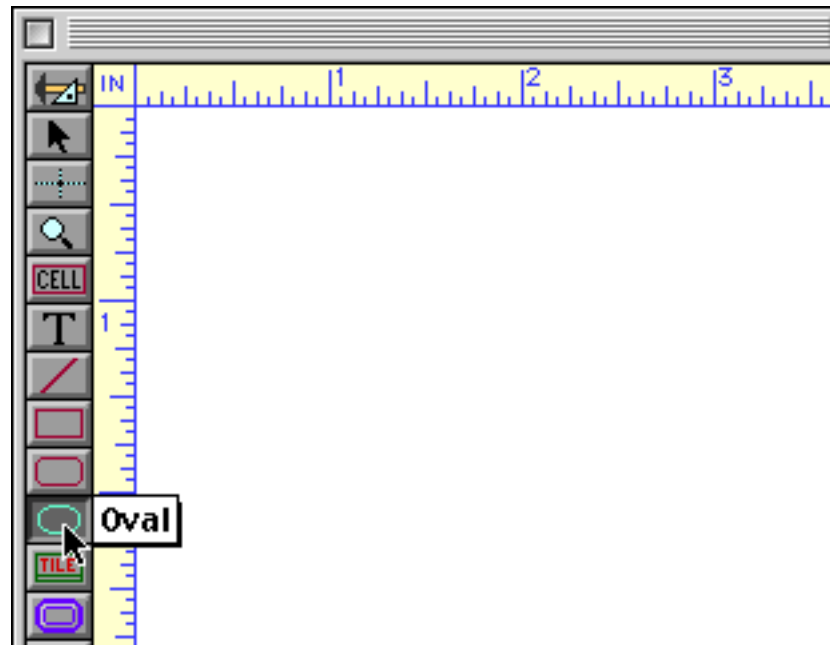
Class	Samples	Object	Tool	Description
Text	<p><b>Quantity</b></p> <p>All prices are F.O.B. Los Angeles, California. Terms are 30 days on approved credit, with a 2% discount for payment within 10 days.</p> <p><b>Name</b></p> 	Click Text		Displays simple text captions.
		Auto Wrap Text		Displays one or more paragraphs of text. May contain fields or variables merged within the text.
		Text Display		Displays text based on a formula. The text can scroll within the object, may be aligned in 9 different positions within the object, and can scale based on the size of the form.
		Data Cell		Used for editing fields. When double clicked, an expandable pop-up editing box appears (similar to editing in the data sheet).
		Text Editor		Used for editing fields or variables. Unlike the Data Cell, there is no pop-up editing box (more like other software applications).
		Word Processor		Used for editing a field or variable containing stylized text. The text may contain different fonts, sizes, styles, margins and tab stops.
Multi-Media		Picture	none	Displays a fixed image. May be used for backgrounds, logos, etc.
		Flash Art		Displays changing images (PICT format) based on a formula. Images may be part of the database or stored as separate files on the disk.
		Super Flash Art		Displays images or QuickTime movies with advanced features like scroll bars, advanced alignment and scaling, and hypertext.
		Chart		Displays column, bar, line, area, scatter and pie charts.
		Flash Sound		Automatically plays sound (Macintosh only).

Class	Samples	Object	Tool	Description
<b>Buttons</b>		Button		Generic button tool for creating push buttons, radio buttons and checkboxes (fields only). (For new applications we recommend the new button tools listed below.)
		Push Button		2 and 3 dimensional push buttons in a variety of styles.
		Data Button		2 and 3 dimensional checkboxes and radio buttons in a variety of styles. These buttons may be tied to a field or a variable.
		Sticky Push Button		2 and 3 dimensional buttons that look like push buttons but act like checkboxes or radio buttons.
		Pop-Up Menu		Pop-up menu tied to a field or variable. May use any font, text size, color, or number of columns.
		List		Scrollable list.
		Flash Art Push Button		Push button with custom artwork.
		Flash Art Data Button		Checkbox or radio button with custom artwork.
		Scroll Bar		Standalone scroll bar.
		<b>Layout</b>		Tile
Super Matrix				Display a repeating matrix that may contain graphics and data. Options include grid lines and the number of rows and columns.
Auto Grow				Adjusts other objects as window size changes, making the form "elastic."

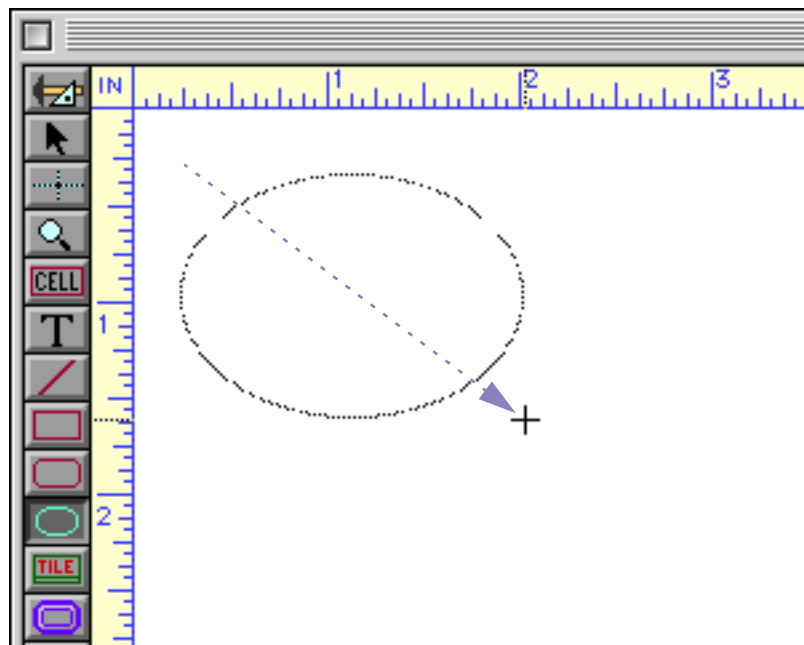


## Creating a Graphic Object

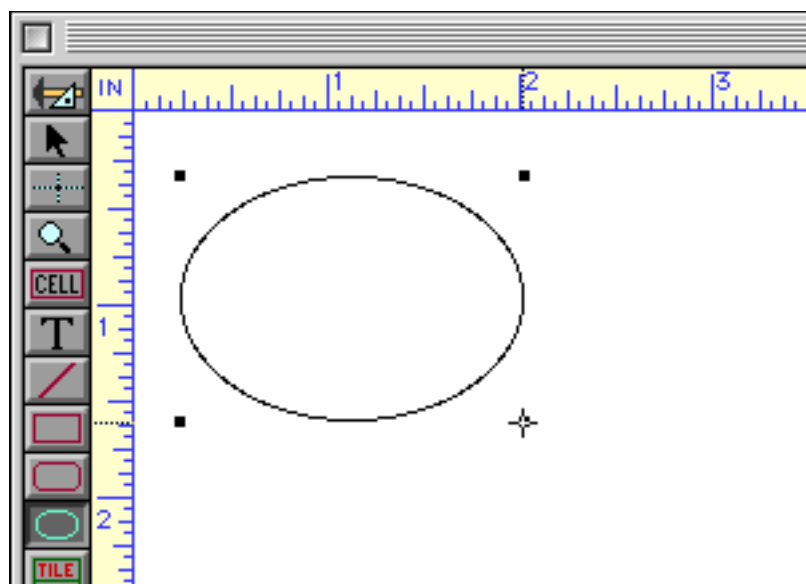
To create a new graphic object, first click on the appropriate tool in the tool palette. For example, to draw an oval you would click on the **Oval** tool.



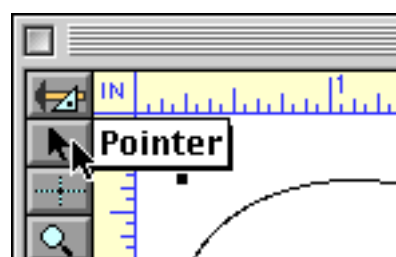
Then move the cursor onto the form and drag the mouse across the surface of the form to define the location and size of the new object (the dragging motion is shown by the dashed arrow in the illustration below). A gray outline of the new object will follow the mouse.



When you release the mouse, the new object will appear.

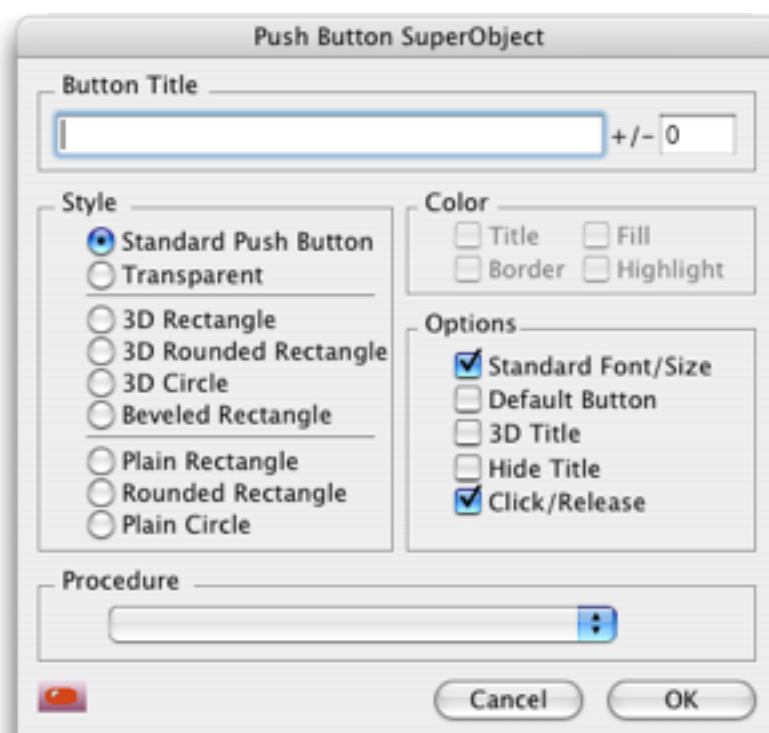


Each time you drag across the form you will create a new shape. When you are finished creating shapes, click on the **Pointer** tool.



**Don't forget to click on the Pointer tool when you are done!** If you don't, the next time you click you will create another graphic object.

The procedure for creating more complex objects is the same as for simple objects: 1) select the tool, 2) drag the mouse across the form. However, when you release the mouse after creating a more complex object a dialog will appear allowing you to configure the new object. Each type of complex object has its own dialog. For example, here is the dialog for creating a push button.



Rather than describing the dialog for each type of item here, each will be described in detail later along with the corresponding objects.

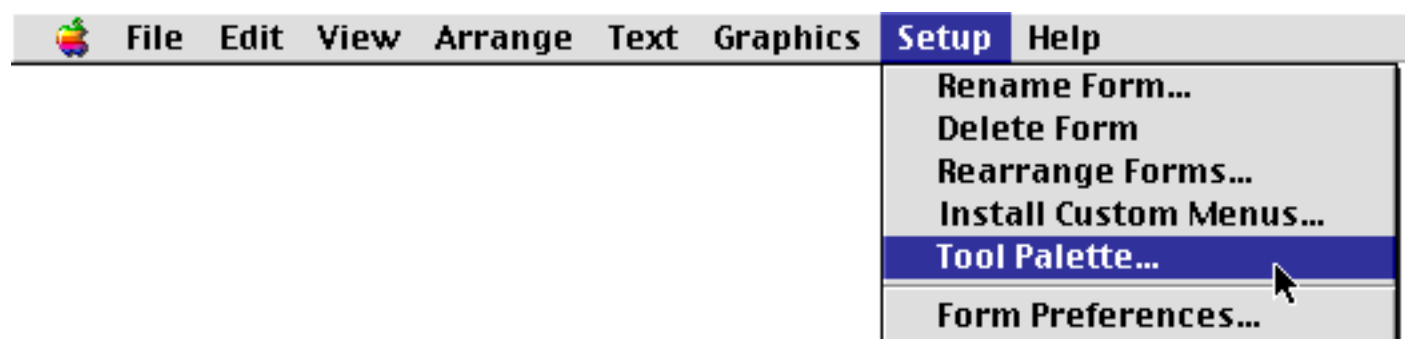
## Creating Perfect Squares, Circles and Lines

If you press the **Shift** key while you create a rectangle or oval, Panorama will automatically force the new shape to be a perfect square or circle. If you press the **Shift** key while creating a line Panorama will force the alignment of the new line to a multiple of 45 degrees (0°, 45°, 90°, 135°, etc.).

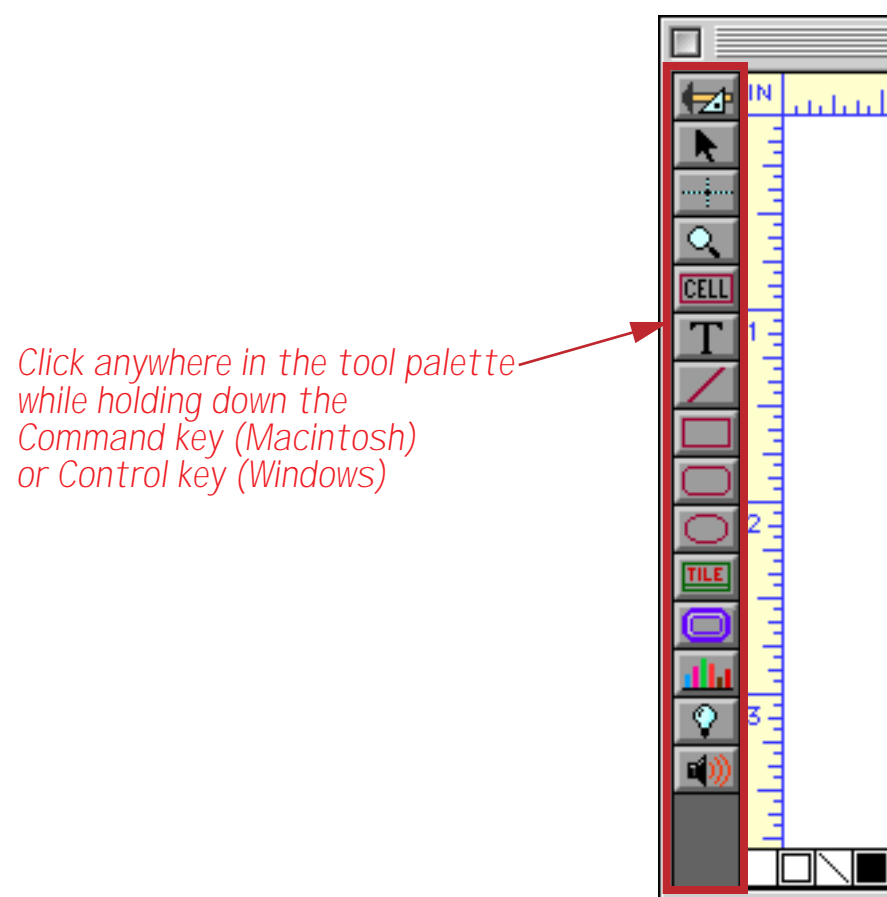
## Customizing the Tool Palette

There are a total of 29 graphic tools available for use in Panorama. Many computer screens are not large enough to handle this complete palette of tools (and we expect the number of tools to increase in future versions). To get around this problem, Panorama allows you to customize the graphic tool palette on the fly. You can configure the palette to contain only the tools that you need right now in any order you want. If your needs change later, you can simply reconfigure the palette at any time.

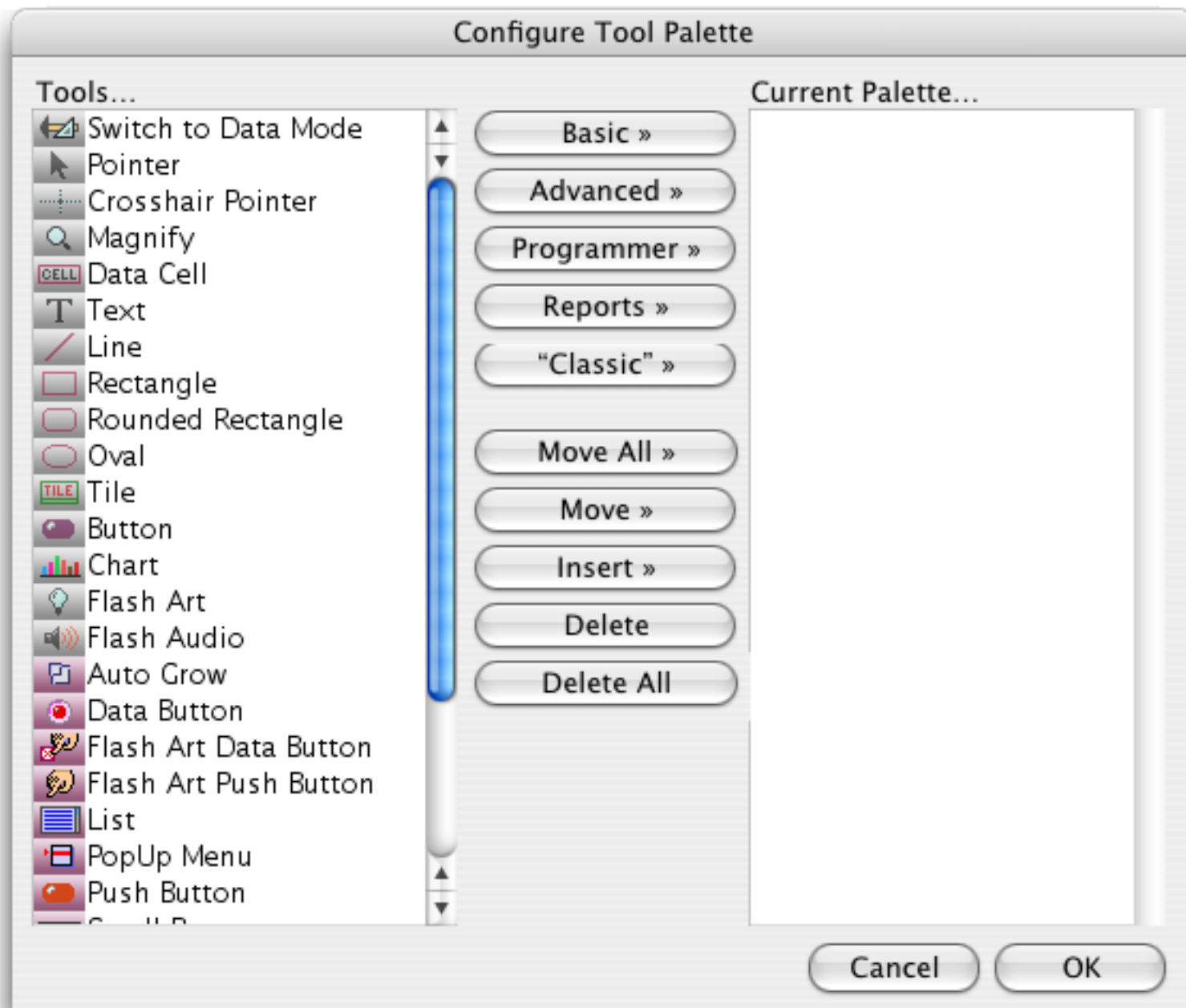
To customize the graphics tool palette, choose **Tool Palette** from the Setup menu.



**Tip:** You can also open this dialog by holding down the **Command** key (Macintosh) or **Control** key (Windows) and clicking anywhere in the graphic tool palette.



The Configure Tool Palette dialog contains two lists of tools. On the left is a list of all tools available.



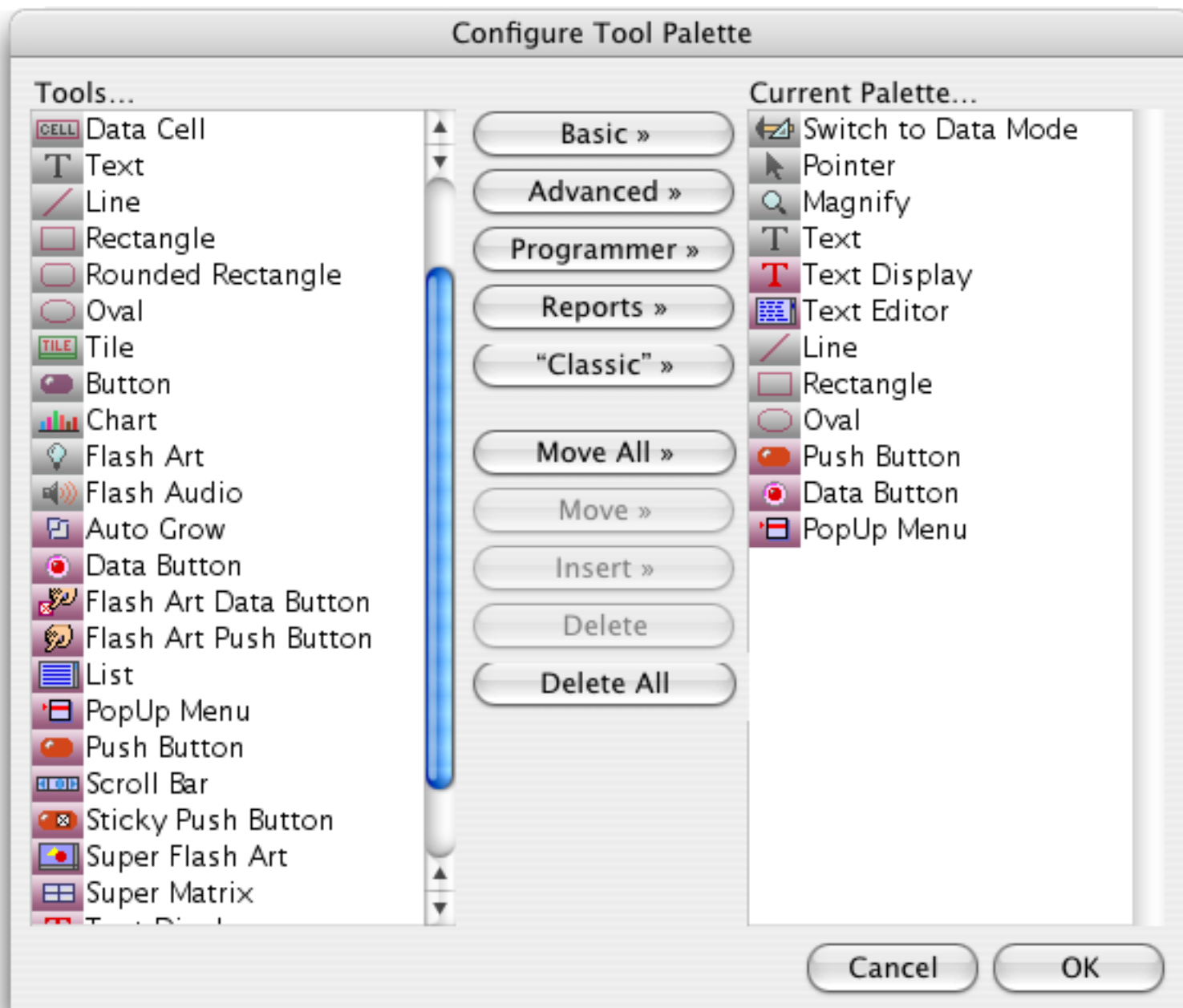
On the right is a list of the tools you currently have installed. If this list is empty, Panorama will use the default tool palette (which is the original tool palette from Panorama II). Tools can be moved into the palette on the right either individually (one at a time) or in groups. To move an individual tool from the left to the right, double click on the tool in the left. Or you can select the tool (or tools) and press the **Move** or **Insert** button.

To move an entire group of tools at once press the **Basic**, **Advanced**, **Programmer**, **Reports** or **“Classic”** buttons. The table below shows the tools included by each of these options.

Basic	Advanced	Programmer	Reports	“Classic”
<ul style="list-style-type: none"> <li>Switch to Data Mode</li> <li>Pointer</li> <li>Crosshair Pointer</li> <li>Magnify</li> <li>Data Cell</li> <li>Text Editor</li> <li>Text</li> <li>Text Display</li> <li>Line</li> <li>Rectangle</li> <li>Rounded Rectangle</li> <li>Oval</li> <li>Push Button</li> <li>Data Button</li> </ul>	<ul style="list-style-type: none"> <li>Switch to Data Mode</li> <li>Pointer</li> <li>Crosshair Pointer</li> <li>Magnify</li> <li>Data Cell</li> <li>Text Editor</li> <li>Text</li> <li>Text Display</li> <li>Line</li> <li>Rectangle</li> <li>Rounded Rectangle</li> <li>Oval</li> <li>Tile</li> <li>Push Button</li> <li>Data Button</li> <li>Sticky Push Button</li> <li>PopUp Menu</li> <li>Super Flash Art</li> <li>Word Processor (Cell)</li> </ul>	<ul style="list-style-type: none"> <li>Pointer</li> <li>Crosshair Pointer</li> <li>Magnify</li> <li>Text Editor</li> <li>Text Display</li> <li>Line</li> <li>Rectangle</li> <li>Oval</li> <li>Super Flash Art</li> <li>Word Processor (Cell)</li> <li>Button</li> <li>Push Button</li> <li>Data Button</li> <li>Sticky Push Button</li> <li>PopUp Menu</li> <li>List</li> <li>Flash Art Push Button</li> <li>Flash Art Data Button</li> <li>Scroll Bar</li> <li>Super Matrix</li> <li>Auto Grow</li> </ul>	<ul style="list-style-type: none"> <li>Switch to Data Mode</li> <li>Pointer</li> <li>Crosshair Pointer</li> <li>Magnify</li> <li>Text</li> <li>Text Display</li> <li>Line</li> <li>Rectangle</li> <li>Rounded Rectangle</li> <li>Oval</li> <li>Tile</li> <li>Super Flash Art</li> </ul>	<ul style="list-style-type: none"> <li>Switch to Data Mode</li> <li>Pointer</li> <li>Crosshair Pointer</li> <li>Magnify</li> <li>Data Cell</li> <li>Text</li> <li>Line</li> <li>Rectangle</li> <li>Rounded Rectangle</li> <li>Oval</li> <li>Tile</li> <li>Button</li> <li>Chart</li> <li>Flash Art</li> <li>Flash Audio</li> </ul>

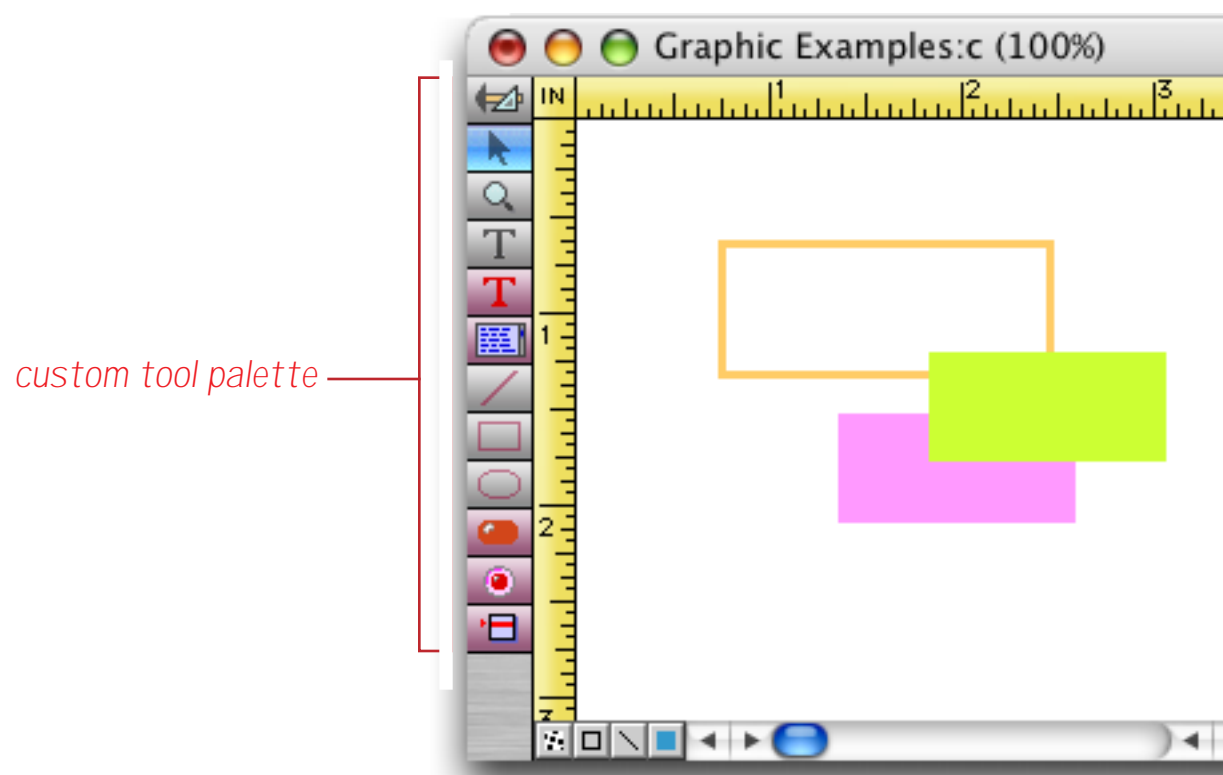
To delete one or more tools from the right hand list, select the tool(s) and press the **Delete** button. You can also delete a tool from the right hand list by double clicking on it. The **Delete All** button clears the list on the right so you can start over or go back to using the default tool palette.

Here is an example of a custom palette configuration:




















When you press the **OK** button the new palette configuration will become active, like this.



If you need to change the tool palette again later, just open the dialog again and make the necessary adjustments.

### Using the Keyboard to Select Common Tools

Usually you will use the mouse to select the tool you want to use. The most common tools, however, can also be activated with the keyboard. This saves you a trip to the tool palette each time you want to select one of these tools. The table below lists the tools that can be selected with the keyboard. Note that these keys are pressed by themselves — not in combination with any other key.

Tool	Key	Notes
		Press the <b>Escape</b> key to toggle between Graphics Mode and Data Access Mode (disabled if tool palette has been disabled)
		Press the <b>P</b> key to select the Pointer tool (except when typing or editing text with the T tool).
		Press the <b>Enter</b> key to select the Pointer tool (at any time, even when typing or editing text with the T tool).
		Press the <b>=</b> key to toggle the crosshair cursor on/off
		Press the <b>T</b> key to select the Text tool
		Press the <b>L</b> key to select the Line tool
		Press the <b>R</b> key to select the Rectangle tool
		Press the <b>O</b> key to select the Oval tool

## SuperObjects

The astute observer will notice that the list of tools in the configuration dialog is divided into two groups - gray tools and purple tools. The purple tools are actually not built in to Panorama, but are special plug-ins called **SuperObjects**. Because they are written as plug-in tools, ProVUE can develop new SuperObjects faster and with more capabilities than for standard objects. You can expect to see many more SuperObjects added to Panorama in the future. You may also notice that some SuperObjects perform functions similar to regular objects, but with more features. However, as far as you, the user, are concerned, you don't really have to worry about whether an object is a SuperObject or not. The techniques for creating and modifying SuperObjects and regular objects are the same.

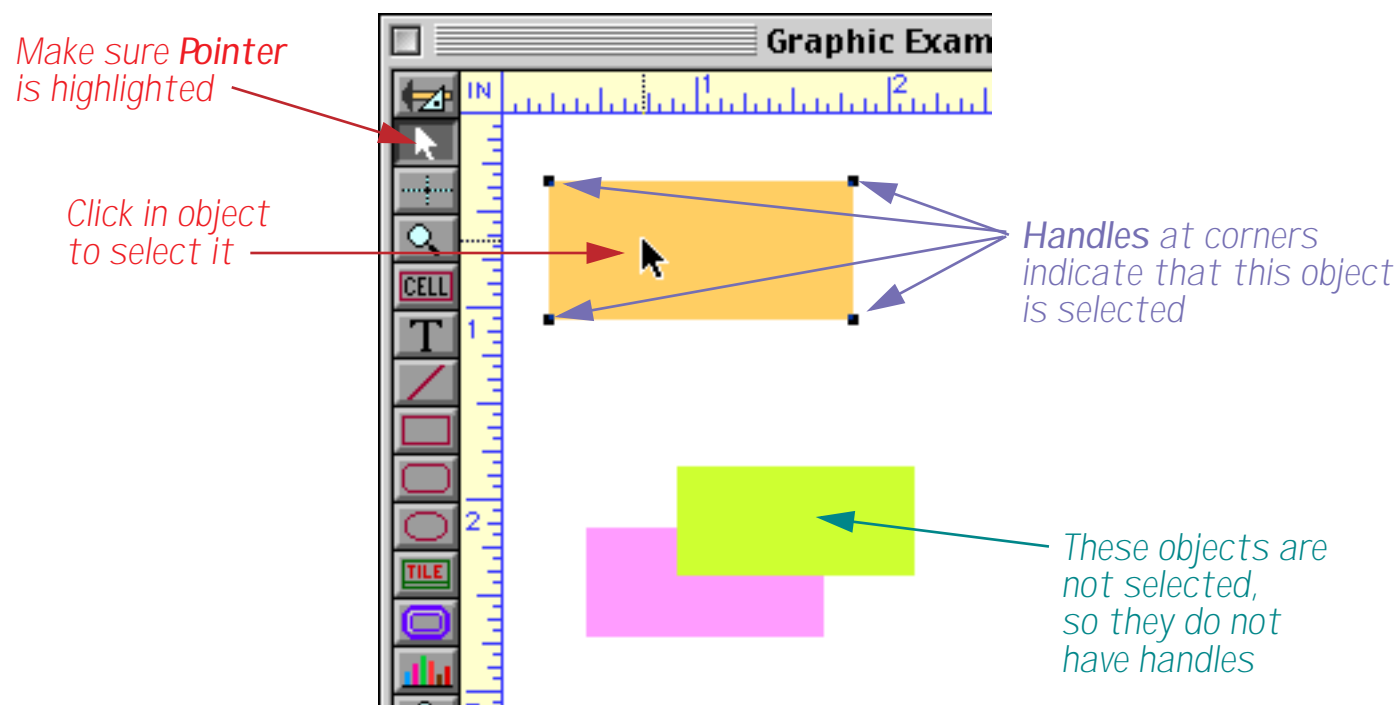
## Modifying Objects

Once an object is created it is far from being cast in stone. At any time you can go back and move, resize, change the color, change the alignment, or make virtually any other change. You can change objects one at a time or in groups. About the only change you cannot make is changing an object into another type of object (for example, you cannot change a square into a circle or change a rectangle into a pop-up menu).

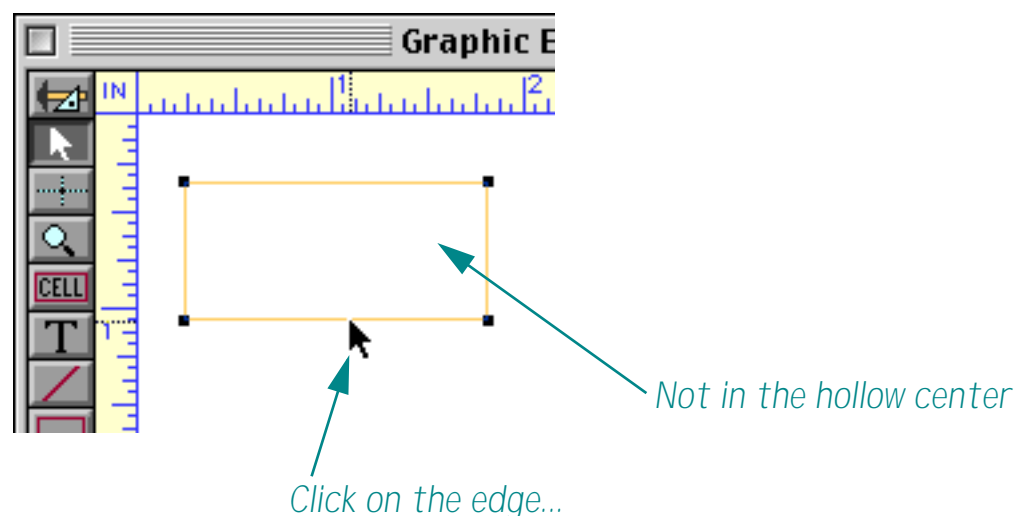
## Selecting a Single Object

Before you can modify an object (change its size, color, pattern, etc.) you must select the object. Selecting an object (or objects) tells Panorama that you want to work with that object. The **Pointer** tool is used for selecting objects. If the **Pointer** tool is not highlighted, click on it before you try to select an object. One of the most common mistakes made by new users is to try to select an object when a tool other than the **Pointer** tool is highlighted.

There are two ways to select an object. The simplest is to click on the object. When an object is selected, **handles** appear at the corners of the object. The **handles** let you know the object is selected and waiting for you to do something with it.



If an object is hollow (transparent, or filled with NONE) you must click on the border of the object to select it. Objects with thin borders may be difficult to click on. If you find it too difficult, remember that you can also select an object by dragging a selection marquee around it (see the next section).



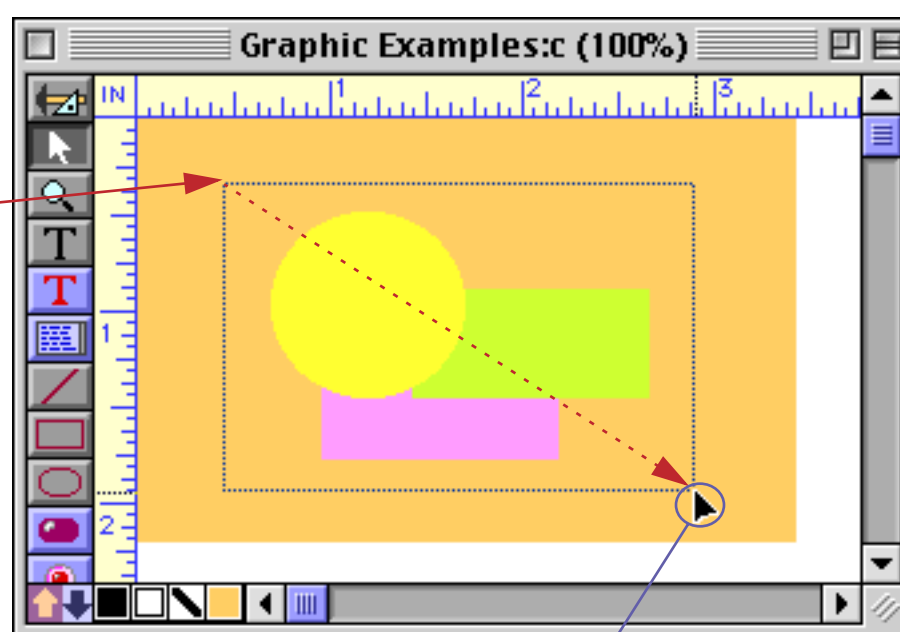
It's possible for one object to be hidden behind another object, making it impossible to click on. See "[Selecting a Completely Hidden Object](#)" on page 279 to learn how to select hidden objects.

### Selecting Multiple Objects at Once

Sometimes you may want to modify several objects at a time. You can select multiple objects by clicking on each object while holding down the **Shift** key, or by dragging a **selection marquee** around the objects. (You can also unselect an object that is already selected by holding down the **Shift** key and clicking on it.) The selection marquee is simply a dotted line that appears when you drag the pointer across the surface of a form. The marquee is like a lasso—it selects any object that is completely enclosed within it.

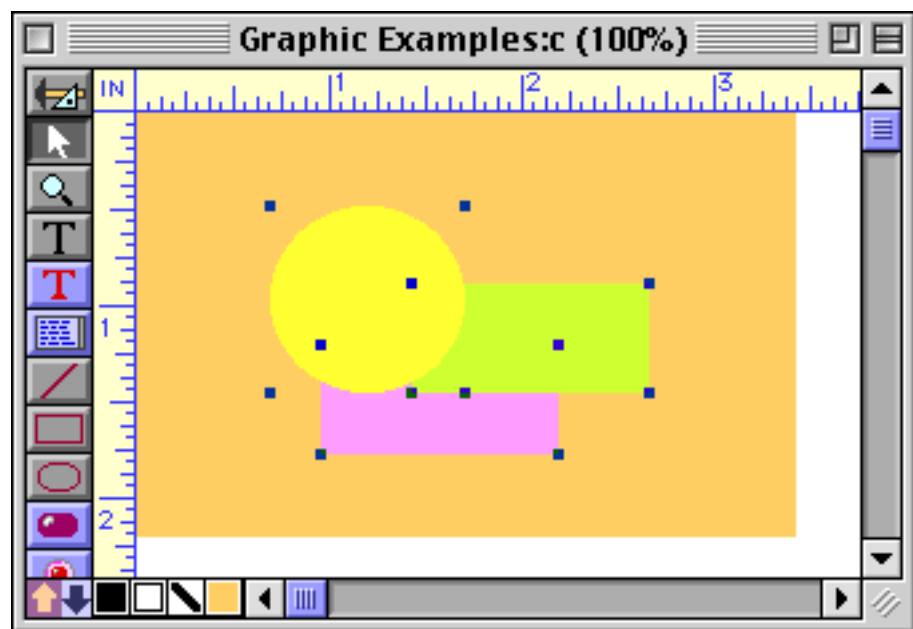
To drag a marquee you normally need to start on an empty spot on the form. If an empty spot isn't convenient, just hold down the **Space Bar** and drag the marquee. Holding down the **Space Bar** removes the stem from the cursor arrow and disables clicking on objects, allowing you to drag a marquee anywhere. Either way, drag the marquee all the way around the objects you want to select. Only objects that are completely inside the marquee will be selected. In this example we'll select the three smaller objects but not the large orange rectangle.

*Hold down Space Bar and click anywhere to start dragging, even on top of an object*

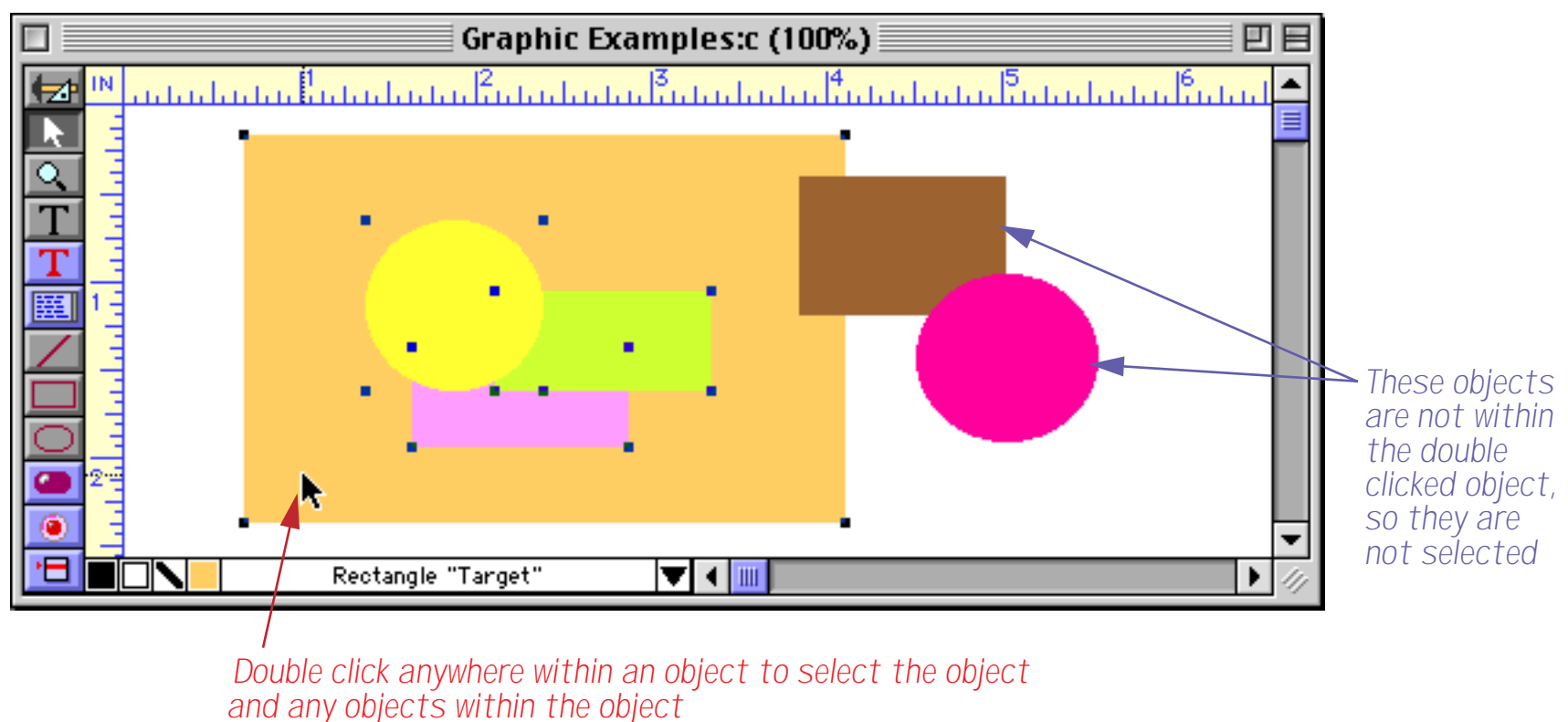


*When the Space Bar is pressed, the cursor arrow loses its tail*

When you release the mouse, the three objects inside are selected. The large orange rectangle is not selected and has not moved.



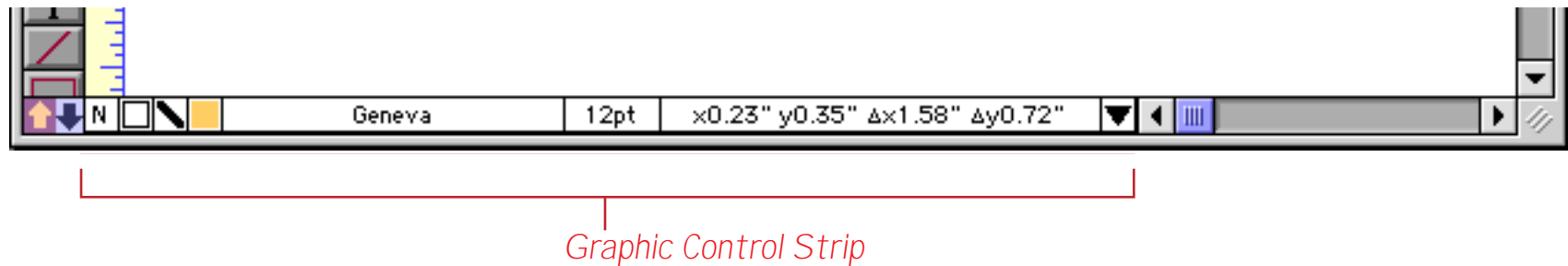
Double-clicking is another shortcut for selecting multiple objects. Double-clicking on an object selects all the objects inside the object as well as the object itself.



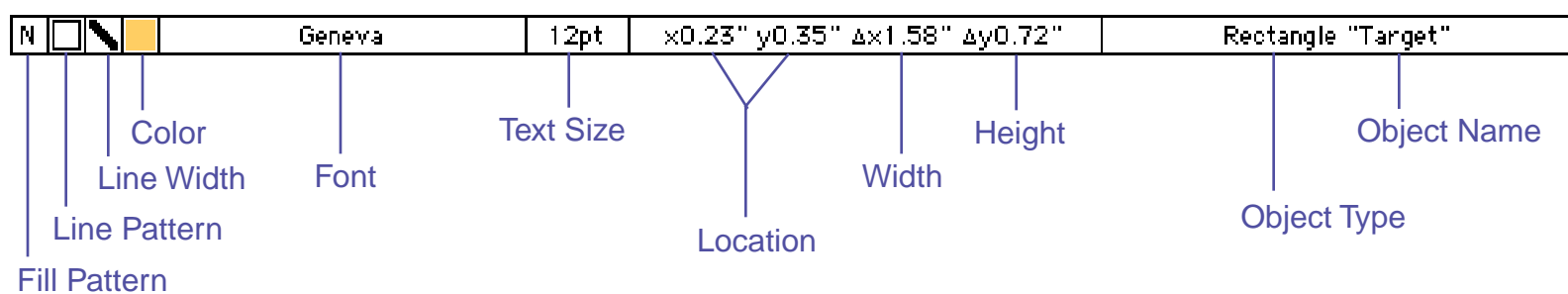
There's one more way to select multiple objects. The **Select All Objects** command in the Edit menu will select every object in the form. (To unselect all objects, click on an empty spot within the form.)

## The Graphic Control Strip

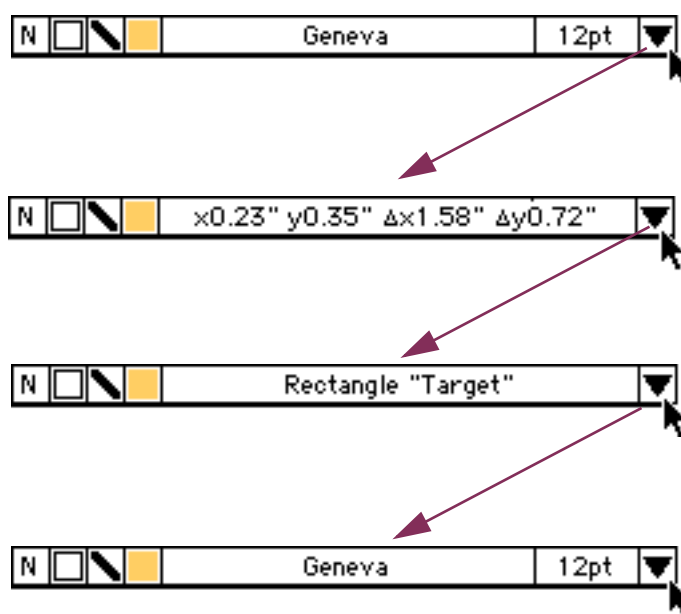
When a Panorama form window is in graphic design mode, a **graphic control strip** usually appears along the bottom of the window. The graphic control strip occupies some of the space that is normally used by the horizontal scroll bar. (If the window is too narrow for both the horizontal scroll bar and the graphic control strip, the control strip will disappear.)



The graphic control strip displays information about the currently selected object or objects (if any), and also allows you to easily change some of the properties of the currently selected objects with pop-up menus and dialogs. The complete graphic control strip has eleven elements.



If the window is not wide enough for all seven of these elements, the control strip will automatically adjust to show fewer elements. When this happens, an extra triangle icon appears at the end of the control strip. Clicking on this icon cycles through the **Font/Text Size**, **Dimension**, and **Object Type/Object Name** control strip elements (the first four elements are always visible unless the window is extremely narrow).

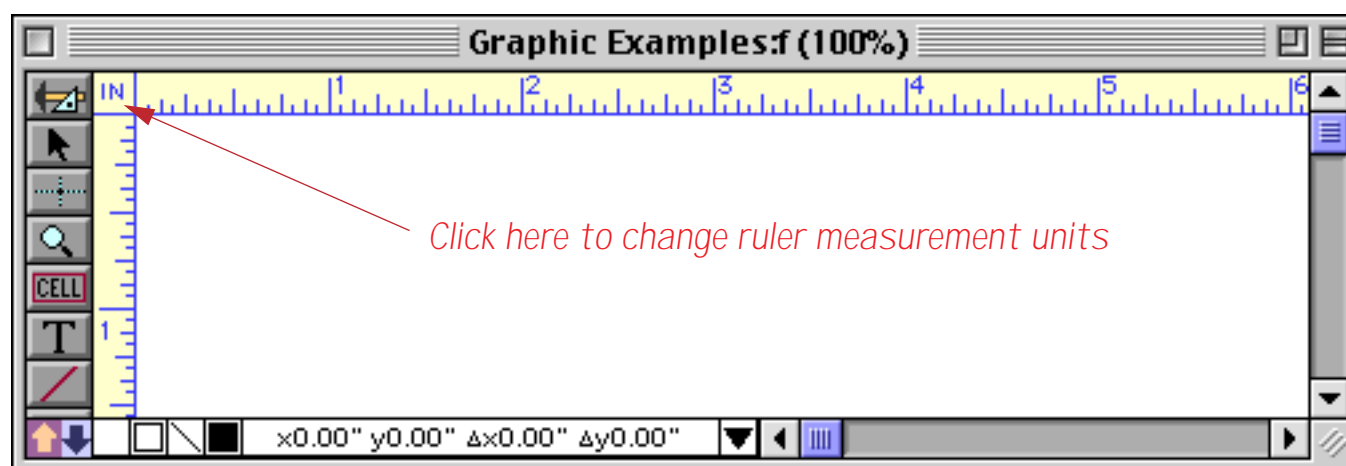


## Rulers

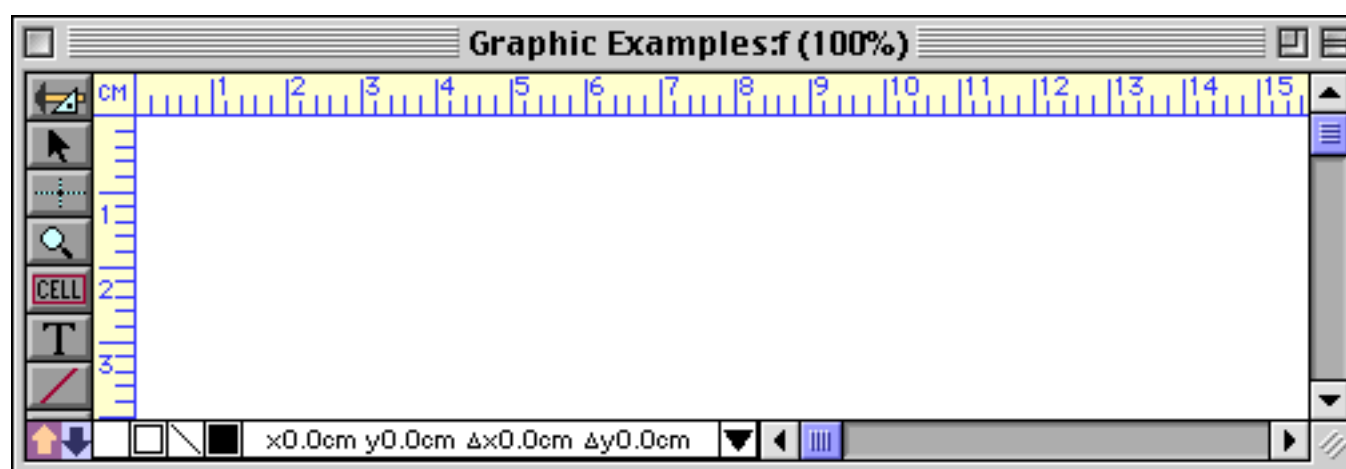
The graphic editor always displays rulers along the top and left sides of the graphic window. The rulers always start from zero in the upper left hand corner of the entire form (not the window). An indicator in each ruler follows the mouse as you move it across the form. These indicators help you measure objects. Note: The rulers do not represent where an object will be printed on a piece of paper. They are only a convenience for sizing and positioning objects on the screen.



The rulers usually show measurements in inches. Click on the box in the upper left corner to toggle between different measurement units—inches, centimeters, pixels, pica and elite.



Clicking once changes the ruler to centimeters.

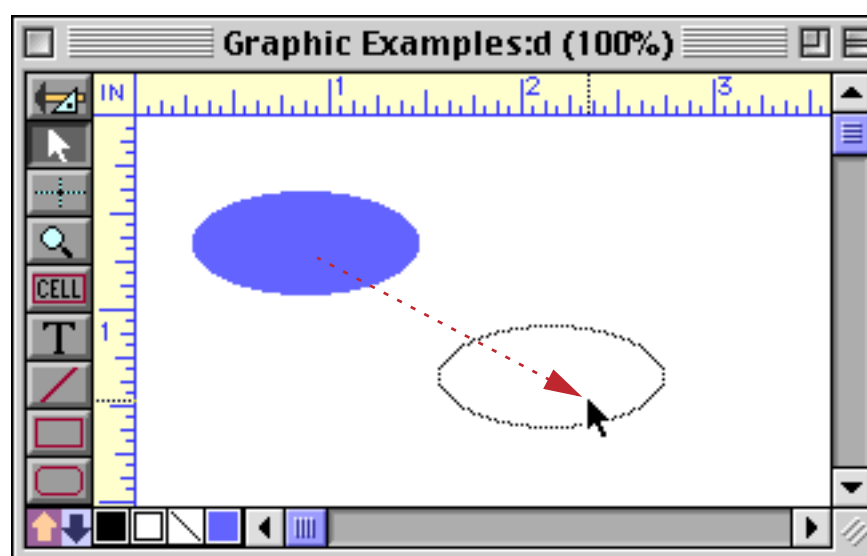


A second click changes the ruler to pixels. A pixel is one screen dot, or 1/72 inch. Additional clicks switch to Pica (1/6 inch) and elite (1/12 inch), then back to inches.

You can also change the measurement units with the **Form Preferences** dialog in the Setup menu. This permanently changes the default measurement units for the current form.

### Moving a Single Object

There are several ways to move a single object, including dragging, nudging, and using the dimensions dialog. To drag an object, the **Pointer** tool must be highlighted. Press on the object, then drag the object to its new position. If you drag the object near the edge of the window, the form will automatically scroll.



When you release the mouse, the object will move to the new position. If an object is hollow (transparent, or filled with NONE), you must click on the border of the object to drag it.

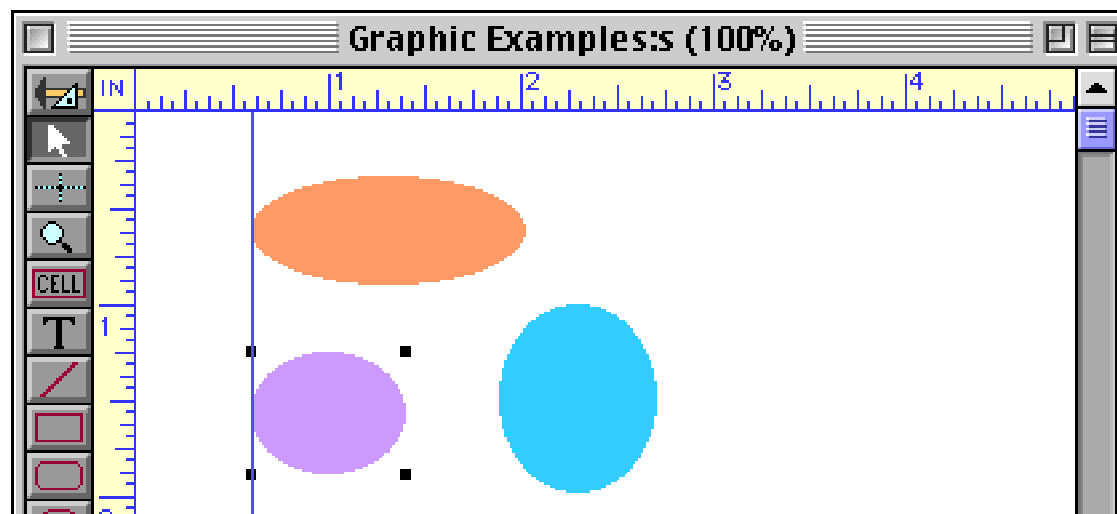
If you want to move the object horizontally or vertically (but not diagonally), hold down the **Shift** key as you drag the object. When the **Shift** key is held down you won't be able to drag an object diagonally. If you don't have far to go you might consider nudging the object instead of using the mouse (see below). This allows you to exactly position the object or handle in one pixel (or less) increments.

### Nudging an Object (or Objects)

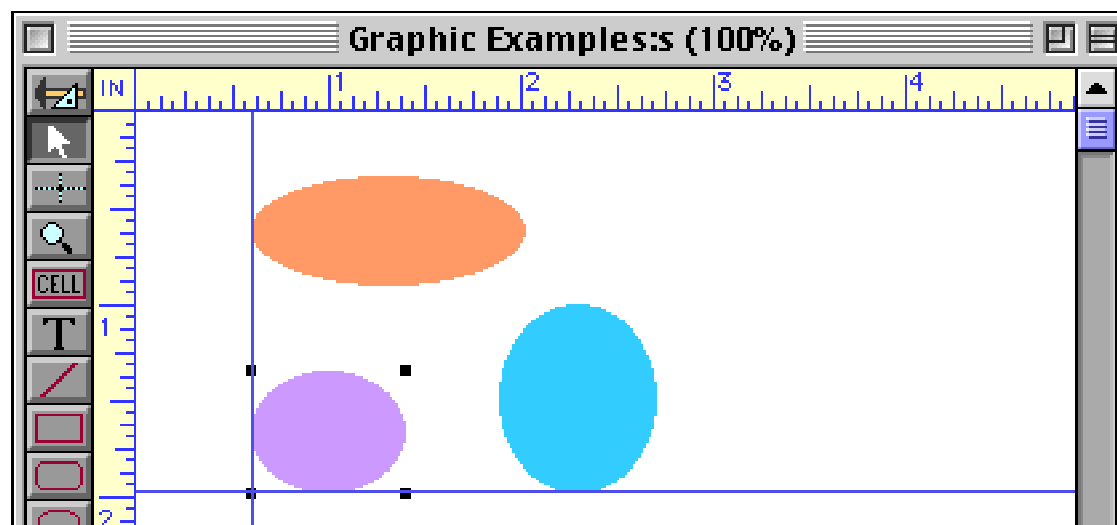
You can use the arrow keys (←, →, ↓, ↑) to nudge selected objects into position. Each time you press an arrow key, the object (or objects) moves one pixel in the direction of the arrow.

### Nudge "Auto Guides"

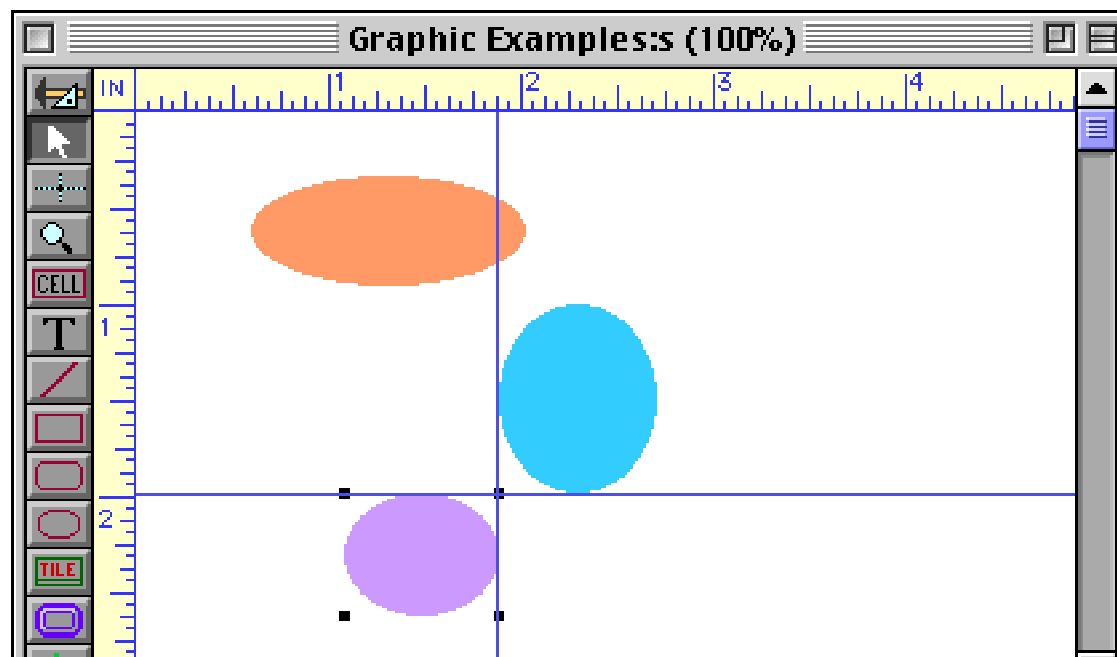
As you nudge an object (or objects) Panorama checks to see if the object is aligned with any other objects on the form. When an alignment occurs a blue guide line briefly appears. In this illustration the guide line appeared as the purple oval was nudged to the left, the guide automatically appeared when the left edge of the purple oval was aligned with the left edge of the pink oval.



The automatic guide will disappear when you click the mouse or press any key, or it will simply disappear by itself after a few seconds. If more than one edge is aligned the multiple guide lines will appear, like this.



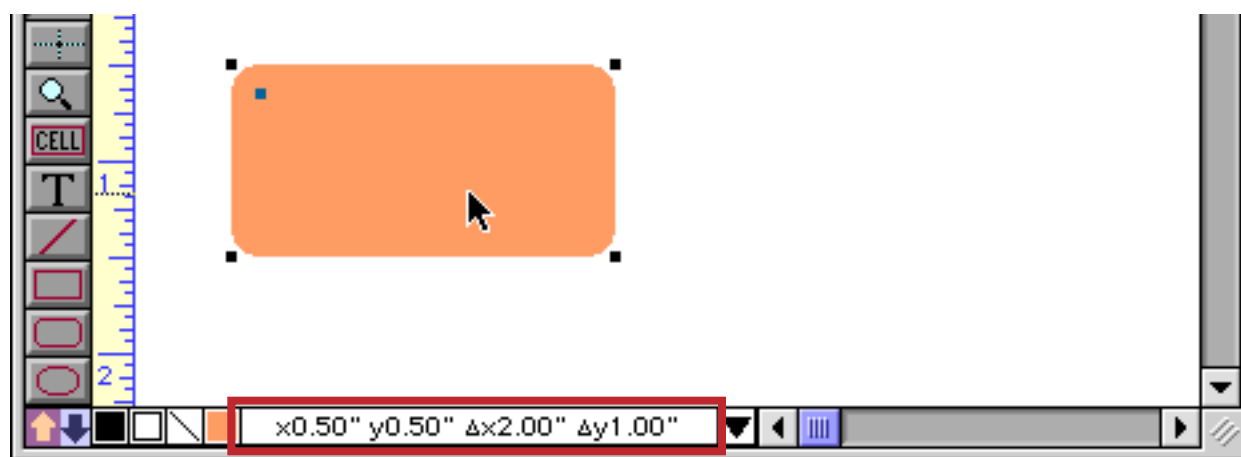
The alignment doesn't have to be top to top or left-to-left, if any edge of the nudged object(s) aligns with any edge of any other object the guides will appear.



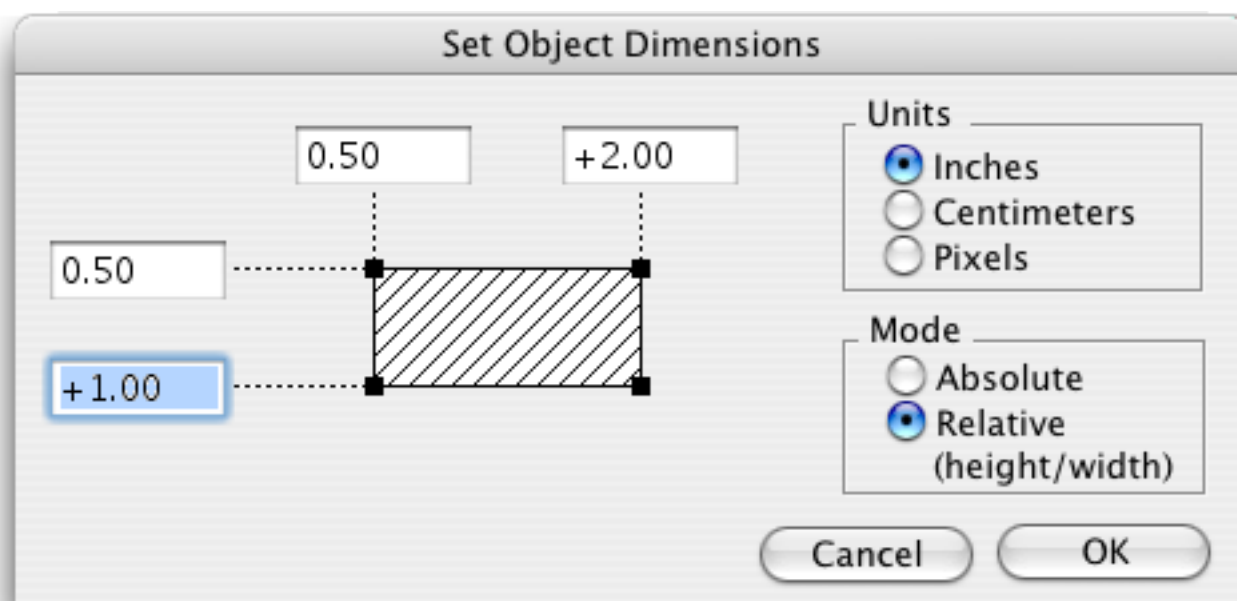
The guides can also appear when nudging an object's size.

### Viewing and Setting Exact Object Dimensions

If the window is wide enough, the Graphic Control Strip will show the exact location and size of the currently selected object. (This is only valid if a single object is selected — it does not reliably display the location or size of multiple objects.) To see the exact location and size of any object, simply click on the object.



The Dimensions dialog allows you to display and change the exact dimensions of any object. To use this dialog, simply select an object and click on the dimensions in the Graphic Control Strip (you can also choose **Dimensions** from the Edit menu).

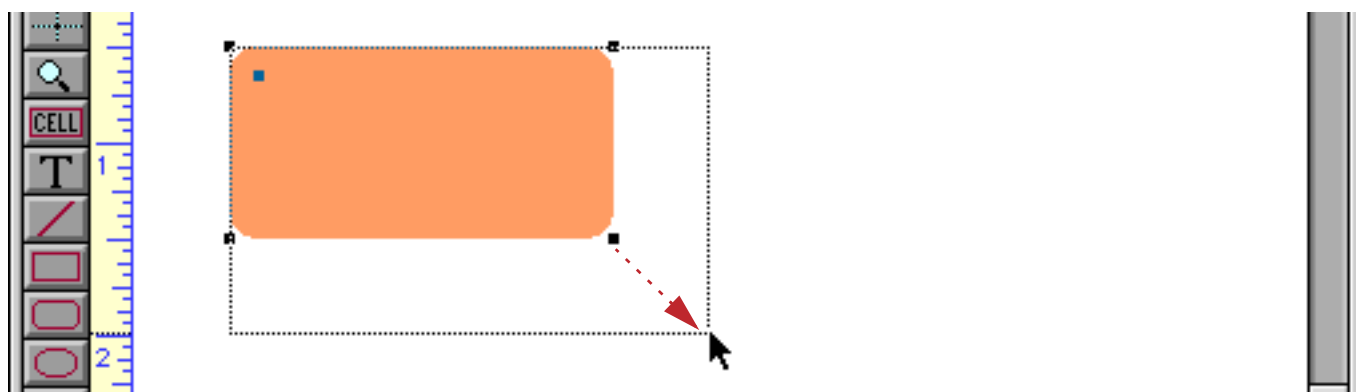


The **Dimensions** dialog gives you the choice of absolute or relative dimensions. **Absolute dimensions** display the position of all four corners of the object, that is each corner's position from the top left corner of the form. **Relative dimensions** display position of the top left corner of the object along with the size of the object; that is, the relative distance from the top left corner to the bottom right corner. Use relative dimensions when you want to move an object without changing its size, or change the size of an object without moving it. (Note: When using Relative dimensions, the object's height and width must have a + symbol in front of the number, as shown above.)

The **Dimensions** dialog can work with dimensions in inches, centimeters, or pixels. The dialog will default to the current ruler measurement units. (See "**Rulers**" on page 259 to learn how to set the ruler units.) Dimensions in inches or centimeters will be rounded to the nearest  $\frac{1}{576}$  inch (0.017 inch).

### Changing the Size of a Single Object

To change the size of an object, first select the object with the **Pointer** tool. Then use the mouse to drag one of the corner handles. As you drag the handle, an outline of the object will follow the mouse. Release the mouse when the corner is in the correct spot.



If you want to change the width or height of an object (but not both at once), hold down the **Shift** key while you change the size. Holding down the **Shift** key prevents the corner from moving diagonally.

## Nudging the Size of an Object

The arrow keys (←, →, ↓, ↑) usually nudge the entire object. However, after you click or drag a handle, the arrow keys will nudge just that handle. Each time you press an arrow key the handle will move one pixel in the direction of the arrow. In other words, each time you press an arrow key the object will grow (or shrink) one pixel in the direction of the arrow (or less than a pixel if you have changed the nudge distance using the **Form Preferences** dialog).

Let's look at the procedure step by step. Start by clicking on the object whose size you want to adjust.



Now click on the corner you want to adjust.



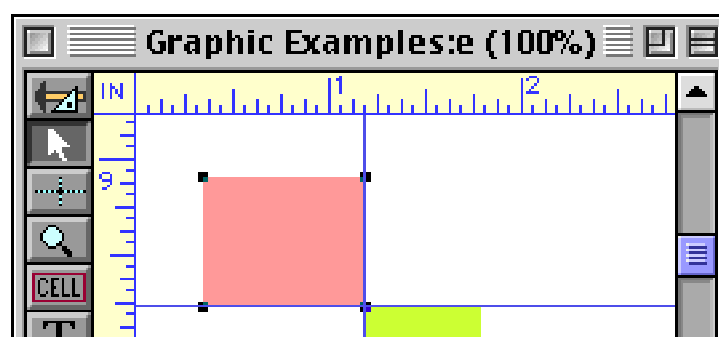
Use the arrow keys to adjust the size of the object in small increments. In this case we pressed the ↓ and → keys about half a dozen times each.



As soon as you click on another object, the arrow keys go back to nudging the entire object instead of just the corner.

### Nudge Size “Auto Guides”

As you nudge the size of an object, Panorama checks to see if any of the edges of the resized objects are aligned with the edges of any other objects. If any edge is aligned a temporary blue guide appears. In this illustration the lower right hand corner of the pink square has been nudged, and is now aligned in two directions with the yellow rectangle.



### Percentage Scaling

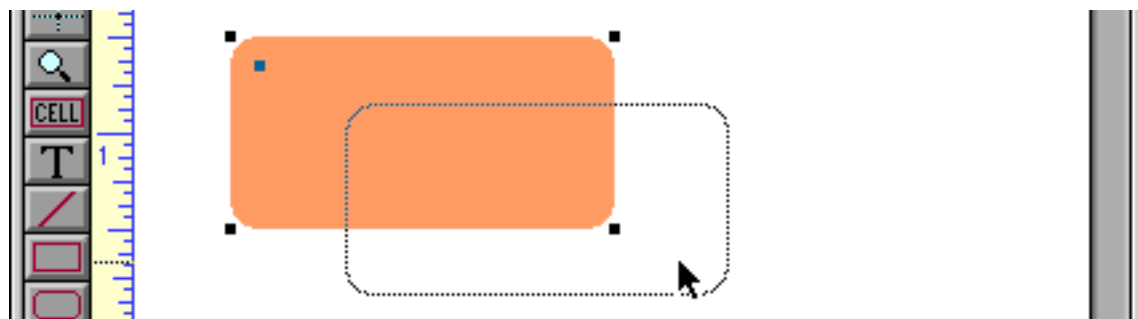
Use the **Scale** command (Arrange menu) to expand or shrink an object by an exact percentage. You can choose one of the pre-defined scales or type in any percentage between 1% and 999%.



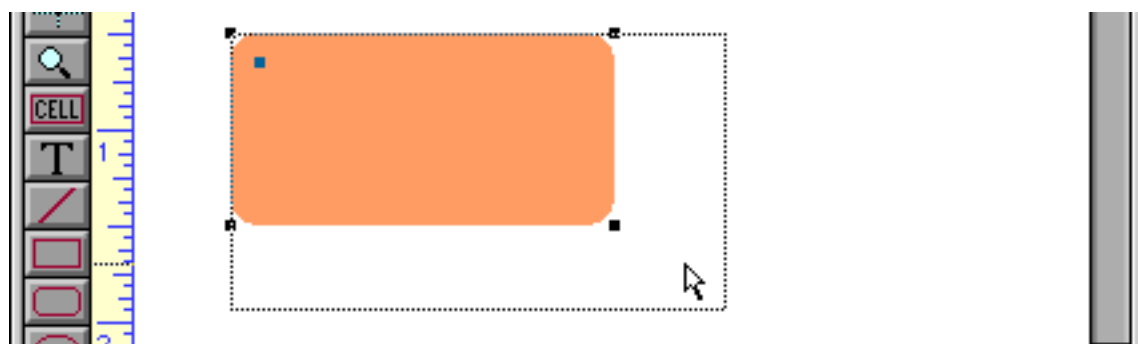
## Resizing Without Handles

Dragging on the inside of an object normally moves the object. But if you hold down the **S** key (the letter S) while you drag, dragging on the inside of an object resizes the object—just like dragging on a handle. This feature can be very handy when you are working on a cluttered form—the handle you want may be hard to find. To remind you that the S key is pressed, the cursor turns into a hollow arrow. When you release the S key Panorama will go back to normal operation.

This operation is easier to see than to explain with words. Here's what happens when you drag on the object normally — it simply moves:



But if you hold down the **S** key (the letter S) while you drag, the object will change size instead of moving. (Notice the hollow mouse arrow.)



When you release the mouse the object will expand or shrink.



Here's another way to view the operation of the **S** key. When you press this key, Panorama behaves as if the handles at the corners of the object had expanded to fill the entire object. No matter where you click, you are clicking on a handle. Therefore, you cannot drag the object, but only resize it. The four colored rectangles in the illustration below symbolize the expanded handles (actually, they would be slightly bigger than this, so that they would cover the entire object).



If you want to change only the width or height of the object, but not both, hold down the **Shift** key at the same time as you hold down the **S** key. This will prevent the object from changing size diagonally.

## Removing Objects

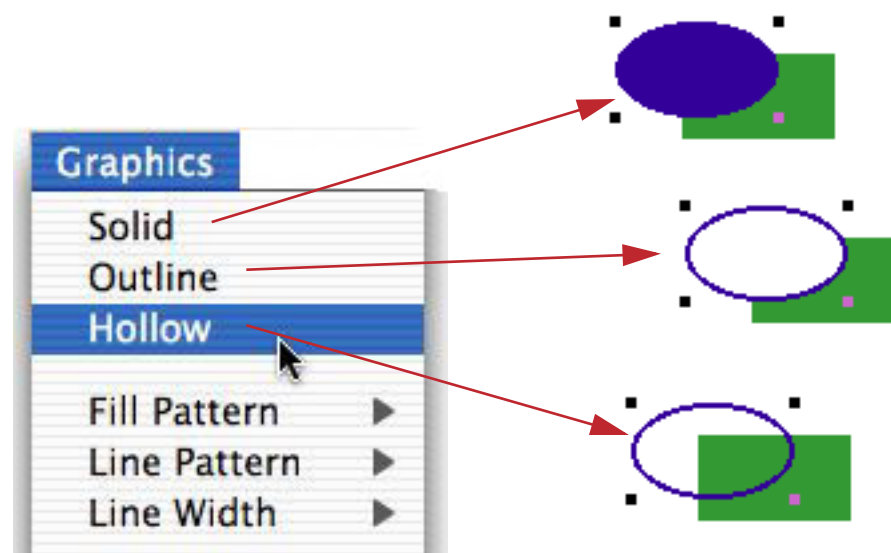
To completely remove one or more objects from the surface of the form first select the objects (See “[Selecting a Single Object](#)” on page 256 and “[Selecting Multiple Objects at Once](#)” on page 257). Then choose **Cut** or **Clear** from the Edit menu. **Cut** places the object on the clipboard so that it can be pasted back into the form in a new location. You can also remove the object by pressing the **Delete** or **Backspace** key (this is the same as choosing **Clear**).

## Modifying Object Attributes

New objects are usually white with a thin black border. To customize an object you can change the fill pattern, border (pen) pattern, line thickness, and color. You can customize these with the Graphic Control Strip or the **Graphics** menu.

### Solid, Outline and Hollow Objects

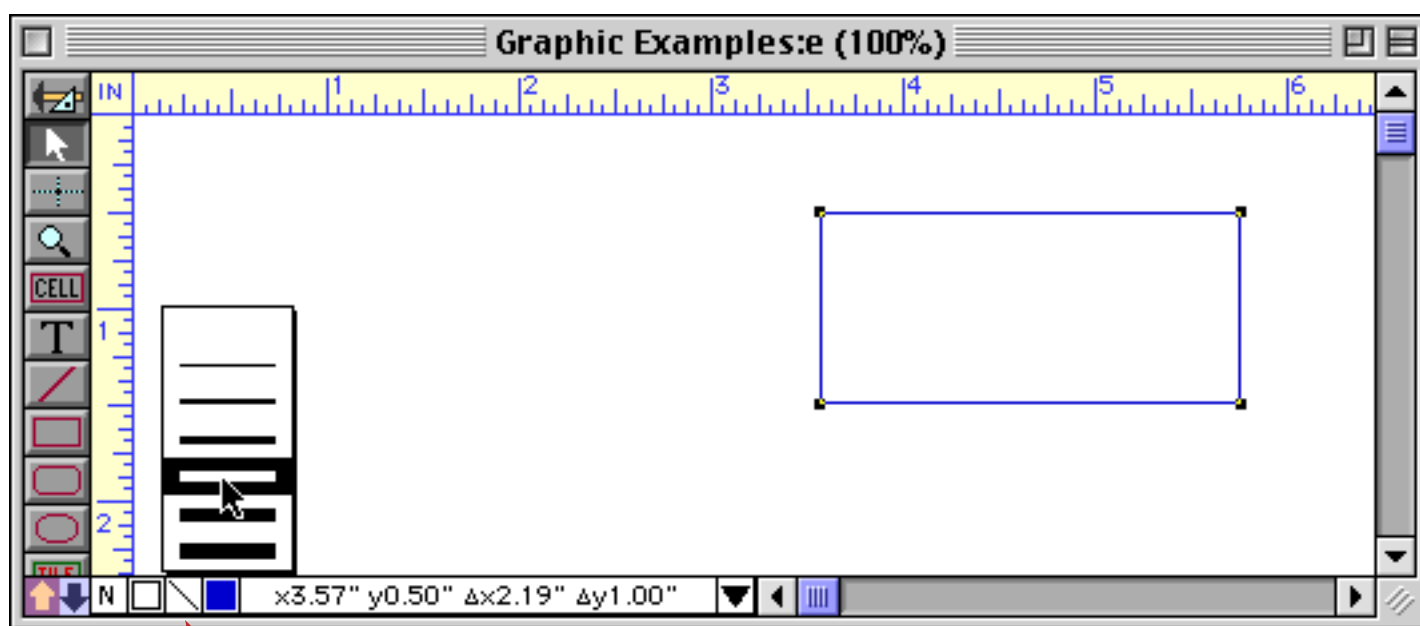
The first three choices in the Graphics menu, **Solid**, **Outline** and **Hollow**, set the fill and line patterns of the selected objects to their most common choices, as shown in the diagram below.



Solid objects are colored all the way through. Outline objects are filled with white with a colored border. Hollow objects have a colored border but a transparent center.

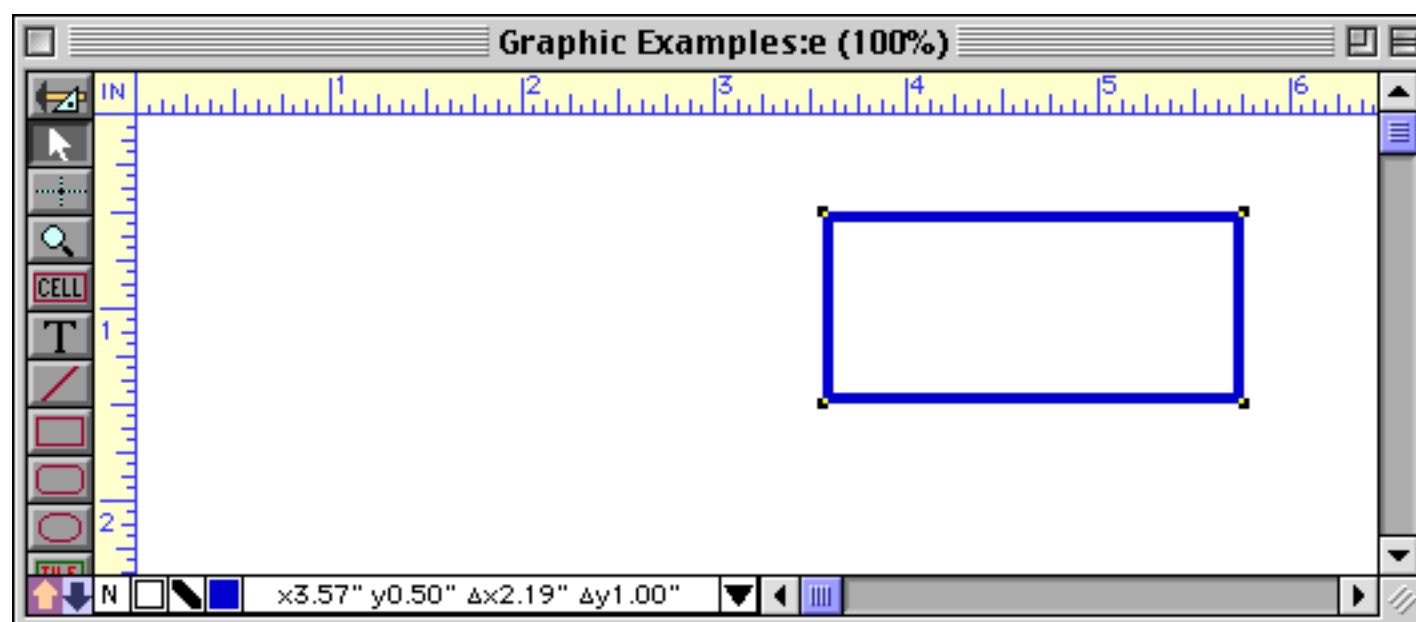
### Line Width

The **Line Width** menu contains seven different line/border thicknesses from 1/4 to 6 points.



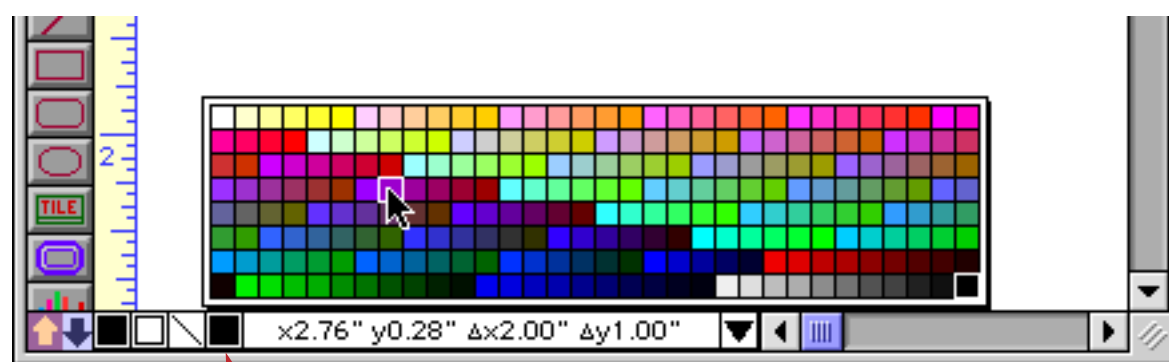
*click here for line width menu (or use the Graphics Menu)*

When you release the mouse the line width of the object will be updated. Notice that the object's line width also appears in the Graphic Control Strip.



## Color

The **Color** menu contains 256 colors. To color an object, first select the object with the **Pointer** tool, then pick the color from the menu.



*click here to set the color (or use the Graphics Menu)*

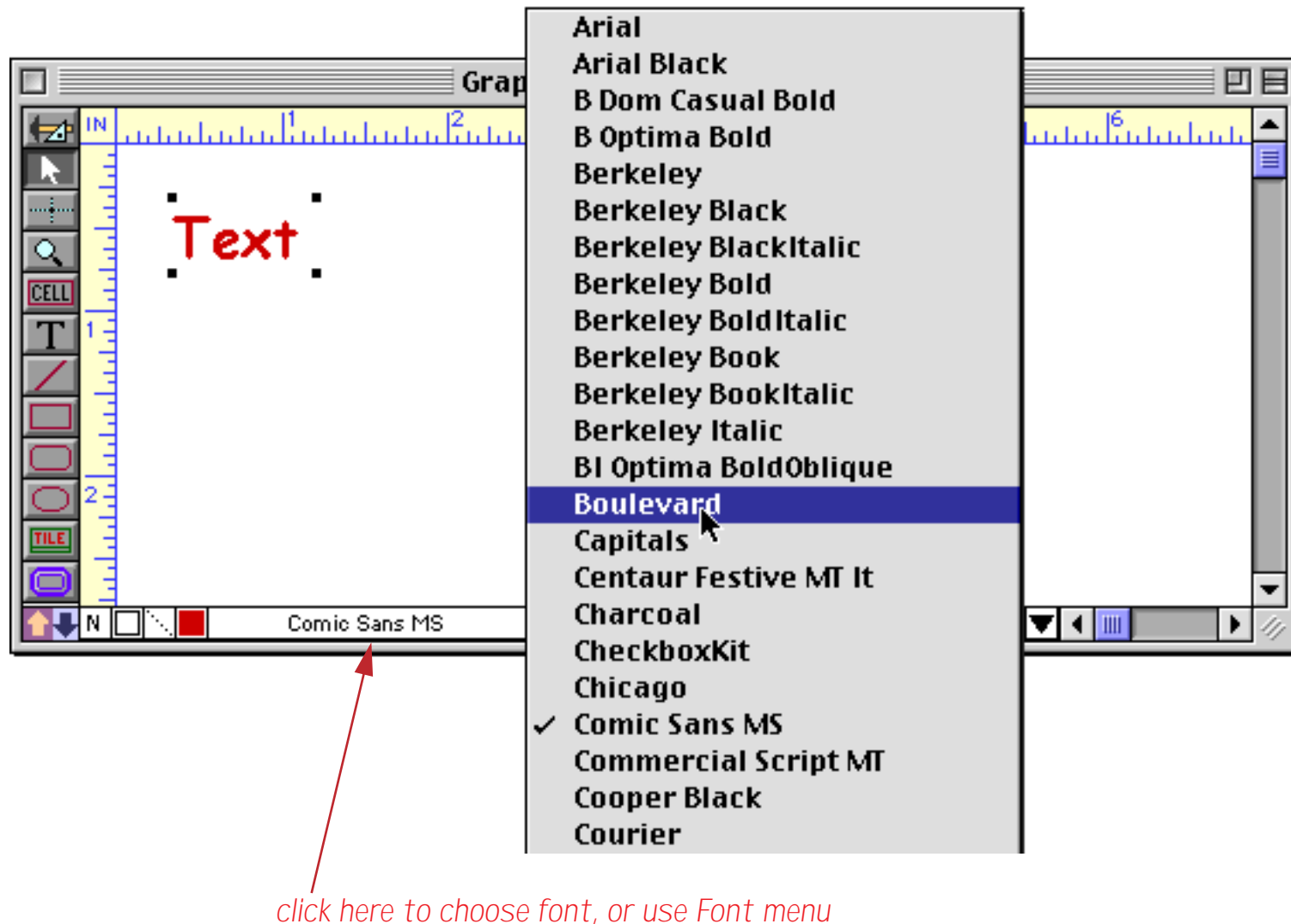
When you release the menu the object(s) will change color. To choose a color that is not one of the 256 colors in the palette, use the **Choose Colors** command in the Graphics menu. The system's standard Color selection dialog will appear, allowing you to choose any of millions of colors. You can also open this dialog by holding down the Control key (Mac) while clicking the color swatch in the Control strip. On PC systems you can right click the swatch.

## Copying and Pasting Colors

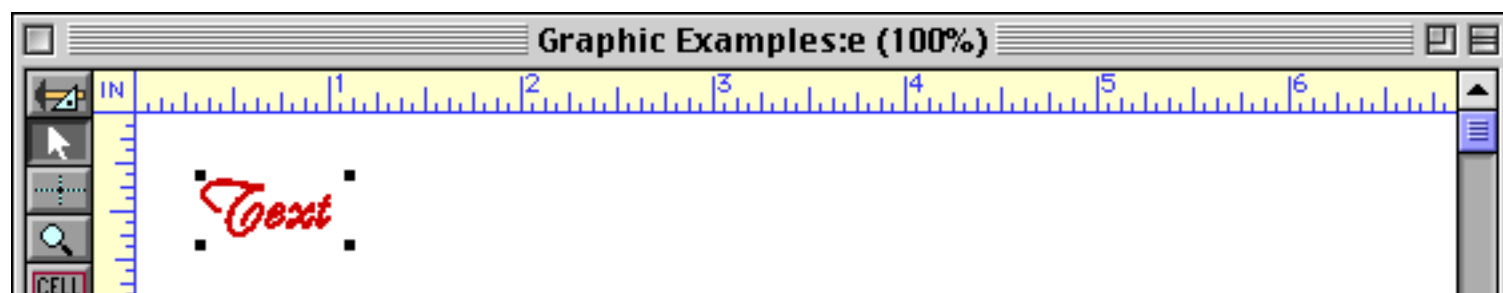
Sometimes you'll want one object to exactly match the color of another object. You can use the **Copy Color** and **Paste Color** commands to transfer a color from one object to one or more other objects. Start by selecting the object that has the color you want, then choose the **Copy Color** command. This copies the color into a special clipboard. Now select the other objects that you want to set to this color and choose the **Paste Color** command. The selected objects will change to the same color as the original object. (Note: The color clipboard is completely separate from the normal clipboard that is used for other copy and paste operations.)

## Font

The **Font** menu lists all the fonts installed on your system. To change the font of an object that contains text, first select the object with the **Pointer** tool, then pick the font from the menu.



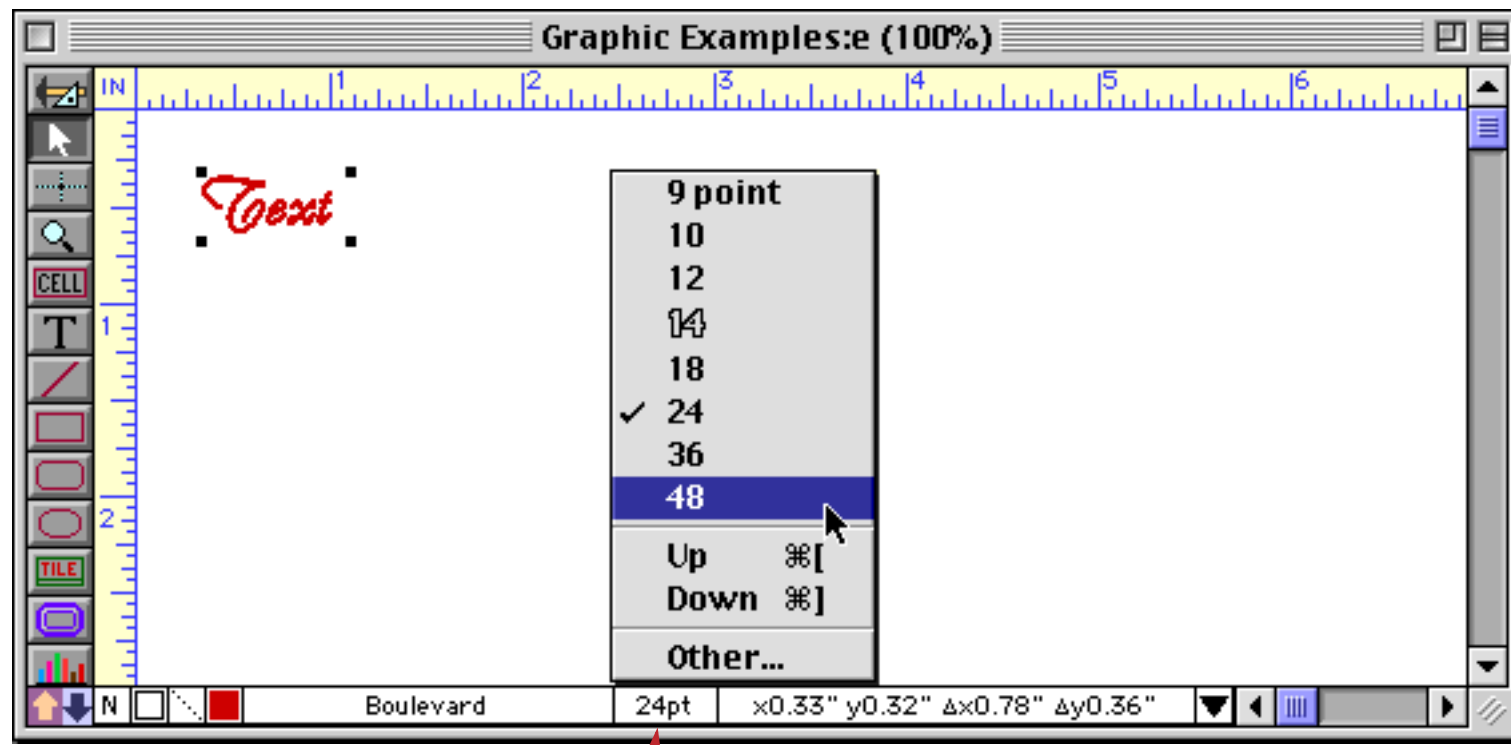
When you release the mouse the object will change to the new font.



You can use the **Font Usage Wizard** to display a list of all the fonts used in every form in your database.

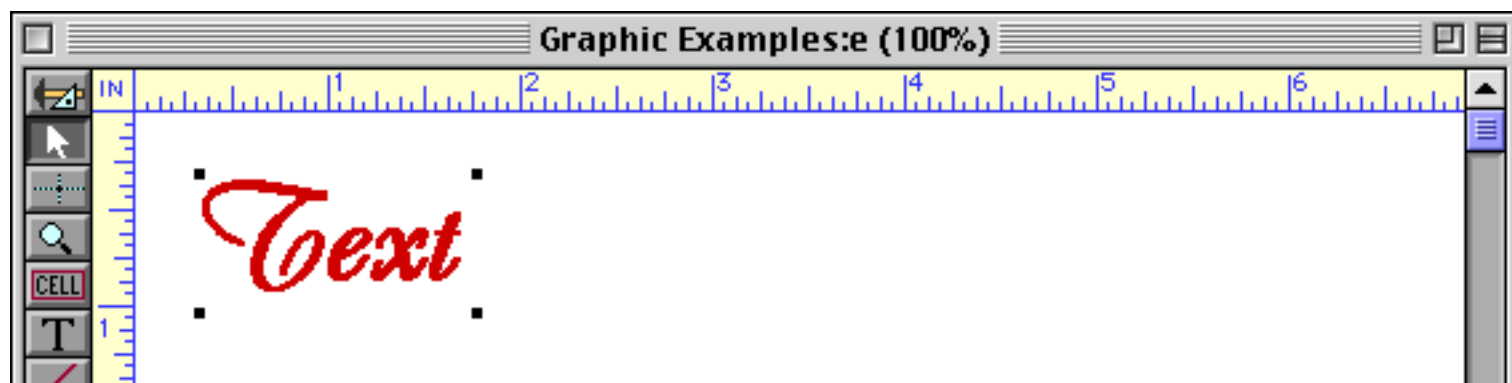
## Text Size

The **Size** menu allows you to adjust the text size of an object. To change the text size, first select the object with the **Pointer** tool, then pick the size from the menu.



*click here to choose size, or use Size menu*

When you release the mouse, the size of the text will change.

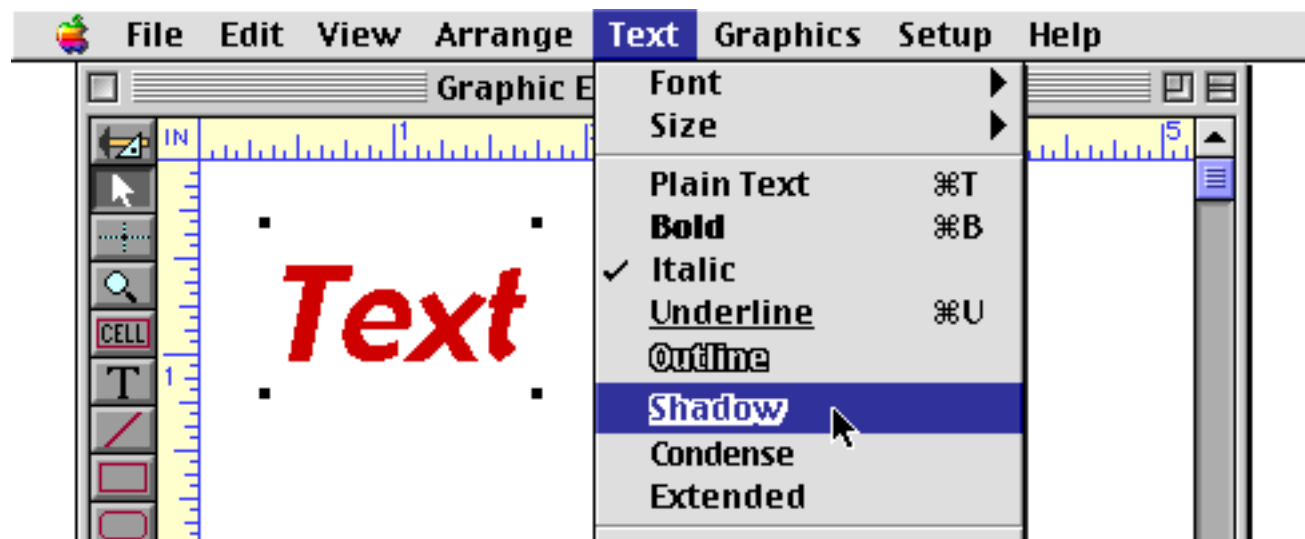


If the size you want is not listed in the menu, choose **Other**. A dialog will appear asking for the new font size. You can type in any integer font size you like — big or small. You can also make small adjustments to the text size by selecting **Up** or **Down**. These commands increase or decrease the text size by one point.

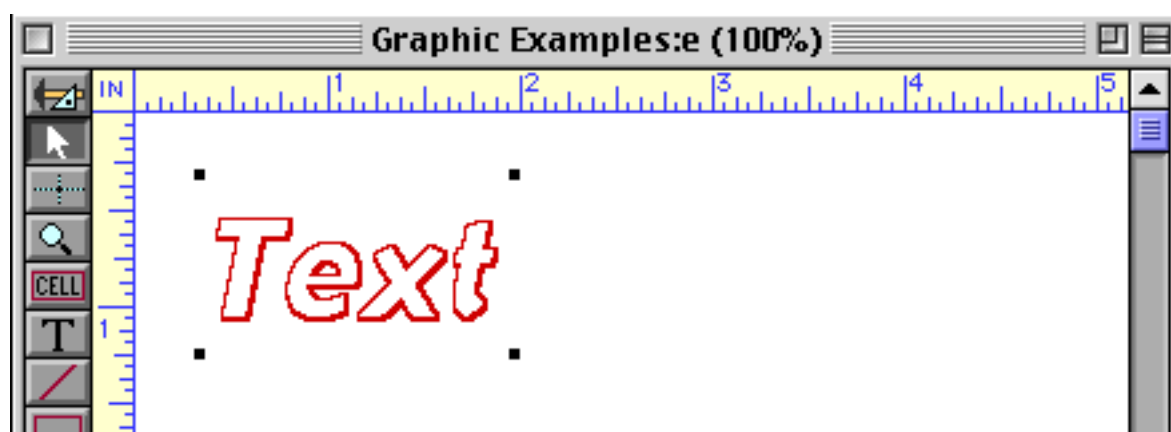


## Text Style

The **Text** menu allows you to change the style of an object containing text.



You must change the style of the entire object — all the characters in the object must have the same style.



The only exception to this rule is the Word Processor SuperObject, which allows different styles, fonts, sizes and colors to be mixed. See “[Word Processor SuperObject](#)” on page 651 of the **Panorama Handbook** for more information.

## The Object Properties Dialog

Many graphic objects have dialogs that control various options for the object. To open the dialog for a particular object, first select the object and then choose the **Object Properties** command in the Edit menu. This command will work for data cells, tiles, buttons, flash art, flash sound and all SuperObjects. There is no dialog for basic shapes like rectangles, ovals, etc. The Object Properties dialog is the same dialog that appears when you create the object.

Another way to open the **Object Properties** dialog is to simply double click on the object in question. In fact, this is the most common way to open this dialog. With most types of objects you can double click anywhere in the object. If an object doesn't have an **Object Properties** dialog (for example basic shapes like rectangles and ovals) then double clicking on the object will select all of the object inside the object.

## Working With Multiple Objects

Most forms contain dozens or even hundreds of objects. Often you'll need to move, resize, or align several objects at once. Panorama has a number of tools to make these kinds of tasks easier.

### Grouping Objects Together

The **Group** command (in the Arrange menu) gathers several objects together and combines them into a single object. Once the objects are grouped together, they act as a single unit. The group can be selected, moved, copied, or resized just as if it was a single object. If you later want to manipulate one of the individual objects within a group you must reverse the process with the **Ungroup** command.

### Moving Multiple Objects

You can move several objects at once by selecting the objects and then dragging one of them. The other selected objects will follow as you drag. You can also nudge the selected objects with the arrow keys.

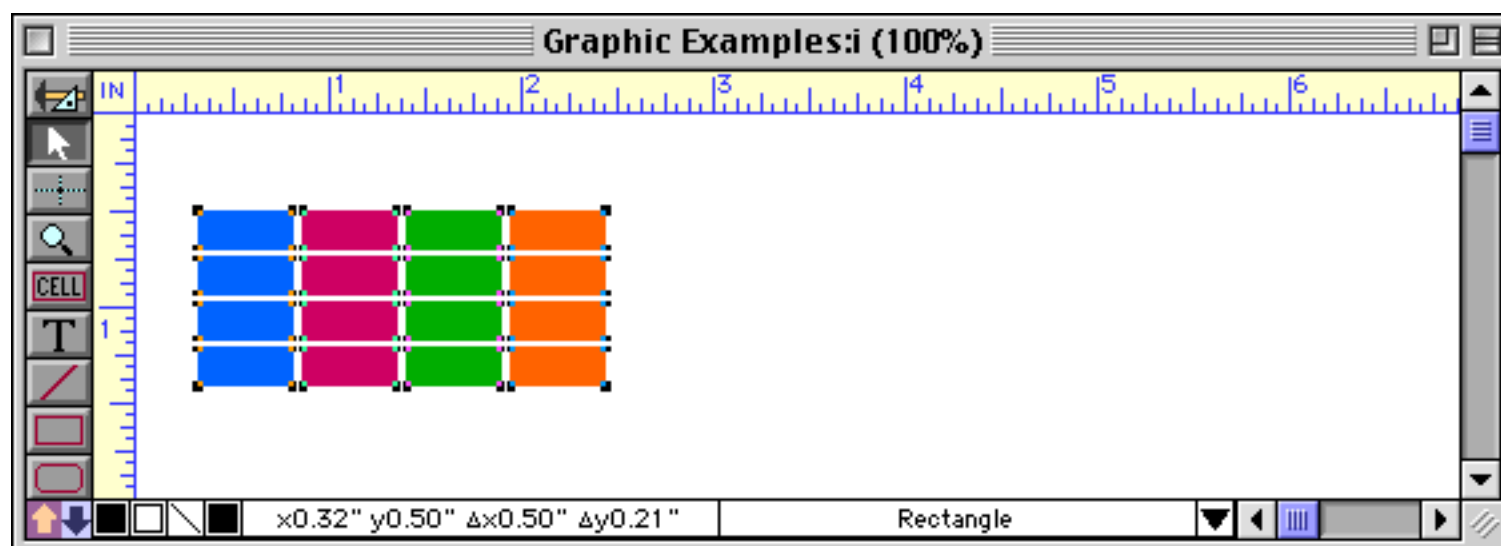
## Resizing Multiple Objects

You have several choices available for resizing multiple objects. You can combine objects into a group and then change the size of the group. You can use the **Dimensions** dialog to make each selected object a fixed height or width, or to increase or decrease the size of each selected object. Finally, you can use **cluster resize** to change the width of a column within a table, or the height of a row.

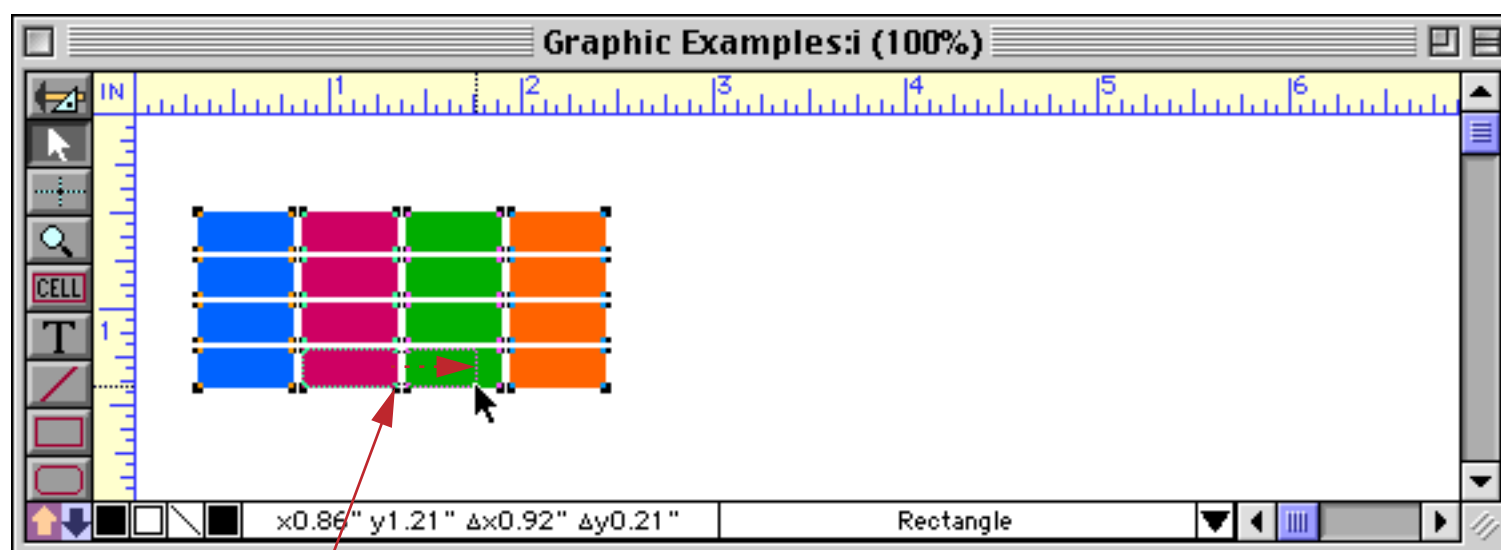
When you change the size of a group object, all of the dimensions inside the group change as well. For instance, if you double the width of a group, all of the individual objects within the group will also double in width. The horizontal spacing between the groups will also double.

## Cluster Resize

Panorama's **Cluster Resize** is automatically active whenever you select several objects and then change the size of one of them. Cluster resize adjusts the sizes and locations of the other selected objects to match the change. For instance, suppose you have a matrix of boxes, and you want to increase the width of one column.

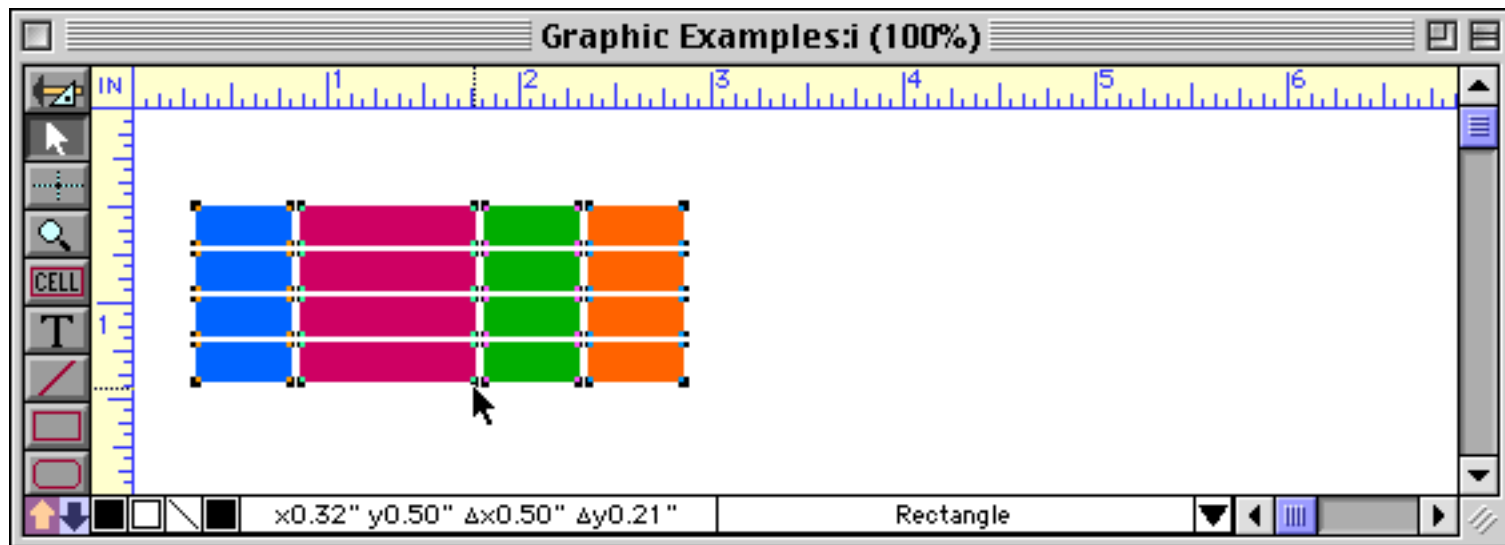


Start by selecting all of the objects. Then grab one of the handles in the column you want to make wider and drag it to expand the box.



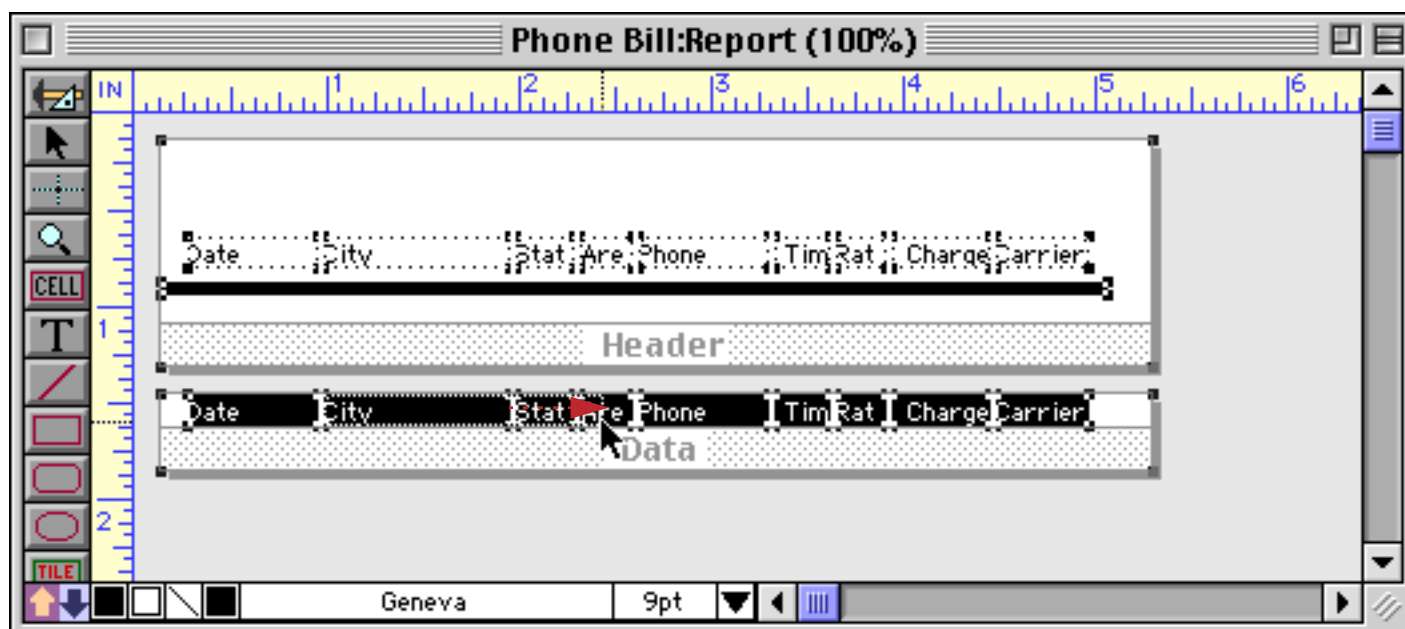
*drag to change the size of one object*

When you release the mouse, cluster resize will kick in. In this case, it will automatically increase the width of all the other boxes in the same column. It will also shift the boxes on the right to make room for the expansion.

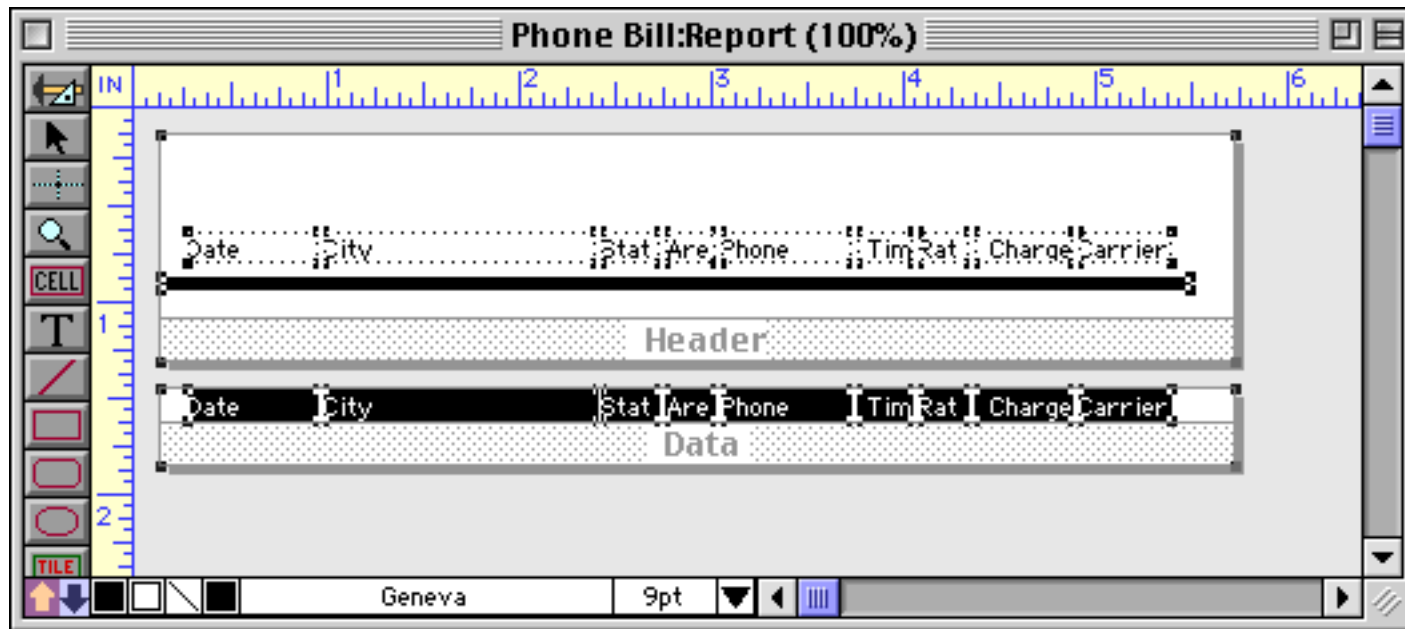


At first glance cluster resize may look a little bit like magic, but actually it is quite simple. After you change the size of any object, Panorama checks to see which edge (or edges, for diagonal moves) of the object you moved—top, bottom, left, or right. It then adjusts the corners of any other selected objects that are in that direction. For example, if you move the right edge of an object, any objects that are even with or to the right of that edge will be adjusted.

Cluster resize is also great for working with reports. If you keep your report tiles lined up, then changing the width of an item in the body of the report automatically adjusts the width of the same item in the report header. (Remember that all the objects must be selected. Only selected objects will be adjusted.) In this example all the objects are selected and we expand the width of the City field.



Cluster resize automatically adjusts all of the other objects, including the report tiles.

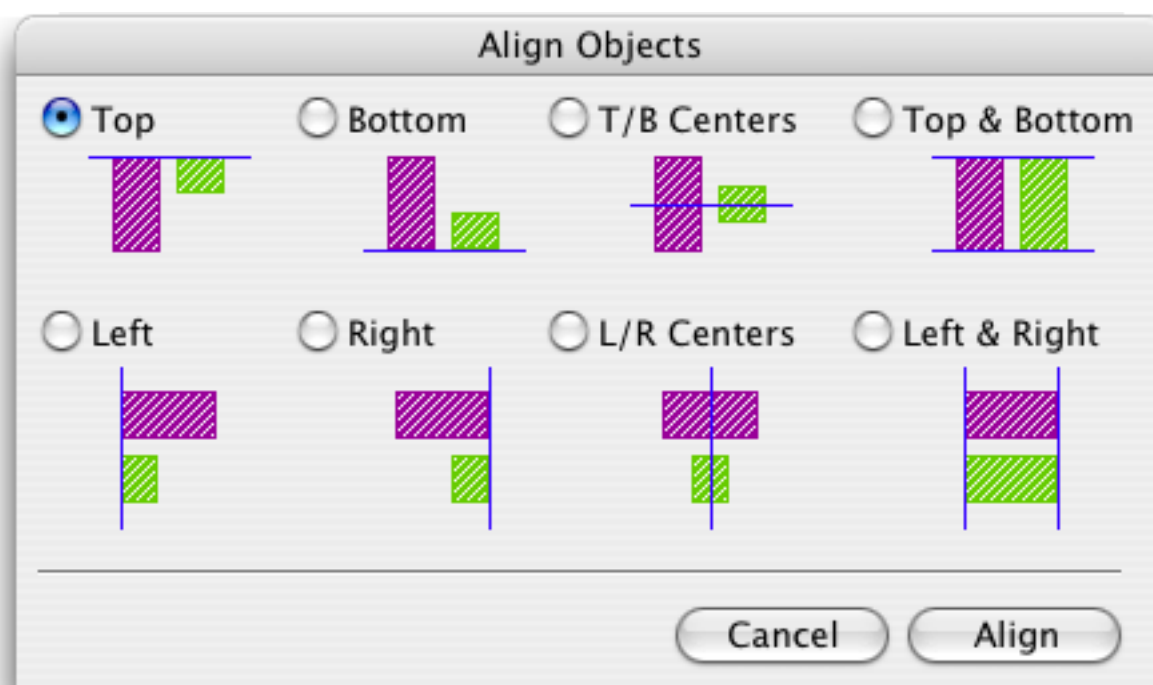


By the way, cluster resize also works when you nudge the size of an object using the arrow keys. Select the objects, then click on the handle you want to nudge. As you use the arrow keys to resize the object, cluster resize will automatically adjust the other objects.

Don't forget that when you are resizing an object you can use the **Shift** key to make sure that only the height or width changes, but not both. For example, if you are adjusting the width of a column using cluster resize, hold down the **Shift** key so that the height of the row doesn't change also. You can also hold down the **S** key to make sure that you are resizing, and not dragging.

### Aligning Objects

If you need to line up several objects, you can either do it by eye or you can let the **Align** command (in the Arrange menu) do it for you. The Align dialog gives you eight options for aligning objects in different directions.



Most of the alignment options simply shift the objects to align them, but the **Left & Right** and **Top & Bottom** options actually change the sizes of the selected items. Instead of shifting the objects, these options actually expand the selected objects to make them all the same width or all the same height. If you ask for **Left & Right**, all the selected objects will be expanded to the width of the widest object. If you ask for **Top & Bottom**, all the selected objects will be expanded to the height of the tallest object.

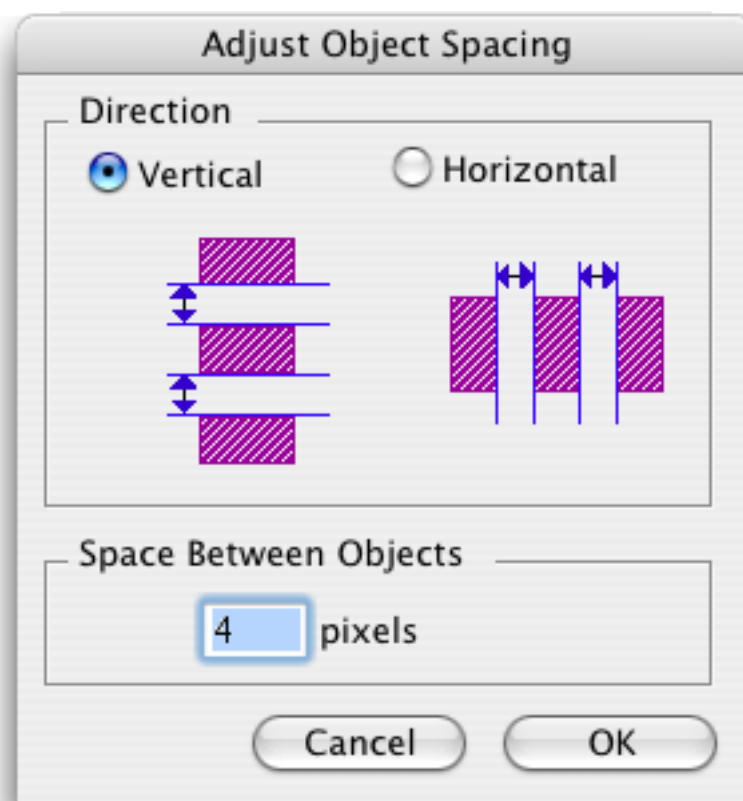


## Adjusting Spacing Between Multiple Objects

The **Spacing** command (in the Arrange menu) allows you to adjust the vertical or horizontal spacing between multiple objects. The dialog will shift the selected objects so that they are evenly spaced. For example, you could start with a somewhat random collection of objects like this.



Open the **Spacing** dialog and set the spacing to 4 pixels between the objects.



When the **Ok** button is pressed the objects will slide vertically into place.



To finish our cleanup we'll use the **Align** command to make all of the objects the same width (by selecting **Align Left & Right**).



## Duplicating Objects

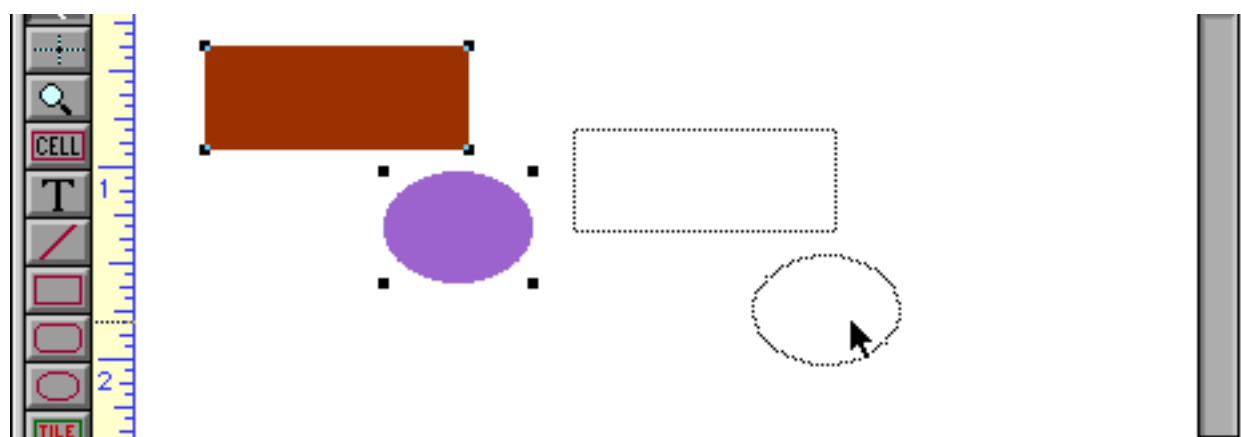
Why do the same work twice? Panorama has several methods for creating a duplicate of one or more objects.

### Duplicate

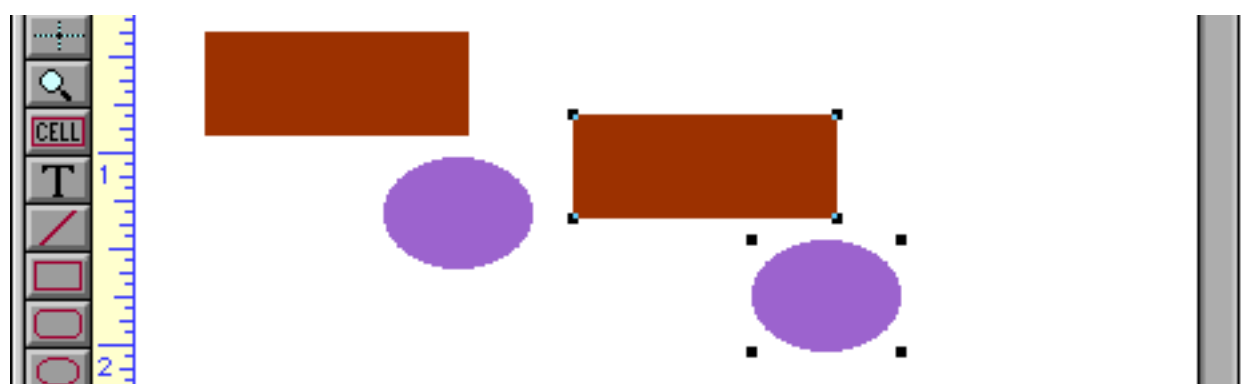
The easiest way to copy objects is with the **Duplicate** command. Just select the objects and choose **Duplicate** from the Edit menu. The new object(s) are placed just below and to the right of the originals. If you duplicate an object (or objects) and then immediately drag (and/or nudge) the copy to a new position, Panorama will memorize this position relative to the original object. Now if you duplicate the copy, Panorama will automatically place the copy of the copy in the same relative position.

### Drag Duplicating

You can also duplicate an object by dragging the object with the a special key held down. On the Macintosh the special key is the **Option** key. On PC systems the special key is the **Alt** key. When the special key is held down you drag a copy of the object(s), instead of the original. Just hold down the key and drag the same way you would to move the object(s).



When the mouse is released a second copy of the object(s) appears at the new location.



If you want the copy to line up with the original hold down **Shift** key and the **Option/Alt** key as you drag the object. The **Option/Alt** key tells Panorama to make copies, while the **Shift** key prevents you from dragging the copy diagonally.

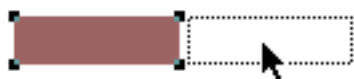
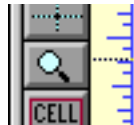
Once you have created a copy by **Option/Alt** dragging, you can make another copy with the **Duplicate** command. The **Duplicate** command will exactly mimic your **Option/Alt** drag, allowing you to quickly create an accurately spaced row or column of objects (see the next section for an example). **Warning:** If you want the **Duplicate** command to mimic your **Option/Alt** drag, you must not do anything in between dragging and pulling down the menu. If you click anywhere else in the form, the **Duplicate** command will not mimic the **Option/Alt** drag.

### Step and Repeat

You can combine **Option/Alt** dragging (see previous section) with the **Duplicate** command to quickly step-and-repeat evenly spaced rows, columns, and complete tables of objects. To create a row of objects, start with just one object.



Hold down the **Shift** and **Option/Alt** keys and drag the object to the right.



Release the mouse to create a copy.



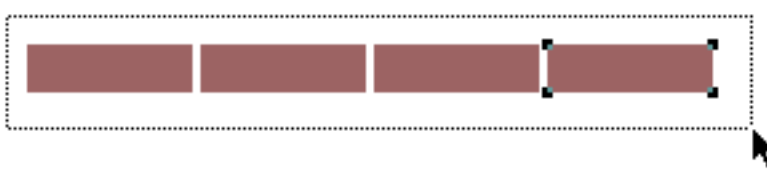
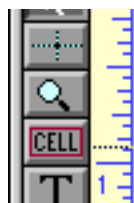
Then use the **Duplicate** command to create another copy of the object. The copy will automatically be placed at the same spacing as the first duplicate.



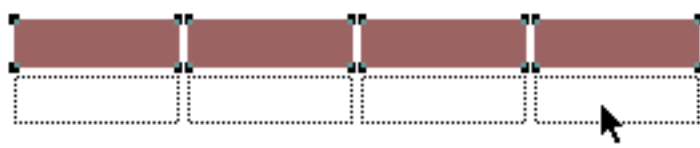
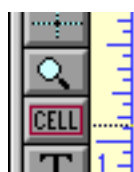
Repeat the **Duplicate** command until the row is complete.



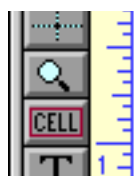
To create a complete table, select all of the objects in the row by dragging a marquee around them.



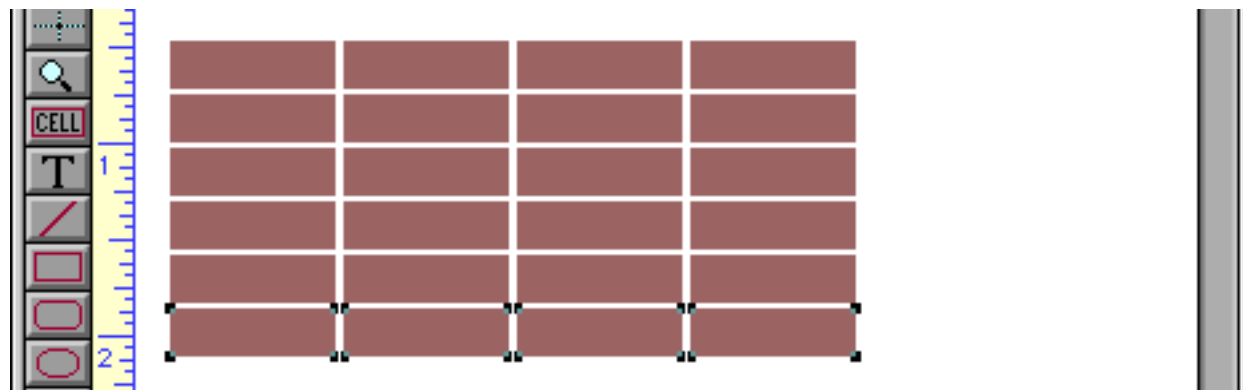
Hold down the **Shift** and **Option/Alt** keys and drag the row down, creating a copy of the row.



Release the mouse to create the new row.



If you don't get the spacing just right, you can nudge the row with the arrow keys (see "[Nudging an Object \(or Objects\)](#)" on page 261). Once the second row is in position, use the **Duplicate** command to step and repeat an additional copy of the row. Repeat the **Duplicate** command until the table is complete.



Once the table is complete, you can modify the width of columns within the table using cluster resize. Or you can change the spacing between the rows and columns using the **Spacing** command.

### Cut, Copy, and Paste

You can use the clipboard to copy objects and to transfer objects from one form to another form. The **Cut** and **Copy** commands copy the selected objects into the clipboard. **Cut** also removes the objects from the form.

The **Paste** command places a copy of the objects in the clipboard on to the form. The object (or objects) is placed into the middle of the current window.

You can also use the **Paste** command to paste in graphics created in another program.

### Copying an Entire Form

The **Copy Form** and **Paste Form** commands (in the Graphics Mode's Edit menu) allow an entire form to be copied at once -- including the form's Page Setup, custom menu setup, form comments -- everything!

To duplicate a form, start with the original form in Graphics Mode. Then choose the **Copy Form** command from the Edit menu. Then choose the **Paste Form** command. (If you want to duplicate the form in another database you must go that database, open a form and switch that form to Graphics Mode before using the **Paste Form** command.) The **Paste Form** command will ask you for the name of the new form. When you press the **Paste Form** button, Panorama will create the new form as an exact duplicate of the old form. (Notice that you do not create the new form in advance — the **Paste Form** command creates the new form for you.)

You can also use the **Copy Form** command with the regular **Paste** command to copy all the objects in a form into another form. When used this way, the **Copy Form** command is the same as choosing **Select All Objects** followed by **Copy** command. When used this way, only the objects are copied and not the Page Setup, custom menus, etc.

## Overlapping Objects

The computer screen is two dimensional, in other words, flat. So what happens if two objects overlap each other? To resolve this question, Panorama treats the objects as if they were placed on a stack of clear sheets. For example, consider the three overlapping objects in this form —



If you could look at this form on edge, it would look like this —



An object that is on top of another object is said to be **in front**. An object that is below another object is said to be **in back**. When you create a new object, the new object is placed in front of all other objects. Objects that were created earlier will be partially or completely hidden if the new object overlaps them.

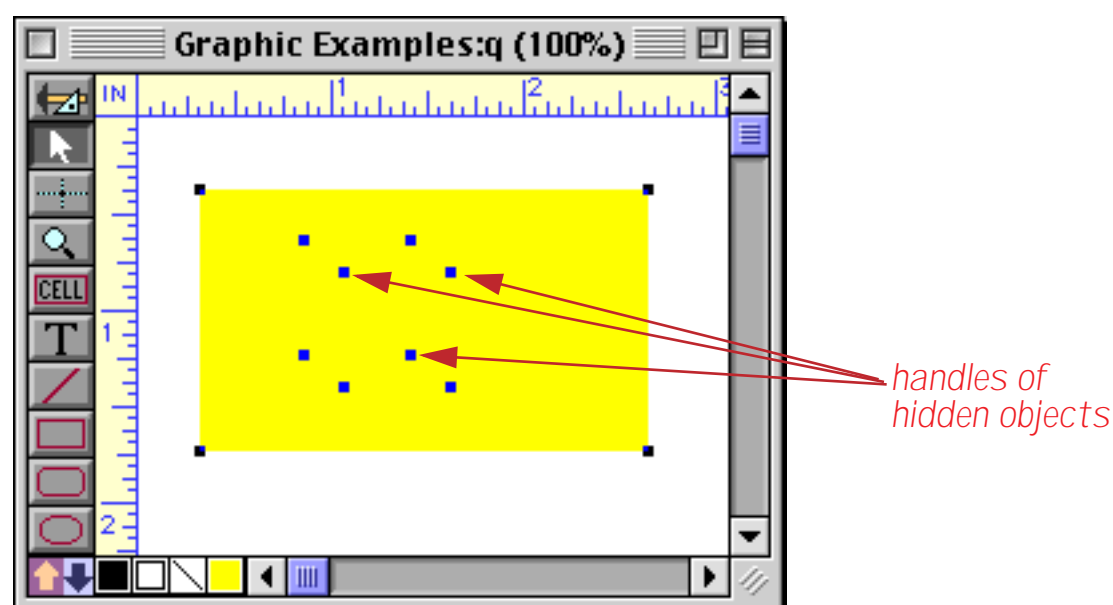
### Changing the Stacking Order

The **Bring to Front** and **Send to Back** commands (in the Arrange menu) change the stacking order of overlapping objects. To put an object behind everything else, select the object and then choose **Send to Back**.

### Selecting a Completely Hidden Object

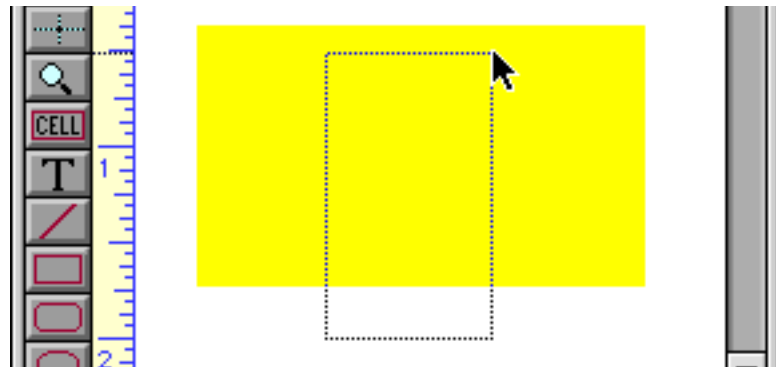
If an object is completely hidden you can't click on it. However, there are several ways to select hidden objects.

To find a hidden object, use the **Select All Objects** command (in the Edit menu). This command makes handles appear for every object, including hidden objects. For example, at first glance the form shown below would appear to have only one object — a yellow box. The **Select All Objects** command reveals that there are two hidden objects behind the yellow box.

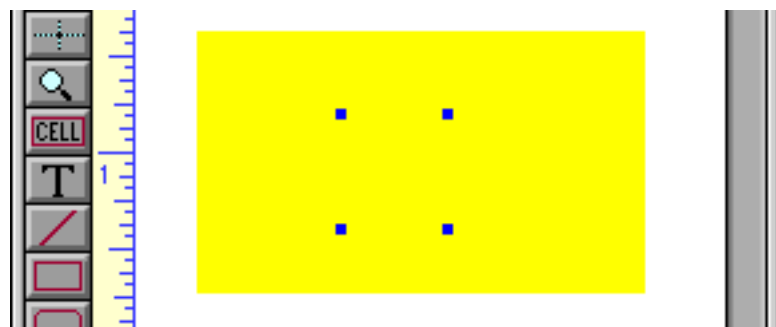




Of course, you can also find a hidden object by moving the objects in front of it out of the way, or by sending the object in front to the back. Sometimes you may be able to select a hidden object by dragging a marquee around the object.



Once the object is selected you can bring it to the front, change the object properties, or nudge the object with the arrow keys.

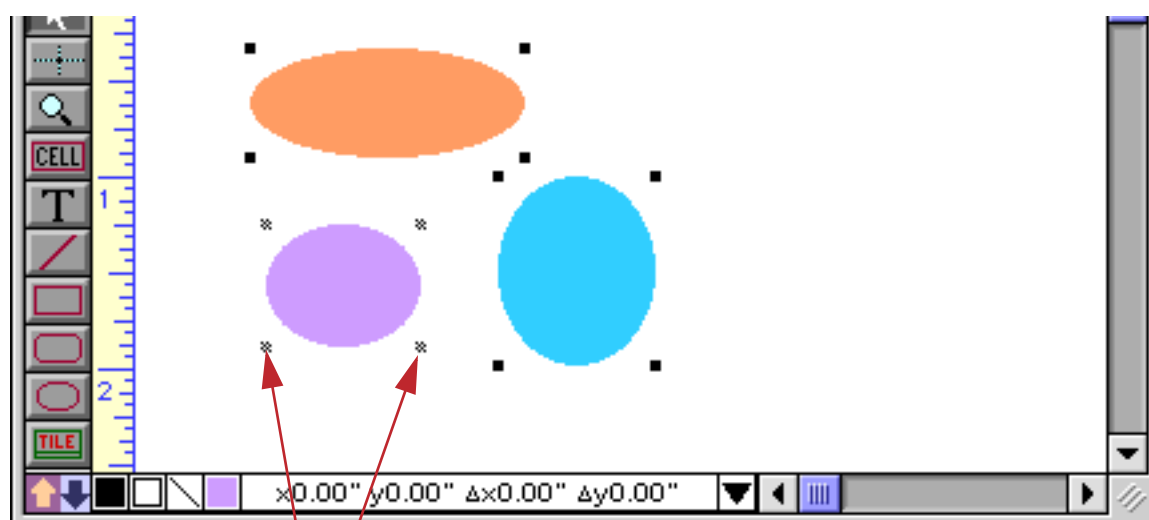


Another technique for selecting a hidden object is to hold down a special key while you click. On the Macintosh this special key is the **Command** key, on PC systems it is the **Control** key. The first time you click, the topmost object will be selected. The next click will select the next object behind the top object (remember, you must hold down the **Command/Control** key). Each time you **Command/Control** click again the next object behind the current object will be selected. When you reach the bottom object Panorama will cycle back to the top and select the topmost object again. You can keep clicking around and around forever.

## Locked Objects

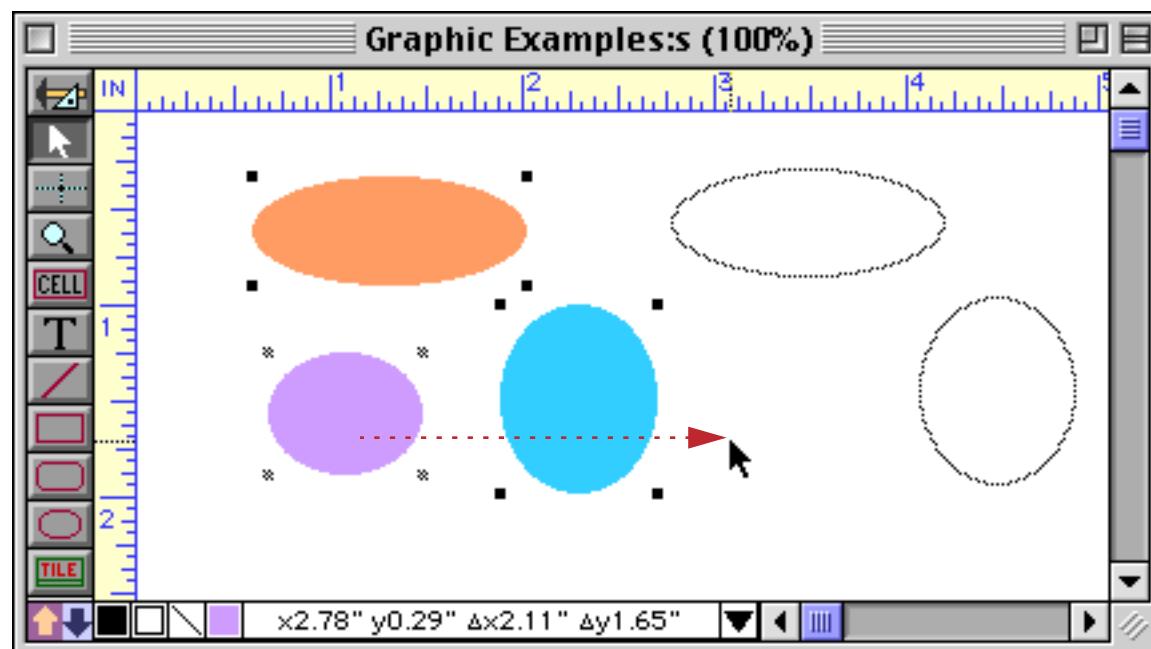
Use the **Lock** command to lock objects. Locked objects can't be moved or resized. Locking is convenient when you want to make sure the work you have already finished isn't disturbed as you continue working.

To lock one or more selected objects, choose **Lock** from the Arrange menu. The object handles will turn gray. This shows that the objects are locked. The illustration below shows three selected objects, one of which is locked.



*gray handles on locked object*

If you attempt to drag these three objects, only the unlocked objects will actually move.



A locked object cannot be moved, resized, or have any of its attributes (color, fill pattern, etc.) changed. However, you can duplicate the object (the copy is not locked).

Use the **Unlock** command to release a locked object(s). Once they are unlocked the objects can be moved and resized normally.

### Ignoring Locked Objects

Although locked objects cannot be moved, they can still be selected by clicking on them. (Otherwise they could never be unlocked again!) However, if you check the **Ignore Locked Objects** option in the Arrange menu, you will not be able to select locked objects.

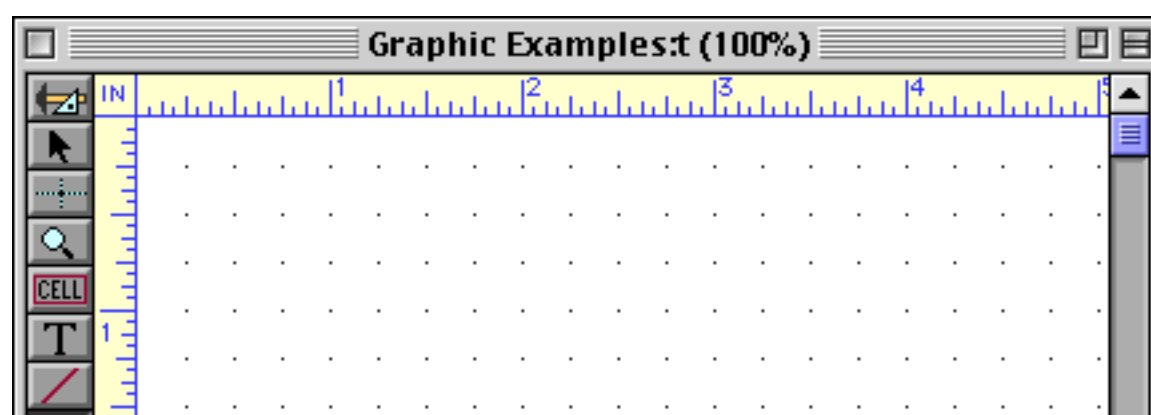
**Tip:** The **Ignore Locked Objects** option is useful for working on top of a large background covering the entire form. For example, you might paste a scanned image into the form, and then create data cells and other objects on top of the scanned background. By locking the scanned image and turning on the **Ignore Locked Objects** option, you won't have to worry about accidentally selecting and possibly moving the scanned background image.

### Alignment Grid

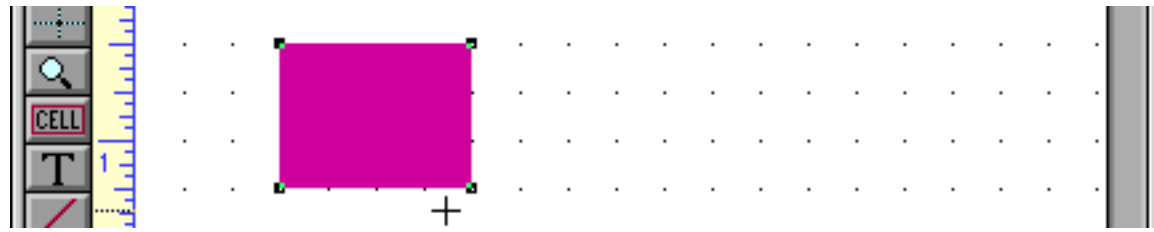
Panorama normally allows you to position objects freely anywhere on a form. The **Grid** dialog (in the Arrange menu) allows you to set up a grid that can help you arrange objects in neat rows and columns.

The grid spacing can be specified in inches, centimeters, or pixels. If you use inches or centimeters, the spacing will be rounded to the nearest 1/576 inch (For example a grid spacing of 0.1 inch will be rounded to 0.098 inches, or 7.25 pixels.)

If you want to make the grid visible, check the **Show Grid** option. When this option is checked a dot appears at each grid point.



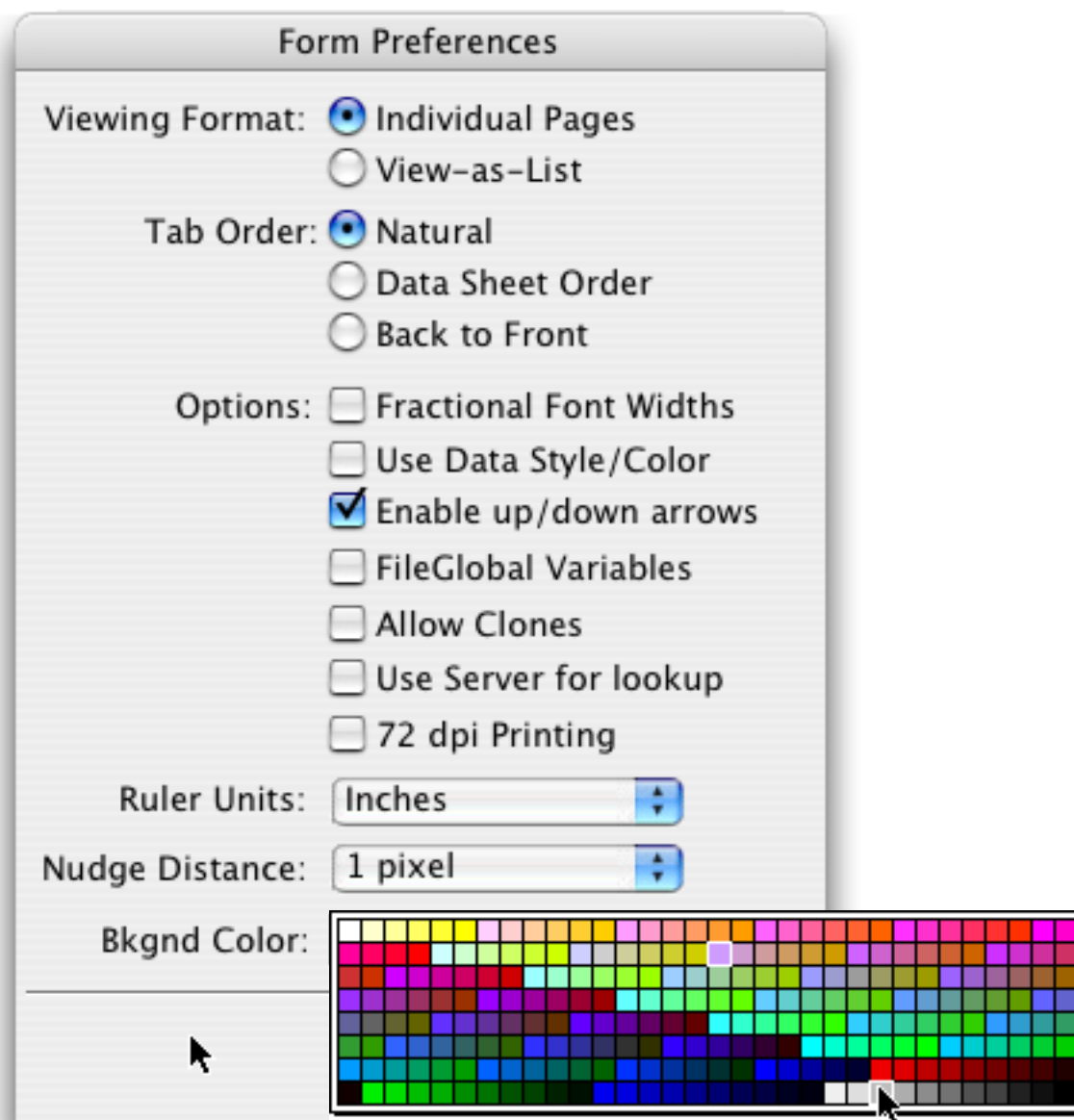
When the **Snap to Grid** option is checked, Panorama will automatically align objects to the grid. Objects are aligned to the grid whenever they are created, moved, or resized. This option does not affect objects already in the form (unless you modify them). Panorama snaps the object into place as it is created.



**Tip:** When **Snap to Grid** is on, every object you create is automatically aligned to the grid. But what if you need one or two objects that are not aligned to the grid? Instead of turning **Snap to Grid** off, you can nudge the object into place using the arrow keys. Both the position and size of the object can be adjusted with the arrow keys.

## Form Background Colors

The default background for a form is white, but you may choose from 256 background colors for any form. To change the background color, open the **Form Preferences** dialog (in the Setup menu) and choose the background color from the pop-up menu.



# Chapter 15: Displaying and Editing Text



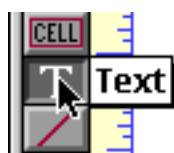
Graphics and icons are powerful tools, but written text is still the primary medium for communication and record keeping. Panorama forms can display both permanent text (for captions, titles, instructions, etc.) and textual information from the database.

## Displaying Text

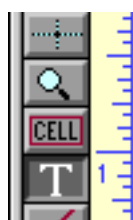
This chapter is divided into two sections, displaying text and editing text. In this first section we'll consider displaying text, both fixed text and text that changes depending on the information in the database or in a variable.

### Fixed Text Objects

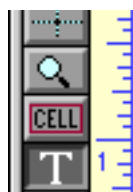
Panorama has two different kinds of fixed text, **click text** and **auto-wrap text**. Both types of fixed text are created with the **Text** tool.



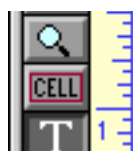
Once the **Text** tool is selected you can create click text simply by clicking on an empty spot on the form.



Now you can simply start typing.



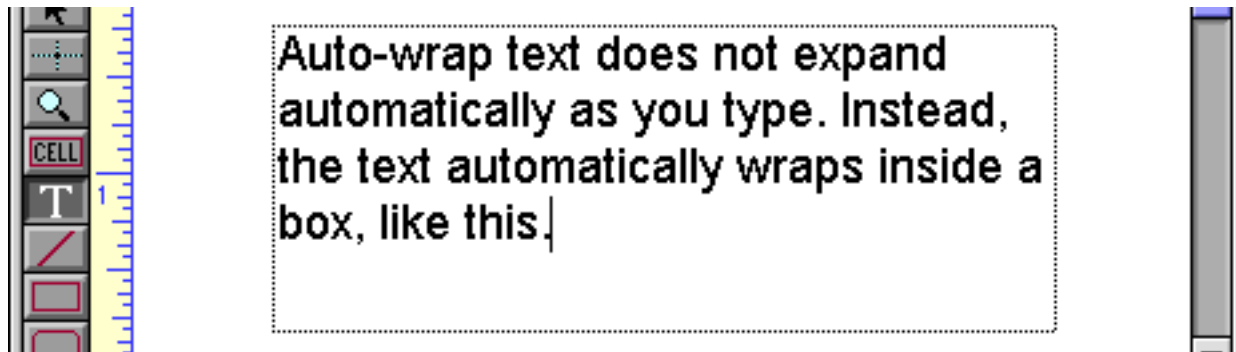
As you type each character, the click text object automatically expands.



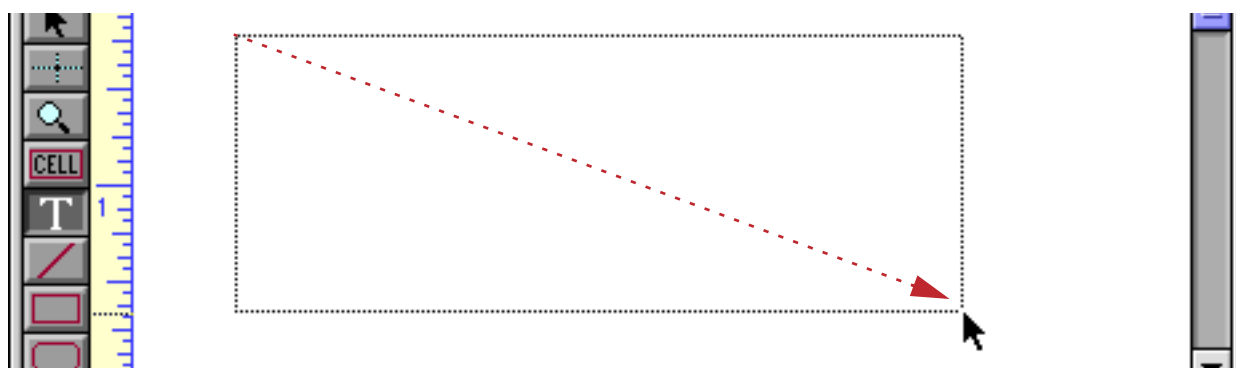
You can add additional lines to the click text by pressing the **Return** key.



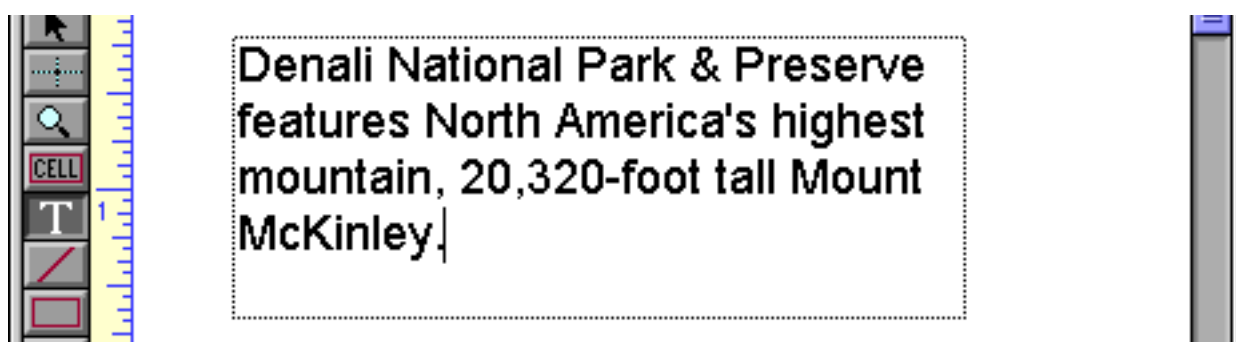
**Auto-wrap text** does not expand automatically as you type. Instead, the text automatically wraps inside a box.



To create **auto-wrap text**, start by selecting the **Text** tool, just like for **click text**. But instead of clicking on the form, drag the mouse to define the size and location of the box (don't worry about exact positioning, you can adjust it later).



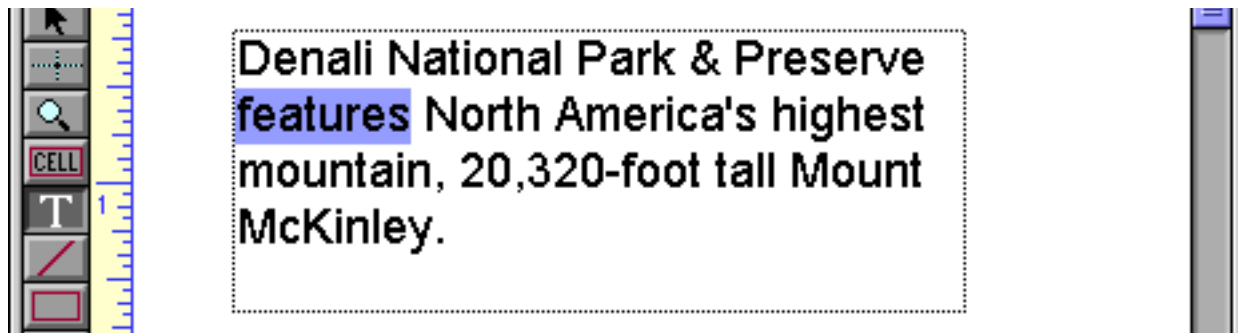
Once the box is defined you can start typing in text. The text will automatically wrap to the next line when it reaches the right edge of the box.



To help you keep track of the type of text you have created, the graphic editor displays a dotted border around all auto-wrap text objects. (This border disappears when the form is switched to data access mode.)

## Editing Fixed Text

To edit text within an object, you must use the **Text** tool. Once this tool is selected, move the mouse over the text you want to edit. When the mouse moves over the text it changes from an arrow to an I-beam. Use the I-beam to edit the text—click to select an insertion point, drag to select a range of characters, or double-click to select a word.



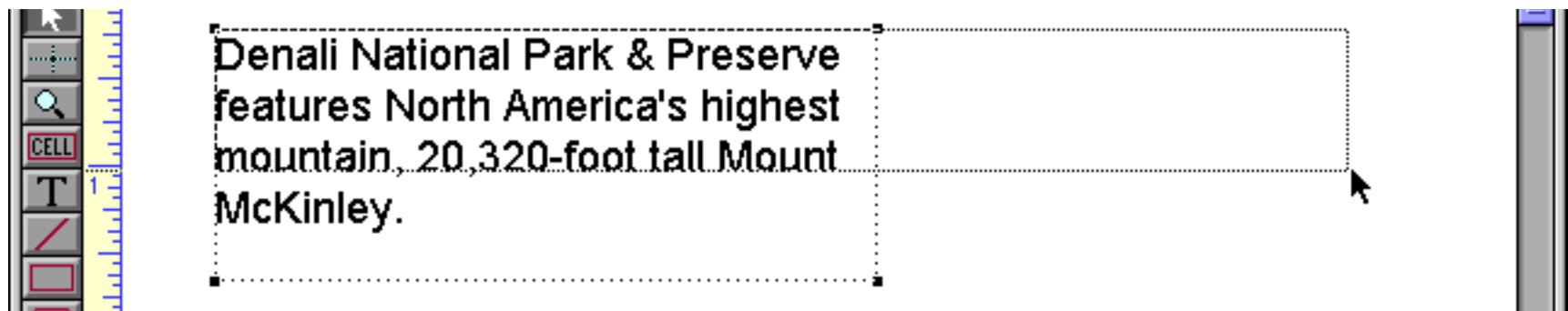
When you have finished editing the text, either click on the next text object you want to edit or click on the **Pointer** tool.

**Note:** If a text object doesn't contain any text at all, the entire object will be deleted from the form. The first thing you should do after creating a text object is to type at least one character into it.

## Moving and Resizing Fixed Text Objects

Text objects can be moved just like any other shape. Use the **Pointer** tool to drag the text to the new location.

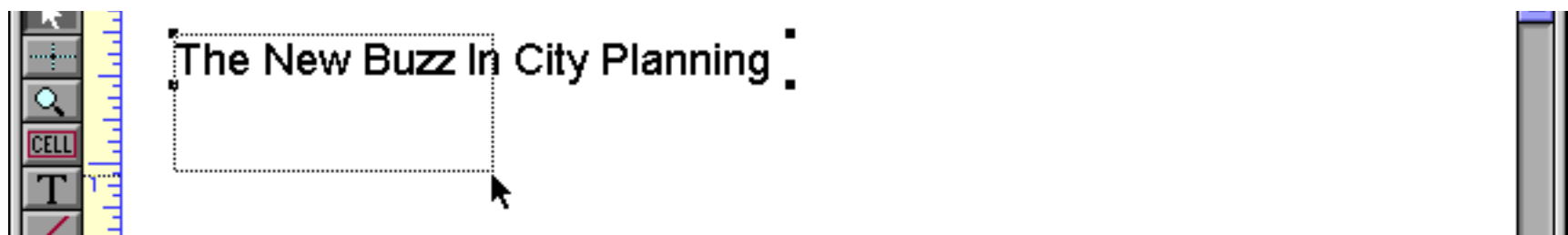
The size and shape of a text object can be changed by dragging one of the handles to a new position.



If this is done to auto-wrap text, the text will re-flow into the new size.



If you change the size of click-text, it will be converted into auto-wrap text.

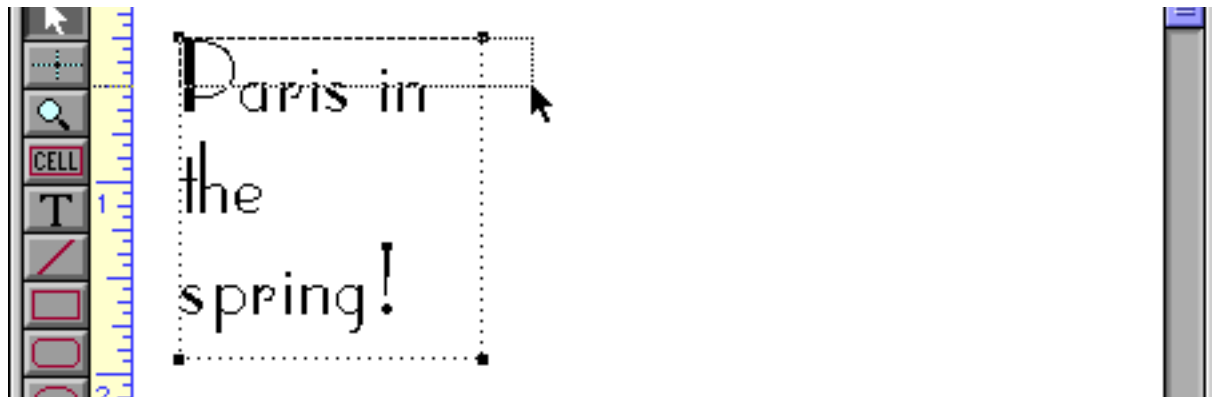




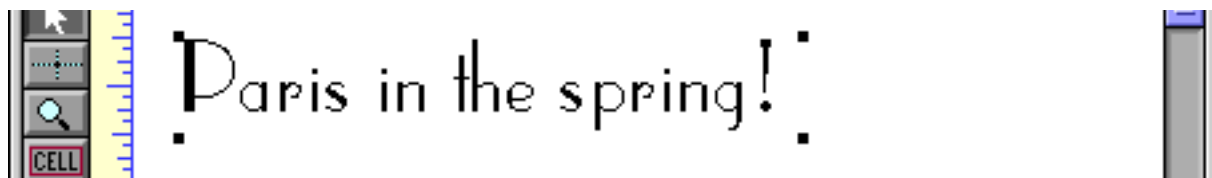
Once the click-text is converted to auto-wrap text it will re-flow into the new size.



To convert auto-wrap text into click text, resize the object so that it is less than one line high or less than one character wide.



Panorama converts the auto-wrap text into click text.



Later the text can be converted back into auto-wrap text simply by expanding the height again.

### Text Font, Size and Style

Text in a form may be displayed using any font installed in your system. However, you cannot mix different text styles, sizes, or fonts within a piece of text. To change the font, size, or style of an entire text object, use either the **Pointer** or the **Text** tools to select the object (or objects), then change the text appearance by choosing from the **Font**, **Size**, and **Style** Menus. You can also change the color of the entire text object with the **Color** menu (in the **Graphics** menu or the **Graphic Control Strip**).

If you need a size that is not listed in the **Size** menu pick **Other**. You can also change the text size in 1 point increments by choosing **Up** or **Down** from the **Size** menu.

### Text Alignment

Text is usually aligned flush left within the text object. Use the **Left Justify**, **Center**, and **Right** commands (**Text** menu) to change the alignment of the text.

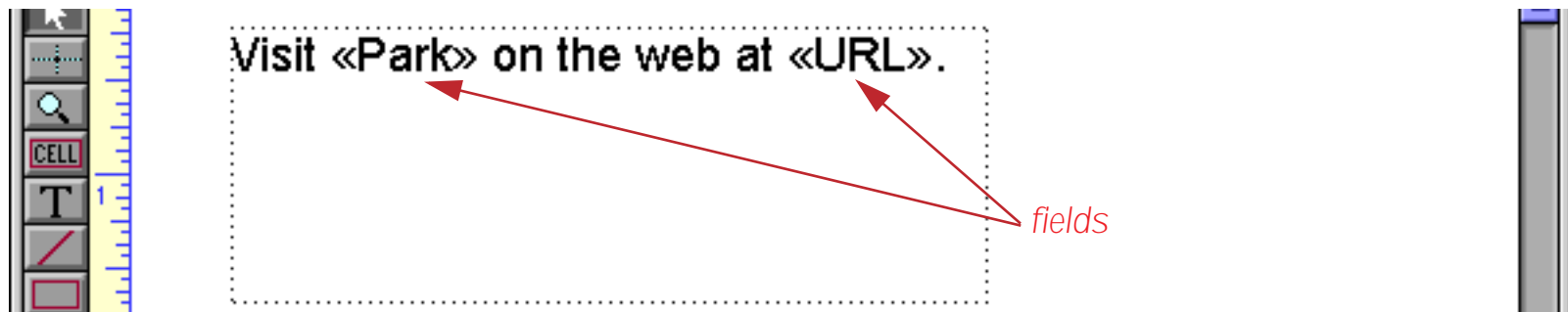
## Displaying Data in Auto-Wrap Text

Auto-wrap text objects are not limited to fixed text. They can also be used to display data, either alone or in combination with fixed text.

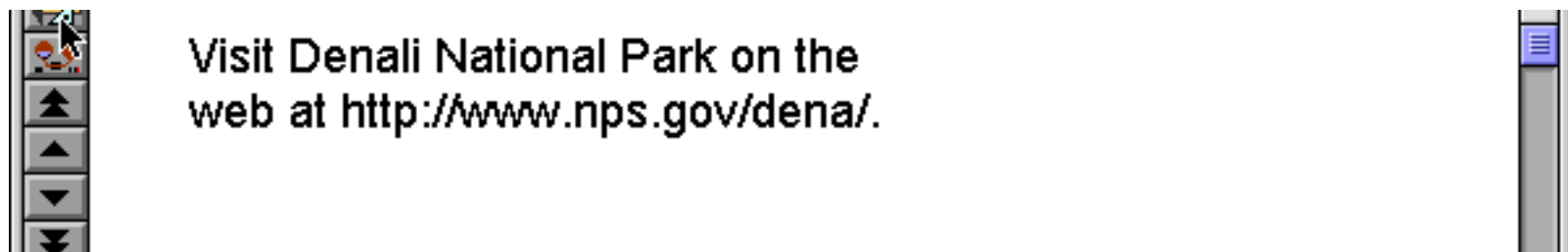
To display a field into the middle of an auto-wrap text object, type the name of the field into the text. Surround the name with the « and » chevron characters (for example «**First Name**» or «**Zip Code**»). On a Macintosh the « chevron is **Option-\*\*** and the » chevron is **Shift-Option-\*\***. On Windows systems the « chevron is **Alt-0171** and the » chevron is **Alt-0187**. To illustrate this technique, suppose you had a database of national parks like this.

Park	Address	City	Sta	Zip	Phone	Fee	URL
Cumberland Island National Seast	P.O. Box 806	St. Marys	GA	31558	(912) 882-4336	\$4.00	<a href="http://www.nps.gov/cuis/">http://www.nps.gov/cuis/</a>
Death Valley National Park	P.O. Box 579	Death Valley	CA	92328	(760) 786-2331	\$10.00	<a href="http://www.nps.gov/deva/">http://www.nps.gov/deva/</a>
<b>Denali National Park</b>	P.O. Box 9	Denali Park	AK	99755	(907) 683-2294	\$5.00	<a href="http://www.nps.gov/dena/">http://www.nps.gov/dena/</a>
Everglades National Park	40001 State R	Homestead	FL	33034	(305) 242-7700	\$10.00	<a href="http://www.nps.gov/ever/">http://www.nps.gov/ever/</a>
Fire Island National Seasore	120 Laurel Str	Patchogue	NY	11772	(631) 289-4810	\$0.00	<a href="http://www.nps.gov/fiis/">http://www.nps.gov/fiis/</a>
Gettysburg National Military Par	97 Taneytown I	Gettysburg	PA	17325	(717) 334-1123	\$0.00	<a href="http://www.nps.gov/gett/">http://www.nps.gov/gett/</a>
Glacier National Park	P.O. Box 128	West Glacier	MT	59936	(406) 888-7800	\$5.00	<a href="http://www.nps.gov/glac/">http://www.nps.gov/glac/</a>

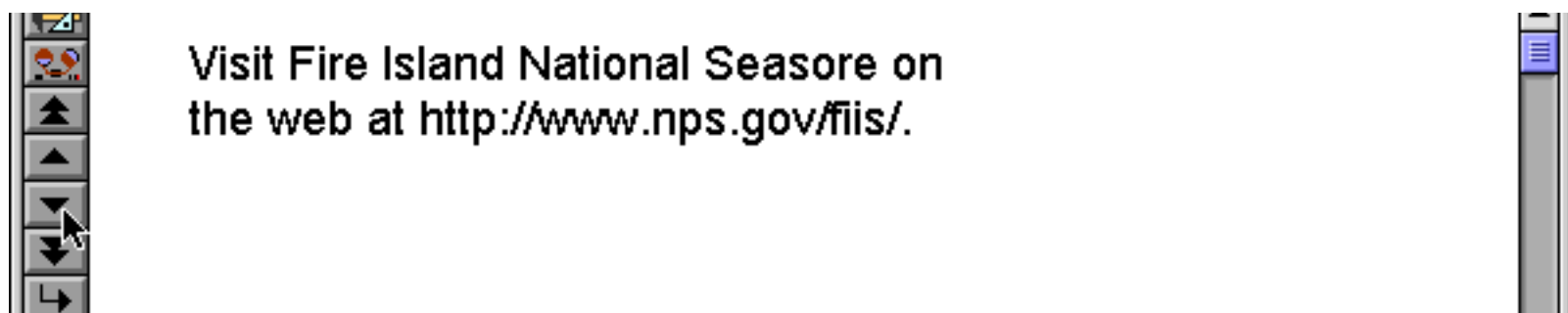
Now you can create an auto-wrap text object that contains fields, like this.



When the form is switched to Data Mode) Panorama will substitute the actual data in this fields, like this.



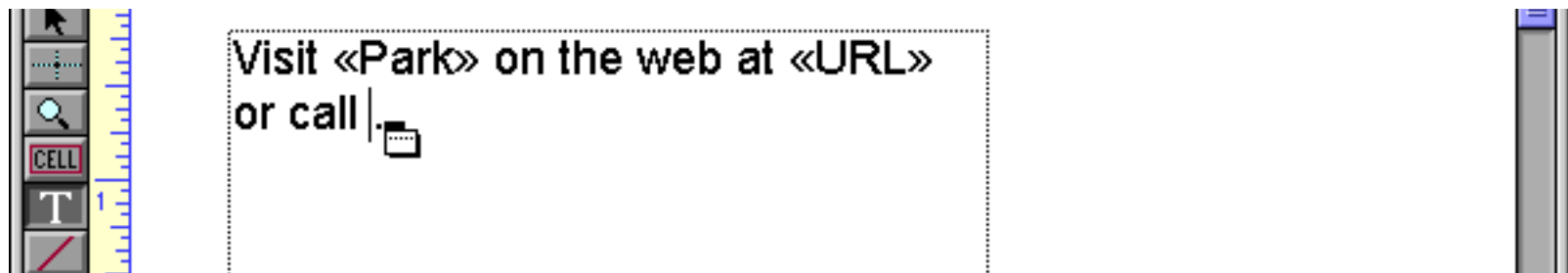
As you move from record to record, the substituted text will change appropriately.



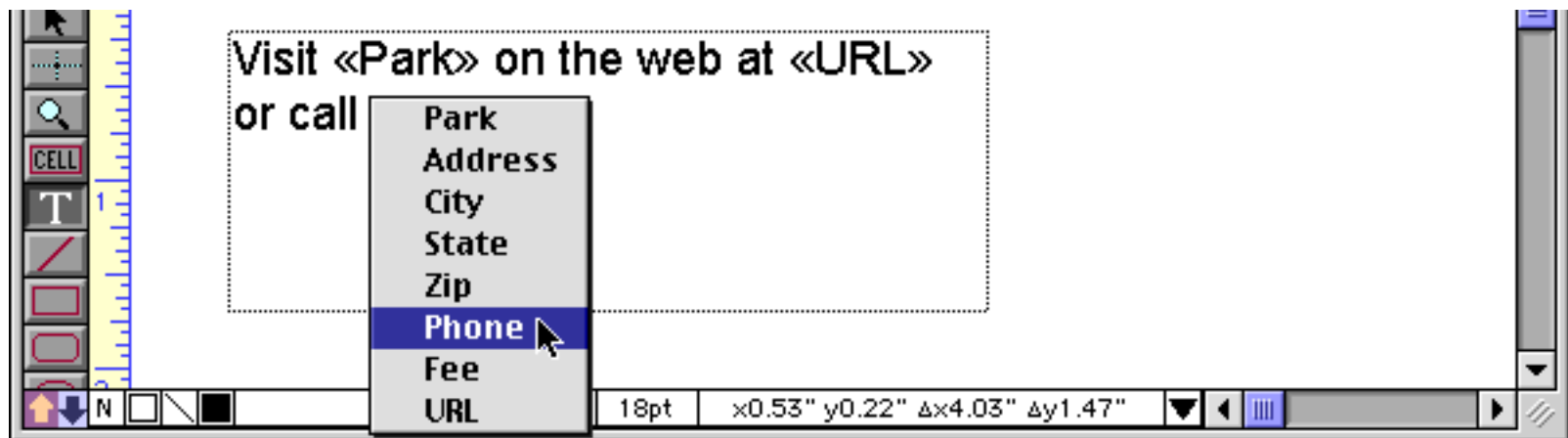
Since this technique “merges” the database information with the fixed text, it is sometimes called **data merging**.

### Data Merge Pop-Up Menu

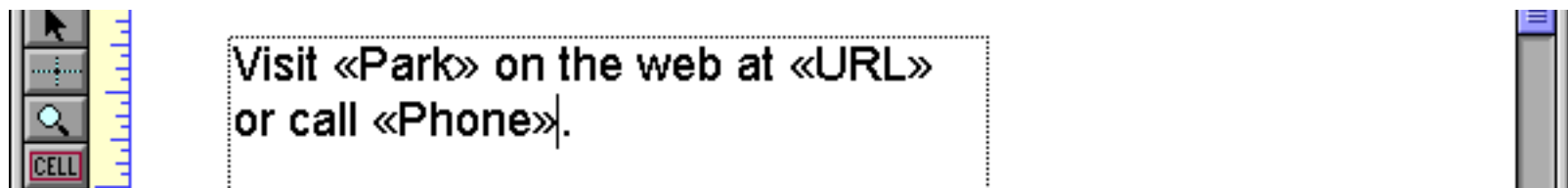
Typing in exact field names with chevrons can be a pain, so Panorama has a pop-up menu that can type in the field names for you, including the « and » chevrons. To use this menu, first select the **Text** tool. Then click on the text to create an insertion point. Once the insertion point is set, press either the **Command** key (Macintosh) or **Control** key (Windows) to change the cursor from an I-beam to a tiny menu icon.



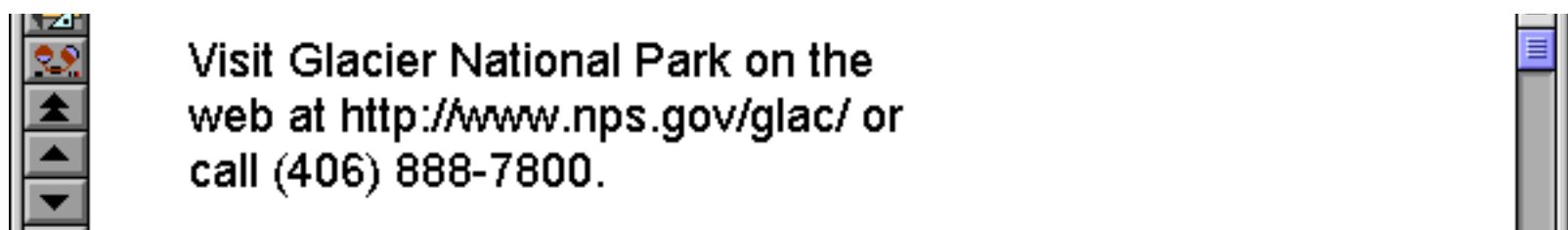
With the **Command/Control** key still held down, press the mouse to activate the pop-up menu



then pick the field name you want to insert.

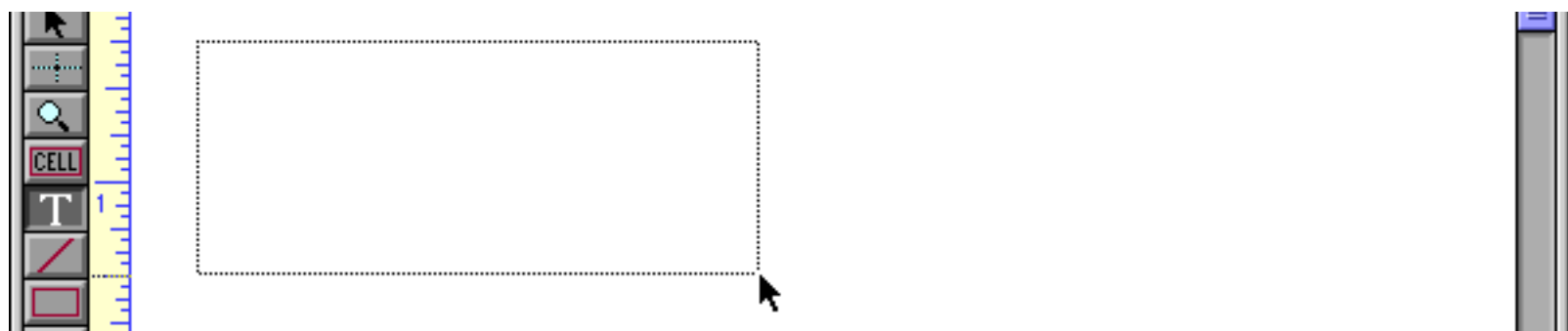


Switch back to Data Access Mode to see the final result.

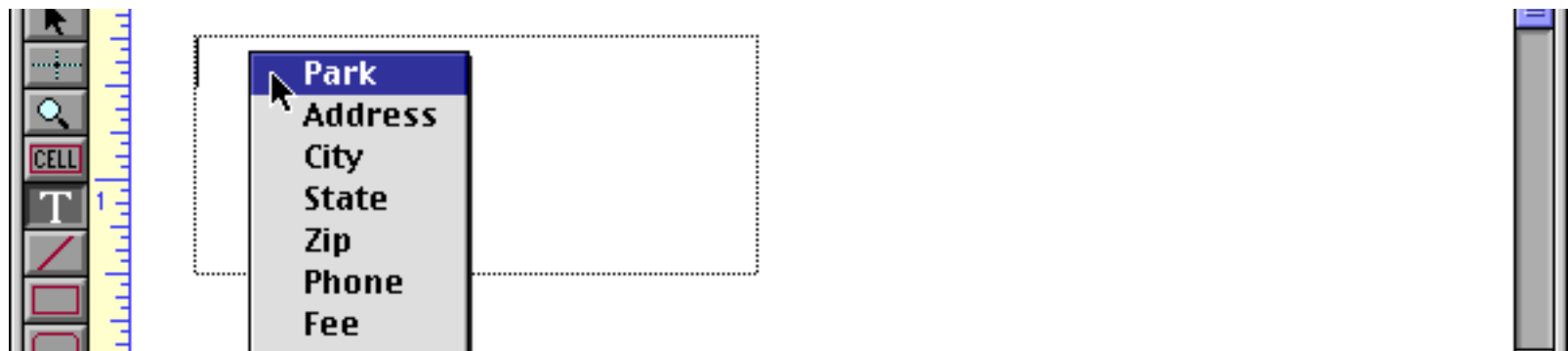


### Using Data Merge to Create Address Labels

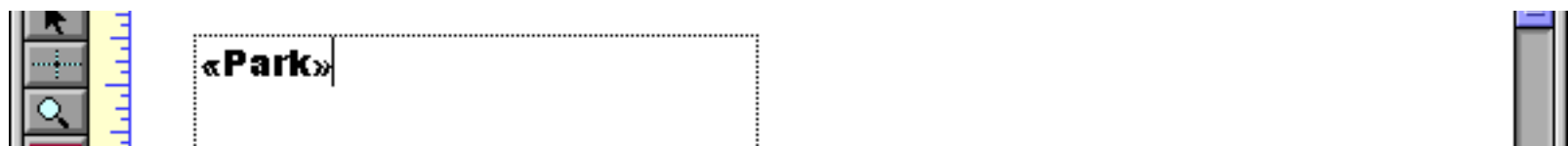
Data merge is an excellent way to create address labels. To create an address label using data merge, start by creating an auto-wrap text object the size of the label.



Hold down the **Command** key (Mac) or **Control** Key (Windows) and select the first field name from the pop-up menu.



When you release the mouse button Panorama will insert the field name. The insertion point is at the end of the line.



The first line is complete, so press **Return** to advance to the second line.



Repeat the same steps for the second line: hold down the **Command/Control** key, select **Address** from the pop-up menu, and press **Return**.



The third line contains three fields: City, State and Zip. Start by using the pop-up menu to enter the City field. Then press the **Comma** and **Space Bar** keys.



Finish the label by inserting the State field using a pop-up menu, typing a **Space** and then inserting the Zip field.



When you switch back to Data Access Mode Panorama will substitute the actual data. Voila! A label!

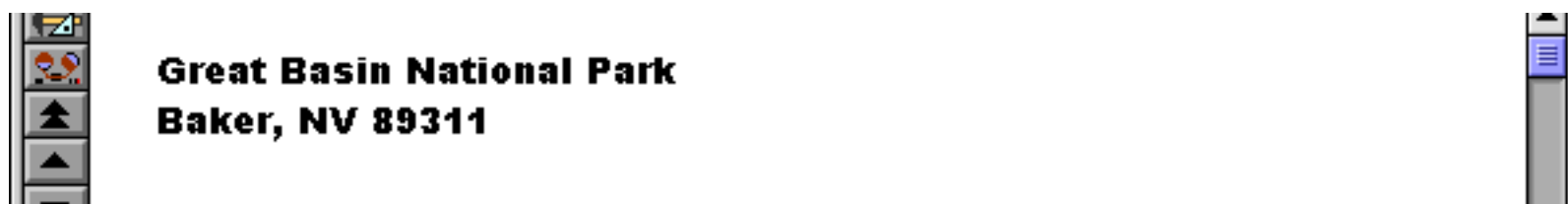


Panorama will automatically wrap the text within the rectangle you provided. It will start a new line whenever you have typed a **Return** (or when a line becomes too long to fit).

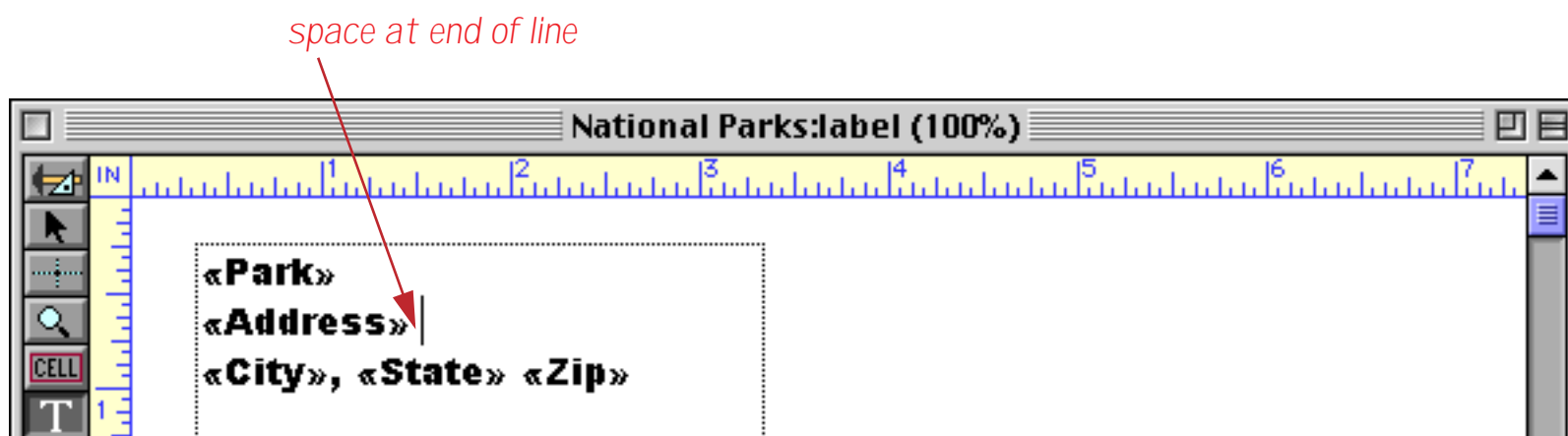
If a field is empty and that causes the entire line to be empty, Panorama will completely remove the line. For example, Great Basin National Park doesn't have a street or P.O. Box, as you can see in the data sheet.

Grand Canyon National Park	P.O. Box 129	Grand Canyon	AZ	86026	(520) 638-2651	\$10.00	http://www.nps.gov/grca/
Grand Teton National Park	P.O. Drawer 17	Moose	WY	83012	(307) 739-3300	\$20.00	http://www.nps.gov/grte/
Great Basin National Park		Baker	NV	89311	(775) 234-7331	\$0.00	http://www.nps.gov/grba/
Great Smokey Mountains National Park	107 Park Headquarters	Gatlinburg	TN	37738	(865) 436-1200	\$0.00	http://www.nps.gov/grsm/

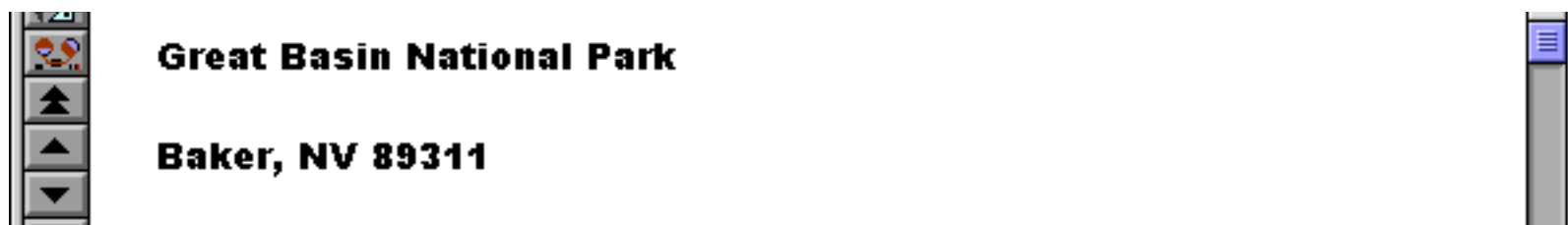
In the label, Panorama will remove the completely blank line.



If you don't want the blank line removed, put a space at the end of the line.



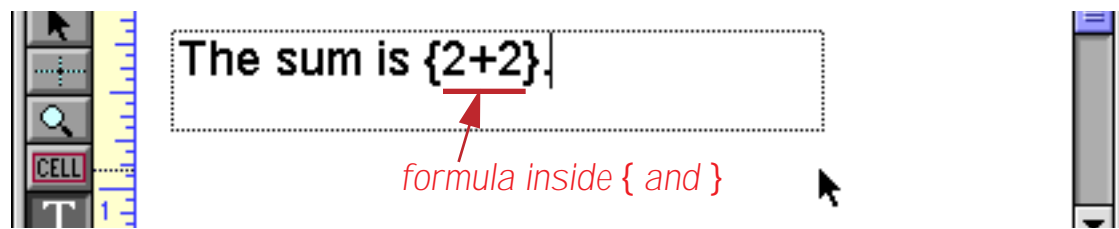
Now the line can never be completely empty, so Panorama will not remove it.



An address label can be used as part of a larger form (like an invoice), or it can be used by itself as a mailing label. If you wish to print a mailing label you must define the overall size of the label by creating one or more report tiles. Report tiles tell Panorama how to print a form. For more information on creating and printing mailing labels see Chapter 22.

## Displaying Formulas in Auto-Wrap Text

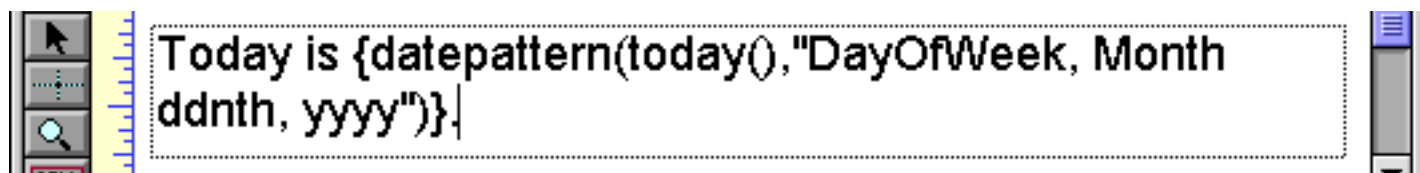
In addition to fixed text and fields, auto-wrap text can also contain complete formulas with text and numeric calculations. Simply type the formula into the text, surrounded by { and } curly brace characters. Here's what a formula looks like in Graphics Mode.



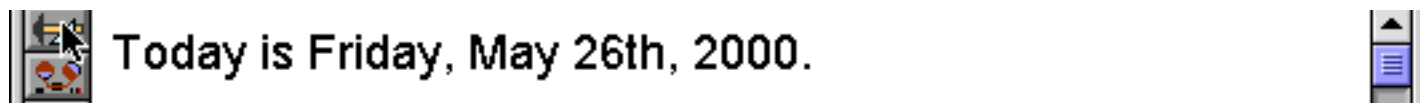
When the form is displayed in Data Mode Panorama substitutes the result of the formula instead of the formula itself.



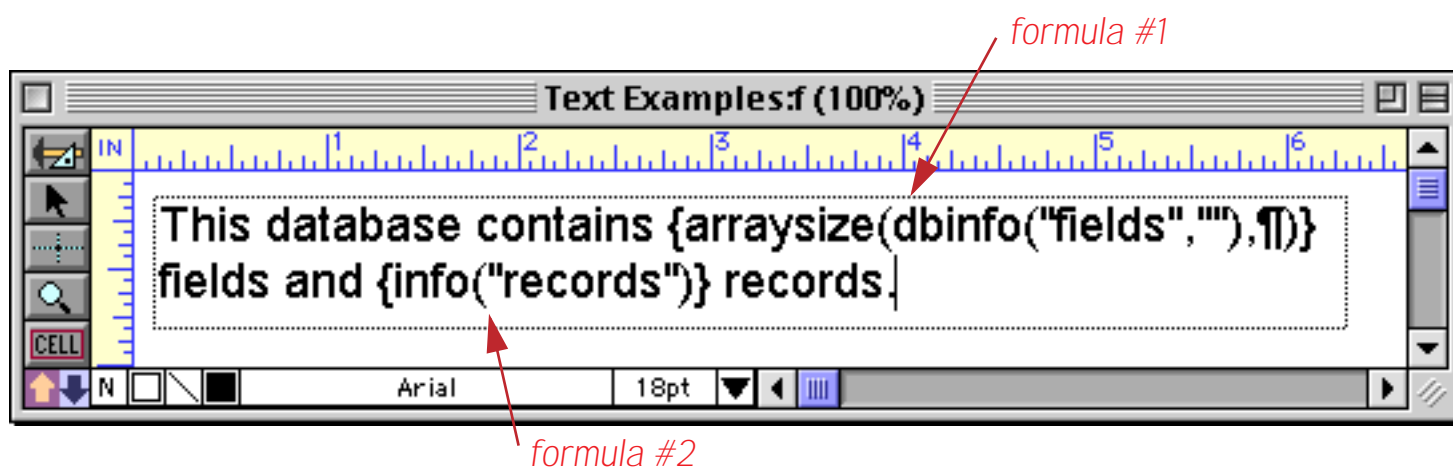
Of course 2+2 is a pretty silly formula. More useful formulas can be formed by combining fields, variables, and functions, like this.



In Data Mode this auto-wrap text object looks like this:



An auto-wrap text object is not limited to a single formula. You can include as many formulas as you need.



Here is the same auto-wrap text object in Data Mode. The formulas have been replaced with their results.



Using a formula gives you almost unlimited possibilities for combining and manipulating data on the fly as it is displayed or printed. By using formulas containing the lookup function you can display or print data from more than one database at once. You can use a formula to display or print computed information that is not stored in the database. You can use true-false formulas to display or print data only if a certain condition is met. The possibilities are almost endless. See Chapter 23 for more information about formulas.

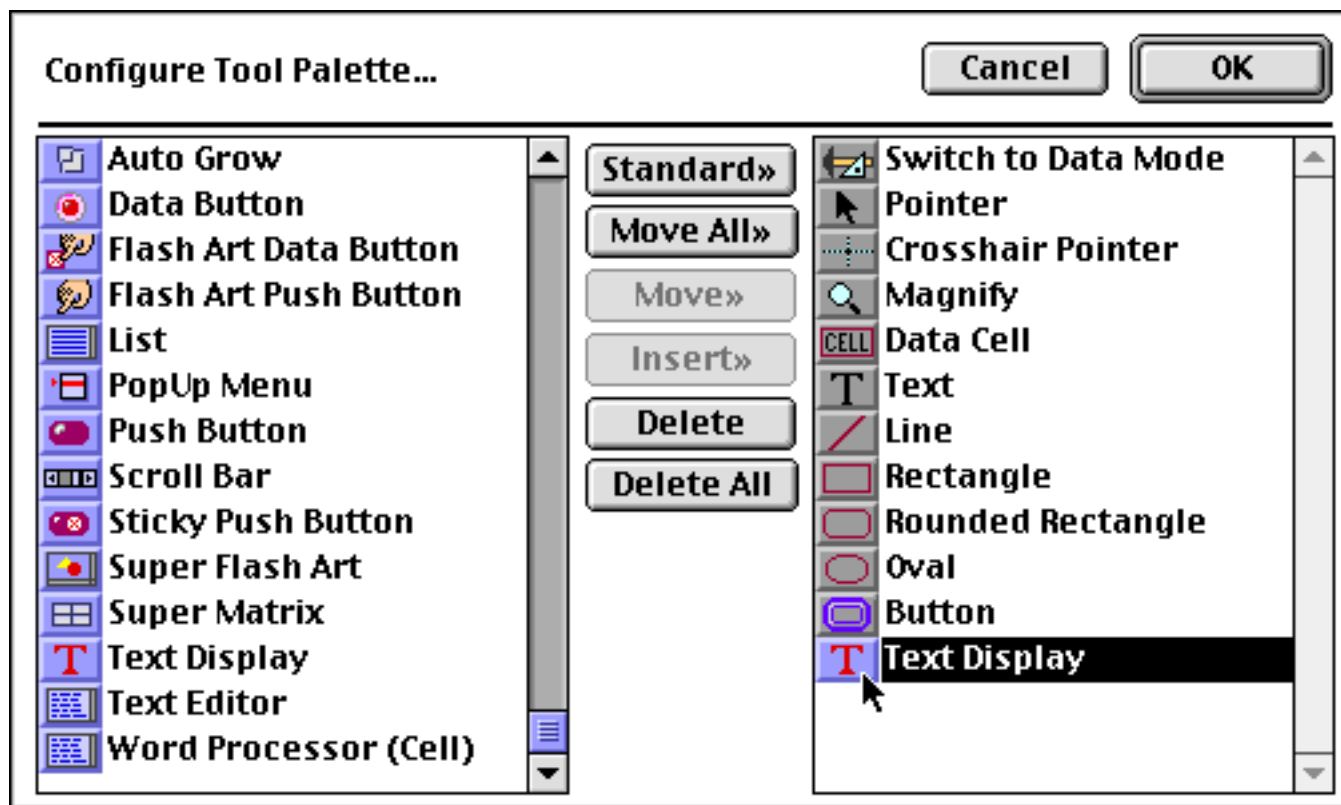


## Text Display SuperObjects™

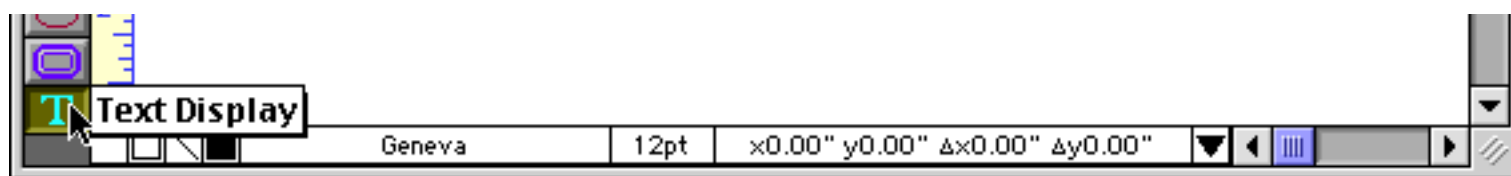
The **Text Display SuperObject** displays text based on a formula. In some ways, this is similar to an auto-wrap text, but there are many more options for calculating the formula and formatting the displayed text. You can store the formula itself in a variable (so it can be changed on the fly), align the text in any corner of the object, automatically scale the text for different size windows, and even change the color of the text on the fly.

### Creating and Modifying Text Display SuperObjects

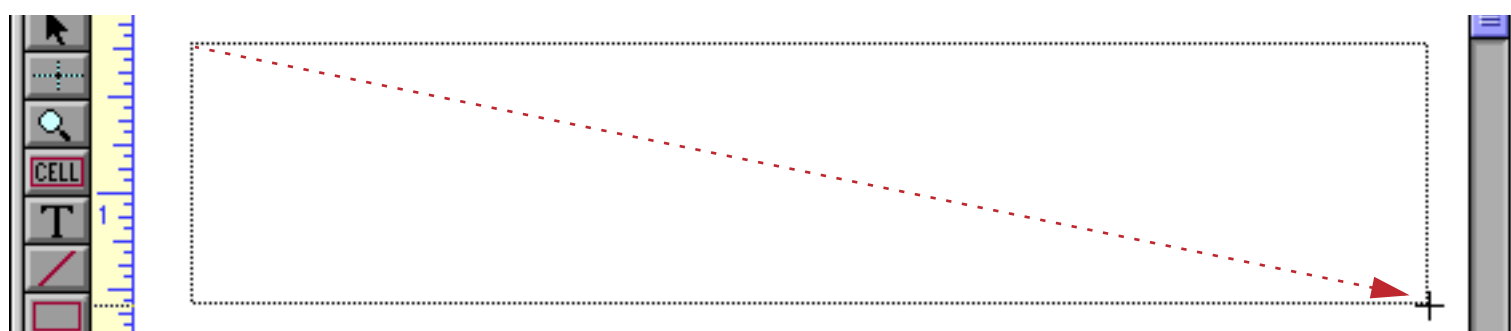
The Text Display SuperObject tool is not in the default tool palette, so you'll need to use the **Tool Palette** dialog to add this tool to the palette if it is not already there (see "[Customizing the Tool Palette](#)" on page 252).



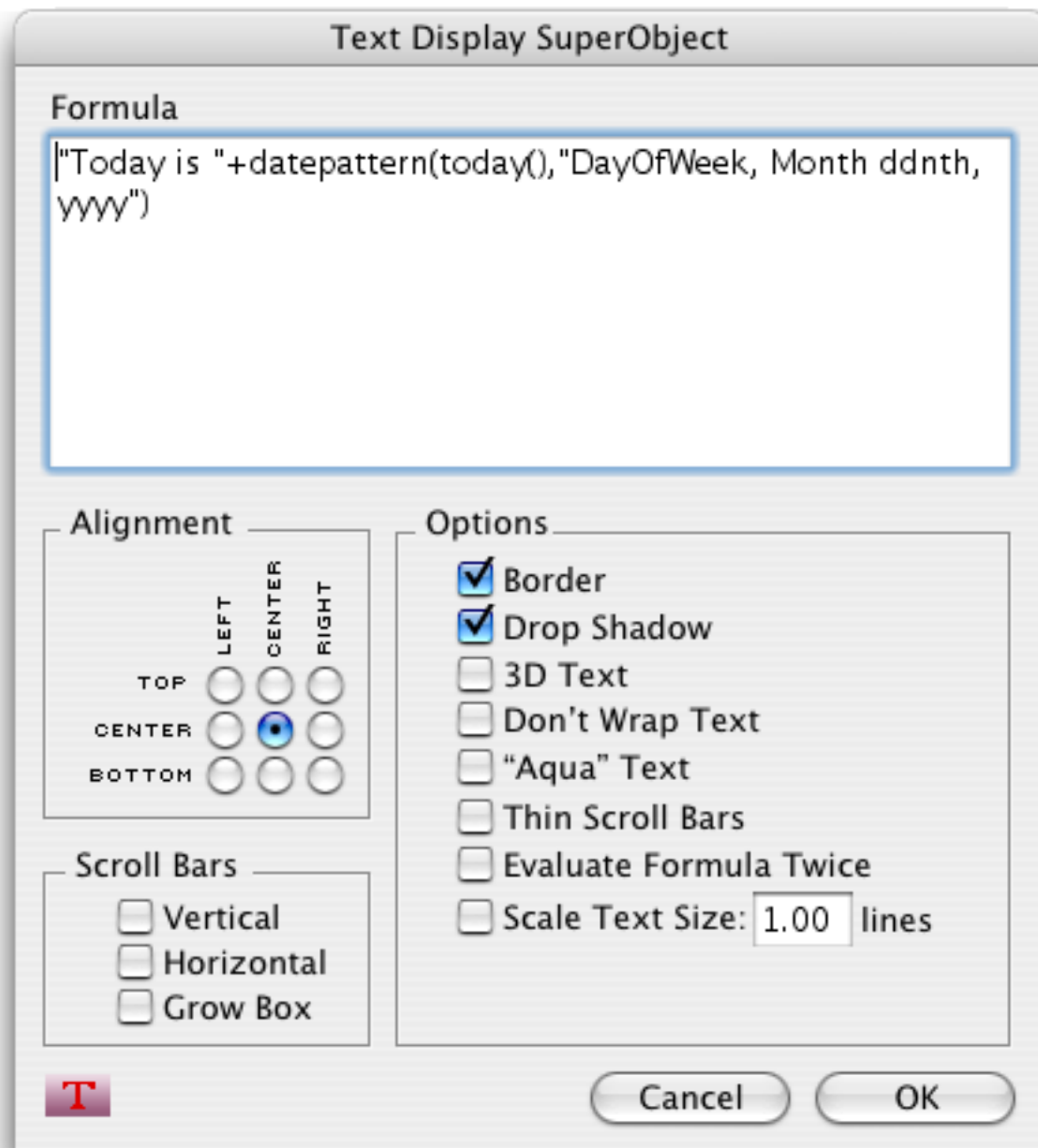
Now that the tool is added to the palette you can select it.



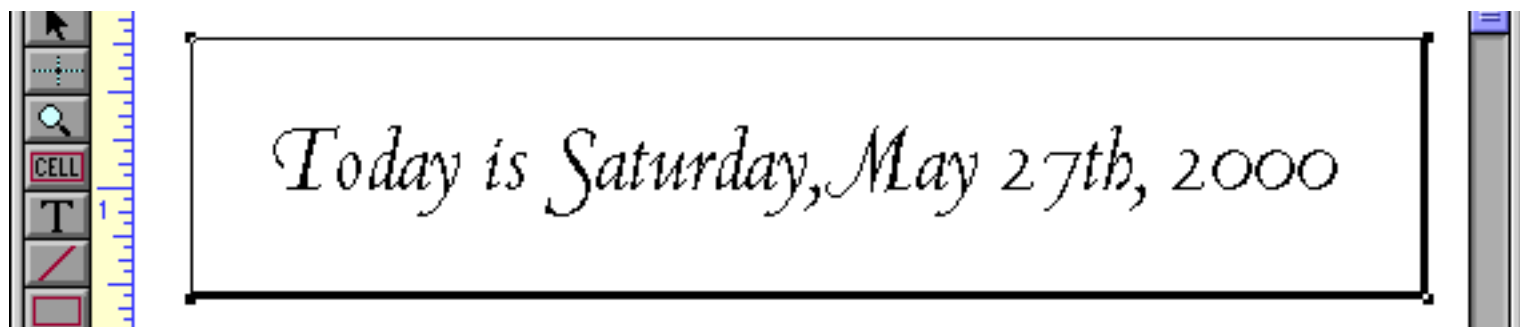
Once the tool is selected, drag the mouse across the form in the location where you want to create the Text Display SuperObject.



When you release the mouse, the Text Display SuperObject configuration dialog will appear.



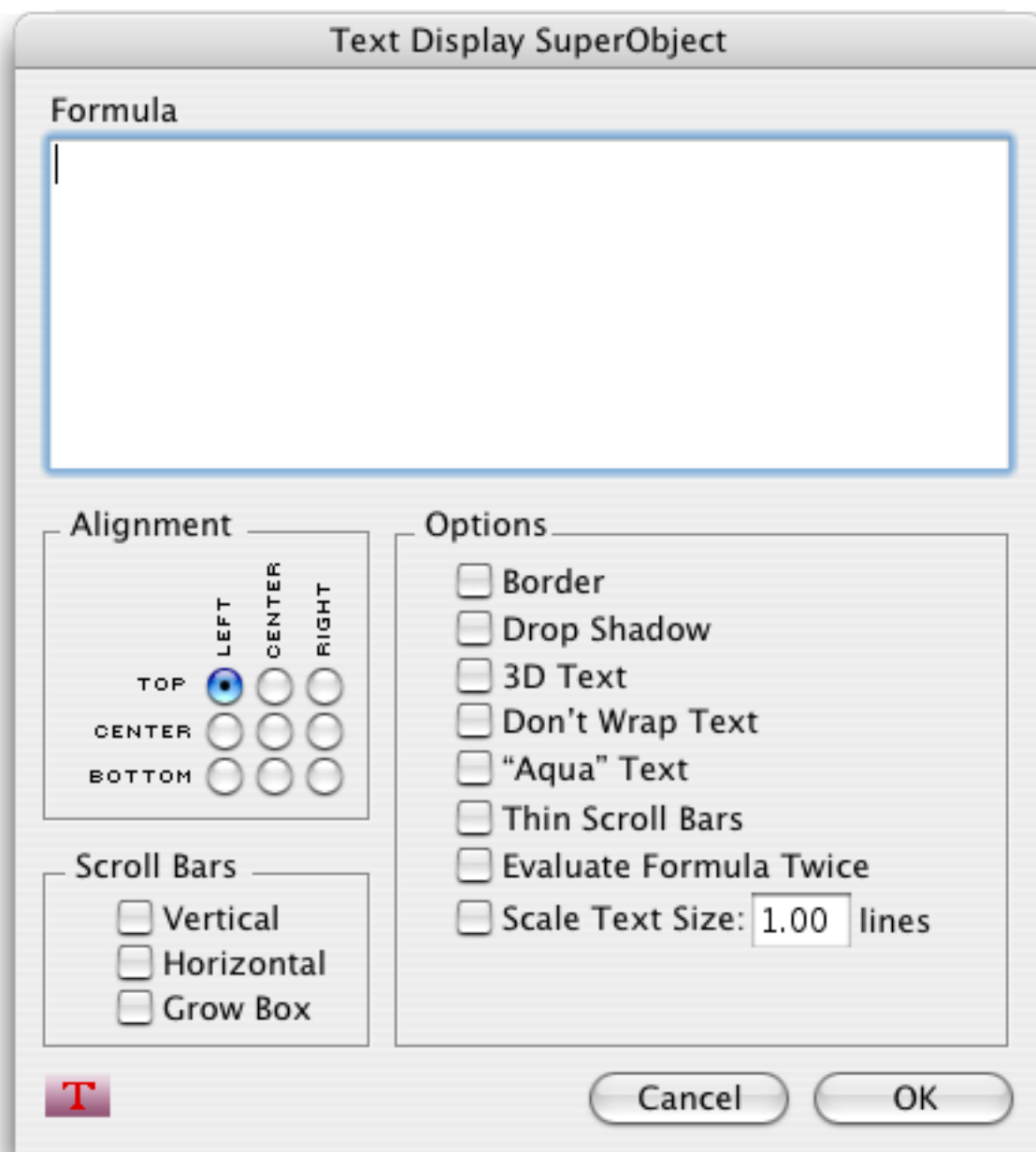
At a minimum you must enter a valid formula into the dialog. For this example we've also turned on the **Border** and **Drop Shadow** options and set the alignment to centered (these options are discussed in detail below). When the **OK** button is pressed the new object appears. (Notice that unlike the auto-wrap text object, the Text Display SuperObject shows the result of the formula in both Graphics Mode and Data Mode, not just Data Mode.)



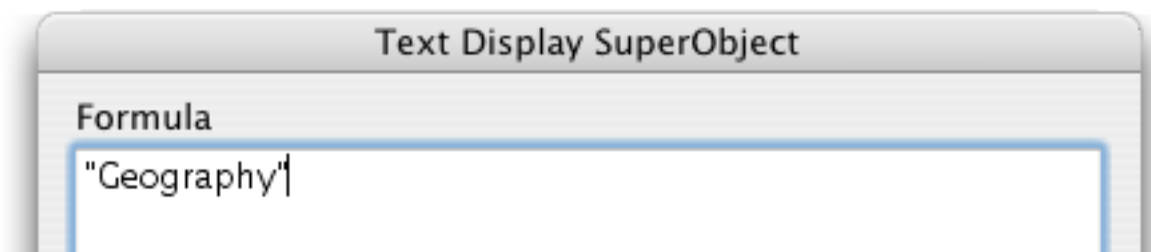
After it has been created you can modify the location, size, font, style and color of a Text Display SuperObject just like any other object. To change any of the object attributes (formula, border, alignment etc.) select the **Pointer** tool and double click on the object. The configuration dialog will appear again. Make your changes and press the **OK** button.

## Text Display Options

The **Text Display SuperObject** configuration dialog is divided into several sections.



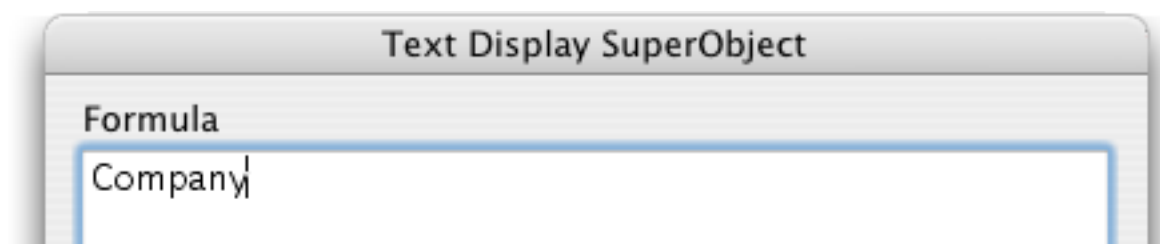
**Formula:** This section of the dialog specifies the formula for displaying text in this object. If you want to display fixed text, remember that you need to surround the text with quotes.



This object will always display the word **Geography**.

▪ *Geography* ▪

To display a field or variable, type in the name of the field or variable. You can use the **Field** menu to type in the name of a field for you (that way you don't have to worry about misspellings. Here's the formula to display the Company field.



And here is the finished object.

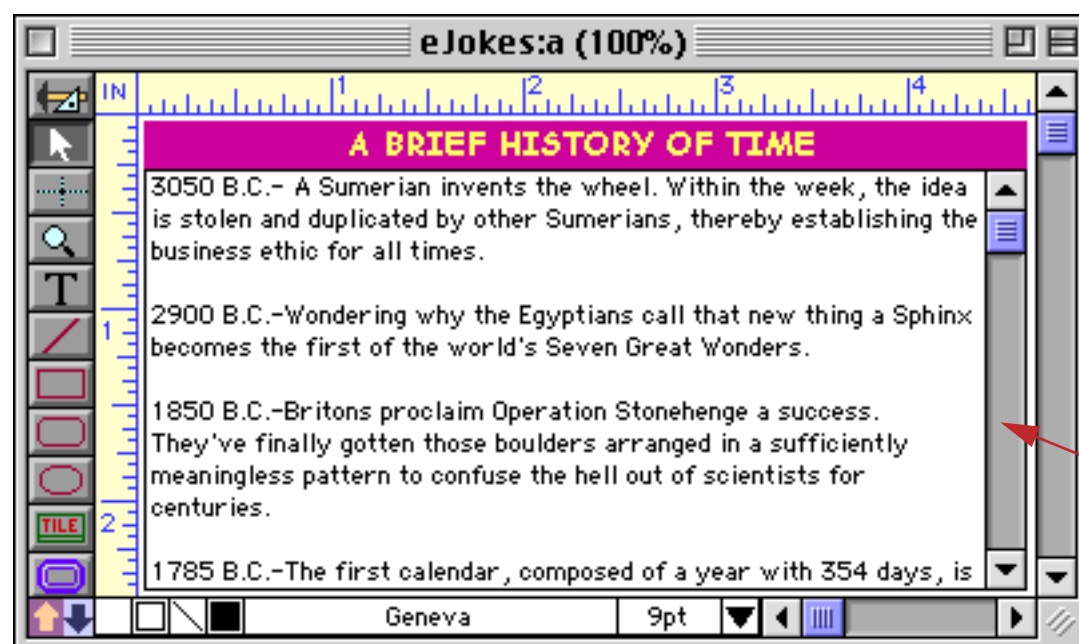
▪  
▪ **Austin Lumber** ▪  
▪

Panorama has hundreds of different functions that you can use to assemble your formula. See "[Using Formulas to Display Text](#)" on page 296 for some useful tips on building formulas for displaying text on a form. You'll find a complete description of formulas in "[Formulas](#)" on page 501.

**"Aqua" Text:** If this option is turned on the text will be smoothed (anti-aliased) if the operating system supports that feature.

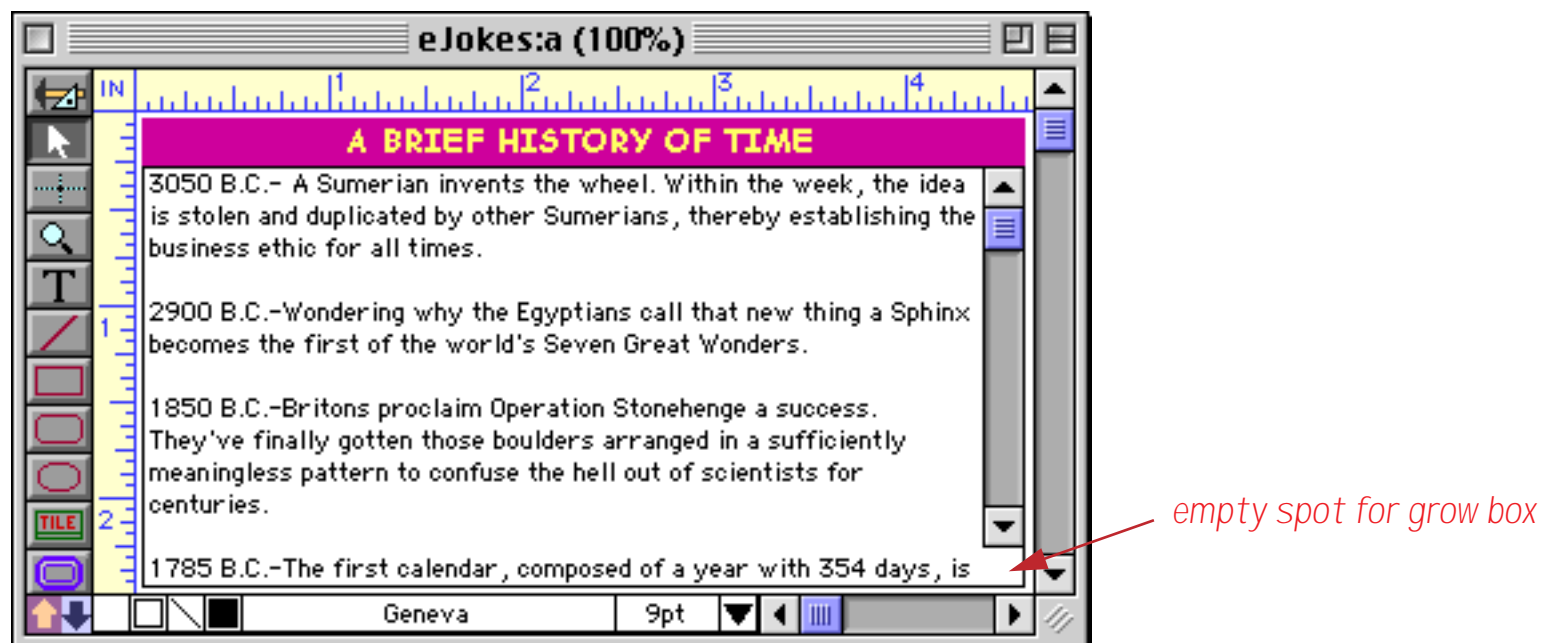


**Scroll Bars:** This section controls what scroll bars (if any) will be available when displaying this text, and whether space will be left for a grow box if only one scroll bar is used. Here is a Text Display SuperObject with the **Vertical Scroll Bar** enabled.

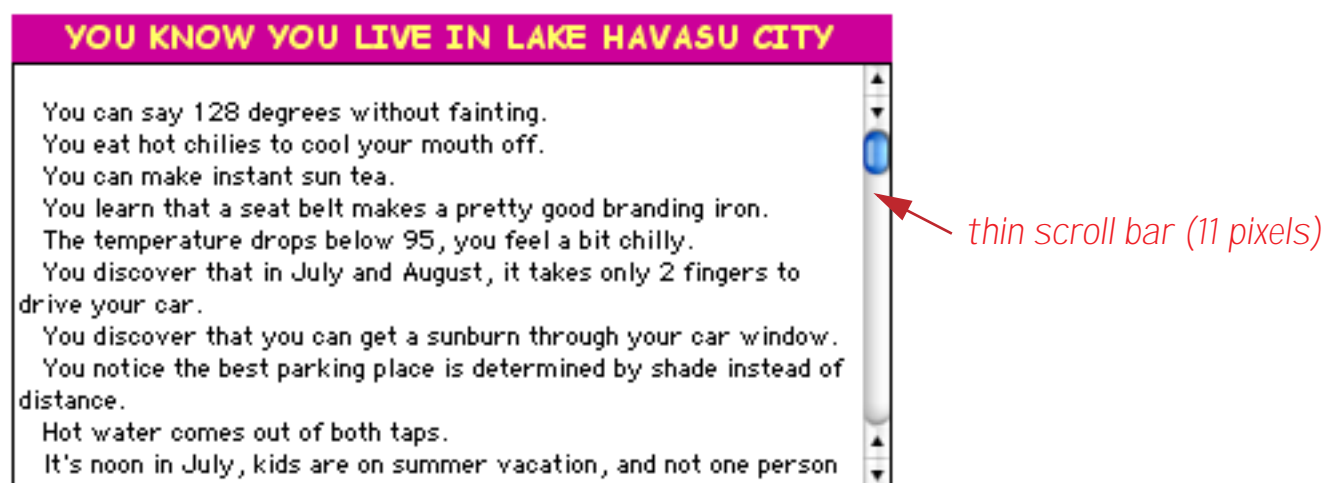


*Text Display scroll bar*

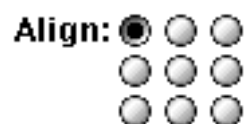
When both the **Vertical** and **Grow Box** options turned on, Panorama leaves an empty spot for a grow box in the lower right hand corner.



**Thin Scroll Bar:** Scroll bars are normally 16 pixels wide. When this option is checked the scroll bar will be only 11 pixels wide.



**Align:** This area contains nine radio buttons, allowing you to control the position of the text within the object.



You have the choice of left, center, or right and top, middle, or bottom. (You can also choose left, center, or right from the **Text** menu.) Note: If you have enabled the vertical scroll bar, you must choose one of the top three alignment buttons. If you have enabled the horizontal scroll bar, you must choose the top left alignment button.

**Other Options:** To learn about any of the other options not mentioned above see Chapter 15 of the **Panorama Handbook**.

### Using Formulas to Display Text

Panorama uses formulas to manipulate numbers and text. Using an auto-wrap text object, a Text Display SuperObject or a Text Editor SuperObject you can display the result of a formula on a form (and, since forms are used to produce reports, on a printed report).

There are an infinite number of ways to combine fields, variables and functions into useful formulas. In the following sections we will explore some of the more common types of formulas used to display information in forms.

### Combining Multiple Text Items Into One

Many times you'll need to combine several different fields or variables together, usually with captions and punctuation (carriage returns, commas, spaces etc.) In a formula two text items can be combined with the + operator.



This formula probably isn't what you had in mind, because the result (seen below in Data Mode) doesn't have a space between the first and last name.



Fixed text items (like captions, spaces and other punctuation) must be enclosed in quotes. Panorama allows several different kinds of quotes, as shown in this table.

Type	Open	Close	Example
Double Quote	"	"	"January"
Single Quote	'	'	'Tuesday'
Curly Braces	{	}	{San Francisco}
Smart Double Quote	“	”	“Gothic”
Smart Single Quote	‘	’	‘Bohemian’

Curly braces cannot be used to quote text in an auto-wrap text object, because they are used to surround the entire formula. Other than that you can use any one of these pairs of quote characters whenever you want.

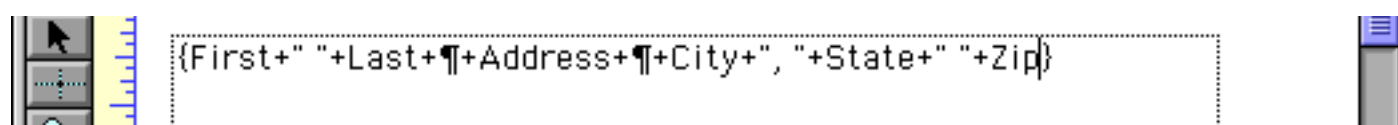
Now that we know how to quote a fixed text item we can add a space between the first and last names.



Switch to Data Mode and voila! The correctly formatted name appears.

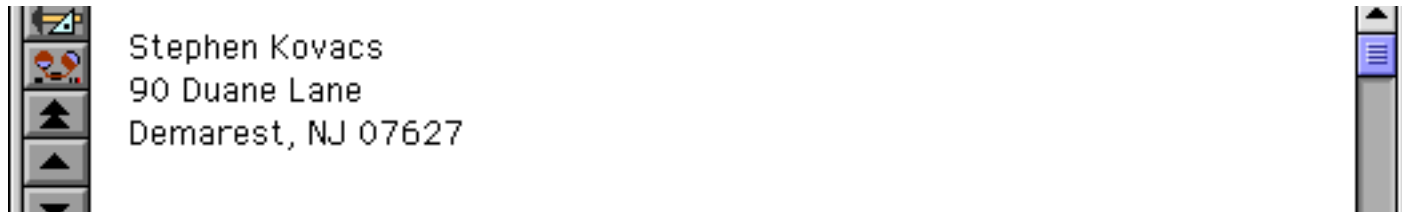


In a formula a carriage return is represented by the ¶ symbol. On the Macintosh you can enter this by typing **Option-7**. On Windows systems press **Alt-0182**. We can use this symbol to help build a complete address label.

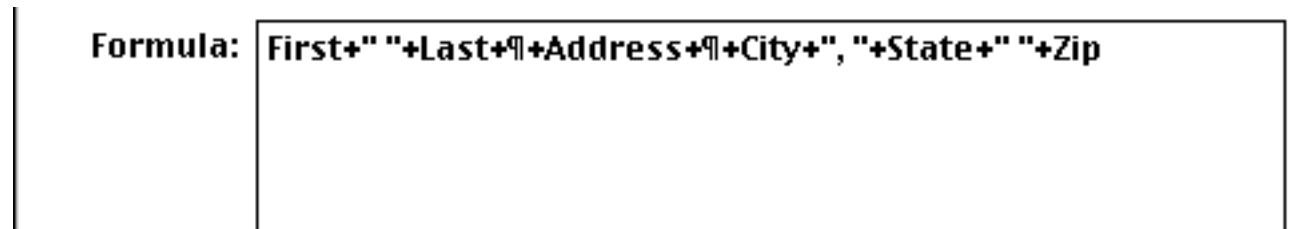




The formula appears all on a single line, but switching to Data Mode shows the finished label on three lines.



Our example used an auto-wrap text object, but the exact same formula could be used with a Text Display SuperObject. In Data Mode this object will look exactly like the previous example. (Of course, using a Text Display SuperObject would give you more options for aligning and scaling the text.)

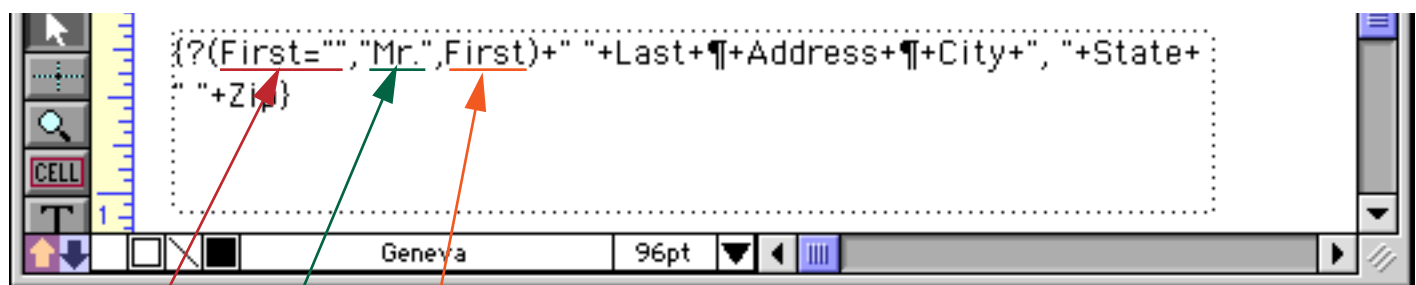


Note: The alert reader will have noticed that it is possible to create a label like this using data merge in an auto-wrap text object, without using all of these quotes and ¶ and + symbols. The data merge is simpler, so why bother with a formula? In this case there is no reason except to illustrate the ability to combine text items together. In the following sections, however, we will expand on this example to show applications that can only be done with a formula.

#### Creating a Smart Formula

In the real world, data often doesn't fit into neat little boxes. Some people will enter their middle initial, some won't. Some motels have off peak rates, some don't. Some countries measure temperature in Fahrenheit, some in Celsius. It takes a bit of work, but using the ?( and sandwich( functions you can set up formulas that display data correctly under changing, sometimes opposite circumstances.

The ?( function allows a formula to make a yes/no, either/or decision. For example, consider the address label created in the previous section. Suppose the first name is missing? Using the ?( function a formula can be constructed that substitutes Mr. for the missing first name.



*if First is not empty, then it will be included here*

*if First is empty, then "Mr." will be substituted where the first name normally goes*

*the function will make a decision based on whether First is empty (equal to "") or not empty*

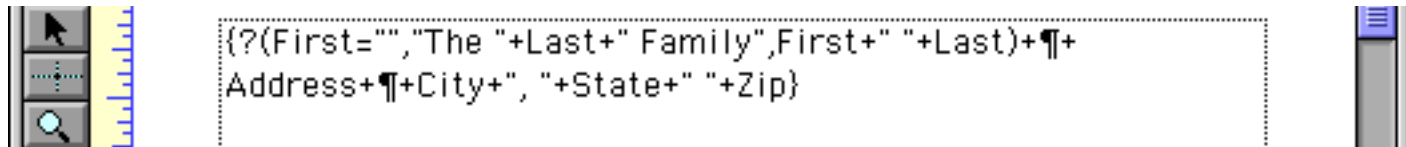
In data mode, anyone with a first name will simply display a standard label including the first and last names.



But if the first name is missing, Mr. will be substituted.



There are many ways you can use the `?(` function. Here is a slight re-arrangement of the previous example.



Here's what this formula produces if the first name is missing.



The `?(` function is a simple but very powerful tool. See [“The ? Function”](#) on page 557 for more detailed information about this function.

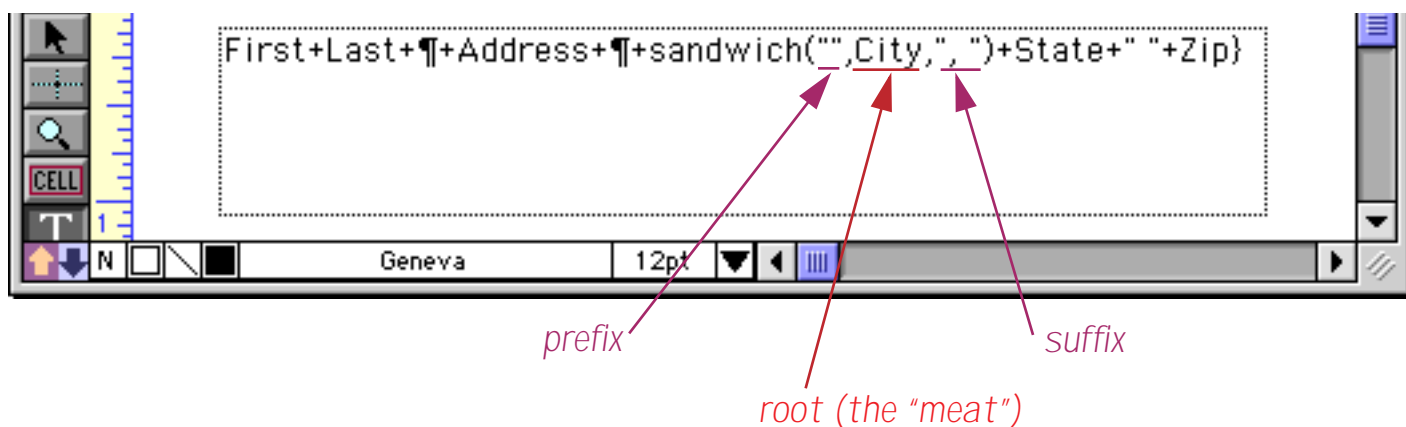
#### Eliminating Unnecessary Punctuation and Blank Areas With the Sandwich Function

Yes, Panorama actually has a function named `sandwich`! If an item of data is missing, you'll usually want to eliminate any punctuation that is associated with that item. For example, if the middle name is missing, you won't want to include the extra space. If the city is missing from an address, you'll want to leave off the comma afterwards, instead of leaving a comma hanging in the air like this.

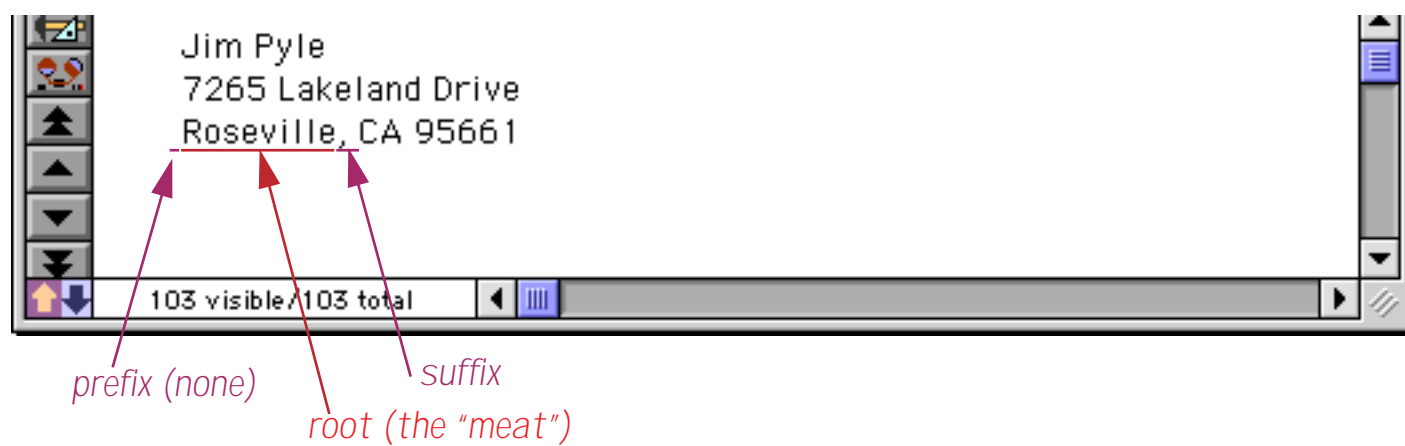


If the company name is missing from an address, you'll want to leave off the following carriage return so there won't be a blank line. All of these tasks can be performed with the `?(` function, but there's also an easier way: the `sandwich`( function.

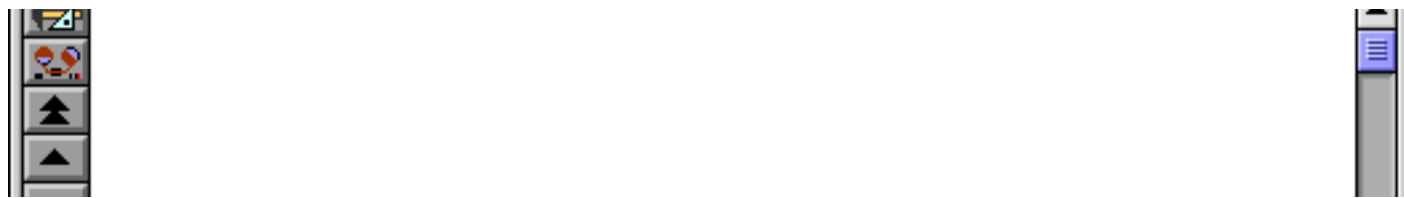
The `sandwich`( function has three parameters: `prefix`, `root` and `suffix`. The root is the main item of text you want to display. The `sandwich`( function will add the prefix and suffix to the beginning and end of the root, kind of like slapping bread around a slice of salami. However, if the root is empty, the `sandwich`( function won't “slap on the bread.”



The results of this function depend on whether or not the **City** field contains any text. If it does, Panorama adds the prefix (which in this case is empty) and the suffix.



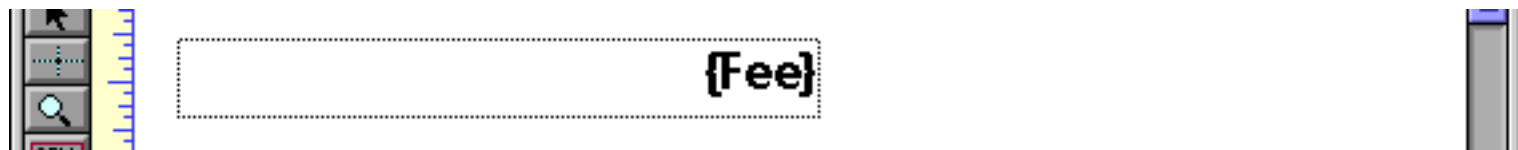
If the **City** field is empty, Panorama leaves out the prefix and the suffix also. Here's our empty record again, but this time, no comma hanging in the middle of the air!



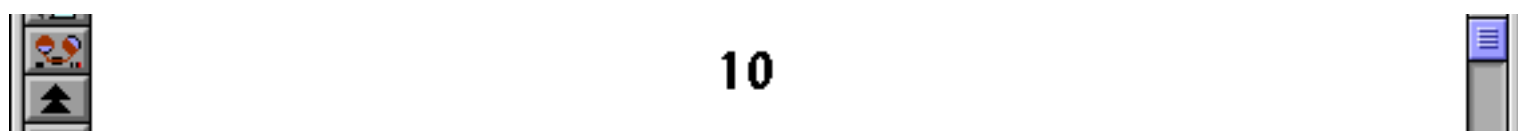
You'll find that the **sandwich()** function is very delicious any time you need to conditionally include spacing or punctuation around a field that might be blank. (Sorry, couldn't resist.)

#### Combining Numbers with Text

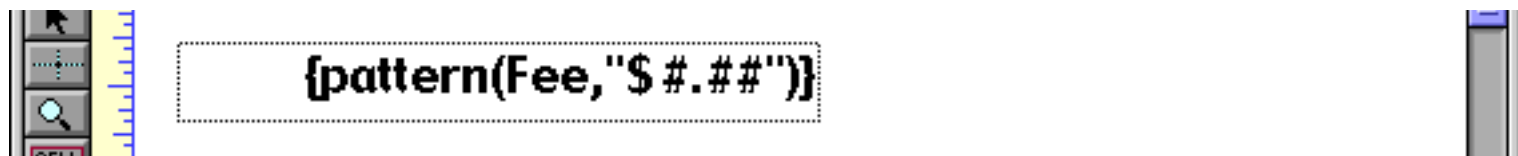
If a formula contains nothing but a single numeric value, Panorama will automatically convert the value to text for you, like this.



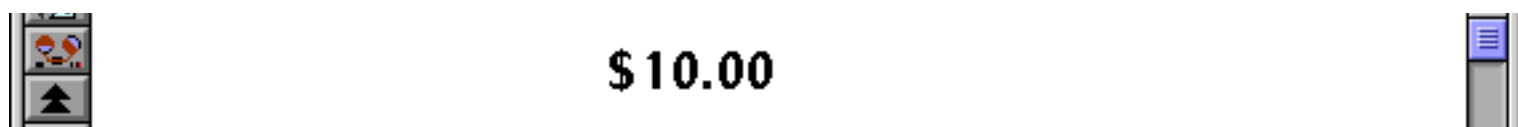
Panorama will decide for itself what format to use for the number.



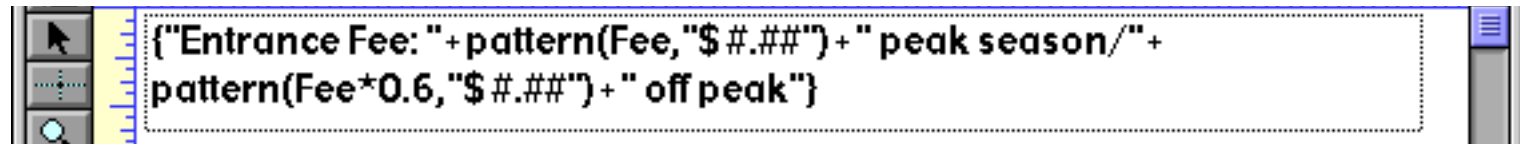
If you don't like the format that Panorama chooses you can use the **pattern()** function to specify the exact format you want to use.



The **pattern()** function gives you total control over the format of the final number. See "[Converting Between Numbers and Strings](#)" on page 538 in the **Panorama Handbook** for a complete description of this function.

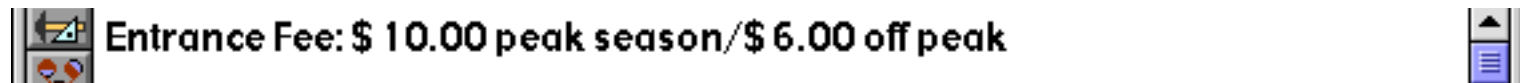


If your formula results in more than a single number (for example two numbers or text and a number) you must convert the numbers to text before they can be used in the formula. This must be done with the `str()` or `pattern()` functions as shown in this example.



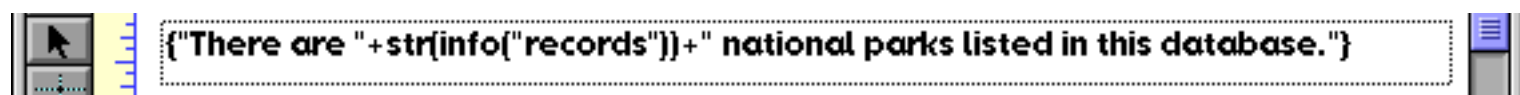
```
{ "Entrance Fee: " + pattern(Fee, "$ #.##") + " peak season/" +  
pattern(Fee*0.6, "$ #.##") + " off peak" }
```

Here's the finished result in Data Mode.



**Entrance Fee: \$ 10.00 peak season/\$ 6.00 off peak**

The `pattern()` function gives you total control over the format of the number. Use the `str()` function if you are content to let Panorama decide what format to use.



```
{ "There are " + str(info("records")) + " national parks listed in this database." }
```

In this case Panorama chose a simple integer format.



**There are 21 national parks listed in this database.**

By the way, in case you haven't guessed, the `info("records")` function calculates the total number of records in the database.

## Displaying Dates

To display a date in a field or variable you must convert that date to text with the `datepattern()` function (see “[Converting Between Dates and Text](#)” on page 545 for all the gory details). Here’s a simple example that prints the current date and time on the top of each page of a report.

*use datepattern() function to convert date to text*

*use today() function to calculate current day*

*this pattern specifies the date format*

Printed on {datepattern(today(),"Month ddnth, yyyy")} at  
{timepattern(now(),"hh:mm am/pm")}

Park	Phone	Web Site
Park	Phone	URL

*use now() function to calculate current time*

*use timepattern() function to convert time to text*

*this pattern specifies the time format*

When this report is printed the date and time will appear at the top of the page, like this.

Park	Phone	Web Site
Assateague Island National	(410) 641-1441	<a href="http://www.nps.gov/asis/">http://www.nps.gov/asis/</a>
Bryce Canyon National Park	(435) 834-5322	<a href="http://www.nps.gov/brcal/">http://www.nps.gov/brcal/</a>
Cape Hatteras National Seashore	(252) 473-2111	<a href="http://www.nps.gov/caha/">http://www.nps.gov/caha/</a>
Cumberland Island National	(912) 882-4336	<a href="http://www.nps.gov/cuis/">http://www.nps.gov/cuis/</a>
Death Valley National Park	(760) 786-2331	<a href="http://www.nps.gov/deval/">http://www.nps.gov/deval/</a>
Denali National Park	(907) 683-2294	<a href="http://www.nps.gov/dena/">http://www.nps.gov/dena/</a>
Everglades National Park	(305) 242-7700	<a href="http://www.nps.gov/ever/">http://www.nps.gov/ever/</a>
Fire Island National Seashore	(631) 289-4810	<a href="http://www.nps.gov/fiis/">http://www.nps.gov/fiis/</a>
Gettysburg National Military Park	(717) 334-1123	<a href="http://www.nps.gov/gett/">http://www.nps.gov/gett/</a>

This example was created with an auto-wrap text object and two embedded formulas. You can create the same effect with a Text Display SuperObject, but in that case you must use a single formula like this.

```
"Printed on "+datepattern(today(),"Month ddnth, yyyy")+
" at "+timepattern(now(),"hh:mm am/pm")
```

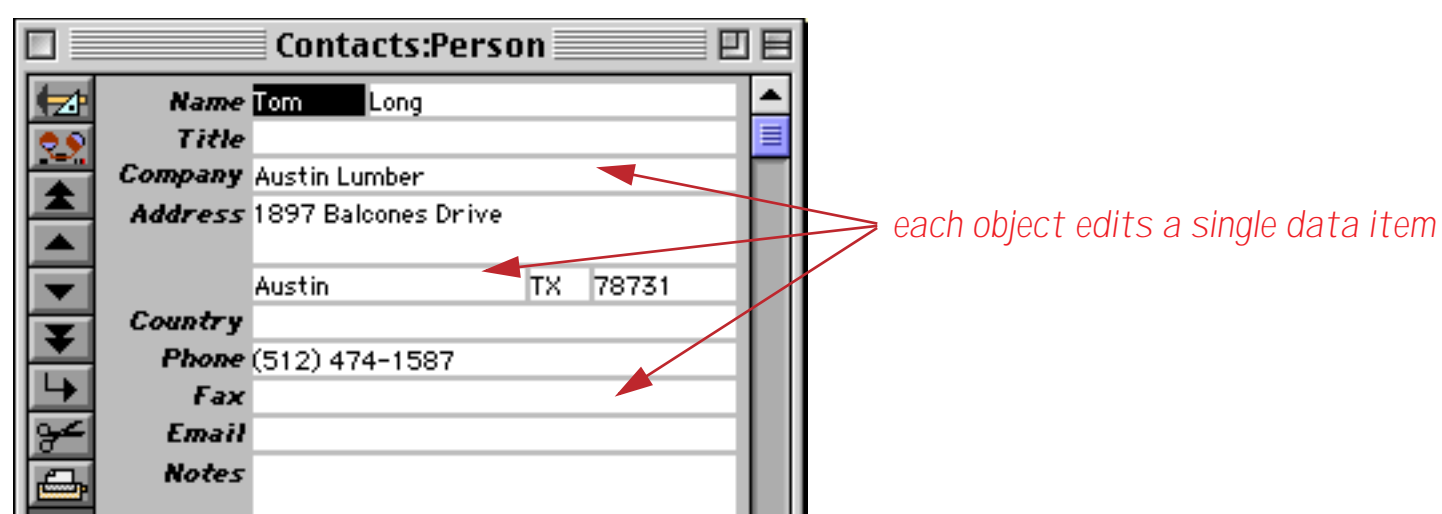
The end result is the same either way. (However, with the Text Display SuperObject you would have the option to center the text vertically or to scale the text automatically. See "[Text Display Options](#)" on page 294.)

## Editing Text

Most data entry and editing is done with the keyboard. The rest of this chapter shows how to set up a form objects for editing text.

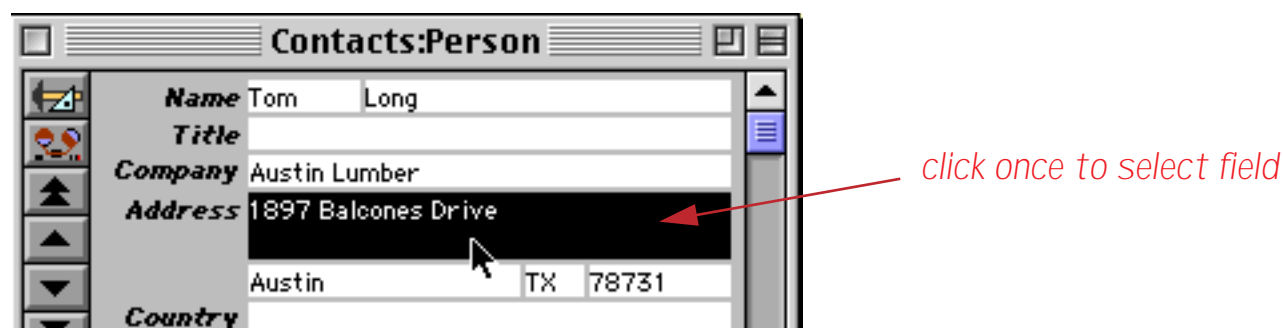
### Types of Data Editing Objects

In a form data entry is done through objects (just like everything else in a form!). Each object allows a specific item of data to be edited (for example a person's first name or a phone number). A collection of data editing objects is assembled to create a complete data entry form.



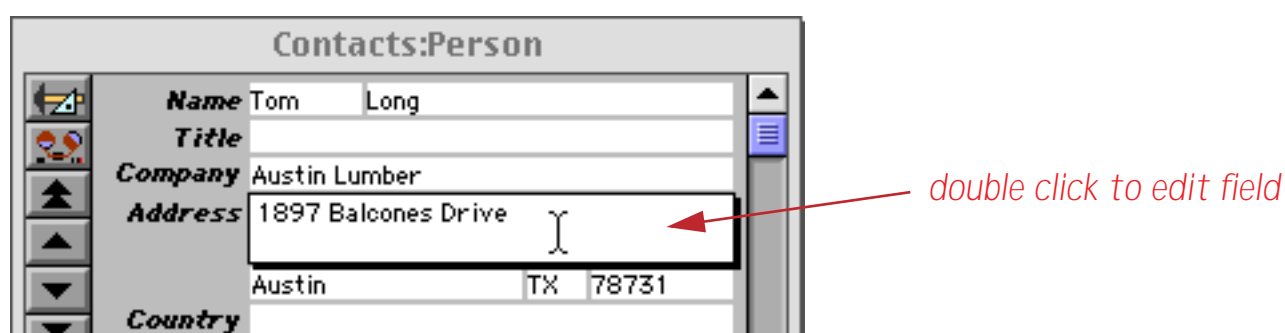
Panorama has two primary types of objects for editing text: **data cell objects** (shown above) and **Text Editor SuperObjects**. You can mix these two types on a single form, but usually you'll want to choose one type per form and stick with it.

**Data cell objects** are the "classic" way to edit data in a Panorama form. In early versions of Panorama (before version 3) this was the only kind of text editing object available. Data cell objects are designed to mimic the way Panorama works in the data sheet. In Data Mode, clicking once on a data cell object selects the field, but does not open the field for editing.

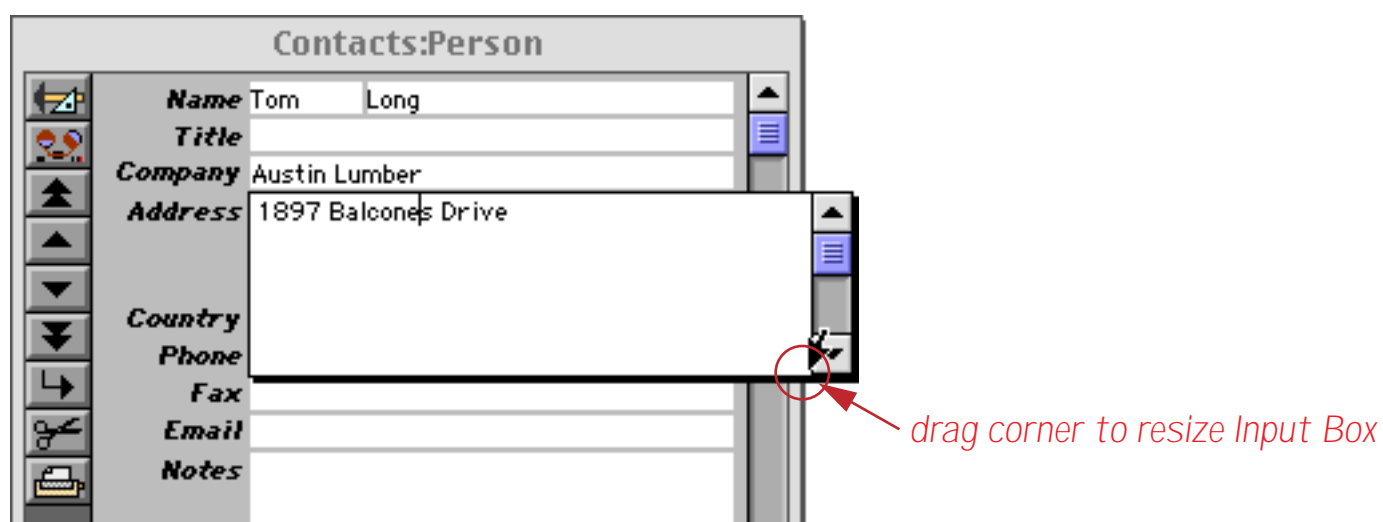




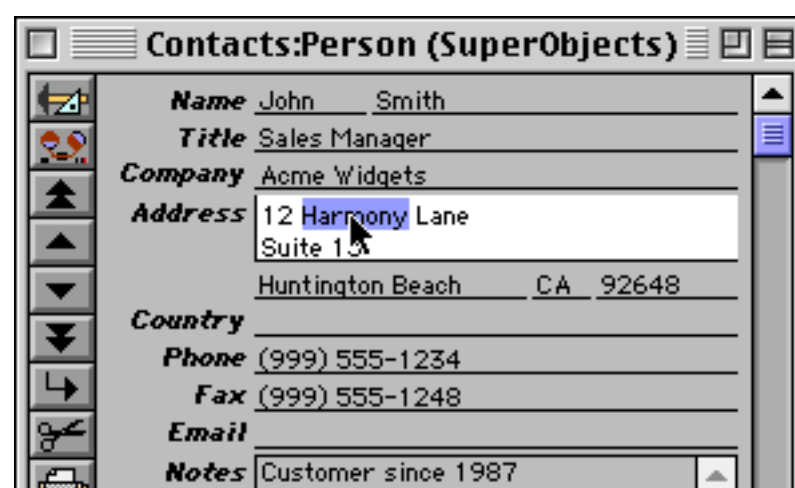
Clicking twice on the object opens the pop-up data editing Input Box you are familiar with from the data sheet (see “[The Input Box](#)” on page 132).



Just as in the data sheet, the data cell object’s Input Box can be expanded by dragging on the lower right hand corner (see “[Expanding the Input Box](#)” on page 132).



As an alternative to data cells, a form may be designed with **Text Editor SuperObjects**. Text Editor SuperObjects allow you to edit text right in the form window—no double click is required. You can simply click or drag on the text to begin editing. Press **Enter** when you are finished. The illustration below shows the effect of double clicking on the word **Harmony**. As you can see, instead of opening an Input Box this selects the word for editing.



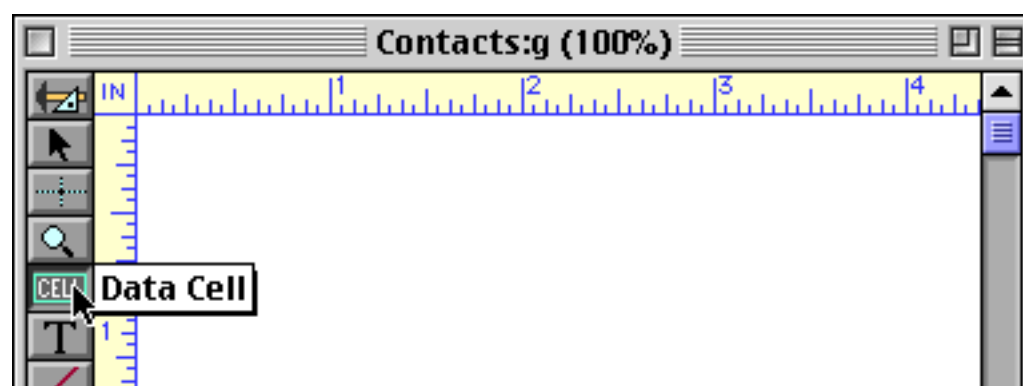
Since the Text Editor SuperObject doesn’t use an Input Box, you cannot expand the size of the editing area “on-the-fly” the same way you can with data cells. The editing area must be defined in advance. On the other hand, the Text Editor SuperObject doesn’t require the extra double click, and works more like other standard applications you may be used to.

The table below summarizes the differences between data cell objects and Text Editor SuperObjects. For many applications, either type will work all depending on your personal preferences. Some advanced features (for example editing variables, see next section) do require the use of Text Editor SuperObjects.

Feature	Data Cell	Text Editor SuperObject
Operation	Edit in pop-up Input Box (similar to data sheet)	Edit directly in form window
Expandable Editing Area?	Yes	No
Double Click before Editing?	Yes	No
Edit Fields?	Yes	Yes
Edit Variables?	No	Yes
Optional Borders?	No	Yes
Custom Object Pattern?	Yes	No

### Working with Data Cell Objects

Data cells are created with the **Data Cell** tool. To create a data cell, start by selecting this tool.



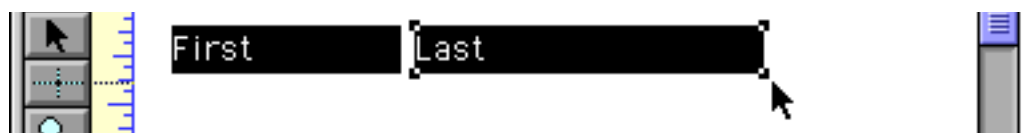
Next drag the mouse across the surface of the form. It's just like creating a rectangle.



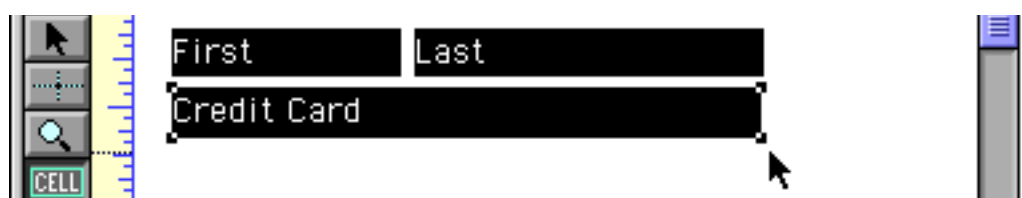
When you release the mouse, Panorama automatically assigns the first field from the database to the new data cell.



As you create additional cells, each cell is automatically assigned to the next field (using the same order that the fields appear in the data sheet). In this case the second field in the database is named **Last**.



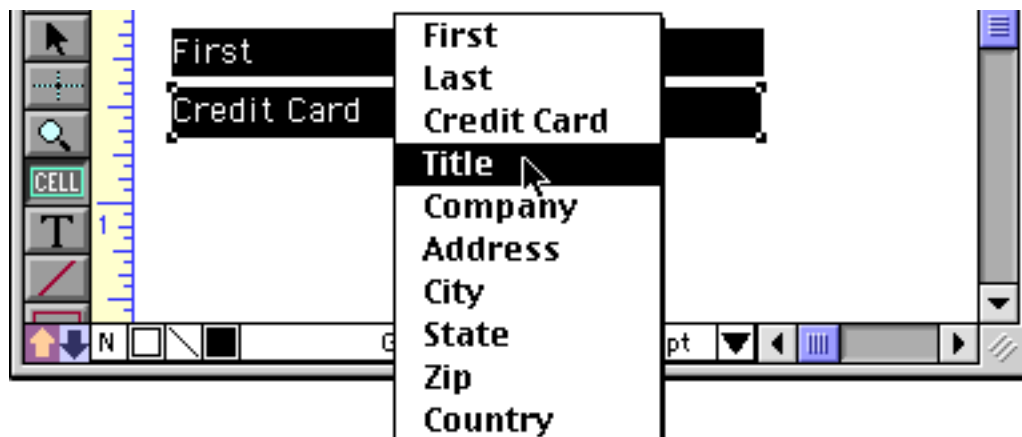
You can continue to add cells to the form. The next field in this database is **Credit Card**.



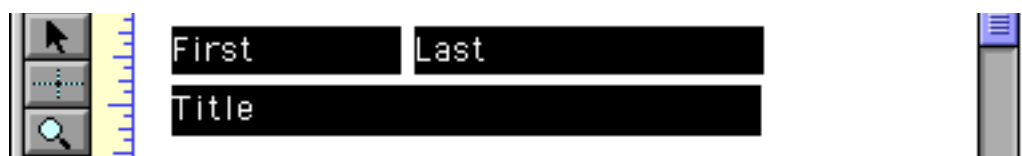
To change the field assigned to a data cell, move the mouse over the cell. The mouse arrow will change to a mini-menu icon.



When you see the mini-menu icon, press the mouse to activate a pop-up menu showing all the fields in the database.

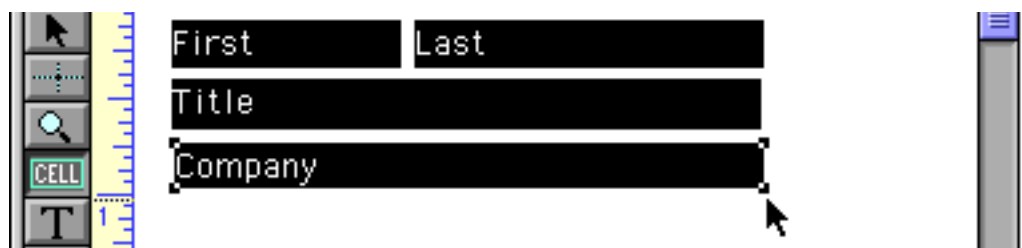


Select the field you want assigned to this data cell and release the mouse.

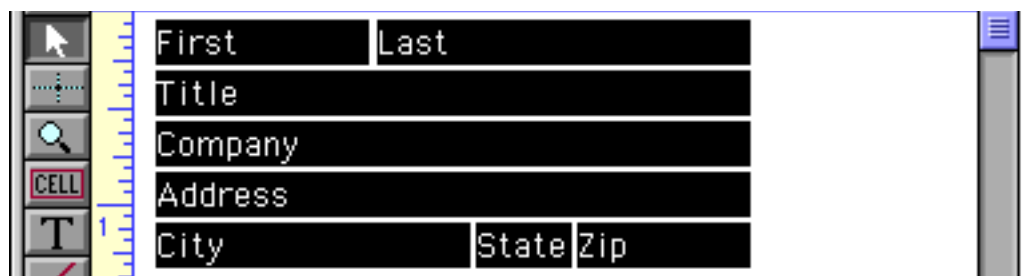


You can use this technique to change the field assigned to any data cell at any time. Remember, however, that you must have the **Data Cell** tool selected. You cannot change the field assignment when any other tool (including the **Pointer**) is selected.

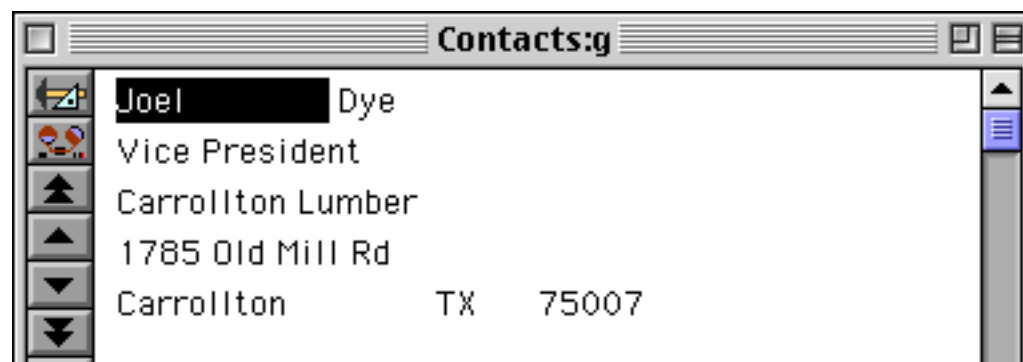
At this point we can continue making additional data cells. The next data cell will be assigned the next field after **Title** (in this case **Company**).



When you create data cells by hand like this, the result is likely to be a bit messy. After creating some more cells, we cleaned up this form using a combination of the **Dimension** dialog, the **Align** dialog, the **Spacing** dialog and nudging with the arrow keys.



These data cells are ready to use. Simply click on the **Switch To Data Access Mode** tool and you are ready to start typing.



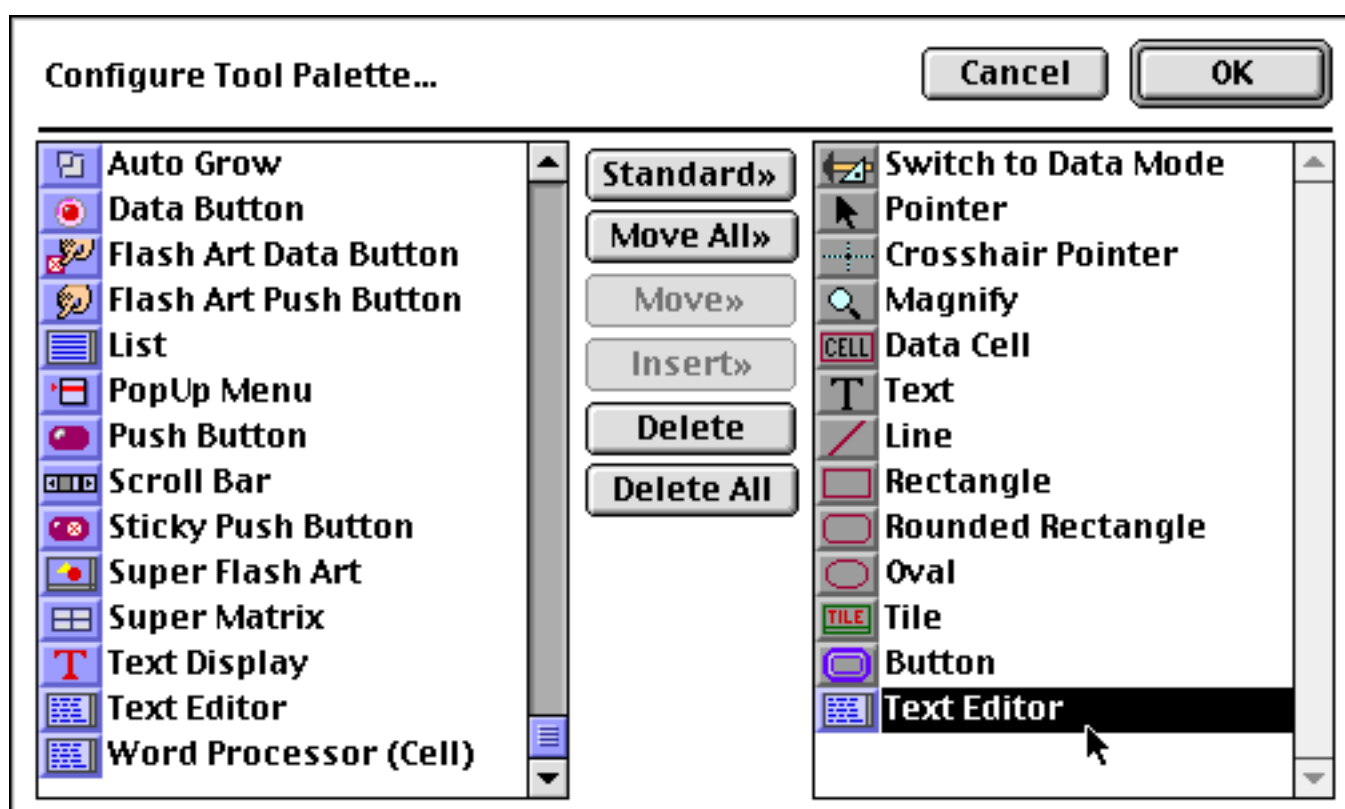
If you need to revise the data cells (or any other form object) later, click on the **Switch to Graphic Design Mode** tool.

### Text Editor SuperObject

The Text Editor SuperObject is used to edit text in a field or variable. Unlike a data cell, the Text Editor SuperObject does not use a temporary pop-up window for editing. Instead, the user simply clicks and edits the text right in the form window, just as they do with most other applications (see [“Types of Data Editing Objects”](#) on page 303). (Of course the down side of this is that the area available for editing is fixed and can't be expanded except by changing the form layout in graphic mode, or with an Auto Grow SuperObject). Another difference from data cells is that Text Editor SuperObjects can edit variables as well as fields. They can automatically draw borders and include one or two scroll bars (or none). (If you wish, you can mix Text Editor SuperObjects with standard data cells objects on the same form).

### Creating and Modifying Text Editor SuperObjects

The Text Editor SuperObject tool is not in the default tool palette, so you'll need to use the **Tool Palette** dialog to add this tool to the palette if it is not already there.



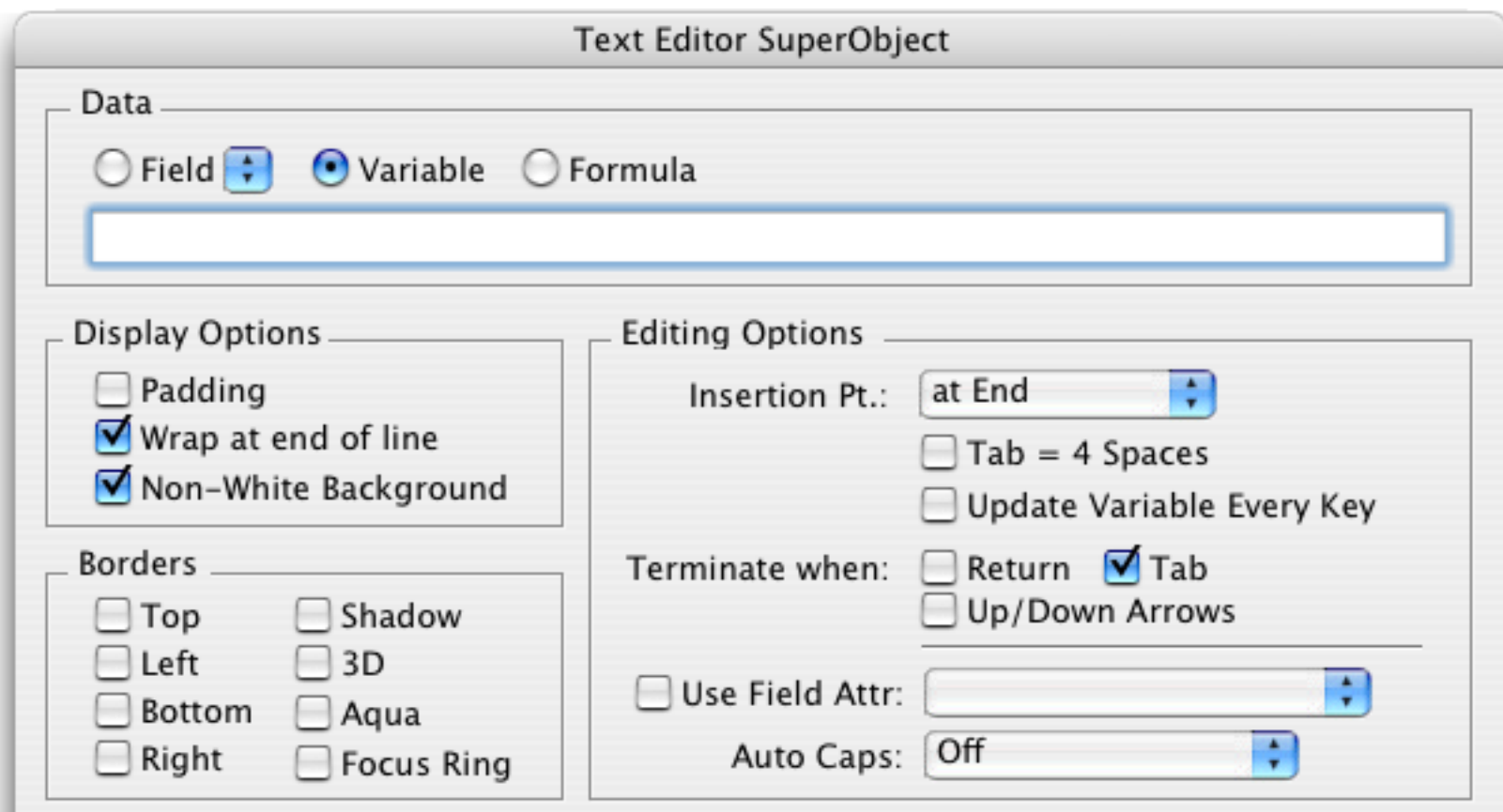
Now that the tool is added to the palette you can select it.



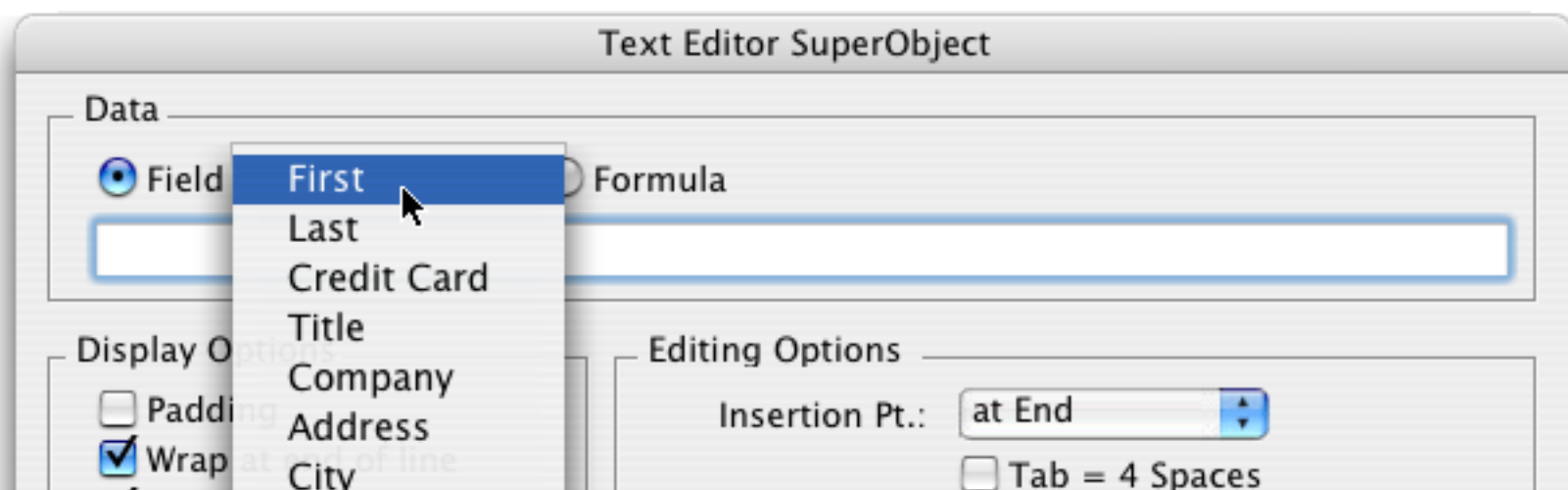
Once the tool is selected, drag the mouse across the form in the location where you want to create the Text Editor SuperObject.



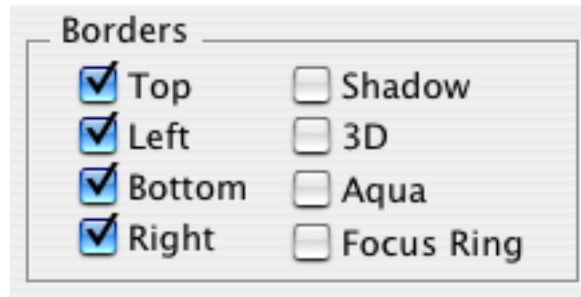
When you release the mouse, the Text Editor SuperObject configuration dialog will appear.



At a minimum you must enter a field name, variable or formula into the dialog. You can use the pop-up menu to select a field.



For this example we've also turned on the **Borders** options.



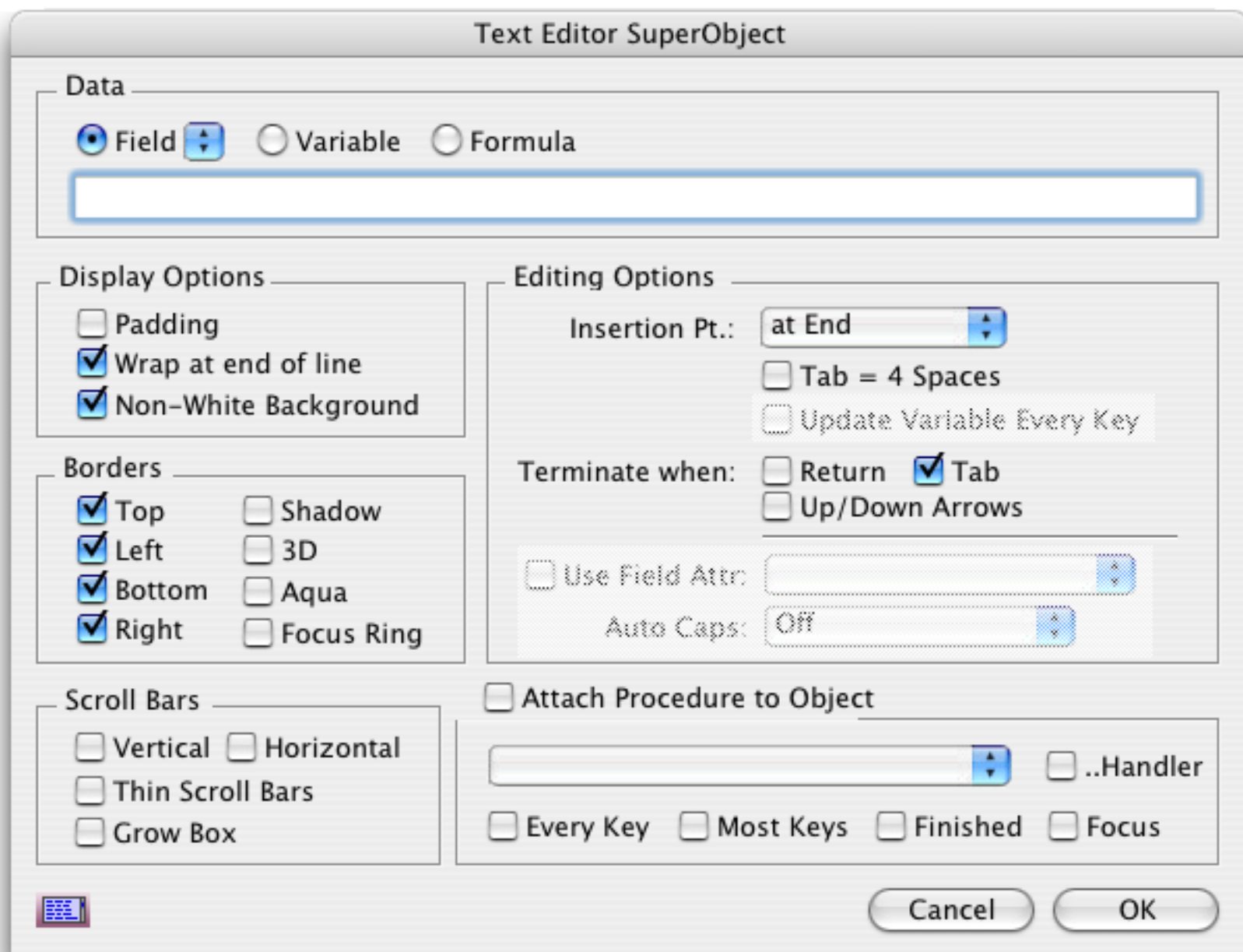
When the **OK** button is pressed the new object appears. (Notice that unlike the data cell object, the Text Editor SuperObject shows the actual data in both Graphics Mode and Data Access Mode, not just Data Access Mode.)



After it has been created you can modify the location, size, font, style and color of a Text Editor SuperObject just like any other object. To change any of the object attributes (scroll bars, border, formatting etc.) select the **Pointer** tool and double click on the object. The configuration dialog will appear again. Make your changes and press the **OK** button.

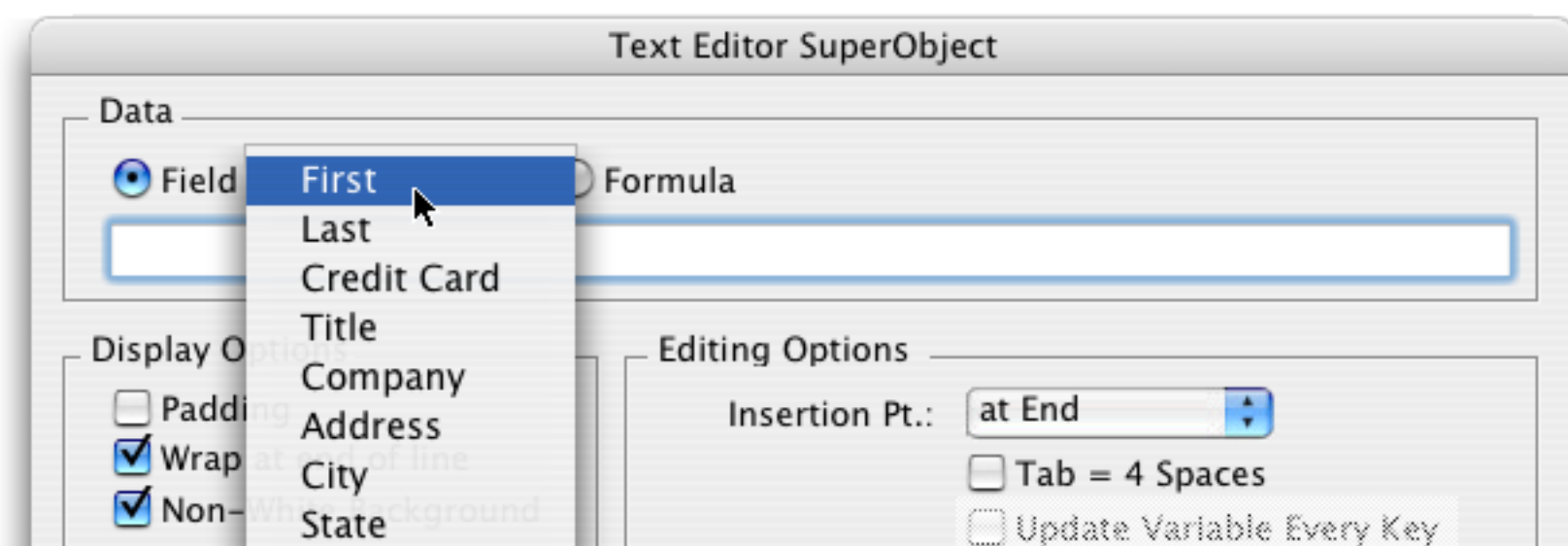
#### Text Editor Options

The SuperObject Text Editor Properties dialog is divided into several sections.

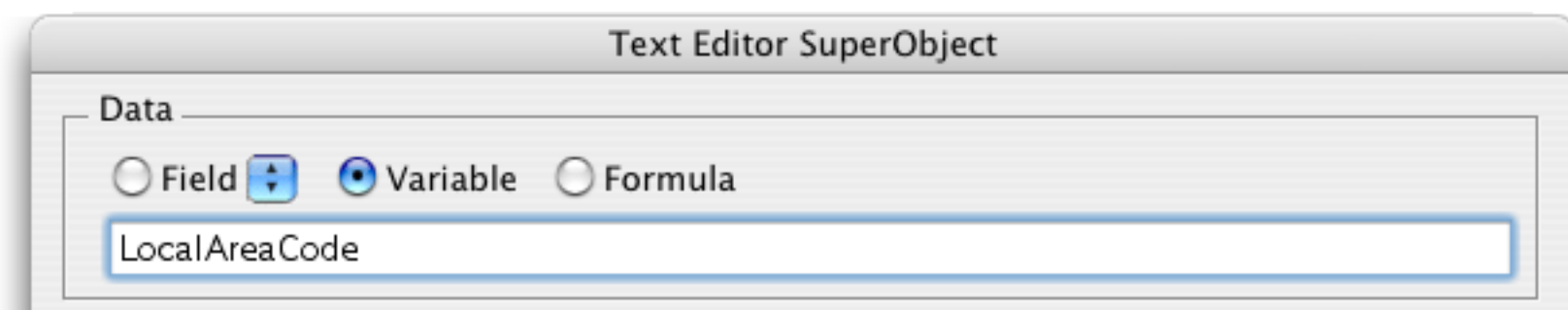




**Data:** Each Text Editor SuperObject edits a single data item, which may be a database field, a variable, or a formula. To edit a field, type in the name of the field or select the field name from the pop-up menu.



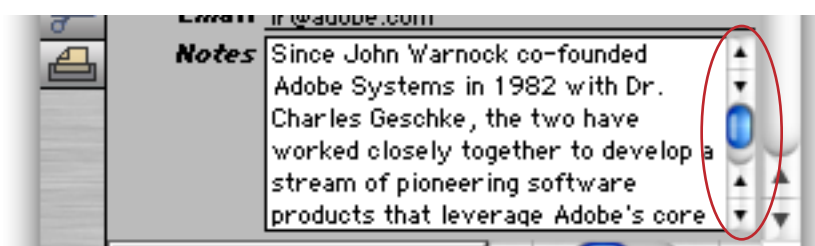
A **variable** is a place to store information independently of any database. The primary use for variables is as temporary storage for procedures (see “[Variables](#)” on page 521 and “[Variables](#)” on page 597). A variable can also be created and edited by a Text Editor SuperObject. Simply click on the **Variable** option and type in the name of the variable.



If you specify a variable that has not already been created with a procedure, Panorama will automatically create a variable with this name the first time the form is displayed. Once it has been created this variable can be used in formulas and procedures, just like any other variable.

The final data option is **Formula**. If you select this item the Text Editor SuperObject is not editing a real data item, but a temporary data item created “on-the-fly” using a formula. To learn more about this option see Chapter 15 of the **Panorama Handbook**.

**Scroll Bars:** This section controls what scroll bars (if any) will be available when editing this text. The **Thin Scroll Bar** option makes the scroll bar (or bars) 11 pixels wide instead of 16.



With the **Vertical** and **Grow Box** options turned on, Panorama leaves an empty spot for a grow box in the lower right hand corner.

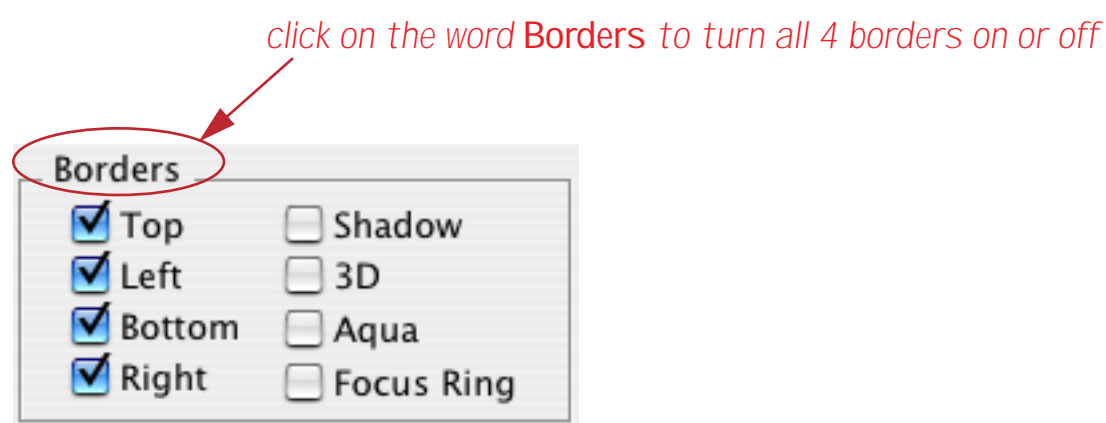
**Padding:** When this option is checked extra padding appears around the top, left and right sides of the text. It's usually a good idea to check this option.

**Wrap at End of Line:** If text being edited is too long to fit on a single line, it will usually "wrap" around to the next line. However, if the **Wrap at End of Line** checkbox is turned off, the text will not wrap. Instead, the text will continue off the right edge. If the horizontal scroll bar is enabled, you can scroll over to see the rest of the text.

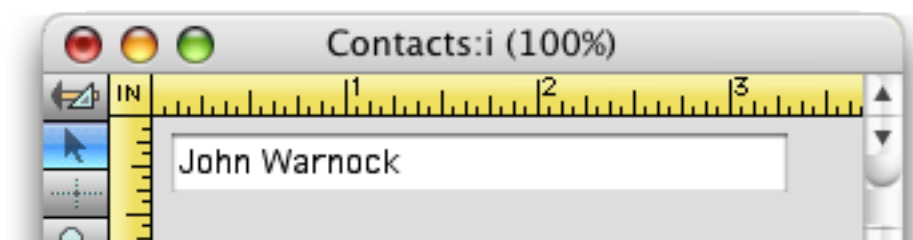
**Non-White Background:** We recommend that you use this option if the Text Editor SuperObject is placed over a color (non-white) background. If this option is turned on, Panorama will temporarily display a white background behind the text while it is being edited. If you don't use this option, you'll find that portions of the background will turn white as you edit anyway. The result is ugly and possibly confusing.

**Terminate When:** You can always press the **Enter** key when you have finished editing a data item. Depending on the options set in this dialog other keys may also cause editing of this cell to finish, including the **Return**, **Tab**, **Up Arrow**, or **Down Arrow** keys. (Note: If you want to be able to tab from this item to the next, be sure to select the **Tab** key as one of the keys that causes termination. On the other hand, if you want to be able to use the **Tab** key inside this field or variable, **Tab** should not be checked.)

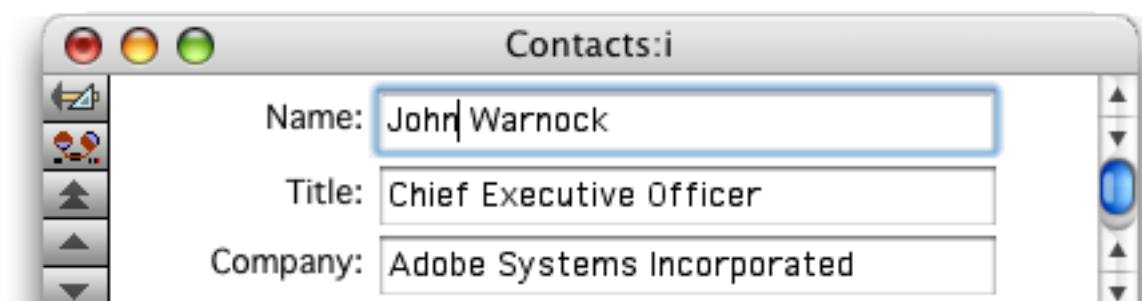
**Borders:** The options in this section control the borders that are displayed around the text (if any). You may separately control the top, bottom, left, and right borders or click on the word **Borders** to turn all four on or off at once.



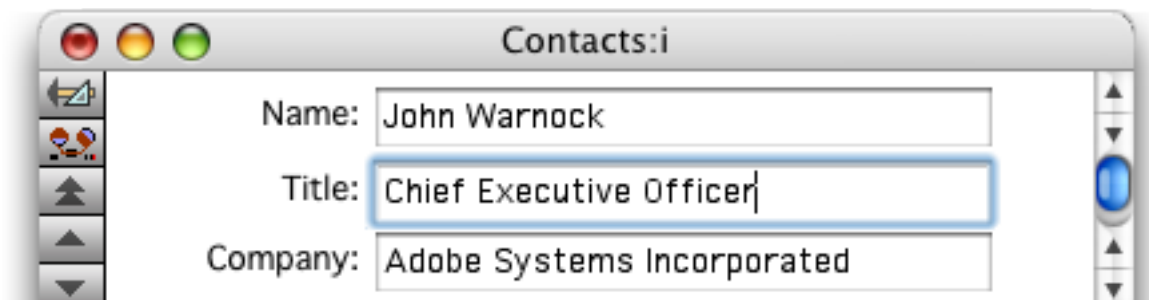
The **Shadow** option makes a drop shadow appear. The **3D** border effect works best with a light gray background. The **Aqua** border effect displays the same soft 3D borders that are used by most OS X applications. This option can work with a light gray or white background.



When the **Focus Ring** option is checked, Panorama will display a blue ring around the object when it is being actively edited.



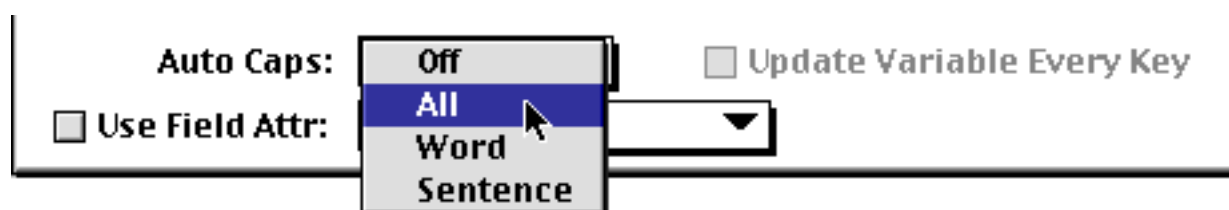
When you shift to editing a different object, the blue ring will move to this new object.



(Note: You can control the background color for the entire form with the **Form Preferences** command in the Setup menu. See “[Form Background Colors](#)” on page 282.)

**Insertion Point:** This option gives you the choice of what text should be selected when you tab into this Text Editor SuperObject. (Of course when you click into a Text Editor SuperObject, the insertion point goes where you click.) The three options are: **At End**, **At Start**, and **ALL** (which selects all the text).

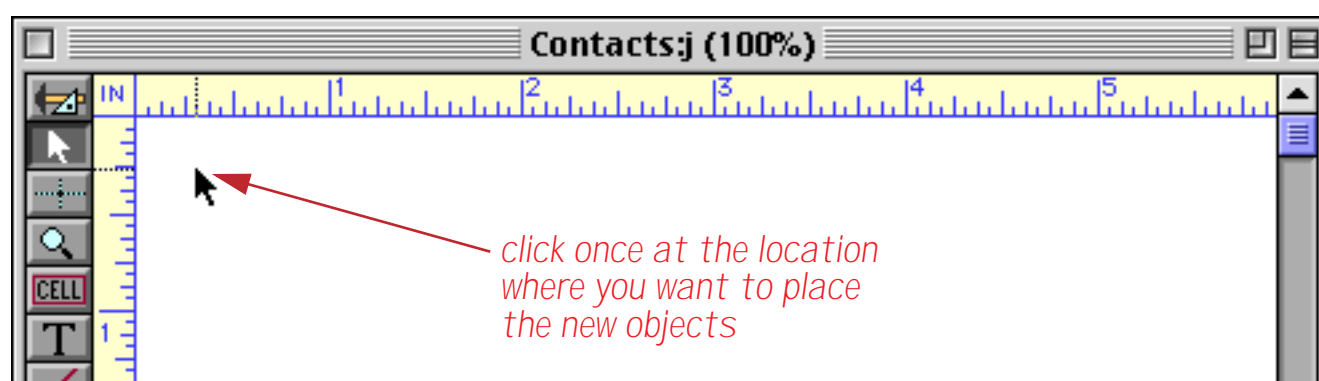
**AutoCaps:** This option only appears if the Text Editor is associated with a variable or formula.



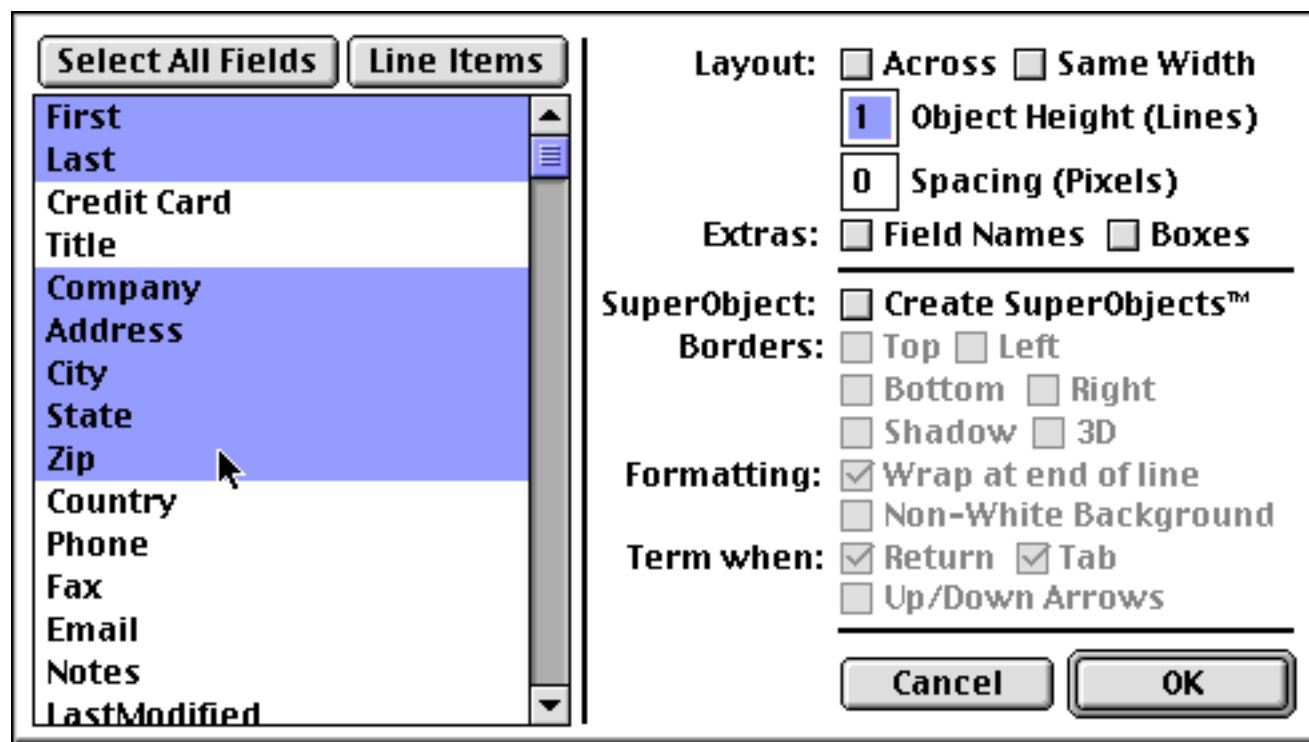
Use the pop-up menu to control automatic capitalization of the text as it is entered. You can force the text to all upper case (**ABC**), word caps (**Abc**), or capitalization of the first letter of each sentence. If the Text Editor SuperObject is associated with a field, the Text Editor SuperObject will automatically use the Auto Caps setting for that field (set in the design sheet or the Field Preferences dialog).

### Automatically Creating Rows or Columns of Data Cells or Text Editor SuperObjects

Earlier in this chapter you learned how to create data cell objects and Text Editor SuperObjects one at a time. The **Auto Cell Layout** command (in the Arrange menu) can automatically generate an entire row or column of these objects. To use this command, first pick the font and size you want to use from the Graphic Control Strip or the Text sub-menus. Once the font is set, make sure the **Pointer** tool is selected and click on the spot where you want to place the new objects.



Then open the **Auto Cell Layout** dialog using the **Arrange** menu. This dialog allows you to choose the fields and arrangement you want.



The box on the left of the dialog lists all the fields in the database. Select each field you want to place in the form. There are several ways to select fields. You can select individual fields by clicking on them. You can select several fields at once by dragging the mouse across them. You can select all the fields by pressing the **Select All Fields** button. If you change your mind, you can de-select a field by clicking on it again.

Once you have selected the fields you want to create, press the **OK** button to place the fields into the form.



The new objects appear just below and to the right of the spot you originally clicked on.

#### Automatic Layout Options

The right hand side of the **Auto Cell Layout** dialog contains options for varying the arrangement of objects that are created.

The **Across** option controls the direction of the generated objects. The normal direction is down.



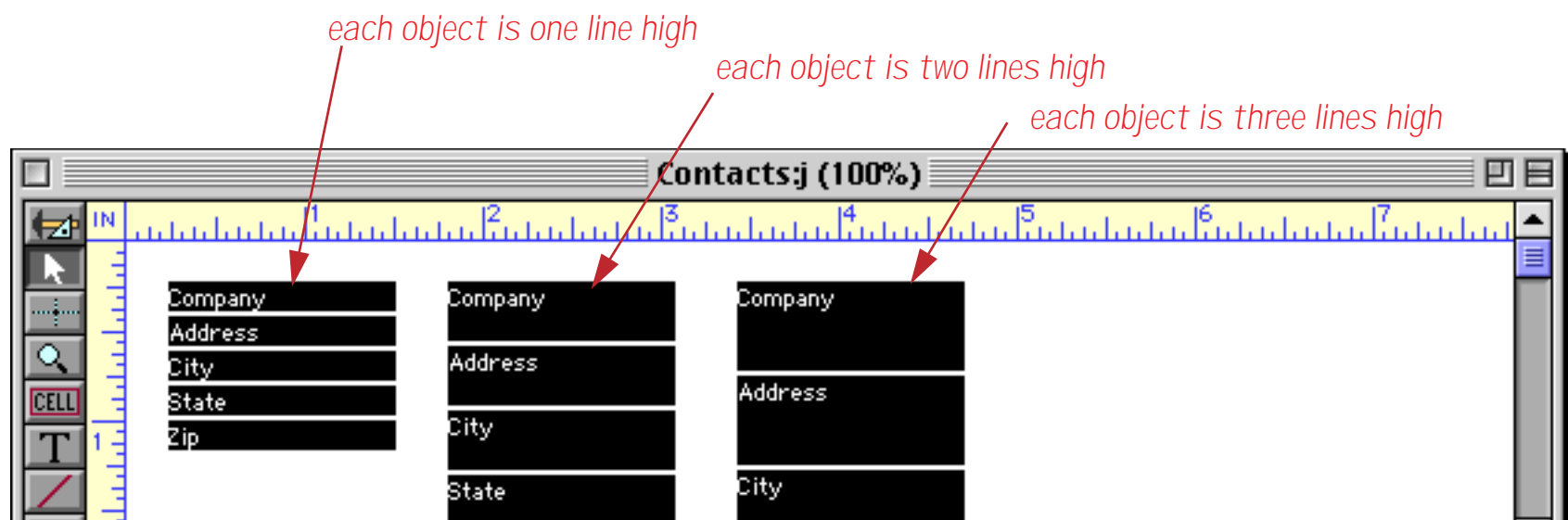
If the **Across** option is enabled the objects are generated horizontally, in a row.



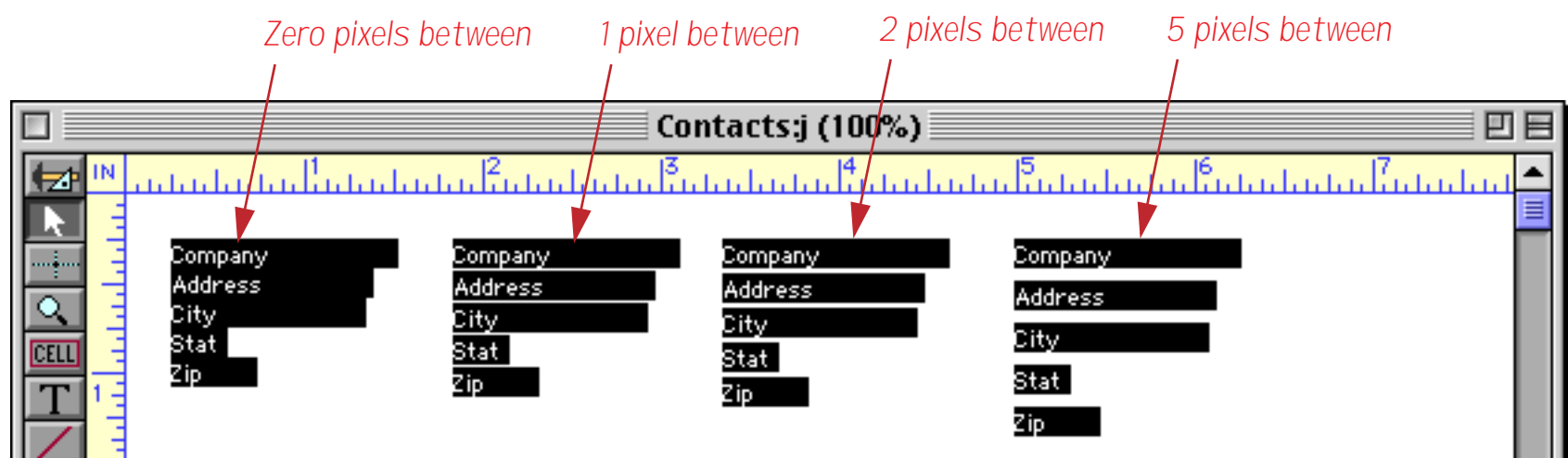
The **Same Width** option controls the width of the generated objects. If this option is off, the width of each object will be the same as the width of the corresponding column in the data sheet (see “[Changing the Width of a Field](#)” on page 104). If the **Same Width** option is enabled, all of the objects will have the same width. (Of course you can always change the width of any object after it has been generated.) This illustration shows what the end result of this option looks like both in normal mode and with the **Across** option enabled.



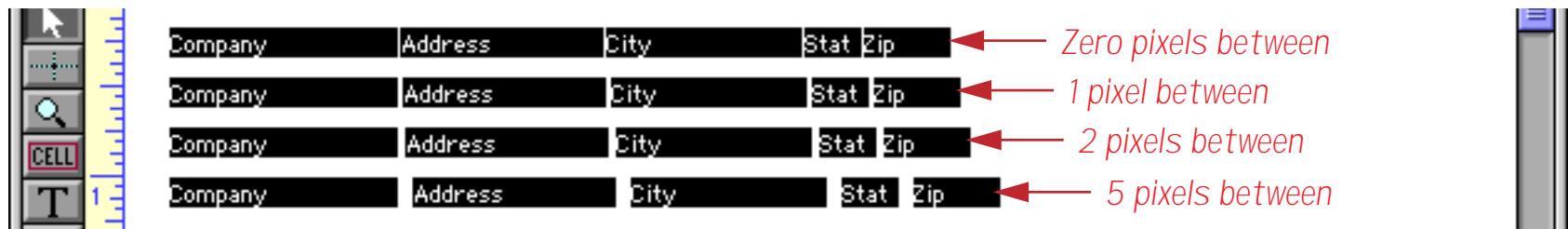
The **Lines High** option specifies the height of each object. Normally each object is one line high in the current font, but you can generate objects that are two lines high, three lines high, or more. Of course you can also change the height of any object after it has been generated.



The **Pixels Between** option specifies the spacing between adjacent data cells. (A pixel is one dot on the screen at 100% magnification, or 1/72 inch.) When you create a column of data cells, this option specifies the vertical spacing between the cells.



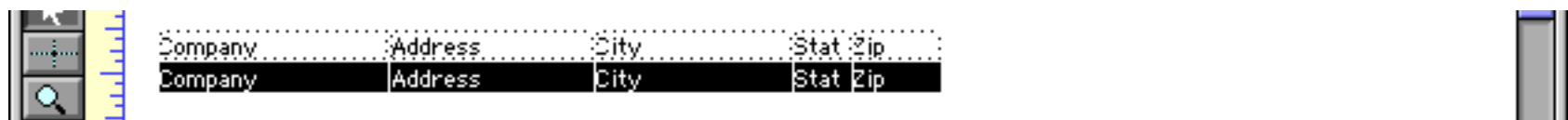
When you create a row of data cells (**Across**), this option specifies the horizontal spacing between the cells.



The **Field Names** option tells Panorama to create a field name next to each generated object. If you create a column of objects, the field names will be placed to the left.



If you create a row of data cells (**Across**), the field names will be placed above the objects, like this.



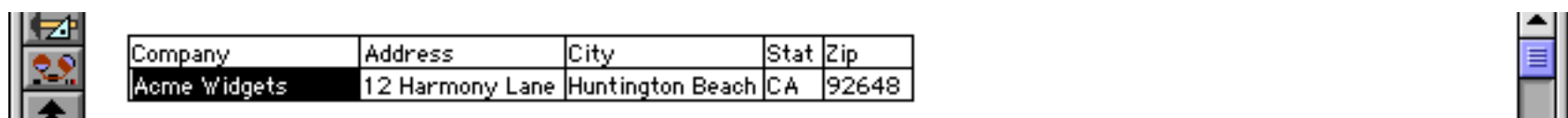
Use the **Boxes** option if you want Panorama to draw a box around each generated object. This usually makes sense only for data cells, since for Text Editor SuperObjects you can generate borders as part of the object itself. This illustration shows what the boxes look like when used with data cells.



It's a bit easier to see what the boxes look like in Data Access Mode.

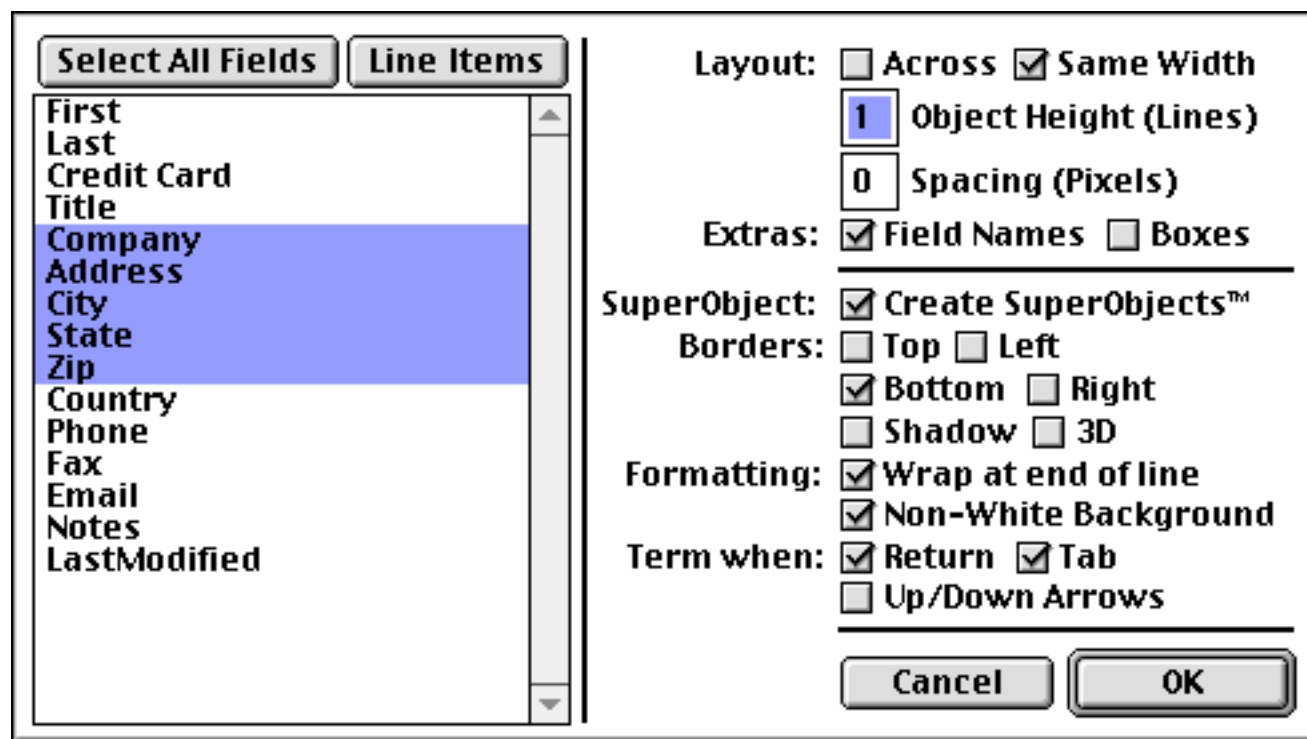


If **Field Names** option is checked Panorama will draw a box around the field names as well.

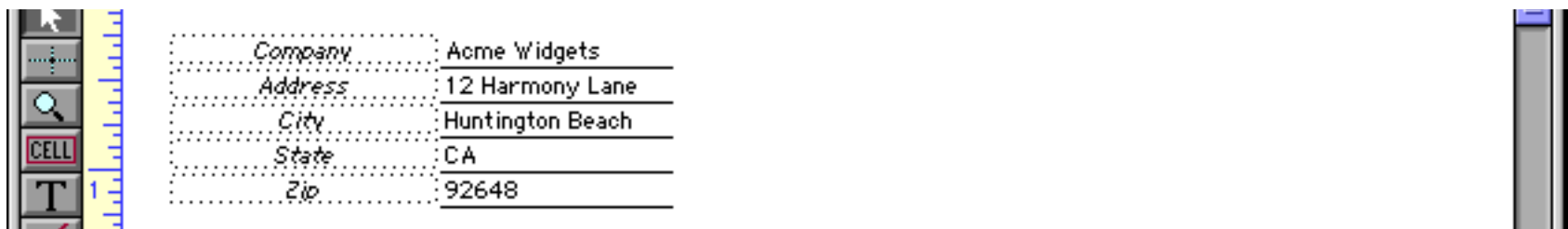




To create Text Editor SuperObjects instead of data cell, check the **Create SuperObjects** option. Once this option is turned on, you can select the options for the Text Editor SuperObjects you want to create — borders, formatting, etc. These options are the same as the options in the Text Editor SuperObject configuration dialog (see “[Text Editor Options](#)” on page 309).



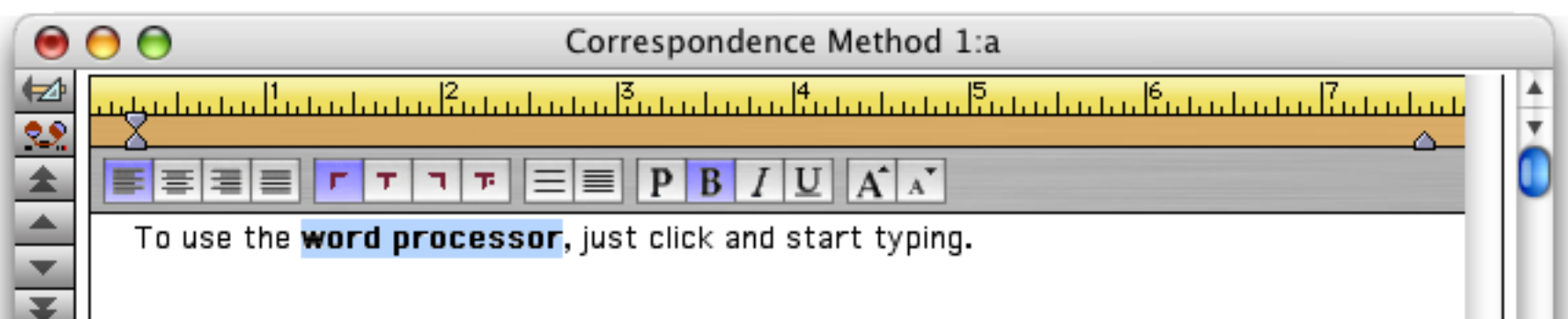
Pressing **OK** generates a column of Text Editor SuperObjects (instead of data cells).



Clicking on the word **Borders** will turn all four borders on or off. If you use the Text Editor SuperObject borders your probably will not want to turn on the **Boxes** option, which adds an additional box around each object (see above).

### Word Processor SuperObject

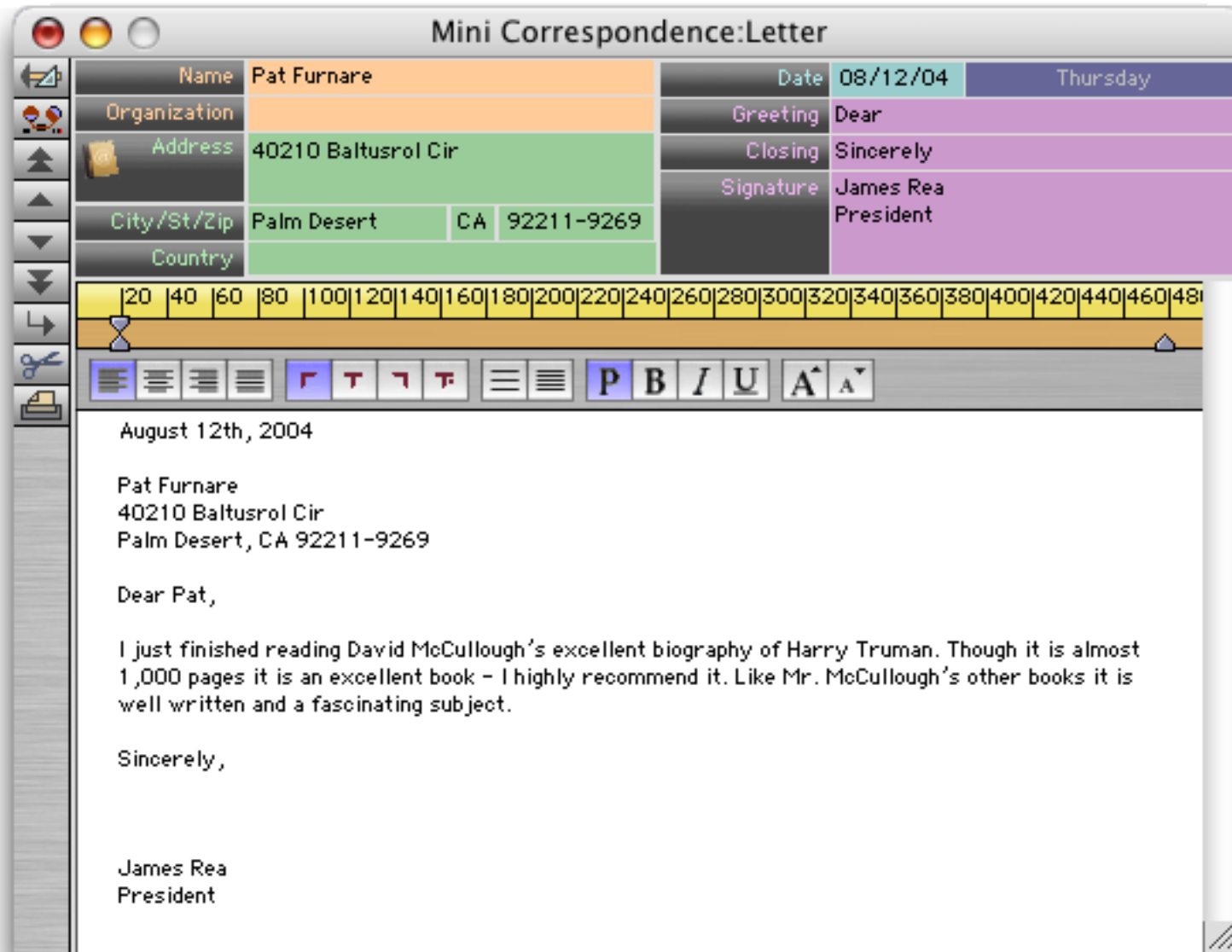
The Word Processing SuperObject™ allows you to include a complete word processor as part of a form. The word processor allows you to mix different fonts, sizes, styles (16 different styles) and colors in a single paragraph. Left, Right, and First Line margins may be set up separately for each paragraph, and you may set up left justified, right justified, center justified and decimal tabs with optional tab leaders. The merge option allows you to merge information from the database (or complete formulas) into the word processing text. The word processor SuperObject also includes most of the options available with the Text Editor SuperObject, including borders and a vertical scroll bar.



To learn more about the Word Processor SuperObject see Chapter 15 of the **Panorama Handbook**.

## Using the Mini Correspondence Wizard

Panorama includes a pre-built database for handling general word processing chores — the **Mini Correspondence** wizard. This database may be used for general correspondence (letters, memos, etc.) and to create mail merge letters that are customized and sent to a group of recipients. You can use the **Mini Correspondence** wizard as is or you can modify it or even take components of the wizard and include them in your own databases.



To learn more about this wizard see Chapter 15 of the **Panorama Handbook**.



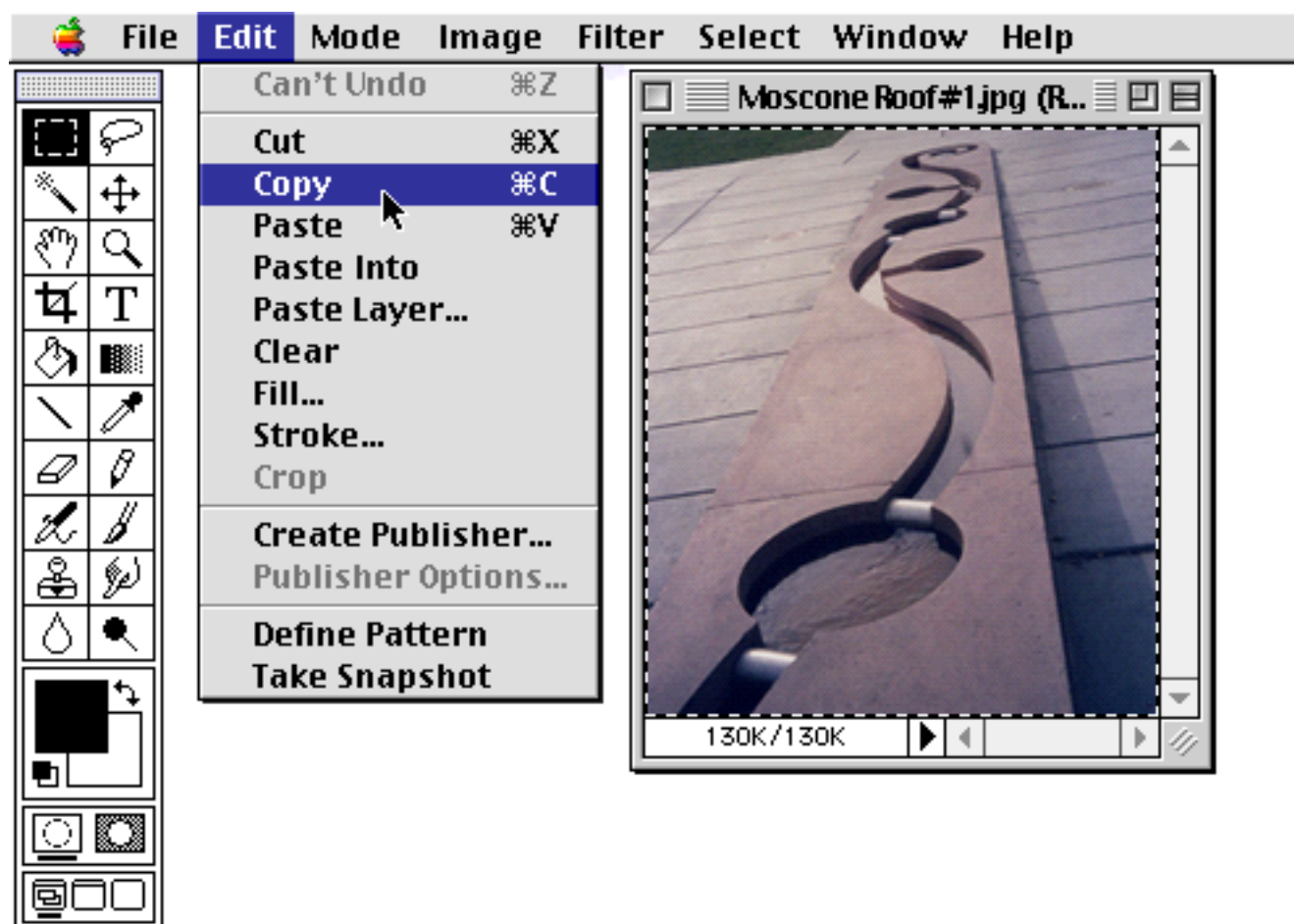
# Chapter 16: Images & Movies



A Panorama form can display images and movies from a wide variety of sources. An image may be fixed (for example a logo or background) or variable (changing from record to record - for example personnel photos or maps associated with individual records). Variable images may be included in the database or (more commonly) displayed directly from files on the disk. Images that are displayed from disk are called **Flash Art™**. Flash Art allows any image to be displayed according to its name.

## Fixed Images

Fixed images are imported into Panorama with the **Paste** command. Before you use the **Paste** command, the graphic picture you want to import must be placed on the clipboard. Most paint and drawing programs allow you to select the graphic elements you want to use and then use the **Copy** command to place the graphics on the clipboard. This illustration shows an image being transferred to the clipboard in Adobe Photoshop.



Once the image has been transferred to the clipboard, switch to Panorama. The form you want to place the image into must be open and in Graphic Display Mode. Then choose **Paste** from the Edit menu to import the image.



The image will be imported into the center of the window. Like any other object, you can move the picture by dragging it or adjust the size by dragging the handles on the corners (or with the **Dimensions** or **Scale** dialogs).



## Flash Art™

Panorama employs a technique called **Flash Art™** to display non-fixed images. Using Flash Art you can display images that are not fixed but change depending on circumstances—for example photos of each person in a personnel database, maps in a contact database or product photos in a catalog database.

When Flash Art displays an image, it does so by name. The actual image may be stored in the database itself (in the Flash Art Gallery, which you can read about in Chapter 16 of the Panorama Handbook), or in separate disk files. You set up a formula that controls what image to display. When the formula matches the name of an image, that image is displayed.

Panorama actually has several different types of objects for displaying images by name. The standard Flash Art object has been standard from the earliest versions of Panorama, and is retained for compatibility with older databases. The newer Super Flash Art object is more customizable (more alignment and scaling options, scroll bars, etc.) and should usually be used for new applications. In addition, the Flash Art Push Button and Flash Art Data Button allow you to create custom buttons from any image. Only Super Flash Art objects will be discussed in this book, for information on these other types see the **Panorama Handbook**.

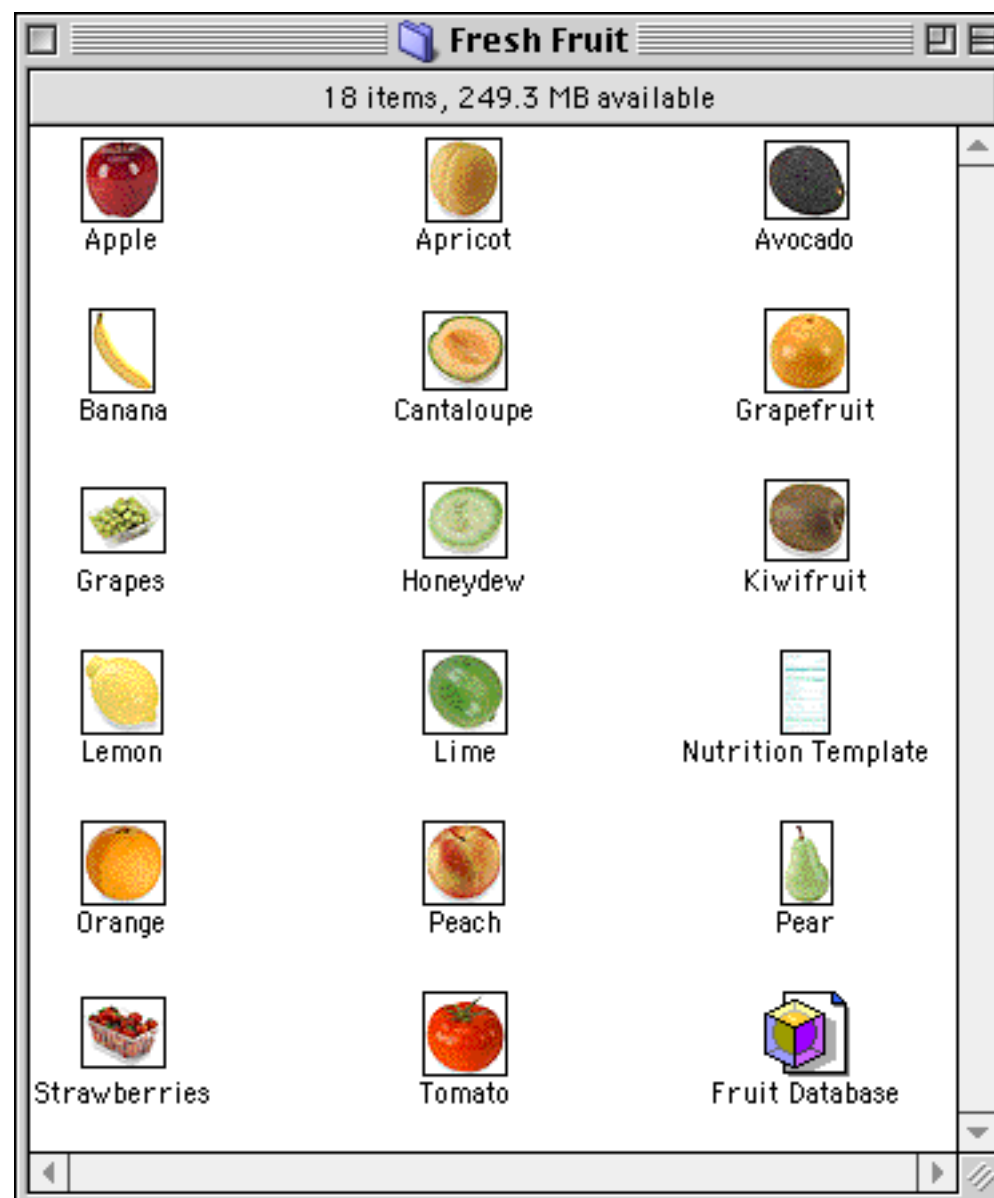


## Creating Super Flash Art Objects

To illustrate the creation of a Super Flash Art object within a form we'll use a database of nutritional information for fruits. This database contains about a dozen different fruits, including the name of the fruit and the FDA nutritional information.

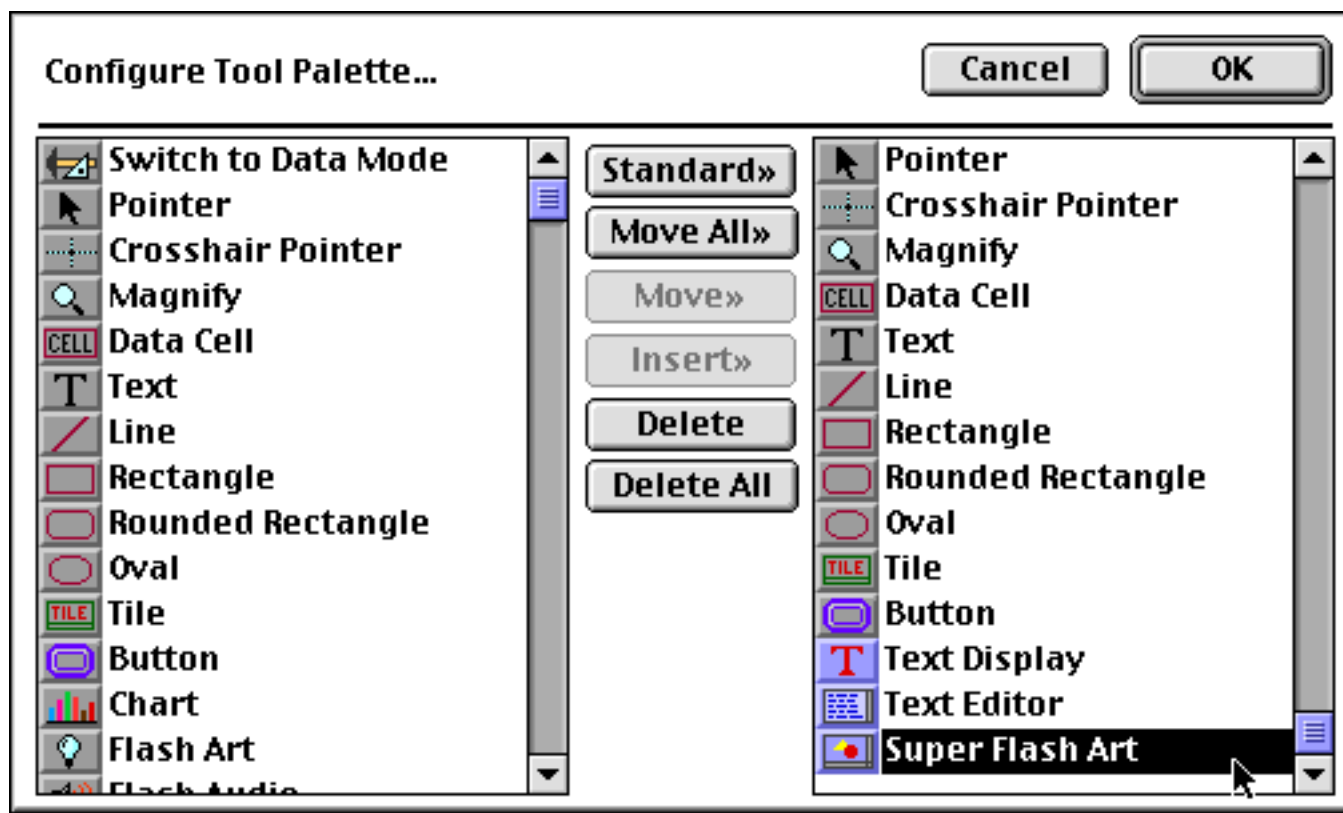
Fruit	Serving Size	Calories	Fat (	Total Fat	Saturated	Cholesterol	Sodium	Potassium	Carbc
Apple	1 med apple (154g)	80	0	0g	0g	0mg	0mg	170mg	22g
Apricot	2 apricots (114g)	60	10	1g	0g	0mg	0mg	0mg	11g
Avocado	1/5 avocado (30g)	55	45	5g	1g	0mg	0mg	170mg	3g
Banana	1 med banana (126g)	110	0	0g	0g	0mg	0mg	400mg	29g
Strawberr	8 medium berries	45	0	0g	0g	0mg	0mg	27mg	12g
Grapes	1-1/2 cups grapes	90	10	1g	0g	0mg	0mg	270mg	24g
Lemon	1 med lemon (58g)	15	0	0g	0g	0mg	0mg	0mg	5g
Lime	1 medium (67g)	20	0	0g	0g	0mg	0mg	75mg	7g
Cantaloupe	1/4 melon (134g)	50	0	0g	0g	0mg	25mg	280mg	12g
Honeydew	1/10 melon	50	0	0g	0g	0mg	35mg	310mg	13g
Orange	1 med orange (154g)	70	0	0g	0g	0mg	0mg	260mg	21g

We've also prepared a collection of photographs of fruit. Each photograph has a name, and the photo name matches the name of the fruit in the database, from [Apple](#) to [Tomato](#). These images have been saved in the PICT format, and are in the same folder as the database.

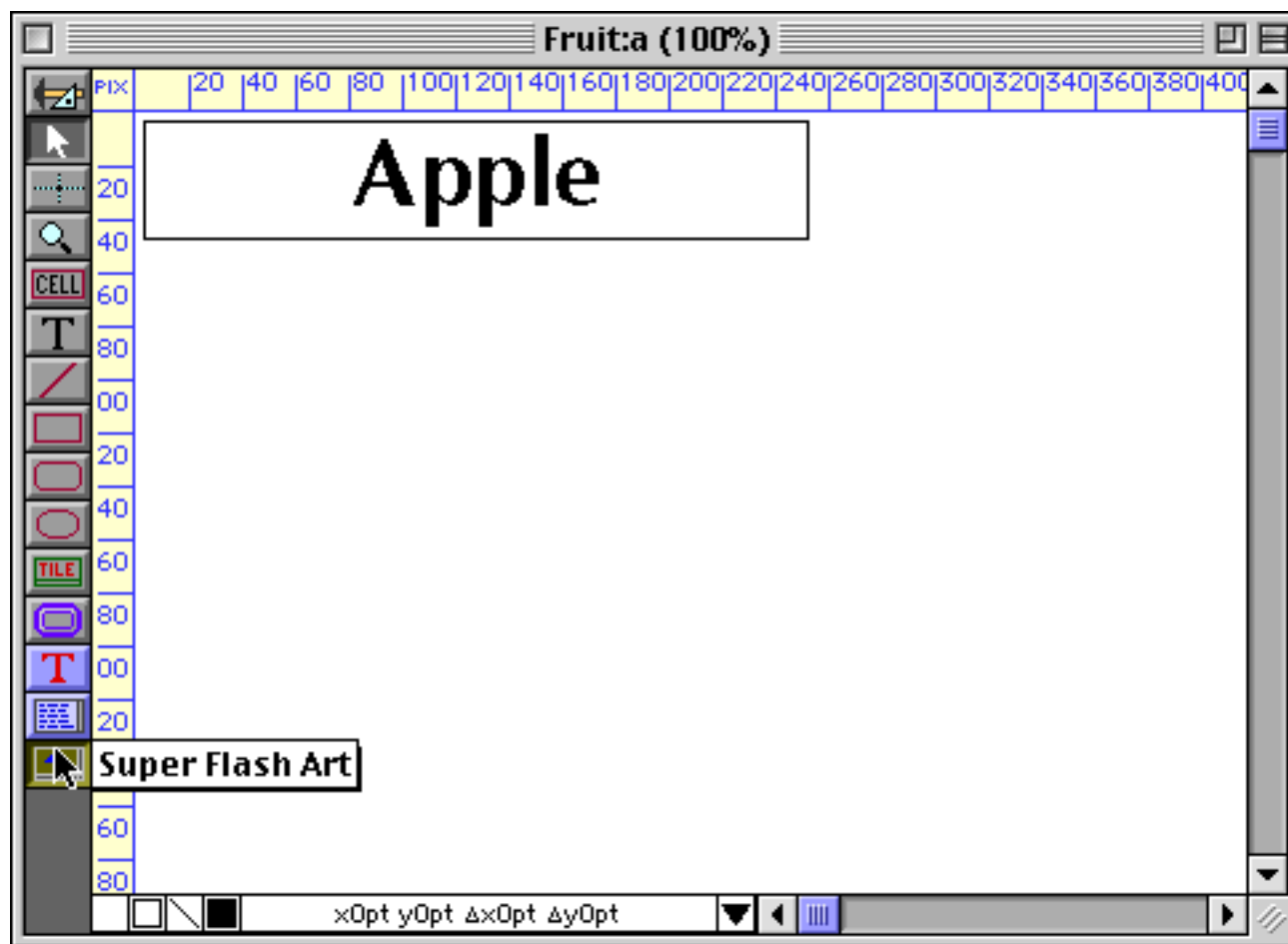




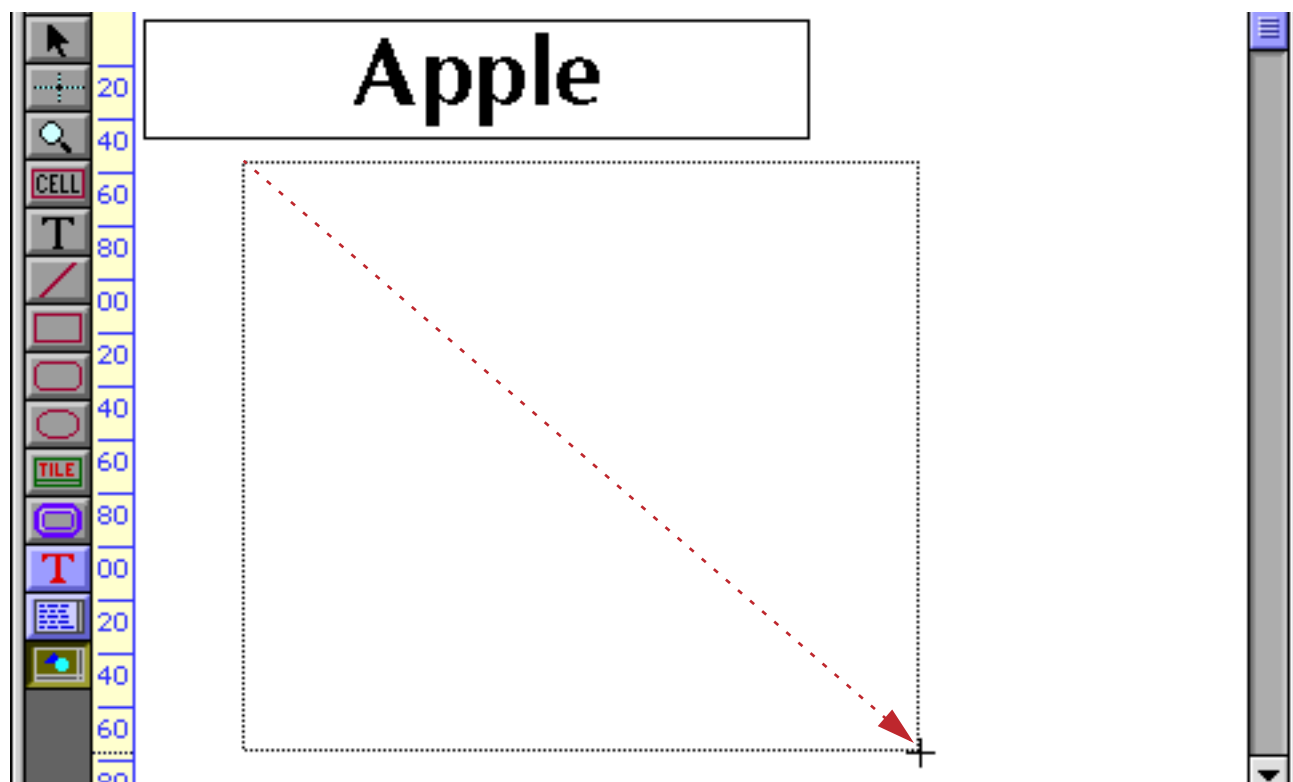
The Super Flash Art tool is not in the default tool palette, so you'll need to use the Tool Palette dialog to add this tool to the palette if it is not already there.



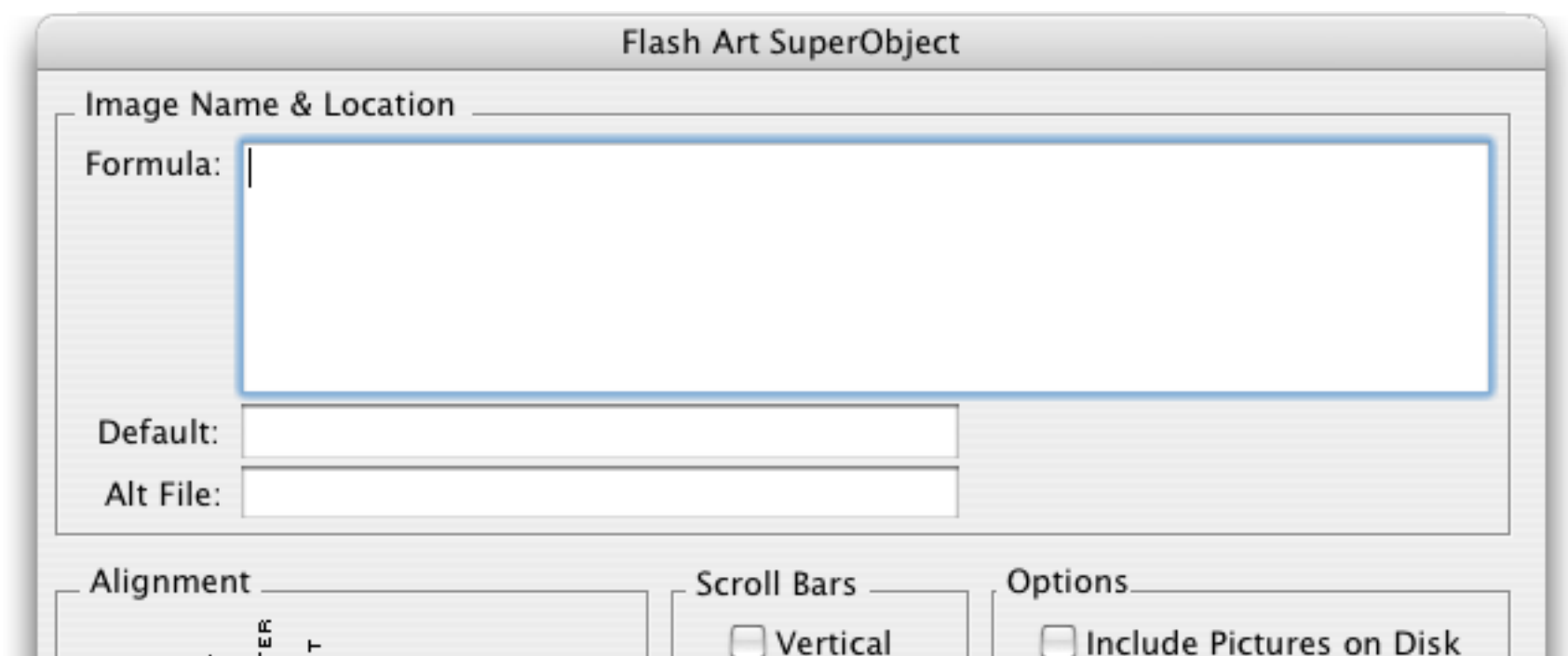
Now that the tool is added to the palette you can select it. Notice that we've set up a Text Editor SuperObject to allow the name of the fruit to be displayed and edited.



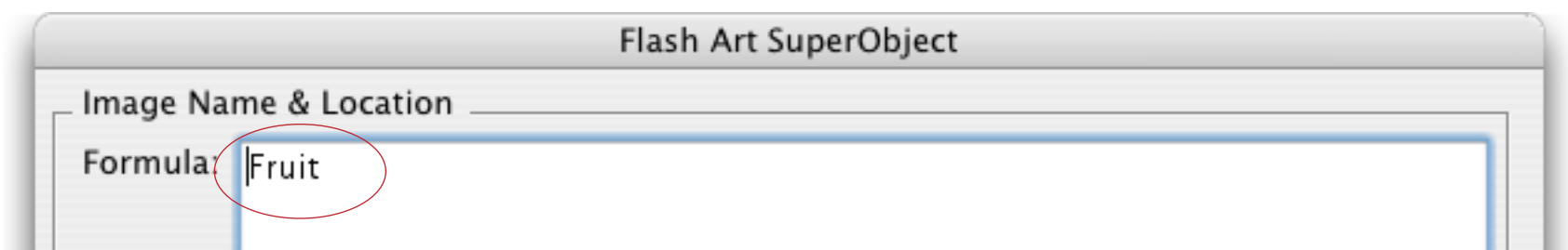
Once the tool is selected, drag the mouse across the form in the location where you want to create the Super Flash Art object.



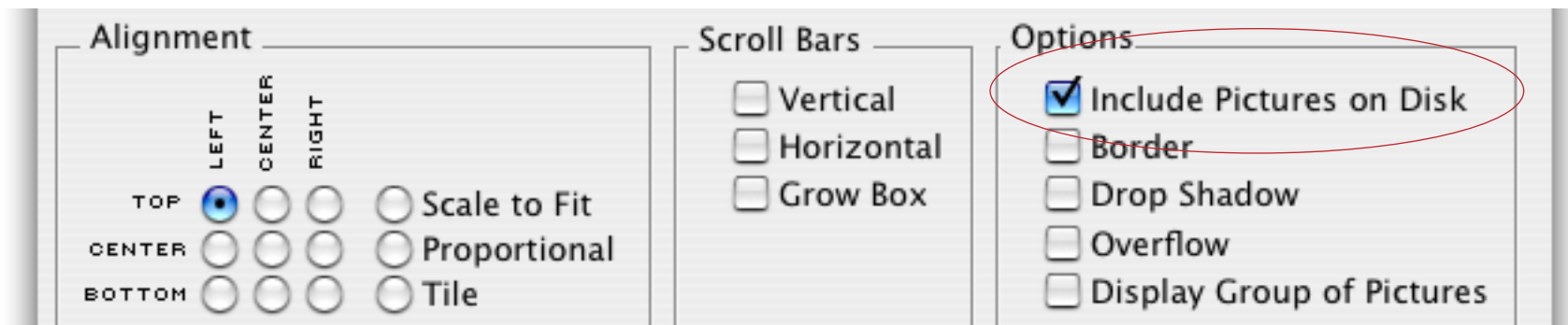
When you release the mouse, the Super Flash Art configuration dialog will appear.



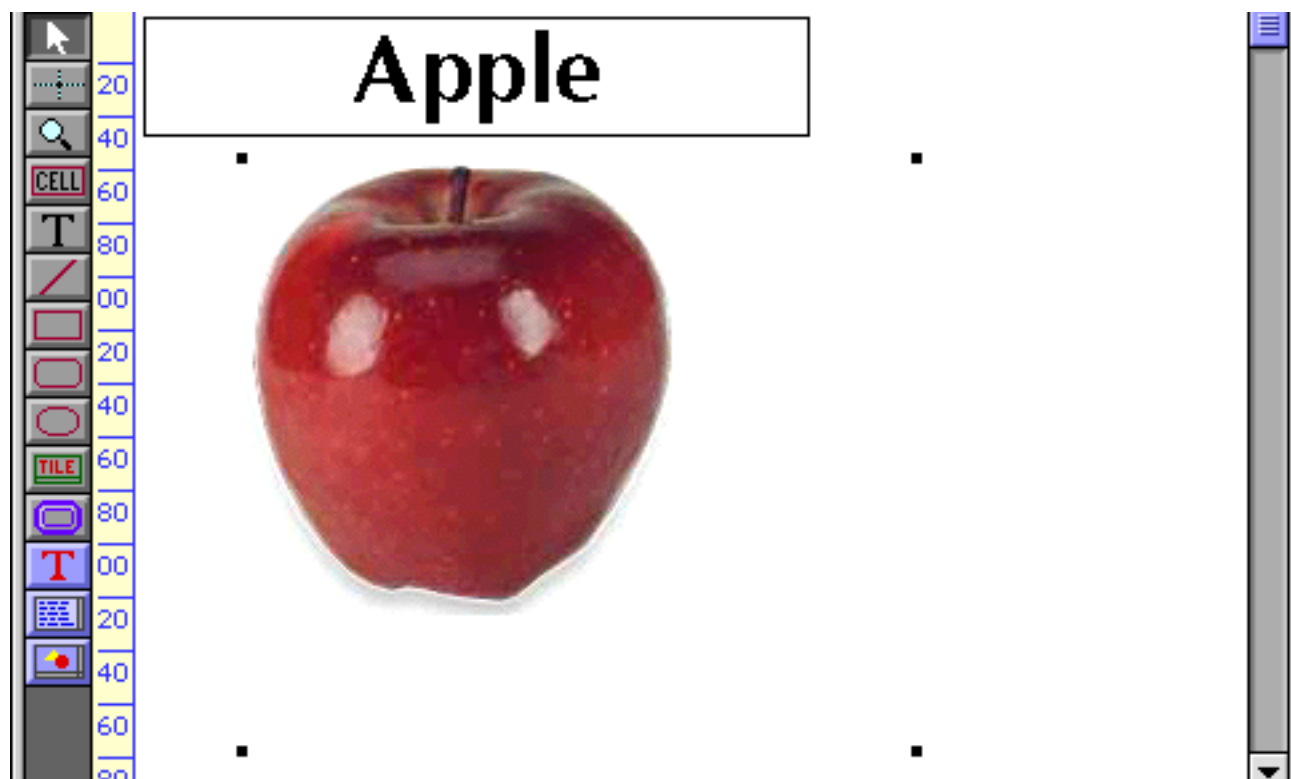
At a minimum you must enter a valid formula into the dialog. In this case the formula is simply **Fruit**.



To allow Panorama to display images directly from disk files you must enable the **Include Pictures on Disk** option.



Now press **OK**. Since the **Fruit** field contains the name of each fruit, this will automatically display the image for the correct fruit, in this case an apple.



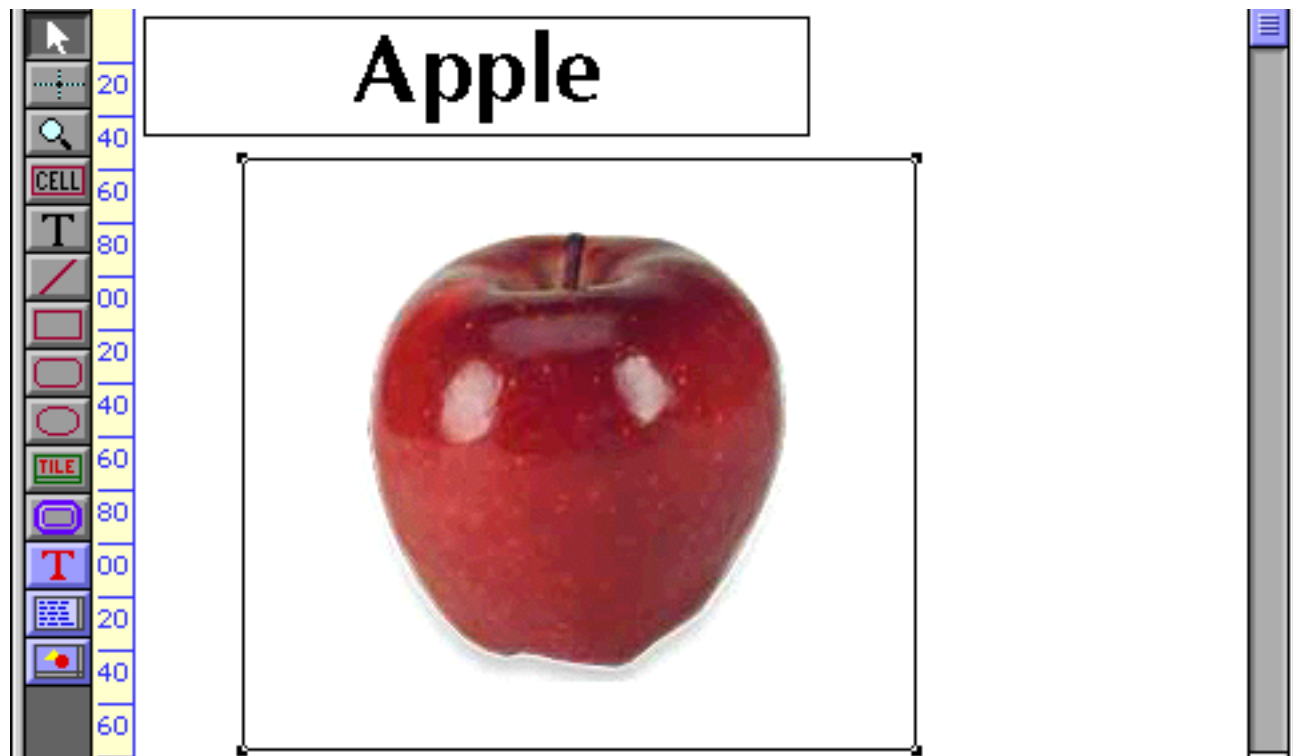
Notice that the apple is displayed in the upper left hand corner of the Super Flash Art object. You can also center the image. To do this, first click on the **Pointer** tool, then double click on the object. This re-opens the configuration dialog, allowing you to change the alignment.



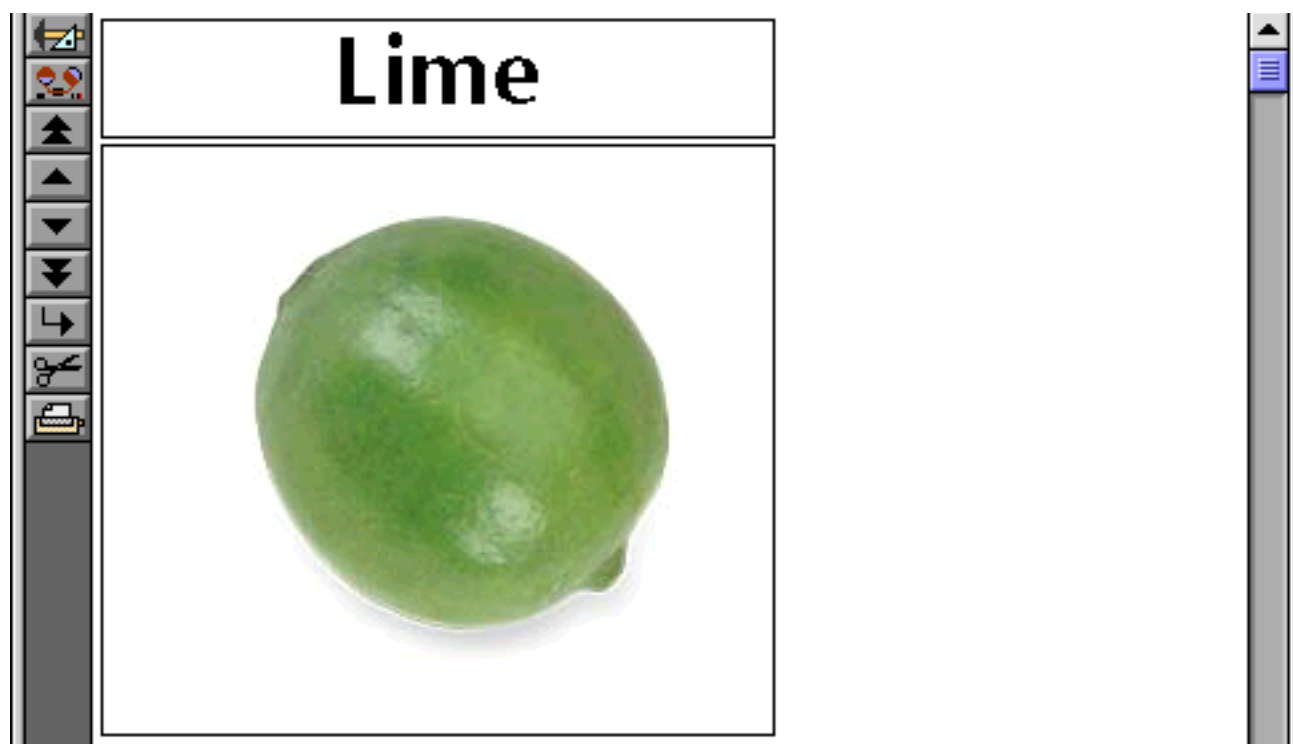
*upper left*

*centered*

We've also turned on the border for this object.



With a few adjustments the form is ready to use. As long as you have a photo with the correct name available, the correct photo will appear for each fruit.



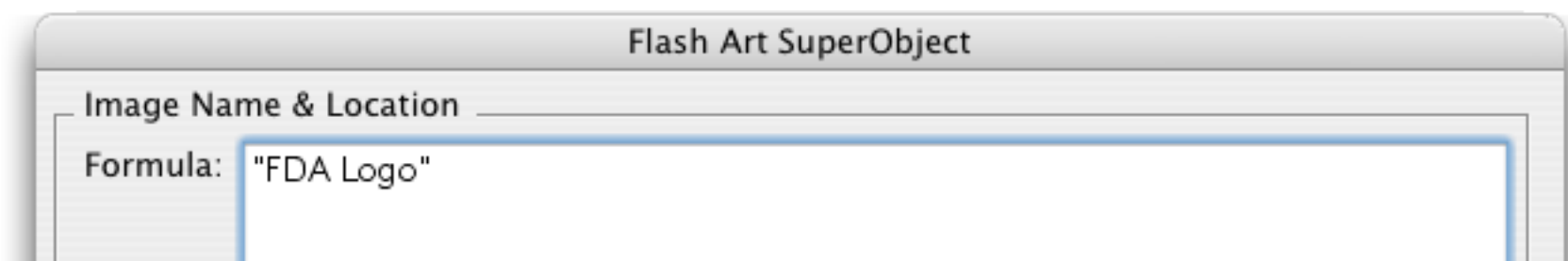
#### Using Flash Art to Display a Fixed Image

Flash art is usually used to display an image that changes, like the fruit in the previous example. However, it can also be used to display a fixed, unchanging image like a logo or a background. Of course you can also use a standard picture object to display a fixed image, so why use Flash Art? The primary advantage is that using Flash Art a single image can be displayed over and over again without taking up any additional memory or disk space. Only one copy of the image is needed, which can be used multiple times in the database (or even multiple databases).

To display a fixed, image, simply enter the name of the image into the Flash Art formula, surrounded by quotes ("). For example, suppose you have an image named **FDA Logo**.



If the image file is in the same folder as the database you can display this image in any form or report by using this formula in the Super Flash Art object.



You can use this logo as many times as you like within a single form or across multiple forms.

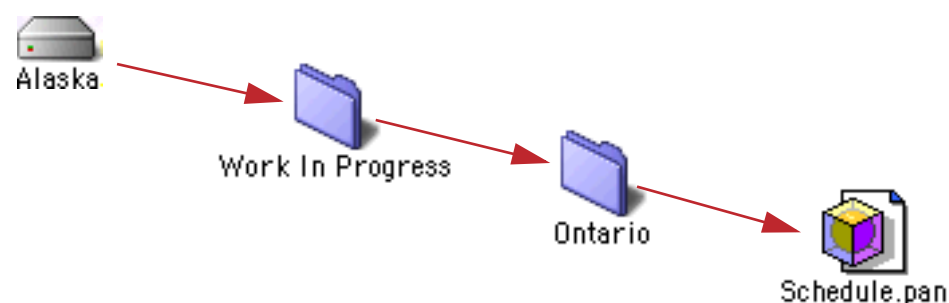


As an added bonus, if you ever need to modify the logo it only needs to be modified in one place.

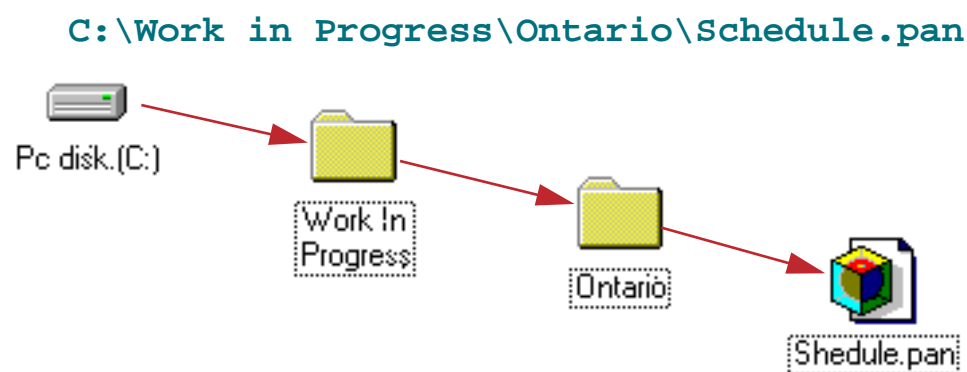
#### Displaying Images in a Different Folder (Directory)

Flash Art images don't have to be in the same folder as the database—they can be in any folder (directory) on your hard disk, or even on a removable media like a CD-ROM or a Zip disk. On the Macintosh, the exact location of any file can be specified by stringing together the name of the volume (disk) and the folders, each separated by a colon.

**Alaska:Work in Progress:Ontario:Schedule.pan**



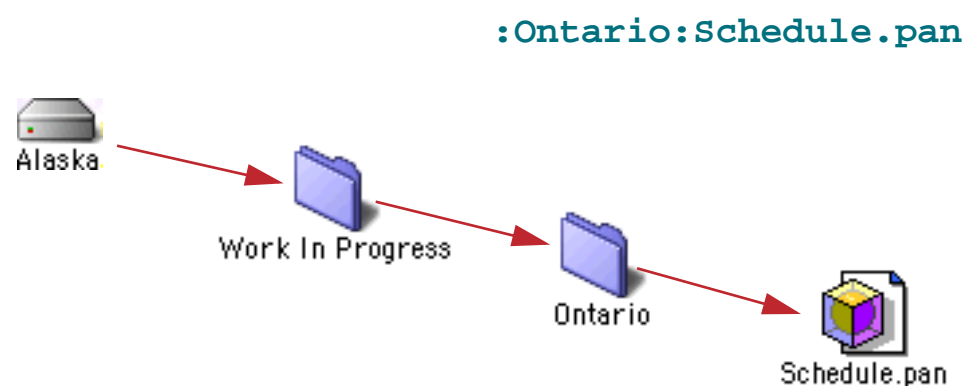
Windows systems are similar, but backslashes (\) are used instead of colons, and drive names always consist of letters followed by a colon (A:, B:, C:, etc.).



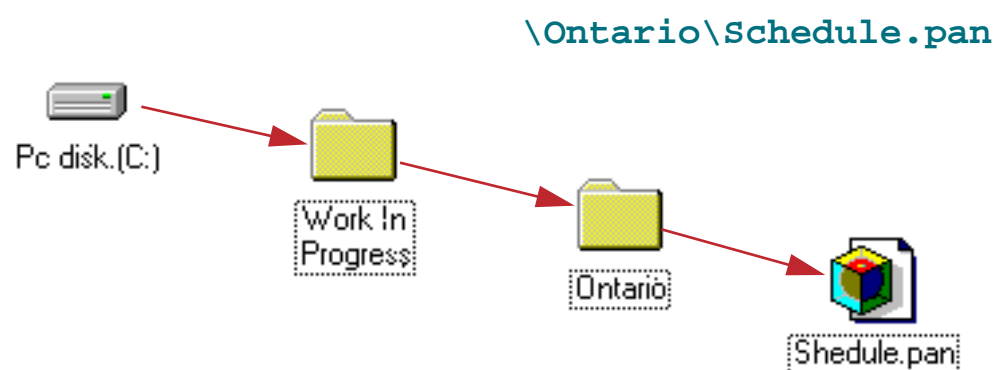
For cross platform compatibility, Panorama also allows you to use colons when using Panorama for Windows, like this:

`C::Work in Progress:Ontario:Schedule.pan`

A file's location may also be specified relative to the current database. For example, suppose the current database was in the `Work in Progress` folder. In that case you could specify the location of the `Schedule.pan` file by simply leaving off left hand portion of the specification. The specification must begin with a colon or backslash to indicate that it is relative to the current folder and not an absolute location.



On PC systems you can specify this relative location like this:



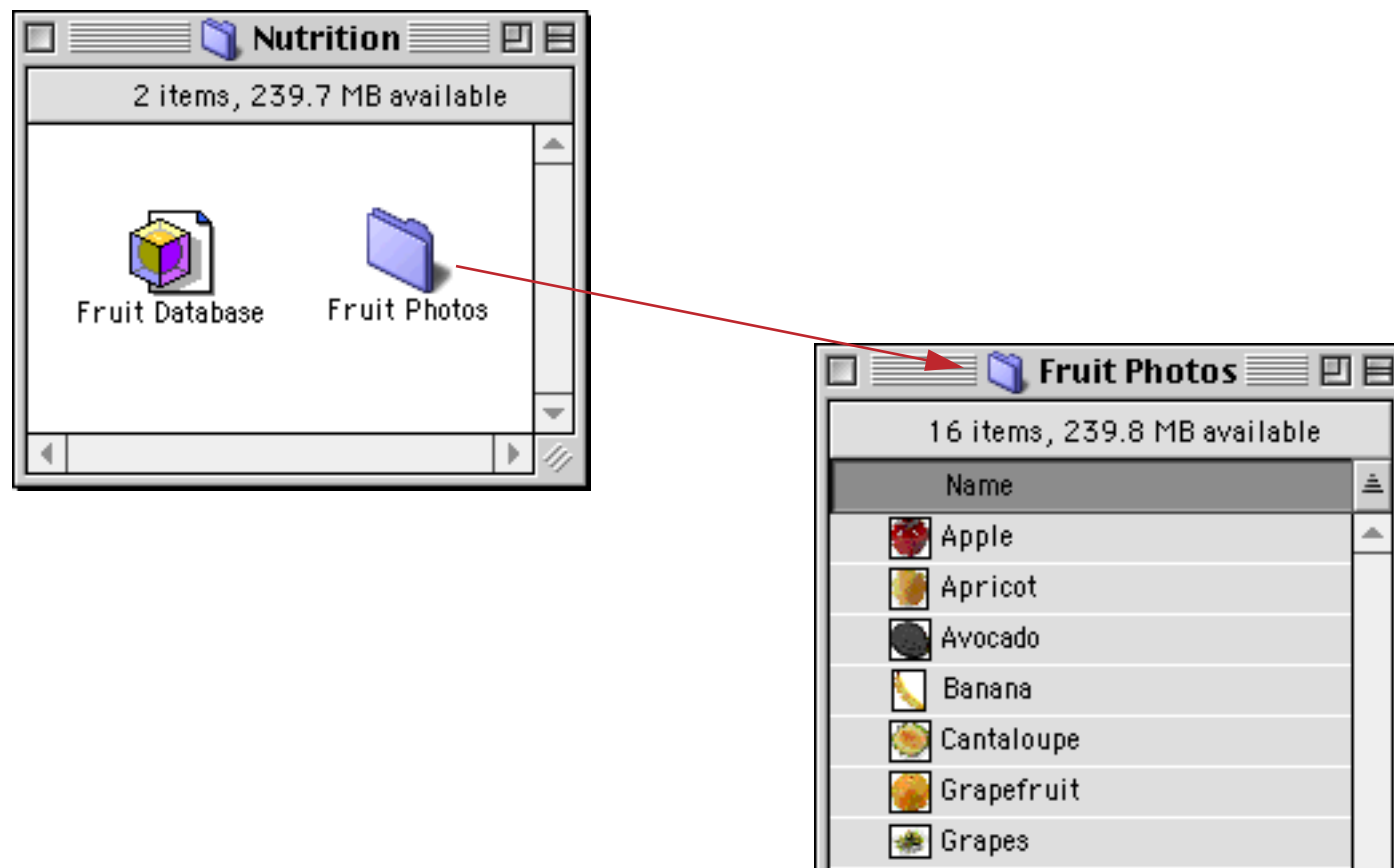
However, keep in mind that on PC systems Panorama will accept `:` instead of `\`. Therefore, the specification

`:Ontario:Schedule.pan`

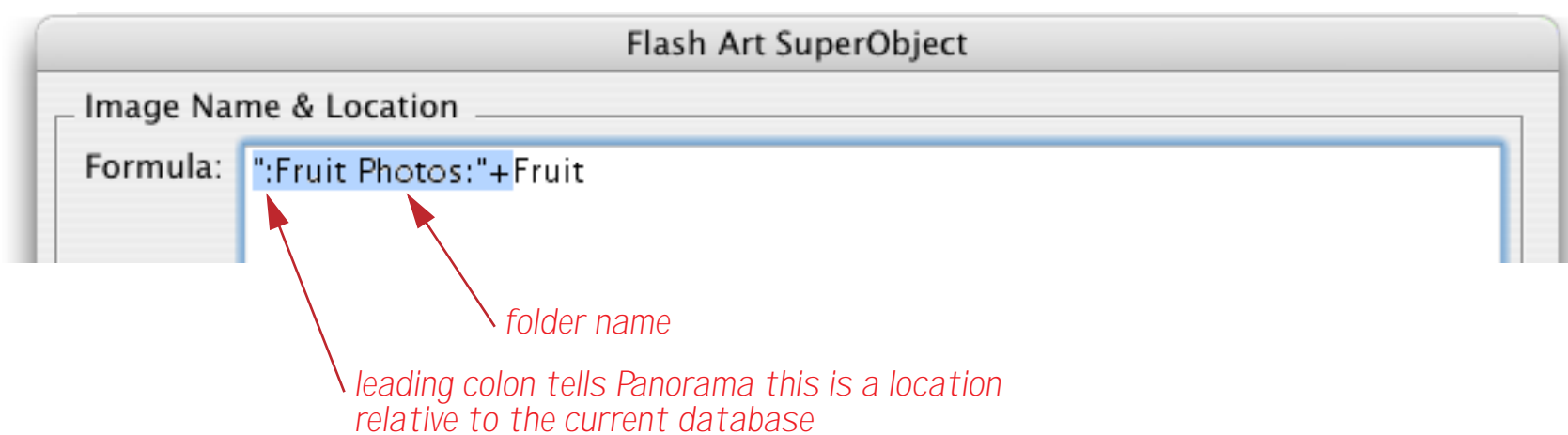
will work on both Windows and MacOS based computers. You should use colons if your database might ever be used on both Macintosh and Windows computers.



To illustrate all of this with a real-world example, let's revise the Fruit Nutrition database from the previous section. Instead of placing all of the image files in the same folder as the database, we will move them to their own folder, with that folder in the same folder as the database itself.

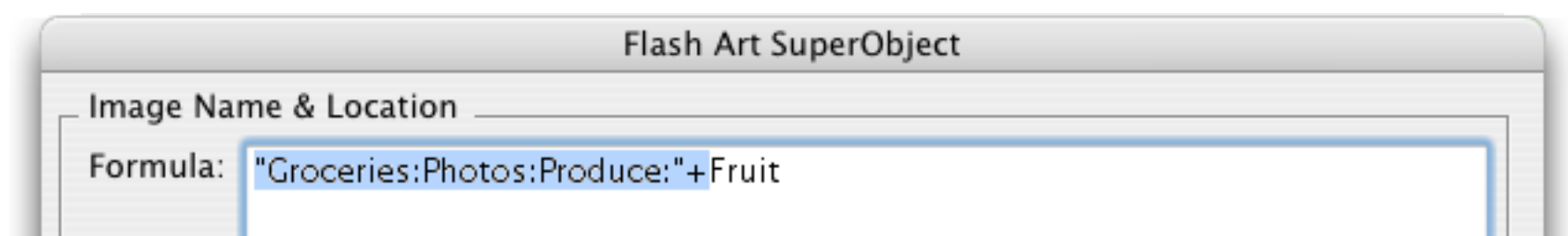


The **Fruit Photos** folder is said to be **nested** inside the **Nutrition** folder, which is the folder that actually contains the database. To display the photos, the formula in our Super Flash Art needs to be modified.

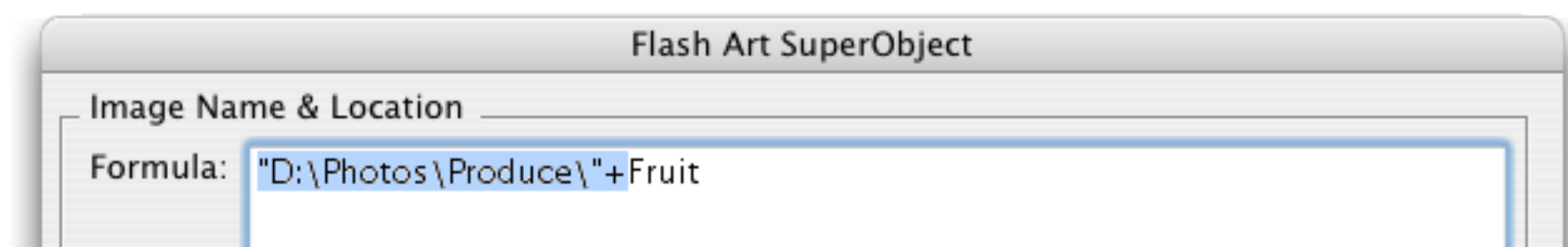


The new formula combines the fruit name with the folder location. For example, if the fruit name is **Apricot**, the result of the formula will be **:Fruit Photos:Apricot** — the exact location for the image.

A similar technique can be used to specify an absolute location for the image. Suppose you have a CD-ROM named **Groceries** that contains the images nested within the **Produce** folder within the **Photos** folder. The formula below could be used to display the images.



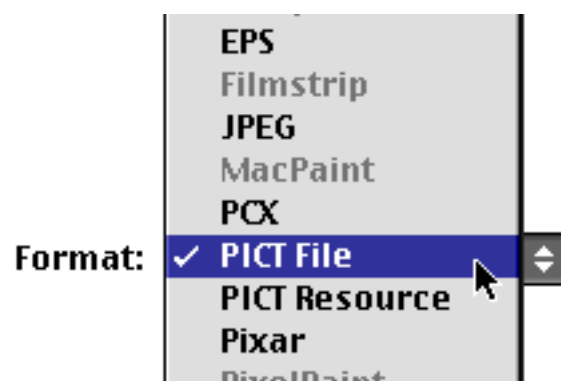
On the PC the CD-ROM drive is usually **D:**, so this would be the formula.



All of the previous examples have assumed that all of the images are all in the same folder. However, that is not necessary. If you wish, the folder location may be included in the database itself, allowing each images to be stored in a different folder. The folder location may be combined in with the image name in a single database field, or they may be stored in separate fields and combined in the Flash Art formula.

#### Displaying Non PICT Images (Enhanced Image Pack)

The standard configuration allows Panorama to display images in PICT format. This is the standard format for images on Mac OS computers, and is easily created using graphics programs on both Macintosh and Windows computers. This illustration shows how the PICT format may be selected when saving a file in Adobe Photoshop.

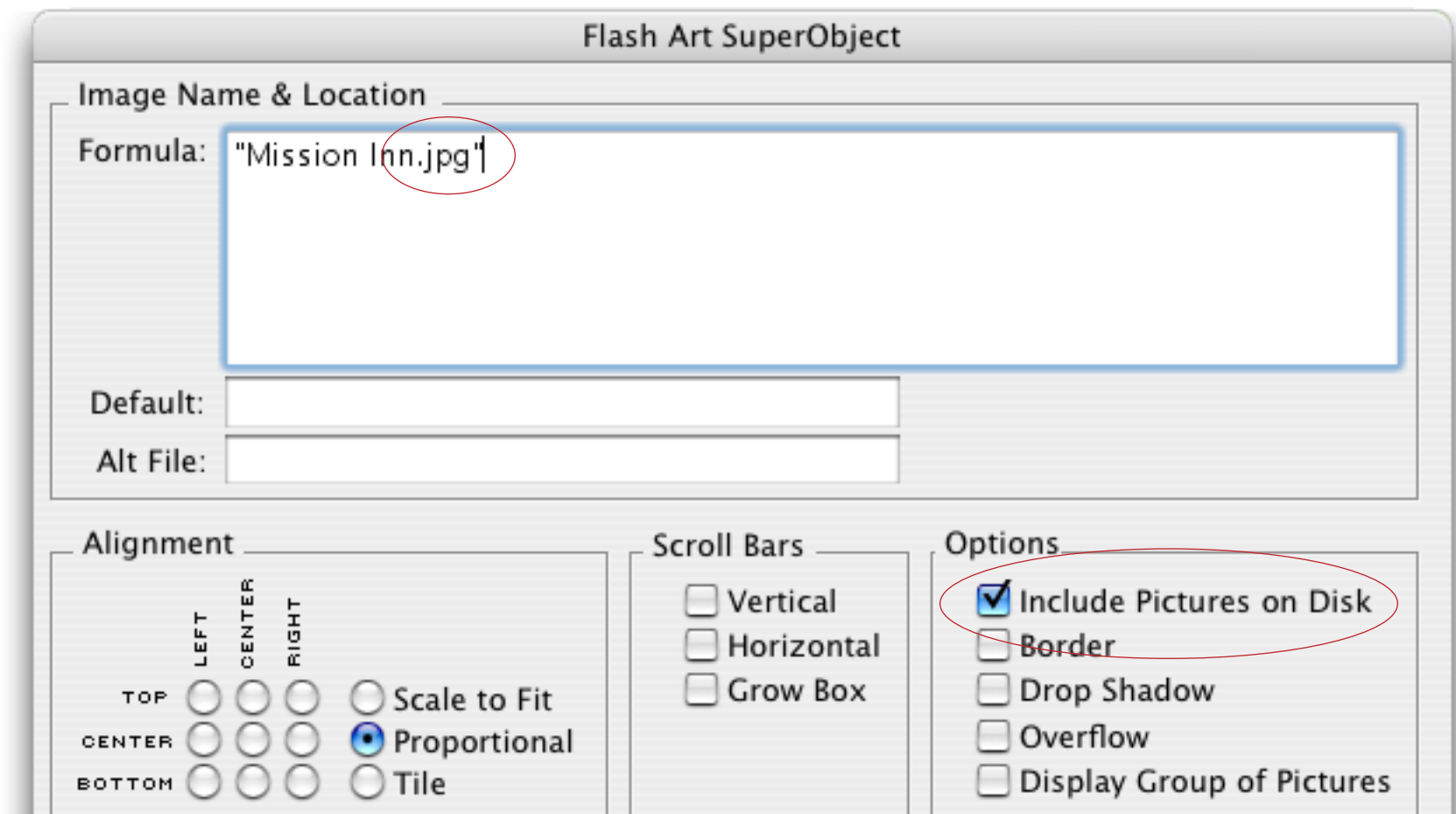


The optional **Enhanced Image Pack** gives Panorama the ability to display a wide variety of other image formats. This table lists some of the most popular image formats that can be displayed with this option.

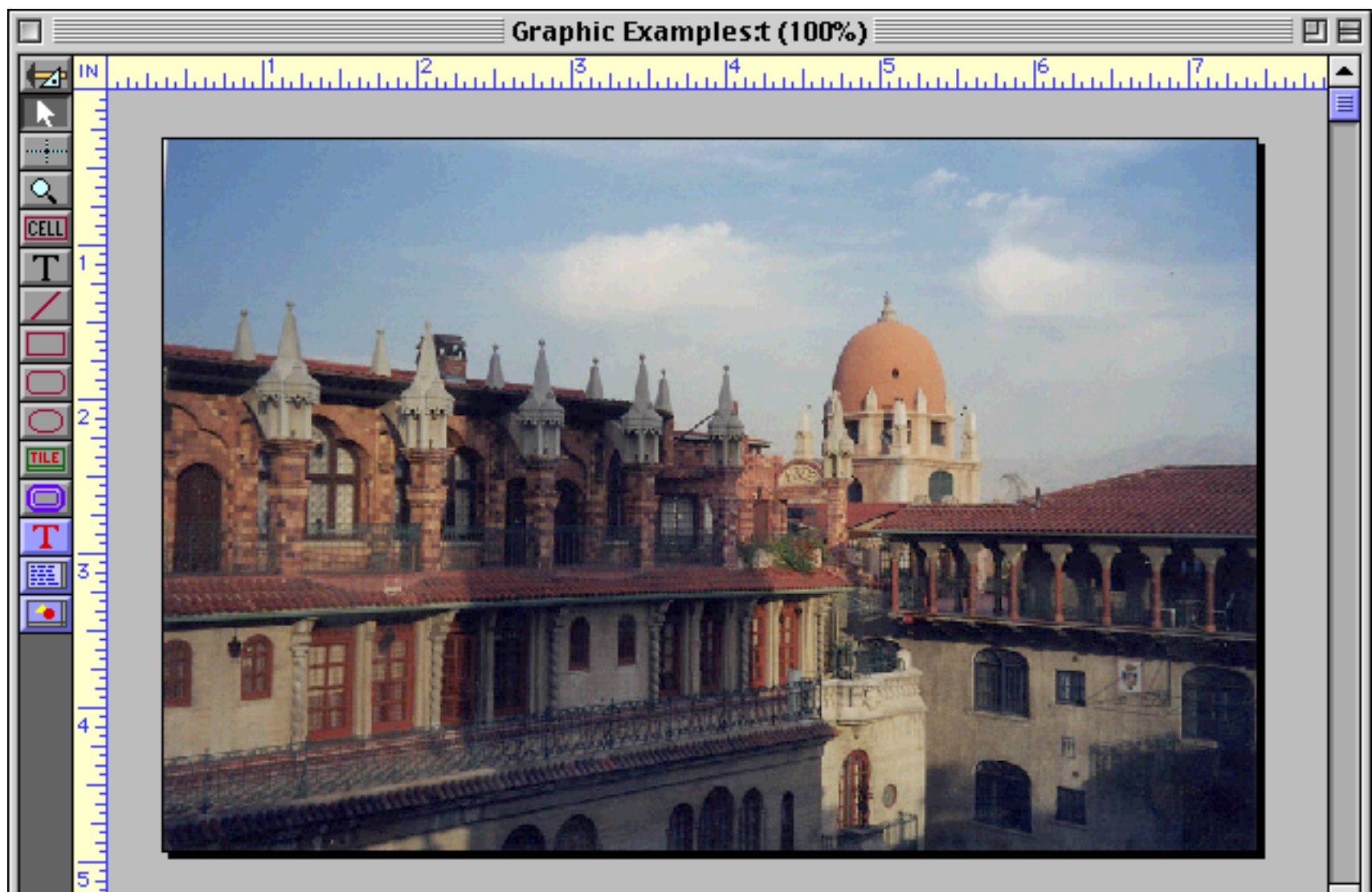
Image Type	PC Extensions	Notes
BMP	.bmp	Windows and OS/2 bitmap
JPEG	.jpg .jpeg	JPEG compressed image
PNG	.png	Portable Network Graphics bitmap
TIFF	.tif .tiff	Tagged Image Format
GIF	.gif	Common web format
PHOTOSHOP	.psb	Adobe Photoshop
FLASHPIX	.fpx	FlashPix bitmap
TARGA	.targa	

The **Enhanced Image Pack** requires that Apple Quicktime 4.0 or later be installed on your computer. If Quicktime is not already installed on your system you can download it from [www.apple.com](http://www.apple.com).

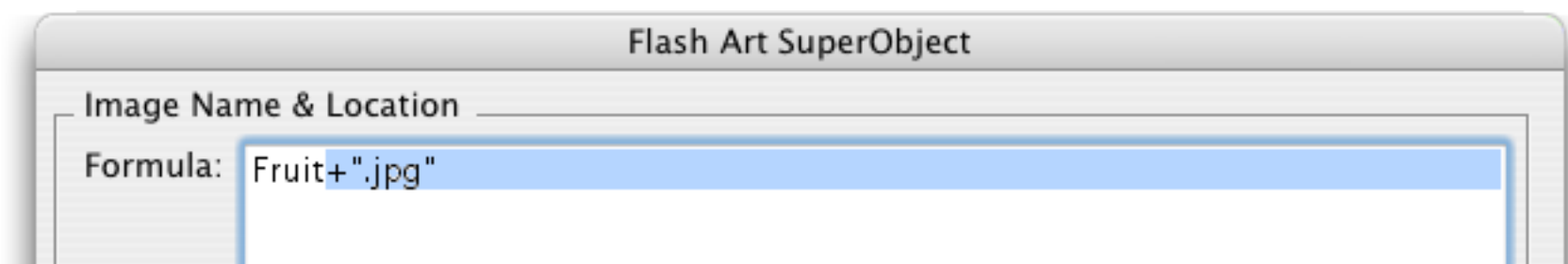
Once the **Enhanced Image Pack** is installed on your system you can display any of these image formats by name, just as with PICT images. This illustration shows how to display a fixed JPEG image.



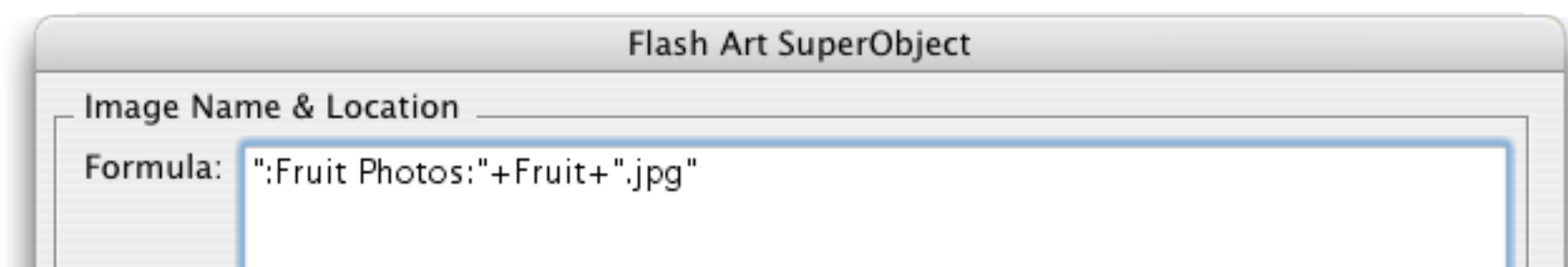
Make sure you enable the **Include Pictures on Disk** option, as shown above. Here is the actual JPEG image being displayed in the form.



Of course usually the images you display will be variable, not fixed. For example, suppose the fruit images used in the previous examples were JPEG images instead of PICT format. In that case you could display the images using this formula.



Or, if the images were nested in a different folder you could use a formula like this (see “[Displaying Images in a Different Folder \(Directory\)](#)” on page 326).



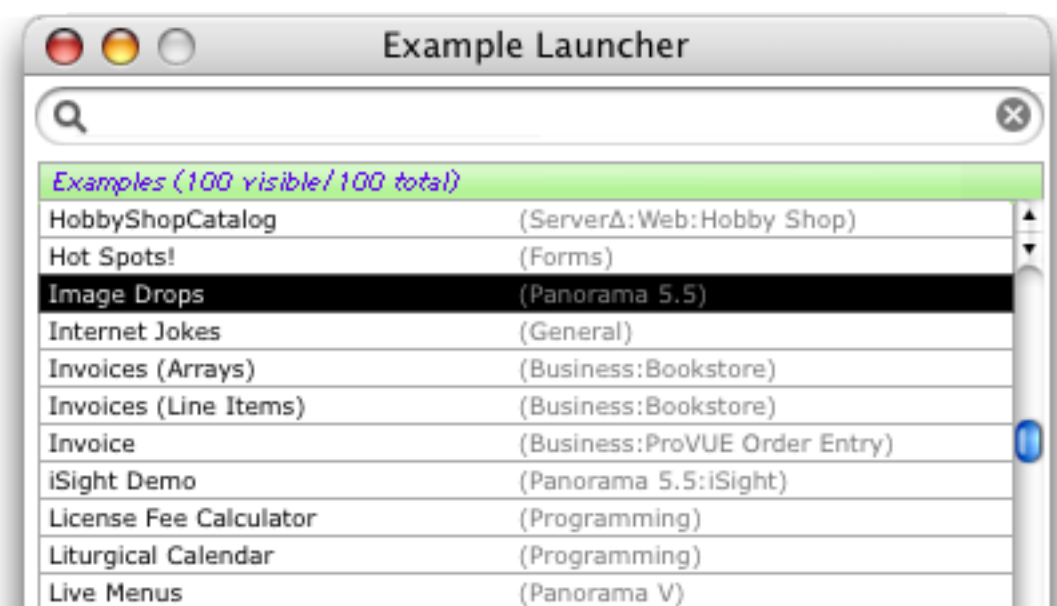
In addition to displaying images the Enhanced Image Pack can also convert an image from one format into another (see “[Converting Between Image Formats](#)” on page 1842 of the **Panorama Handbook**). For more information on ordering the **Enhanced Image Pack** visit our website at <http://www.provue.com>.

### Flash Art Image Drag and Drop

If you have a collection of image files on the hard drive the simplest way to get them into a database may be to simply drag them from the Finder onto your database (Macintosh only). You can add this feature to your existing databases or you can start with the **Image Drops** example database that comes with Panorama.

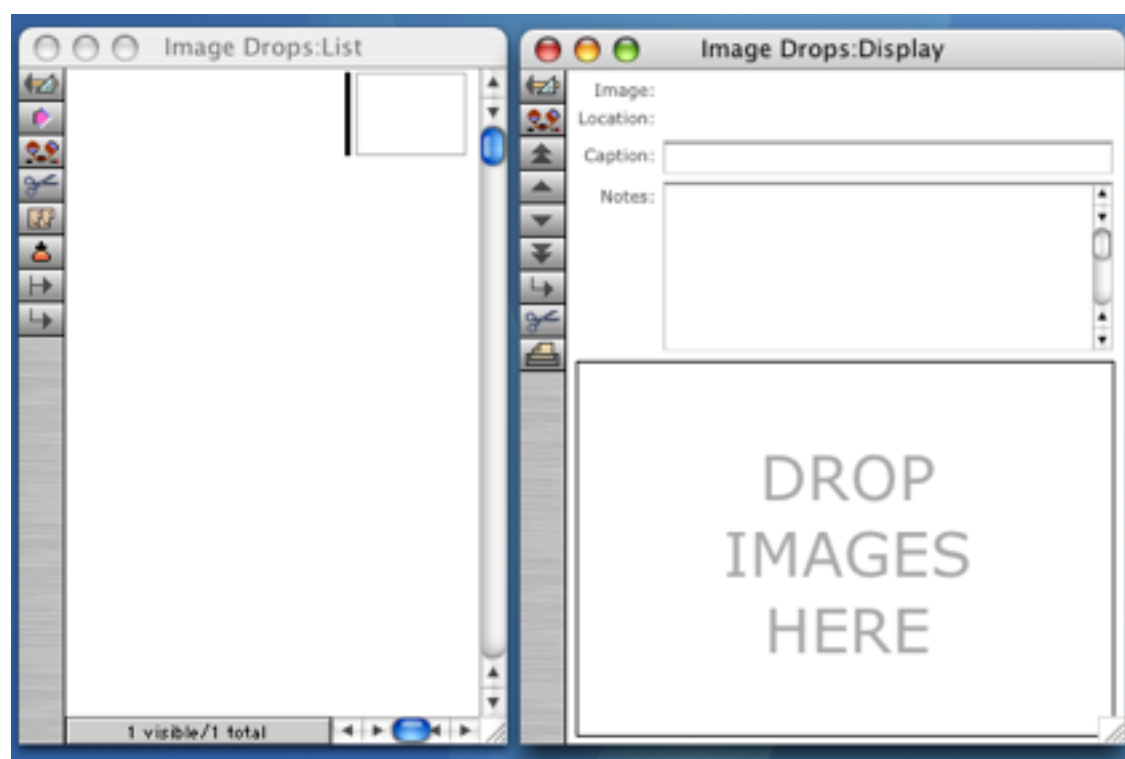
### The Image Drops Example Database

To try out the **Image Drops** example start by opening the **Example Launcher** wizard (in the **Demos** submenu of the **Wizard** menu). Scroll down and find **Image Drops**, then double click to open it.

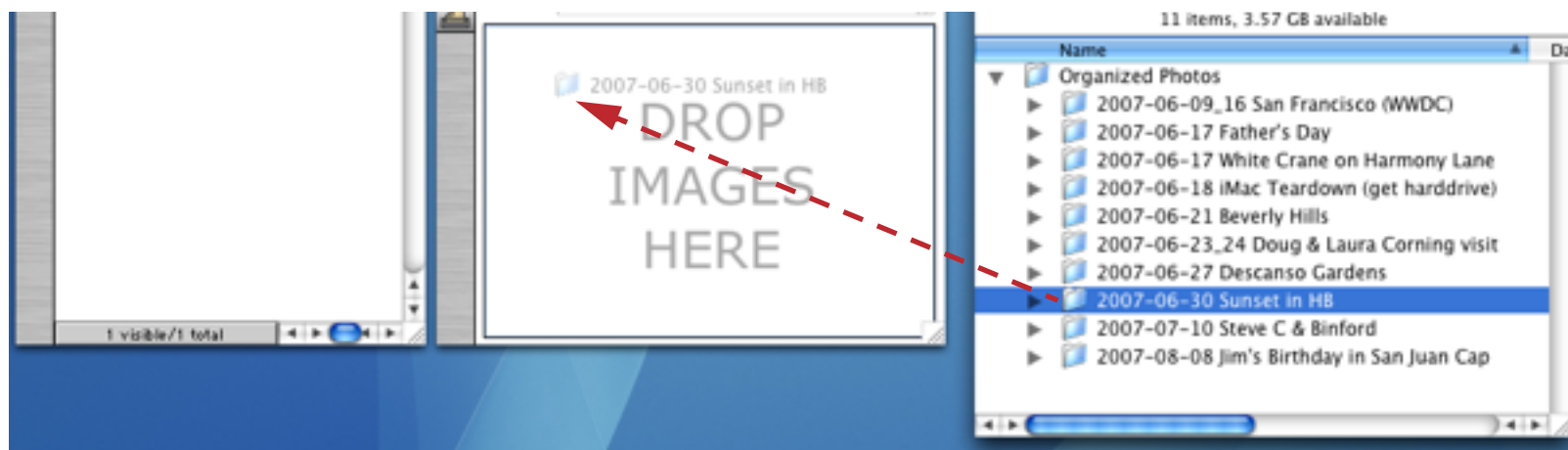




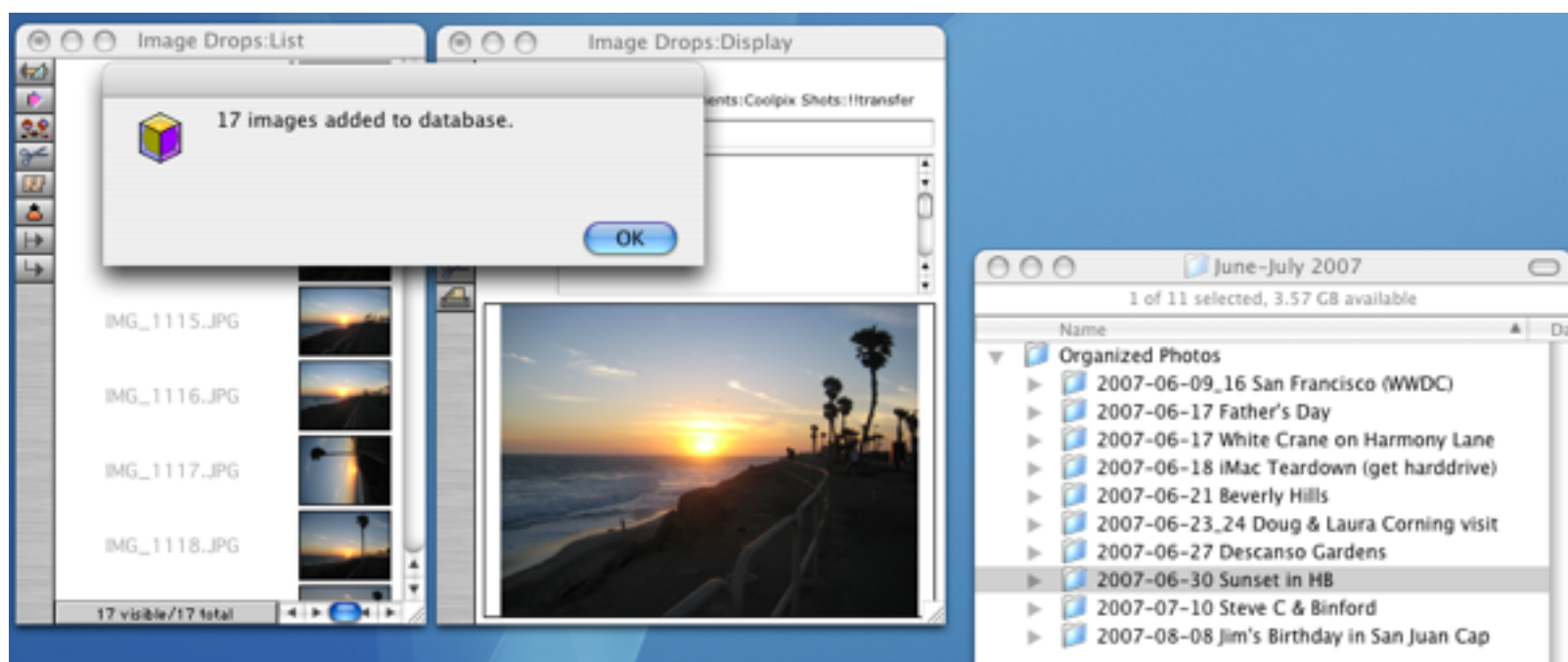
This example initially opens with two windows — a list of images on the left and a detailed image form on the right.



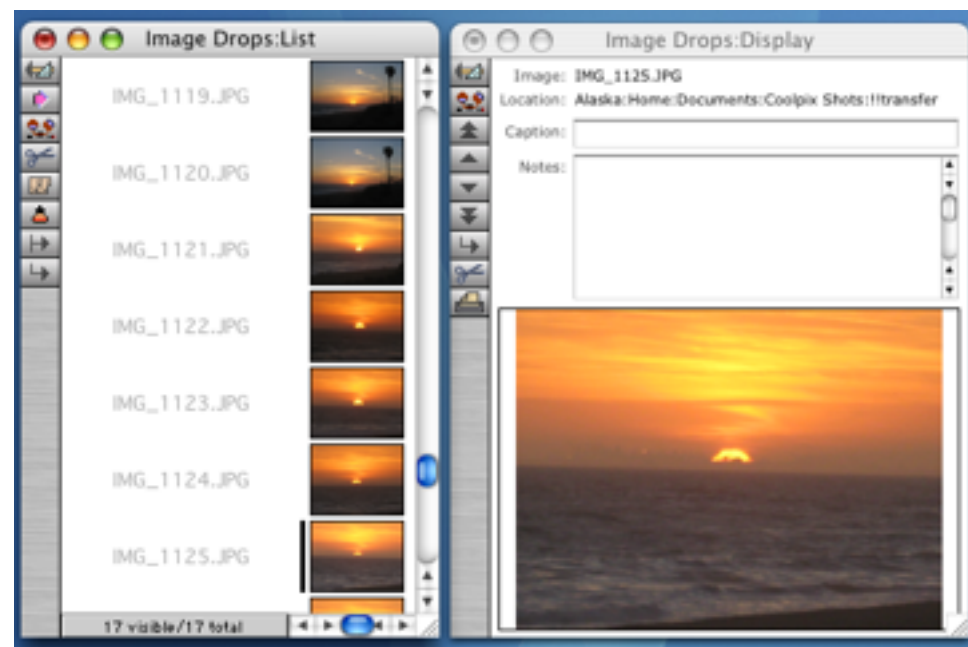
To add images to this database simply drag one or more image files (or even a folder that contains images) from the Finder into the area marked **DROP IMAGES HERE**. (Note: If the images are JPEG, TIFF, or other non PICT images you must have the Enhanced Image Pack installed, see “[Displaying Non PICT Images \(Enhanced Image Pack\)](#)” on page 329).



After a short delay Panorama will report the number of images added to the database.



I can move from record to record to see the different images that have been added.



Of course the images themselves haven't been brought into the database — just the names and locations, as we can see by opening the data sheet.



In this database the image location (path) and name are stored in a field called **Image**. The other fields in the database can be used for anything you want.

#### Adding Drag and Drop Images to a Database

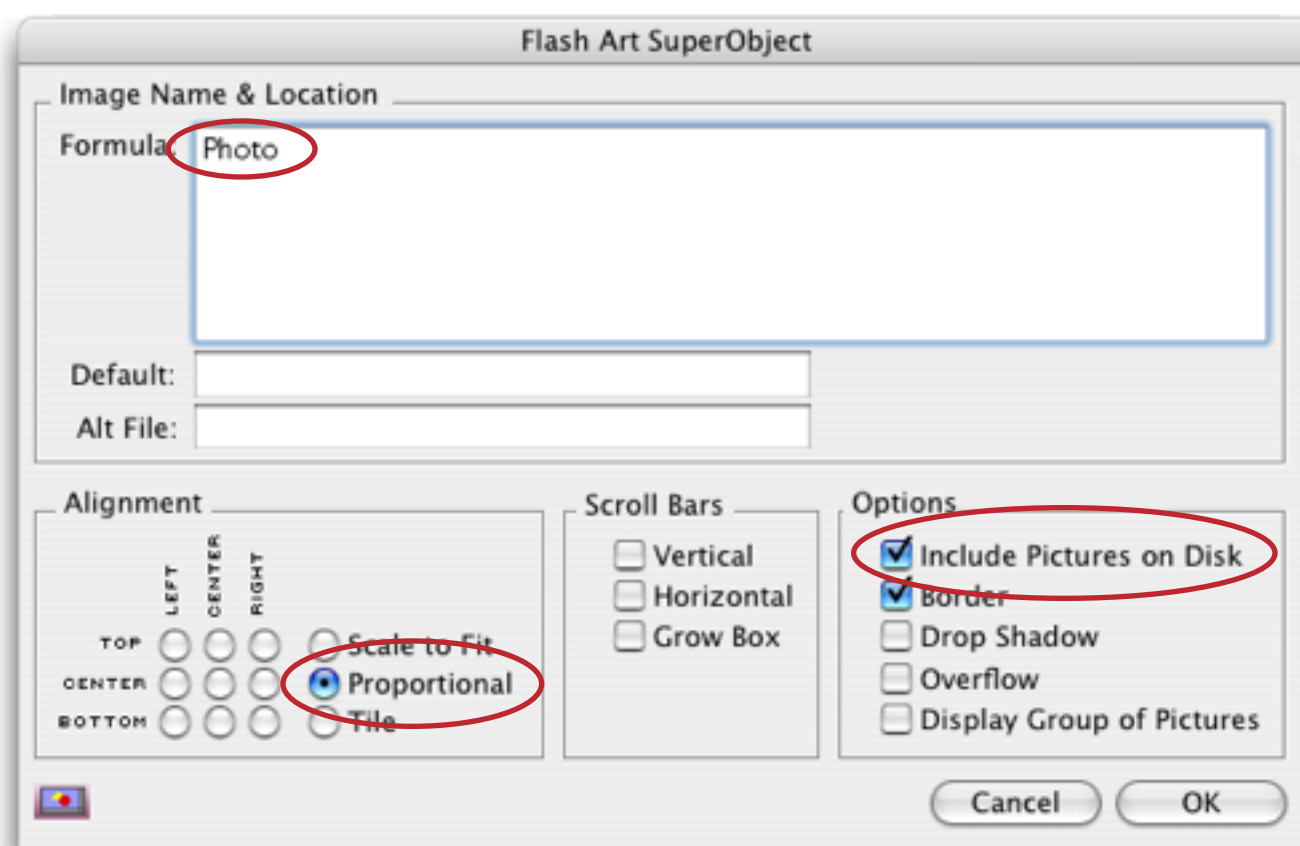
If you are starting from scratch one easy way to make an image database is to start with the example Image Drops database. Simply open this database, use the **Reveal in Finder** command to locate the file on the hard drive and make a copy for your own use. Add whatever fields, forms and procedures you need.



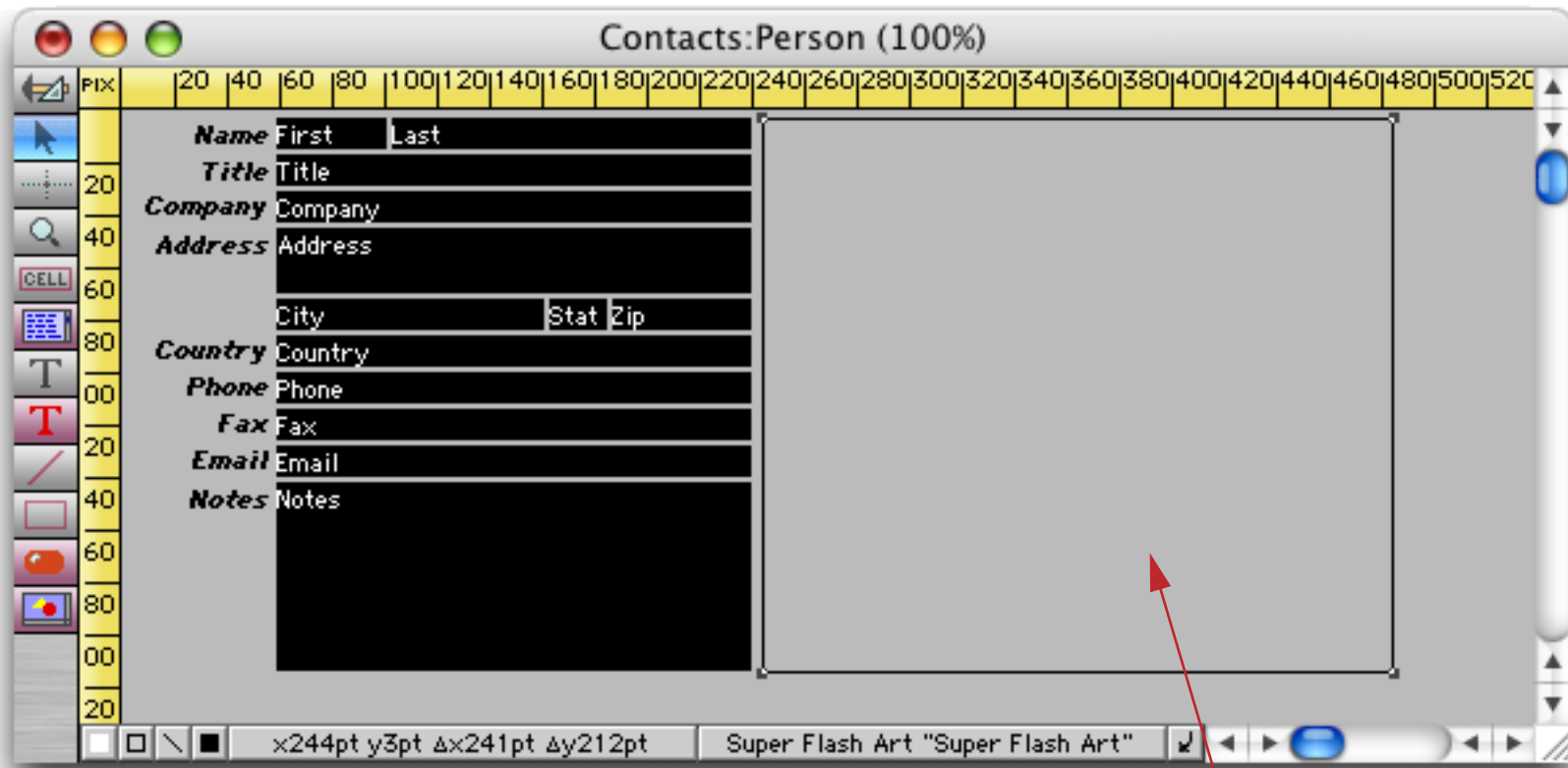
It's also relatively easy to add image drag and drop to an existing database. The first step is to add a field to the database to hold the image path and name. To illustrate this I'll use the design sheet to add a field named **Photo** to a contacts database (you can also use the Add Field command in the Setup menu).

Field Name	Type	Digits	Align	Output Pattern	Input Pattern	Range	Choices	Link	Clair	Tab	Caps
First	Text	0	Left			Alphabetic			Off	1 S	Word
Last	Text	0	Left			Alphabetic			Off	1 S	Word
Credit Card	Text	0	Left			Any	Cash Ch		Off	Off	Off
Title	Text	0	Left			AZ az			On	2 S	Word
Company	Text	0	Left			AZ az			Off	2 S	Word
Address	Text	0	Left			AZ az09			Off	2 S	Word
City	Text	0	Left			AZaz			On	2 S	Word
State	Text	0	Left			Alphabetic			Off	1 S	All
Zip	Text	0	Left			09AZ			Off	1 S	Off
Country	Text	0	Left			AZaz			Off	2 S	All
Phone	Text	0	Left		( ) _ _ - _	Numeric			Off	Off	Off
Fax	Text	0	Left		( ) _ _ - _	Numeric			Off	Off	Off
Email	Text	0	Left			AZaz09@@..			Off	Off	Off
Notes	Text	0	Left			Any			Off	Off	Off
<b>Photo</b>	<b>Text</b>	<b>0</b>	<b>Left</b>			<b>Any</b>			<b>Off</b>	<b>Off</b>	<b>Off</b>
Sequence	Numeric	0	Right			Any			Off	Off	Off
LastModified	Numeric	0	Right			Any			Off	Off	Off

The next step is to add a Super Flash Art object to one or more forms in the database. The **Formula** for this object must be set to the name of the field you just set up, in this case **Photo**. Make sure that the **Include Pictures on Disk** option is set. For most photographic applications you'll also want to set the **Proportional** alignment option.

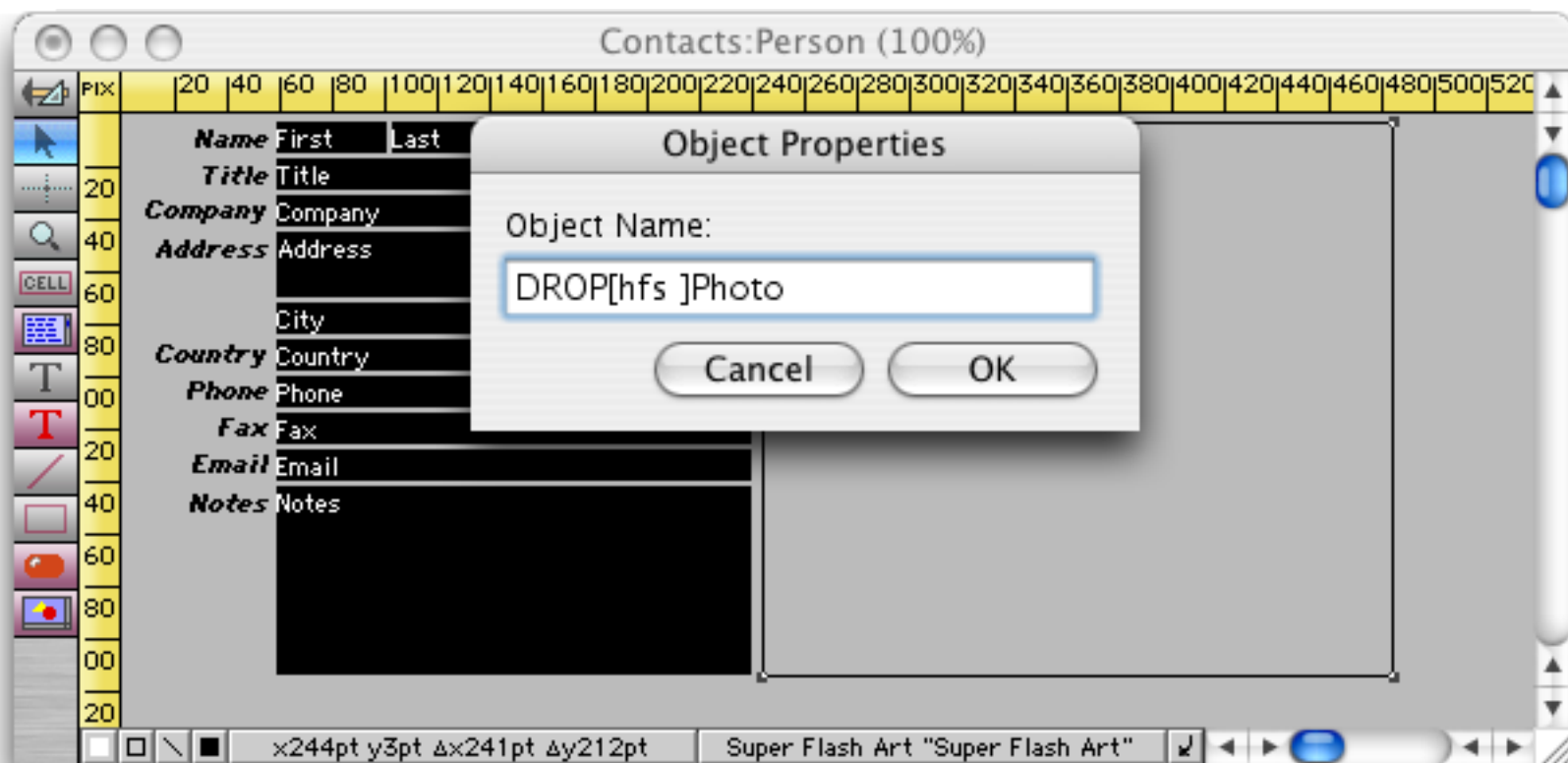


I've set up the super flash art object next to the contact information on the form.

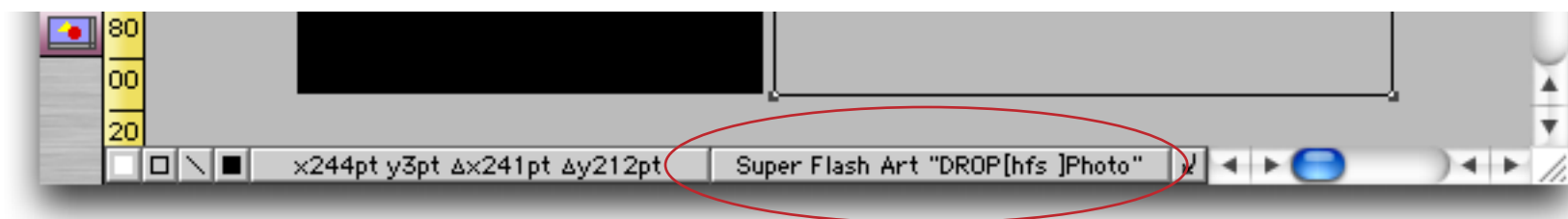


*super flash art object*

However, I'm not quite done with this object yet — I need to give it a special name. To give an object a name select the object and choose **Object Name** from the **Edit** menu (or click on the object name area in the Graphic Control Strip, see "[The Graphic Control Strip](#)" on page 259). I'll set the object name to **DROP[hfs ]Photo** (more on this in a moment).



When I'm done selecting the object again (by clicking on it) should show the name in the Graphic Control Strip (if that section of the strip is visible — see [“The Graphic Control Strip”](#) on page 259).

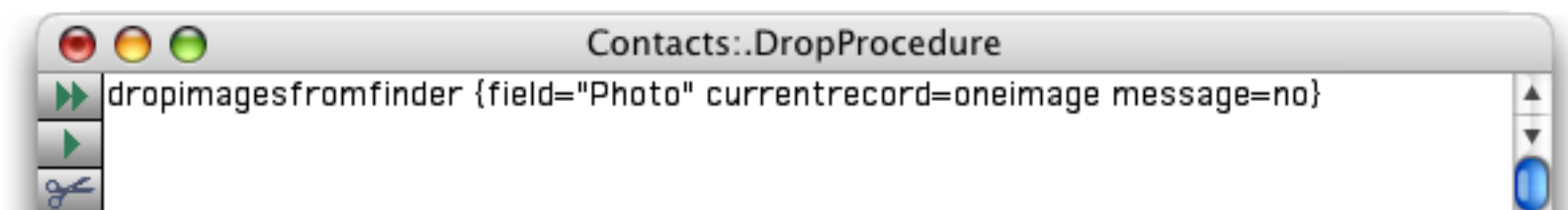


So why does the object name need to be **DROP[hfs ]Photo**? The name is divided into three portions.

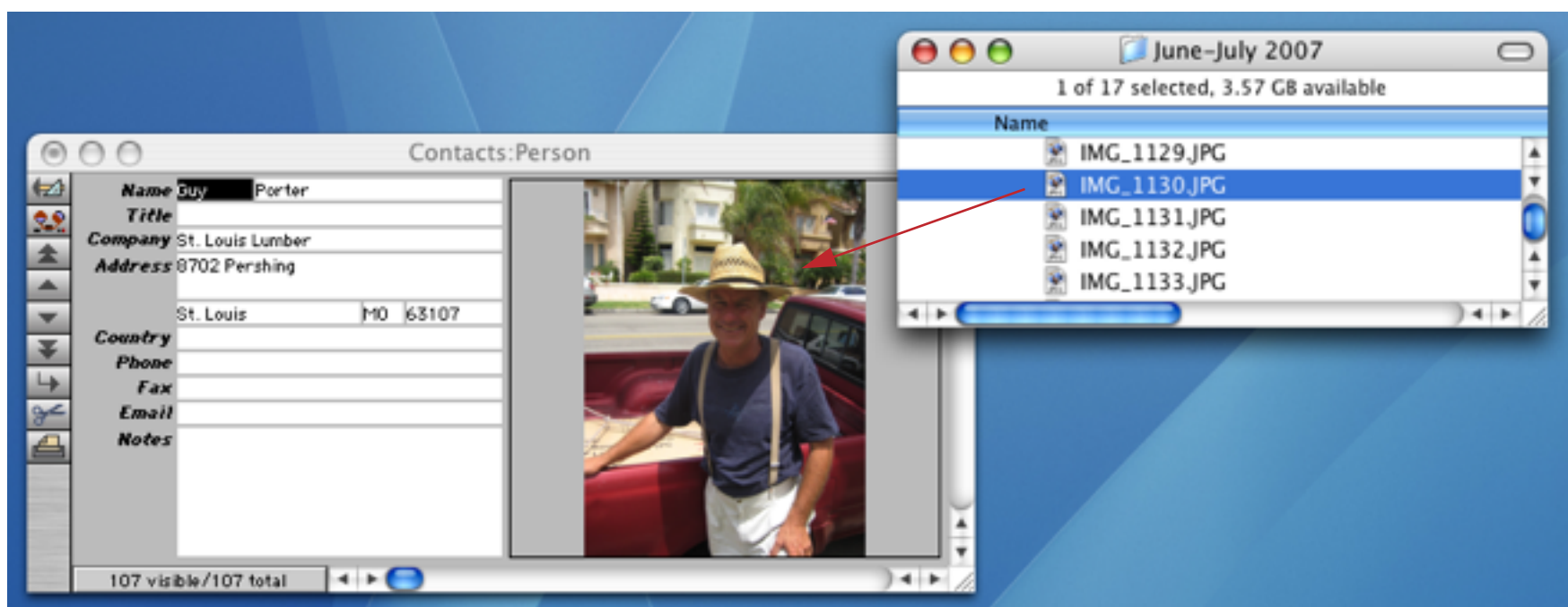
<b>DROP</b>	This portion of the name tells Panorama that data can be dropped on this object
<b>[hfs ]</b>	This portion tells Panorama what kinds of data can be dropped on this object, in this case files or folders (hfs stands for heiarhial file system, the original file system used by the Macintosh). If this section of the object name is [] then any type of data can be dropped on the object.
<b>Photo</b>	This final portion is optional. If you have multiple objects in a database that can have items dropped on them you can use this final portion to determine what object was dropped on.

To learn more about this topic see [“Receiving Dragged Data”](#) on page 1802 of the **Panorama Handbook**.

The final step is to set up a special procedure to handle the dropped files. This is easy, because the procedure only needs one line of code! The procedure must be named **.DropProcedure** — here it is:



Now I'm ready to drag an image from the Finder onto my form. Voila! The image appears.



Because I programmed this option with the **currentrecord=oneimage** option the photo appears in the current record instead of being added to a new record. (This also means that you can only drop one image at a time — if you drop more than one image only the first one will be used). To add an image to a different record simply move to that record and then drag the image onto the form.

### DropImagesFromFinder Options

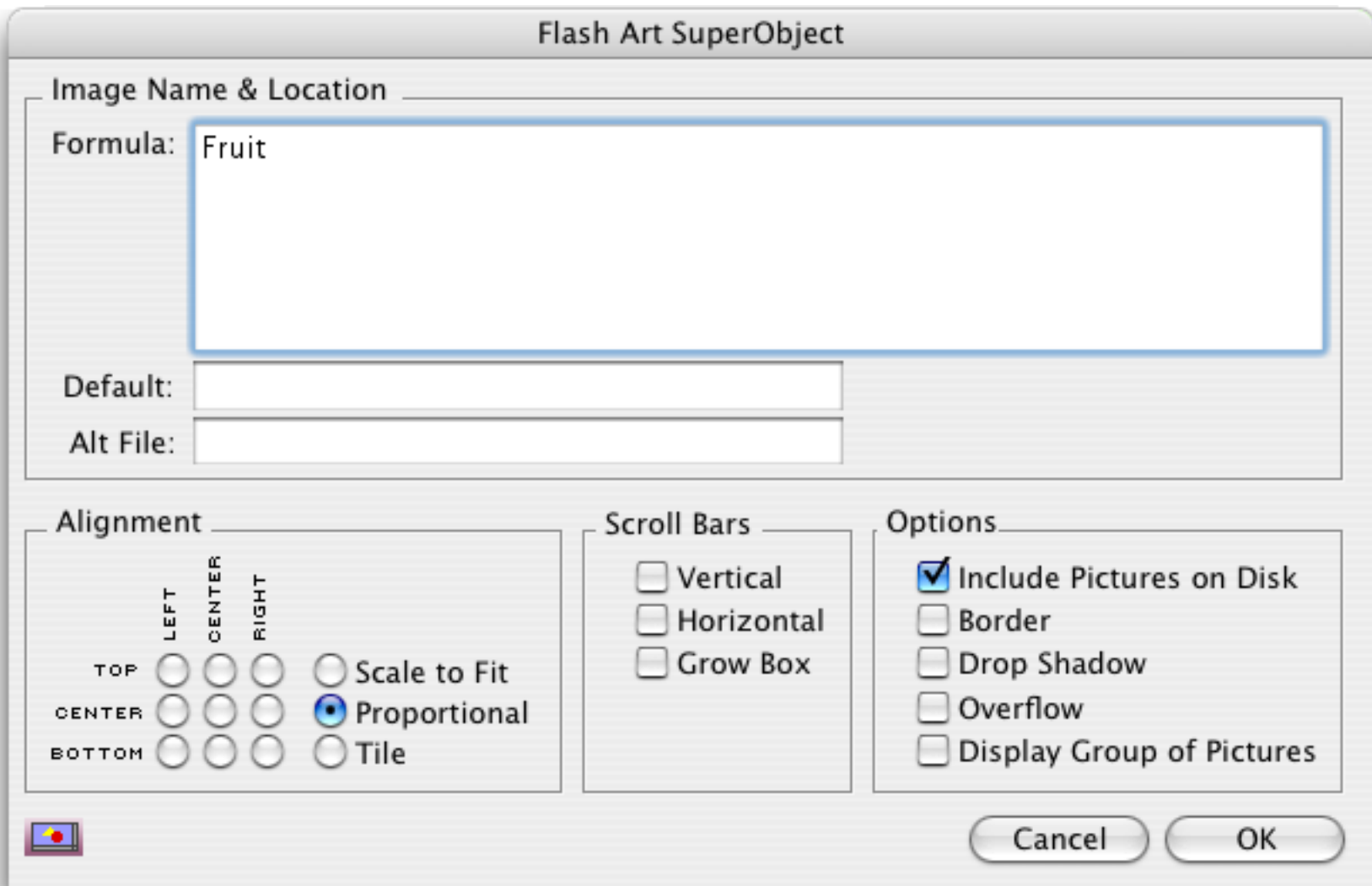
The key to image drag and drop is the `DropImagesFromFinder` statement. This statement takes the dropped files and adds their name and location to the database. The operation of this statement can be customized with a list of options. Each option must be specified using the format `optionname="value"` (the quotes may be omitted if there is no special characters or punctuation in the value).

Option	Description
<code>field=fieldname</code>	This option specifies what field the image paths and names should be placed into. If this option is omitted the image path & name will be placed in the current field.
<code>type=imagetype</code>	<p>This option specifies a type of image that will be allowed. You may choose from <code>jpeg</code>, <code>tiff</code>, <code>png</code>, <code>pict</code>, <code>bmp</code>, <code>pdf</code>, <code>gif</code>, <code>photoshop</code>, <code>mov</code>, <code>mpeg</code>, <code>wav</code>, <code>aiff</code>, <code>aac</code>, <code>mp3</code>, and <code>sd2</code>. (Note that some of these are movie types, you can drag and drop movies in addition to still images.) You can explicitly specify multiple types by using multiple <code>type=</code> options, for example:</p> <pre>type=jpeg type=tiff type=png</pre> <p>You can also specify “composite” types: <code>image</code> for all image types, <code>movie</code> for all movie types, or <code>sound</code> for all sound types, or <code>all</code> for all media types (including image, movies and sound) , for example:</p> <pre>type=image</pre> <p>If no type is specified the statement defaults to all image types (but not movie or sound).</p>
<code>currentrecord=oneimage</code> <code>currentrecord=yes</code> <code>currentrecord=no</code>	<p>If this option is <code>oneimage</code> the dropped image will replace the current image in the database. If multiple images are dropped only the first image will be used, any others will be ignored.</p> <p>If this option is <code>yes</code> all dropped images will be added to the current record only (as a carriage return delimited array). This allows multiple images per record (which could be displayed with a super matrix).</p> <p>If this option is <code>no</code> dropped images will be added to new records at the end of the database (unless they are duplicates and the <code>skipduplicate</code> option has been set to <code>no</code>).</p>
<code>skipduplicates=yes/no</code>	If this option is <code>yes</code> , images that are already in the database will not be re-added. The default if this option is not included is <code>yes</code>
<code>relativepaths=yes/no</code>	If this option is <code>yes</code> , only images that are in the same folder or in subfolders of the database will be added. Any images in other locations will be ignored. Instead of storing the absolute location of the image (starting with the disk name) relative paths will be used. The advantage of relative paths is that the database and images can be moved to a different location or even a different computer and the images will display correctly without adjustment. The disadvantage is that images that are not in the same folder as the database (or subfolders of that folder) cannot appear in the database. The default for this option is <code>no</code> .
<code>message=yes/no</code>	If this option is <code>yes</code> , the statement will display an alert with the results (how many images added and skipped). The default is <code>yes</code> .

By combining these options you can add dropped images to the database just about any way you want to.

## Super Flash Art™ Options

The SuperObject™ Flash Art configuration dialog has numerous options for customizing each object you create.

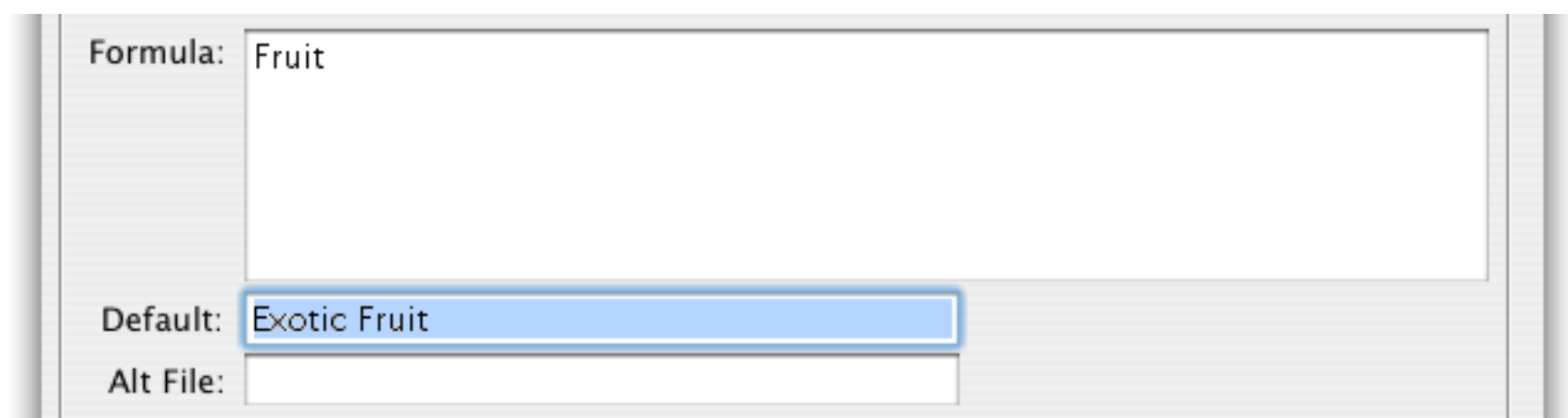


### Formula

This section of the dialog contains the formula that calculates the image name. The formula may be up to 255 characters long (to create longer formulas, see the next paragraph). Use this formula to combine one or more fields and/or variables into a picture name. See “[Formulas](#)” on page 501 to learn more about Panorama’s formula capabilities.

### Default

This is the name of the default image that should be displayed if Panorama cannot locate an image with the name specified by the formula. For example, consider the Fruit database used in the previous example. Suppose you encounter a new type of fruit for which you don’t have a picture? You could simply leave the image blank. Or, you could create a default image that will be used in these situations, like this.

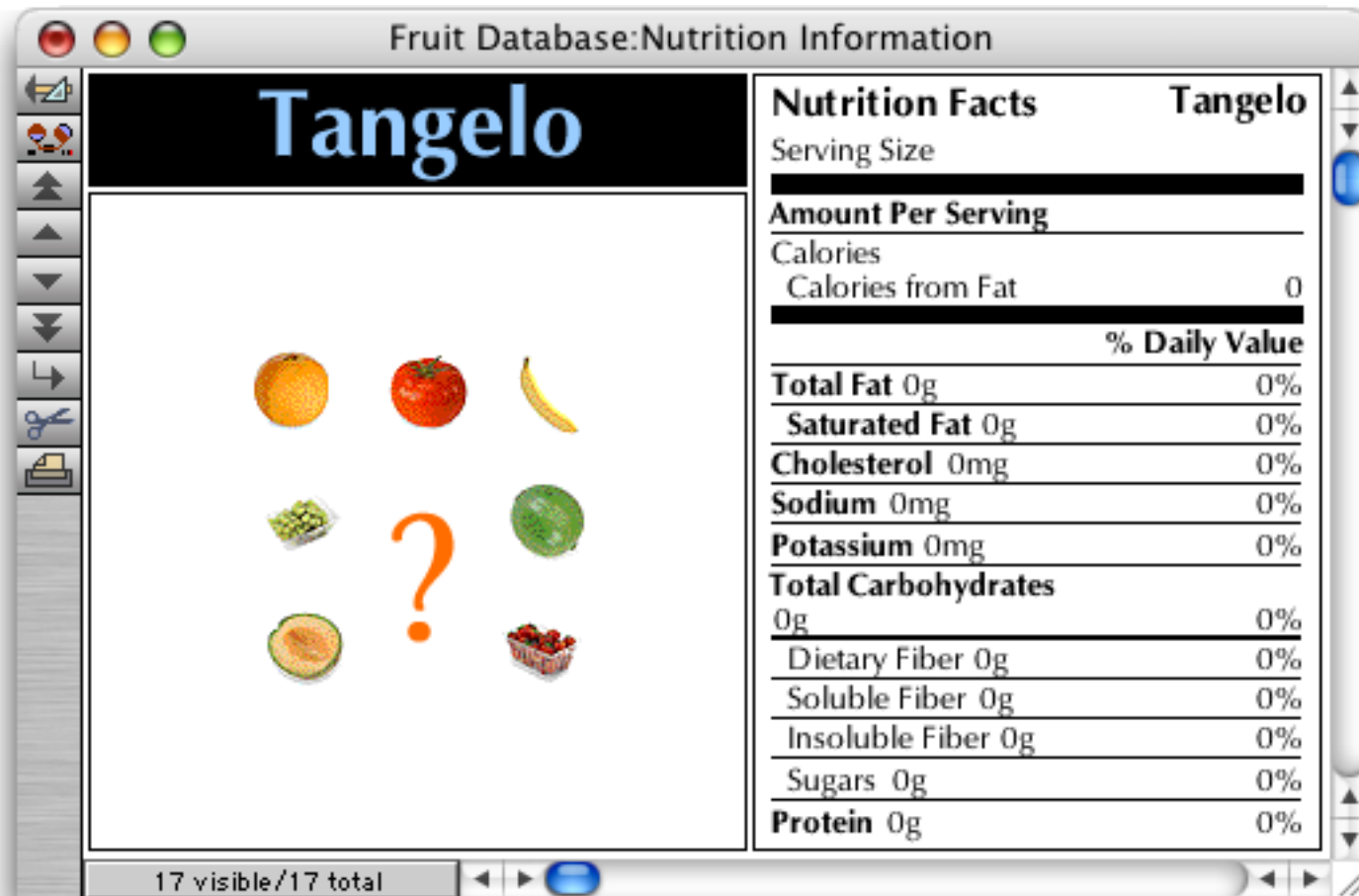




Now you can specify this image as the default. Notice that no quotes are needed (or allowed!). The default image cannot be changed on the fly — it is not specified by a formula like the main image.

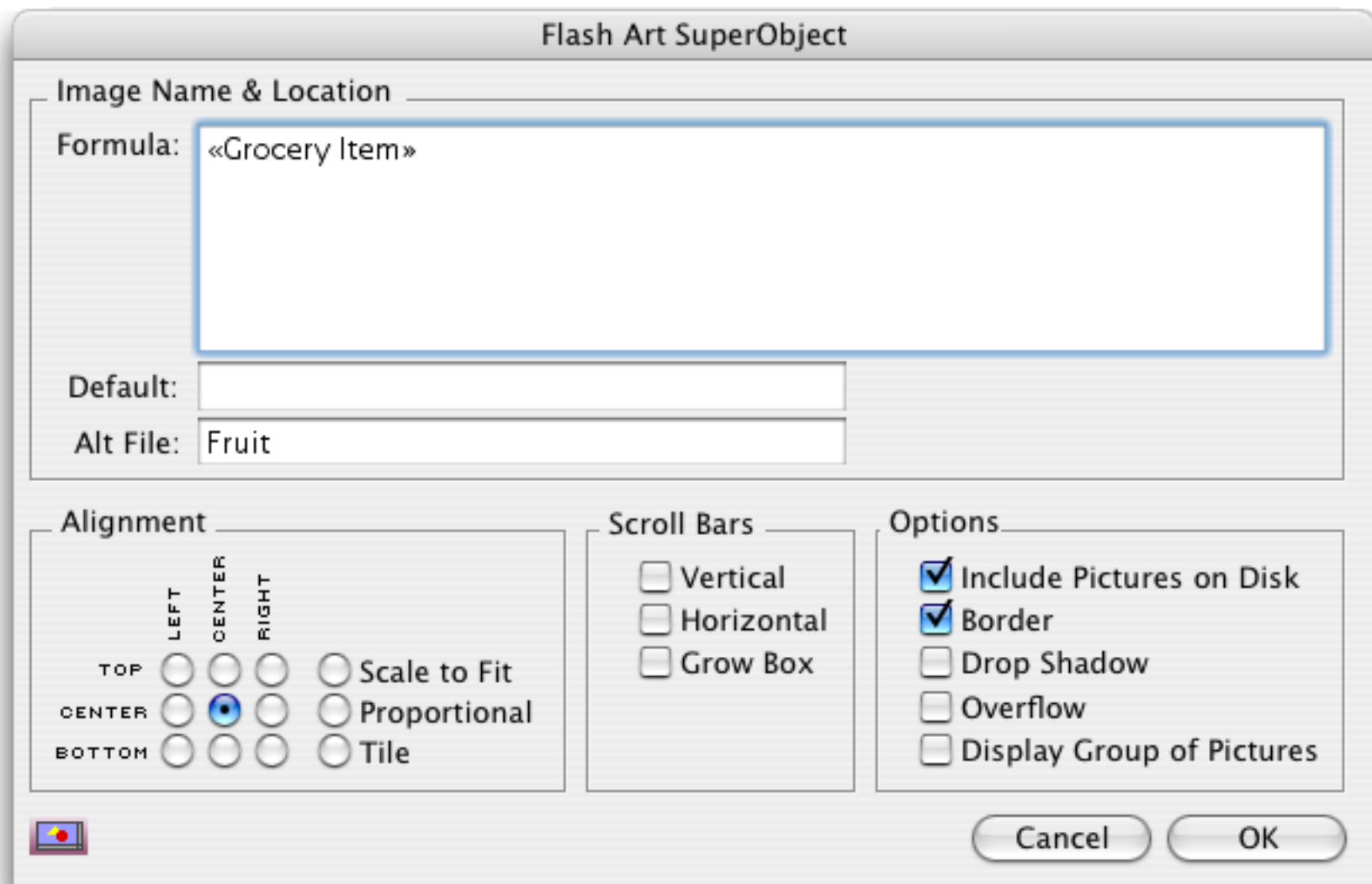


Now if you add a new fruit to the database for which there is no picture, the default image, [Exotic Fruit](#), will be displayed.





For example, suppose you had a **Grocery** database that was always used with the **Fruit** database created earlier. A flash art object in the **Grocery** database can display fruit images in the **Fruit** database, as long as the **Fruit** database is open.



Even if the database containing the images is in a different folder, you should only type in the name of the database. Since the database must be open in memory, Panorama doesn't need to know the location of the database on the disk.

#### **Include Pictures on Disk**

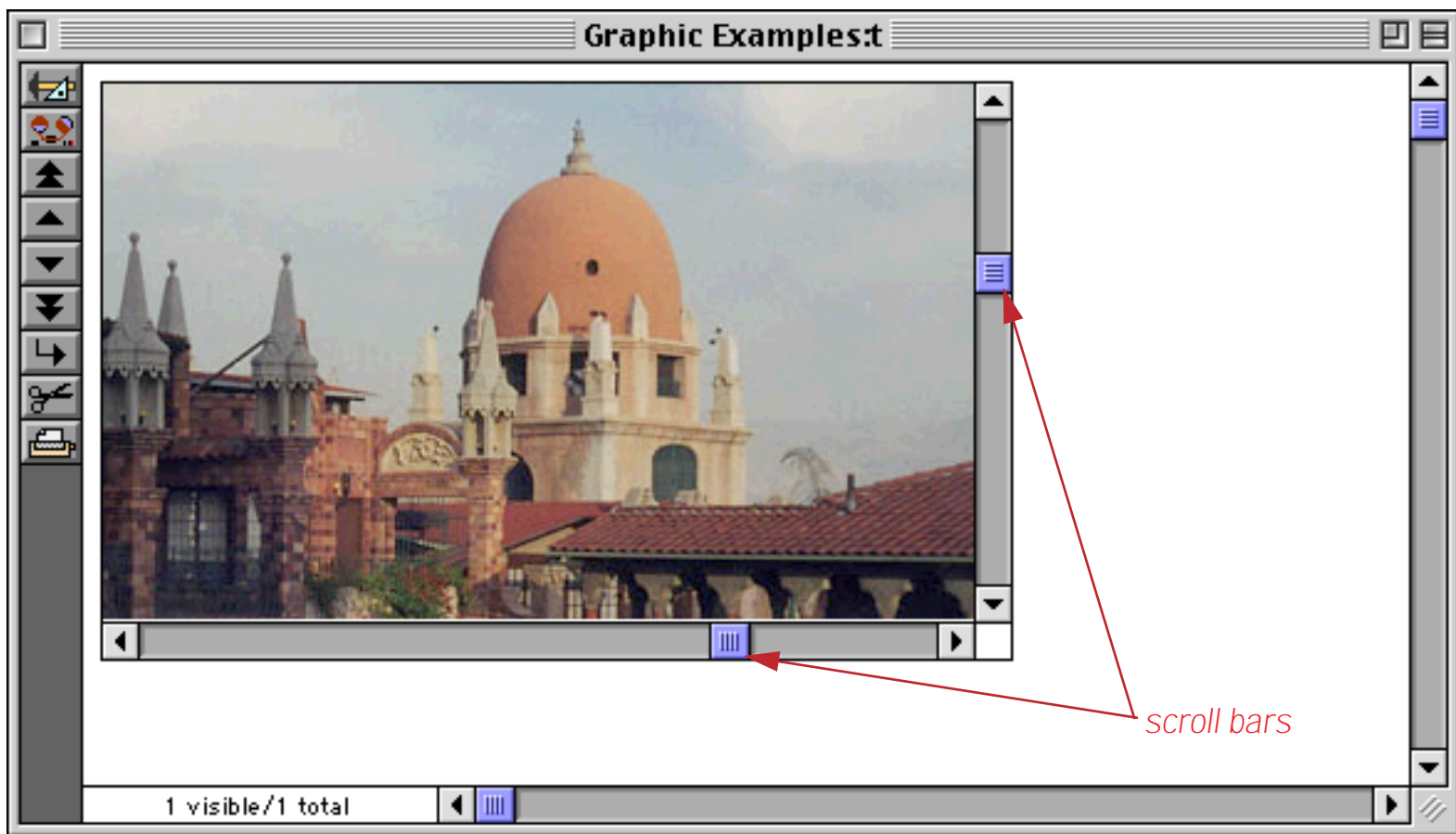
Check this option if you want images in separate disk files to be displayed (the normal setting).

#### **Border**

If this option is enabled, Panorama will draw a 1 pixel border around the Super Flash Art object.

### Scroll Bars

If you expect that the images you will be displaying may be too large to fit into the dimensions of the Super Flash Art object, you can enable scroll bars that will allow the user to shift around and view different parts of the picture. You can enable a Vertical scroll bar, a Horizontal scroll bar, or both.



You also have the option of leaving space for a Grow Box in the lower right hand corner of the Super Flash Art object. The Grow Box is simply a 16 by 16 pixel box that is left empty. It's up to you to draw an icon in this area (if you want). If the bottom right hand corner of the Super Flash Art object is in the same position as the bottom right hand corner of the window, you can disable the window's normal scroll bars and use the Super Flash Art object's scroll bars and Grow Box instead (see "[Elastic Forms](#)" on page 418).

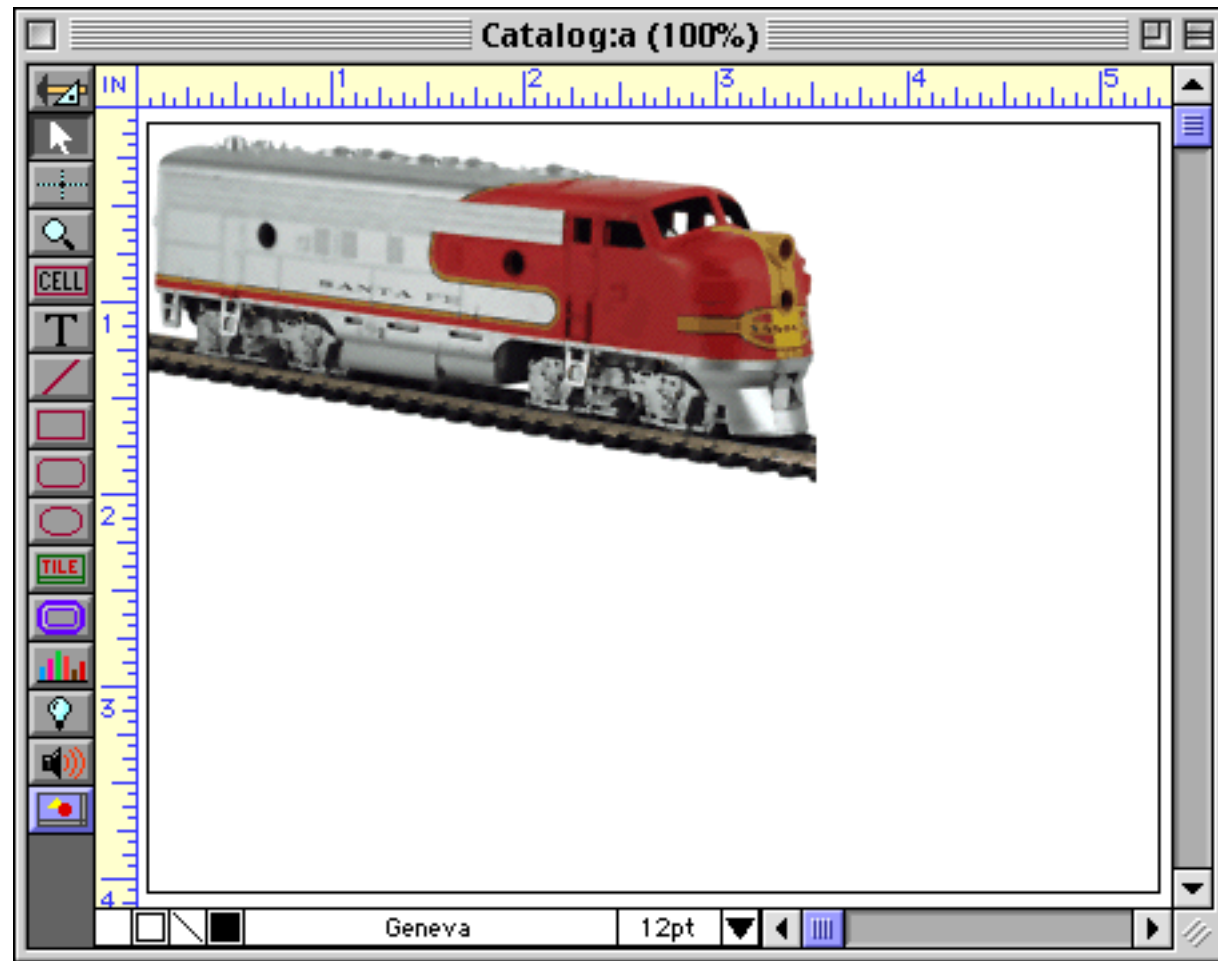
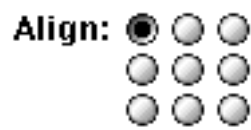
### Align

The dimensions of the picture being displayed often do not exactly match the size of the Super Flash Art object. This section of the dialog specifies how the picture should be adjusted if it is too large or too small for the Super Flash Art object.

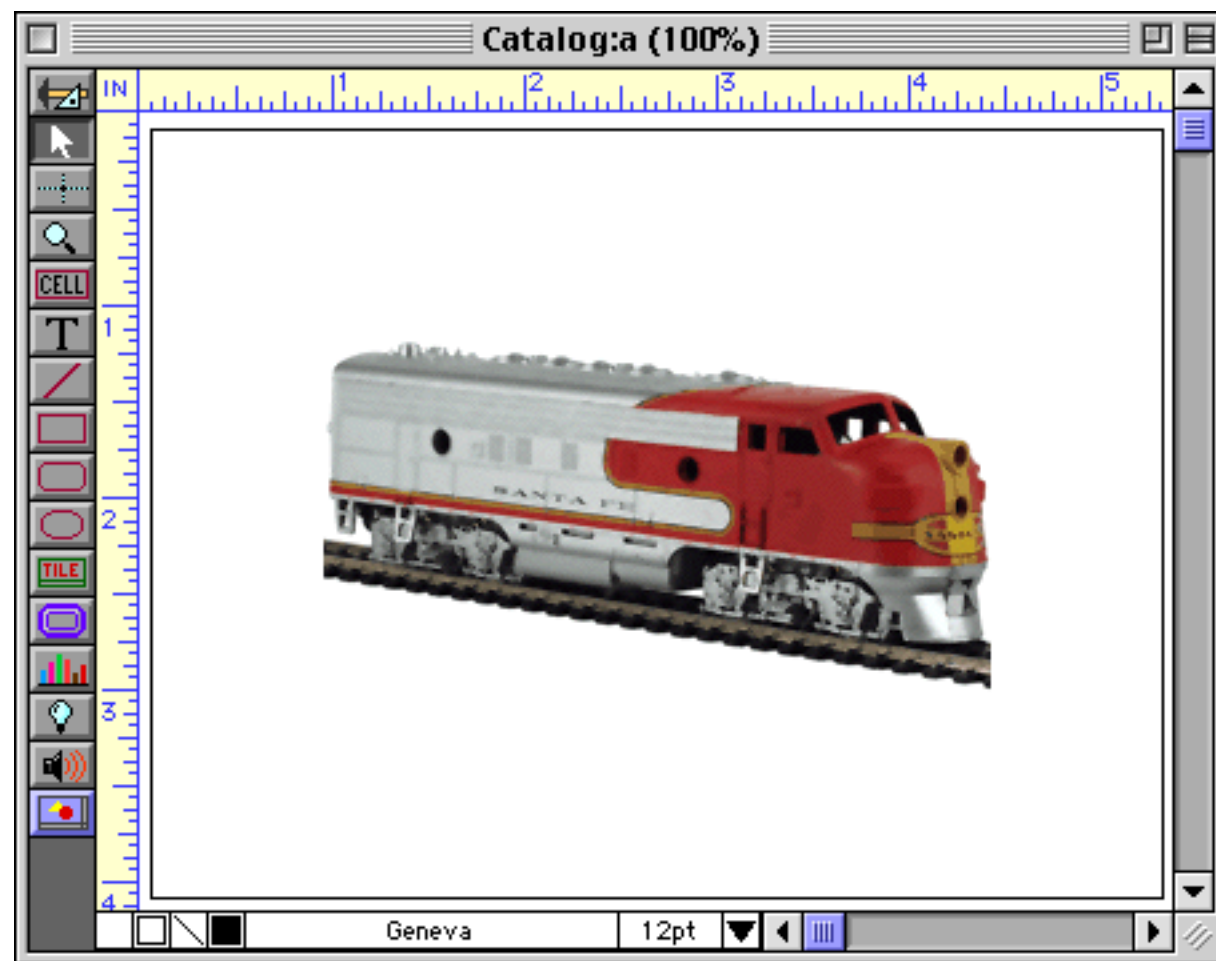
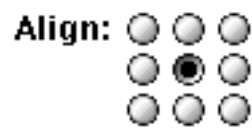
Align:

Scale to Fit  
 Proportional  
 Tile

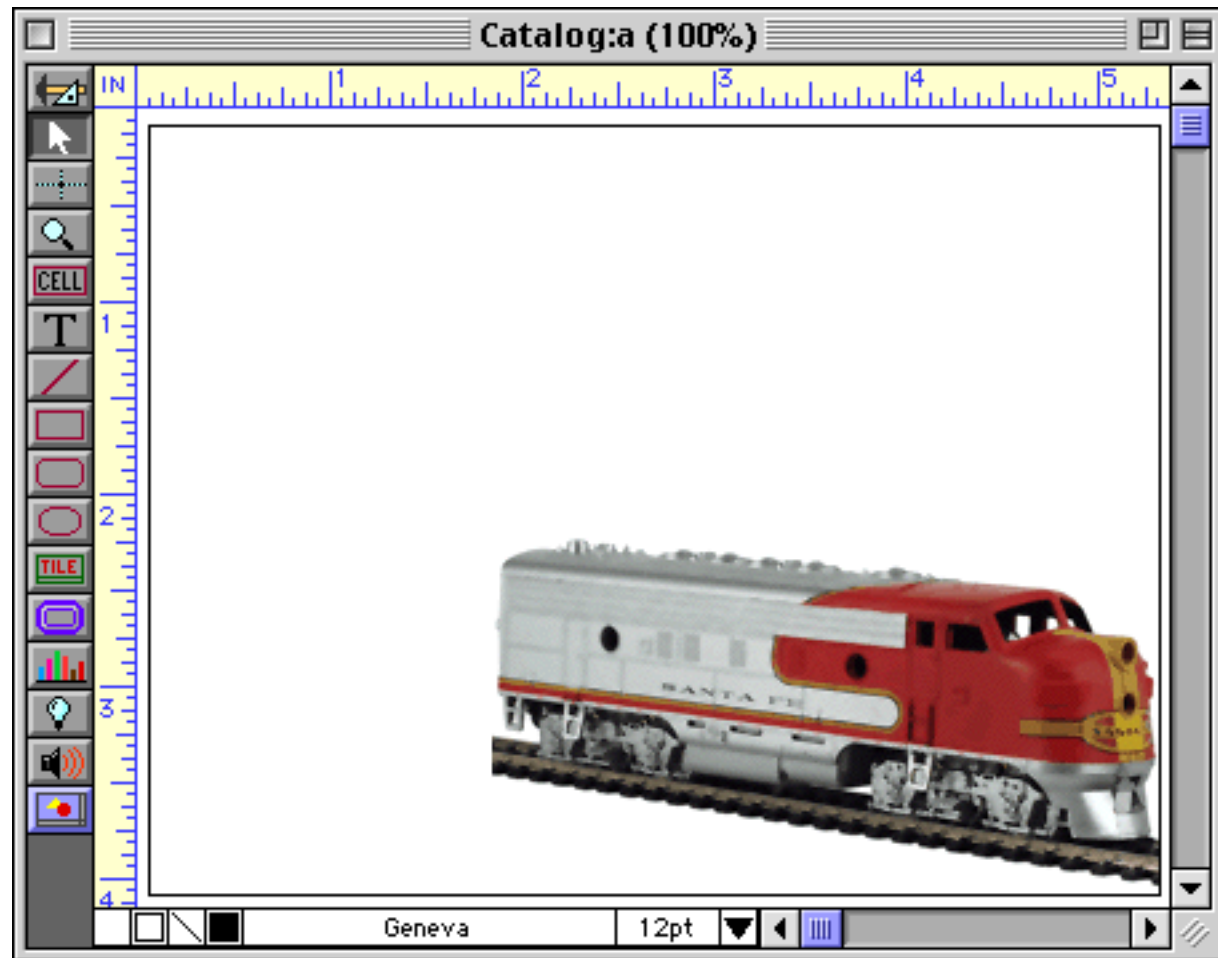
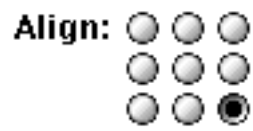
The left side of this section contains nine radio buttons in a tic-tac-toe arrangement. These buttons allow the picture to be aligned within the Super Flash Art object. The picture can be aligned with any corner, centered on any side, or centered in the middle of the object. Wherever the object is aligned, it will be displayed actual size; i.e. it will not be enlarged or reduced. For example, here is an image displayed in the top left corner.



Here is the same image displayed in the middle center.

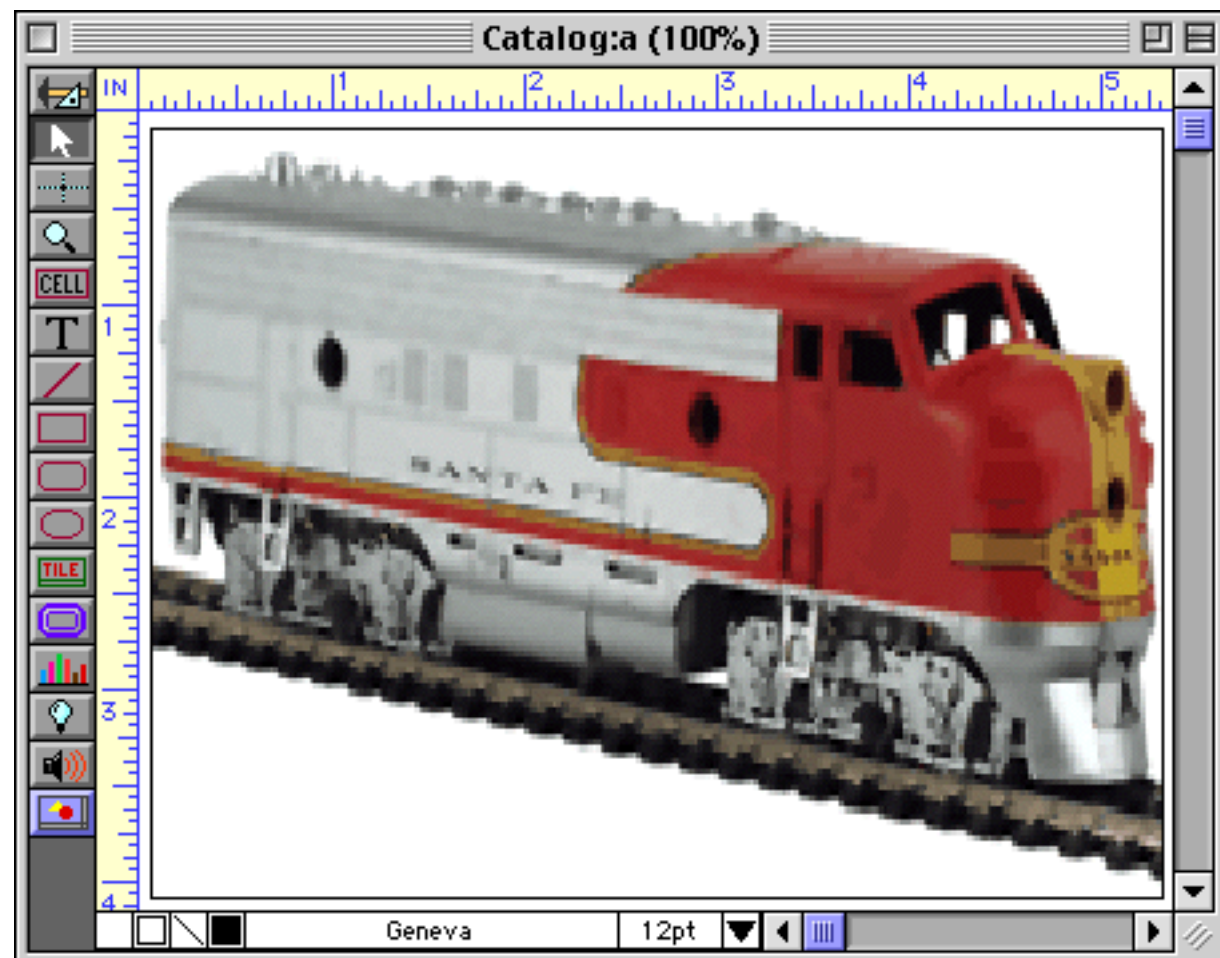


And again in the bottom right.



The **Scale to Fit** option will enlarge, reduce, and/or stretch the picture so that it exactly fits in the Super Flash Art object. The picture may be distorted to make it fit, as you can see in this illustration.

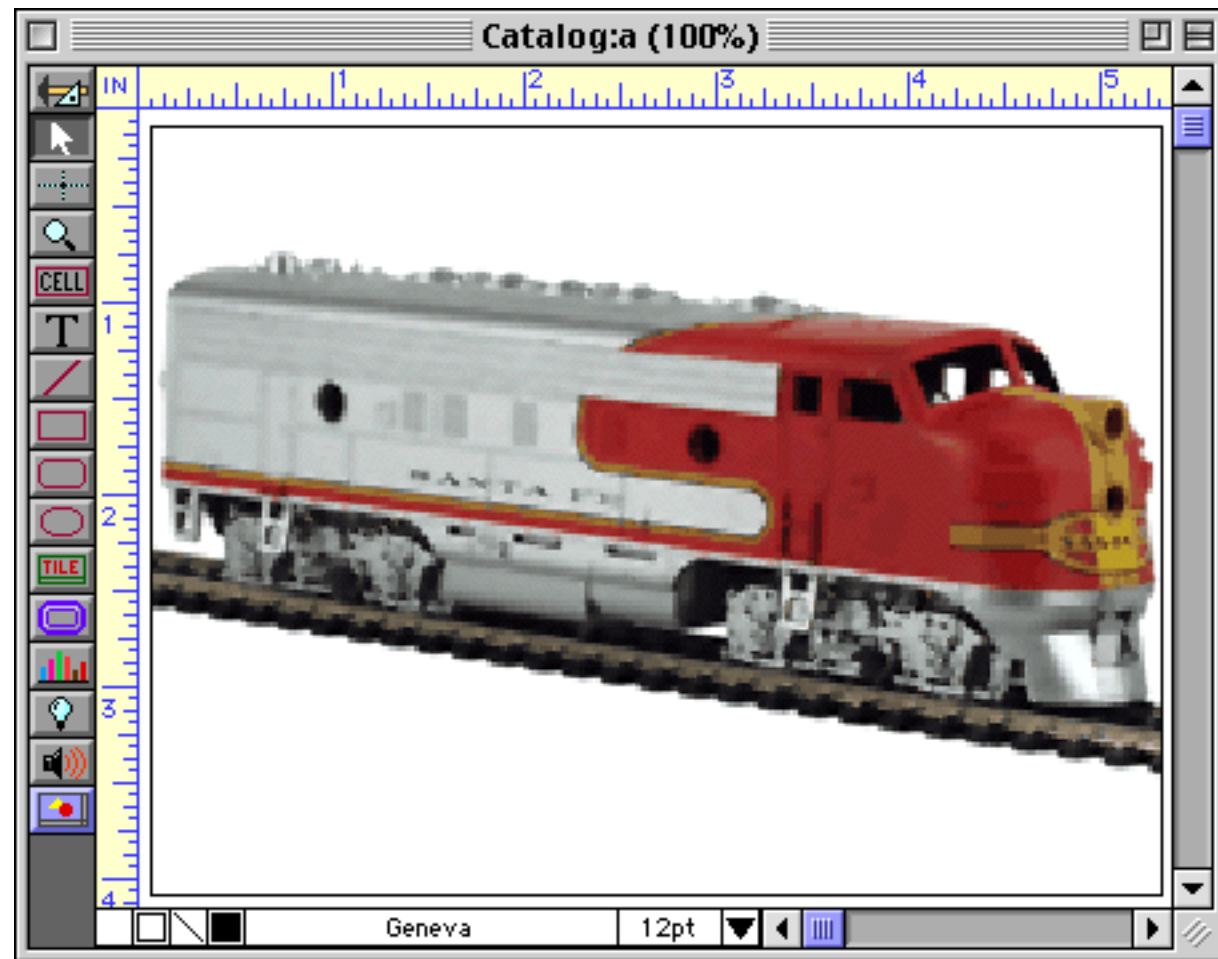
- Scale to Fit**
- Proportional**
- Tile**



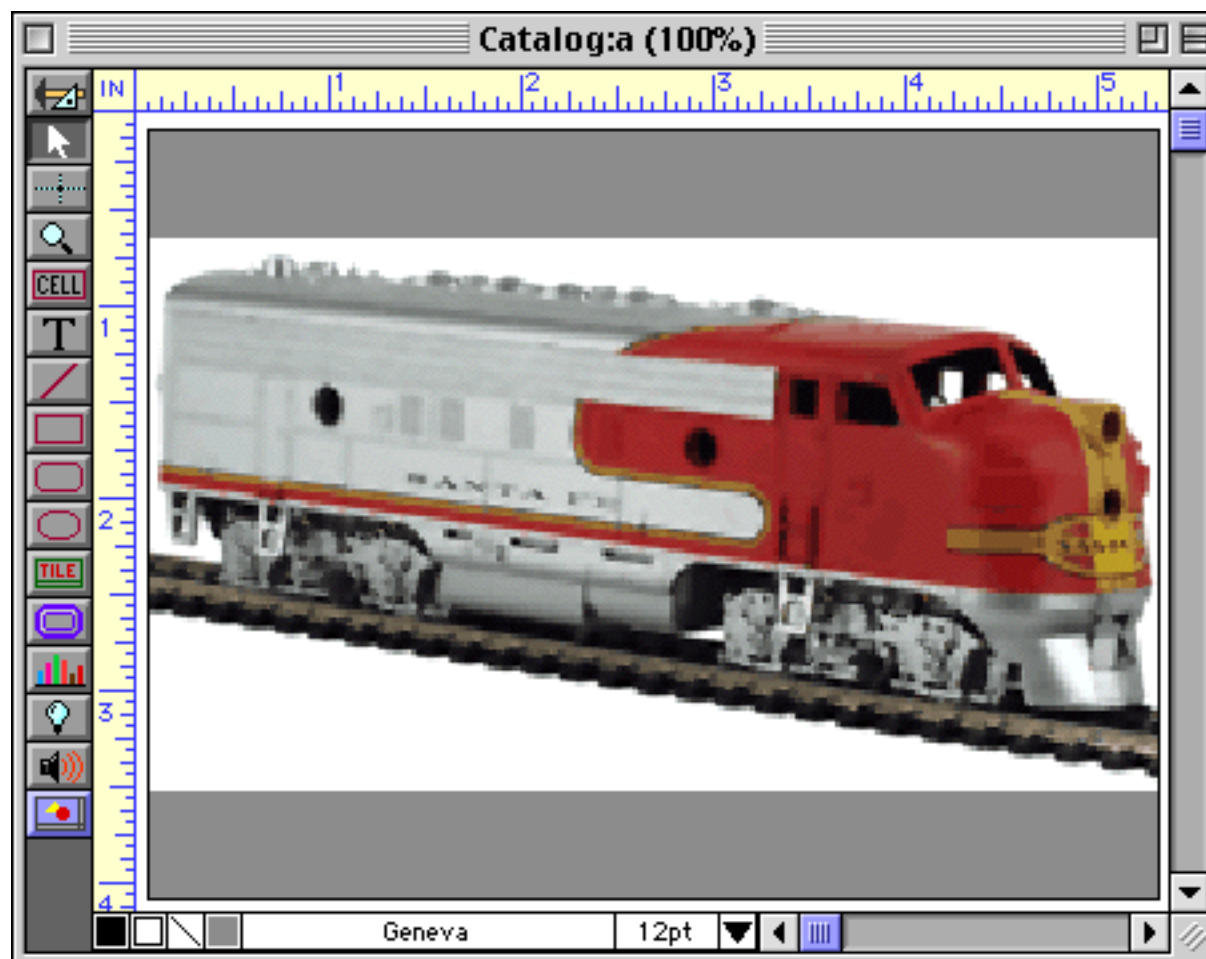


The **Proportional** option will enlarge or reduce the object as much as possible so that it will fit into the object, but it will not distort the picture. In other words, it will not change the proportions of the picture.

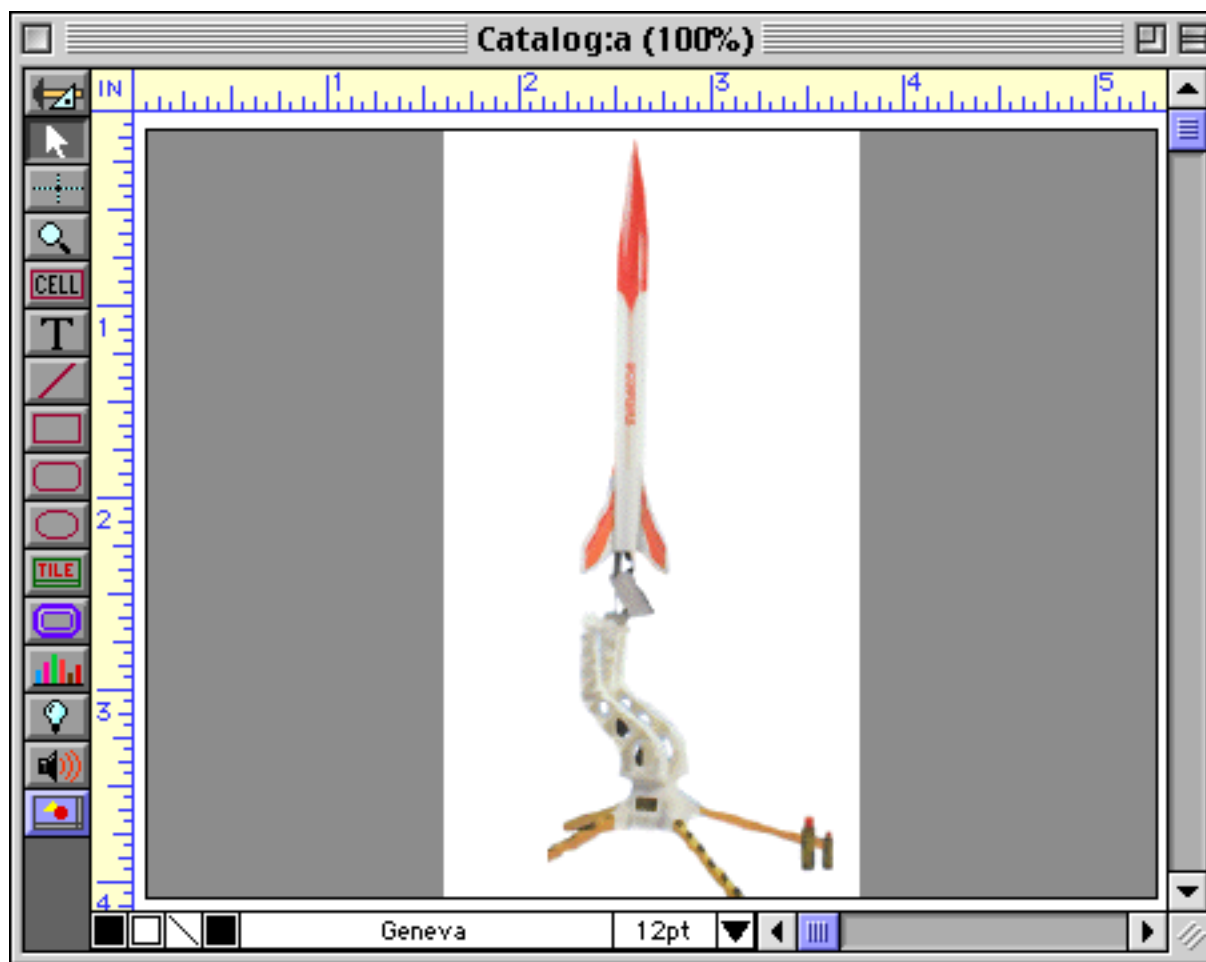
- Scale to Fit
- Proportional
- Tile



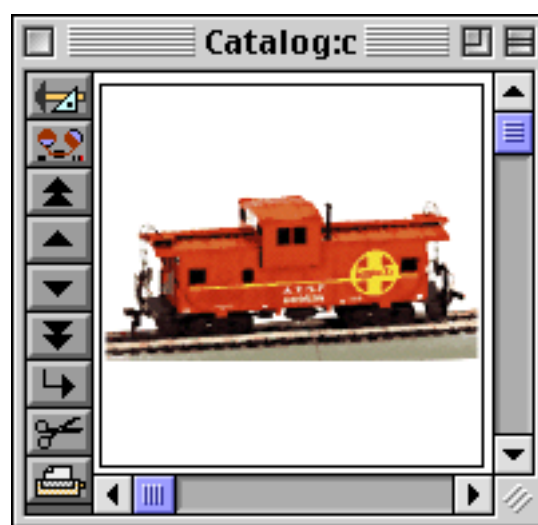
If the proportions of the original picture do not match the proportions of the Super Flash Art object, Panoramica will leave a border along the top and bottom or left and right. We've added a gray background to this example so you can clearly see the borders. (The gray background is simply a solid rectangle object placed behind the Flash Art SuperObject.)



Depending on the aspect ration of the image (width vs. height) the borders may also be on the sides instead of the top and bottom, like this.

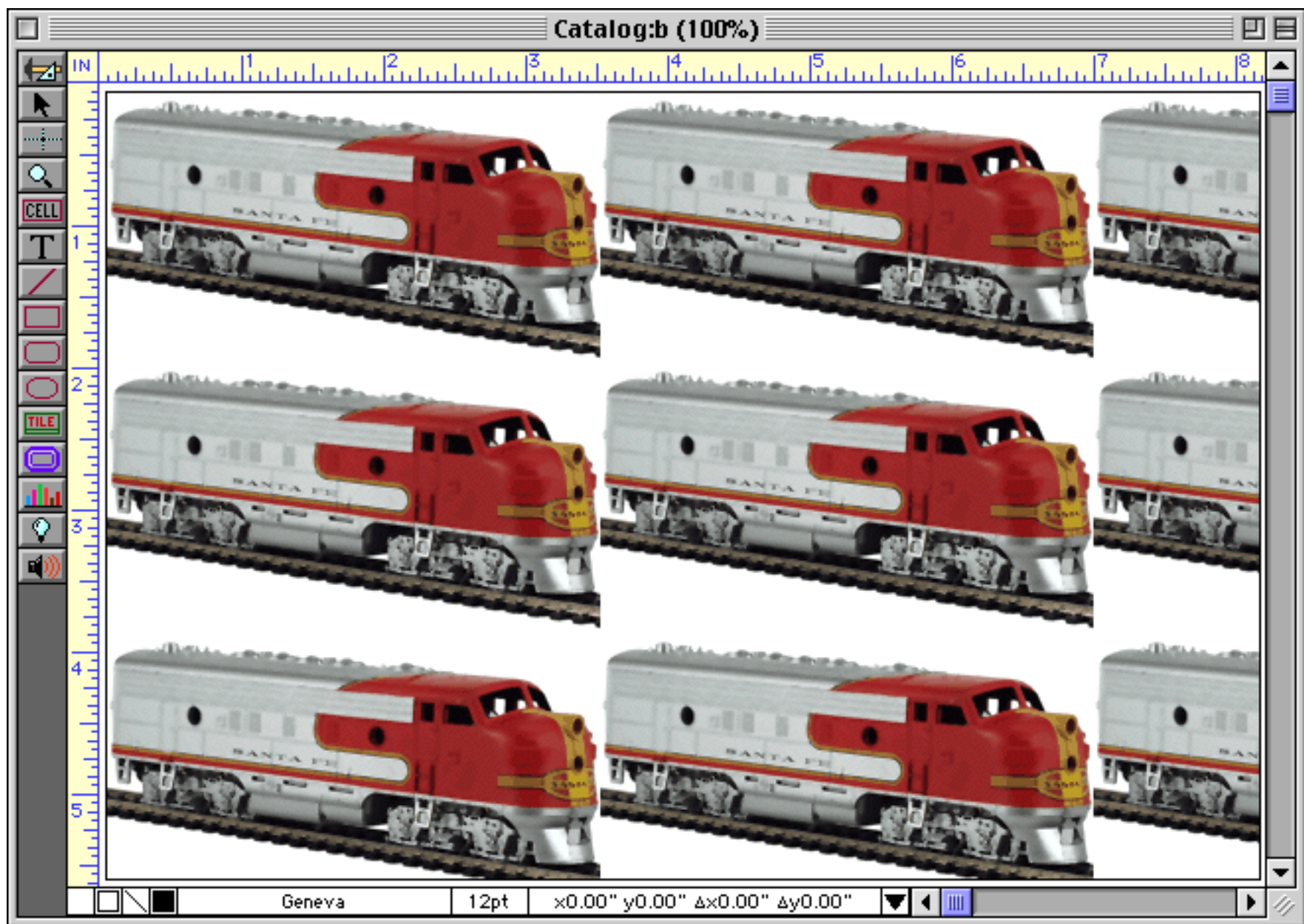


Keep in mind that the **Scale to Fit** and **Proportional** options may be used to reduce an image as well as to enlarge it. Here's a typical reduced image that has been scaled down with the **Proportional** option.





The **Tile** option displays the picture over and over again in a tile arrangement, starting from the upper left. This option allows you to cover a large area with a small picture.

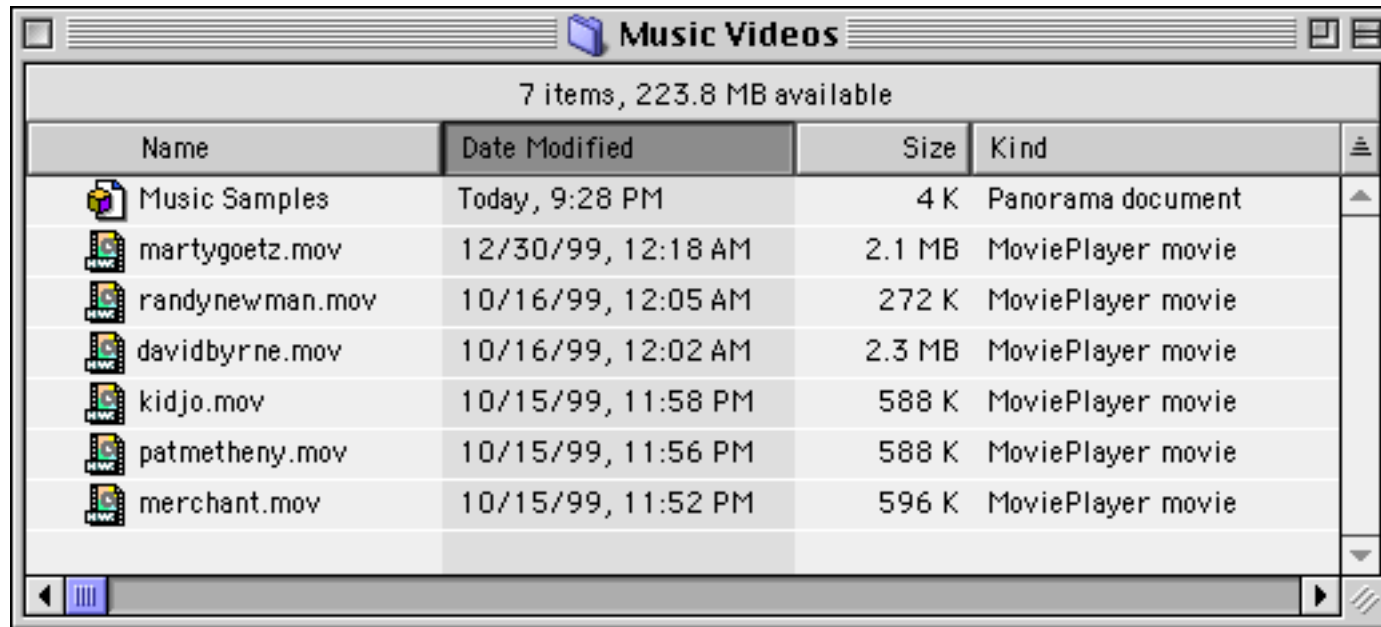


The only disadvantage of this technique is that if the original picture is really small there will be a perceptible delay as the tile pattern is drawn.

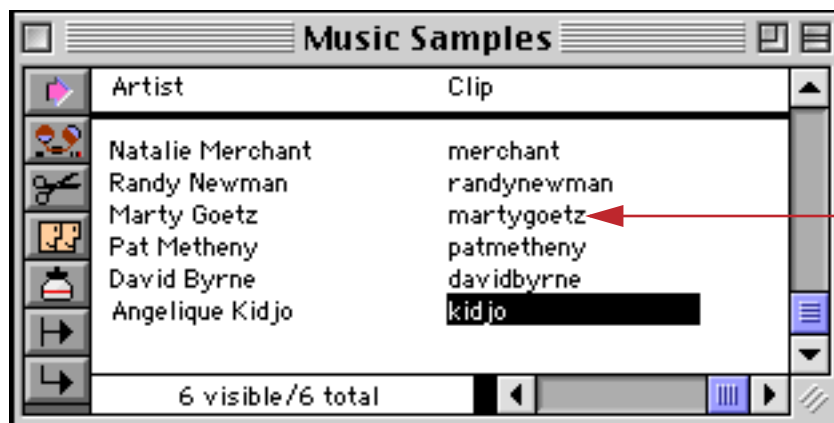
## Displaying Movies in a Form

Panorama's Super Flash Art object is capable of displaying movies as well as still images. To do this you must have Apple's QuickTime software installed on your computer. If you don't have this software you can download it from [www.apple.com](http://www.apple.com) for either Macintosh or Windows PC systems.

The first step in displaying movies is to create some movie files.

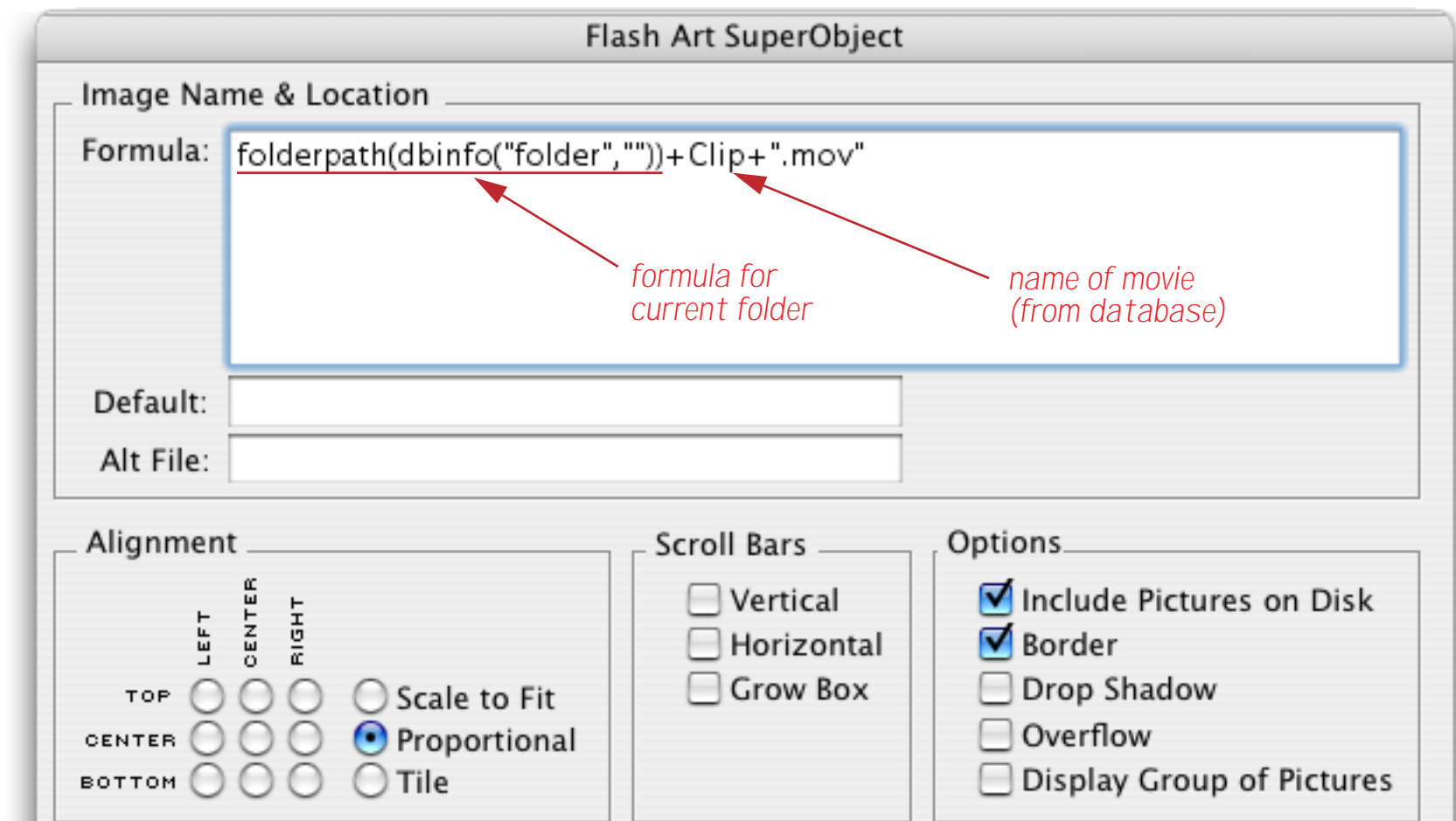


Next, you'll need to create a database. In this case we created a very simple database that has the artist's name along with the name of the movie (we'll add the `.mov` extension in a moment).

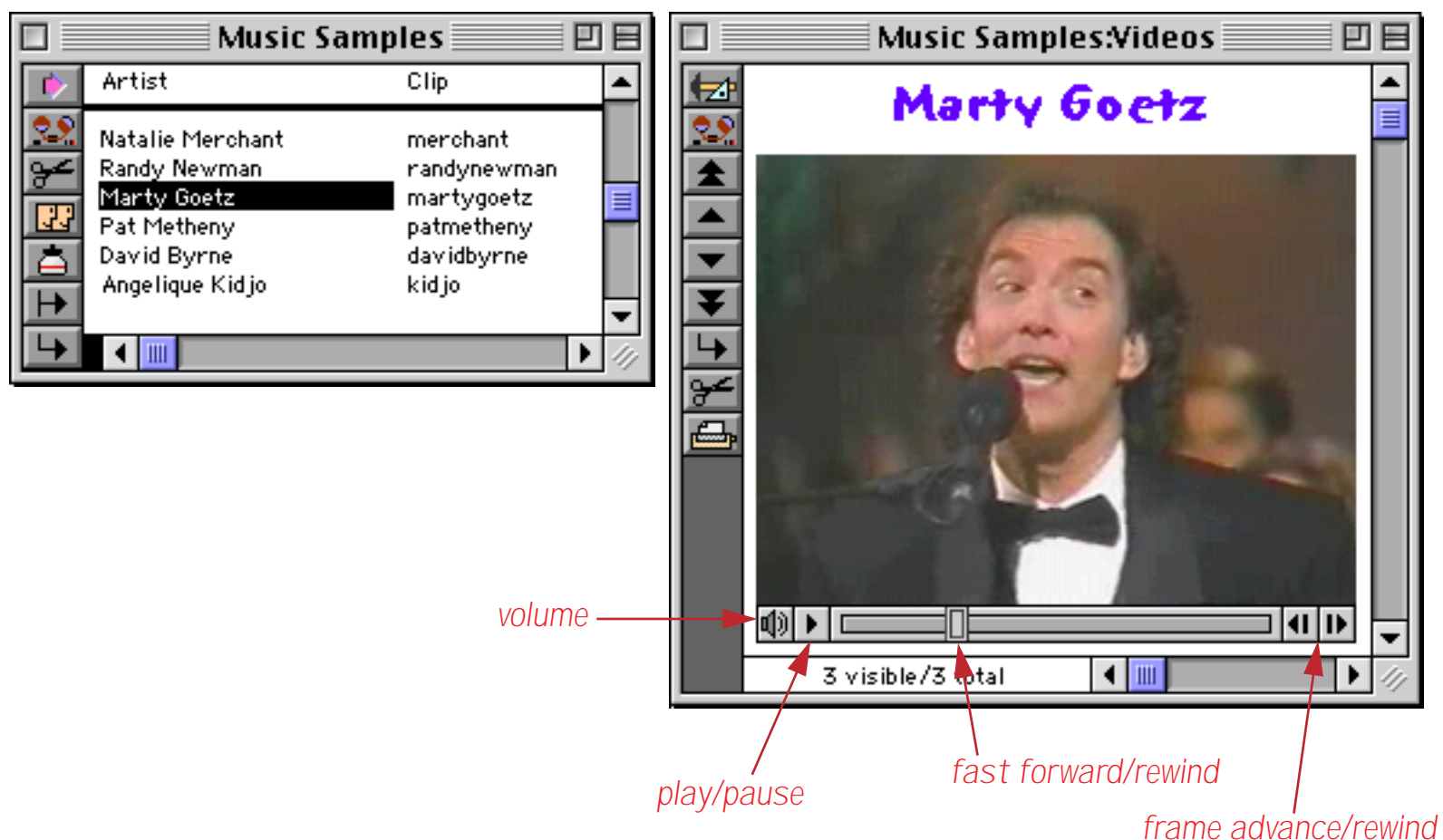


*same as movie names above*

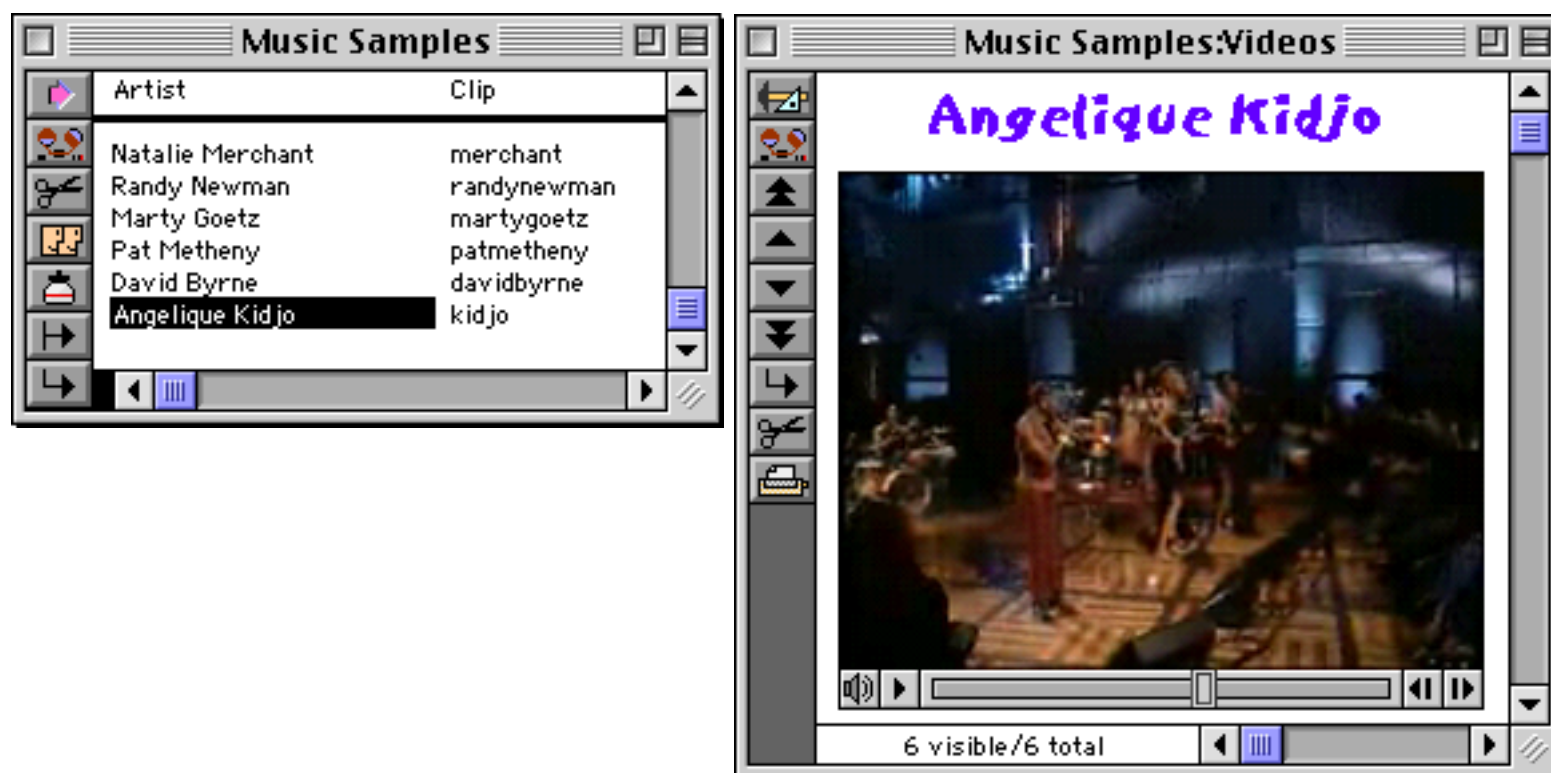
To display the movies we'll need to create a form, and then to create a Super Flash Art object within the form (see "[Creating Super Flash Art Objects](#)" on page 321). Unlike still images, movies always require you to specify the complete location of the file. The formula below shows how to do that (assuming that the movies are in the same folder as the database) and also adds the `.mov` extension. Make sure that the **Include Pictures on Disk** option is enabled, because a movie cannot be pasted into the Flash Art Scrapbook.



Switch to Data Access Mode to try out the movie. The normal QuickTime controls appear at the bottom of the Super Flash Art object.



To see a different movie simply move to the record for the movie you want to view.



In Graphics Mode editing a Super Flash Art object that is set up to display a movie can be a bit tricky. Even though you are in Graphics Mode, clicking on the movie tends to activate the movie controls instead of selecting the object. If you have difficulty, try dragging a marquee around the object instead of clicking on it. Instead of double clicking on the object to open the configuration dialog, try using the **Object Properties** command in the Edit menu.

Once a movie has been set up it can be controlled with a procedure as well as with the movie control strip. The procedure can start and stop the movie, move to a specific spot within the movie, change the playback rate and the volume. To learn how to program a movie see "[Super Flash Art Commands \(Including Movie Control\)](#)" on page 1838 of the **Panorama Handbook**.

For those of you that can't stand not knowing who Marty Goetz and Angelique Kidjo are, check them out at <http://www.martygoetz.com> (Marty Goetz) and at <http://wwwusers.imagnet.fr/~kidjo/home.html> (Angelique Kidjo).





# Chapter 17: Buttons & Widgets



The Industrial Revolution introduced machinery with all kinds of knobs, levers, and doo-dads. In our electronic age these have been replaced by virtual push buttons, checkboxes, “radio” buttons, pop-up menus, scrolling lists, and other widgets. Panorama has a number of tools for incorporating these types of controls into your forms.

Although most of the buttons and widgets discussed in this chapter can be used without programming, most of them are often combined with Panorama procedures. Because of this, we will often reference procedures and programming techniques throughout this chapter. Before reading this chapter you may want to skip ahead and learn how to create and work with procedures (see “[Procedures](#)” on page 1338) and variables (see “[Variables](#)” on page 1189).

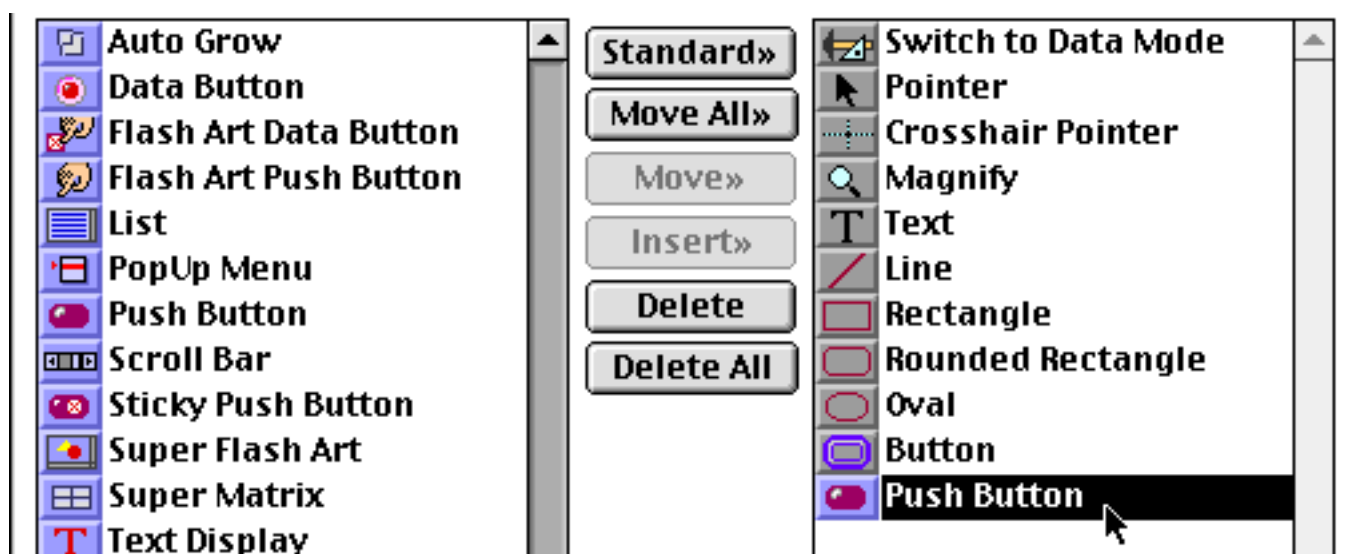
## Push Buttons

Push buttons have one mission in life—to start something. In Panorama, clicking on a push button triggers a procedure (a program). You push the button and the program starts, simple as that. Panorama has three different types of button objects—Super Object Push Buttons (next section), “Classic” Button Objects and Flash Art Push Buttons. Only the first type are discussed here, to learn about the other two see Chapter 17 of the [Panorama Handbook](#).

### Super Object Push Button

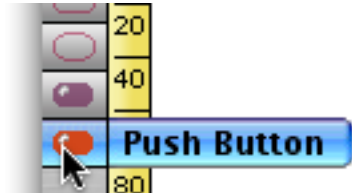
The SuperObject Push Button makes it easy to create attractive buttons that trigger procedures. The SuperObject Push Button can create circular or oval buttons, and buttons with various 3-D effects.

The SuperObject Push Button tool is not in the default tool palette, so you’ll need to use the **Tool Palette** dialog to add this tool to the palette if it is not already there.

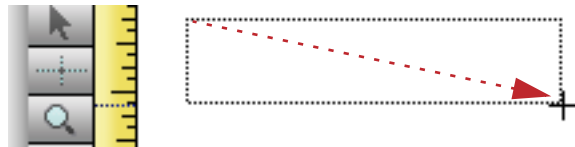




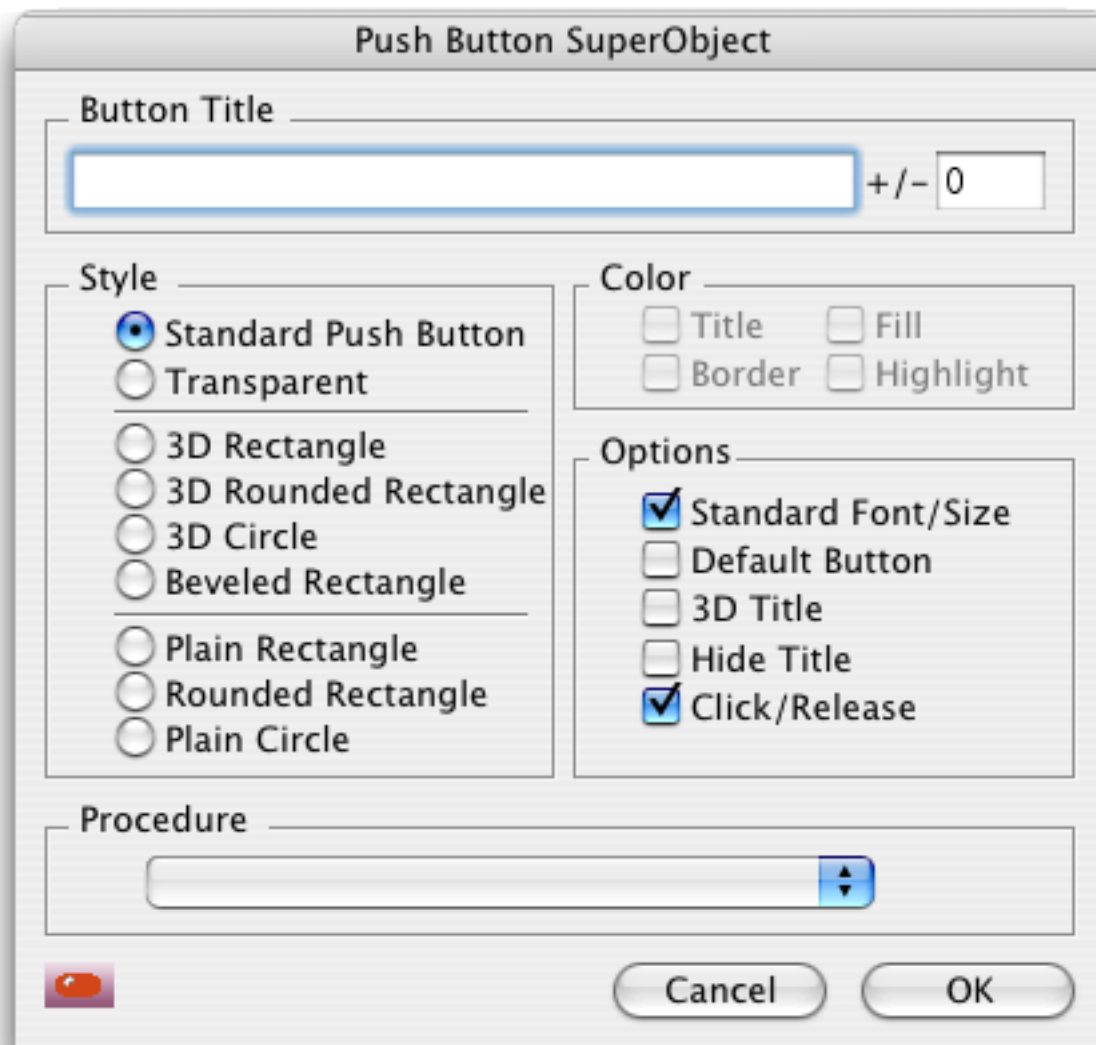
Now that the tool is added to the palette you can select it.



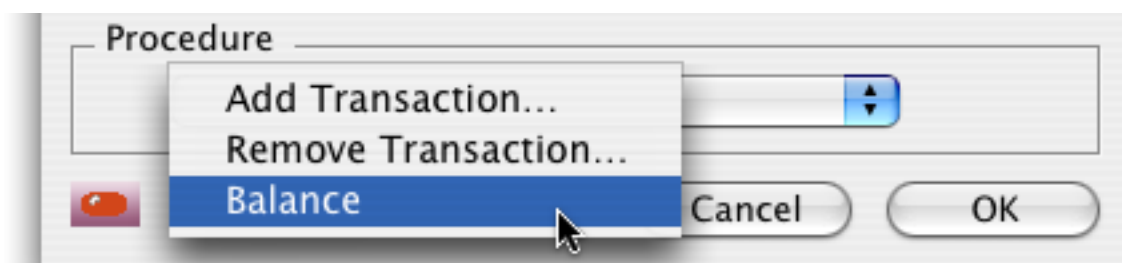
Once the tool is selected, drag the mouse across the form in the location where you want to create the button.



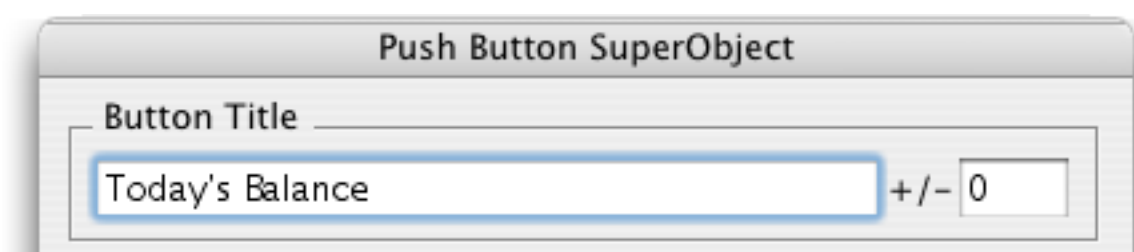
When you release the mouse, the SuperObject Push Button configuration dialog will appear.



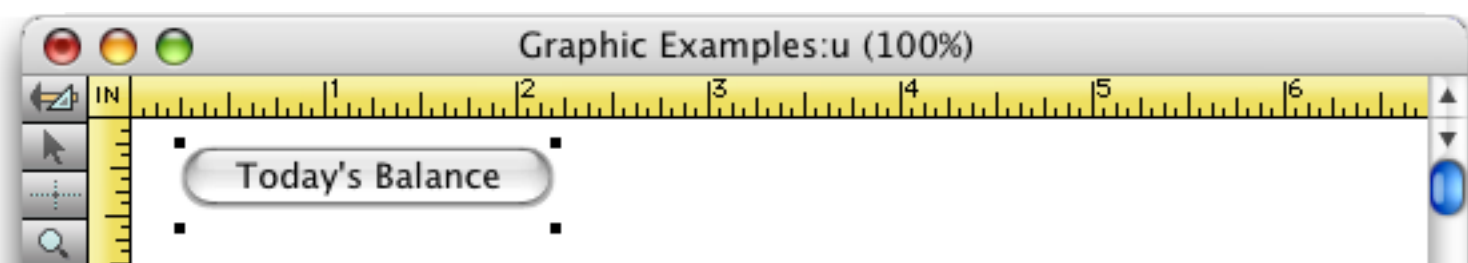
Use the pop-up menu to select the procedure that will be triggered by this button. The button can trigger any procedure in the current database. (If you haven't created the procedure yet you can still create the button, then go back later and choose the procedure.)



Usually you'll want to set up a title for the button, and a style. The title can be the same as the procedure name or it can be different.



When you press the **OK** button the new button will appear in your form.

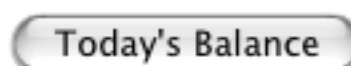


You can modify the appearance of the title with the **Font** and **Size** menus (see “[Font](#)” on page 507 and “[Text Size](#)” on page 509). To re-open the configuration dialog you can either double click on the button or select the button and open the dialog with the **Object Properties** command in the Edit menu.

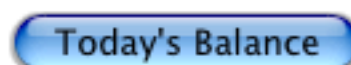
#### Push Button Styles

There are over a dozen controls for changing the push button appearance. The primary control is the button style.

**Standard Push Button:** This option displays a standard button for the current platform. The appearance of the button will vary depending on the operating system you are using: Windows, OS X, or OS 9. The example below is for OS X.



The appearance of a **Standard Push Button** can be modified using the **Default Button** option. When this option is used the button appearance changes to indicate that this is the default button (the button in a dialog that is activated automatically when the Enter key is pressed).

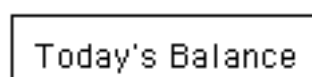


Remember, the **Default Button** option only changes the appearance of the button. You still have to program the button to make it the default (see “[Custom Dialogs](#)” on page 1639).

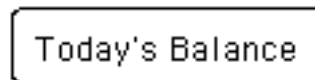
**Transparent:** This option displays a transparent button. A transparent button will be completely invisible except for the title (and you can make that invisible too with the **Hide Title** option).

Today's Balance

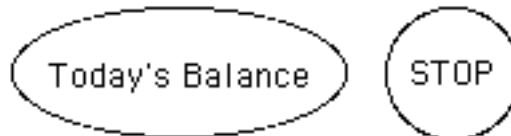
**Rectangle:** This button style has square corners.



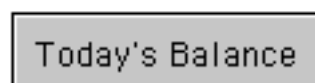
**Rounded Rectangle:** This is a standard two dimensional button. Panorama will display the name of the button inside the button. If you want this to look like a standard Macintosh button, you should use Chicago 12 point type.



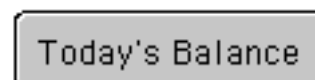
**Circle/Oval:** This style has an oval border. If the button's dimensions are square the button will be a circle.



**3D Rectangle:** This style looks like a 3 dimensional rectangle with beveled edges.



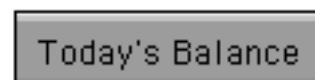
**3D Rounded:** This style looks like a 3 dimensional rounded rectangle with beveled edges on all four sides.



**3D Circle/Oval:** This style looks like a 3 dimensional oval with beveled edges.



**Beveled Rectangle:** This style looks like a 3 dimensional rectangle that is beveled on the top and bottom. This style looks a lot like the buttons on many VCR's and stereos.



Note: On black and white (b/w) monitors, Panorama automatically converts the 3D buttons to the corresponding 2D button.

### Button Title

The button title is the title that appears on the button. This title will be centered in the middle of the button. The procedure that is triggered by the button can find out what button was pressed with the `info("trigger")` function. This function will return `Button.` followed by the title of the button. Using this function you can have several buttons that trigger a single procedure. The procedure can then use the `info("trigger")` function to find out which button was actually pressed. The procedure will usually use `if` or `case` statements to decide what button was pressed, like this.

```
local actionDate
case info("trigger") = "Button.Mon" actionDate=date("Monday")
case info("trigger") = "Button.Tue" actionDate=date("Tuesday")
case info("trigger") = "Button.Wed" actionDate=date("Wednesday")
case info("trigger") = "Button.Thu" actionDate=date("Thursday")
case info("trigger") = "Button.Fri" actionDate=date("Friday")
endcase
select Date=actionDate
```

### Title Positioning

Panorama attempts to center the title both vertically and horizontally within the SuperObject push button. However, some fonts have non-standard vertical dimensions and need an adjustment to center properly. If necessary, you can adjust the title's vertical position with the +/- option. For example, to move the title up by 2 pixels, enter 2 (or +2) into this option. To move the title down by one pixel enter -1.



### Standard Font/Size

If this option is selected the button title will always be displayed in the operating system's standard font and size. For example, on OS X systems the standard font is 14 point Lucinda Grande.

### 3D Title

If the **3D Title** option is turned on, Panorama will display the title with a white shadow, giving it an "etched" 3D look. Note: This option is ignored if you have use the Standard Push Button option. Also if you display the button on a black and white (b/w) monitor, Panorama ignores this option and displays the title normally.

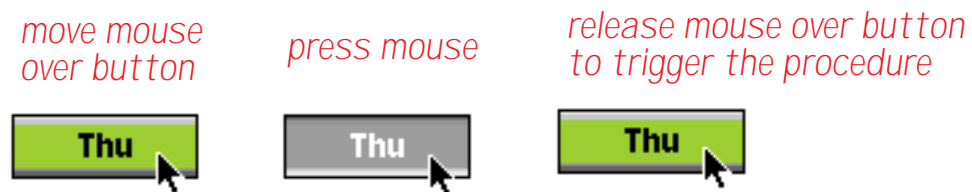


### Hide Title

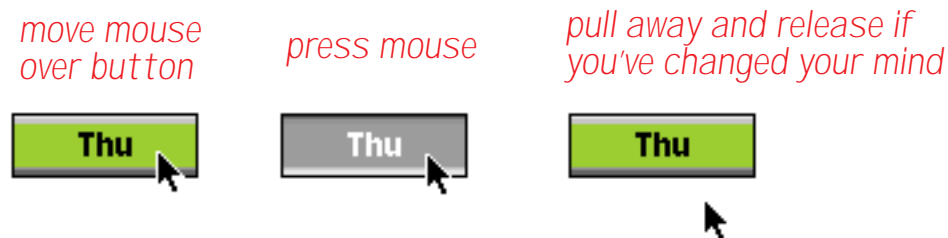
This option hides the title. In other words, the button will be blank, even though it has a title. Use this option if you want to place some graphics on top of the button, or to make a transparent button completely invisible. The title will be invisible so that it will not interfere with the graphics, but the procedure can still find out the title with the `info("trigger")` function.

### Click/Release

Unlike most options, **Click/Release** is enabled by default—you have to turn it off if you don't want it. When this option is enabled, the button acts like a normal button—it highlights when you press on it, then activates (triggers a procedure) when you release the mouse.



If you decide you don't want to activate the button you can pull the mouse away before you release.



When the **Click/Release** option is disabled, there is no highlighting when you press the mouse. Instead, the procedure is triggered immediately as soon as you press the mouse. There's no chance to back out by pulling away from the button.

### Color Options

Like other graphic objects, you can assign any color to a Push Button (see “[Color](#)” on page 504). Unlike most other types of objects, however, you can control what portions of the button are displayed in color. (However, if you are using the **Standard Push Button** style then these color options are not available.) The four options are:

**Title:** If this option is checked, the title will be displayed in color, otherwise the title will be displayed in black.



**Border:** If this option is checked, the border will be displayed in color, otherwise the border will be displayed in black. The option can be used with 2D or 3D buttons, like this.

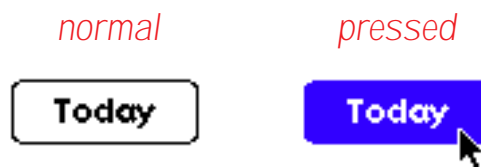


**Fill:** This option only applies to 3D buttons. If the **Fill** option is checked, the body of the button will be displayed in color instead of gray.

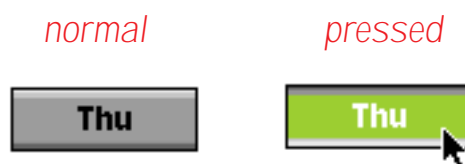


This option only looks good with a few very light colors (use your judgement).

**Highlight:** If this option is checked, the button will highlight in color when you press on it. For 2D buttons, Panorama simply displays the highlight color when the button is pressed (without this option the button will turn black when pressed).



For 3D buttons, Panorama will use the selected color instead of the dark gray it normally uses.



Note: Both 2D and 3D buttons will ignore this option if the **Click/Release** option is not checked.

If you need to make your own custom push buttons you can use the Flash Art Push Button SuperObject. See Chapter 17 of the **Panorama Handbook** to learn about this type of button.



## Data Buttons

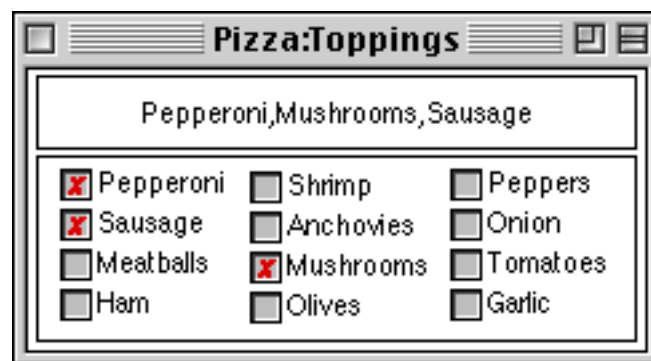
Data buttons are buttons that have a value associated with them. The value may be stored in a database field or in a variable (see “[Variables](#)” on page 1189 and “[Variables](#)” on page 1370). An on/off data button is usually called a **checkbox**.

Taxable

Data buttons may be grouped together as **radio buttons**. Only one button in the group may be selected at a time.

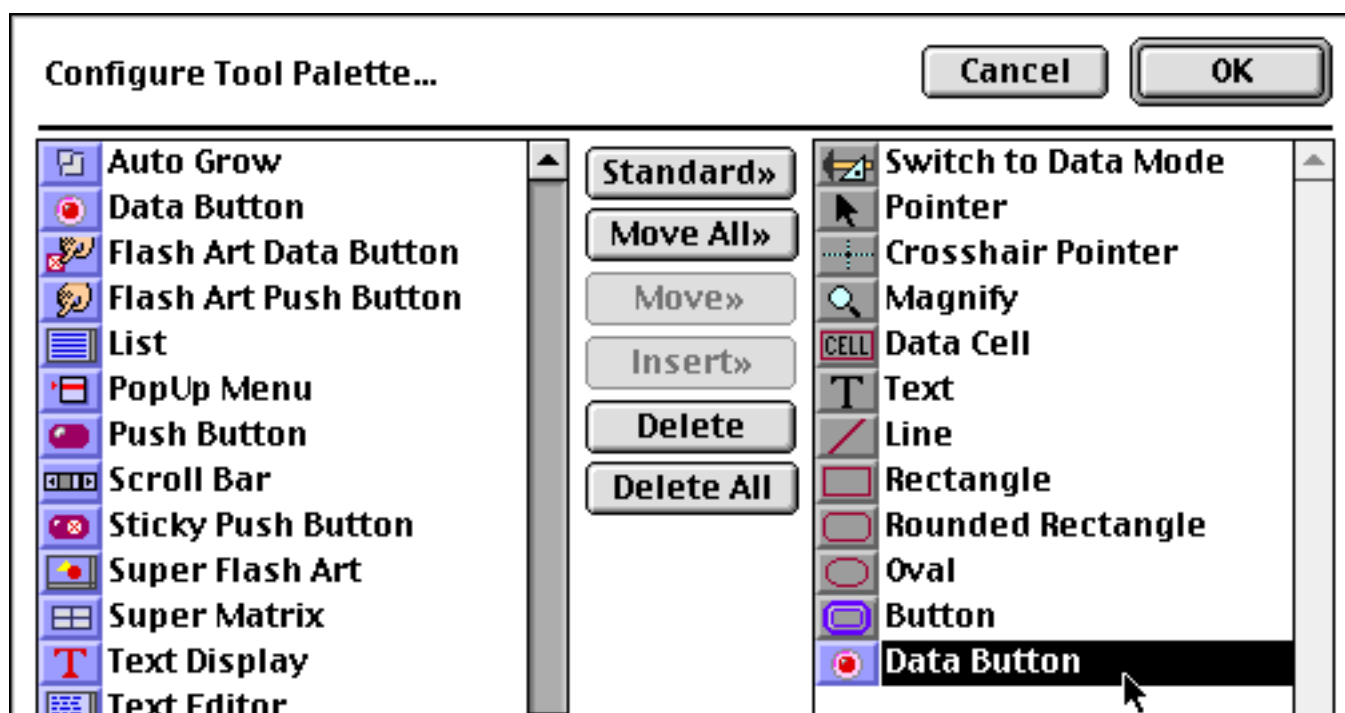
Priority Mail  
 UPS Next Day Air  
 Federal Express  
 Airborne  
 DHL

Panorama also allows you to build a group of buttons where more than one member of the group can be selected at a time. See “[Multiple Value Button Groups](#)” on page 365 to learn more about this option.

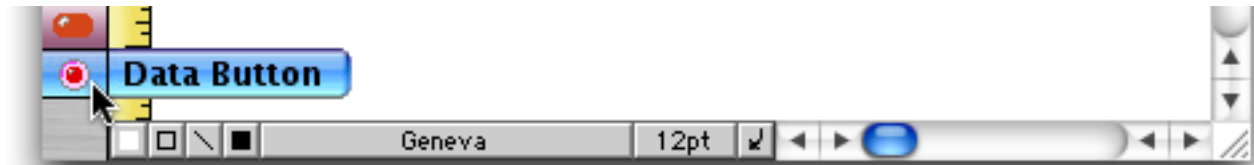


## Data Button SuperObjects™

The Data Button SuperObject™ tool can be used to create checkboxes and radio buttons in a variety of styles. Unlike the “classic” button tool, this tool can work with global variables as well as fields. The SuperObject Data Button tool is not in the default tool palette, so you’ll need to move the use the Tool Palette dialog to add this tool to the palette if it is not already there.



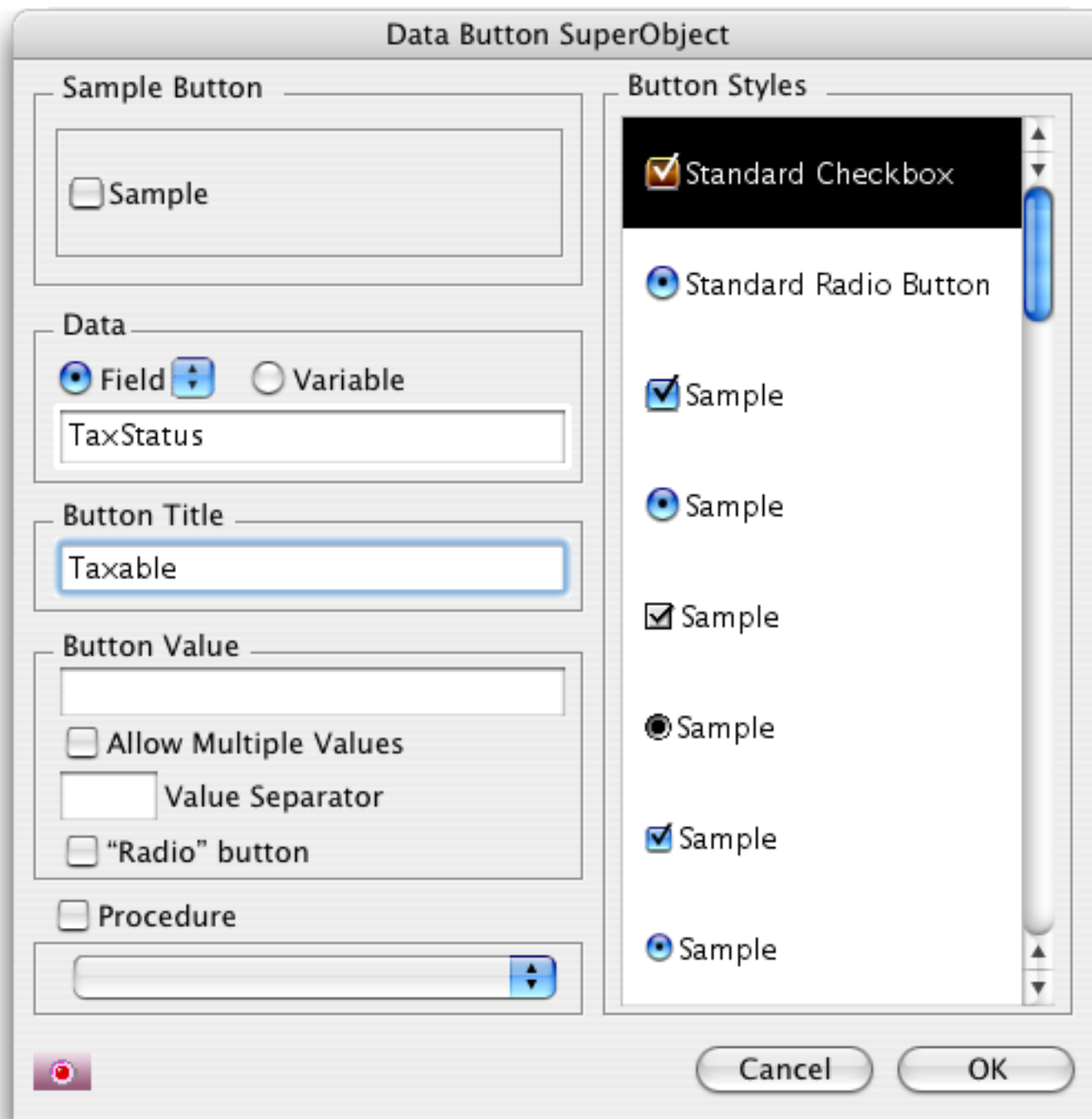
Now that the tool is added to the palette you can select it.



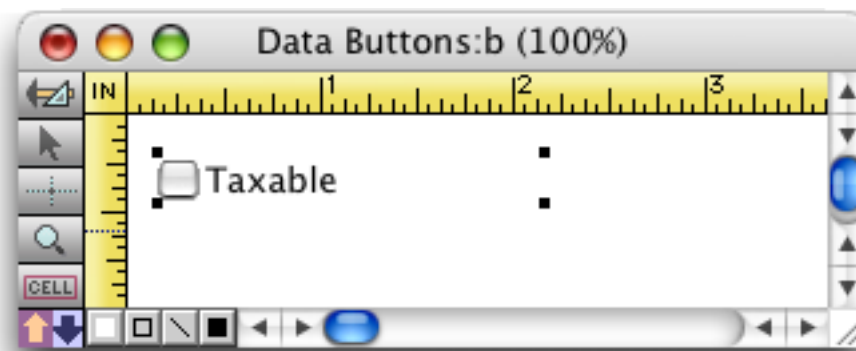
Once the tool is selected, drag the mouse across the form in the location where you want to create the button.



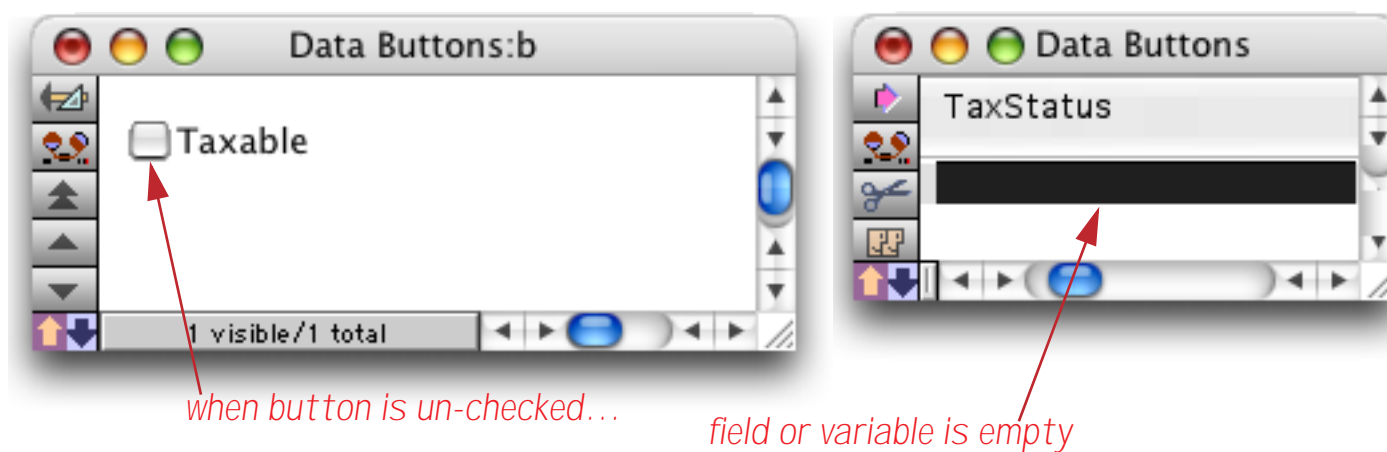
When you release the mouse, the SuperObject Data Button configuration dialog will appear.



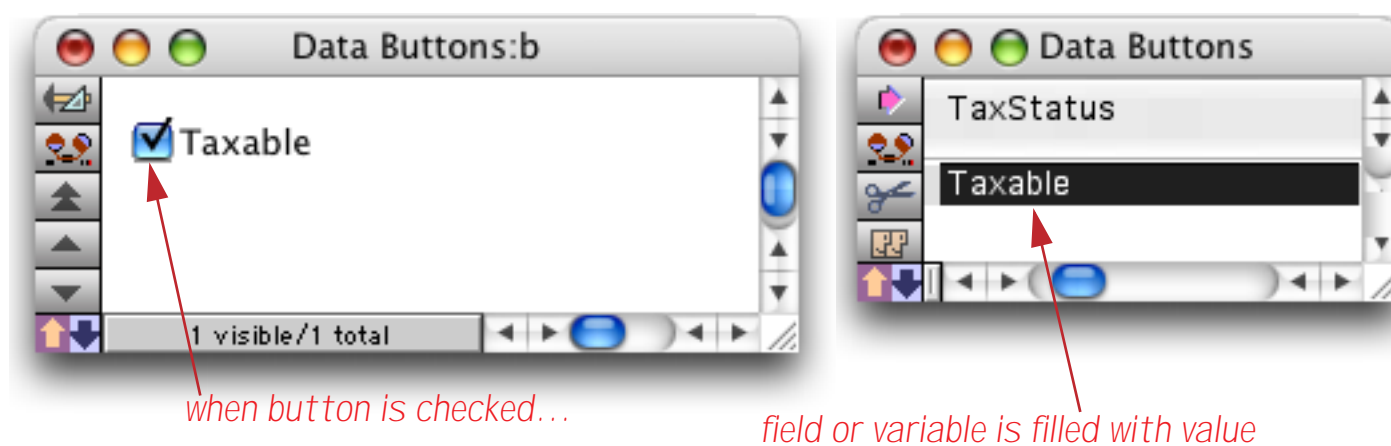
At a minimum you must select a field or variable to store the value, and enter a Title. When you press **OK** the new button will appear on the form.



To try out the button, switch to Data Access Mode. Since this button's value is stored in a field we can use the data sheet to watch the value as the button is checked and unchecked.



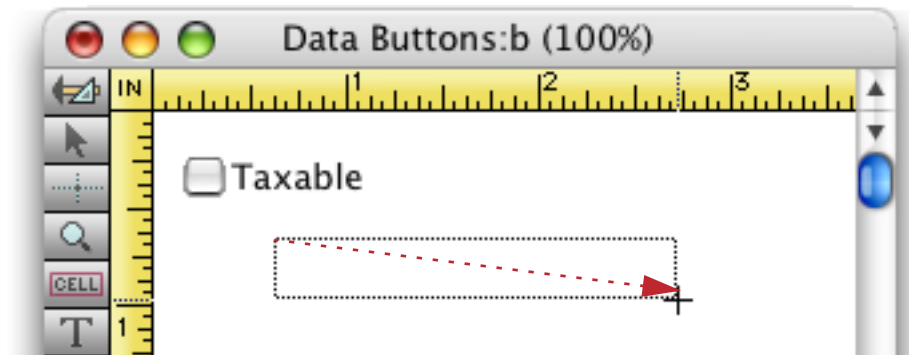
Clicking on the checkbox causes Panorama to fill the field or variable with the value.



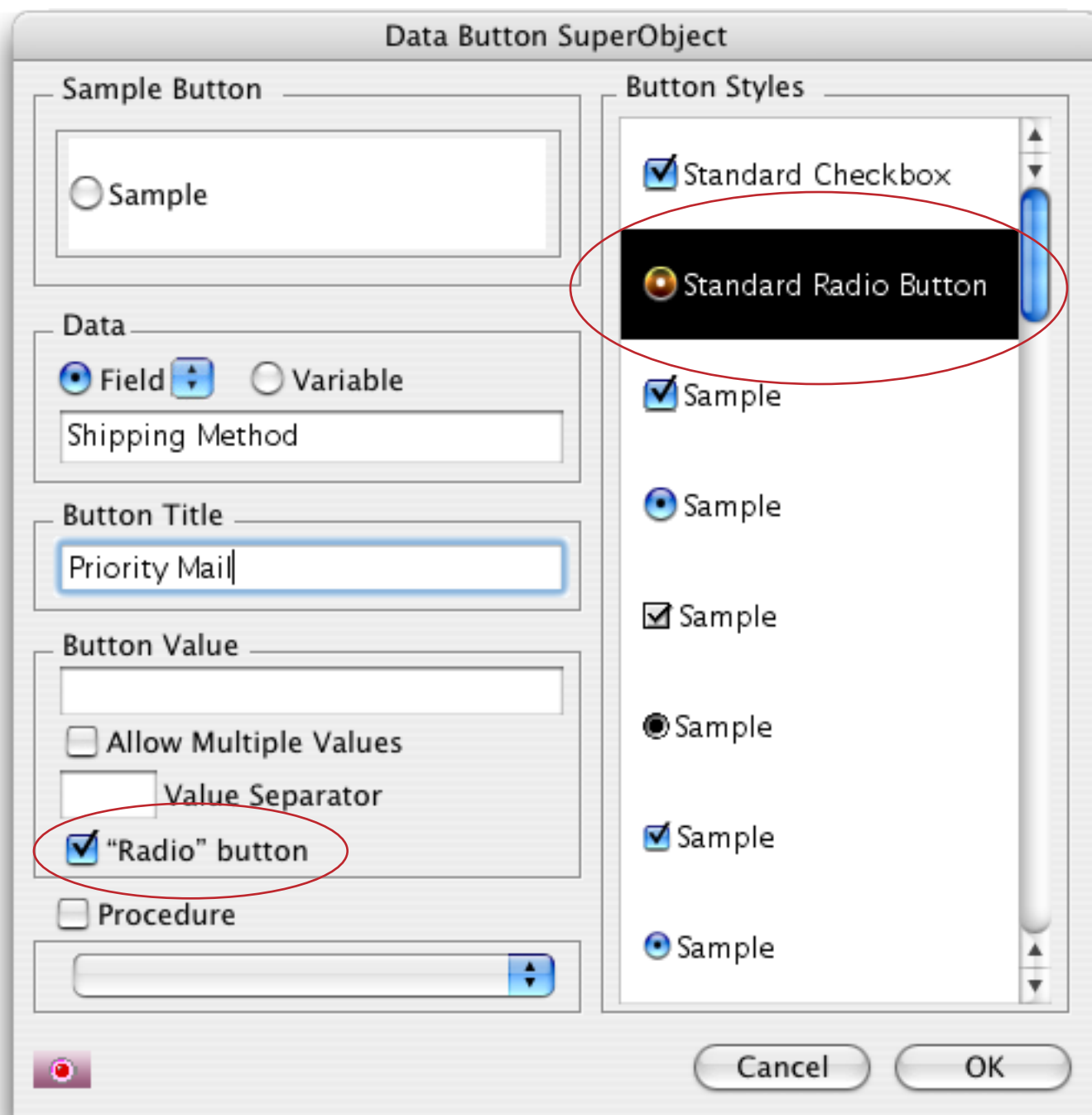
If you click again to un-check the button the field or variable will become empty again. (You may wonder if typing the value into the Data Sheet will cause the button to become checked. The answer is yes!)

### Creating a Group of Radio Buttons

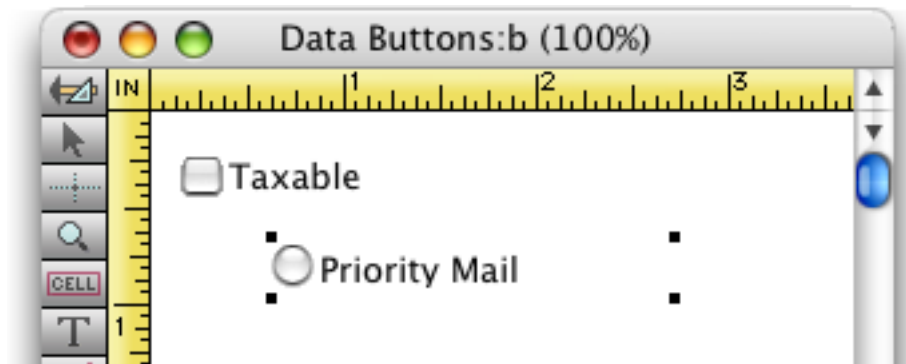
The easiest way to create radio buttons is to create a single button and then make copies. For example, suppose you want to make radio buttons for shipping options — [Priority Mail](#), [UPS Next Day Air](#), [Federal Express](#), [Airborne](#) and [DHL](#). Start by selecting the Data Button tool (see previous section) and dragging to create the first button.



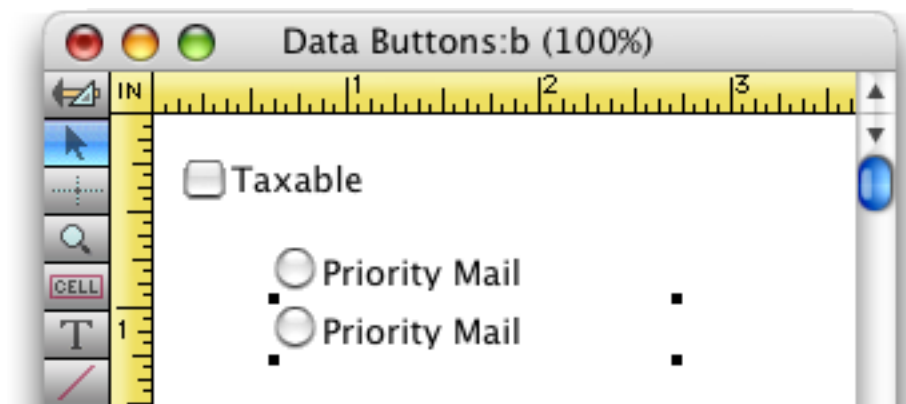
The field or variable and title are set up exactly the same as for a checkbox. You should also enable the **“Radio” button** option, and select a radio button style from the list of styles on the right hand side of the dialog. (Note: The visual style does not affect the operation of the button, so Panorama will happily let you create a radio button that looks like a checkbox, or vice versa.)



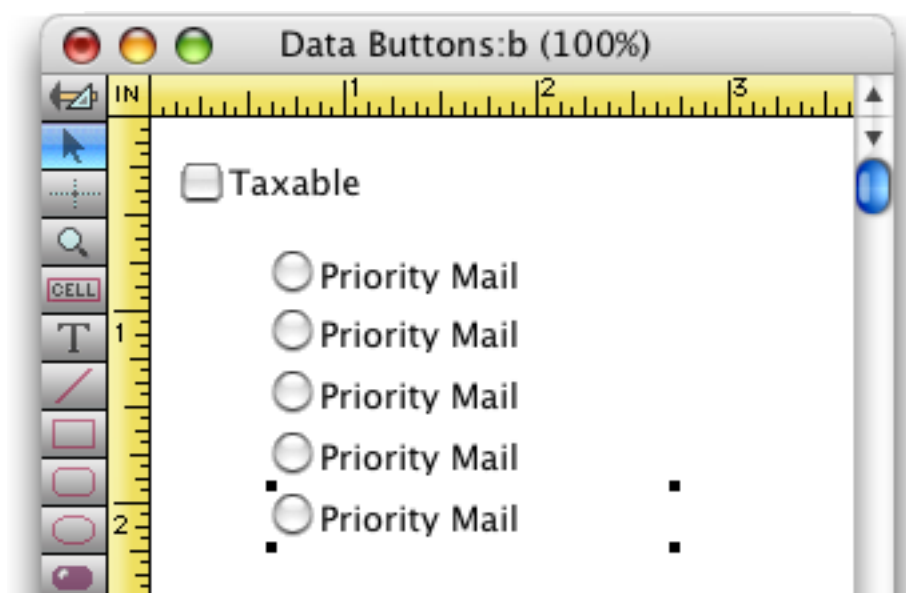
Press **OK** to create the first radio button.



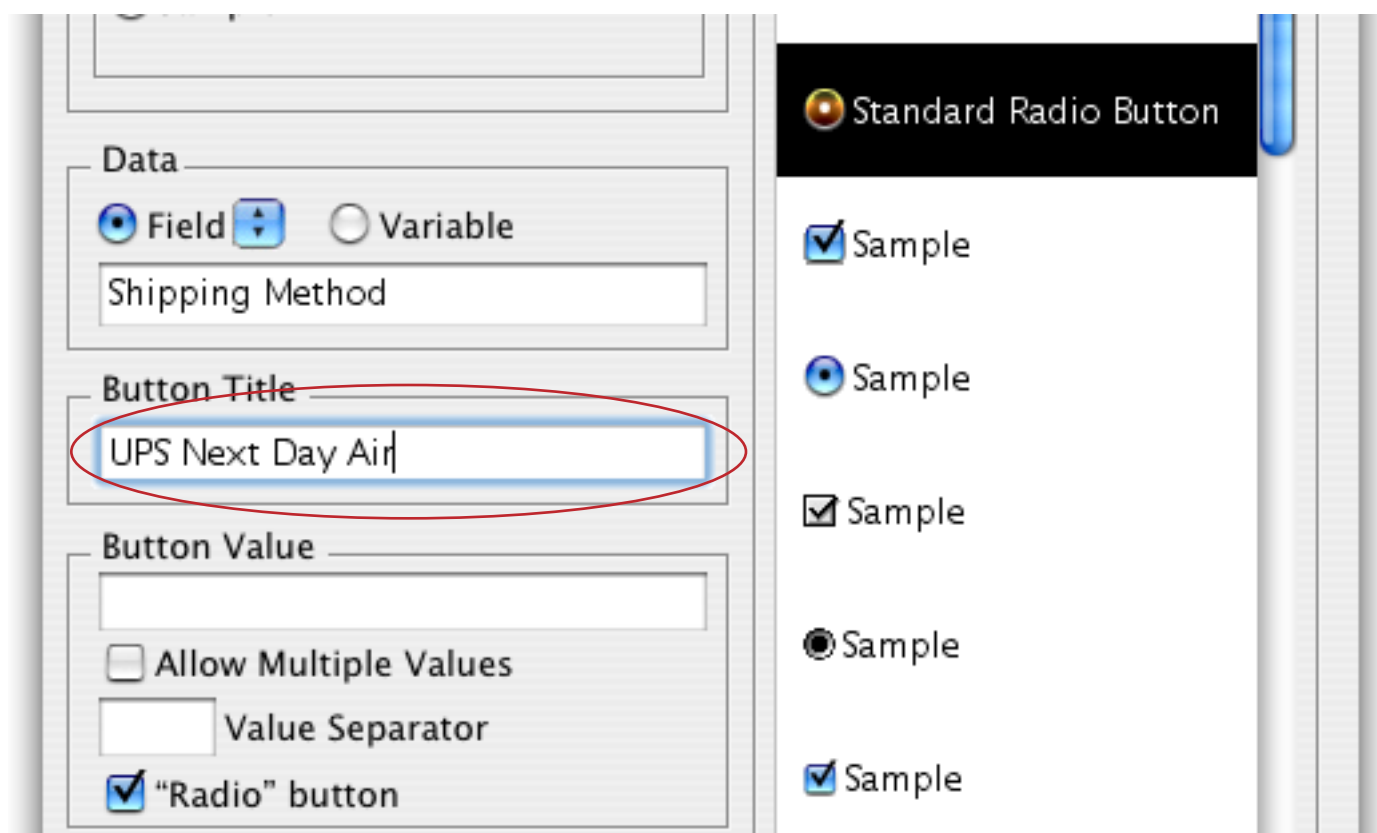
To duplicate this button, make sure that the **Pointer** tool is selected and hold down the **Shift** key and the **Option** (Mac)/**Alt** (PC) key while you drag the button. When you release the mouse Panorama creates a copy of the selected object.



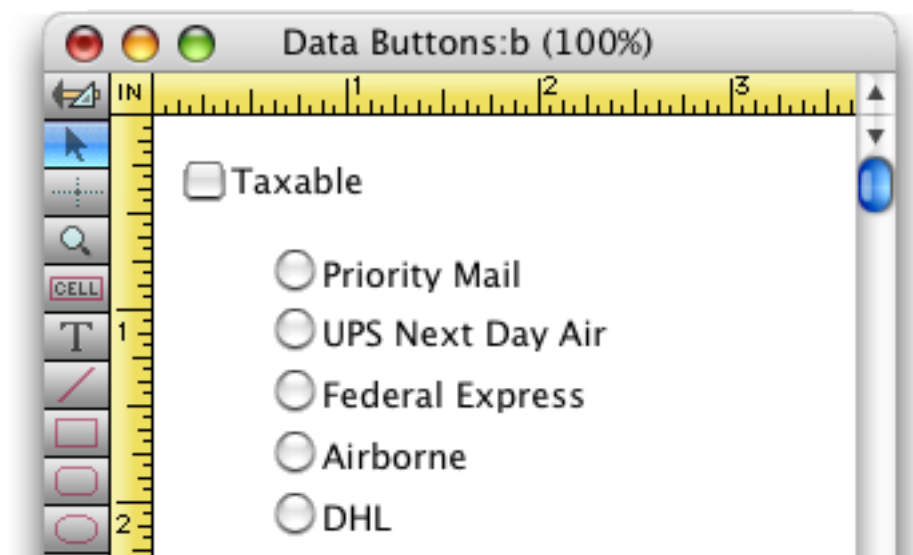
To make additional copies simply use the **Duplicate** command.



Now you need to go back and change the title of each button (except the first). To change the title, simply double click on the button, then fill in the new title.



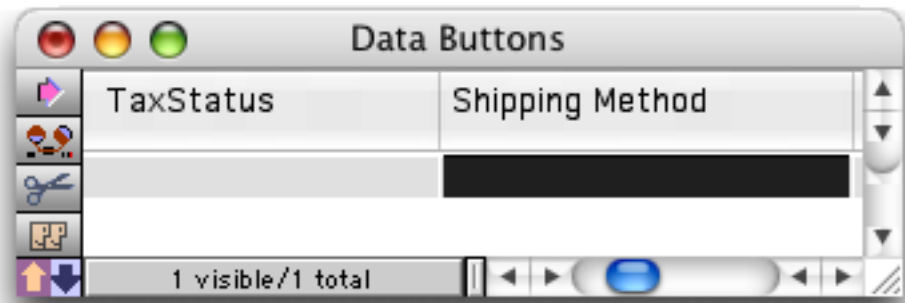
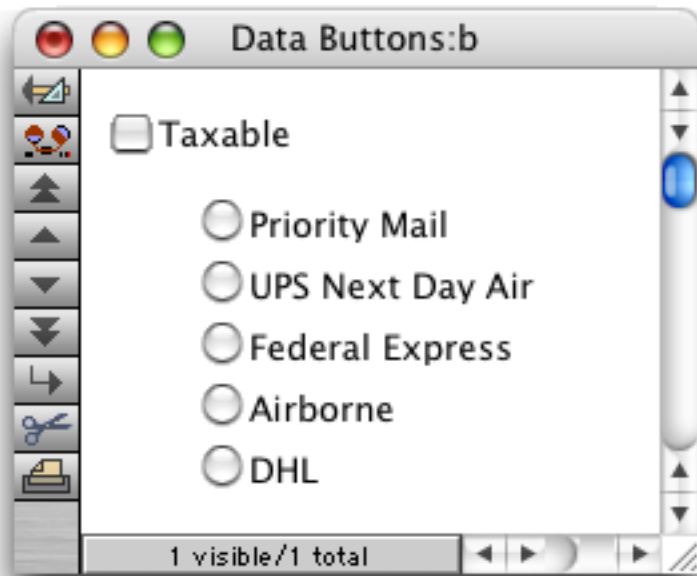
All the other options remain the same — only the title needs to be changed. Repeat this process for each button in the group.



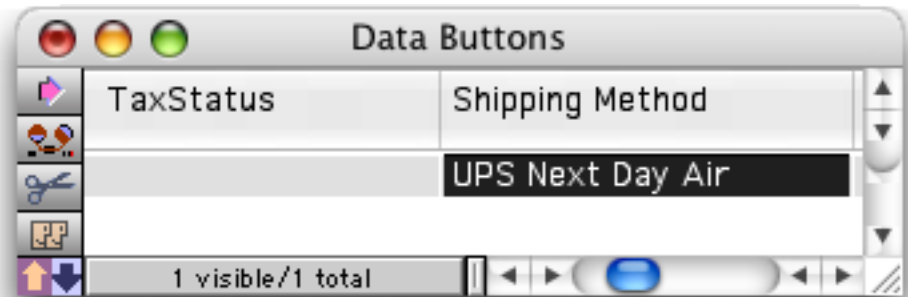
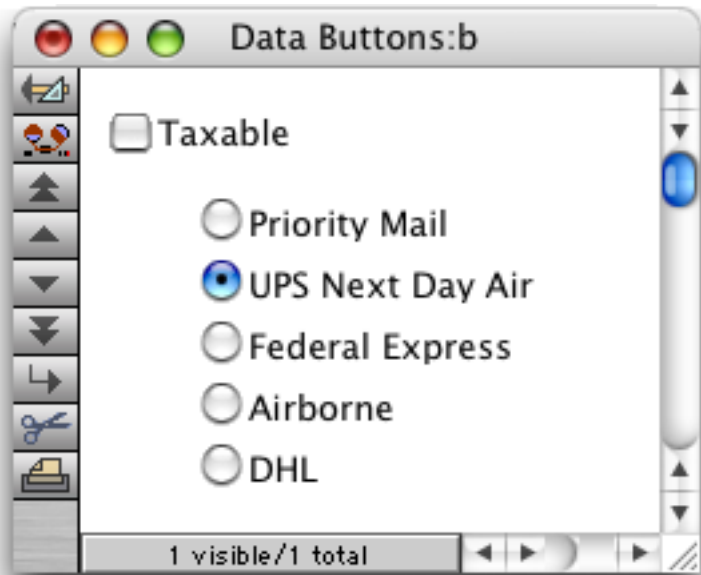
If necessary, you can adjust the spacing of the buttons with the **Spacing** command.



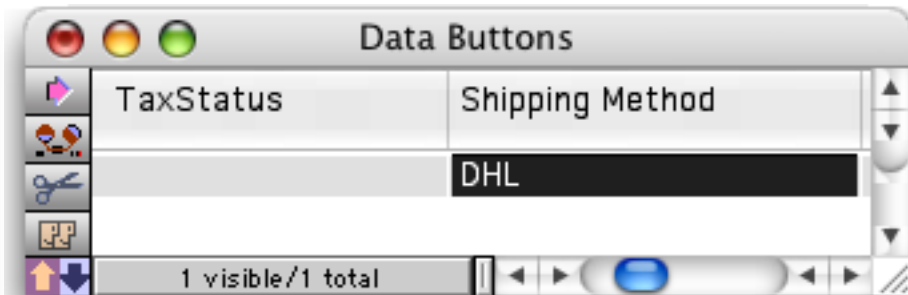
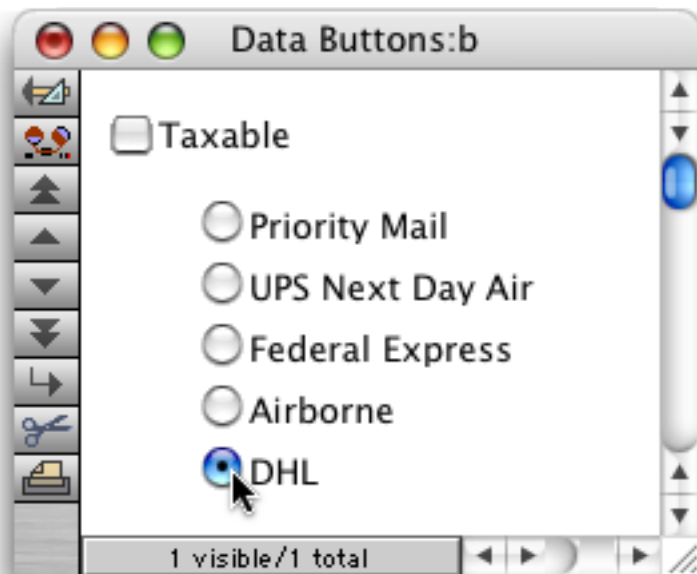
To try out the buttons, switch to Data Access Mode. Since the button's value is stored in a field we can use the data sheet to watch the value as the different buttons in the group are checked. To start out, none of the buttons are checked and the field is empty.



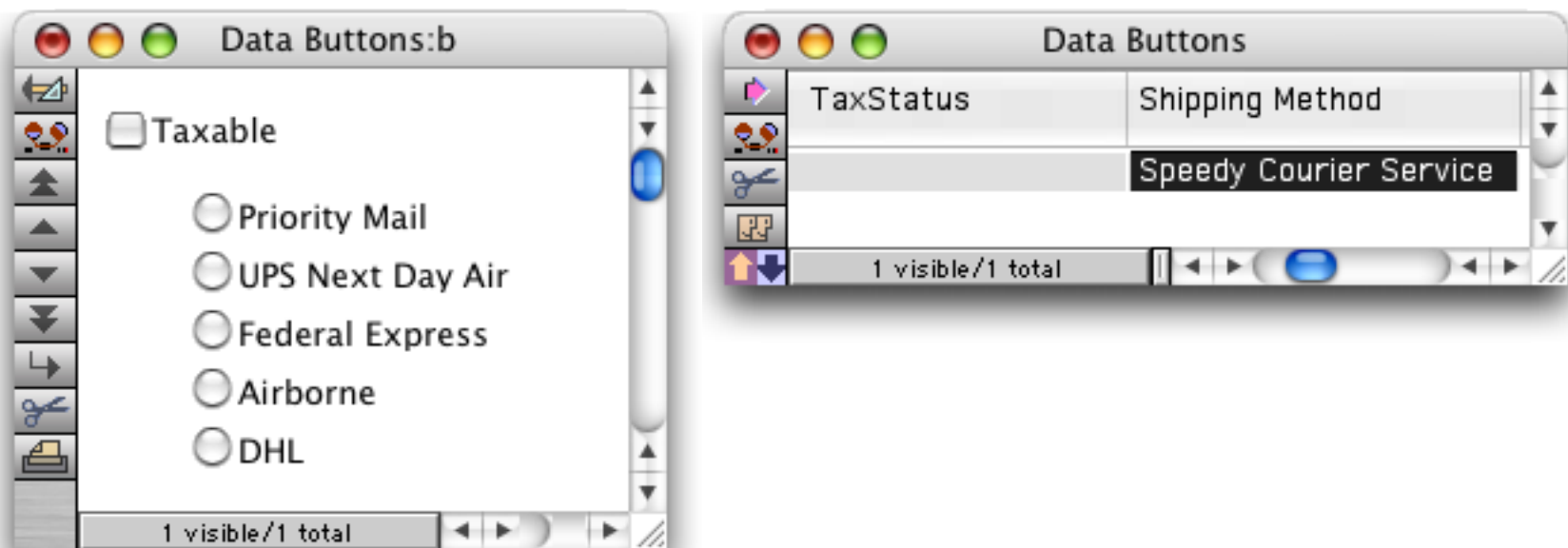
When a button is checked, the corresponding value appears in the field.



Clicking on any button causes the corresponding value to appear in the field.



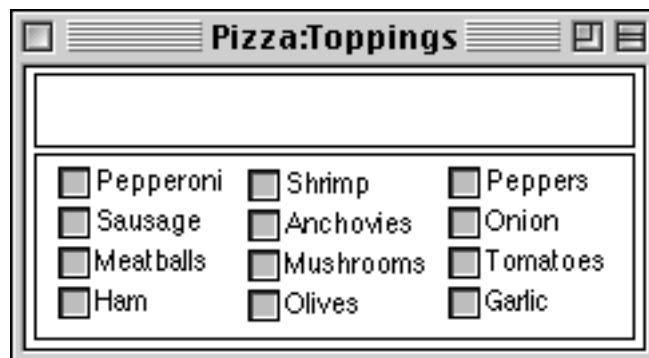
The synchronization between buttons and the field also works in reverse. If you type a value into the data sheet, the corresponding button will be activated. If none of the buttons match the value, all of the buttons will be turned off.



The buttons can also be completely turned off by clearing the contents of the field (making it empty).

#### Multiple Value Button Groups

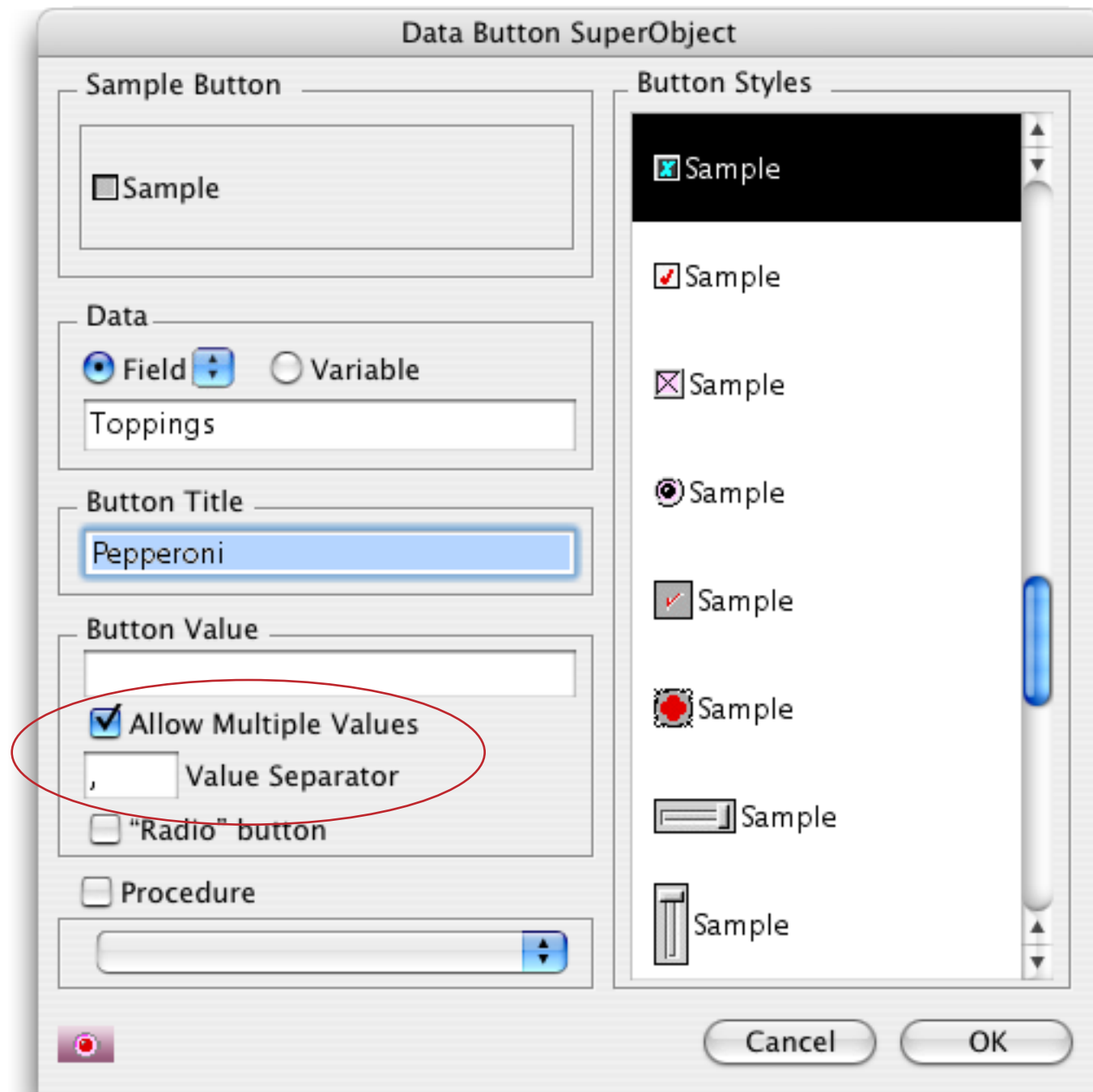
A group of radio buttons works fine as long as only one value at a time is valid. You can't ship a package by both **Priority Mail** and **Federal Express**, only one or the other! Some applications, however, require that multiple options be selected. For example, consider a group of buttons for selecting pizza toppings.



In this application there may be zero, one, two, or even 12 values at one time! There are two ways to create a group of buttons like this.

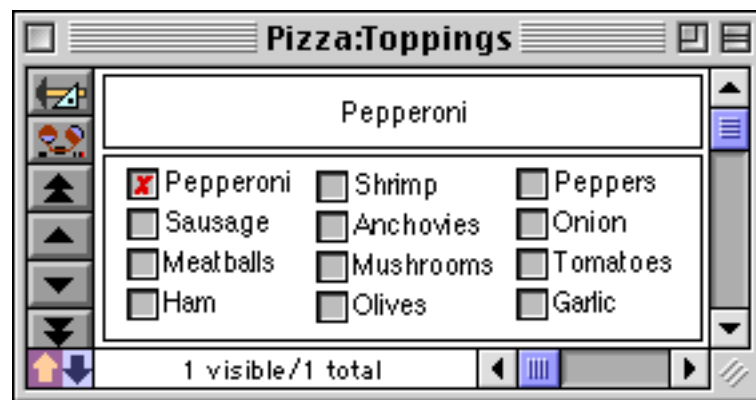
The first method is simply to create a field or variable for each option, and then create a standard checkbox for each field.

The second method allows you to combine all of the values into a single field. Just as with a group of radio buttons, you'll start by creating a single button and making copies (see previous section). However, instead of enabling the "Radio" button option you'll enable the **Allow Multiple Values** option. You also need to specify what character(s) you want placed between each value. In this example, we've chosen a comma.



Set up the first button using the options shown above. Then make copies of the button and adjust the titles, just as described for radio buttons in the previous section.

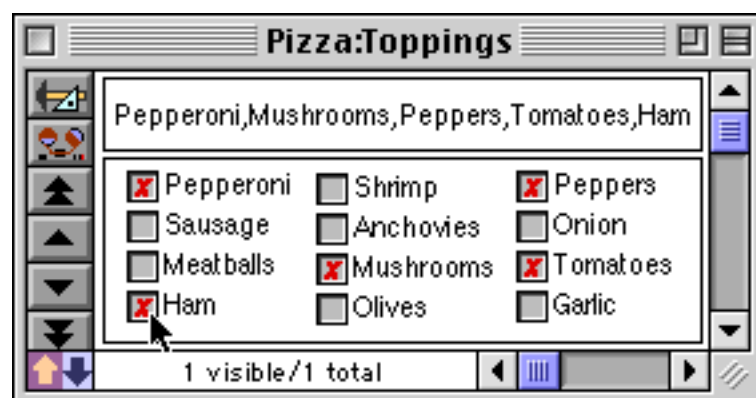
Switch to Data Access Mode to try out the group of buttons. When a single option is clicked the value appears in the field, just as for a standard checkbox. (A Text Display SuperObject has been added to the form to display the value of the **Toppings** field, see “[Text Display SuperObjects™](#)” on page 586 to learn how to create such an object.)



When a second button is clicked, that value is added to the field.



You can keep adding as many values as you like.



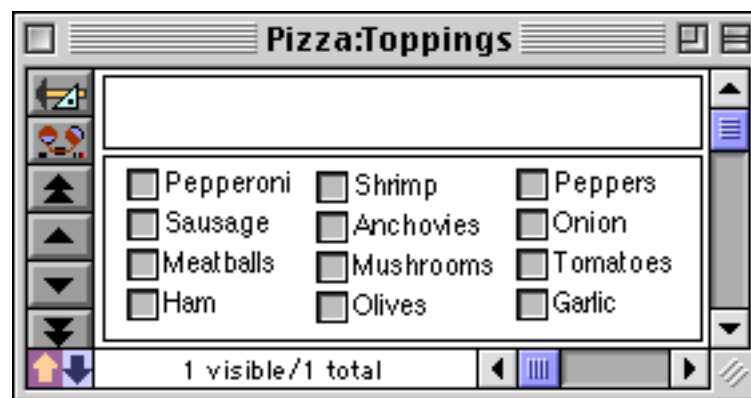
If you un-check a button, that value will be removed from the field (or variable). In this case we have removed **Peppers**.



If **Peppers** is re-enabled, it is added to the end of the value. You cannot control the order of the options within the field or variable.



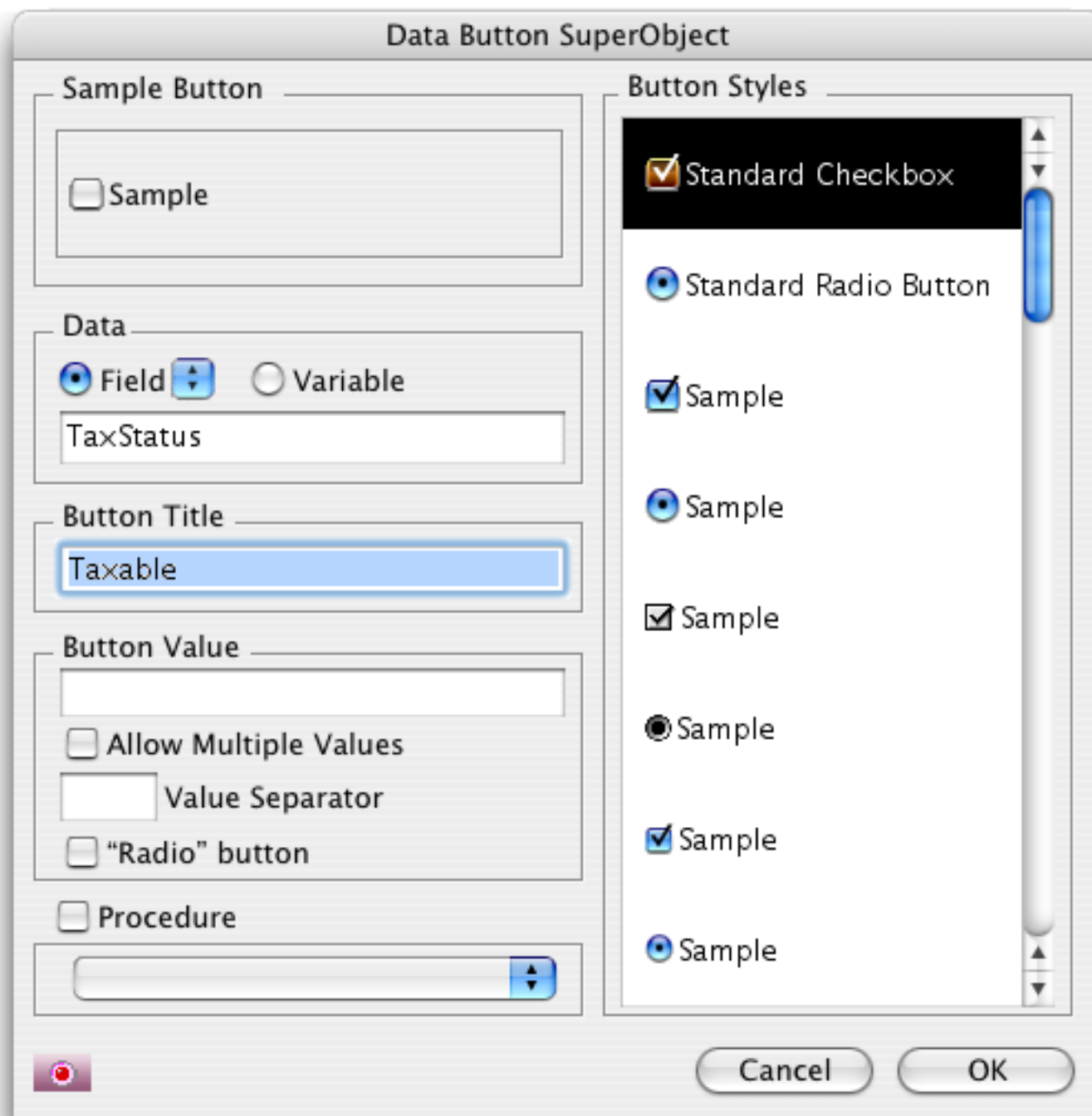
If you un-check all of the buttons the field will become completely empty.



Like other types of buttons, the synchronization between the buttons and the field or variable works both ways. You can type in a value or combination of values into the field or variable and the appropriate buttons will automatically "light-up."

## Super Data Button Options

The left hand side of the Data Button dialog controls the operation of the button, the right hand side controls the appearance of the button.



### Data

This section of the dialog specifies the field or variable associated with this button. Type the name of the field or variable into the box (or select the field name from the pop-up menu next to the **Field** radio button). If the button is associated with a variable that has not been created with a procedure, Panorama will automatically create a global variable with this name whenever the button appears. This global variable can be used in formulas and procedures just like any other global variable (see “[Variables](#)” on page 1189 and “[Variables](#)” on page 1370 of the **Panorama Handbook**).

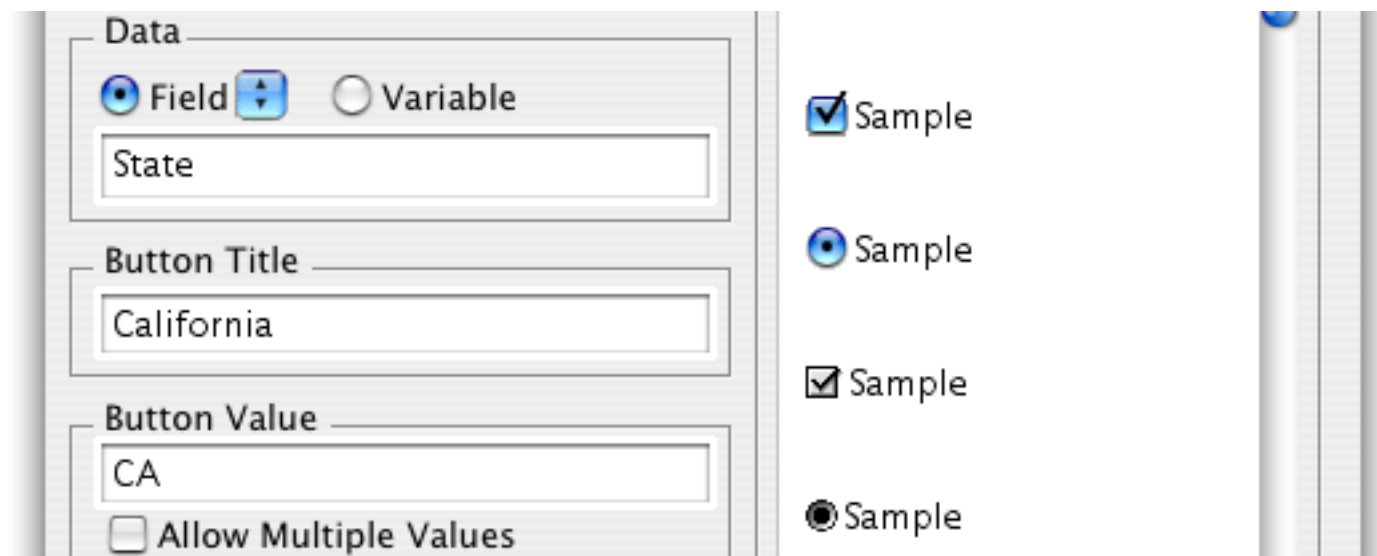
### Title

This section of the dialog specifies the title of the button. This is the title that is actually displayed on the screen. If the value is empty (see next section) Panorama will use the title as the value.



## Value

This section of the dialog specifies the data value corresponding to this button. Each button can have a single value, which you should type into the box next to the word **Value**. If you leave this box empty, Panorama will use the **Title** for the data value (see previous section). This example shows how you can use a value that is different from the title.



Using this technique you can make a group of radio buttons for western states.

- Arizona
- California
- Nevada
- Oregon
- Washington

The **State** field will not contain the fully spelled out state names, but only the 2 letter abbreviations (**AZ**, **CA**, **NV**, etc.).

### Allow Multiple Values

It's possible to group together multiple buttons associated with the same field or variable. Normally clicking any button in the group erases the current value and replaces it with the new value (normal radio button behavior). However, if you have the **Allow Multiple Values** option turned on, clicking on the button will add the new value to the existing data in the field or variable, with the **Value Separator** in between.

As an example of the **Allow Multiple Values** option, consider pizza toppings. A pizza may have one, two, three or more toppings, or even none at all. Using the **Allow Multiple Values** option, you can create a series of checkboxes that will generate a list of toppings in a single field or variable. As the user clicks on each topping, it will be added to the end of the list. To remove a topping from the list, click on it again. See See "[Multiple Value Button Groups](#)" on page 365 for an example of this technique.

### Value Separator

This is the text that will appear between each value in a multiple value list. Common separators include commas, spaces, slashes and hyphens. The separator may be up to 6 characters long, so you can use a multi character separator like comma-space. (However, if you want to process the value with Panorama's array functions you should use only a single character separator. See "[Text Arrays](#)" on page 1229 for more information on these functions.

## "Radio" button

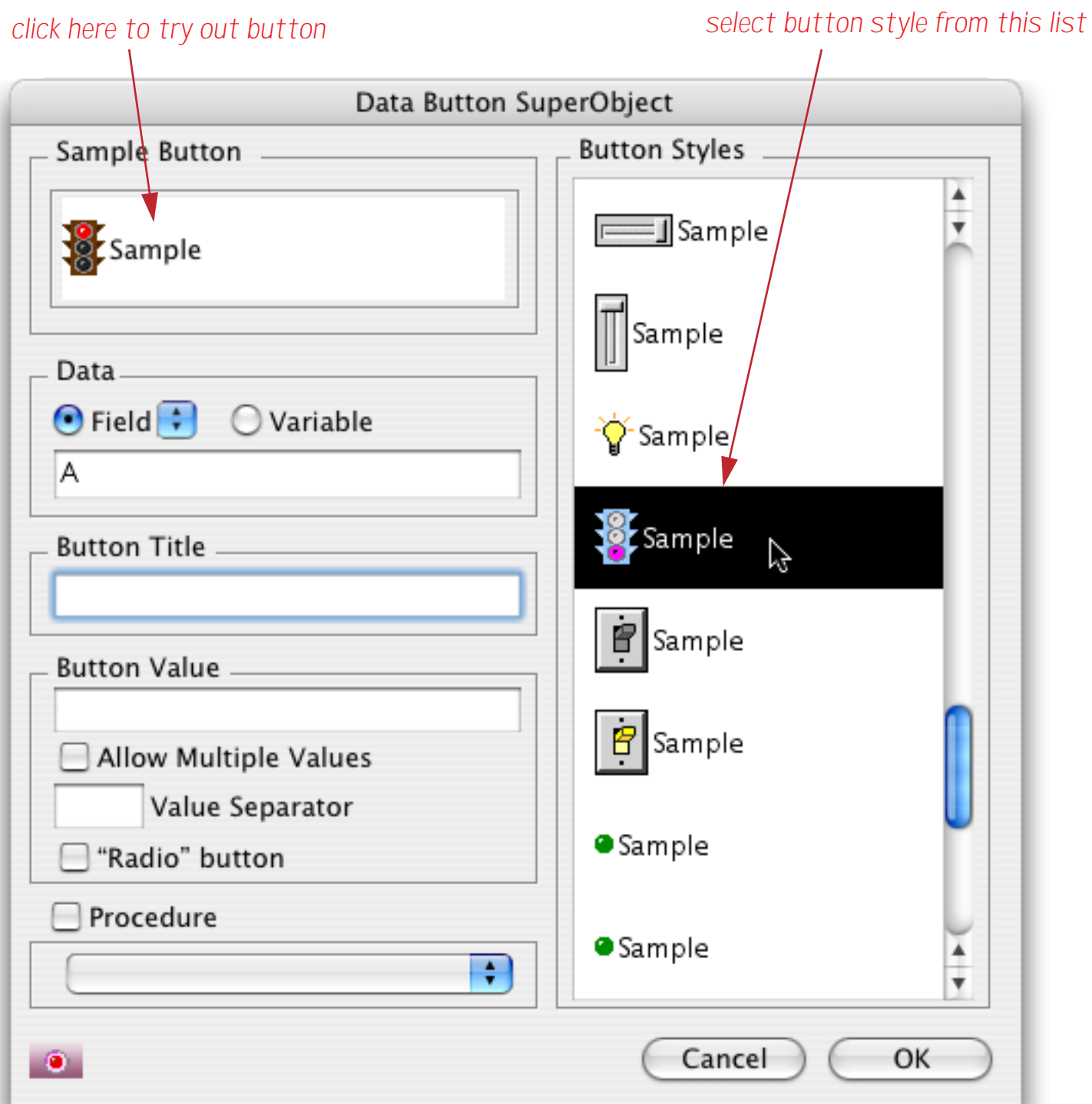
The data button normally toggles the value on and off each time you click on it. Turning on the **"Radio" button** option prevents the button from toggling off. In other words, you can turn the button on, but you can't turn it off again. Usually this option is only used for radio button combinations, where the value will be automatically turned off by clicking on another button in the group. This option prevents the user from turning off all the buttons in the group, so there is always at least one radio button checked.

## Procedure

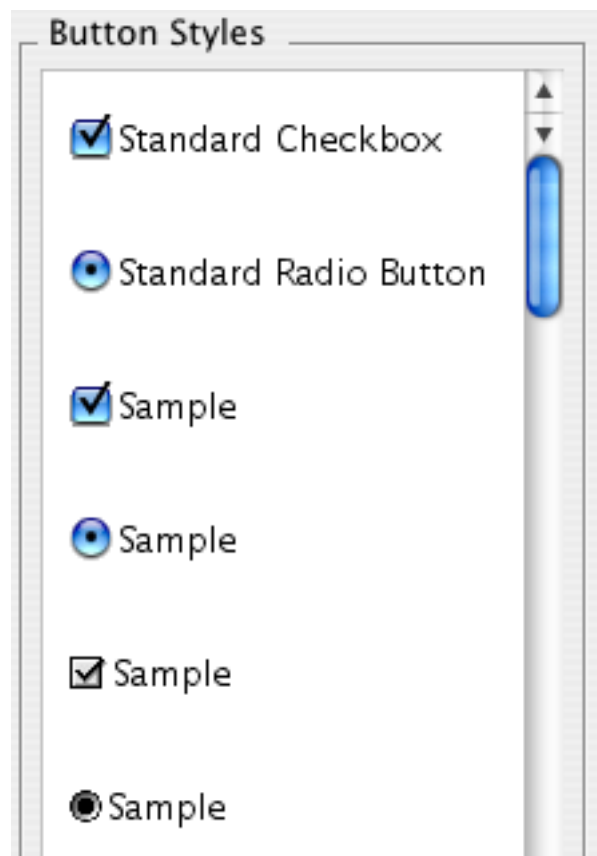
This section of the dialog specifies the procedure that will be triggered when this button is clicked (if any). The procedure can get information about what button was clicked by using the `info("trigger")` function. Even if you don't specify a procedure here, clicking on the button will trigger any automatic formulas and procedures associated with the field for the button.

## Sample

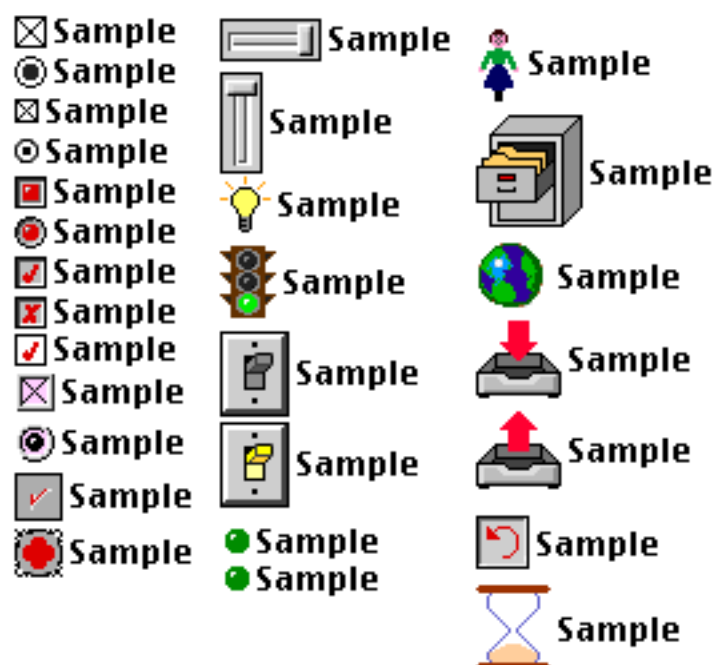
This section of the dialog shows a sample of the button. You can select the button style from the scrolling list on the right. Click on the sample button on the left to see what the button style will look like in both the on and off positions.



The first two styles, **Standard Checkbox** and **Standard Radio Button**, are special. Data buttons created with these styles will adjust automatically depending on what operating system is being used. In other words, the same button will change appearance depending on whether the database is being used on Mac OS X, Mac OS 9, or Windows.



With all of the other options what you see is what you get. Here are some of the other built-in choices available. (If you need to create your own custom button, use the Flash Art Data Button.)

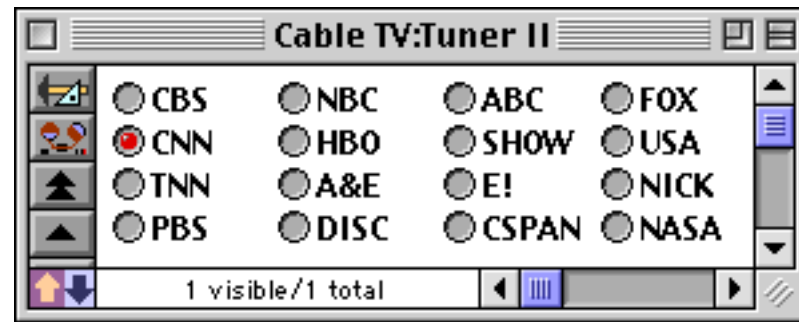


Note: Button operation is not affected by the button's appearance. You can create a button that "looks" like a radio button but acts like a checkbox, or visa versa.

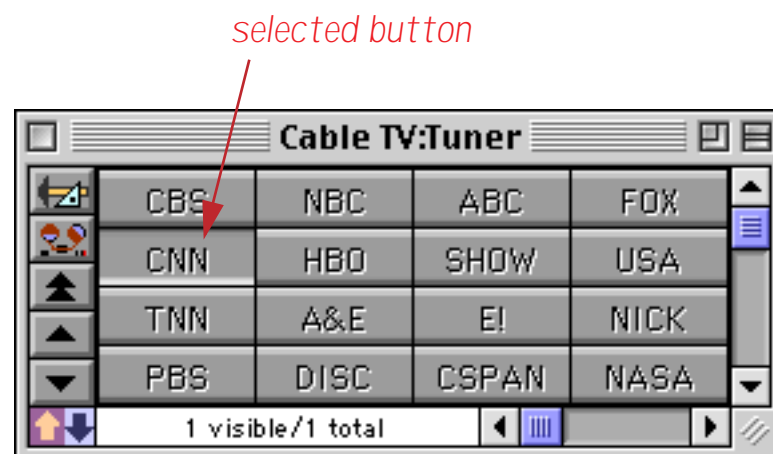
If you need to make your own custom data buttons you can use the Flash Art Data Button SuperObject. See Chapter 17 of the **Panorama Handbook** to learn about this type of button.

## Sticky Push Button SuperObjects™

Sticky Push Buttons look like Push Buttons (see “[Super Object Push Button](#)” on page 351), but they operate like Data Buttons (see “[Data Button SuperObjects™](#)” on page 358). Instead of popping back up when you release the mouse, a Sticky Push Button stays pushed in like a checkbox or radio button. Like Data Buttons, Sticky Push Buttons are associated with a field or variable, and can be used separately or in groups (radio sticky push buttons, anyone?) To illustrate, here is a collection of options created as regular data buttons.

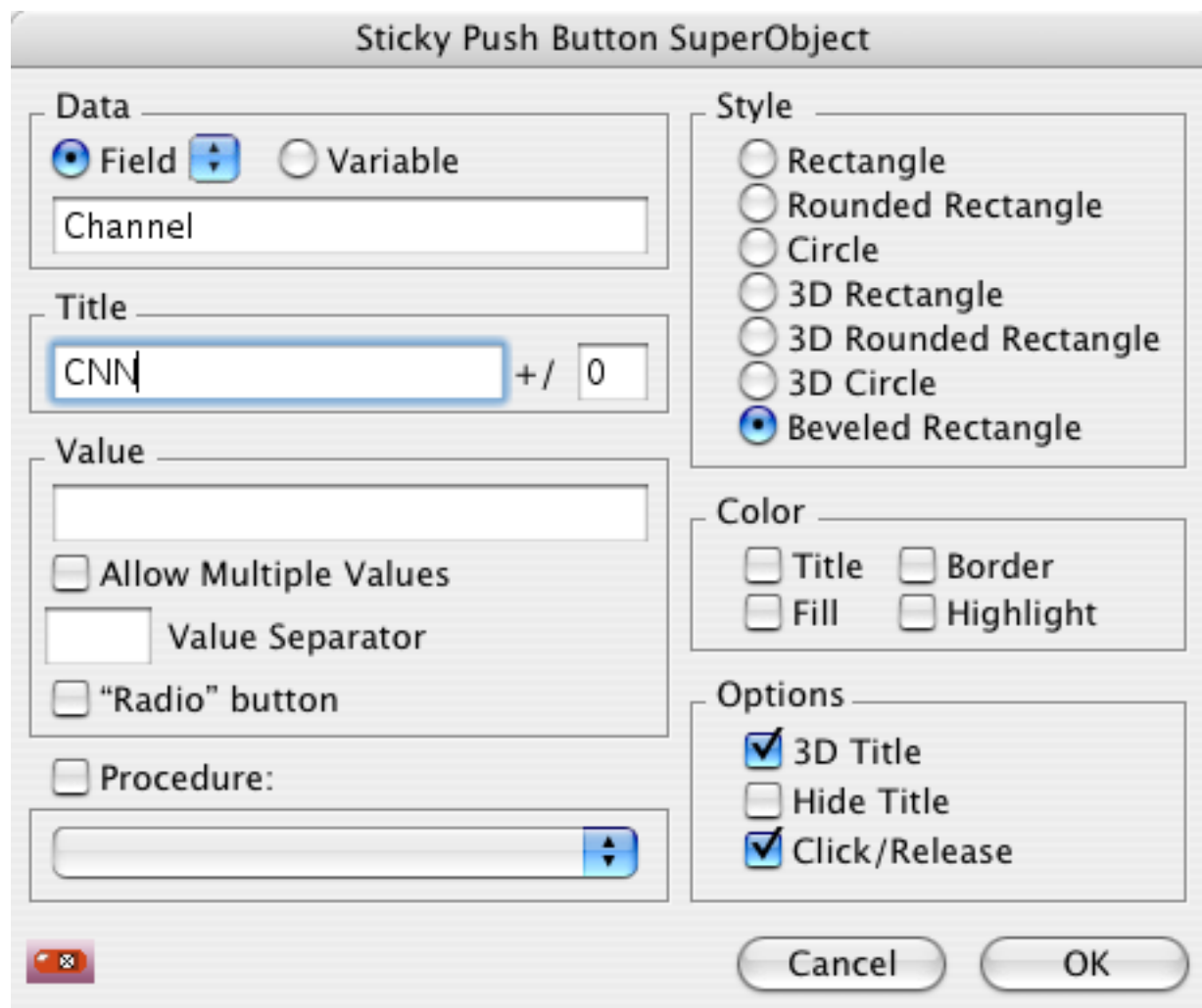


Here is the exact same example, but created with Sticky Push Buttons. Notice that CNN is also selected in this group of buttons.



These two examples operate exactly the same — the only difference is their appearance.

The Sticky Push Button configuration dialog is a combination of the options for Push Buttons and Data Buttons.



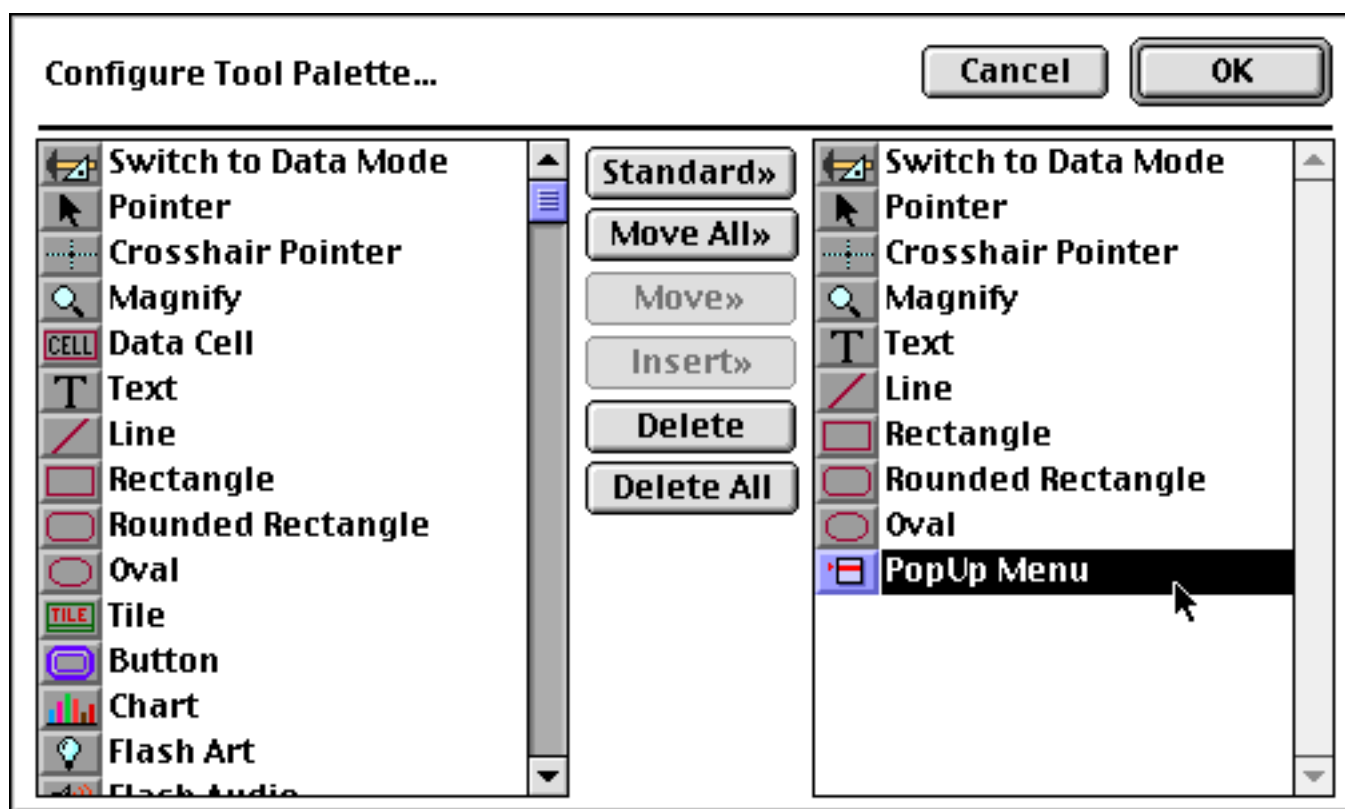
## Pop-Up Menus

Radio buttons work well when there are only a few options. When you get past a dozen or so options, you'll probably want to use a pop-up menu or scrolling list instead. Panorama has three methods for creating pop-up menus: 1) Pop-Up Menu SuperObjects, 2) "Classic" buttons, and 3) Procedures (programming). Only the first is discussed in this book, see the [Panorama Handbook](#) to learn about the others.

### Pop-Up Menu SuperObjects™

The easiest and most flexible way to create a pop-up menu is with the Pop-Up Menu SuperObject™. A Pop-Up Menu SuperObject™ may be associated with any field or global variable. When the user makes a selection from the pop-up menu, the corresponding field or variable is automatically updated with the new value. Because the list of menu choices is calculated with a formula, the pop-up menu can change on the fly if necessary. You can also choose the menu font, color, and style (multi-column or scrolling).

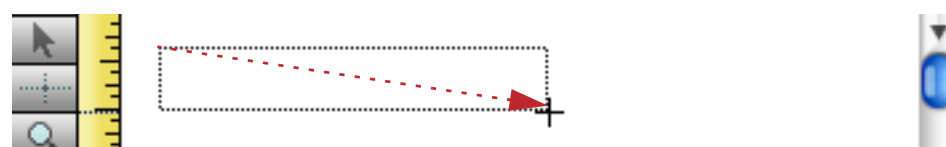
The Pop-Up Menu SuperObject tool is not in the default tool palette, so you'll need to move the use the Tool Palette dialog to add this tool to the palette if it is not already there.



Now that the tool is added to the palette you can select it.



Once the tool is selected, drag the mouse across the form in the location where you want to create the pop-up menu object.





When you release the mouse, the Super Pop-Up Menu configuration dialog will appear.

Pop-Up Menu SuperObject

Data

Field  Variable

Shipping Method

Menu Formula

```
replace("A;B;C;D;E;F;G;H;I;J",";","&#92)
```

Menu Action

Procedure:

Menu Type

Standard  
 Multi-Column  
 Chicago 12  
 Combo Box  
 Popup/Combo Box

Display Options

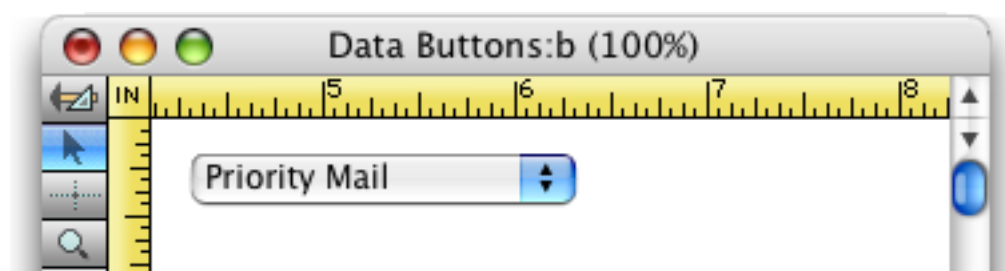
Show Value  
 Show Triangle  
 Standard Icon  
 Drop Shadow (1 Px)  
 Drop Shadow (2 Px)

Color

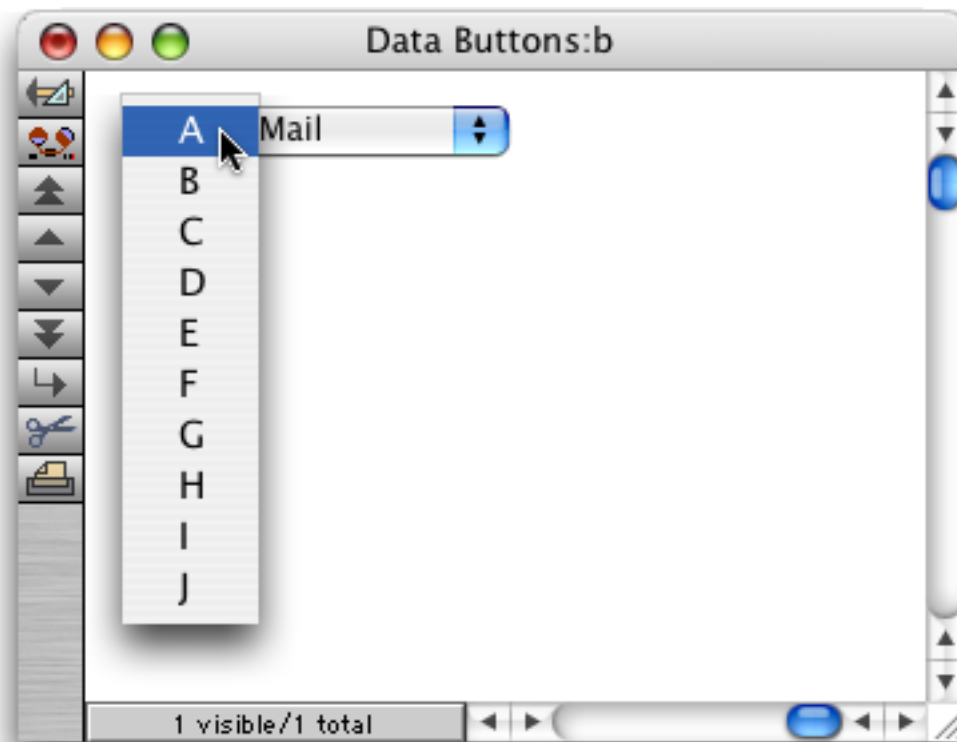
Value  
 Fill  
Fill Color:   
 Shadow

Cancel OK

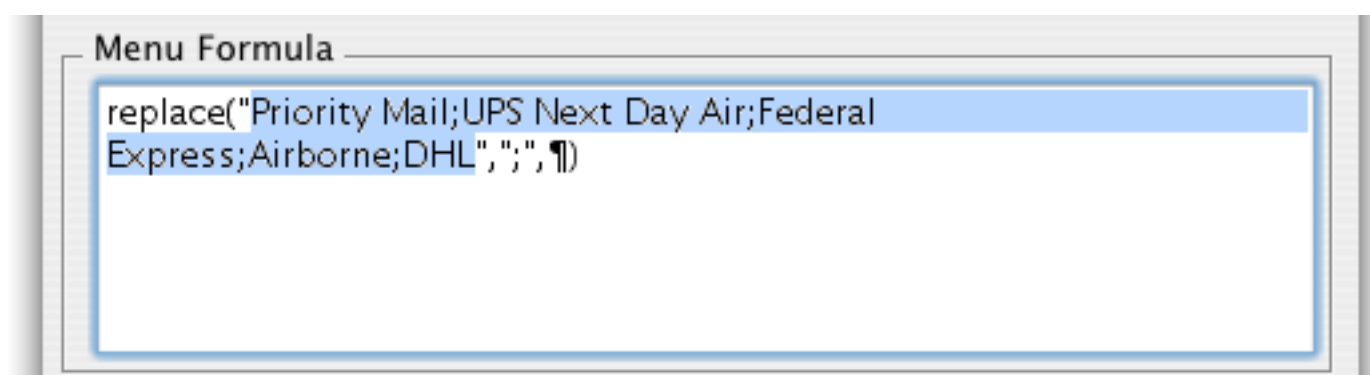
At a minimum you must select a field or variable for the pop-up menu. This field or variable will hold the result of the pop-up selection. Press **OK** to create the pop-up menu object.



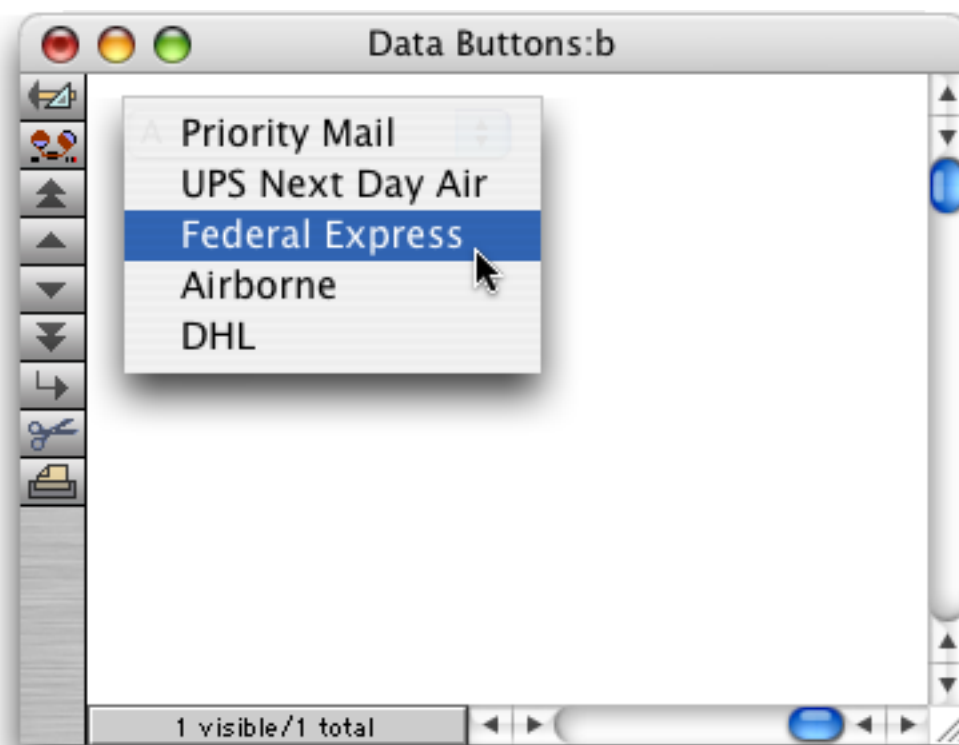
Switch to Data Access Mode to try out the pop-up menu.



The default pop-up menu contains ten items, A thru J. To change the items in the menu, switch back to Graphics Mode, select the **Pointer** tool and double click on the pop-up object. In the formula, replace [A;B;C;D;E;F;G;H;I;J](#) with the actual items you want to appear in the menu, with each item separated by a semicolon.



Press **OK** and switch back to Data Access Mode to try out the revised pop-up menu.



### The Pop-Up Menu Formula

The menu formula calculates the list of menu choices. Each menu choice is on a separate line, separated by a carriage return.

When the user presses on the pop-up menu button, Panorama takes the formula, calculates it, splits the result into individual menu items (one per line), then displays the pop-up menu and allows the user to make a selection. All this happens in the blink of an eye as the user clicks on the button.

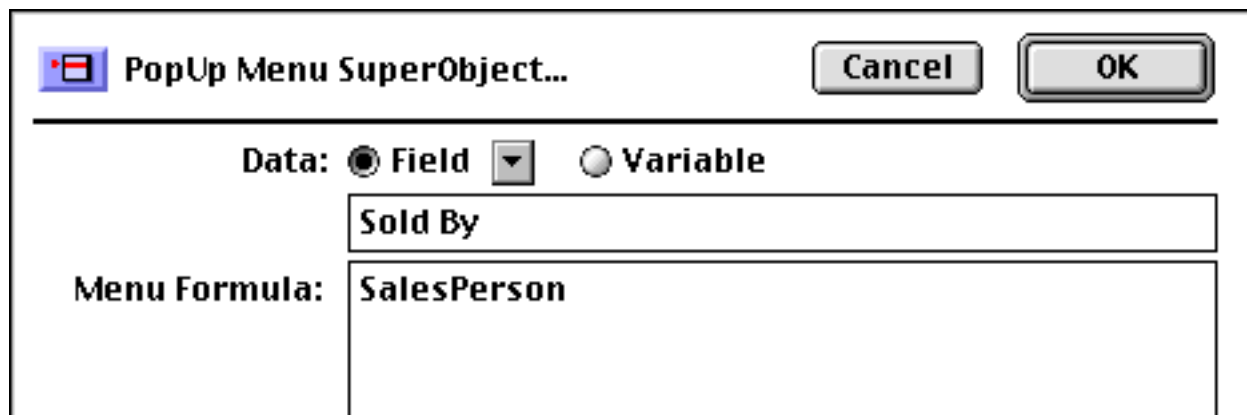
So much for theory, now let's take a look at some real world menu formulas. Suppose you want to create a pop-up menu with three choices: **Gold**, **Silver** and **Bronze**. Keeping in mind that the ¶ symbol represents a carriage return, the most basic formula that can be used to create this menu would be:

```
"Gold" + ¶ + "Silver" + ¶ + "Bronze"
```

To type the ¶ symbol, press **Option-7** on the Macintosh and **Alt-0182** on the PC. If your menu has a lot of items it can be kind of a pain to type in this symbol over and over again. We can use Panorama's **replace()** function to make this formula easier to type (see "**REPLACE()**" on page 5665). In the formula below, the **replace()** function will change the semicolons into carriage returns. This allows you to type the menu choices with just a semicolon in between them. This option is so useful that Panorama preloads this function into the menu formula.

```
replace("Gold;Silver;Bronze", ";", ¶)
```

The real power of the menu formula is unleashed when you use a variable or field in the formula. Since the variable or field may be changed at any time with a procedure (or even with standard data entry), the pop-up menu can change at any time. For example, suppose you want to create a pop-up menu of salespeople in your company. Simply create a **permanent variable** named `SalesPeople` and fill it with the name of each salesperson in your company on a separate line. You can create a preferences form that allows you to edit the list using a Text Editor SuperObject. When you create the pop-up menu, the menu formula will simply be `SalesPeople`.



Each time you click on the pop-up menu it will display the current list of salespeople.



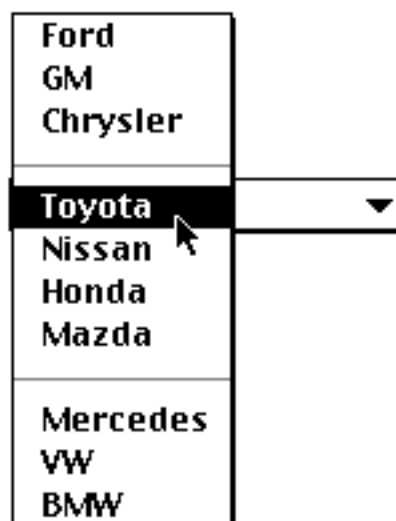
This list can be edited at any time simply by editing the permanent variable.

#### Dividing Lines in the Menu

To put a dividing line in a pop-up menu, simply create a line with the entry `(-`. The formula below produces a menu of car manufacturers in three sections: US, Japanese, and German.

```
replace("Ford;GM;Chrysler;(-;Toyota;Nissan;Honda;Mazda;(-;Mercedes;VW;BMW", ";", "\n")
```

The dividing lines are not enabled in the menu, so the user cannot accidentally choose a dividing line.



## Pop-Up Menu Options

The SuperObject™ Pop-Up menu dialog is divided into several sections.

The screenshot shows the 'Pop-Up Menu SuperObject' dialog box. It is organized into several sections:

- Data:** Contains radio buttons for 'Field' (selected) and 'Variable'. Below is a text box containing the letter 'A'.
- Menu Formula:** A large text area containing the formula: `replace("A;B;C;D;E;F;G;H;I;J",";",␣)`.
- Menu Action:** A checkbox labeled 'Procedure:' followed by a dropdown menu.
- Menu Type:** Radio buttons for 'Standard' (selected), 'Multi-Column', 'Chicago 12', 'Combo Box', and 'Popup/Combo Box'.
- Display Options:** Checkboxes for 'Show Value', 'Show Triangle', 'Standard Icon', 'Drop Shadow (1 Px)', and 'Drop Shadow (2 Px)'. The first three are checked.
- Color:** Checkboxes for 'Value', 'Fill', and 'Shadow'. Below 'Fill' is a 'Fill Color' selection box.

At the bottom, there are 'Cancel' and 'OK' buttons, and a small icon in the bottom-left corner.

### Data

This section of the dialog specifies the field or variable associated with the pop-up menu. Type the name of the field or variable into the box (or select the field name from the pop-up menu next to the **Field** radio button). If the pop-up menu is associated with a variable that has not been created with a procedure, Panorama will automatically create a global variable with this name whenever the form containing this object appears. This global variable can be used in formulas and procedure just like any other global variable.

### Menu Formula

This section specifies the choices listed in the actual pop-up menu. Instead of simply typing in the choices, you enter a formula that calculates the choices. (Since the formula can use a field or variable, this allows the menu to be changed easily on the fly.) The formula must calculate the list of choices, with each choice separated by a carriage return. (Remember, a carriage return can be represented by the ¶ symbol (Mac: **Option-7**/PC: **Alt-0182**) in a formula.) See "[The Pop-Up Menu Formula](#)" on page 378 for a detailed discussion of the menu formula.

## Menu Type

The Pop-Up Menu SuperObject™ supports three menu styles: **Standard**, **Multi-Column** and **Combo Box**. **Standard** menus look and operate just like all of the other menus on your system. **Multi-Column** menus will automatically split the menu into an array of two or more columns. This allows dozens or even hundreds of options to appear on the screen at one time.



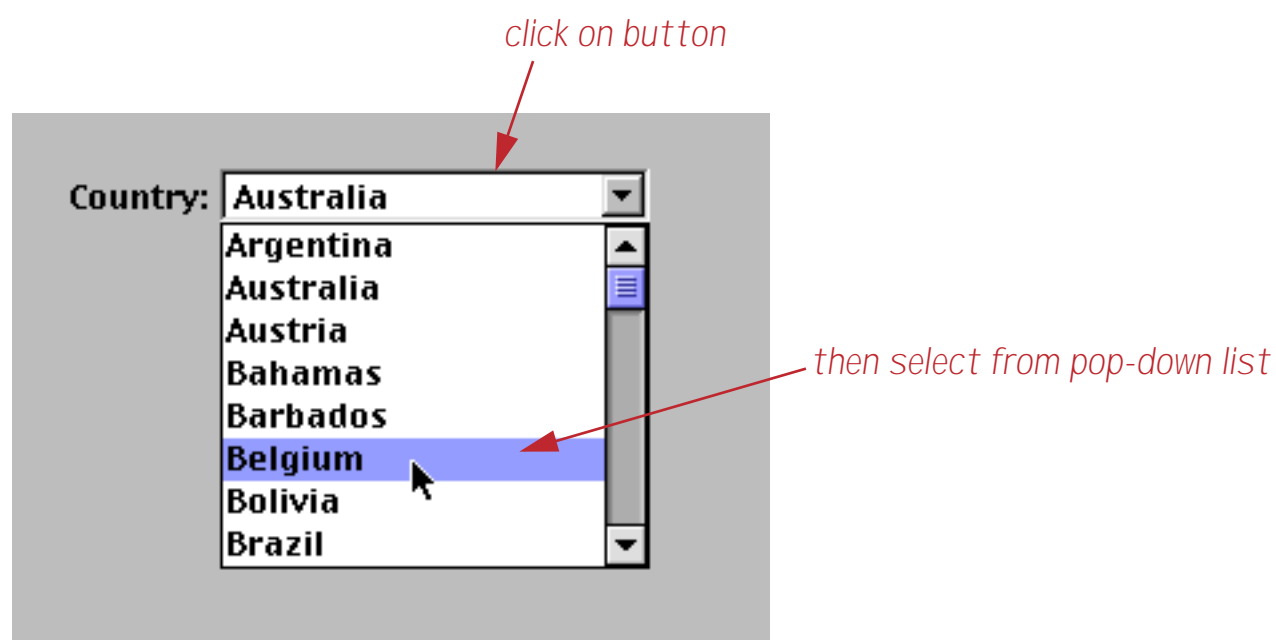
The **Chicago 12** option forces the actual pop-up menu to always appear in the standard system font, no matter what font and size are selected for the pop-up object. (On early Macintosh systems this font was Chicago 12, hence the name of the option.) If you leave this option off, the menu can be displayed in any font and size. Just select the Pop-Up Menu SuperObject™ and choose the font and size the normal way. (Note: If you select the **Scrolling** option, this option is ignored and the menu will always be displayed in Chicago 12 point type. Only **Multi-Column** menus can be displayed in non-standard sizes.)

The **ComboBox** option makes the pop-up button look and operate like a Microsoft Windows style Combo Box. This type of button looks best on a gray or colored background.

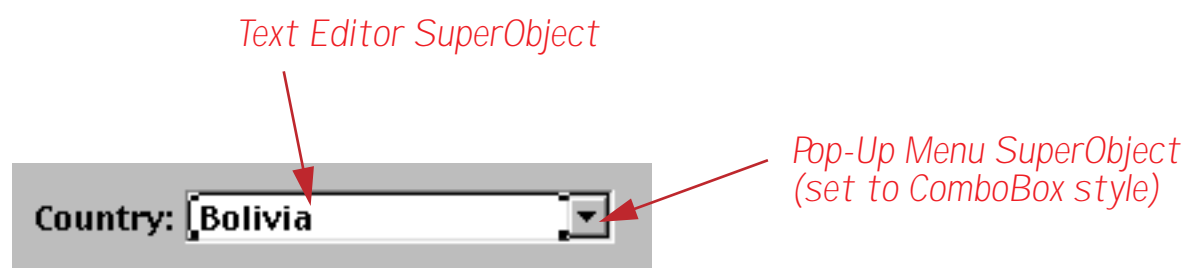




To use a Combo Box, simply click anywhere on the box to make a pop-down list appear, then make your choice from the list.



If you would like to be able to directly edit the value (the Country in this case) you can superimpose a Text Editor SuperObject on top of the Combo Box.



Now you have a choice of editing methods. You can click or drag on the text to edit it directly, or click on the Combo Box icon to pop-down the list of choices.

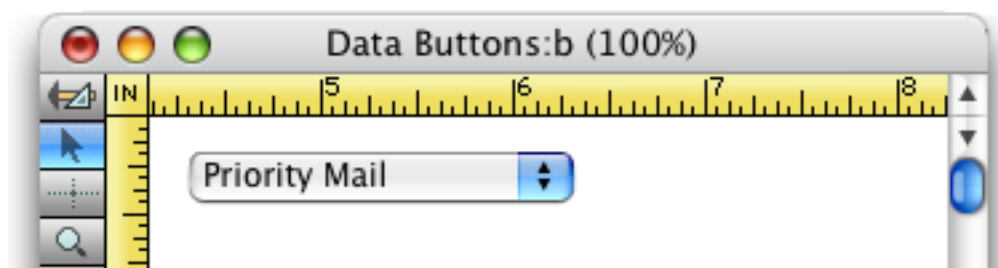


The final menu type is **Mac Pop-Up/Windows Combo Box**. The appearance and operation of this type depends on the type of computer being used. On a Macintosh computer this button will look and operate like a standard pop-up menu. On a Windows computer this button will look and operate like a Combo Box.

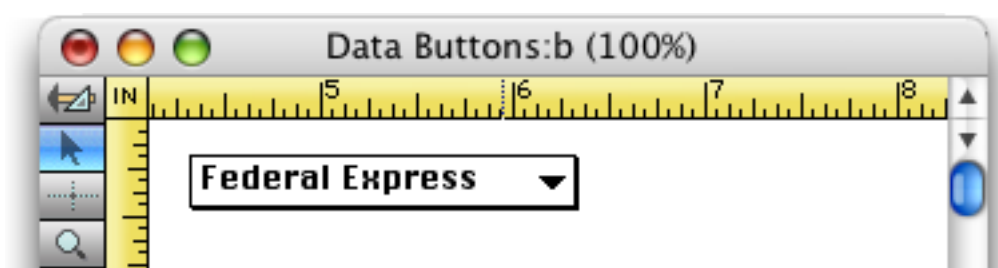
## Display Options

This section controls how the pop-up menu object is displayed on the form. If the **Show Value** checkbox is turned on, the field or variable associated with the pop-up menu will be displayed in the object. If the **Show Triangle** option is on, a small downward pointing triangle will appear on the right side of the pop-up menu.

You can only pick one of the last three options (or none). The **Standard Icon** option is the default choice, and displays the standard pop-up menu graphics for the operating system being used. Here is the standard icon for Mac OS X.



If the **Drop Shadow** checkbox is on, Panorama will automatically draw a drop shadow around the edges of the pop-up menu object. You can select whether the drop shadow is 1 or 2 pixels deep.



If you turn off all three of these options the pop-up menu will be invisible. This can be handy if you want the pop-up menu to appear with a custom image.

## Procedure

The pop-up menu can optionally trigger a procedure whenever the user makes a selection. This procedure can perform additional tasks that need to be done when a selection is made. Since the pop-up menu updates a field or variable, the procedure can simply read the menu choice from that field or variable.

## Pop-Up Menu Font, Size and Dimensions

If you are using the **Standard** menu style Panorama will use the standard menu font and size for the operating system in use (for example 13 point Lucinda Grande on Mac OS X).

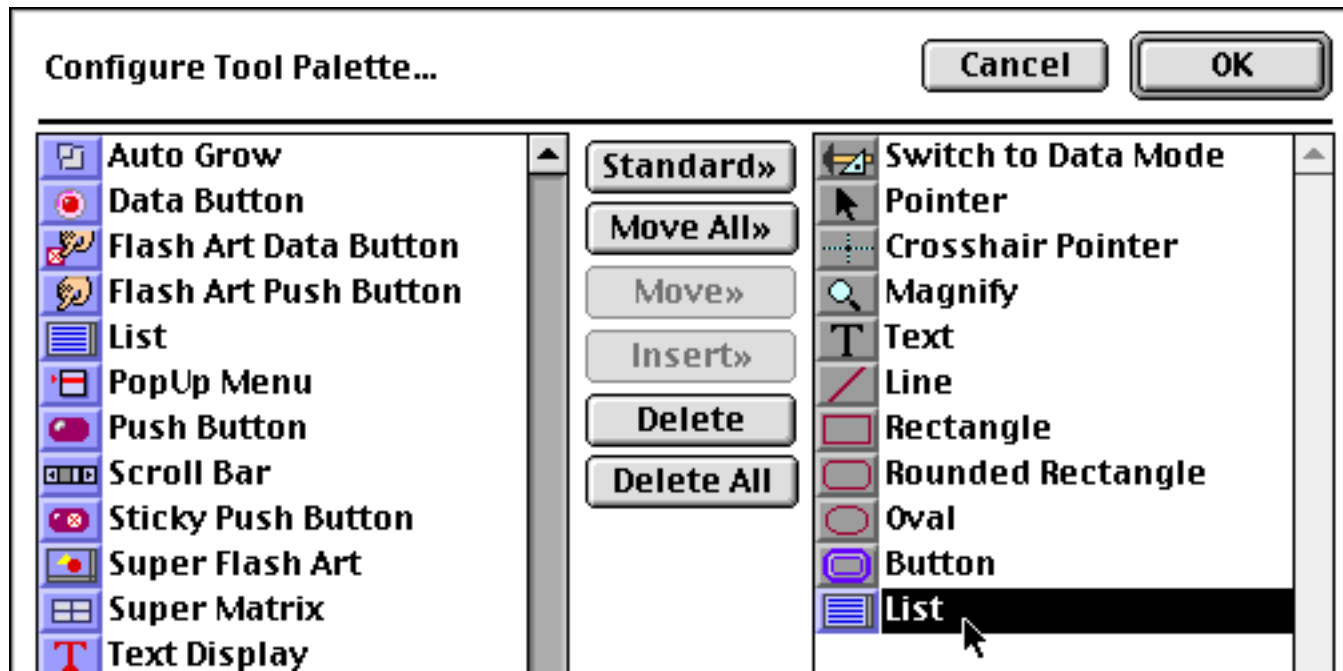
If you are using a **Multi-Column** menu you can use other fonts and point sizes. To get the most options in the least space, use Geneva 9 point (Alpine 9 point on a PC). For the best performance, you should stick to simple fonts and sizes that you actually have available in your system file (these sizes appear outlined in the Size menu.)

## List SuperObjects

Using a List SuperObject™ it's easy to add a scrolling list to a Panorama form or dialog. This is the same type of scrolling list used in the standard Open and Save dialogs. The scrolling list allows a large amount of data to be displayed in a small area. Using the scroll bars, the user can quickly locate the items they are interested in. Each Panorama scrolling list can be filled with information from a field or variable or from an entire database.

### Creating List SuperObjects™

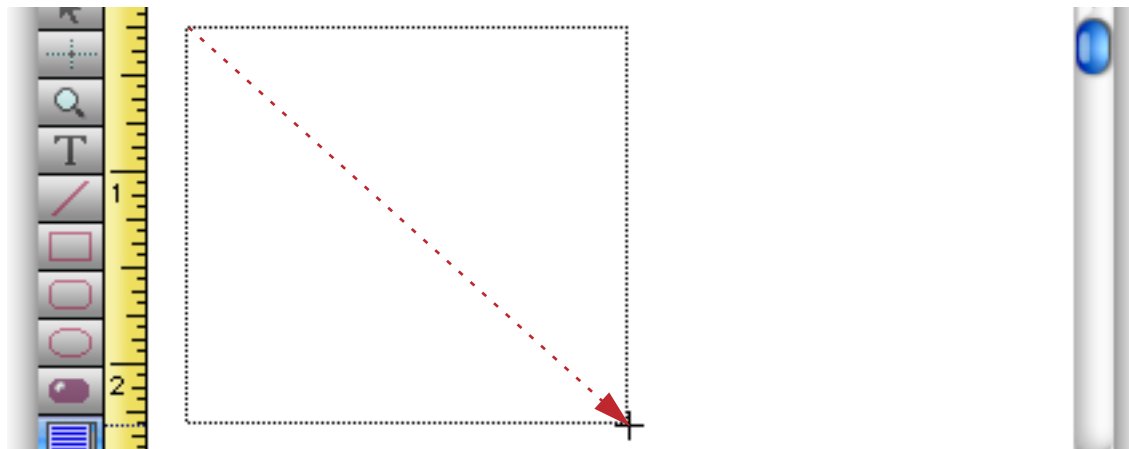
List objects are created just like any other SuperObject™. First make sure that the List tool is installed in the tool palette.



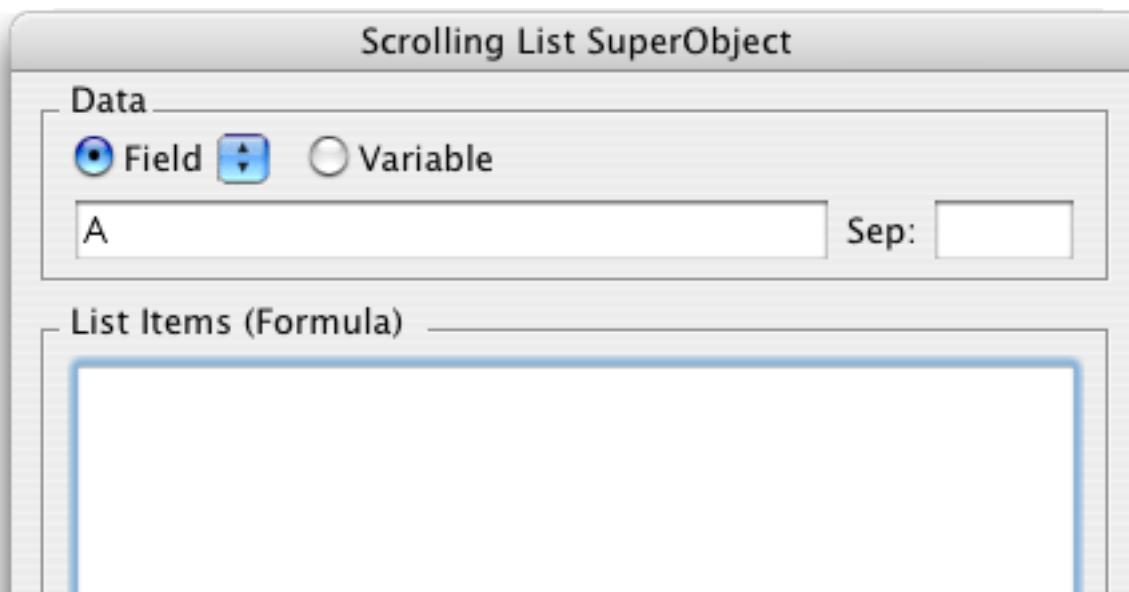
Select the List tool...



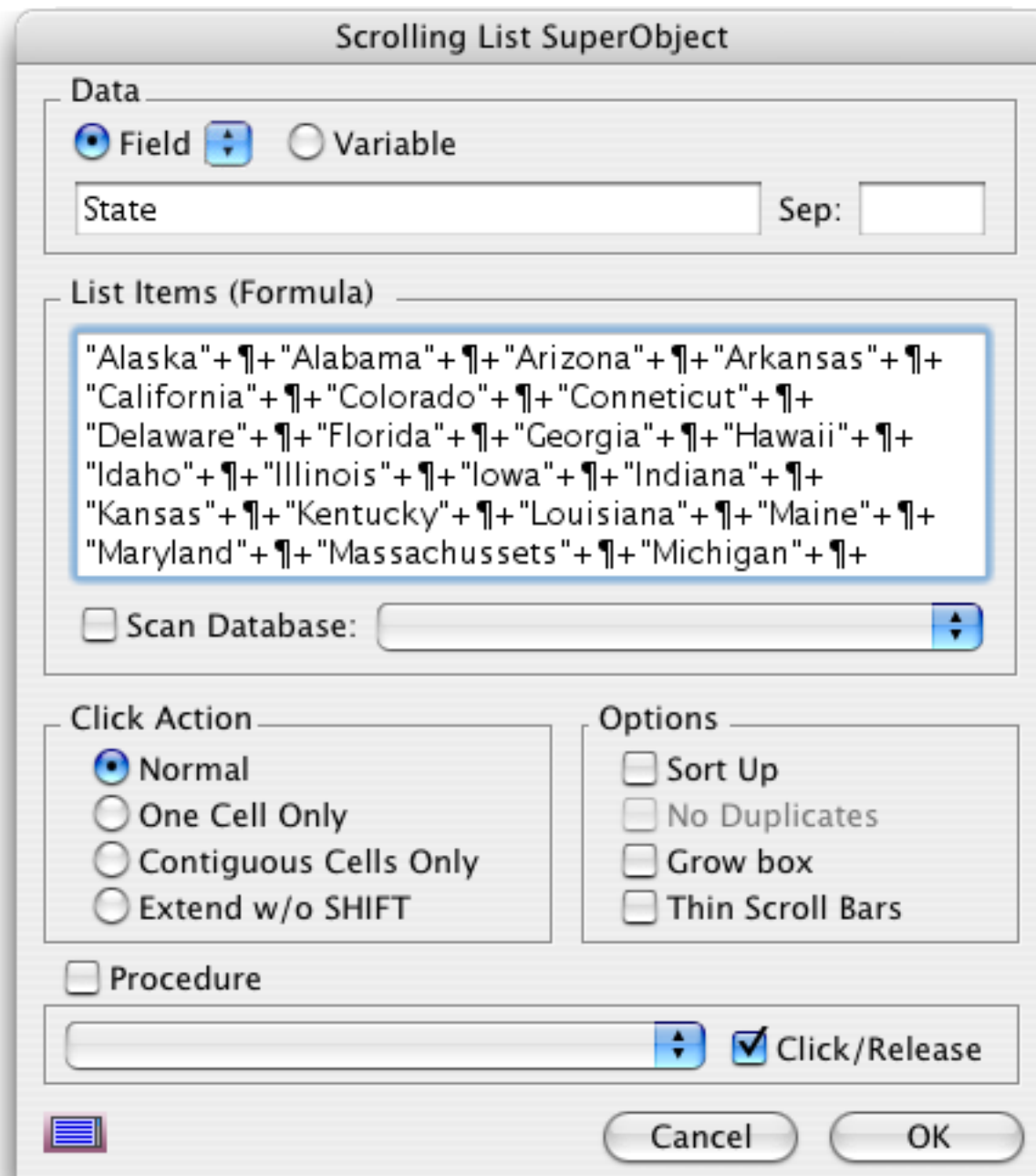
then drag the mouse across the form in the spot where you want the text to appear.



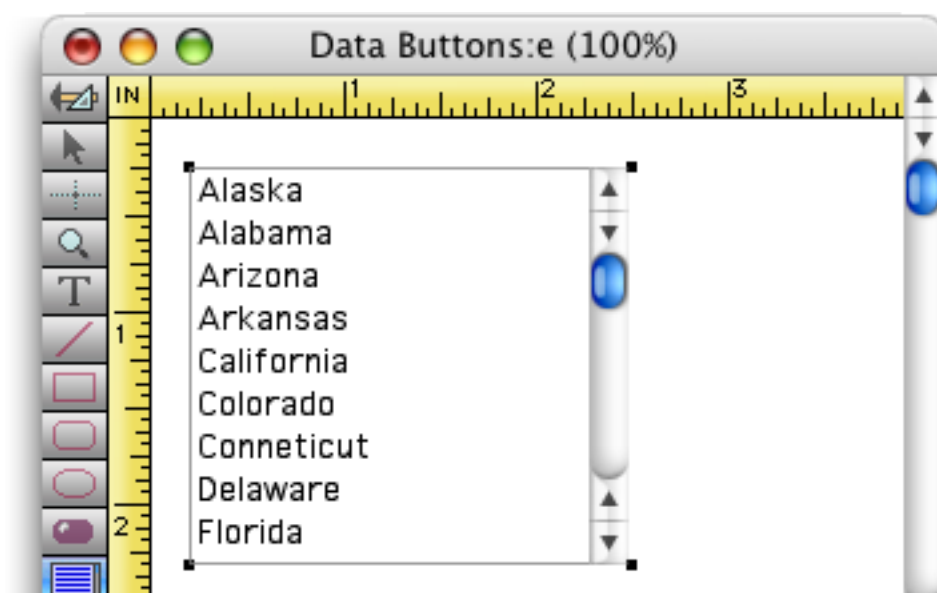
When you release the mouse, the List configuration dialog appears.



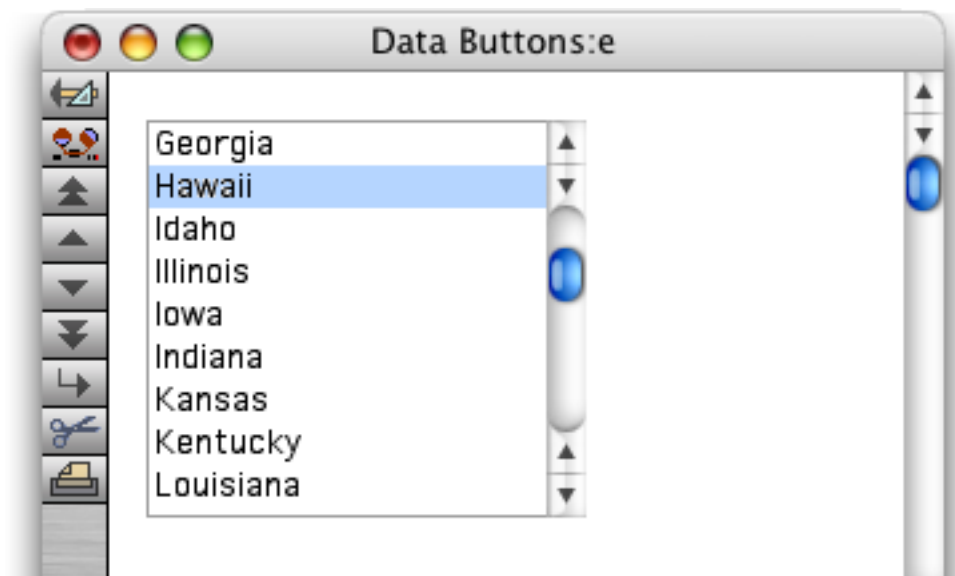
In some ways configuring a list is similar to configuring a pop-up menu. You need to specify a field or variable to hold the result, and you need to supply a formula to generate a list of items. Each item is separated by a carriage return. Here's a typical configuration to display a list of state names. (To type the ¶ symbol, press **Option-7** on the Macintosh and **Alt-0182** on the PC.)



Press **OK** to create the List object.

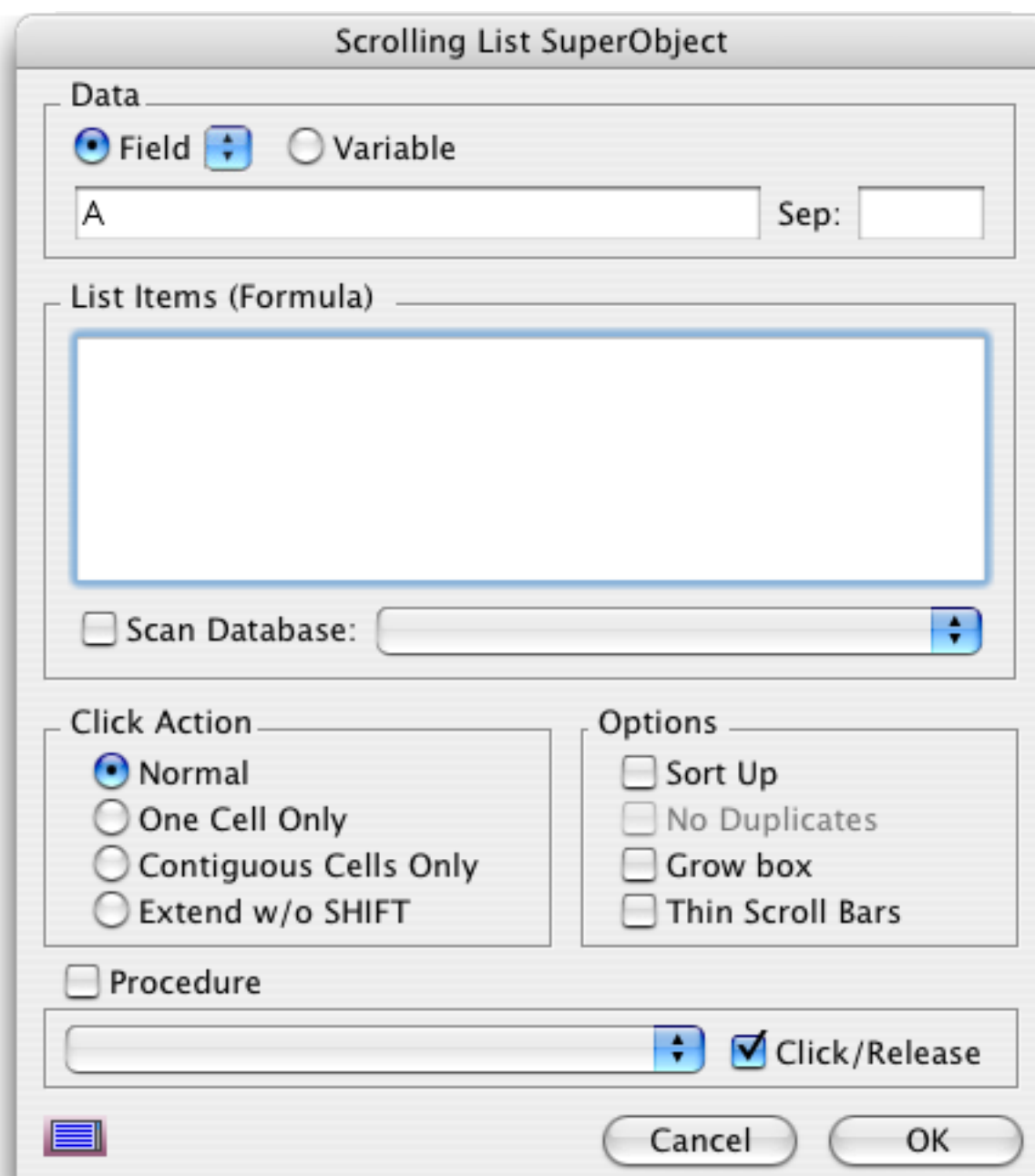


To actually use the list, switch to Data Access Mode. In this mode you can use the scroll bar to slide the contents of the list up and down, and click on list items to select them.



### List Options

The SuperObject™ List dialog is divided into several sections. This dialog allows you to configure the way the text is calculated and formatted. The options in this dialog are described in the following sections.

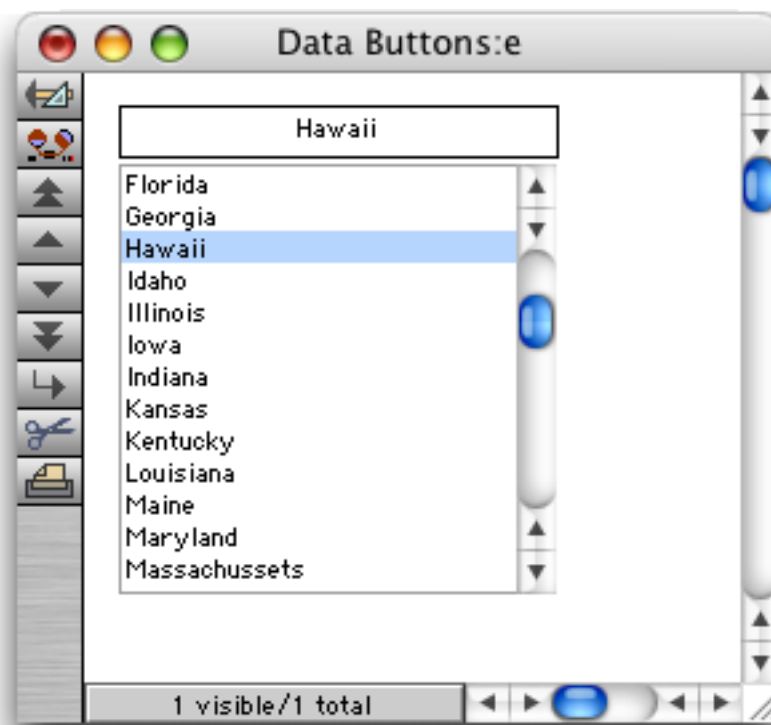


Once the options are set, press the **OK** button and the List object is ready to use. If you need to change the options later, double click on the List object to re-open the dialog.

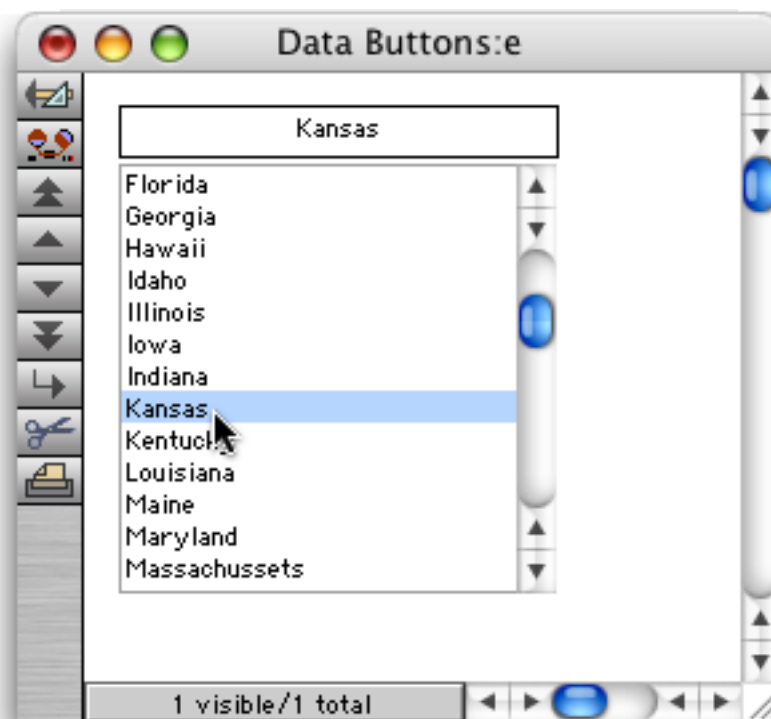
### Data

This section of the dialog specifies the field or variable associated with this list, if any. This field or variable doesn't contain the list itself, but only the selected value(s) within the list. Type the name of the field or variable into the box (or select the field name from the pop-up menu next to the **Field** radio button). If the list is associated with a variable that has not been created with a procedure, Panorama will automatically create a global variable with this name whenever the list appears. This global variable can be used in formulas and procedures just like any other global variable.

When first learning how to use the List SuperObject it's handy to include a Text Editor SuperObject on the form to allow you to see and modify the data value. In this example both the Text Editor and the List objects are displaying the same value — the field **State**.

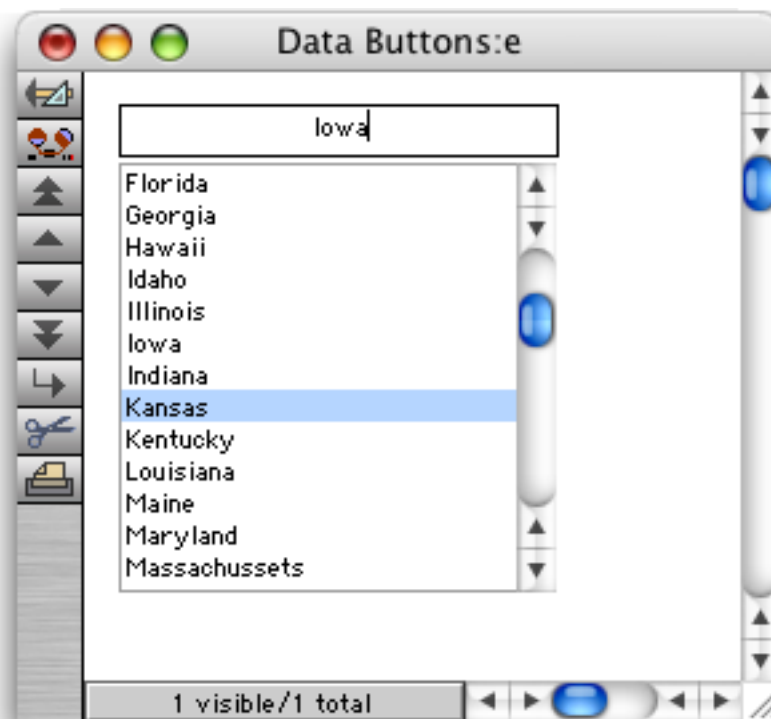


Clicking on an item fills the field with that value.

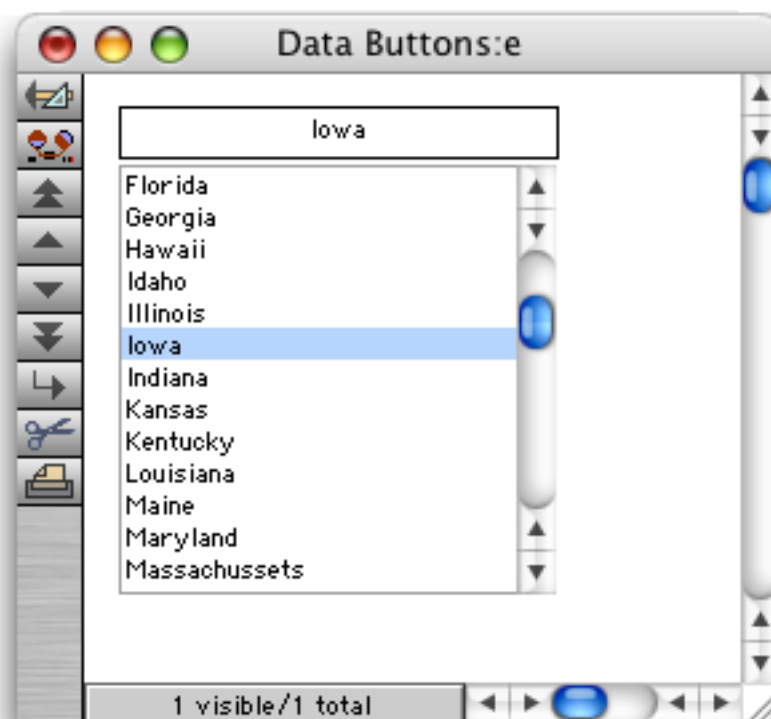




It's also possible to work this process in reverse. If you type a value into the Text Editor SuperObject the List will automatically select the corresponding item in the list, if any. For example, you could type **Iowa** into the field:



When you press **Enter** Iowa will be selected in the list.

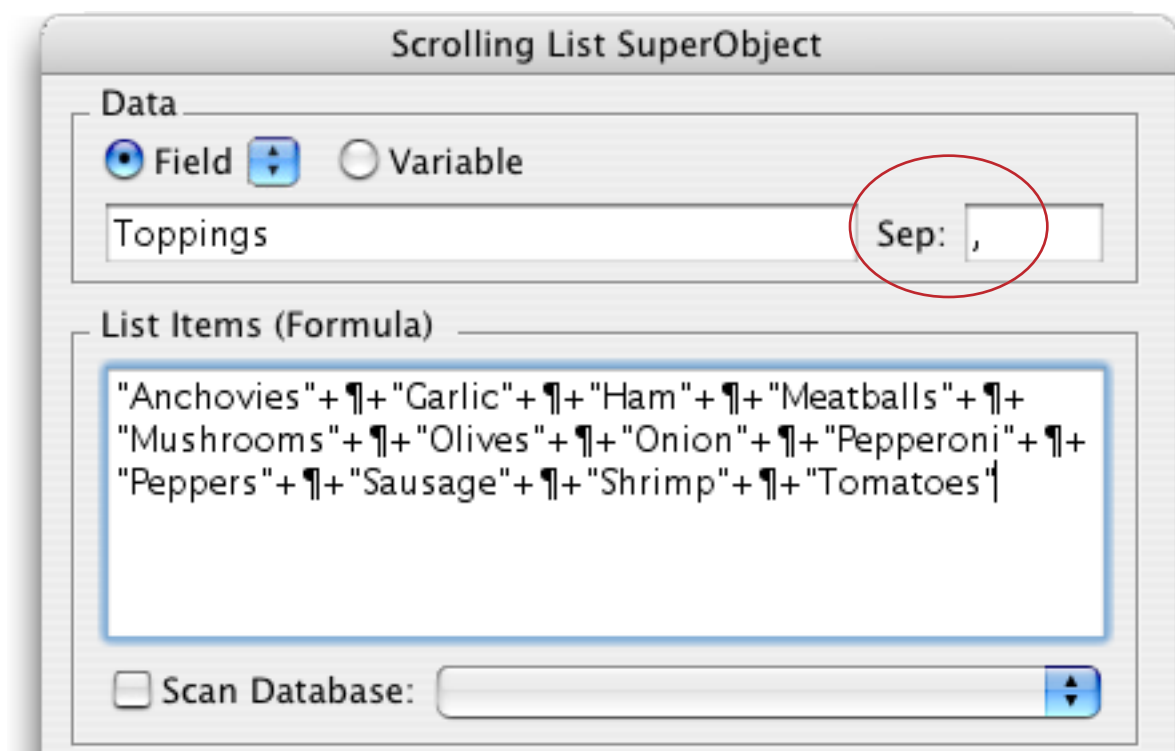


The item must be spelled exactly the same as it appears in the list, including capitalization. Only **Iowa** will work, not **iowa** or **IOWA**.

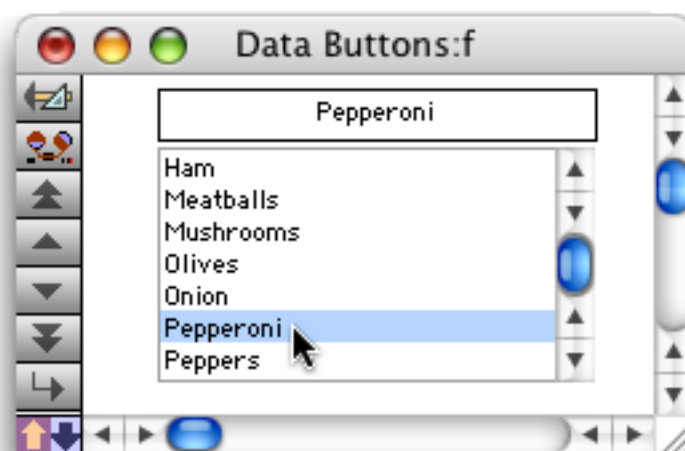
#### Sep

This is the separator text that will appear between each value if multiple items are selected in the list. Common separators include commas, spaces, slashes and hyphens. The separator may be up to 6 characters long. (Note: If the separator is left blank, Panorama will use a carriage return as the separator.)

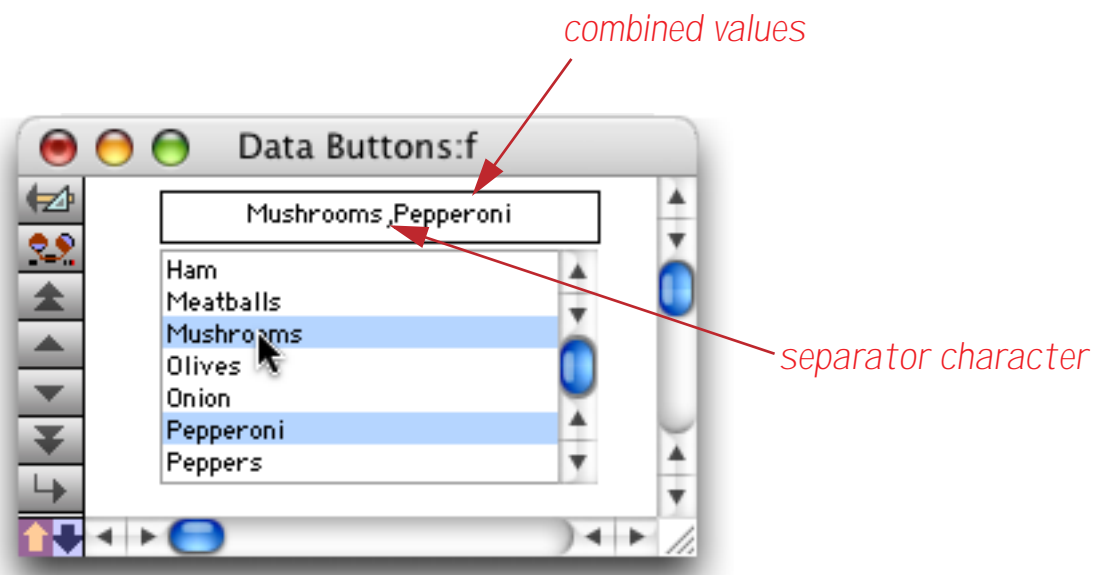
As an example of multiple items, consider pizza toppings. A pizza may have one, two, three or more toppings, or even none at all. You can create a list that shows all the different pizza toppings. In this example the separator character is a comma.



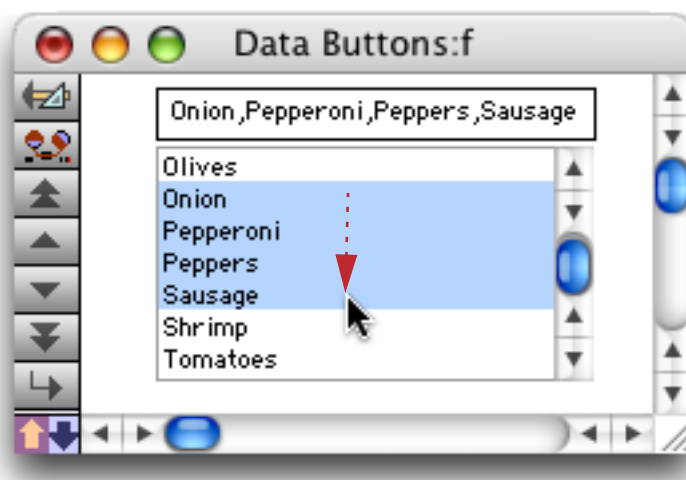
When you click on a single topping it appears in the field, just as in the previous example.



By holding down the **Command** key (Mac) or the **Control** key (PC) you can click on and select additional items, as shown below. The data field (or variable) will contain all of the selected items, with each item separated by a comma (or whatever separator character you have specified.) The values will always be in the same order as they appear in the list, no matter what order you click on them (unlike a group of radio buttons — see “[Multiple Value Button Groups](#)” on page 365).



You can continue to select additional items if you wish. You can also de-select an item by holding down the **Command** (Mac) / **Control** (PC) key and clicking on the selected item. Another option is to hold down the **Shift** key and drag across several items to select them all.



You can use any separator character you want. If you leave the **Sep** option empty, Panorama will use a carriage return as the separator. In other words, each item will be on its own separate line.

## Database

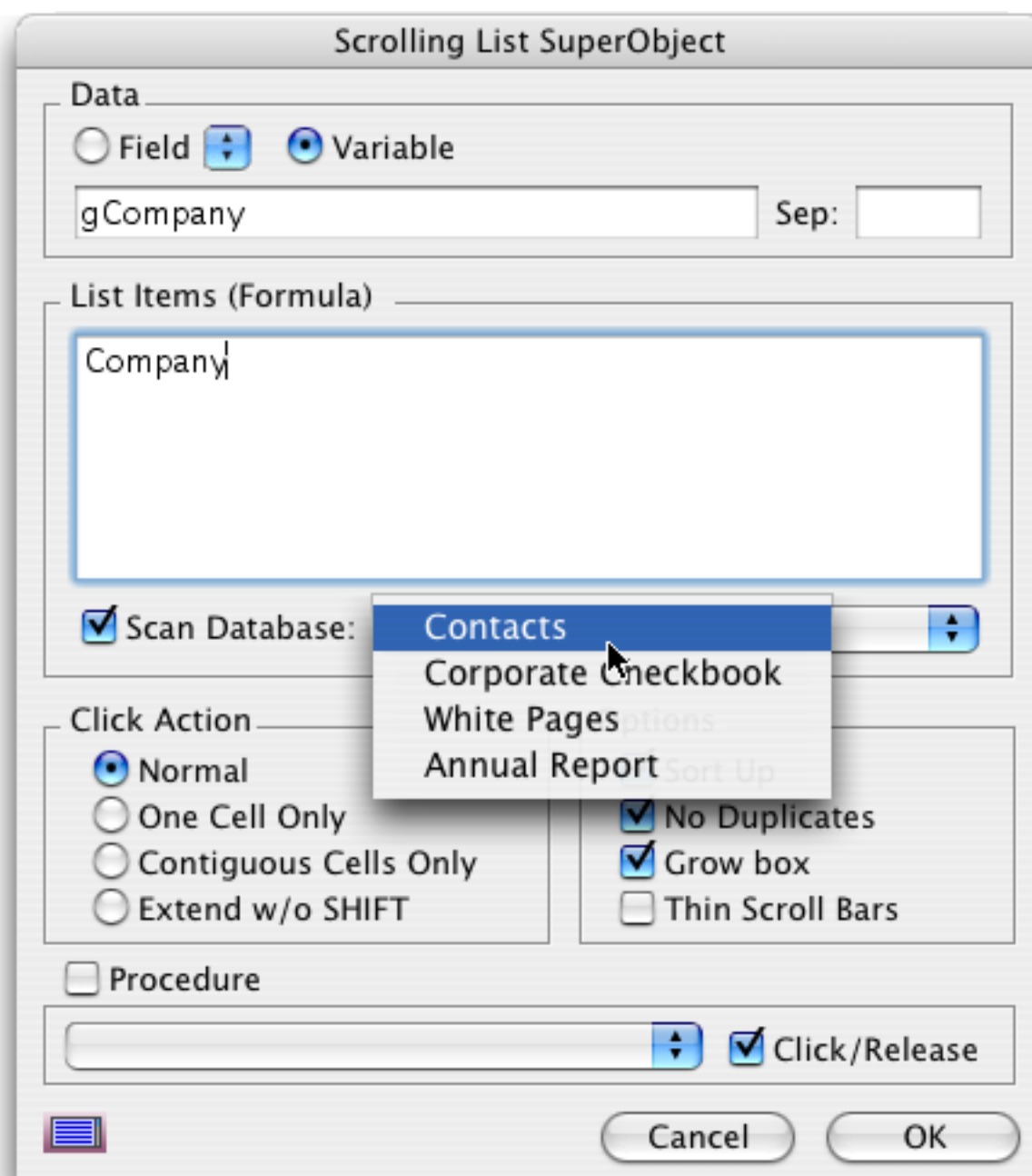
There are two ways that Panorama can build the list of items: it can use a formula to build the list or it can scan a database to build the list. So far all of the examples have shown building the list with a formula. If you want the list built by scanning an entire database, select the name of the database from the pop-up menu in this section (the database must be open). To illustrate this feature we'll use this contact database.



Title	Company	Address	City	State	Zip	Country
Sales Manager	Acme Widgets	12 Harmony Lane	Huntington Beach	CA	92648	
Owner	Brian's Appliances	1844 Tiburon	Hollister	CA	95023	
		182 Dell Rd	Northbrook	IL	60062	
Sales Manager	Jim's Appliances	58272 Auburn Rd	Fort Wayne	IN	46825	
Vice President	D.S. Plumbing	683 Elm St	Batavia	IL	60510	
Sales Manager	Latham Video	4792 Latham	Mountain View	CA	94041	
President	P.T. Plumbing	1009 Secret Bay	Davis	CA	95616	
		S.W. Plumbing	Fountain	CO	80817	
Vice President	Evanston Lumber	498 Noyes	Evanston	IL	60201	
		3050 North Main	Sand Springs	OK	74063	

107 visible/107 total

Create a List SuperObject the usual way as described earlier in this Chapter. Select the database to be scanned from the pop-up menu of open databases. In this case you'll select the name of the current database.



**Scrolling List SuperObject**

**Data**

Field  Variable

gCompany Sep:

**List Items (Formula)**

Company

Scan Database: **Contacts**

- Corporate Checkbook
- White Pages
- Annual Report

**Click Action**

Normal

One Cell Only

Contiguous Cells Only

Extend w/o SHIFT

No Duplicates

Grow box

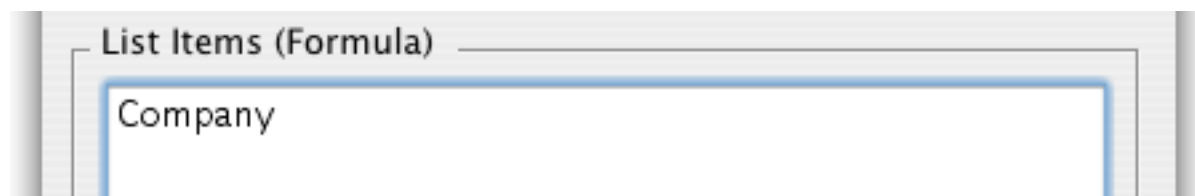
Thin Scroll Bars

Procedure

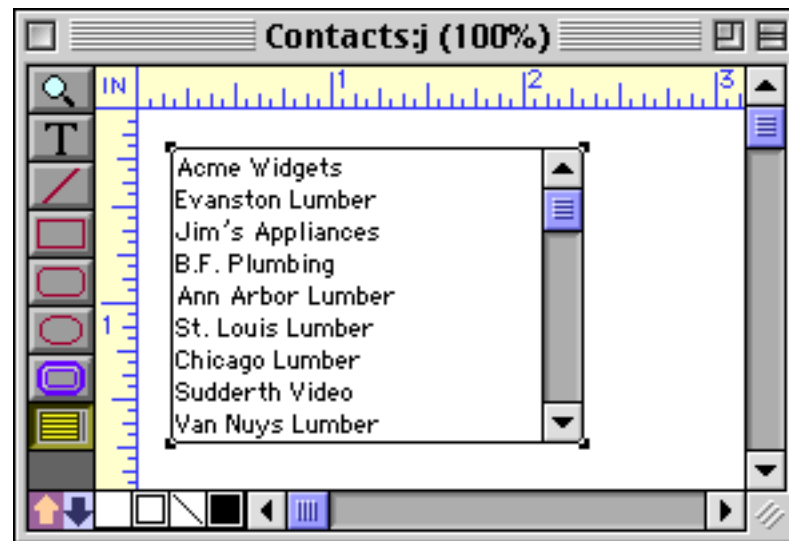
Click/Release

Cancel OK

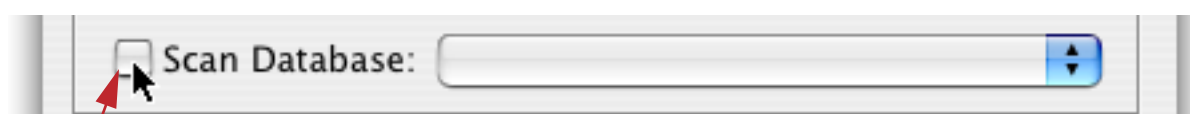
In the formula field you must enter a formula that will be used to process each field within the database. Usually this is simply a field name, which can either be typed in or selected from the **Field** menu. For this first example we'll build a list of company names.



When you press **OK** the list appears, already filled in with the items scanned from the database.

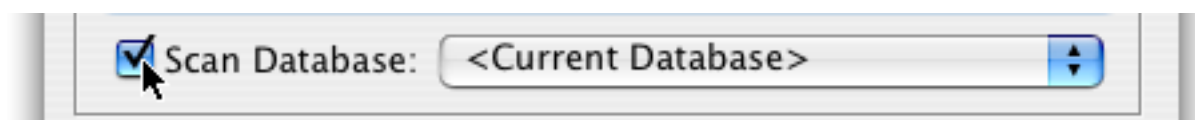


If the database name ever changes, this list will stop working. Here's how this problem can be fixed. Open the configuration dialog again (by double clicking on the object). Click on the **Database** checkbox to disable the database scanning procedure.



*Click to turn off database scanning*

Now click the checkbox again to turn database scanning back on. As you can see, this enables scanning of the current database, no matter what the name is.



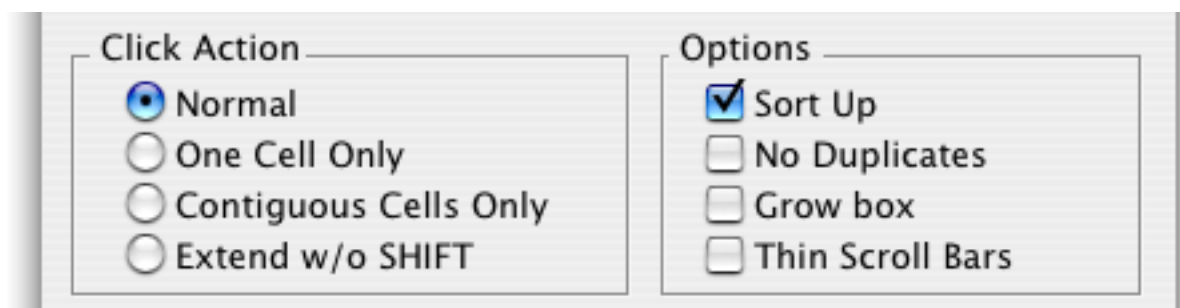
This trick only works with the current database. If the list is built from another database you'll have to make sure that the database name doesn't change (or if it does, you'll have to open the configuration dialog and adjust the name to make the list work again).

### Sort Up

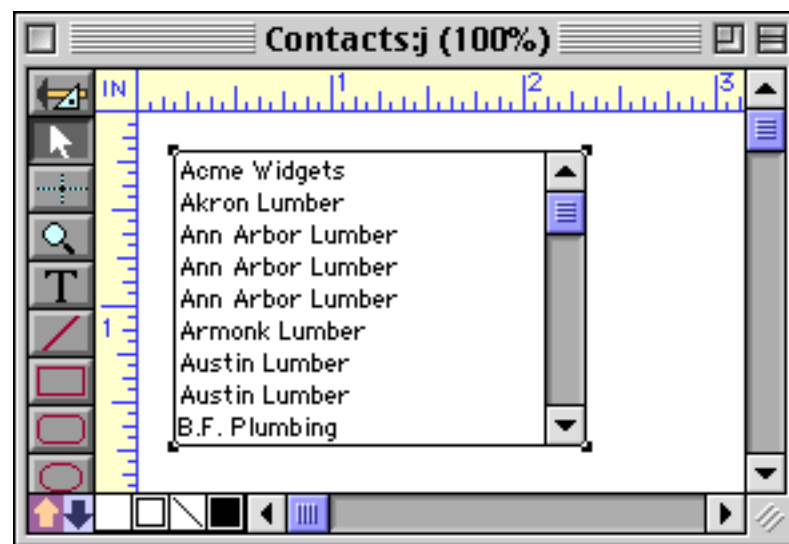
If this checkbox is turned on, the list of items will be sorted in ascending alphabetical order, otherwise the list will be displayed in the order the data was scanned. For example, the list of companies built in the previous example displays the company names in the order they appear in the database, which is definitely not alphabetical.



To display the list in order, enable the **Sort Up** option.



Here's the alphabetized list.

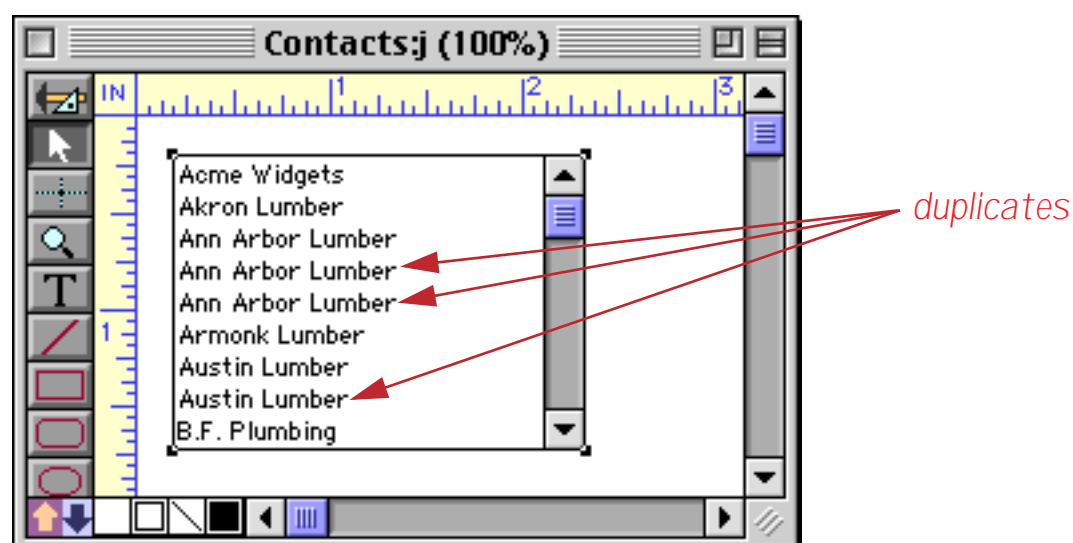


The **Sort Up** option works both with database scanning and when the list is generated by a single formula.

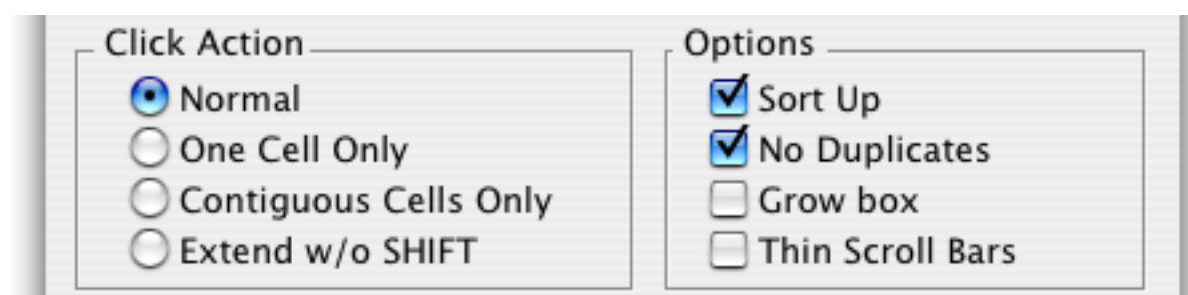


### No Duplicates

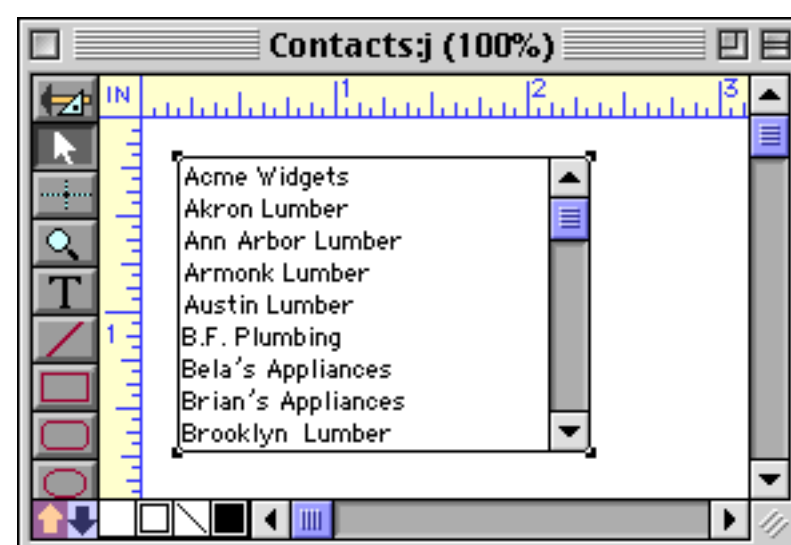
If this checkbox is turned on, duplicate data items will be eliminated from the list of items. (This option is not available if the **Sort Up** option is not also turned on.) For example, the list generated in the previous example contains many duplicate company entries.



To automatically remove the duplicate items enable the **No Duplicates** option.



Here is the revised list, without the duplicates.



### Formula

This section of the dialog contains the formula used to extract the data items from the database, field or variable. See "[Building the List](#)" on page 395 for details on how this formula is used.

### Grow Box

This option makes the scroll bar shorter to reserve space for a grow box in the lower right hand corner of the object.

### Thin Scroll Bars

If this option is checked the scroll bar will be narrower (11 pixels wide instead of 16 pixels).

### Procedure

This pop-up menu allows you to specify a procedure that will be triggered every time the user clicks on an item in the list.

### Click/Release

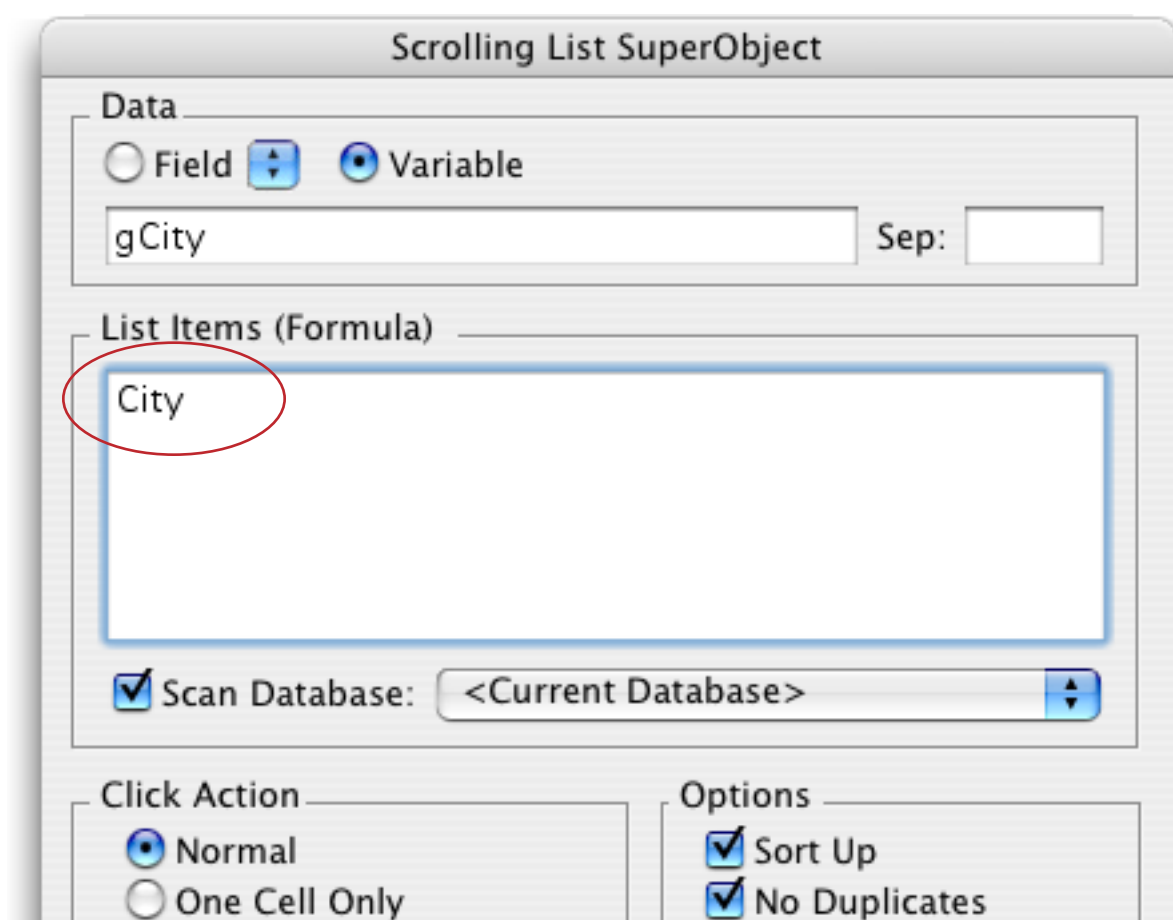
This option controls when the procedure (if any) is triggered. Normally, the **Click/Release** option is enabled and the procedure is triggered when you release the mouse. If the **Click/Release** option is disabled the procedure will be triggered immediately, as soon as you click on the list. This means that you cannot hold down the **Shift** key and drag to extend the selection (see “[Grow Box](#)” on page 394). Turn off the **Click/Release** option if you want to trigger a procedure to drag list items (see “[Using Drag and Drop to Change the Order of Items in a List](#)” on page 1860).

### Building the List

There are two ways that Panorama can build the list of items: it can scan a single field or variable, or it can scan an entire database. Panorama normally builds the list once when the form containing the list is first opened. If you need to update the list or change it later, you’ll need to send the list a command with a procedure (see “[List SuperObject™ Commands](#)” on page 1856 of the **Panorama Handbook**).

**Scanning a single formula:** If this method is selected, Panorama calculates the result of the formula, then scans the result to separate it into individual items for the list. Each line (separated by carriage return) in the result is treated as a separate item. This is the same method used by the Pop-Up Menu SuperObject to build a list; see “[The Pop-Up Menu Formula](#)” on page 378 for tips and tricks for setting up this formula.

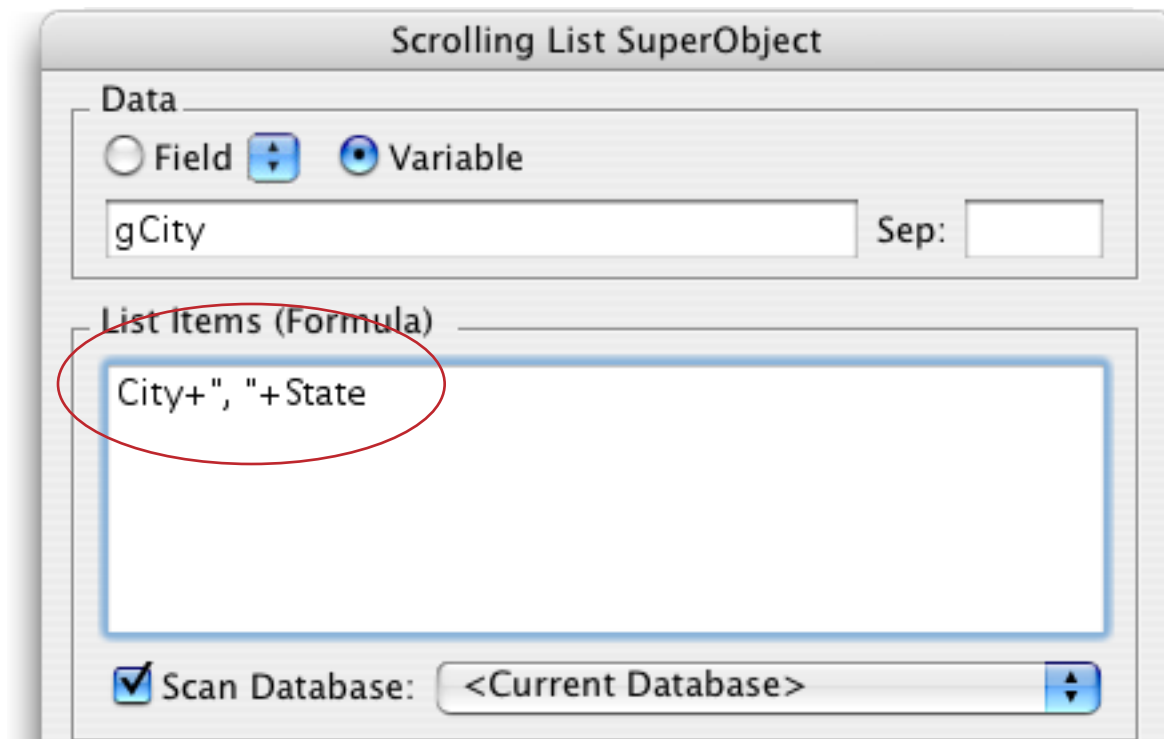
**Scanning an entire database:** If this method is selected, Panorama will scan the entire specified database, building up the list one record at a time as it scans. The formula determines what information is extracted from each record in the database and places it in the list. For example, suppose you want to build a list of cities extracted from an address database. For this application, the formula would be simply:



For this application you would probably want to turn on the **Sort Up** and **No Duplicates** options, so that each city will be listed only once.



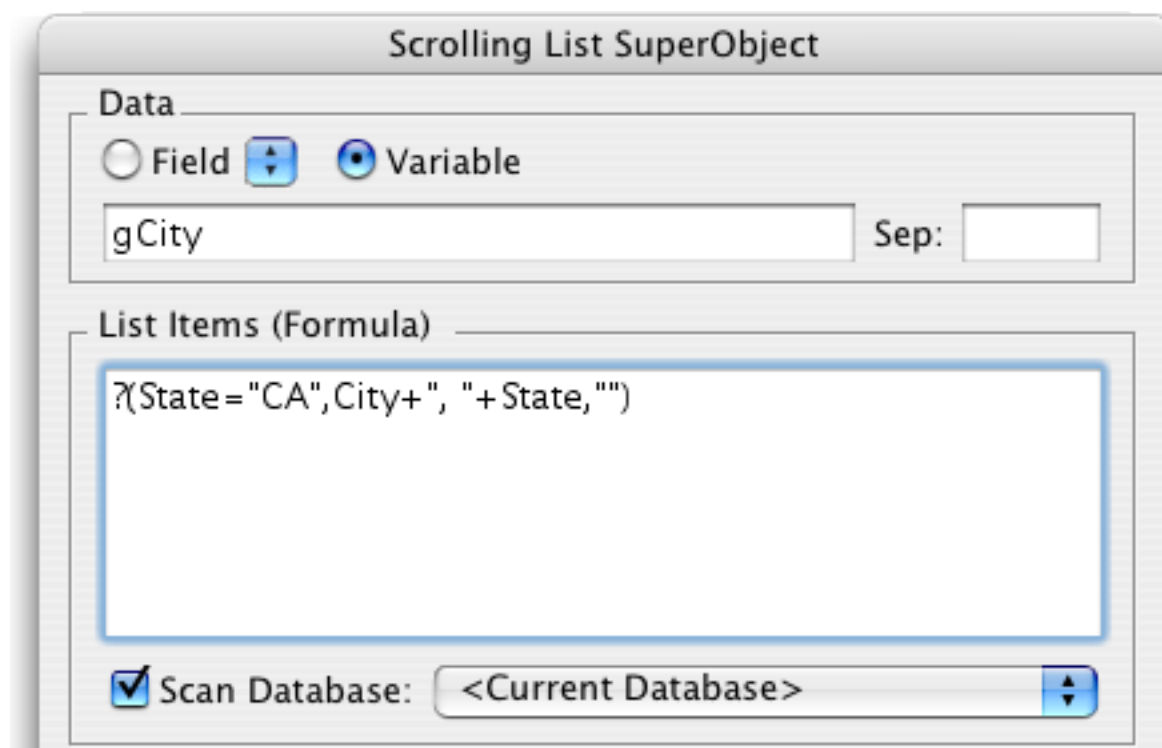
If you wanted both the city and state to appear in the list, a more complex formula could be used:



Here's the revised list.



If you want to build a list of only part of the database, use the `?()` function to select only the information you want (see “[The ? Function](#)” on page 1266). If a record should not be included in the list, the result of the formula should be an empty string (“”). The formula below will fill the list with cities in California.



All other states will be left off the list.



### Maximum List Size

The List SuperObject is capable of handling lists up to a few thousand items. For large lists, you may need to increase your scratch memory allocation. However, keep in mind that the List SuperObject is not intended to replace Panorama’s normal database operation. The performance of the list (speed) will degrade as the size of the list increases. Very large lists are also difficult for the user to handle. It’s usually best to keep the list size to several hundred items or less. If necessary, you should split the list into multiple lists: for example by state, by price, or even by starting letter (A-Z).



# Chapter 18: Form Goodies



This chapter covers some cool things you can do with forms that don't fit into any particular category. **View-as-List** forms allow you to display multiple records at a time, instead of just a single record at a time like a normal form. **Elastic Forms** allow you to create forms that automatically stretch to fit any window size. **Super Matrix** objects allow you to quickly build repeating arrays like monthly calendars and photo arrays.

## View-As-List Forms

Panorama allows you to set up blank forms as individual pages or as a continuous sheet (**view-as-list**). When forms are set up as individual pages you see one record at a time. You can flip through the records just as you would shuffle through a stack of paper forms.

A **view-as-list** form displays data as a continuous sheet, as shown below. Instead of flipping from record to record, you scroll up and down through the data in a manner similar to the data sheet. However, unlike the data sheet, a view-as-list form allows you to arrange the data any way you like, and even include graphics in the display. On the other hand, view-as-list forms are slower than the data sheet (because of the overhead in displaying the graphics) and they are much more work to set up.

Date	Num/Pay To (Category)	Amount	Balance
01/17/99	1913 California Capitol (Insurance)	28.00	35,023.26
01/17/99	1914 U S Postmaster (Postage)	75.00	34,948.26
01/17/99	1915 Sacramento Bee (Advertising)	795.00	34,153.26
01/18/99	DEPOSIT	+3,846.32	37,999.58
01/22/99	1916 Walthers (Purchases)	12,463.00	25,536.58
01/22/99	1917 Blue Cross Of Calif (Insurance)	279.03	25,257.55
01/22/99	1918 Sherman Douglas Ins (Insurance)	418.60	24,838.95
01/22/99	1919 Cannon Astro (Office Supplies)	145.72	24,693.23
01/25/99	1920	1,885.40	22,807.83

411 visible/411 total

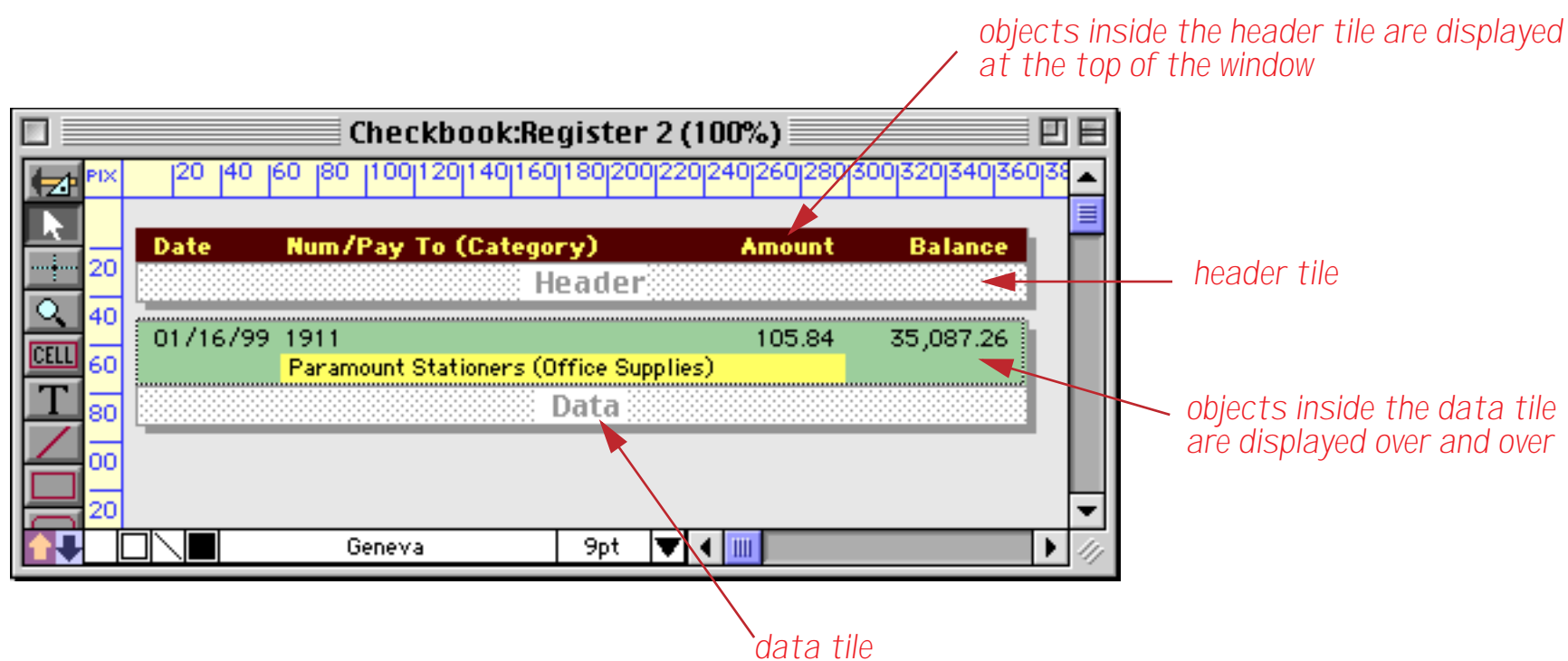


## How View-As-List Forms Work

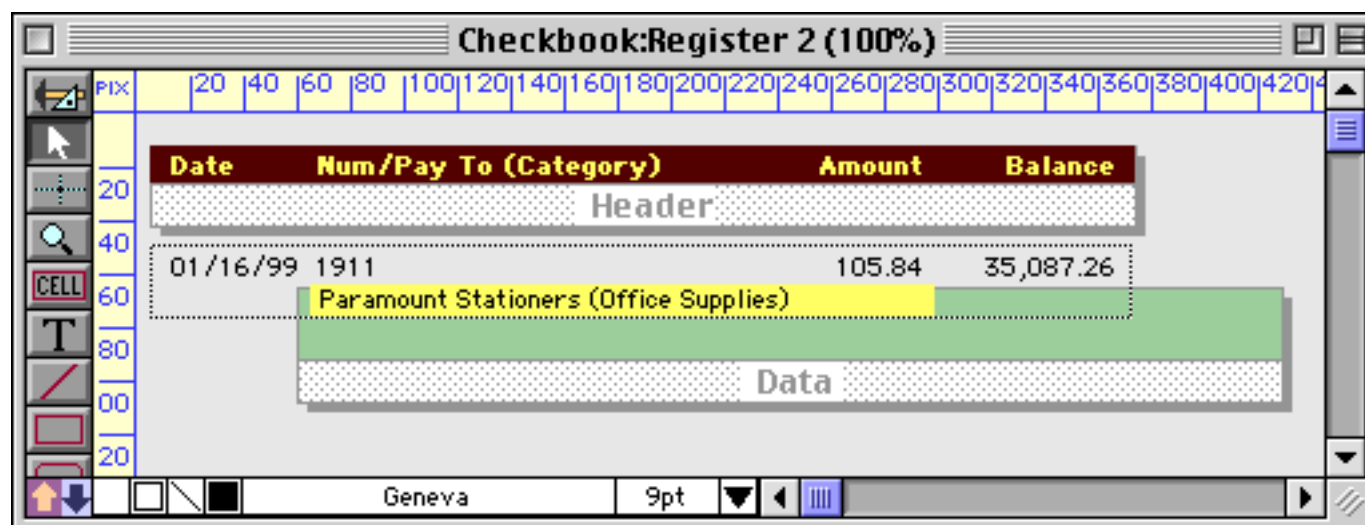
Panorama displays a View-As-List form by taking a collection of graphic objects and displaying them over and over again, once for each record. For the form in the previous section the collection of graphic objects looks like this.

01/16/99	1911	105.84	35,087.26
Paramount Stationers (Office Supplies)			

As you can see, this collection of objects is just a small portion of a form. How does Panorama know which collection of objects to include? You tell it by enclosing the objects in a special object called a **tile**. Panorama has over 40 different kinds of tiles, but for now we are interested in just two types—**data tiles** and **header tiles**.



Panorama doesn't care what the exact location of the tile is. It simply looks to see what is contained within the tile. For example, you could move the tile to a new position like this (see "[Working with Tiles](#)" on page 1040).



In Data Access Mode this new configuration will look like this.

Date	Num/Pay To (Category)	Amount	Balance
	Paramount Stationers (Office Supplies)		
	California Capitol (Insurance)		
	California Capitol (Insurance)		
	U S Postmaster (Postage)		
	Sacramento Bee (Advertising)		
	DEPOSIT		

The height of the Data Tile controls the spacing of the records. For example, we can reduce the spacing by reducing the height of the tile (see “[Working with Tiles](#)” on page 1040).

Date	Num/Pay To (Category)	Amount	Balance
01/17/99	1913 California Capitol (Insurance)	28.00	35,023.26

Here's what this configuration looks like.

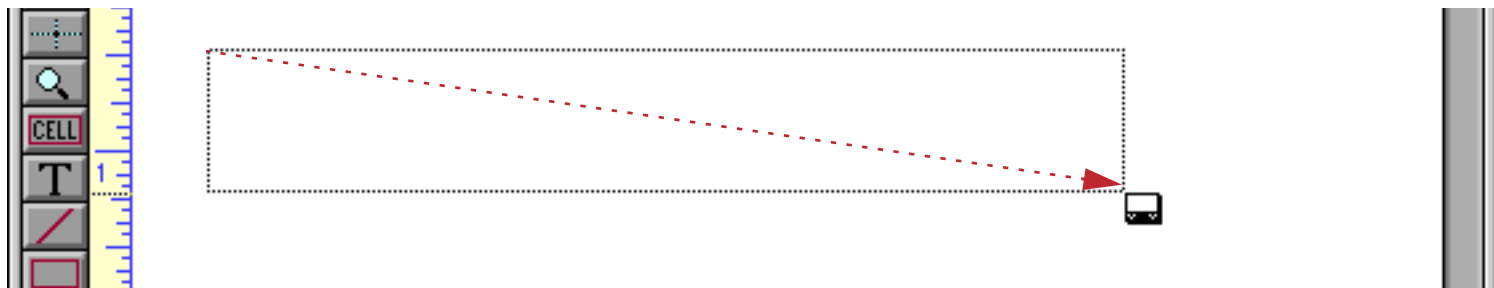
Date	Num/Pay To (Category)	Amount	Balance
	Paramount Stationers (Office Supplies)		
	California Capitol (Insurance)		
	California Capitol (Insurance)		
	U S Postmaster (Postage)		
	Sacramento Bee (Advertising)		
	DEPOSIT		
	Walthers (Purchases)		
	Blue Cross Of Calif (Insurance)		
	Sherman Douglas Ins (Insurance)		
	Cannon Astro (Office Supplies)		
	Walthers (Purchases)		
	Nehs (Office Supplies)		

## Creating a View-As-List Form

The first step in creating a view-as-list form is to create a Data Tile. To do this, select the Tile tool.



Now drag the mouse across the form to define the location and size of the tile.

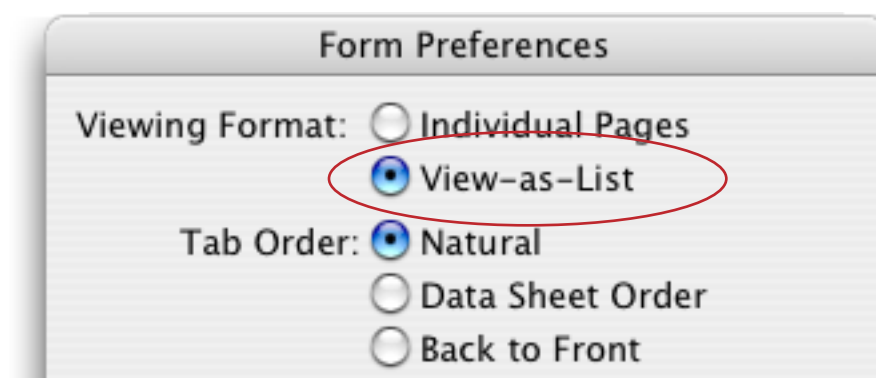


When you release the mouse Panorama will create the tile.

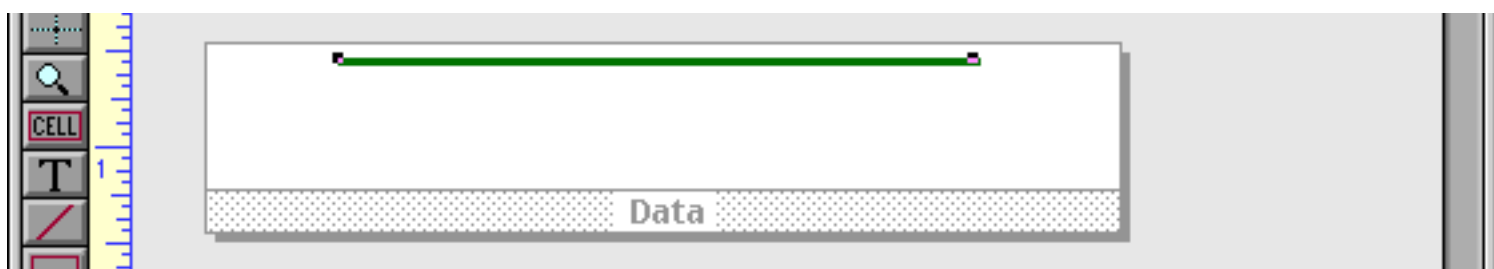


As you can see, the area outside of the form turns gray. This indicates that any object placed in this area will not be included as part of the form when in Data Access Mode.

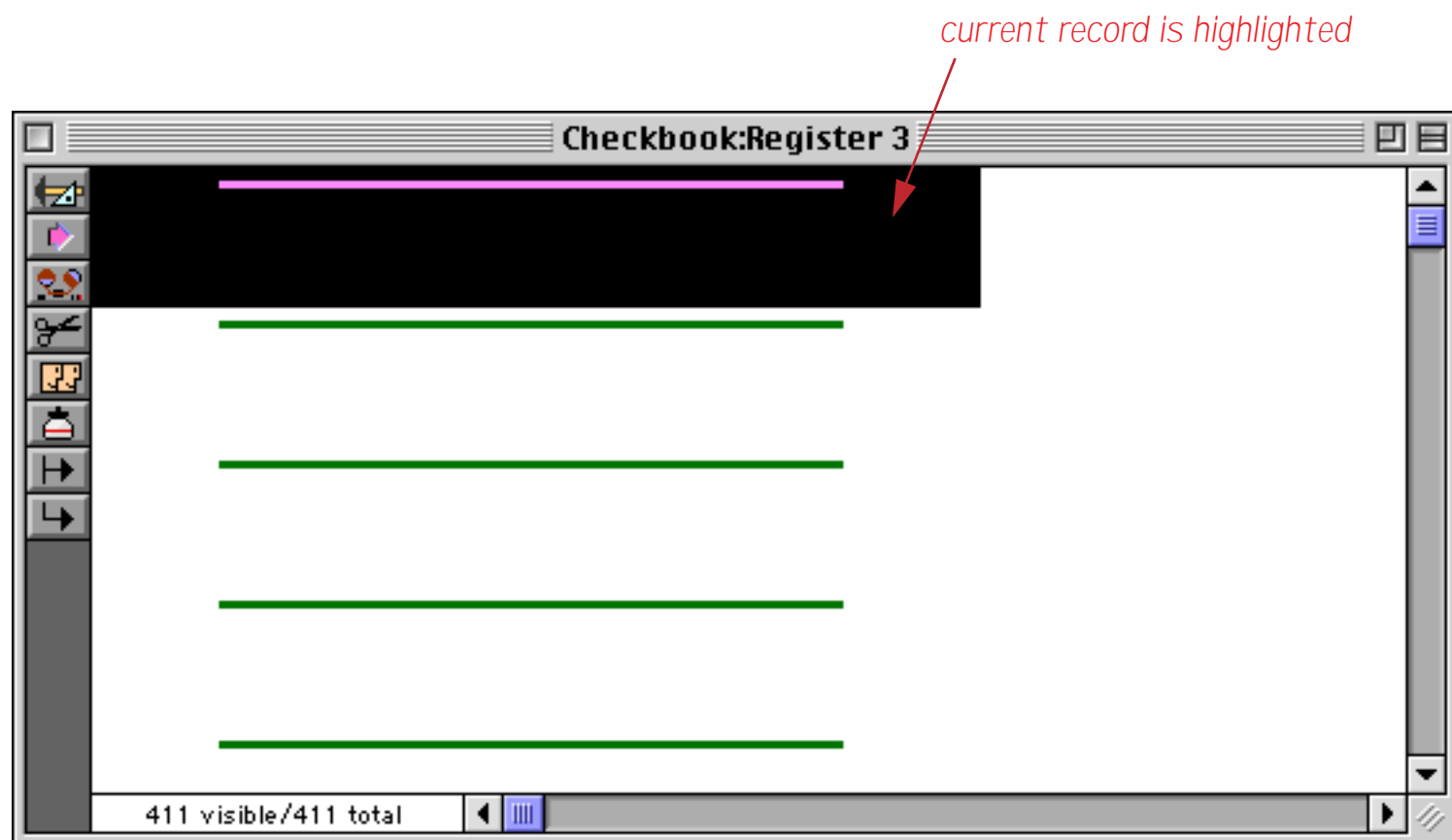
The next step is to enable the **View-As-List** option in the **Form Preferences** dialog. You'll find this dialog in the Setup menu.



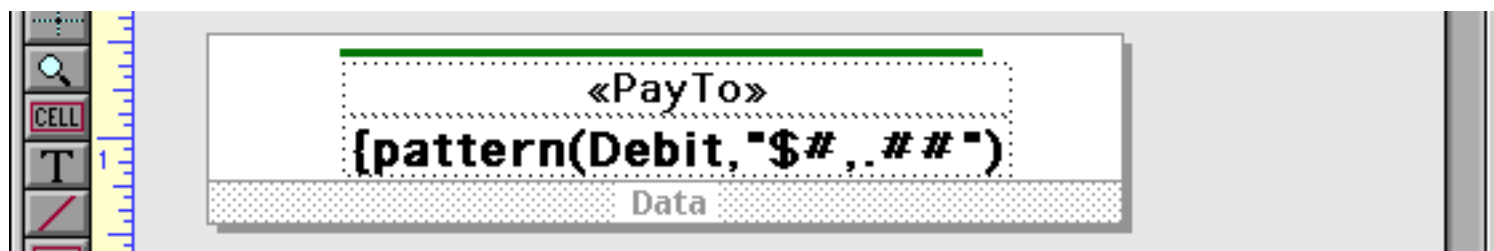
Press the OK button to confirm the new preferences. Now we're ready to start adding graphics to the form. We'll start by using the **Line** tool to add a horizontal line across the top of each record.



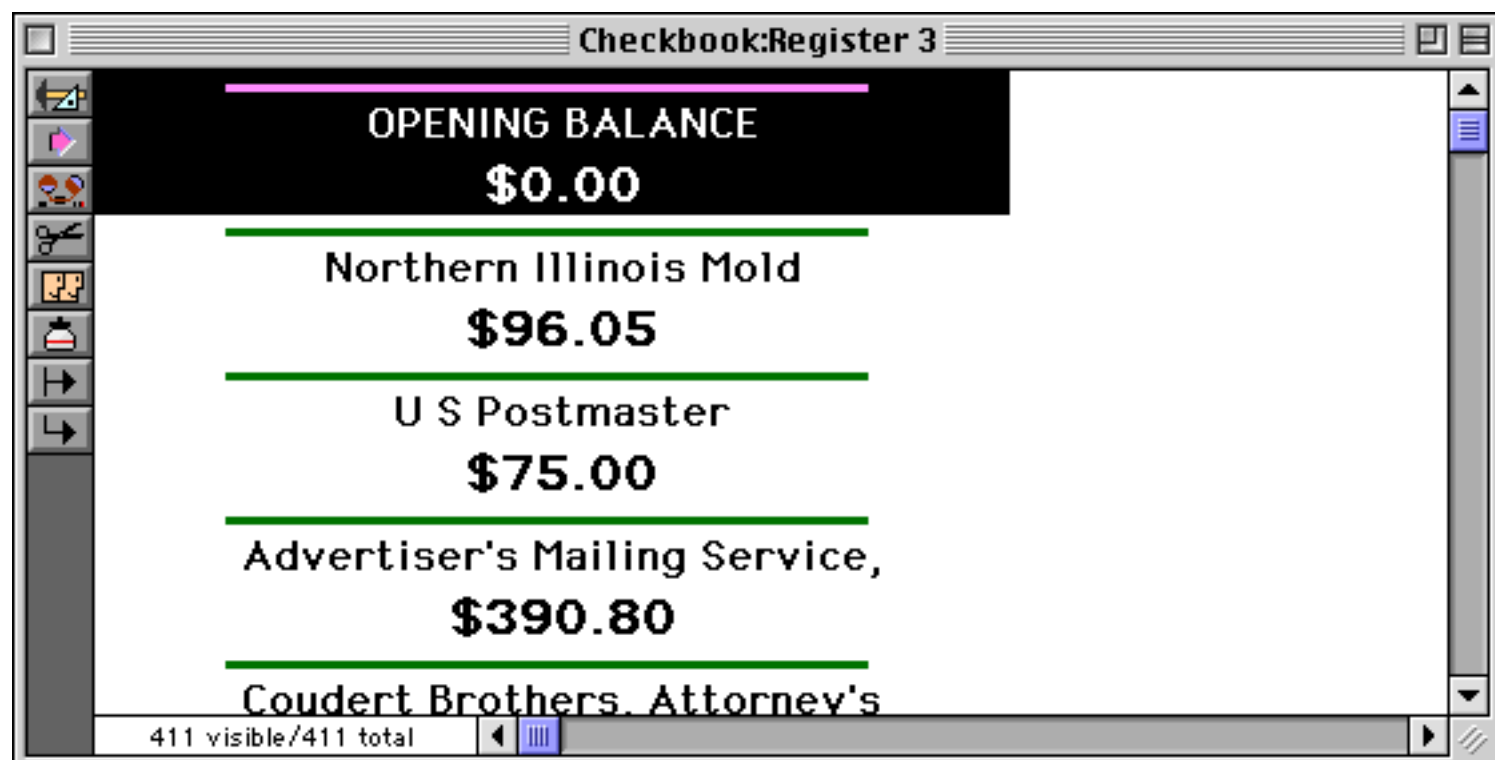
Switch to Data Access Mode to see what this configuration looks like. The line is repeated once for each record. (You'll also notice that the current record is highlighted in reverse - see "[Buttons on a View-As-List Form](#)" on page 417 for more information about this.)



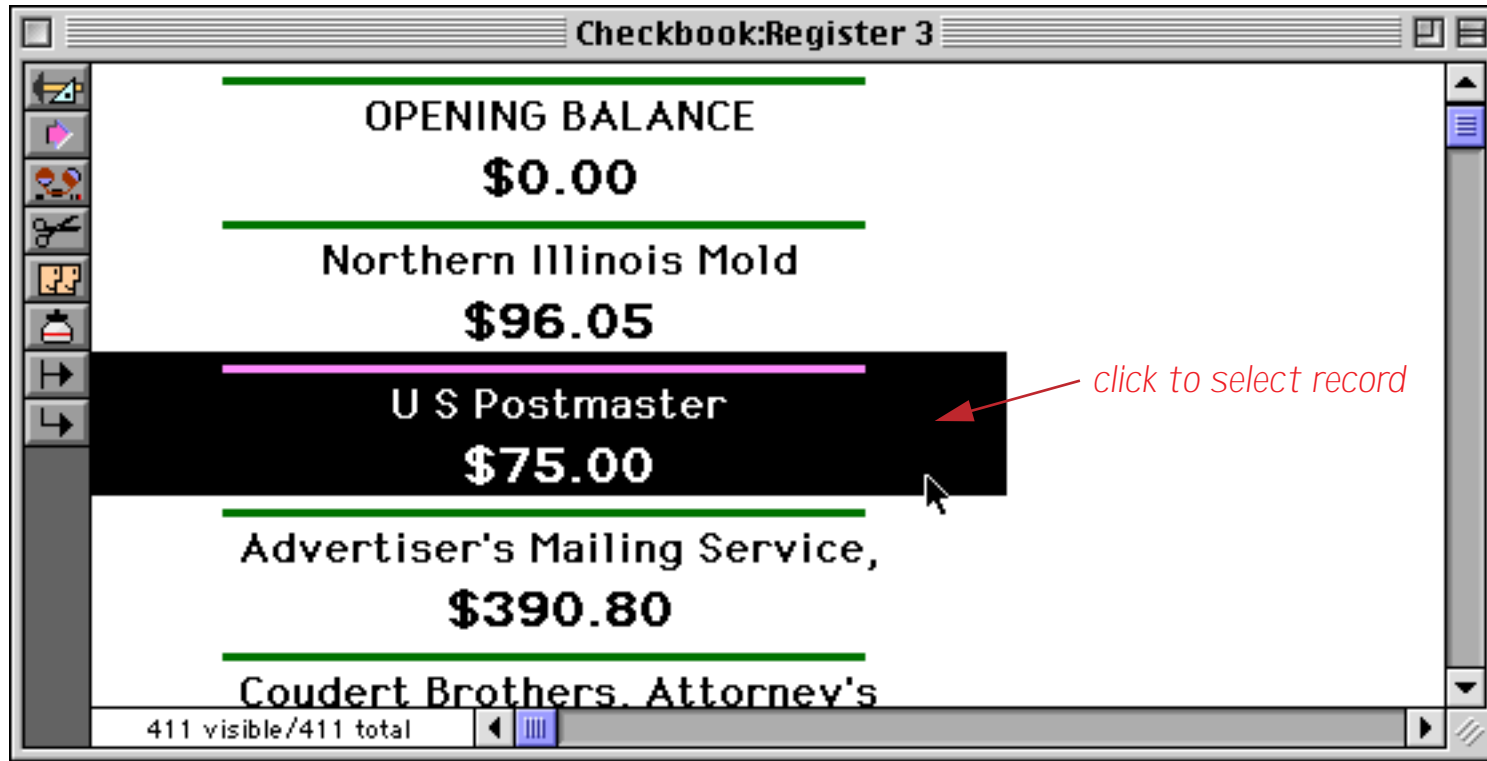
Got the idea? Now we can go back to Graphics Mode and add more objects. In this example we used auto-wrap text objects to display two fields from the database. You can also use Data Cells, Text Editor SuperObjects, Text Display SuperObjects or even Flash Art.



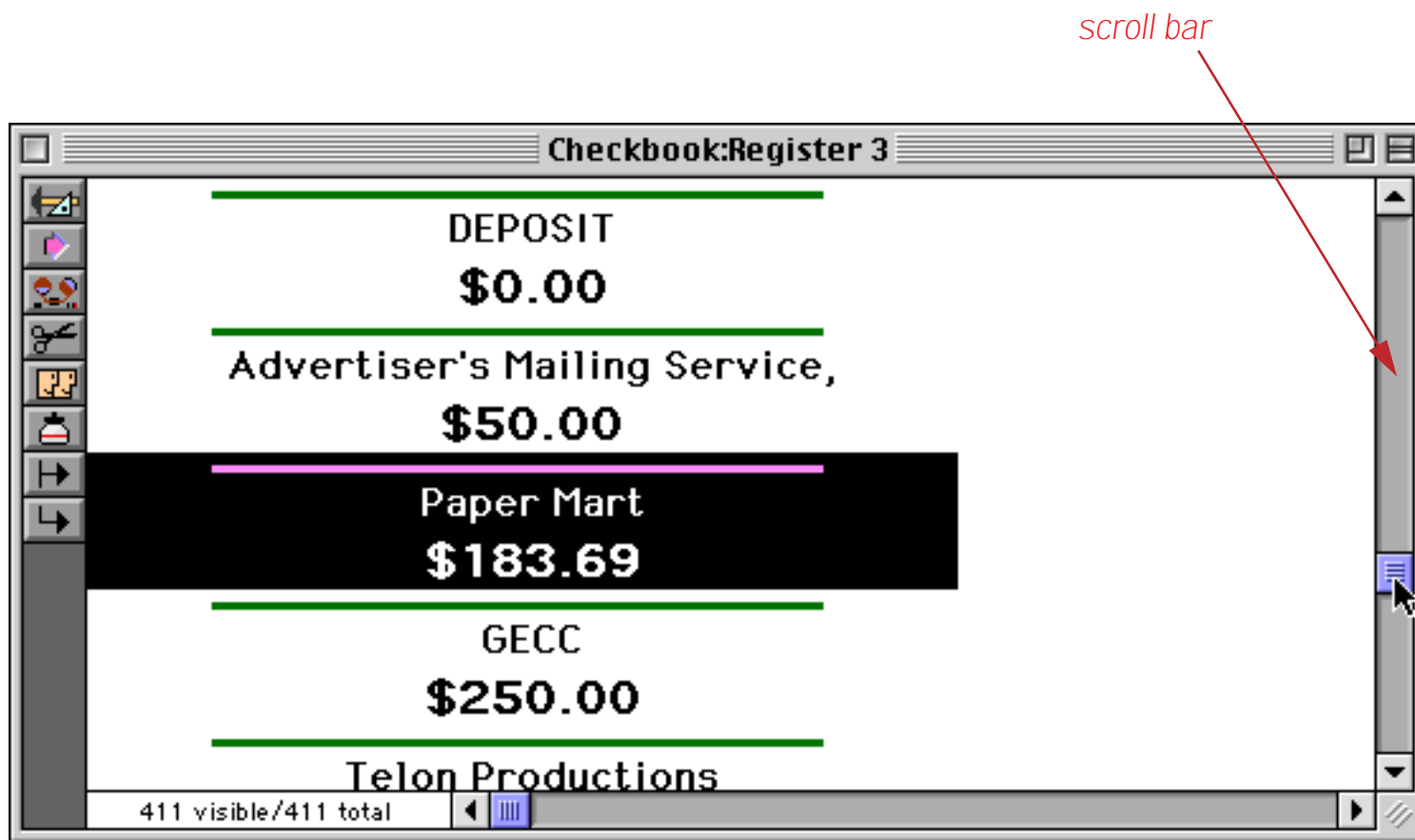
You can switch to Data Access Mode at any time to check out your work.



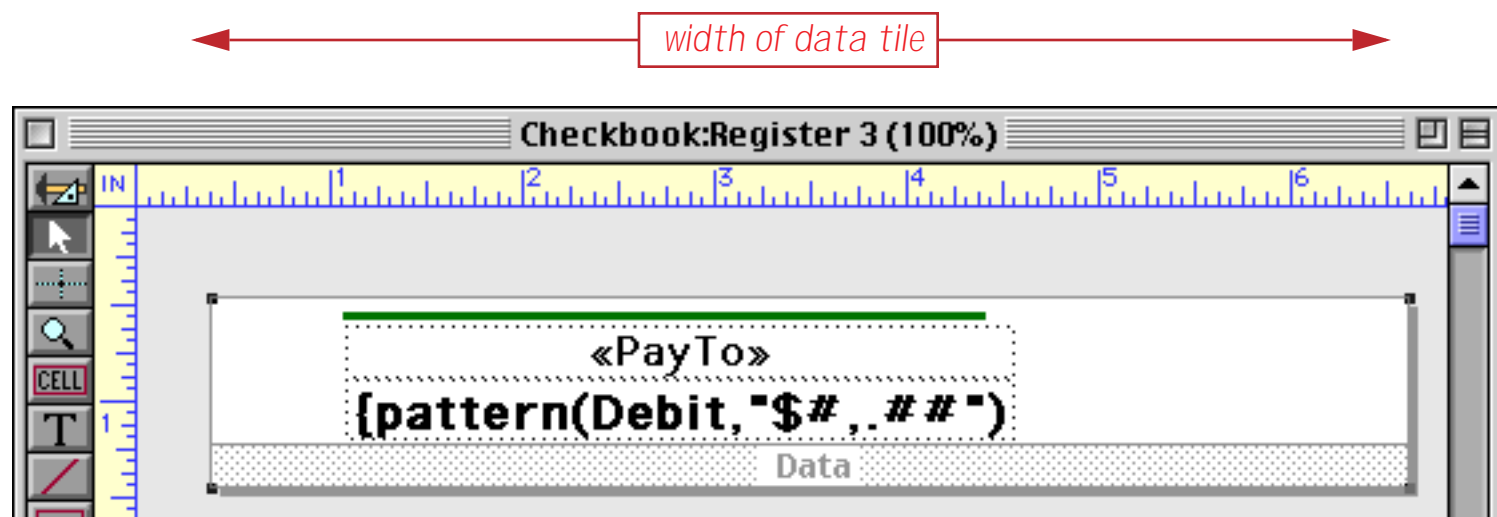
The view-as-list form kind of works like the data sheet. You can select a different record by clicking on it.



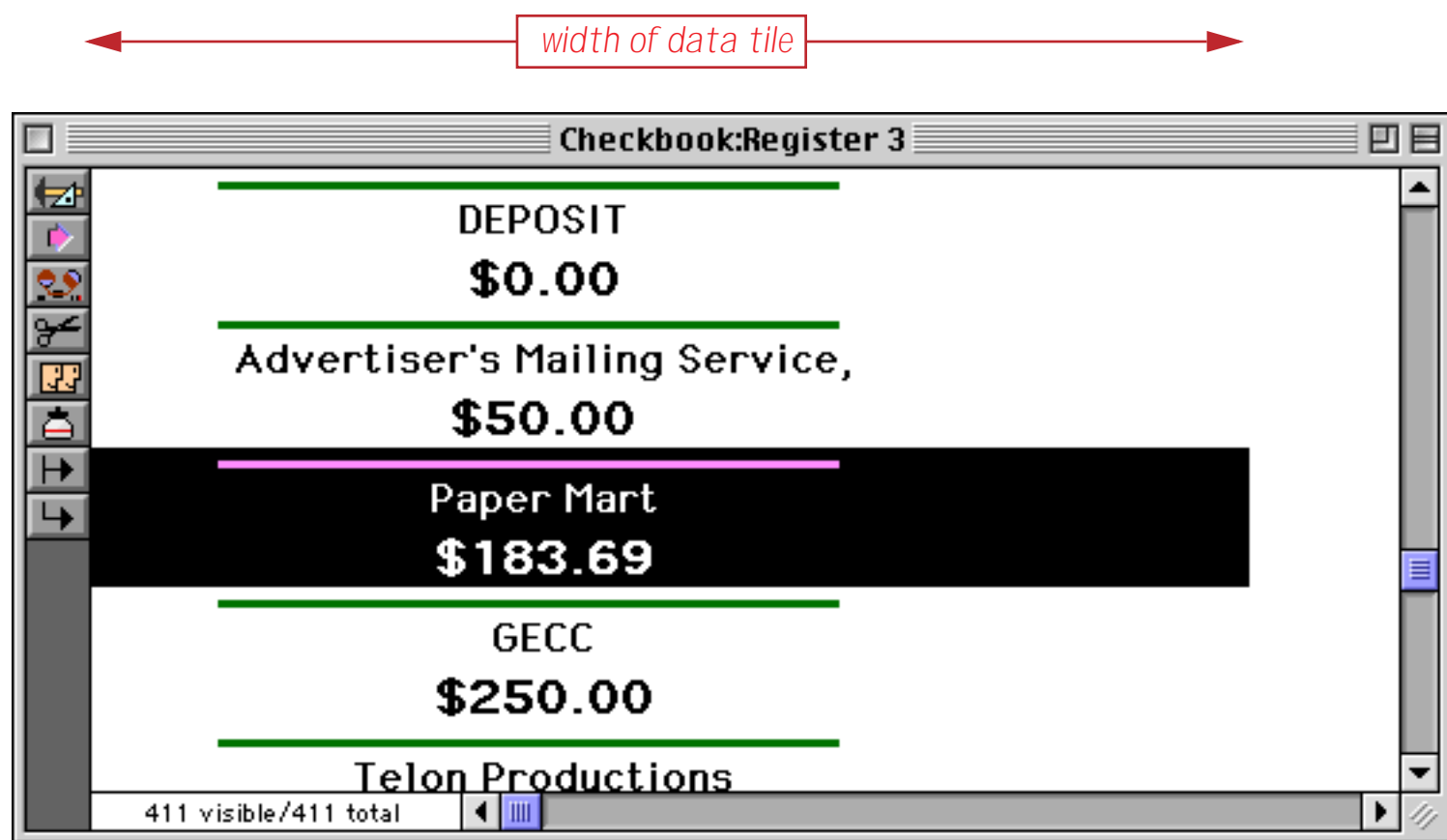
Or you can use the vertical scroll bar to navigate to any location within the database.



The width of the current record's highlighting corresponds to the width of the data cell. By changing the width of the data cell you can change the width of the highlight.



In Data Access Mode you can see the new, revised highlight width.



Since this form doesn't contain any Text Editor SuperObjects or Data Cells you cannot edit the data using this form. See "[Editable View-As-List Forms](#)" on page 410 to learn how to create an editable view-as-list form.

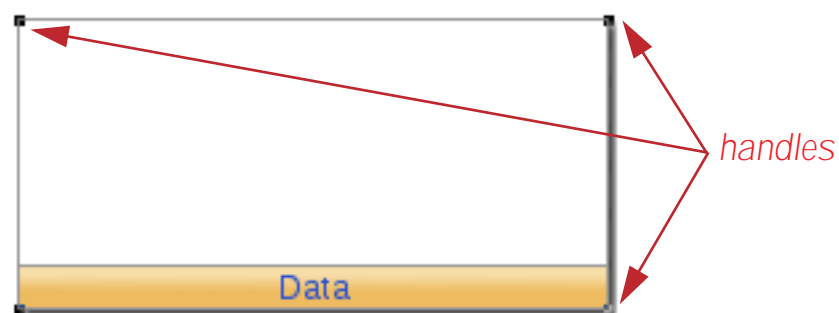
#### Working with Tiles

Like all other graphic objects, tiles can be moved and resized with the **Pointer** tool. However, tiles are slightly different than other objects. On the screen, a tile looks similar to an upside down window. Tiles are divided into two parts: the surface and the drag bar.





Unlike other graphic objects that can be manipulated by clicking anywhere in the object, a tile is only sensitive to clicks on its drag bar. To move a tile, press the mouse on the drag bar and drag the tile to the new position. To select a tile, click on the drag bar. When the tile is selected, four handles appear around the corners of the tile.



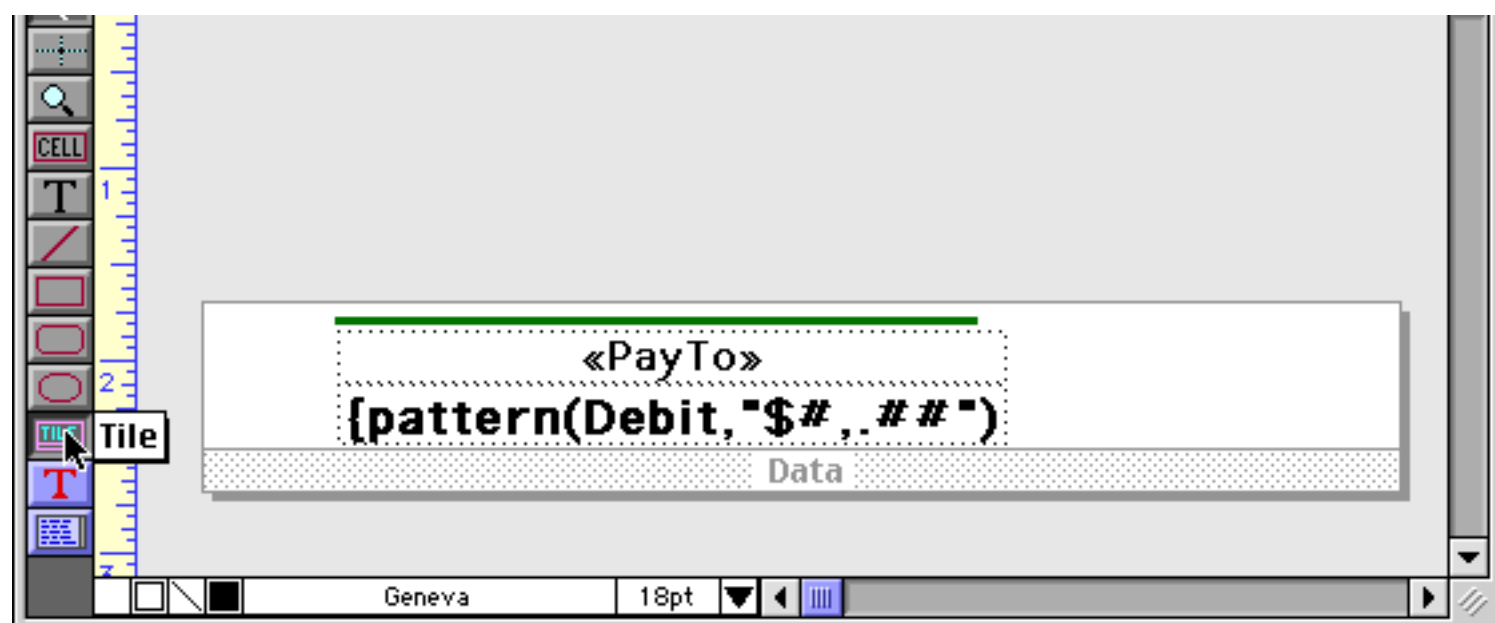
You can use these handles as grips to change the size of the tile. Tiles can also be moved or resized with the **Dimensions** command.

The surface of the tile is not sensitive to the mouse. In other words, clicking on the surface area does not select the tile, and you cannot move the tile by dragging on the surface.

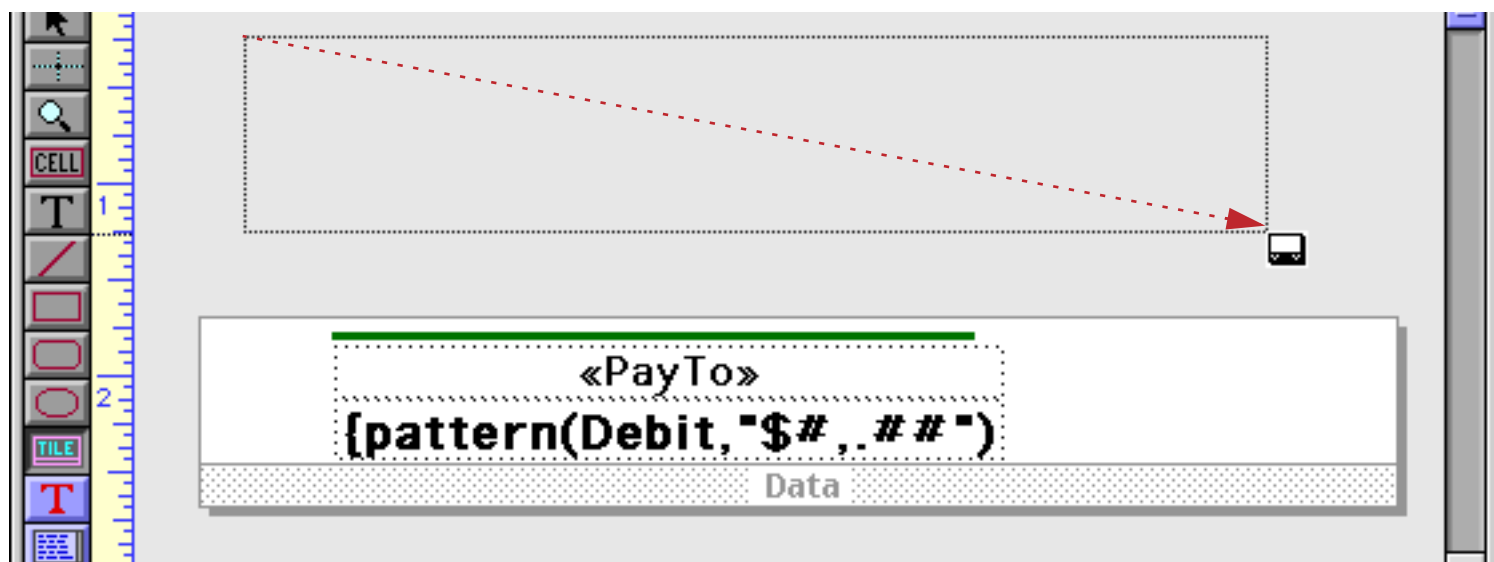
#### Adding a View-As-List Header

To add a header to your view-as-list form you'll need to add a second tile. Surprisingly enough this is called a **header tile**. There are two methods for creating this tile—you can create it from scratch or you can create it from a copy of the data tile.

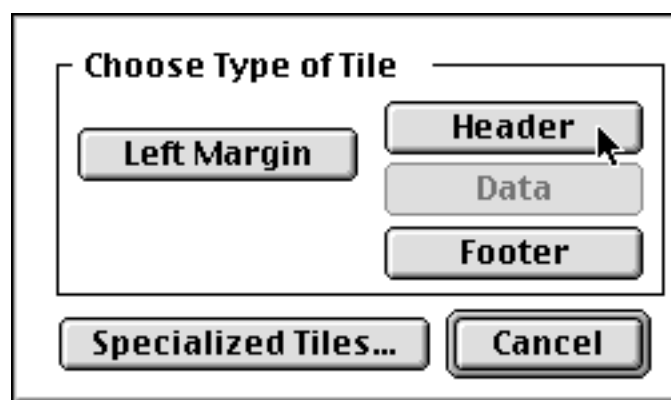
To create a header tile from scratch start by selecting the **Tile** tool.



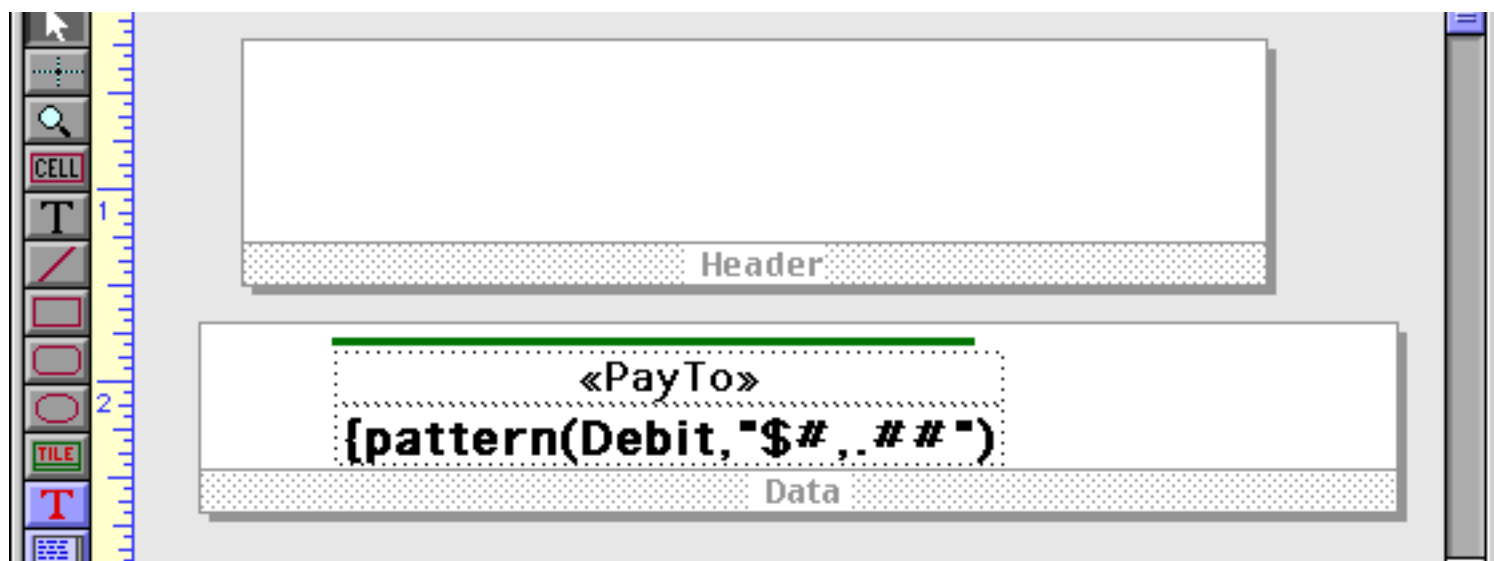
Now drag the mouse across the form where you want the tile to appear. The header tile does not have to line up with the data tile, although that can make it easier to visualize the final result.



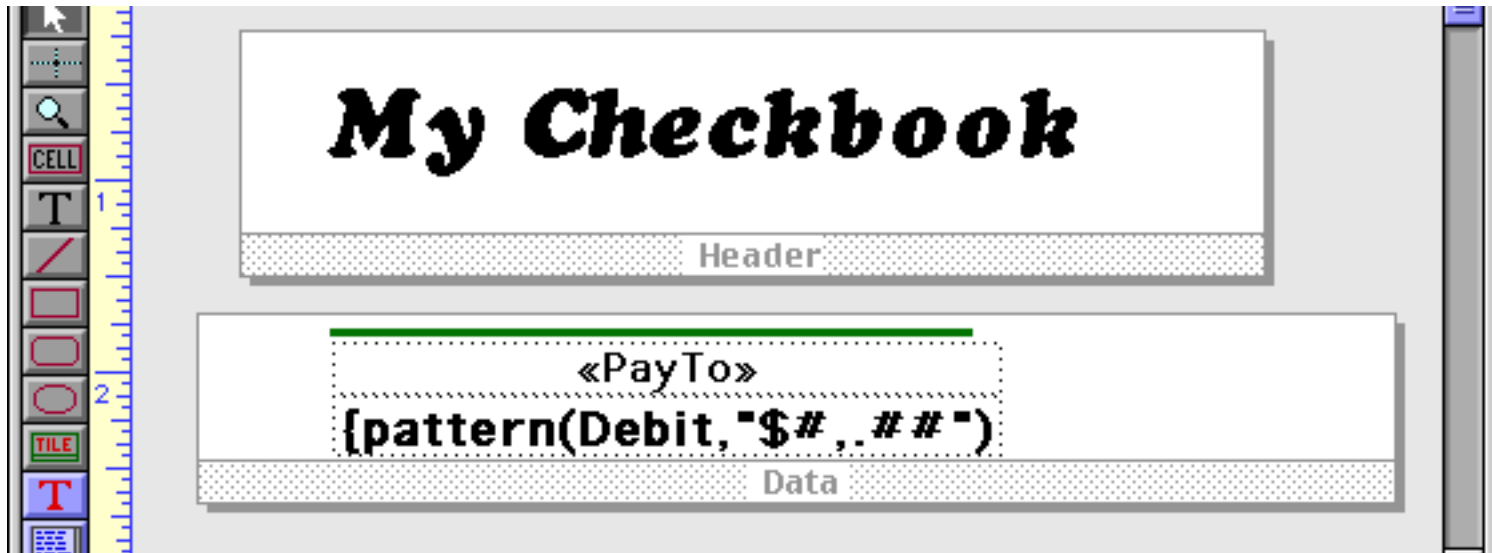
When you release the mouse a tile configuration dialog appears. This dialog allows you to select the type of tile you are creating.



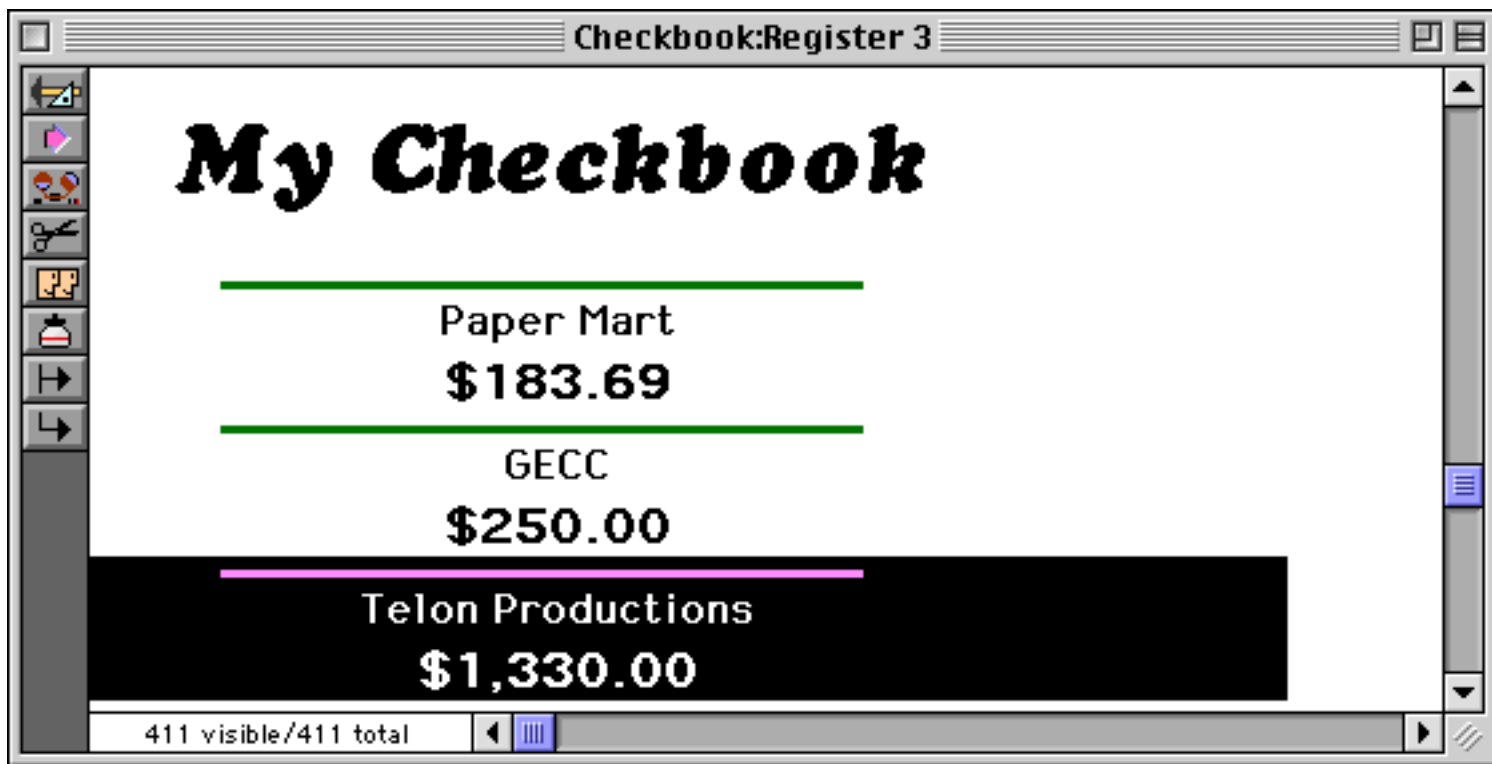
Press the **Header** button. When you press the button, Panorama creates the new tile.



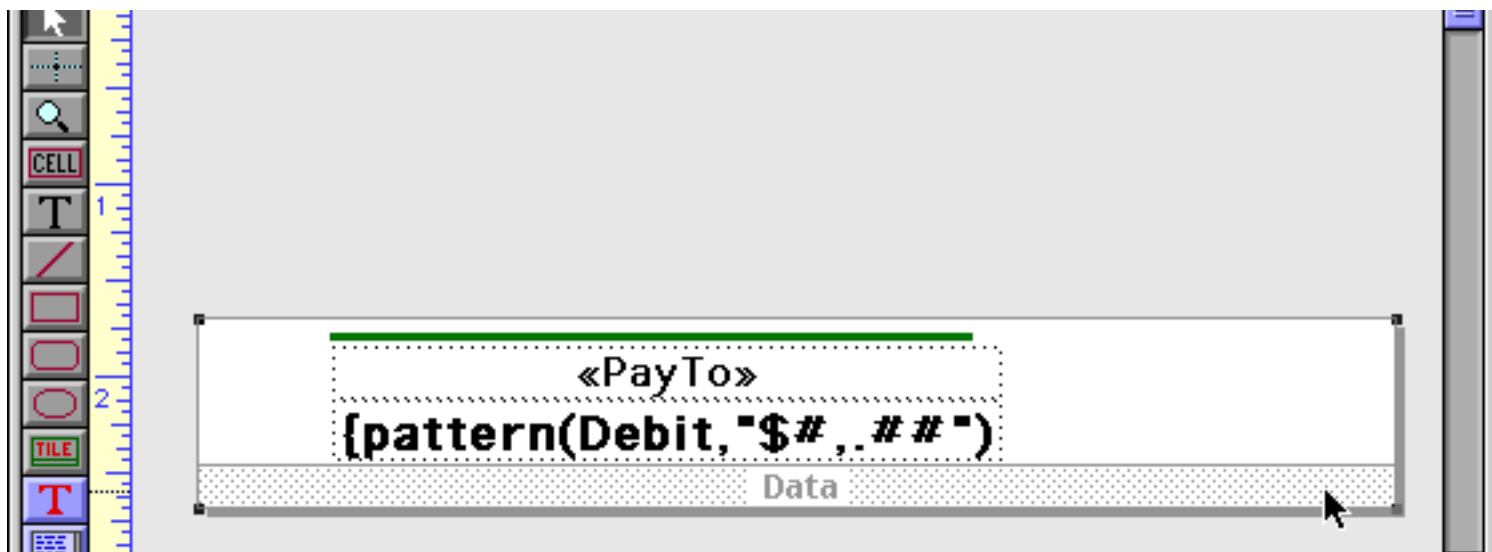
Now you should add any graphics and text that you want to appear on the header.



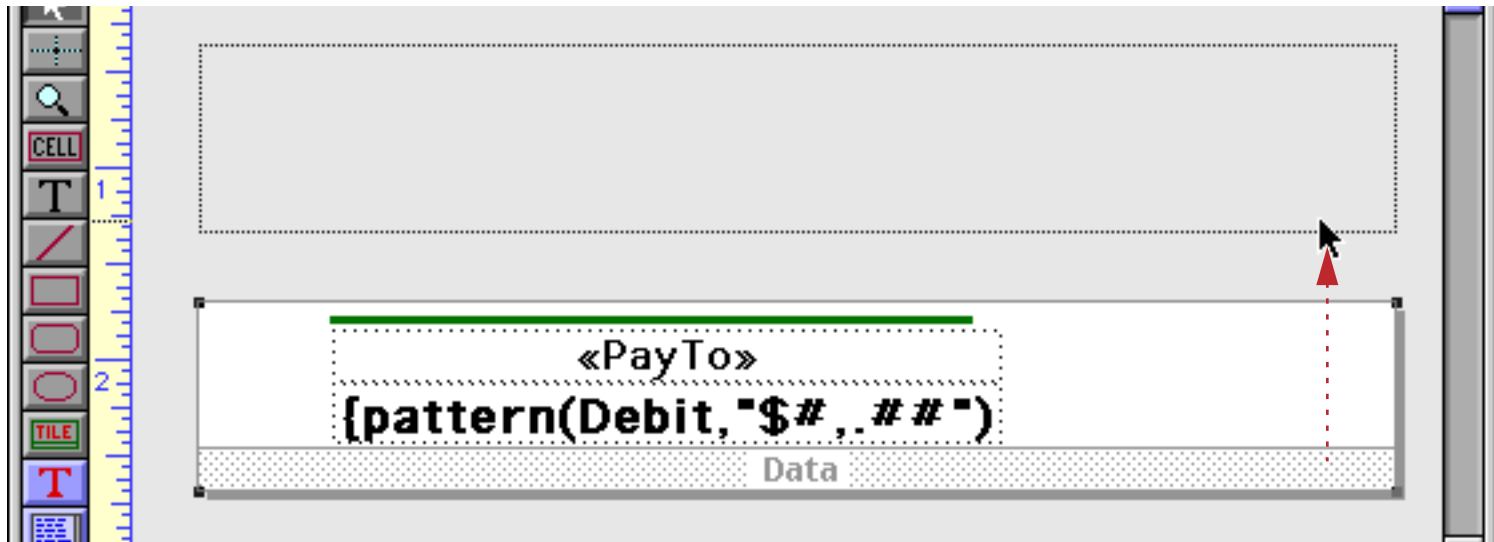
Switch to Data Access Mode to see the finished result.



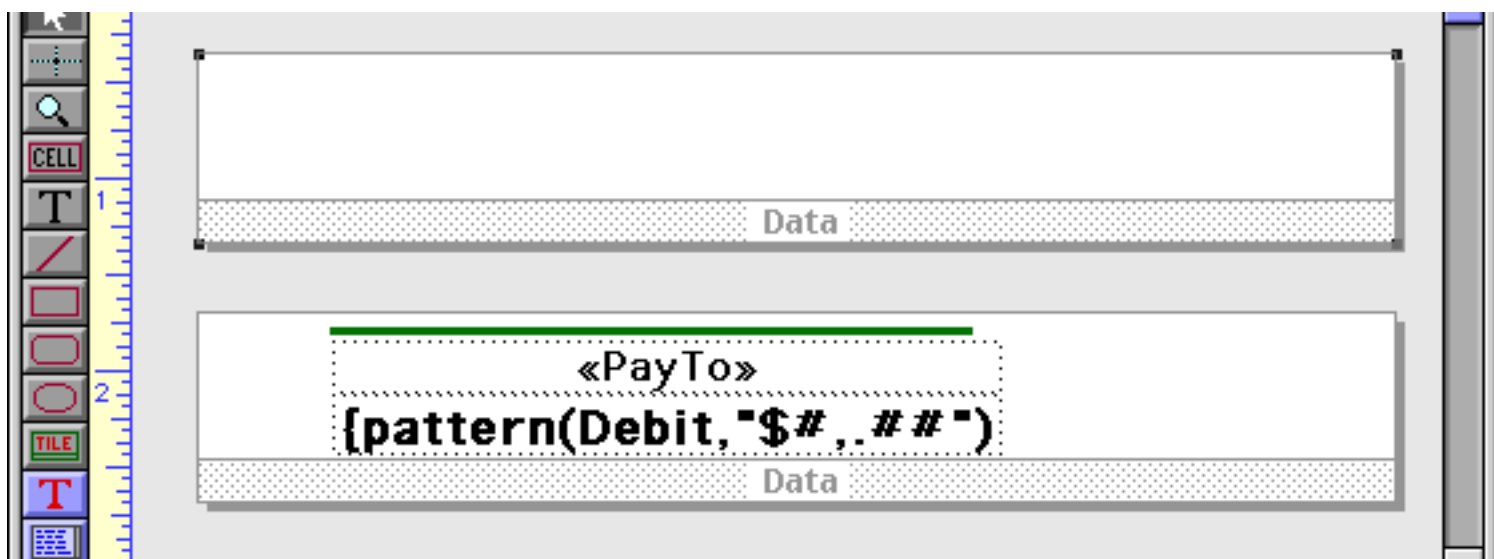
Now let's create a header tile by duplicating the data tile. Start with just the data tile.



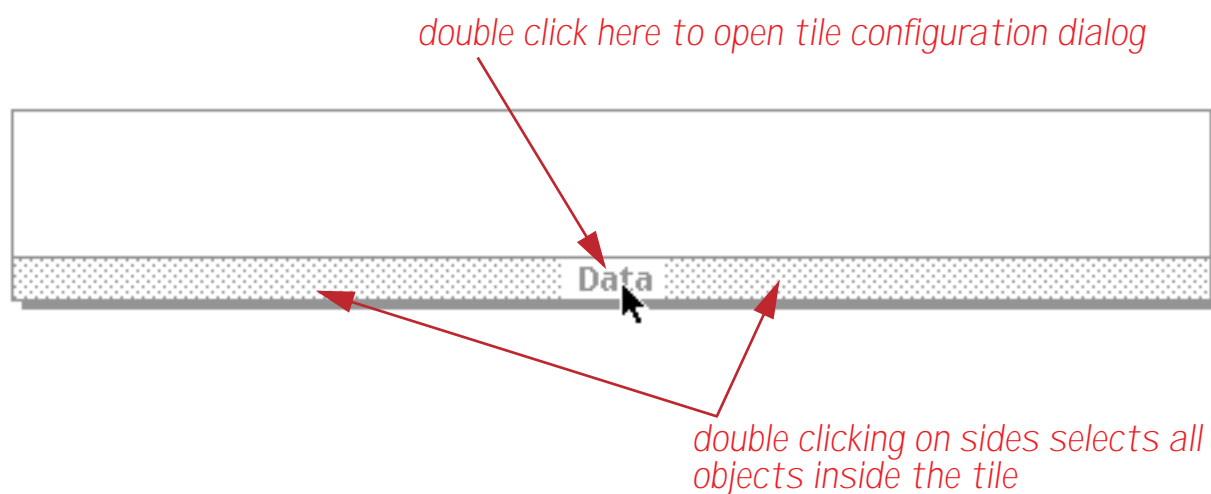
To duplicate the tile, hold down the **Option** key (Mac) or **Alt** key (Windows) and drag the tile (see “[Drag Duplicating](#)” on page 539). You may also want to hold down the **Shift** key at the same time to make sure that the two tiles stay in perfect alignment.



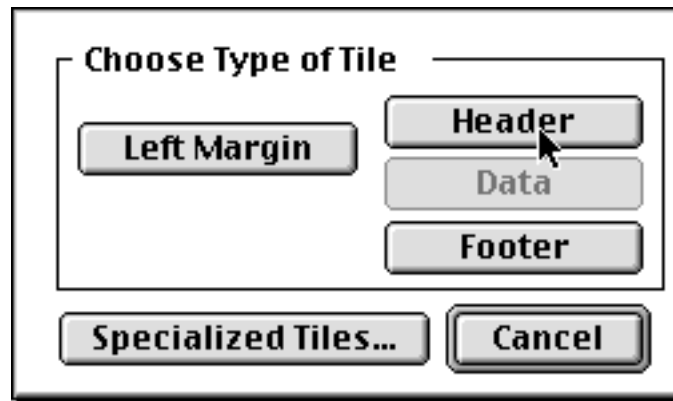
When you release the mouse your form will contain two data tiles.



To transform the new data tile into a header tile, double click on the word **Data**.



This opens the tile's configuration dialog.



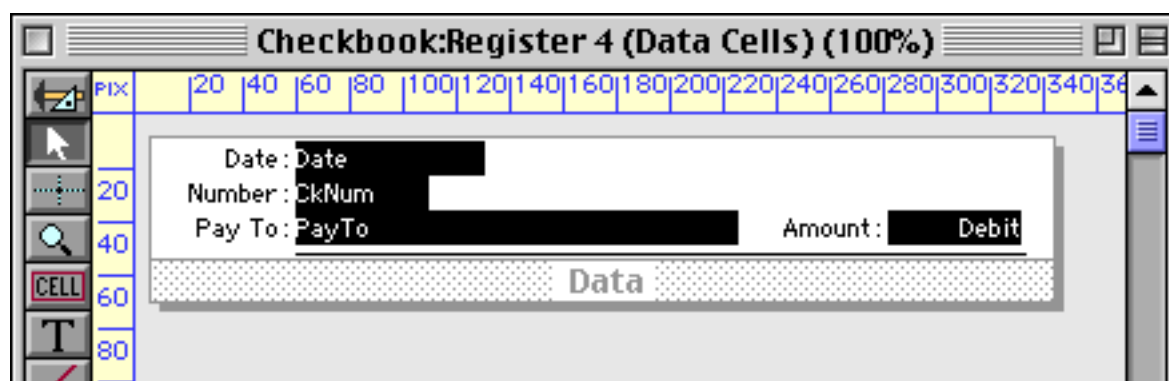
Press **Header** to convert the data tile into a header tile.



Voila! Now you can add any graphics and text you want to add to the header, and your form will be complete.

**Editable View-As-List Forms**

Adding data cells or Text Editor SuperObjects to the data tile makes it possible to edit the database using the view-as-list forms. The operation of each is a bit different. Here is a view-as-list form designed with Data Cells for editing.



Switch to Data Access Mode to see the list view.

The screenshot shows a window titled "Checkbook:Register 4 (Data Cells)". The window contains a list of transactions. Each transaction is displayed with its date, number, payee, and amount. The cells for "Date: 07/03/99", "Number: 2174", and "Amount: 183.69" are highlighted in black. The status bar at the bottom indicates "411 visible/411 total".

Date	Number	Pay To	Amount
07/03/99	2174	Paper Mart	183.69
07/03/99	2175	GECC	250.00
07/03/99	2176	Telon Productions	1,330.00
07/09/99	2177	Walthers	1,955.75
07/09/99	2178	MDC	741.50

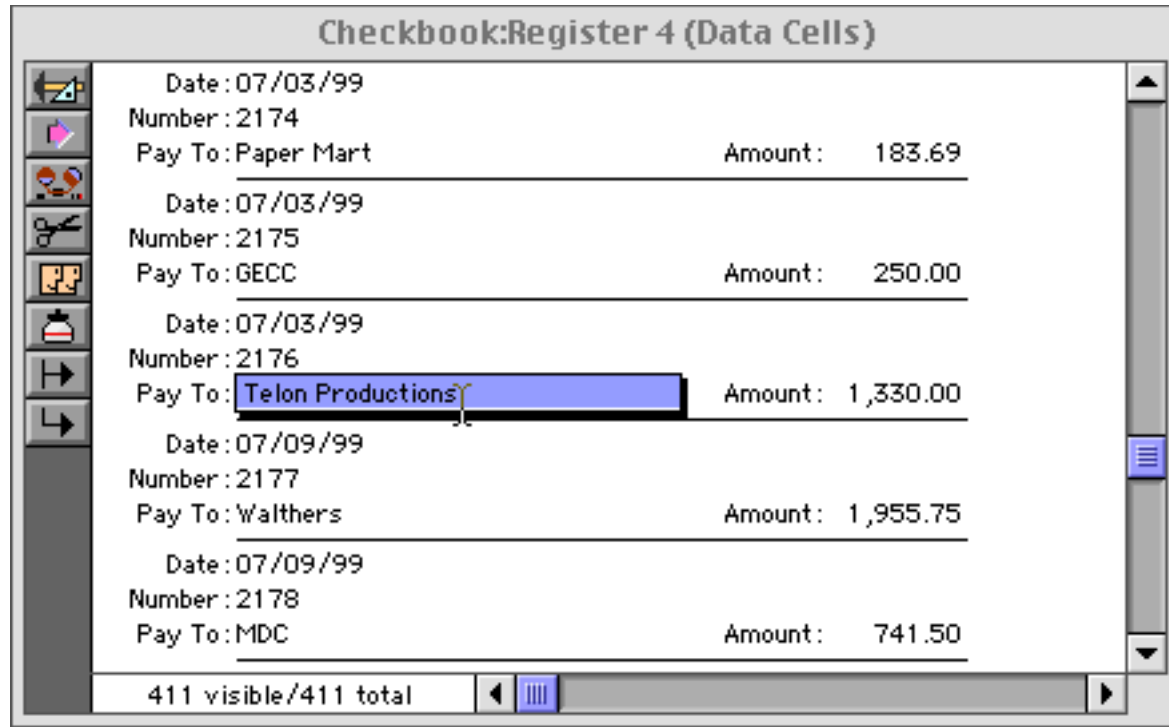
As you can see, when Data Cells are added to the form Panorama no longer highlights the entire record. Only the current cell itself is highlighted. You can click on any visible cell to highlight it.

The screenshot shows the same window as above. A mouse cursor is pointing at the cell "Pay To: Telon Productions", which is highlighted in black. The status bar at the bottom indicates "411 visible/411 total".

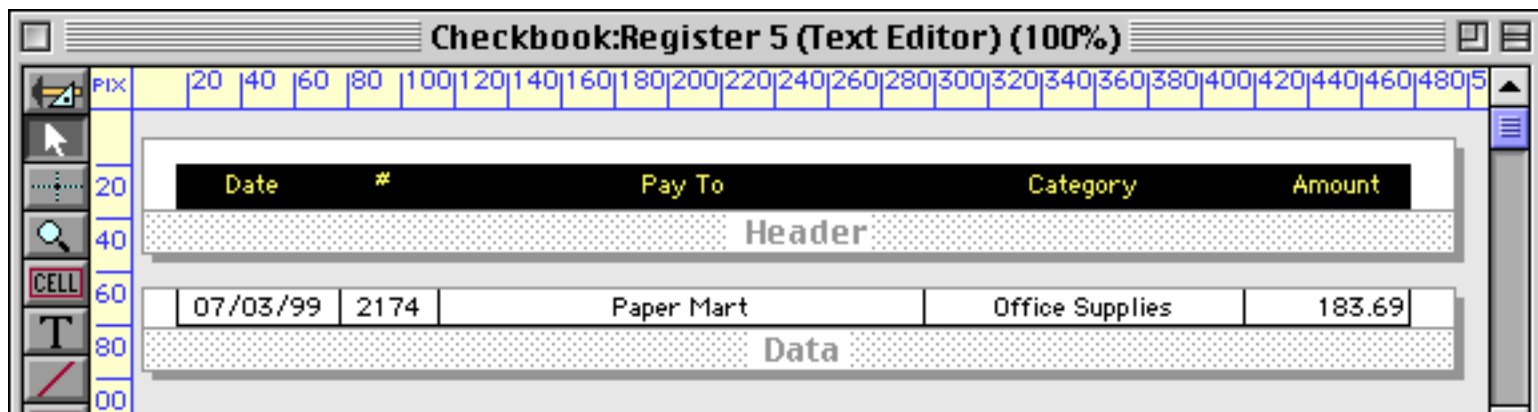
Date	Number	Pay To	Amount
07/03/99	2174	Paper Mart	183.69
07/03/99	2175	GECC	250.00
07/03/99	2176	Telon Productions	1,330.00
07/09/99	2177	Walthers	1,955.75
07/09/99	2178	MDC	741.50



To edit a cell, double click on it.

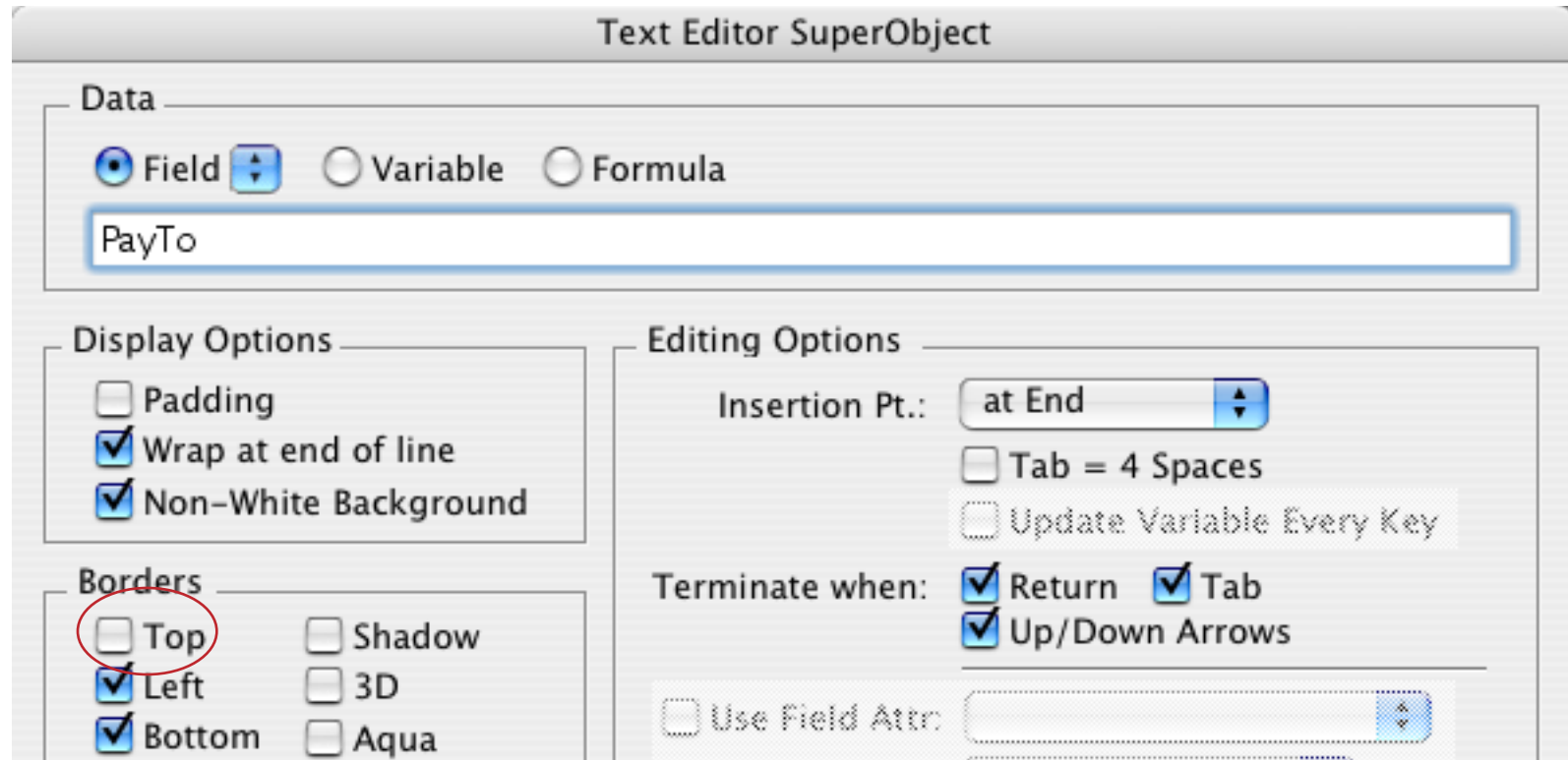


Here is a form created using Text Editor SuperObjects.



In this case, each object was created with borders on the left, bottom and right but not on the top. Here is an “exploded view” of these objects, along with the configuration dialog for one of the objects.

07/03/99 | 2174 | Paper Mart | Office Supplies | 183.69



Switch to Data Access Mode to see the finished result.

Date	#	Pay To	Category	Amount
07/03/99	2174	Paper Mart	Office Supplies	183.69
07/03/99	2175	GECC	Fixed Assets	250.00
07/03/99	2176	Telon Productions	Purchases	1,330.00
07/09/99	2177	Walthers	Purchases	1,955.75
07/09/99	2178	MDC	Purchases	741.50
07/09/99	2179	Blue Cross Of Calif	Insurance	177.25
07/09/99	2180	Advertiser's Mailing Service, Inc.	Advertising	56.20
07/09/99	2181	Blue Dolphin Press	Printing	92.28
07/09/99	2182	Cable & Wireless	Telephone	82.38
07/11/99	2183	Advertiser's Mailing Service, Inc.	Advertising	25.70
07/15/99		DEPOSIT		
07/16/99	2184	Advertiser's Mailing Service, Inc.	Advertising	42.50
07/16/99	2185	Railroad Model Craftsman	Advertising	453.42
07/16/99	2186	U S Postmaster	Postage	75.00
07/16/99	2187	Pacific Bell	Telephone	26.94
07/16/99	2188	G T E	Telephone	210.67
07/16/99	2189	Unocal	Auto	38.11

411 visible/411 total

As you can see, one potential disadvantage of this technique is that nothing is highlighted, so it is very difficult to see which record is current. (If you need to be able to tell which record is highlighted, use Data Cells.) To edit a particular item just click on it.

Date	#	Pay To	Category	Amount
07/03/99	2174	Paper Mart	Office Supplies	183.69
07/03/99	2175	GECC	Fixed Assets	250.00
07/03/99	2176	Telon Productions	Purchases	1,330.00
07/09/99	2177	Walthers	Purchases	1,955.75
07/09/99	2178	MDC	Purchases	741.50
07/09/99	2179	Blue Cross Of Calif	Insurance	177.25
07/09/99	2180	Advertiser's Mailing Service, Inc.	Advertising	56.20
07/09/99	2181	Blue Dolphin Press	Printing	92.28
07/09/99	2182	Cable & Wireless	Telephone	82.38
07/11/99	2183	Advertiser's Mailing Service, Inc.	Advertising	25.70
07/15/99		DEPOSIT		
07/16/99	2184	Advertiser's Mailing Service, Inc.	Advertising	42.50
07/16/99	2185	Railroad Model Craftsman	Advertising	453.42
07/16/99	2186	U S Postmaster	Postage	75.00
07/16/99	2187	Pacific Bell	Telephone	26.94
07/16/99	2188	G T E	Telephone	210.67
07/16/99	2189	Unocal	Auto	38.11

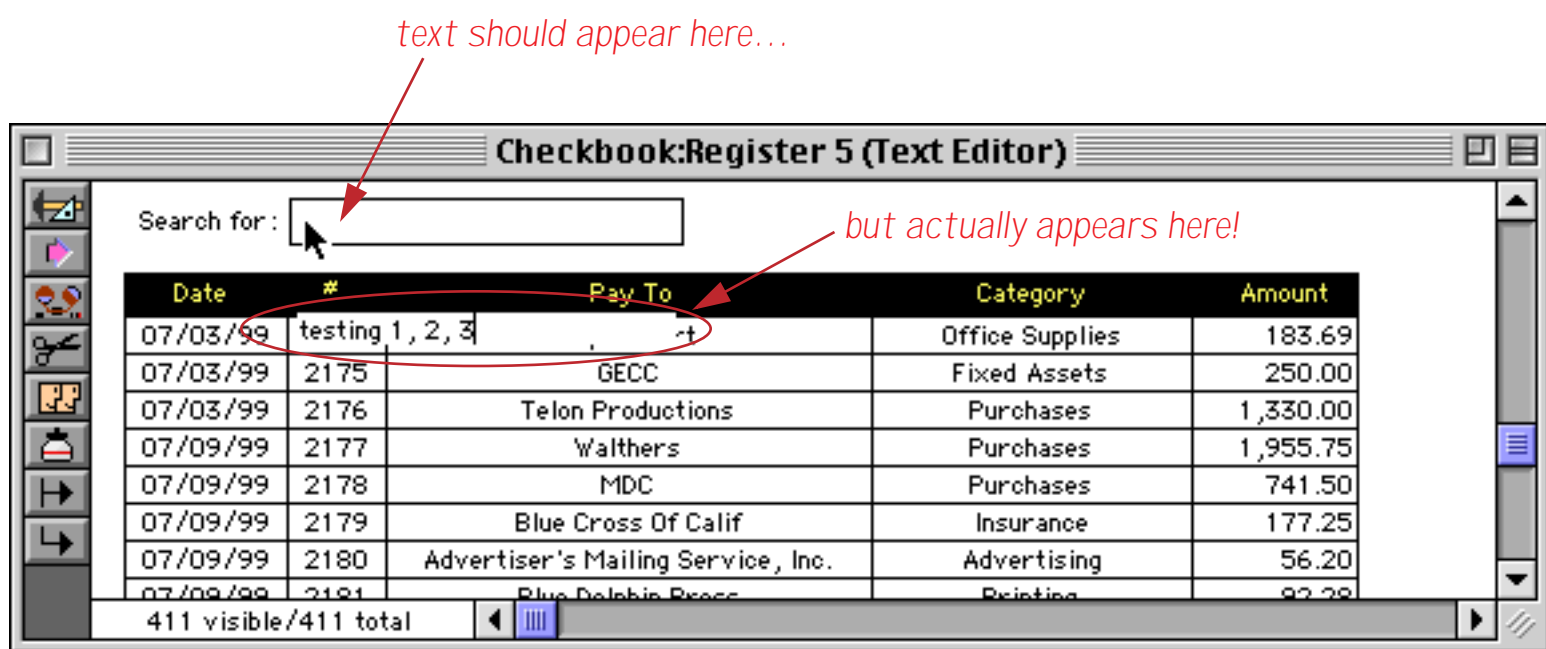
411 visible/411 total

You might be tempted to try placing a Text Editor SuperObject or Data Cell onto the header tile, like this.

Text Editor SuperObject

Date	#	Pay To	Category	Amount
Header				
07/09/99	2179	Blue Cross Of Calif	Insurance	177.25
Data				

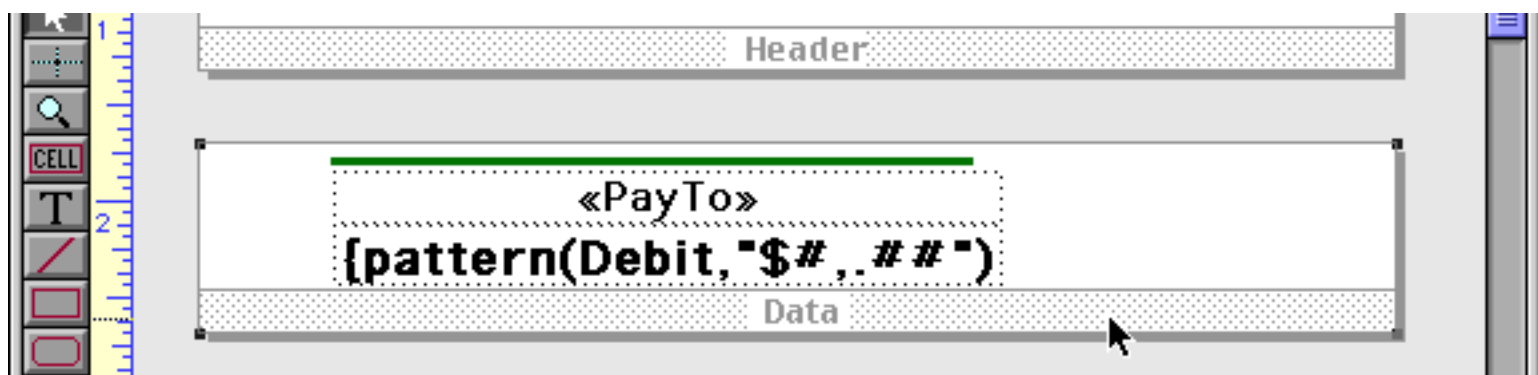
Unfortunately, this does not work correctly. When you attempt to edit the text, the edited text appears in the wrong position, something like this.



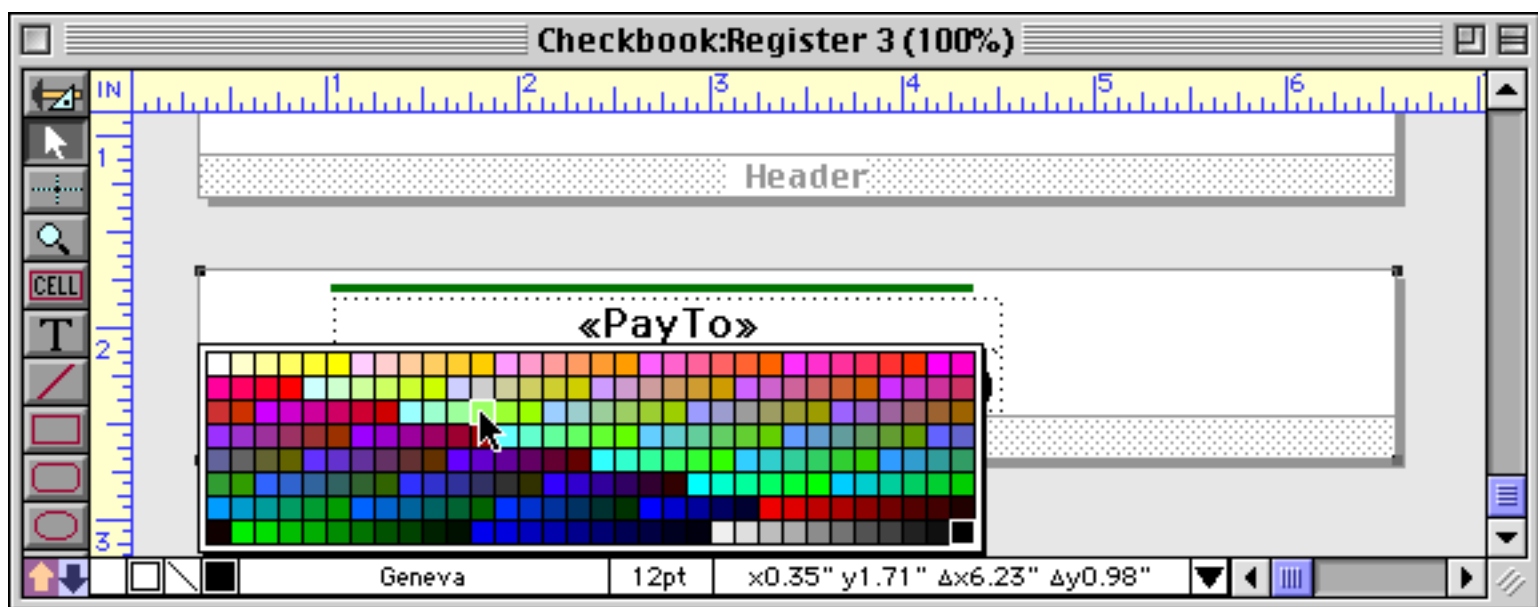
This problem may be corrected in a future version of Panorama, but for now you can only edit text within the data tile.

#### View-As-List Background Colors

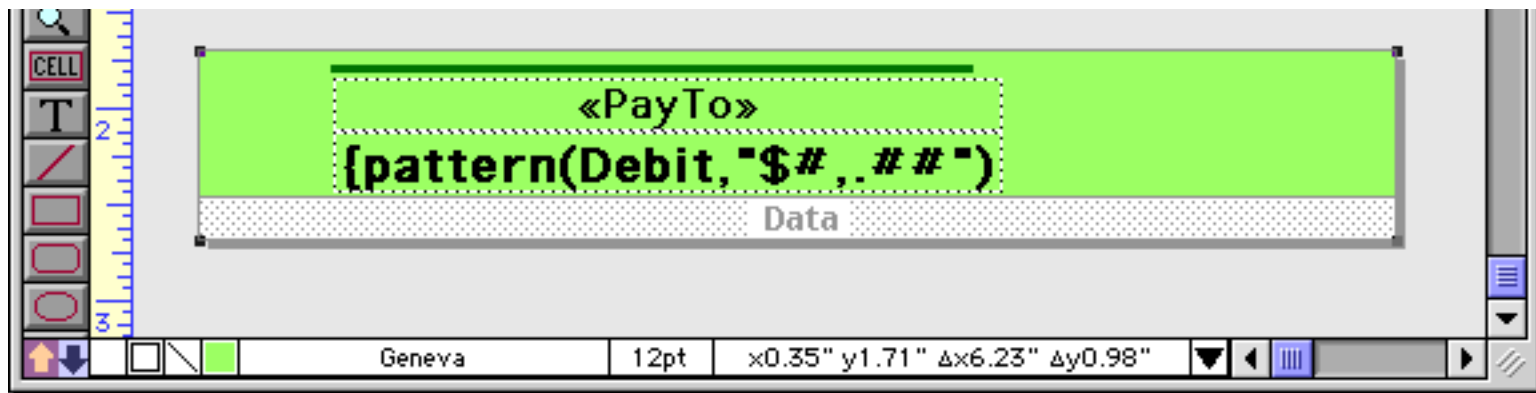
A tile's surface is normally white. However, it is possible to change the color of the tile surface to any color in Panorama's color palette. To change the color, start by clicking on the tile's drag bar to select it (see "[Working with Tiles](#)" on page 405).



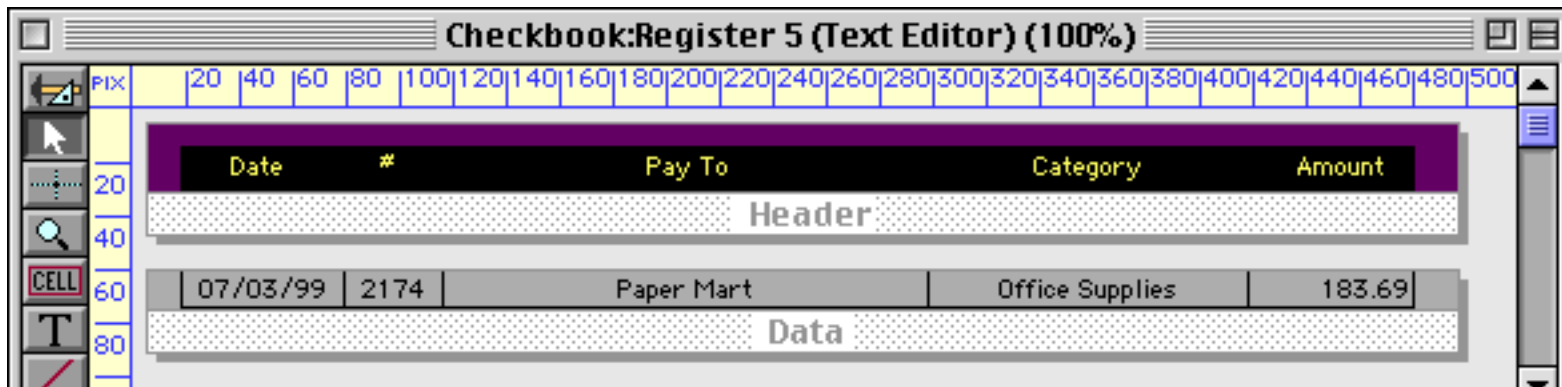
Once the tile is selected you can select a new color from the Graphic Control Strip (see "[Color](#)" on page 504). You can choose any color you want, but usually the lighter grays and pastels work best.



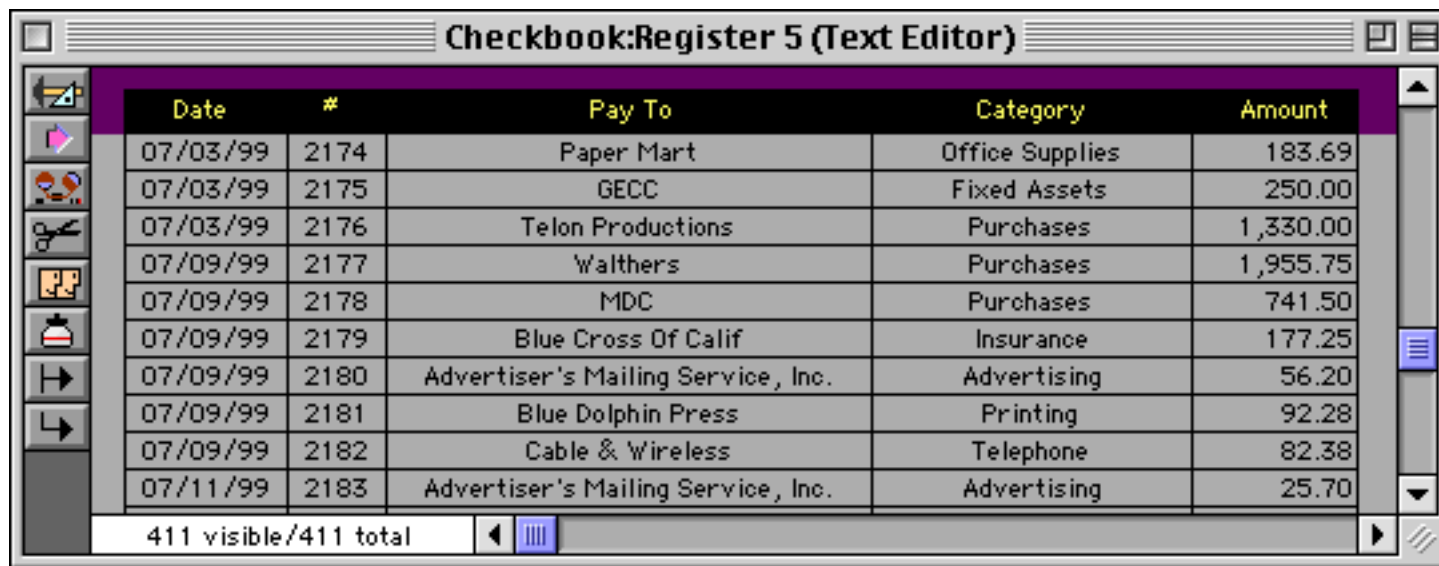
When you release the mouse the tile's surface changes to the new color. By the way, if you want to change the tile's surface back to white, select black (the color in the far bottom right).



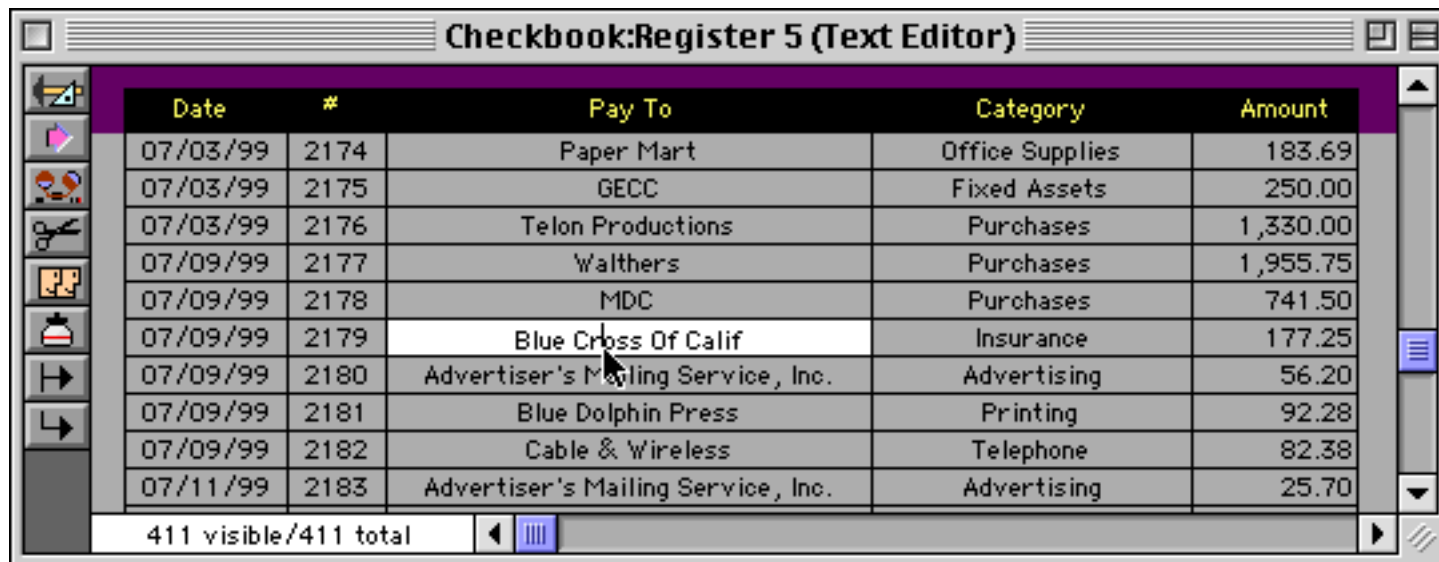
Colored tiles work well with Text Editor SuperObjects. Here's an example.



Here's the same form in Data Access Mode.



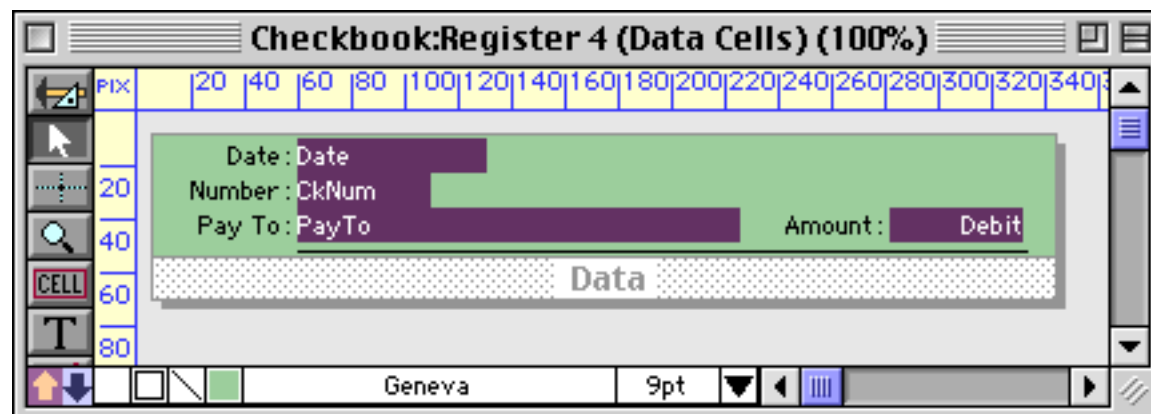
When you click on an item to edit it, the entire area turns white (be sure you have selected the Text Editor **Non-White Background** option, see “[Text Editor Options](#)” on page 621).



Date	#	Pay To	Category	Amount
07/03/99	2174	Paper Mart	Office Supplies	183.69
07/03/99	2175	GECC	Fixed Assets	250.00
07/03/99	2176	Telon Productions	Purchases	1,330.00
07/09/99	2177	Walthers	Purchases	1,955.75
07/09/99	2178	MDC	Purchases	741.50
07/09/99	2179	Blue Cross Of Calif	Insurance	177.25
07/09/99	2180	Advertiser's Mailing Service, Inc.	Advertising	56.20
07/09/99	2181	Blue Dolphin Press	Printing	92.28
07/09/99	2182	Cable & Wireless	Telephone	82.38
07/11/99	2183	Advertiser's Mailing Service, Inc.	Advertising	25.70

411 visible/411 total

Colored tiles can also work with Data Cells.



Checkbook: Register 4 (Data Cells) (100%)

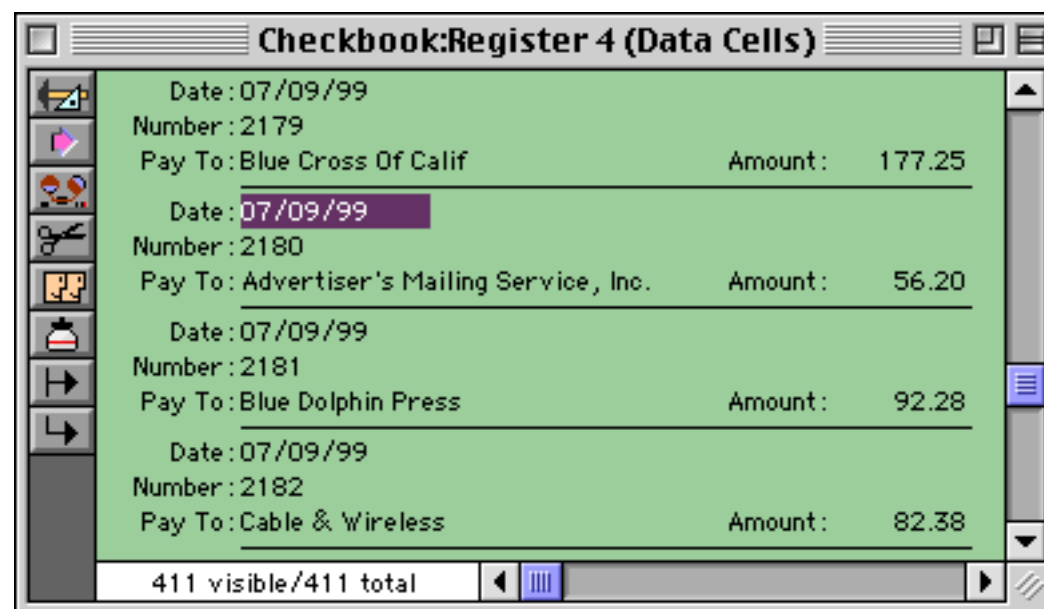
PIX | 20 | 40 | 60 | 80 | 100 | 120 | 140 | 160 | 180 | 200 | 220 | 240 | 260 | 280 | 300 | 320 | 340

Date: Date  
 Number: CkNum  
 Pay To: PayTo  
 Amount: Debit

Data

Geneva 9pt

Here's what this form looks like in Data Access Mode.



Checkbook: Register 4 (Data Cells)

Date: 07/09/99  
 Number: 2179  
 Pay To: Blue Cross Of Calif  
 Amount: 177.25

Date: 07/09/99  
 Number: 2180  
 Pay To: Advertiser's Mailing Service, Inc.  
 Amount: 56.20

Date: 07/09/99  
 Number: 2181  
 Pay To: Blue Dolphin Press  
 Amount: 92.28

Date: 07/09/99  
 Number: 2182  
 Pay To: Cable & Wireless  
 Amount: 82.38

411 visible/411 total

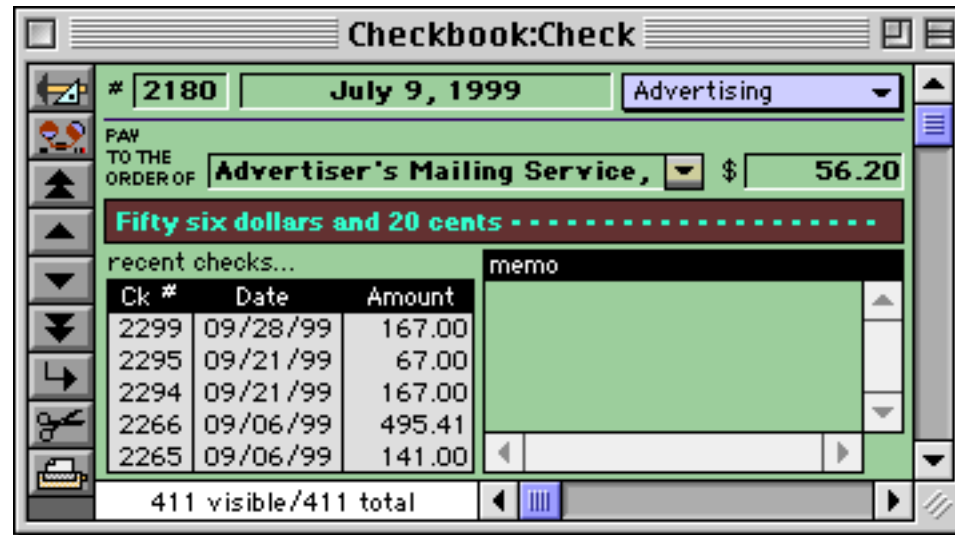
### Buttons on a View-As-List Form

You can place any kind of button on either the data tile or header tile of a view-as-list form — push buttons, data buttons, pop-up menus etc. Data buttons that are placed on the data tile should be linked to fields. Data buttons that are placed on the header tile should only be linked to variables, not fields.

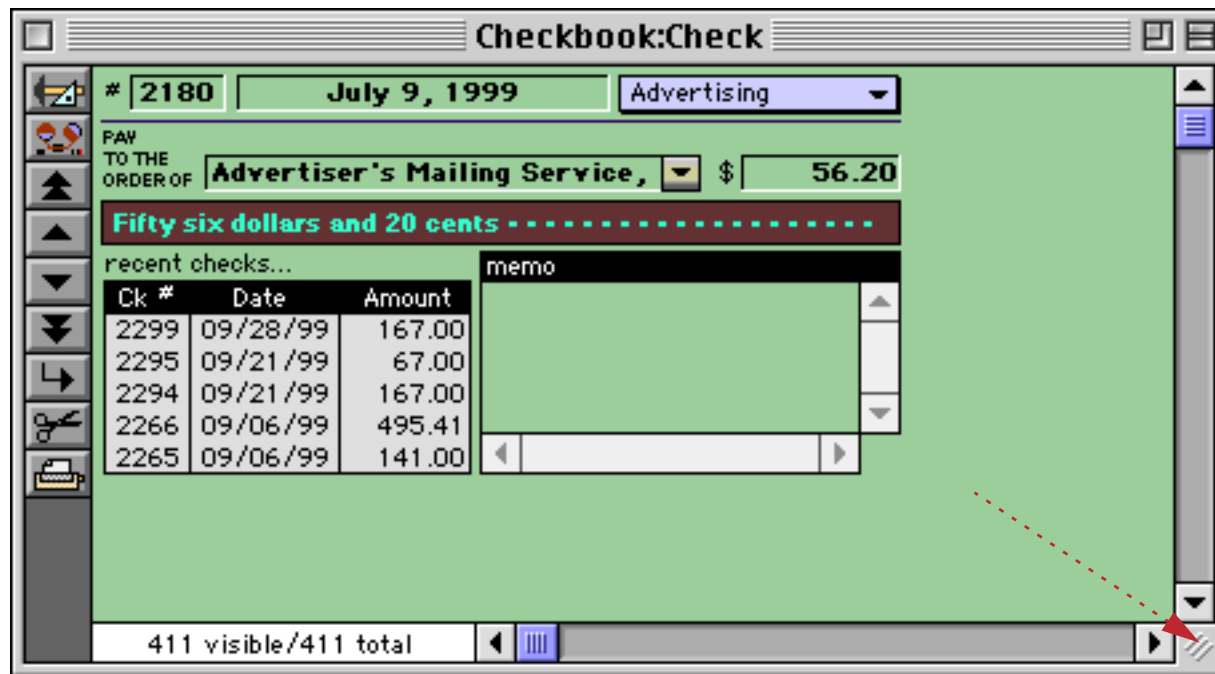


### Elastic Forms

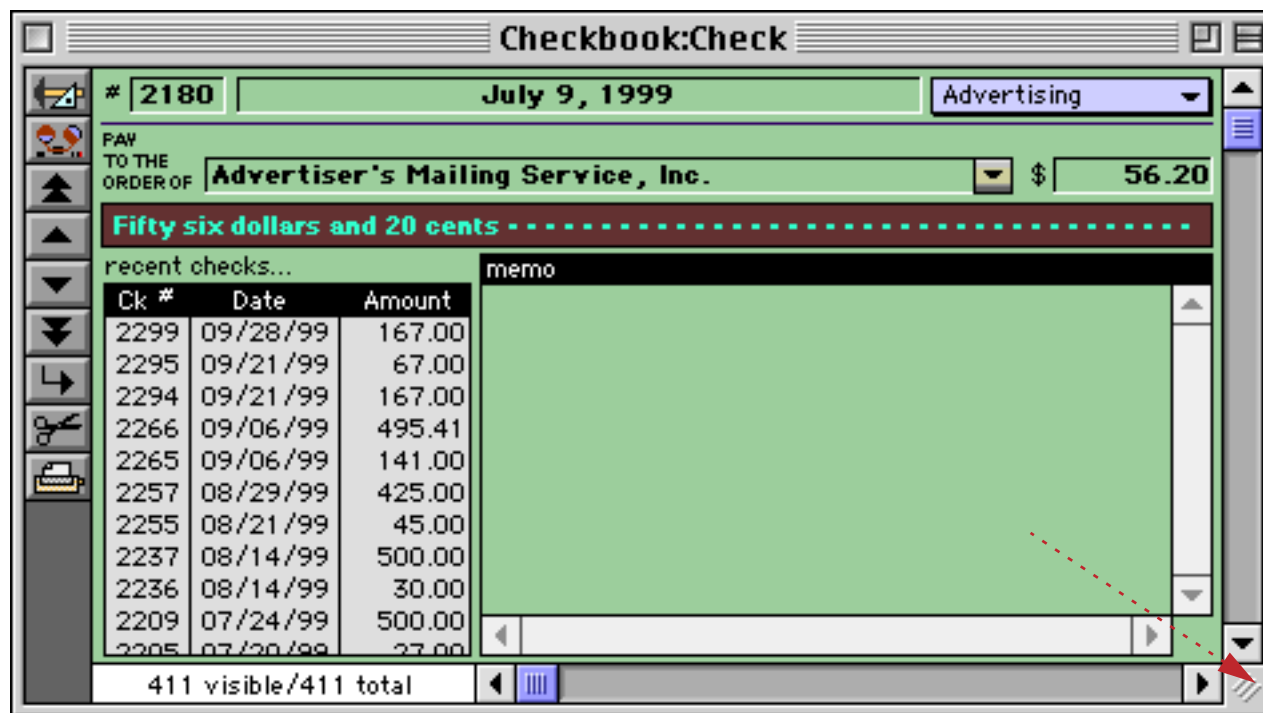
Elastic Forms is a feature that allows a form to adjust intelligently when the window containing the form is resized or zoomed. When the form is designed, you decide how the individual elements will expand or shift as the form changes size. Here's an example of a typical form.



If the window containing an ordinary form is resized, the form remains the same, in this case leaving a large blank area.



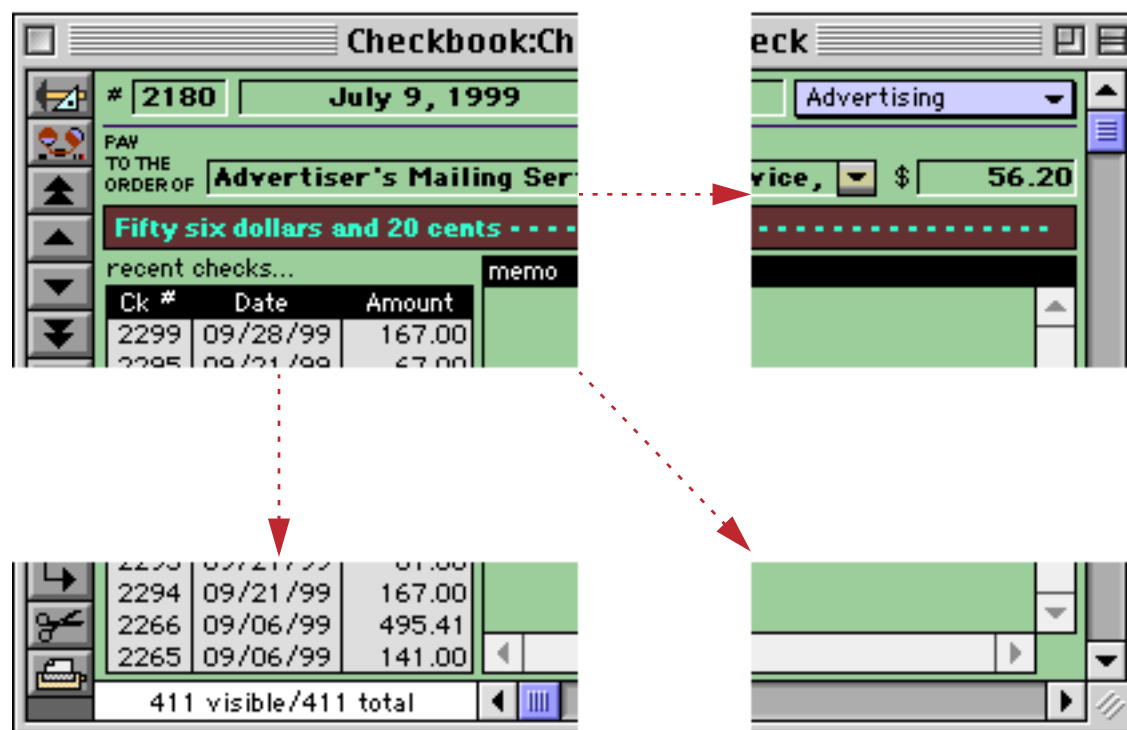
However, if the form is elastic the objects within the form will adjust themselves to the new size.



The best part is that once you learn how, you can turn almost any form into an elastic form in just a couple of minutes!

### Theory of Elastic Forms

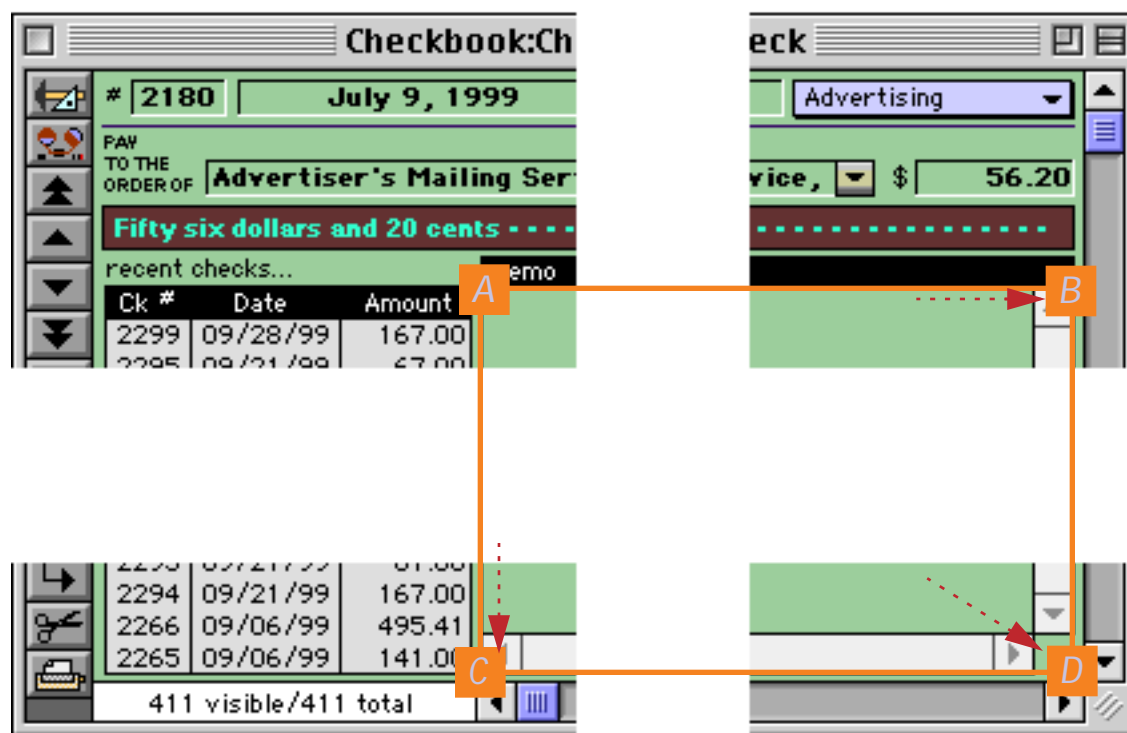
The basic theory of elastic forms is simple. The form is divided into four quadrants, like this.



The upper left hand quadrant always remains fixed, no matter how large or small the window gets. The upper right hand quadrant slides back and forth horizontally as the window size changes, while the lower left hand quadrant slides vertically. The lower right hand quadrant slides diagonally, always sticking to the bottom right hand corner of the window.

As Panorama adjusts the form, it doesn't adjust entire objects. Instead it adjusts individual points (i.e. object corners). If an object is split across multiple quadrants, that object will adjust in size as the form expands and shrinks. If an object is not split across multiple quadrants, it will remain the same size but may slide to a new position depending on which quadrant it is in.

As an example consider the Text Editor SuperObject shown below. The four corners of this object are split across the four quadrants. Point A, in the upper left quadrant, stays put. Point B, in the upper right quadrant, slides to the right. Point C slides down, while Point D slides diagonally. The end result is that the object expands to fill up the newly available space.



On the other hand the check number, amount and the category pop-up menu are all contained within a single quadrant, so these objects do not change in size. The Date and Pay To fields are split across two quadrants, so they expand in width but not in height.

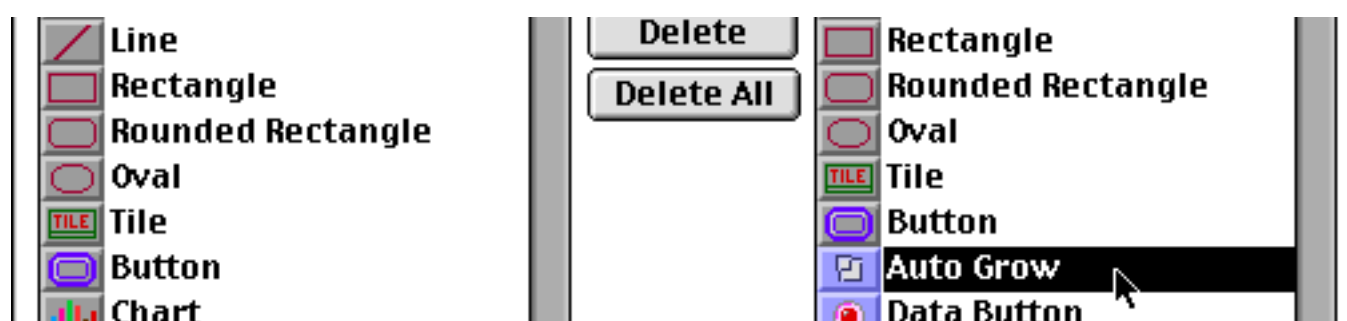
As you can see, the point where these four quadrants come together is very important (see “[Defining the Quadrants](#)” on page 420). You’ll need to pick this point carefully to create a form that expands and shrinks the way you want it to. Usually you’ll have one primary object that will expand and shrink as the window expands and shrinks. The quadrant meeting point should be inside this object. You may also have secondary objects above or below this object that need to expand and shrink in width only. The quadrant meeting point should line up with the middle of these objects. (If it is impossible to line up the quadrant meeting point with all of the secondary objects, you can create one or more slave Auto Grow objects. Slave Auto Grow objects allow you to create extensions that stick out of one or more quadrants (see Chapter 18 of the [Panorama Handbook](#).)

### Building an Elastic Form

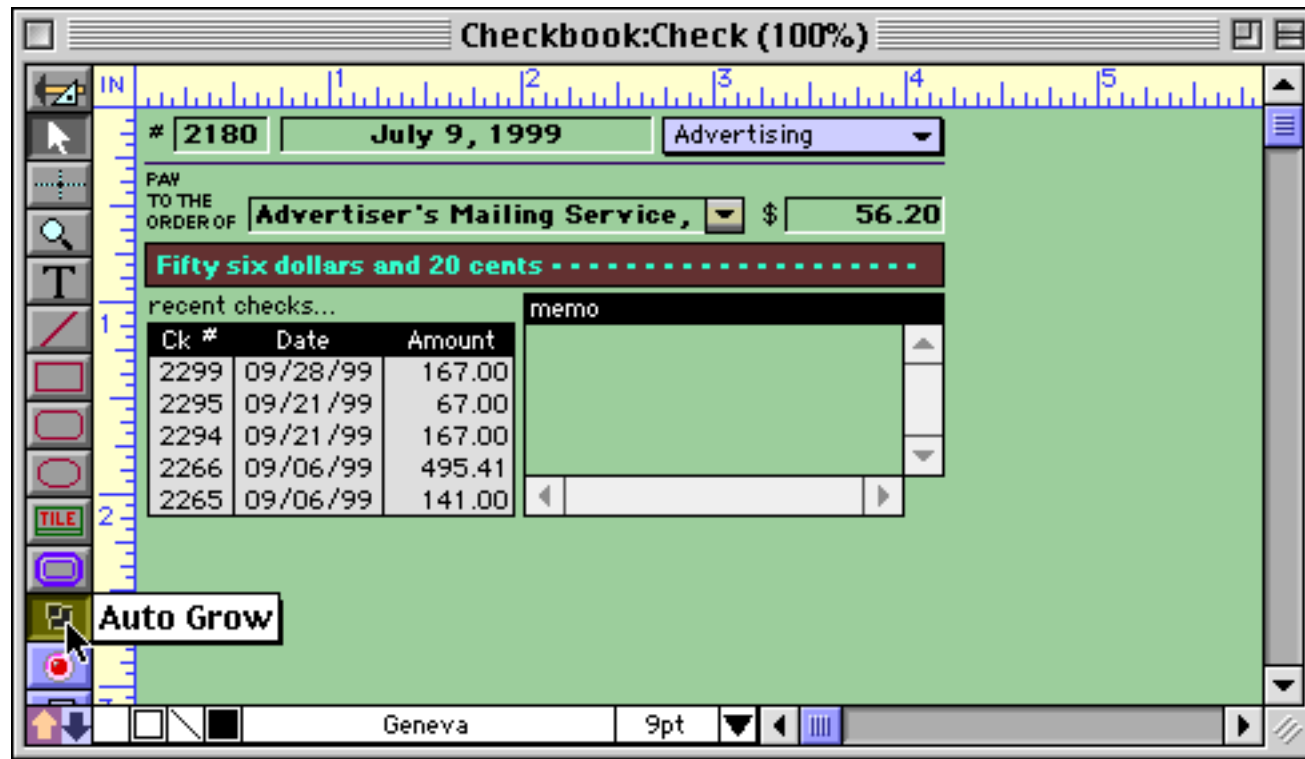
The first step in building an elastic form is to create a regular non-elastic form. Just use Panorama’s regular shape and user interface elements. It’s usually best to create the form in its smallest possible configuration. In other words, any form elements that may expand and shrink should be created in their “maximum shrink” size and shape. Once the form is created, be sure to save the database before going further.

### Defining the Quadrants

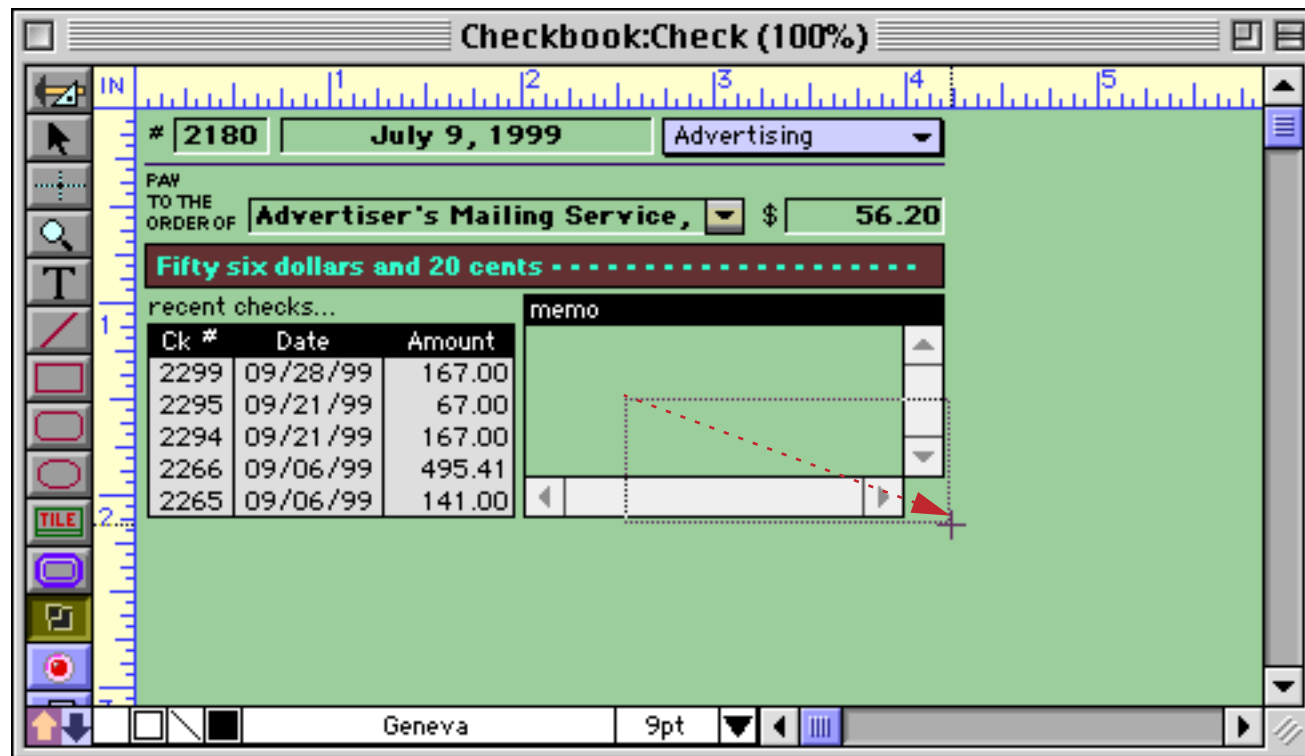
Once the form is created, the next step is to divide the form into four quadrants. This is done with the **Auto Grow SuperObject™**. The Auto-Grow tool is not in the default tool palette, so you’ll need to use the **Tool Palette** dialog to add this tool to the palette if it is not already there.



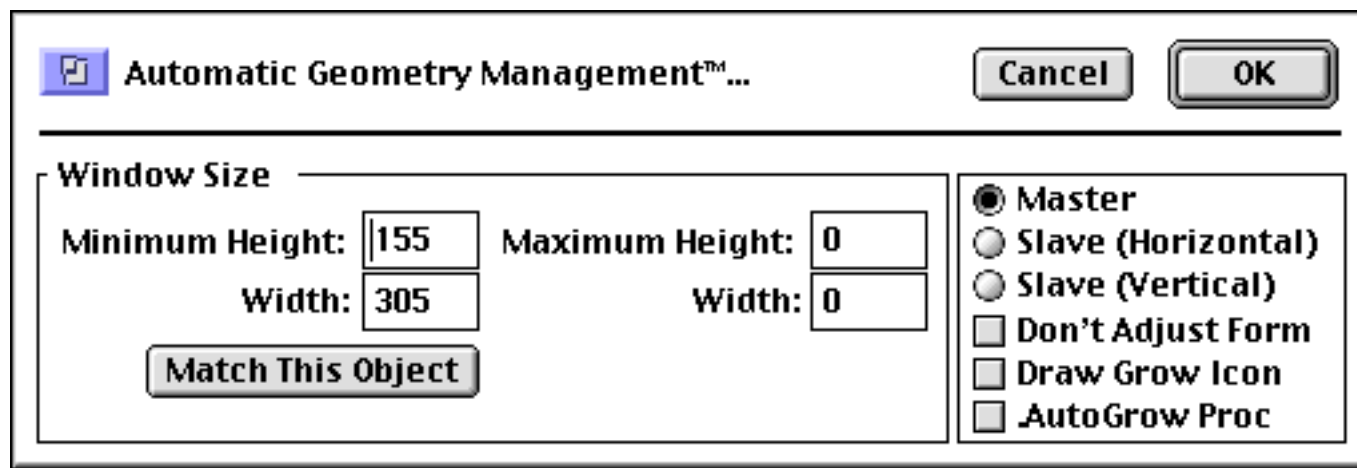
Now that the tool is added to the palette you can select it.



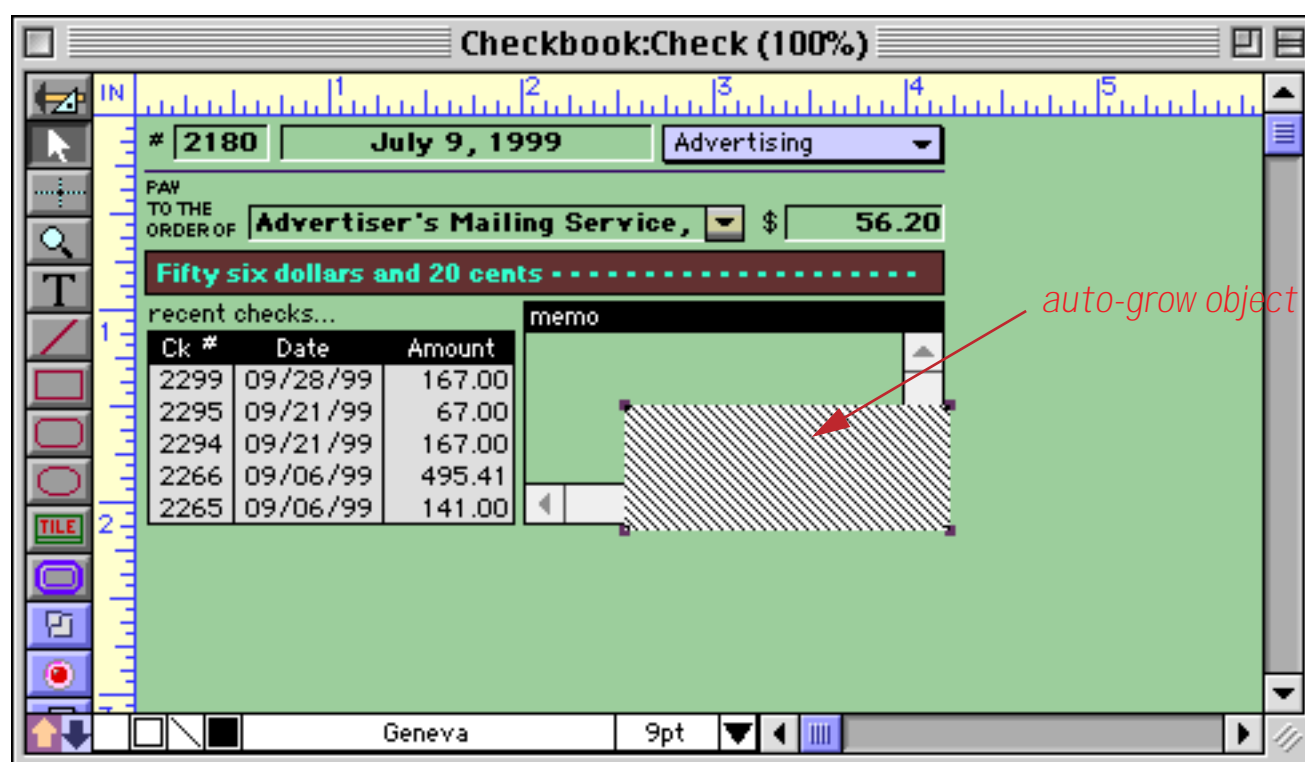
Once the tool is selected, drag the mouse across the form in the location where you want to create the auto-grow object. The Auto Grow SuperObject should cover the lower right hand quadrant of the form (the quadrant that will shift down and to the right diagonally when the window expands). The upper left corner of this Auto Grow SuperObject will determine the point where the four quadrants meet.



When you release the mouse, the Auto-Grow configuration dialog will appear.



For a basic elastic form just press the **OK** button.



Using the mouse and the arrow keys, make sure the Auto Grow object is positioned exactly where you want the lower right quadrant of the form to be. Be sure to allow some space for a slight margin on the right and bottom side of the form.

Once the Auto Grow object is set up, switch the form to Data Access Mode. At first, all you'll notice is that the Auto Grow object disappears.

Ck #	Date	Amount
2299	09/28/99	167.00
2295	09/21/99	67.00
2294	09/21/99	167.00
2266	09/06/99	495.41
2265	09/06/99	141.00

Now change the size of the window. As soon as you change the size of the window, the form will adjust to the new window size.

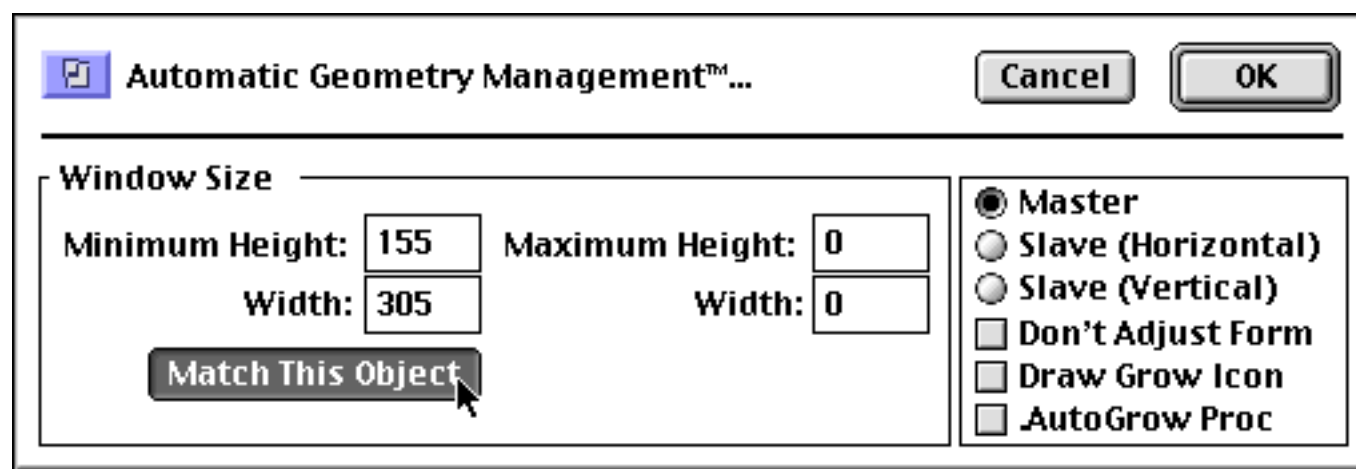
Try making the window larger or smaller, or zooming the window. The form will automatically adjust to the new size.

If the form doesn't quite adjust the way you wanted, set it back to the minimum size and then go into Graphics Mode and adjust the auto-grow object. In extreme cases you may need to **Revert To Saved** to get the original form configuration back (you *did* save the database before you tried your elastic form, right?).

Once the Auto Grow object is exactly positioned over the lower right quadrant of the form you can set the minimum window size. The following discussion assumes that the form is currently in its minimum window size configuration, with all expandable objects set to their smallest size. In other words, the bottom right hand corner of the Auto Grow object defines the smallest possible window size.

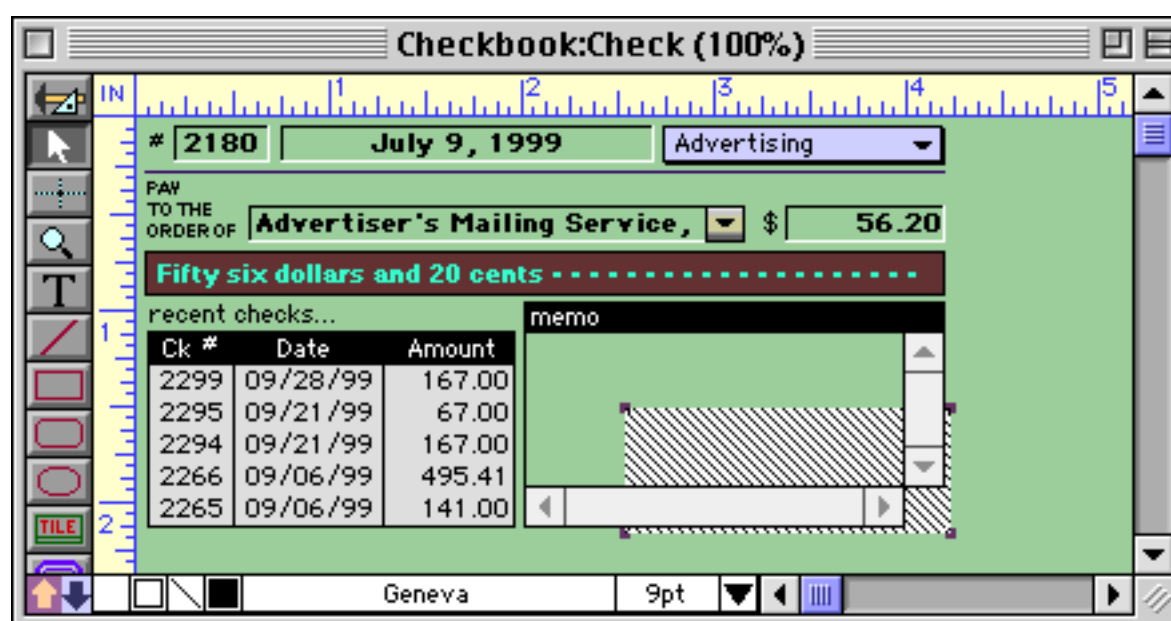


To set this minimum size, double click on the Auto Grow object, then click on the **Match This Object** button.



The dialog will show the dimensions of the minimum window size. Click on the **OK** button to permanently set this size.

After the minimum window size is set, use the **Send to Back** command (Arrange menu) to move the Auto Grow object underneath all of the other objects in the form.

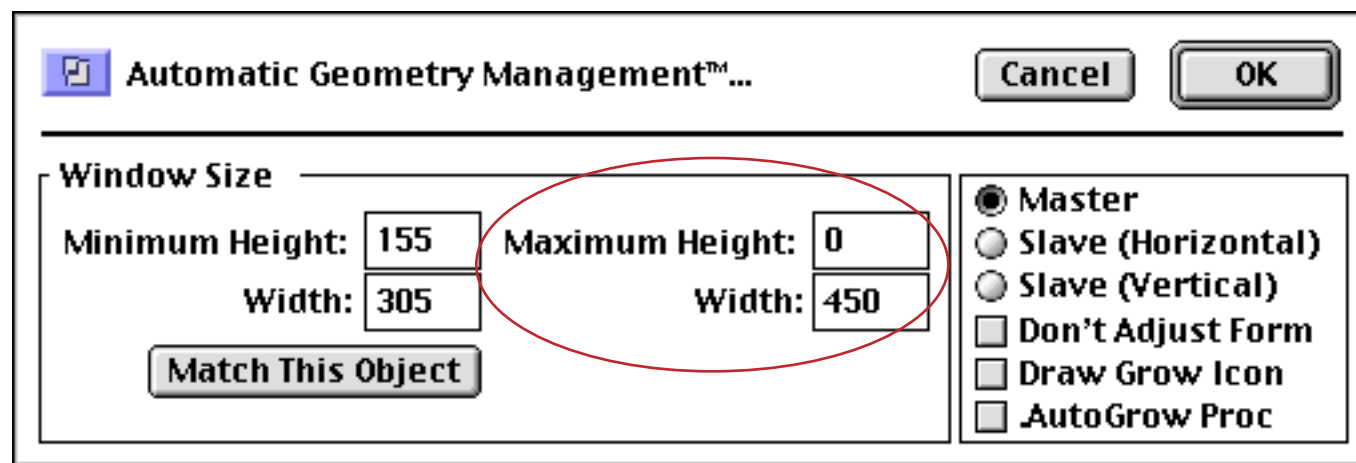


This prevents the Auto Grow object from interfering with the operation of other objects (for example buttons). Whether it is on top or on the bottom, the Auto Grow object will become invisible when the form is switched from Graphics Mode to Data Access Mode.

The Auto Grow dialog also allows you to manually set the minimum window size by typing in the dimensions. This is necessary if the form elements are not currently at the minimum window size. We recommend that you avoid this if at all possible. Only edit the form when it is in its minimum size configuration.

### Maximum Window Size

Panorama windows can normally be expanded to the full height and width of the computer screen (or even across multiple monitors if your computer has them). In some cases, however, you may wish to restrict the maximum height or width of the window. Double click on the auto-grow object to set the maximum dimensions.



In this case the **Maximum Height** is set to 0. When the maximum is set to zero, Panorama treats it as if there is no maximum. In other words, the vertical expansion of this window is unlimited, but the horizontal expansion is limited to 450 pixels. You can set a limit on the vertical expansion (height), horizontal expansion (width) or both.

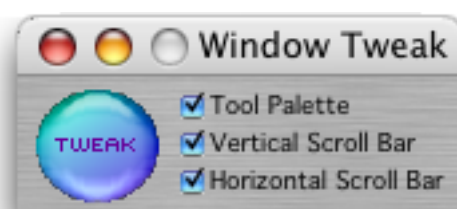
You can set a maximum window size even if the form is not elastic. To do this, create an auto-grow object anywhere on the form. In the configuration dialog, set the maximum dimension and also enable the **Don't Adjust Form** option. When this option is enabled Panorama does not adjust the objects in the form when the window is resized.

If you want the window to be a fixed size (not expandable) set the minimum and maximum dimensions to the same value.

### Removing the Window's Scroll Bars

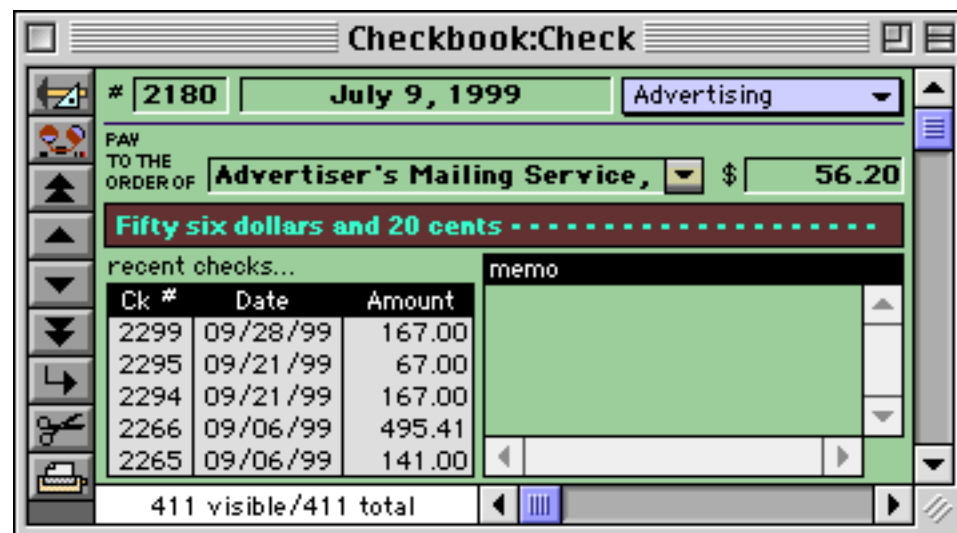
Form windows normally have scroll bars on the right and bottom edges of the window. When using an elastic form these scroll bars really aren't necessary because the form objects are always adjusted to fit inside the visible area. To remove the scroll bars use the **Window Tweak** wizard, which you'll find in the **Form Tools** submenu of the Wizard menu. When you select this wizard it will automatically toggle the scroll bars and tool palette from the current window (if they are currently visible they will be removed, if they are currently disabled they will be made visible).

When you open the **Window Tweak** wizard a small window appears in the lower right hand corner. The buttons on this window allow you to perform additional "tweaks" without using the menu.

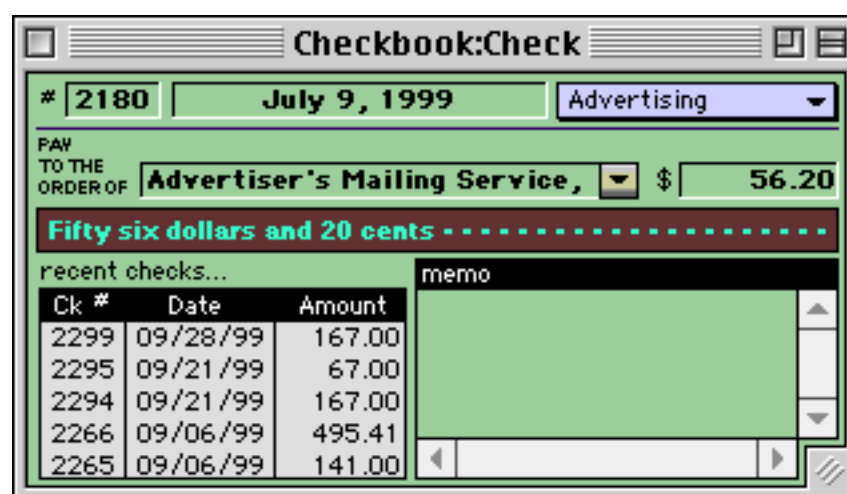


Press the **Tweak** button to toggle the palette and scroll bars of the window just below the Window Tweak window. Using the checkboxes you can modify the "tweak" action to exclude the tool palette or either scroll bar.

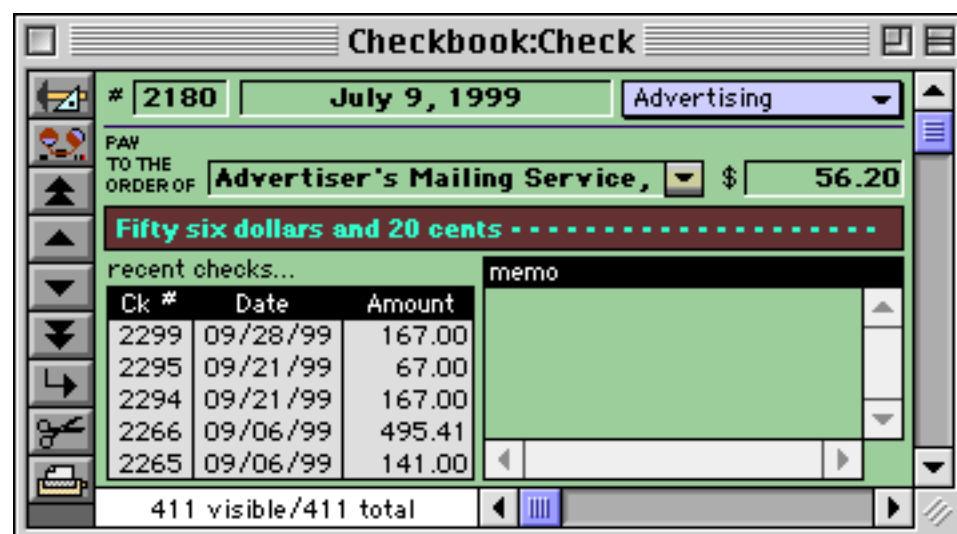
To illustrate the “tweak” action we’ll use this Checkbook database. Here is the original window.



Once the form is set up and ready to go, run the Window Tweak wizard. Voila! The scroll bars and tool palette magically disappear, and the window size shrinks slightly to compensate.



When you need to get the palette and scroll bars back again, run the Window Tweak wizard again (or click on the **Tweak** button). Poof! The missing items re-appear (and the window expands slightly).



Each time you run the Window Tweak wizard the scroll bars and palette will toggle on or off.

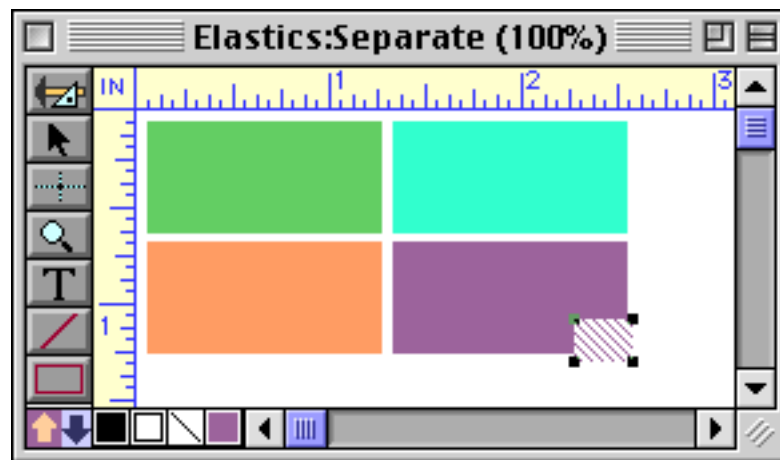
### Modifying an Elastic Form

If you need to change an elastic form, we recommend that you start by resizing the window to its minimum size. Once the window is at the minimum size, you can switch to Graphics Mode and then expand the window (if necessary). In Graphics Mode the window can be expanded without automatically adjusting the form.)

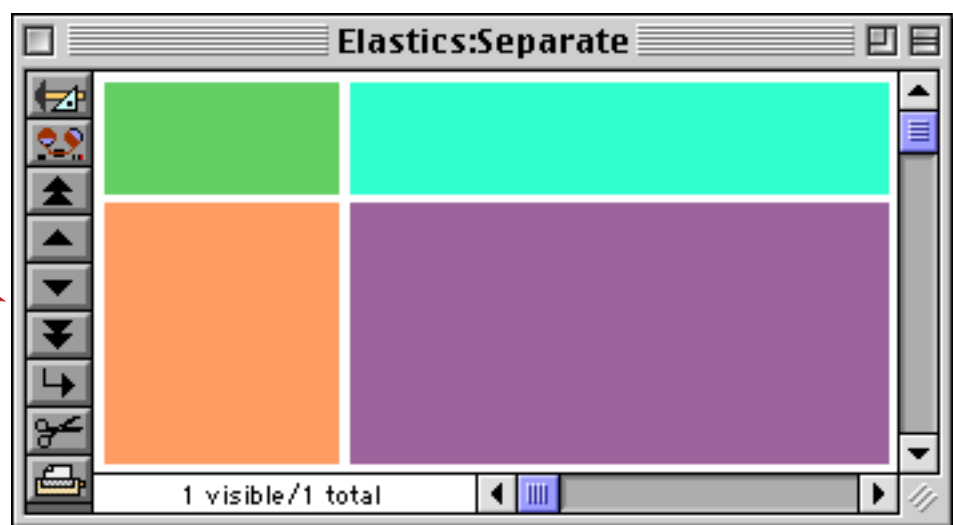
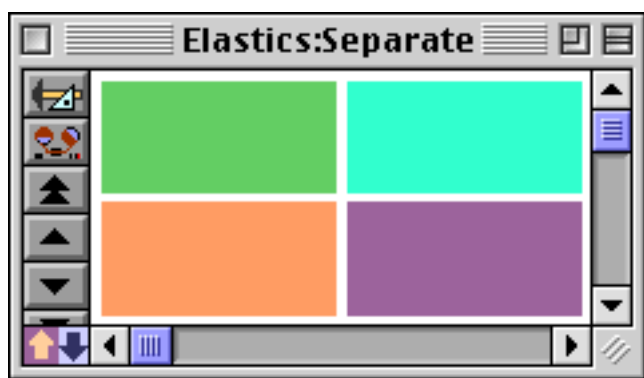
If you change the position or size of the Auto Grow object, you must re-set the minimum window size. Simply double click on the object, then click on the **Match This Object** button, just as you did before. The dialog will show the new minimum window size. Click on the **OK** button to permanently set this size.

#### Expanding Multiple Objects Proportionally

Panorama normally adjusts each object separately, either by expanding them or sliding each object. For example, consider the form shown below.

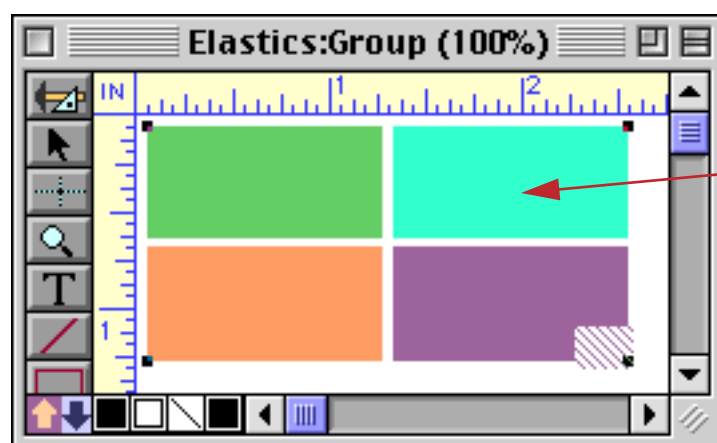


This form expands like this.



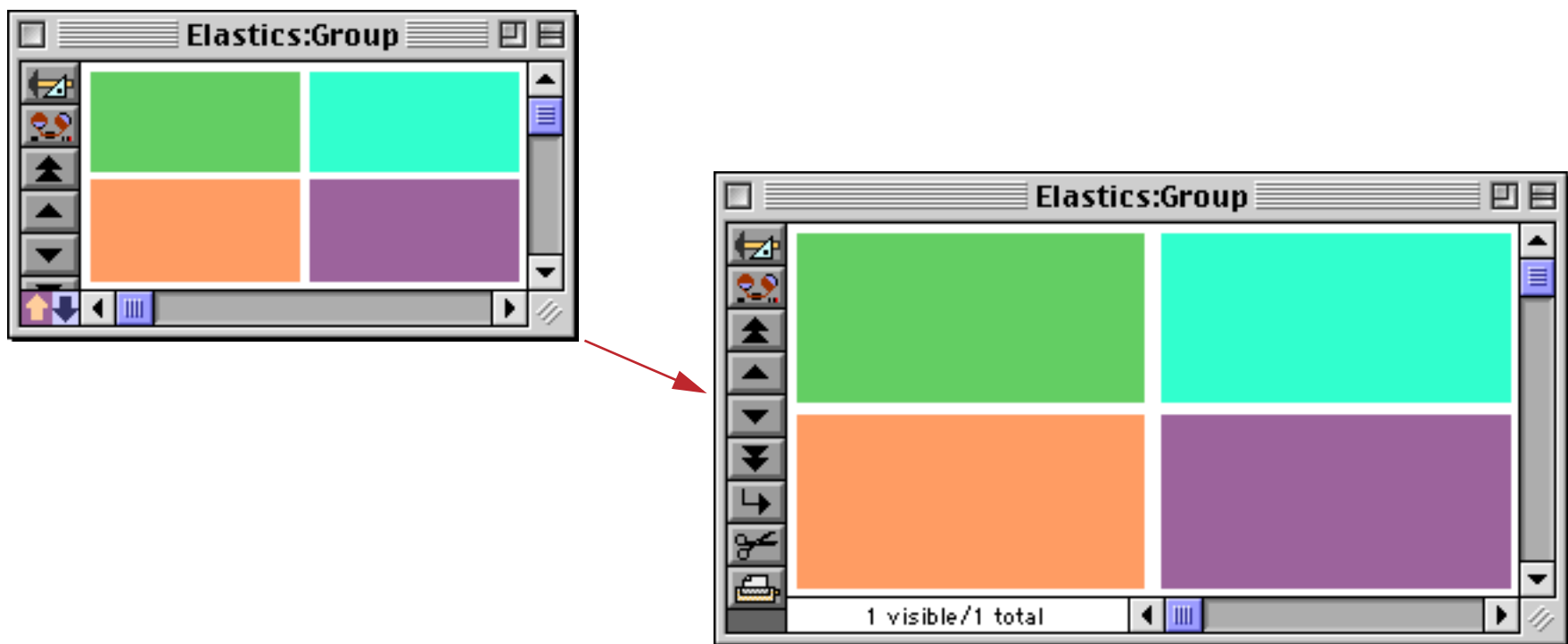
Sometimes, however, you may want several objects to expand or shrink proportionally as a group. For example, if the window grows by an inch, you want four objects to grow equally by 1/4 inch.

One way to do this is by grouping the objects (see [“Grouping Objects Together”](#) on page 514).



*four objects combined  
into single group*

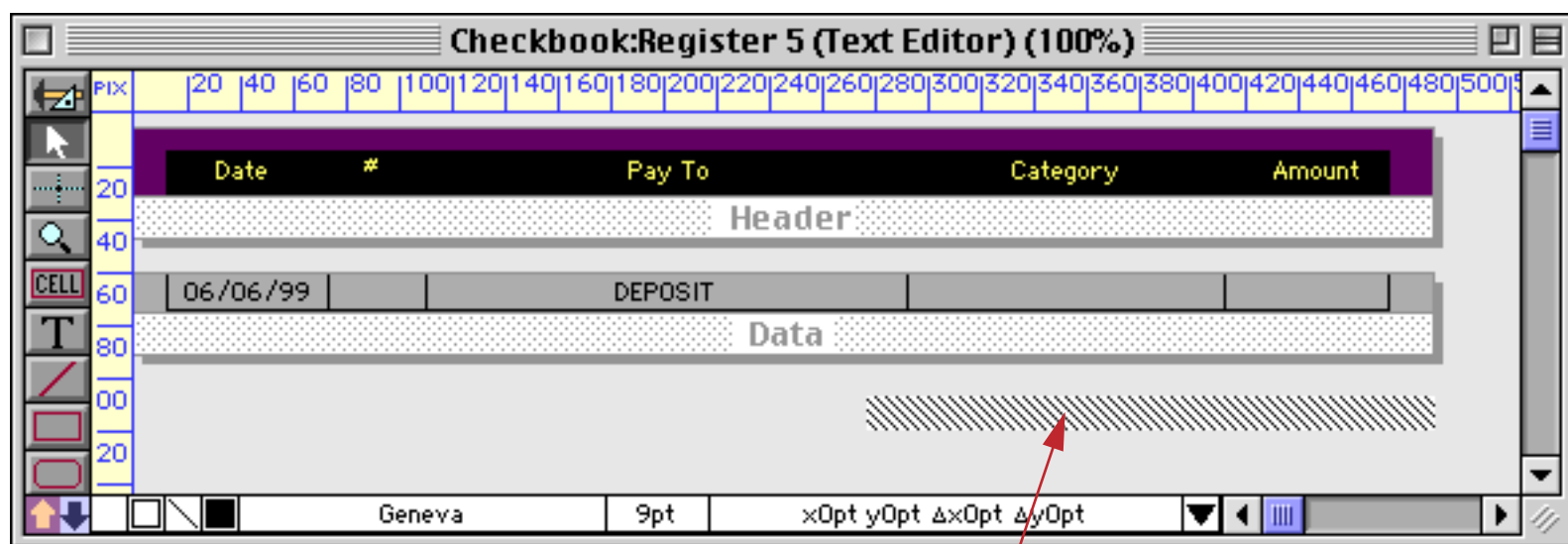
The form adjusting mechanism treats the group as a single object. Within the group, each object will expand or shrink proportionally as the entire group expands or shrinks. Notice that the gaps between the objects also expand proportionally.



Another method is to use a Matrix SuperObject. Again, as the entire matrix expands or shrinks, the individual cells in the matrix expand or shrink proportionally. This method is ideal for calendars. If you use the Text Display SuperObject to display text, the text can automatically increase or decrease in size as the form expands and shrinks (see “[Text Display Options](#)” on page 589 of the **Panorama Handbook**).

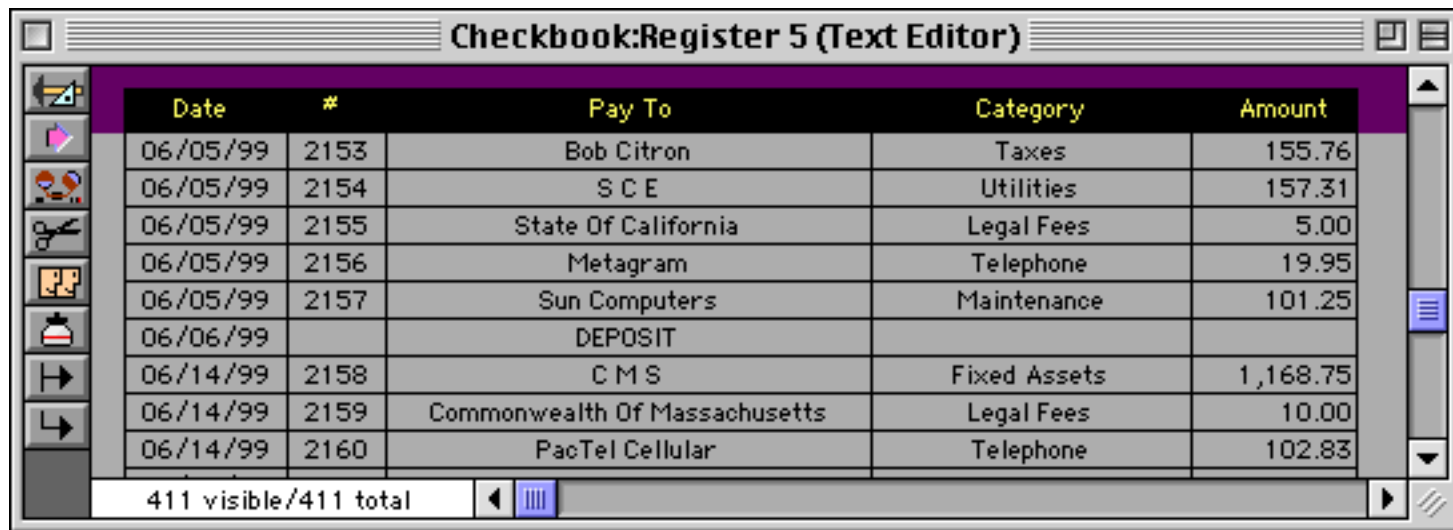
#### Elastic View-As-List Forms

Starting with Panorama 4.0 it is possible to create an elastic view-as-list form. In this case the vertical position of the auto-grow object is unimportant. The horizontal position of the object should correspond to the right edge of the visible area. It's easier to set this up if the data tile is aligned with the left edge of the form. In the example below the width of the Deposit field will increase when the form is expanded.



*auto-grow object*

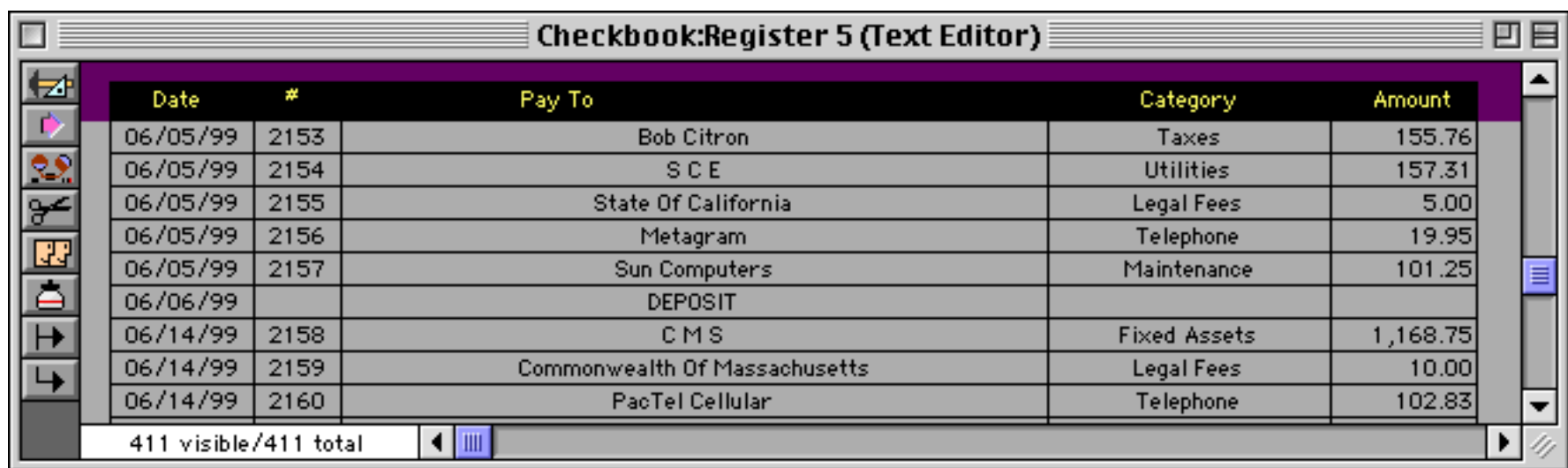
Here's what this form looks like in a minimum width configuration.



Date	#	Pay To	Category	Amount
06/05/99	2153	Bob Citron	Taxes	155.76
06/05/99	2154	S C E	Utilities	157.31
06/05/99	2155	State Of California	Legal Fees	5.00
06/05/99	2156	Metagram	Telephone	19.95
06/05/99	2157	Sun Computers	Maintenance	101.25
06/06/99		DEPOSIT		
06/14/99	2158	C M S	Fixed Assets	1,168.75
06/14/99	2159	Commonwealth Of Massachusetts	Legal Fees	10.00
06/14/99	2160	PacTel Cellular	Telephone	102.83

411 visible/411 total

When the window is expanded, the form adjusts automatically.



Date	#	Pay To	Category	Amount
06/05/99	2153	Bob Citron	Taxes	155.76
06/05/99	2154	S C E	Utilities	157.31
06/05/99	2155	State Of California	Legal Fees	5.00
06/05/99	2156	Metagram	Telephone	19.95
06/05/99	2157	Sun Computers	Maintenance	101.25
06/06/99		DEPOSIT		
06/14/99	2158	C M S	Fixed Assets	1,168.75
06/14/99	2159	Commonwealth Of Massachusetts	Legal Fees	10.00
06/14/99	2160	PacTel Cellular	Telephone	102.83

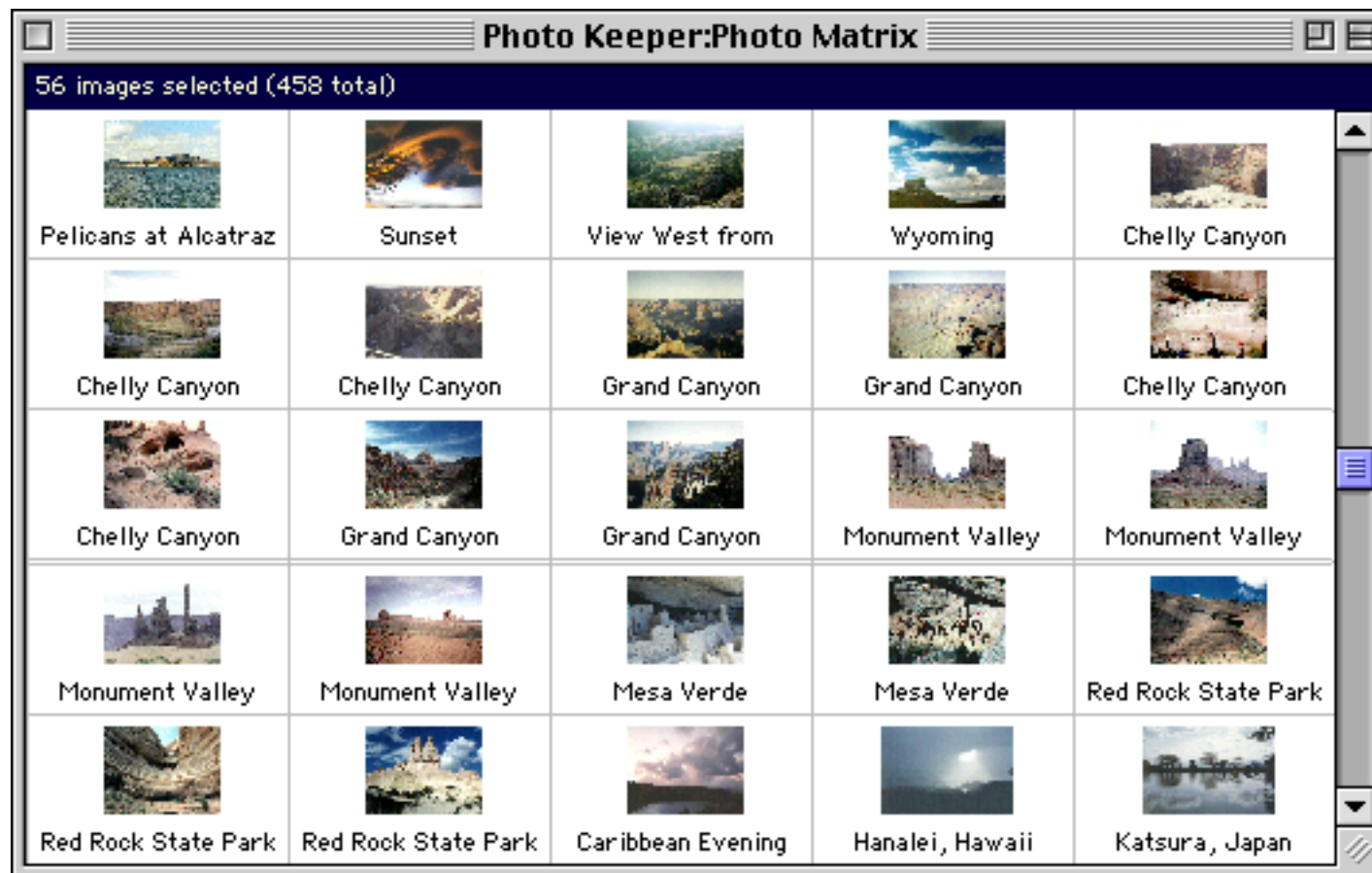
411 visible/411 total

You can also use the auto-grow object to set the minimum and maximum dimensions of the view-as-list form, just as with a regular form.



## Super Matrix Objects

Some applications require a rectangular array (or matrix) of data, pictures, and/or pushbuttons (for example a monthly calendar or a thumbnail artwork preview).



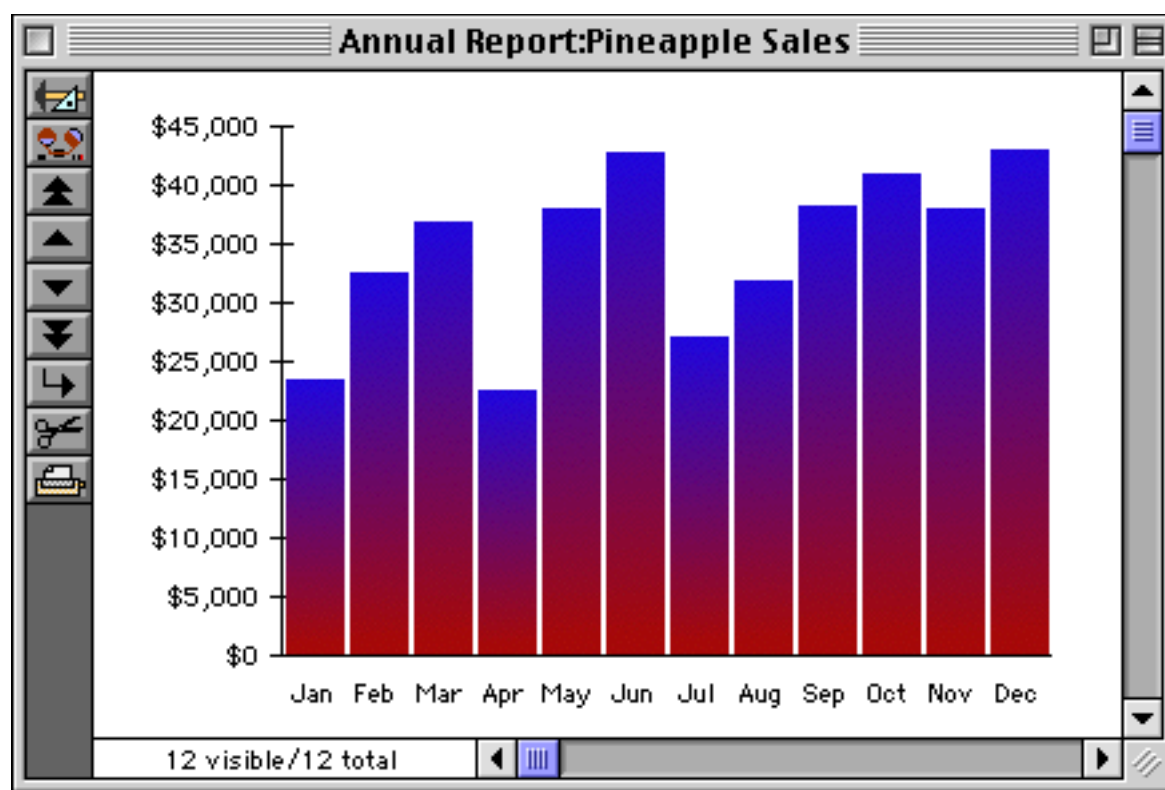
Such a matrix consists of a series of cells assembled into rows and columns. Of course, you can build up such a matrix from individual objects, but the Super Matrix SuperObject™ allows the entire matrix to be built by repeating a single template. The template may contain artwork and text that will be repeated over and over again for each cell in the matrix. The template can use formulas to display the appropriate information in each column and row. This system has several advantages: 1) The array can be built quickly, 2) If changes are necessary later, they only have to be made once in the template and are automatically repeated throughout the entire matrix, 3) It is very easy to change the number of columns or rows in the matrix, 4) The matrix and template can be constructed to adjust automatically as the window changes size and shape.

The creation of Super Matrix objects is beyond the scope of this book. To learn all of the details see Chapter 18 of the [Panorama Handbook](#).

# Chapter 19: Charts



Many databases are filled with numbers. A chart can bring those numbers to life, so Panorama can draw five different kinds of charts—Bar, Line, Area, Pie, and Scatter.



To learn more about creating charts with Panorama see Chapter 19 of the [Panorama Handbook](#).



# Chapter 20: Printing Basics



Since we haven't quite arrived at the age of the totally paperless office, printing is still an important function of any computer program—including Panorama. This chapter covers the basics of printing. In the next chapter you'll learn how to design and use custom reports.

## Printing Different Views

You can print any of Panorama's six kinds of views—data sheet, design sheet, flash art gallery, form, crosstab, or procedure. To print a view, you must make it visible in the top window. Use the **View** menu if the view you want to print is not currently visible and on top.

Once the view is visible in the top window, use the **Print** command in the File Menu to print the contents of the view.

## Printing the Data Sheet

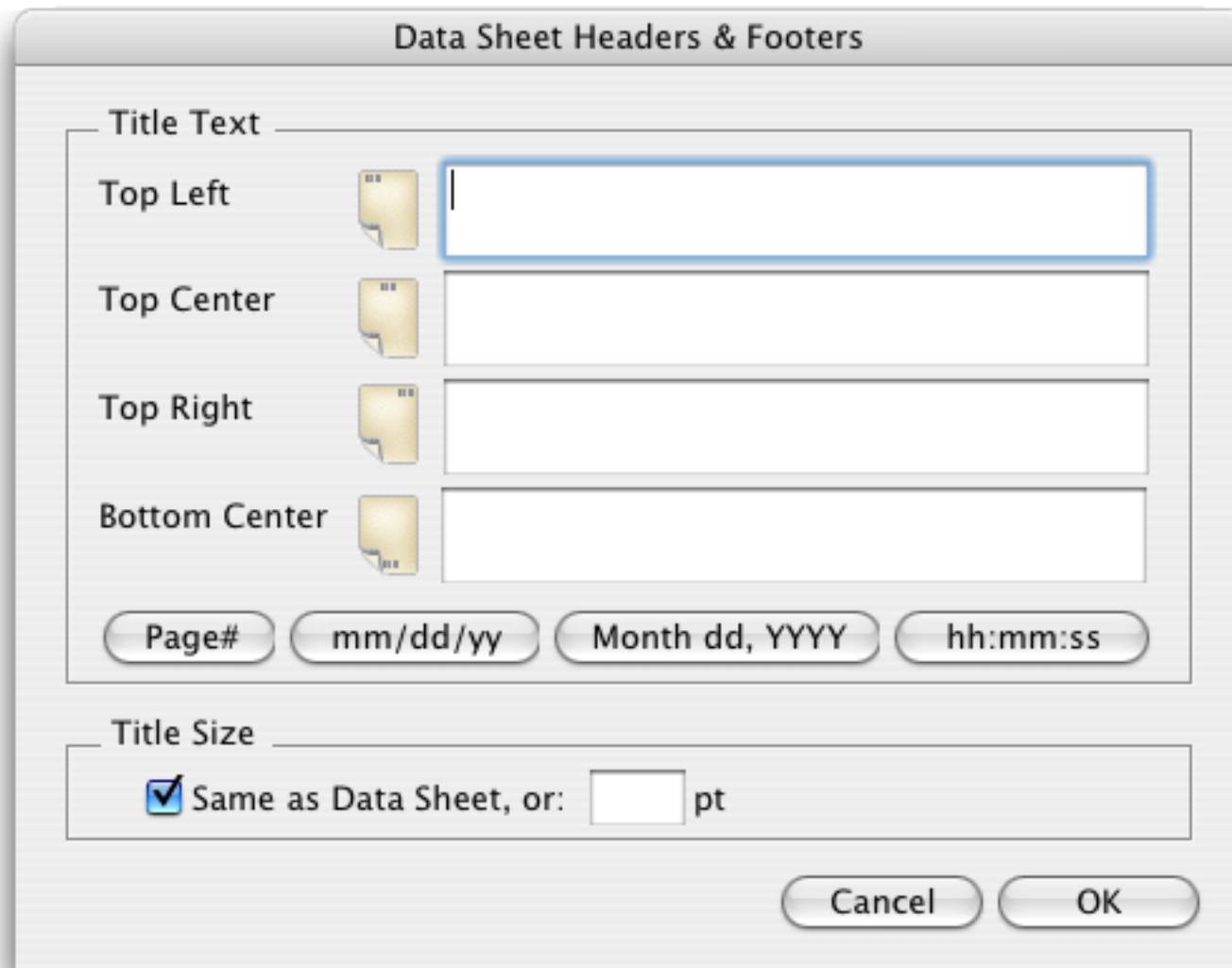
Panorama prints the data sheet exactly as it appears on the screen. If the data sheet is too wide to fit on the page, Panorama will print extra pages until all the columns are printed.

You can get more data sheet columns on a page several ways. One method is to use the **Page Setup** dialog to switch to a wide paper orientation (sideways or landscape), or to reduce the printout to a smaller size (Macintosh only). You can also use a smaller font.

You can also print the design sheet or a crosstab sheet. These views will also print extra pages if they contain more columns than will fit on a single page.

### Printing Data Sheet Headers & Footers

The Headers/Footers dialog (File Menu) sets up headers and footers for the data sheet, design sheet, or any crosstab. (You can set up separate headers and footers for each of these windows.)

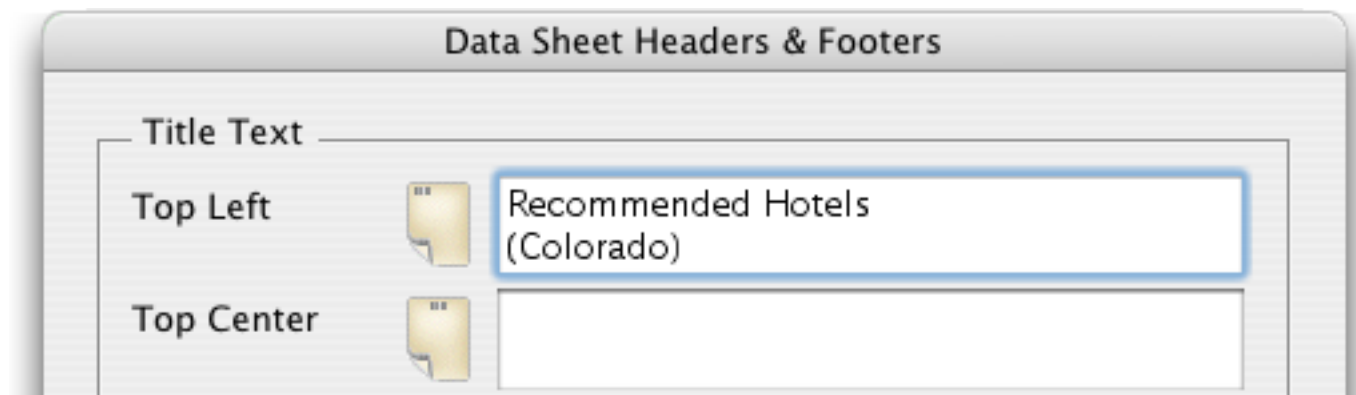


The Headers/Footers dialog allows you to position headers and footers in four locations on the printed page: top left, top center, top right, and bottom center.

Top Left Header		Top Center Header			Top Right Header	
Hotel	City	Rate	Units	Phone	Stars	
6 & 40 Motel	Idaho Springs	19.00	30	567-2691	2	
ABC Motel	Gunnison	26.00	18	641-9909	2	
Airport Village Motor H	Denver	34.00	131	388-4821	3	
Alamosa Inn Best Weste	Alamosa	35.00	143	589-5123	3	
Best Westernn Slpiner	Steamboat Springs	34.00	32	879-1430	3	
Best Westernn Spa Motor	Denver	35.00	70	292-0220	3	
Best Westernn Stagecoac	La Junta	30.00	60	384-5476	3	
		Bottom Center Header				



If you want to create a header or footer that is more than one line high, just press the **Return** key and type in the additional lines, like this.



When printed this header will look like this.

Recommended Hotels (Colorado)						Page 1
Hotel	City	Rate	Units	Phone	Stars	
6 & 40 Motel	Idaho Springs	19.00	30	567-2691	2	
ABC Motel	Gunnison	26.00	18	641-9909	2	
Airport Village Motor H	Denver	34.00	131	388-4821	3	
Alamosa Inn Best Weste	Alamosa	35.00	143	589-5123	3	
Alamosa Lamplighter M	Alamosa	30.00	73	589-6636	3	

You can insert special codes into the header/footer text to print the page number, date, and time. Here's an example of how to insert the page number (the printed result is shown above). On the Macintosh the « » chevron characters are produced by typing **Option-\  
On PC systems these characters are produced by typing **Alt-0171** and **Alt-0187**.**

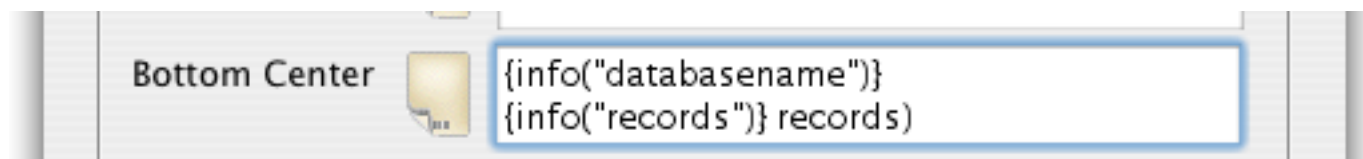


The table below lists some of the special codes that can be inserted into a header or footer:

Description	Code	Example
Page Number	«page #»	1
Date	«date:mm/dd/yy»	3/7/02
Date	«date:Month ddnth, yyyy»	April 8th, 2003
Time	{timepattern(now),"hh:mm:ss am/pm"}	2:23:12 PM
Database Name	{info("databasename")}	Hotels



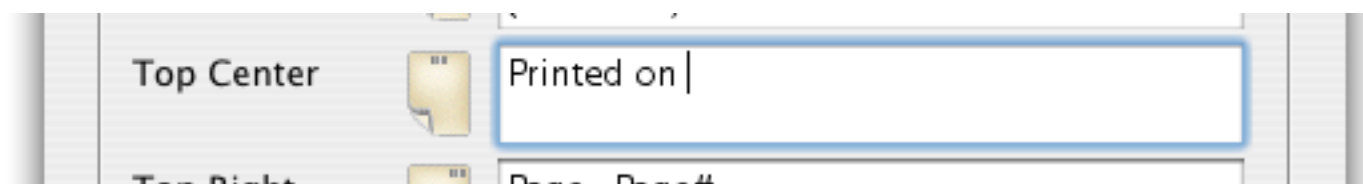
As the last two entries in this table show, you can actually insert any formula into a header or footer by surrounding the formula with { and }. Here's an example that uses two formulas to display the database name and number of records in the footer.



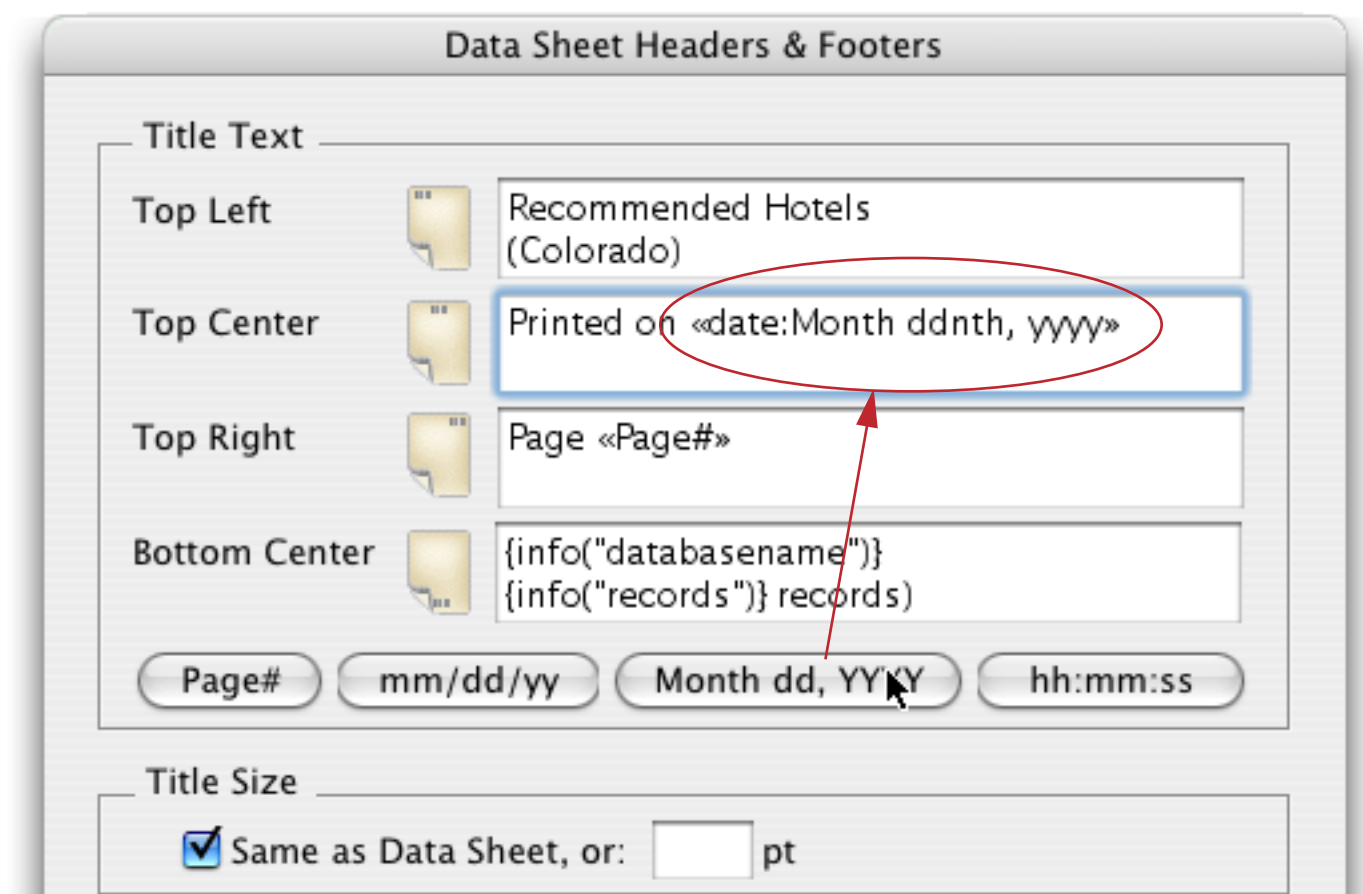
Here's what the finished footer looks like. See "[Formulas](#)" on page 1155 to learn more about formulas.

Best Western Sands	Cortez	30.00	81	565-3761	4
Best Western Shangri L	Breckenridge	38.00	41	453-2225	3
Best Western Silver Ki	Leadville	33.00	62	486-2610	3
Best Western Slpiner	Steamboat Springs	34.00	32	879-1430	3
Best Western Spa Motor	Denver	35.00	70	292-0220	3
Colorado Hotels ( 439 records)					

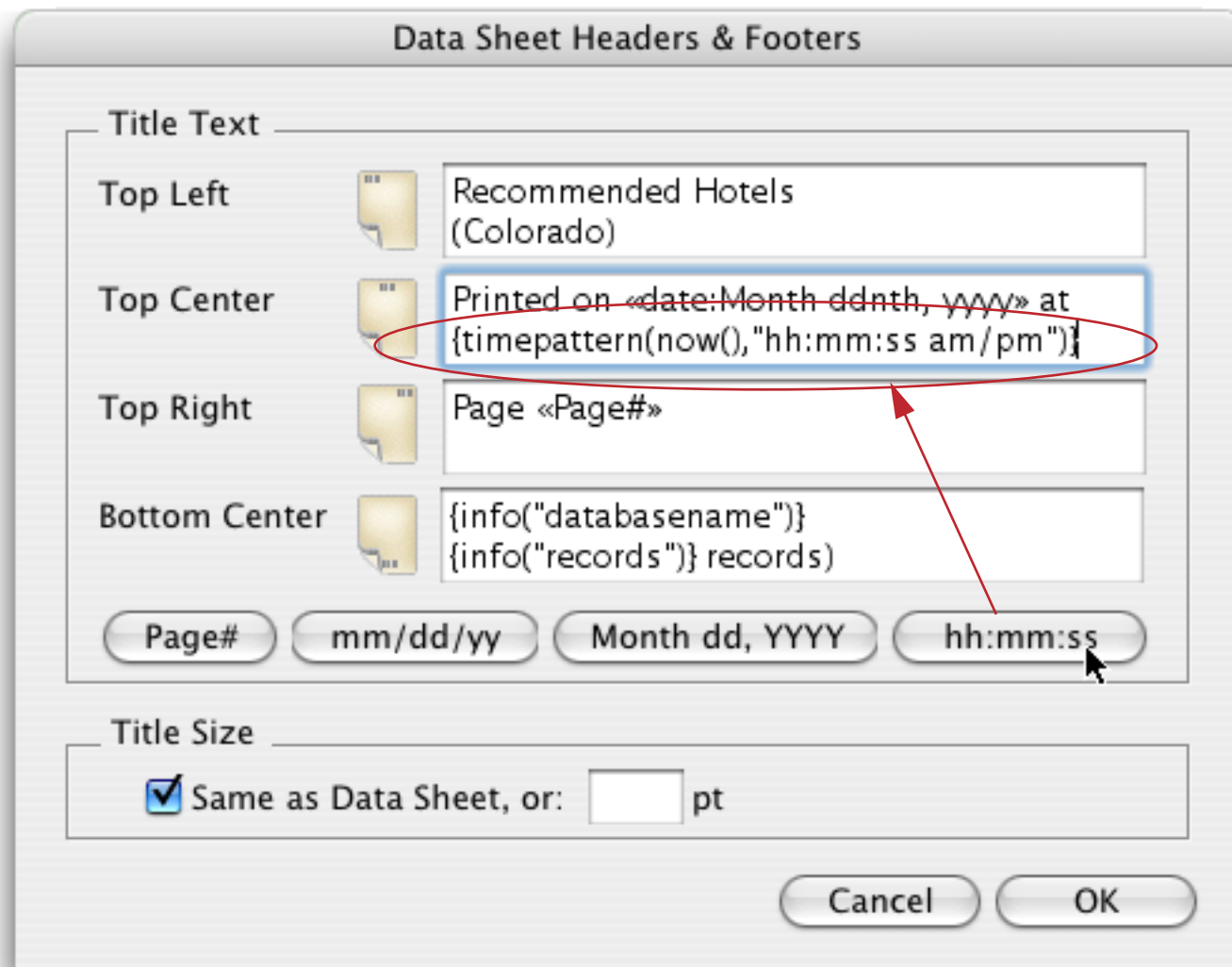
The four buttons just below the header/footer area will insert the most common codes into a header or footer for you. For example, suppose you wanted the top center header to show the time and date the database was printed, like this: **Printed on May 23rd, 2000 at 4:21 PM**. Start by opening the Headers/Footers dialog. Type **Printed on** into the **Top Center** header.



Now press the **Month dd, YYYY** button. Panorama will insert the code for you.



Next type in **at** and press the **hh:mm:ss** button.



Now press **OK**, and print or preview the data sheet. The top of the printed page will look like this:

Hotel	City	Rate	Units	Phone	Stars
6 & 40 Motel	Idaho Springs	19.00	30	567-2691	2
ABC Motel	Gunnison	26.00	18	641-9909	2
Airport Village Motor H	Denver	34.00	131	388-4821	3
Alamosa Inn Best Weste	Alamosa	35.00	143	589-5123	3
Alamosa Lamplighter M	Alamosa	30.00	73	589-6636	3
Alpenglo Motor Lodge	Winter Park	44.00	12	726-5294	2
Alpine Motel	Duray	28.00	12	325-4546	2
Alpine North Motel	Durango	34.00	21	247-4042	3

## Printing a Form

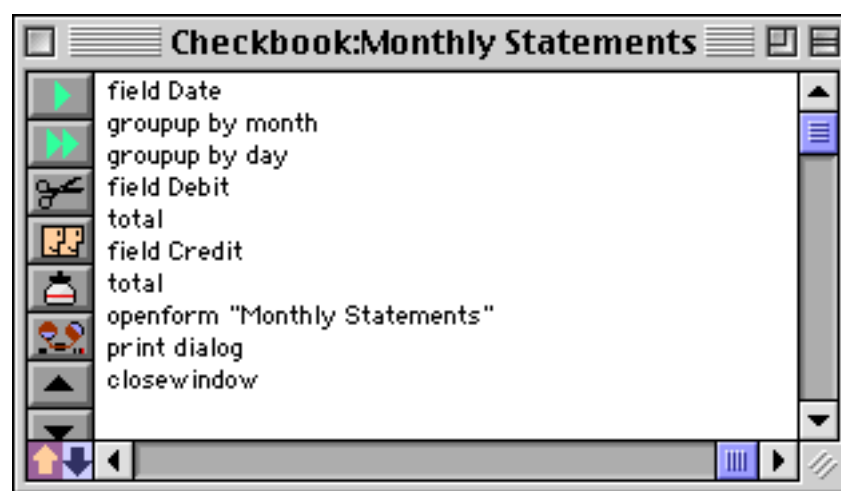
When you ask Panorama to print a data sheet, it prints an exact line-for-line replica—with one line for every visible record in the database. But when a form view is printed it would not be practical to print the entire form for every record. Instead, Panorama lets you specify what part of the form should be printed where on the report. To do this you use special graphic objects called **report tiles**. Each report tile defines a section of the form that is to be included in a printed report. By combining several tiles you can create a complex report with headers, footers, subtitles, summaries, etc. Report tiles and their applications are covered in detail in the following chapter, see “[Custom Reports](#)” on page 1039.

If you ask Panorama to print a form that has no report tiles, it will simply print the upper left hand corner of the form (8" by 10"), one record per page. In other words if your database contains 187 visible records, Panorama will print 187 pages. For most reports you will want to use report tiles so that more than one record is printed per page.

## Preparing Data For Printing

Before you print the database, you may want to prepare it for printing. If you want the data printed in a certain order (for example alphabetical by name), you must sort the database before you print it (see "[Sorting](#)" on page 325). If you want to print only a portion of the database (for example, only zip codes in California), you must use the **Find/Select** command to make the rest of the database invisible (see "[Searching and Selecting](#)" on page 333). If you want to print subtotals or other summary information, you must group and total the database before printing (see "[Summaries and Outlines](#)" on page 371).

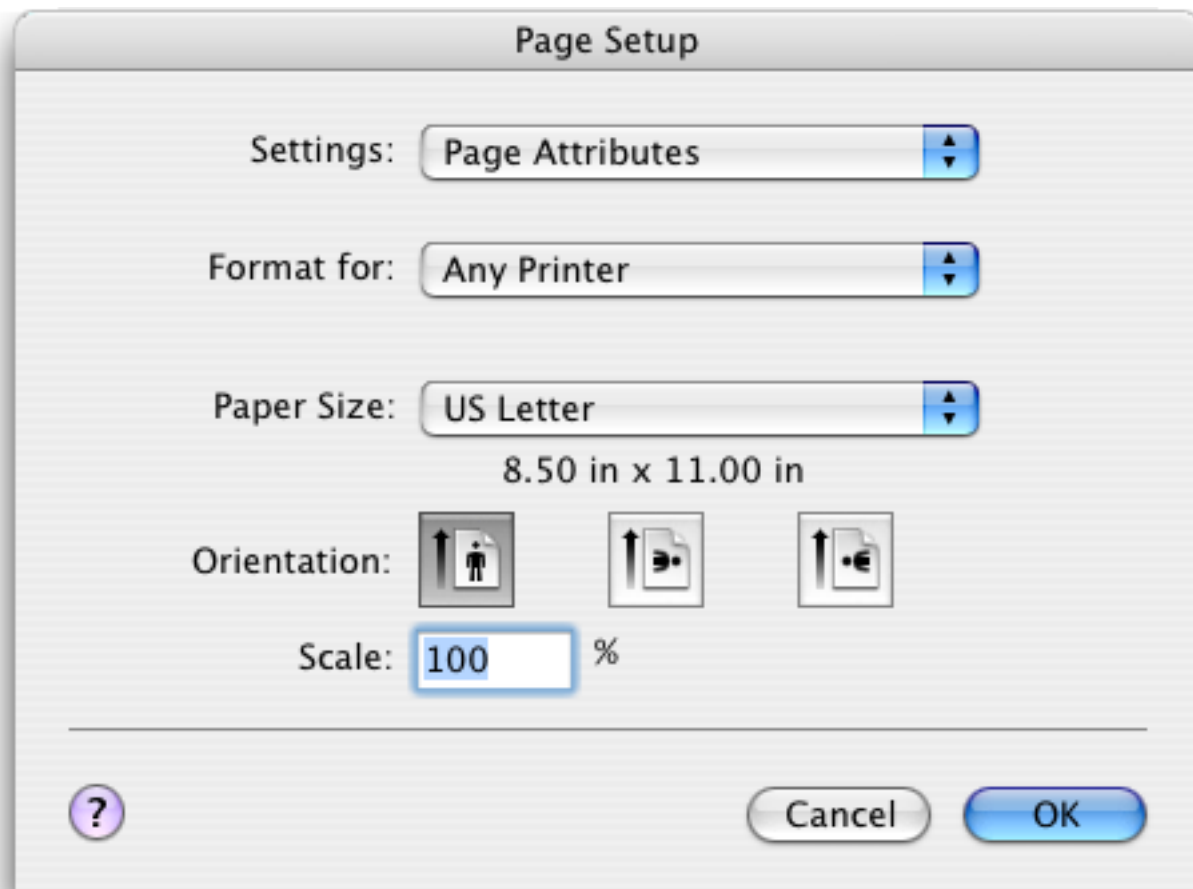
If you are going to print a report often, you may want to create a procedure to automatically prepare the data before printing. For example, here is a procedure that groups the database by month and by day, calculates the totals, then opens the form [Monthly Statements](#) and prints the database.



For more information on procedures see "[Procedures](#)" on page 1337.

## The Page Setup Dialog

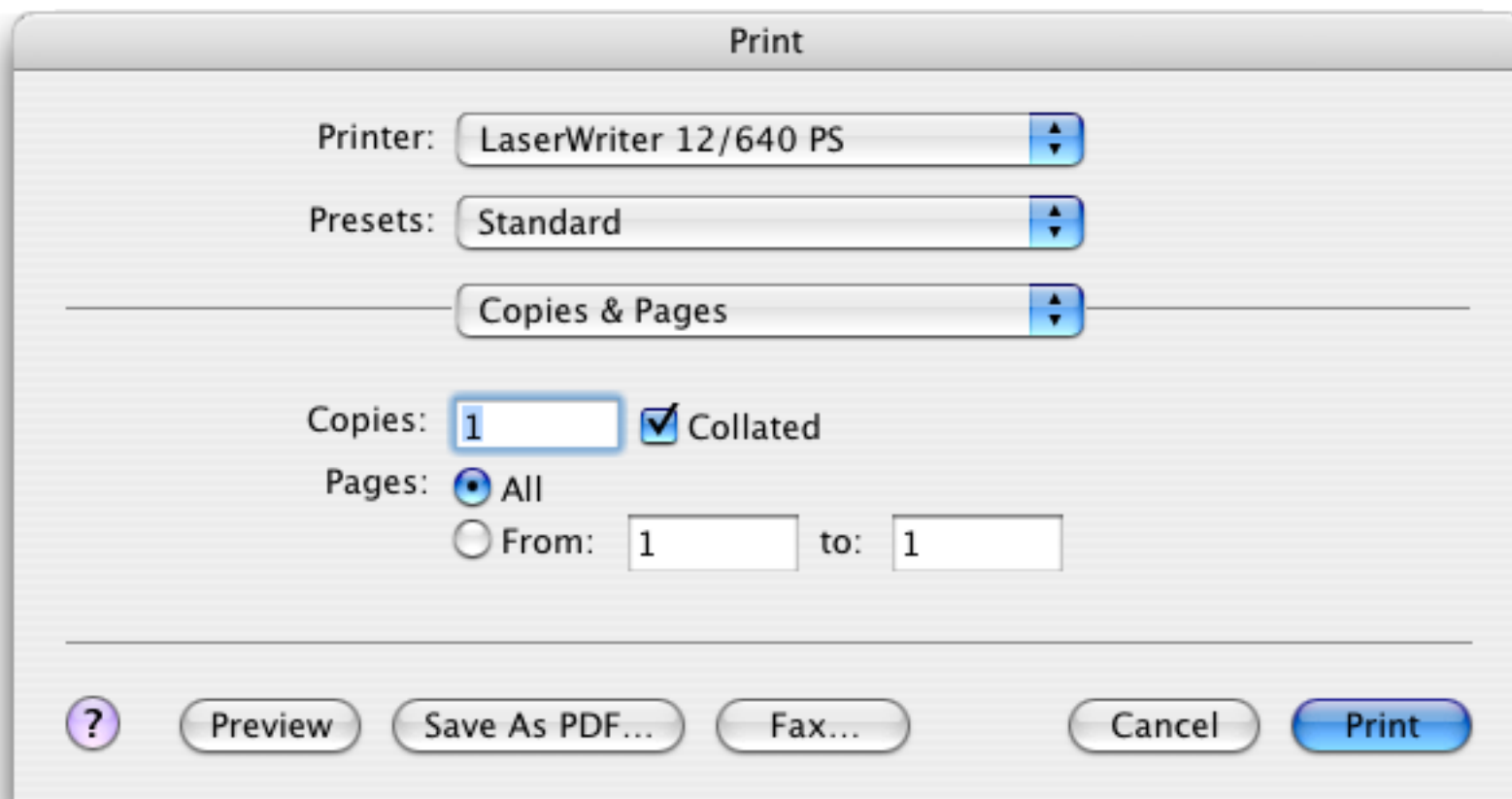
The **Page Setup** command in the File Menu displays a dialog that allows you to specify several printing options. The exact options available depend on the operating system and what kind of printer you are using, but in general you can control the page size, orientation (tall or wide), and print reduction factor. Here is a typical **Page Setup** dialog.



Each form view has its own separate page setup. The page setup is remembered as part of the form. For example, a single database can have an invoice that is printed using the tall orientation, and a report that is printed using the wide orientation (sideways). You don't have to remember to switch the page setup when you switch forms—Panorama will do it for you. Incidentally, be sure to save the file after you change the page setup. If you save the file, Panorama will remember the page setup the next time the file is opened. (However, not all print options are saved as part of the database. The exact options that are saved vary from printer to printer.)

## The Print Dialog

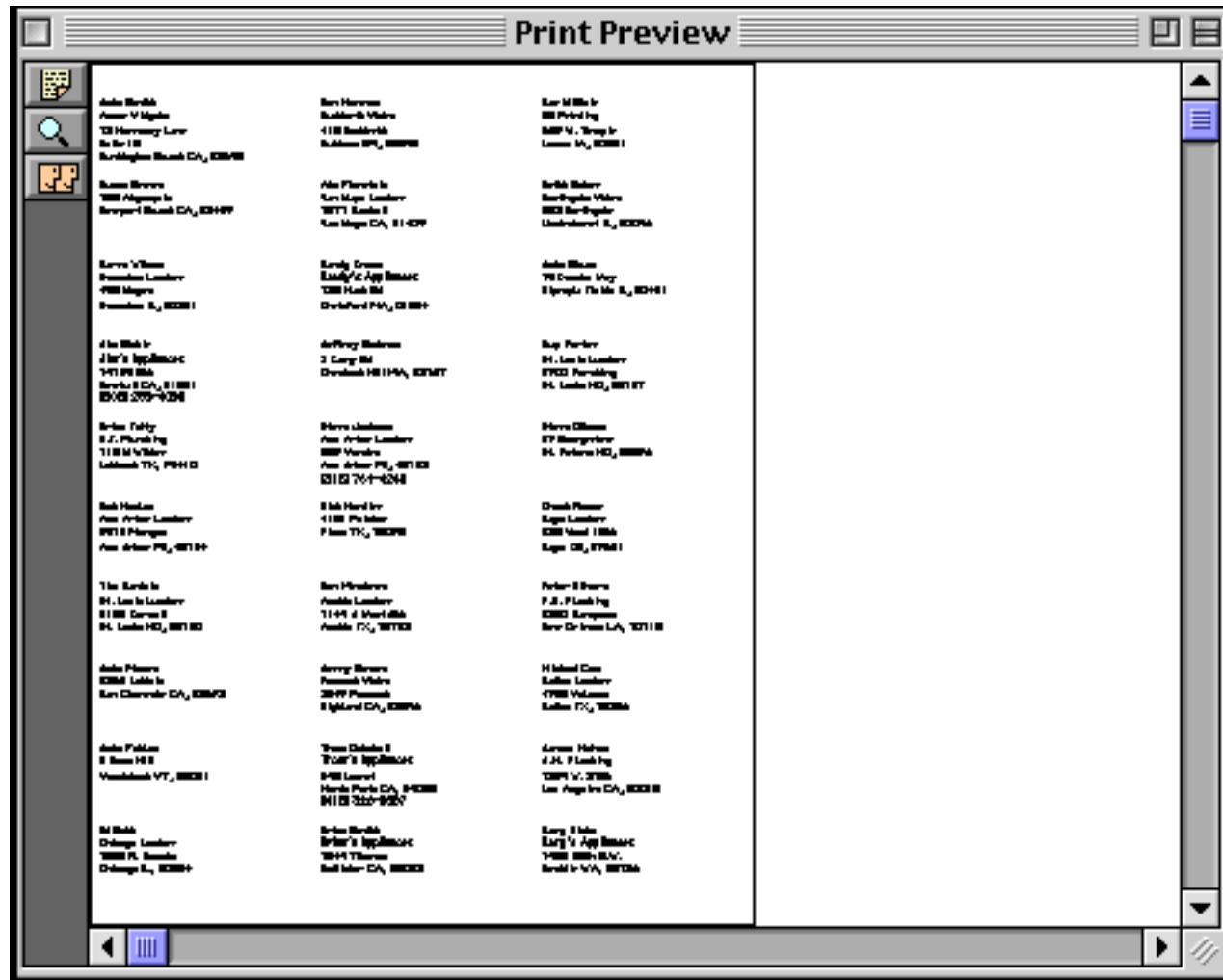
The **Print** command also displays a dialog box allowing you to choose printing options. You can choose which pages to print, how many copies to print, and whether you want to manually feed the paper. The exact options will depend on the operating system and printer you are using. Here is a typical **Print** option dialog.



For the exact details on the operation of this dialog see the documentation that came with your printer.

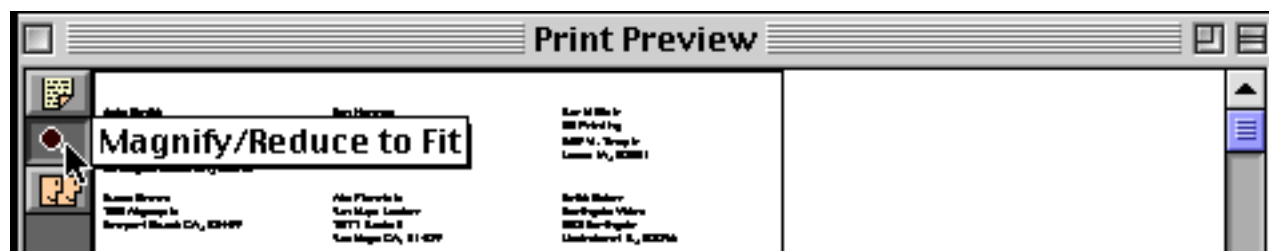
## Print Preview

The **Print Preview** command (File Menu) allows you to see what a report will look like without using any paper. This command opens a new window called **Print Preview**.



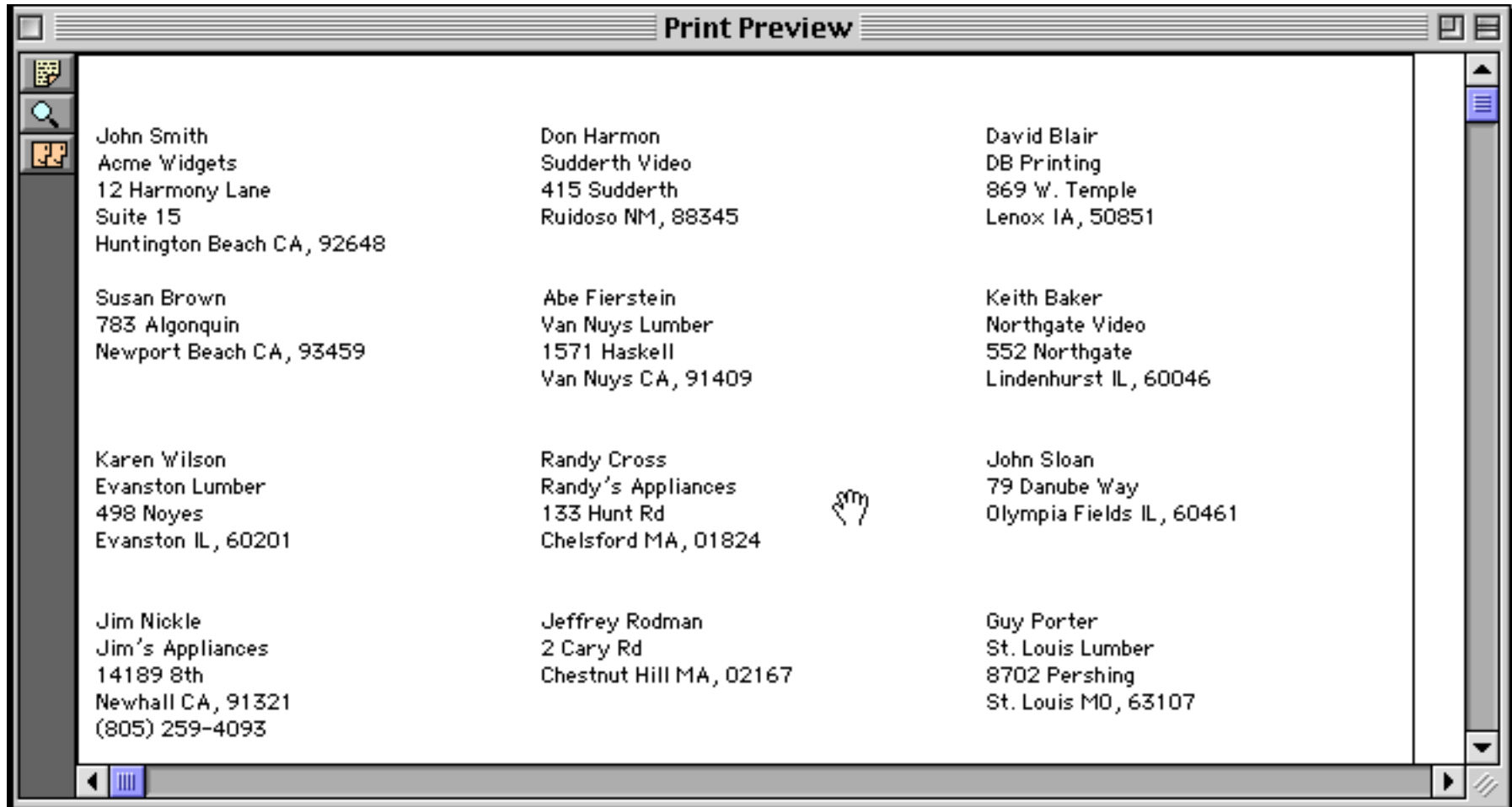
The preview window normally fills the entire screen, but you can change the size and location of the preview window after it is opened using the grow box and drag bar. Only one preview window can be open at a time, and all other Panorama windows are disabled when the preview window is open. You can still see the other windows, but you cannot bring them to the front or click on them.

The image of the printed page is reduced so that the entire page fits in the window. To expand the image to full size, click on the magnifying glass tool.



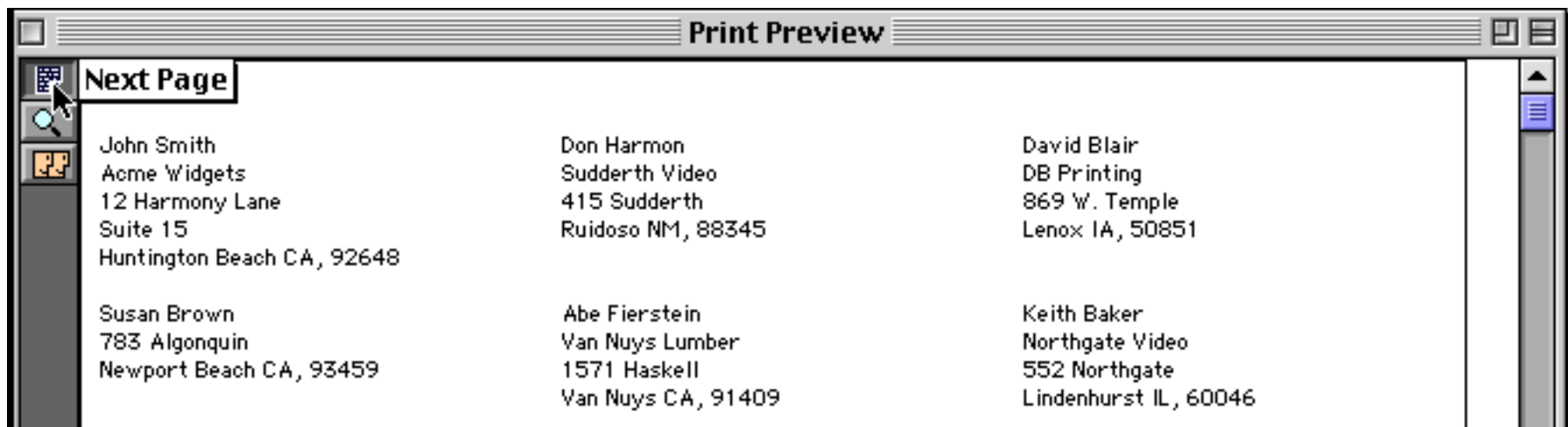


Each time you click on this tool, the window toggles between reduce to fit and 100% magnification, like this.



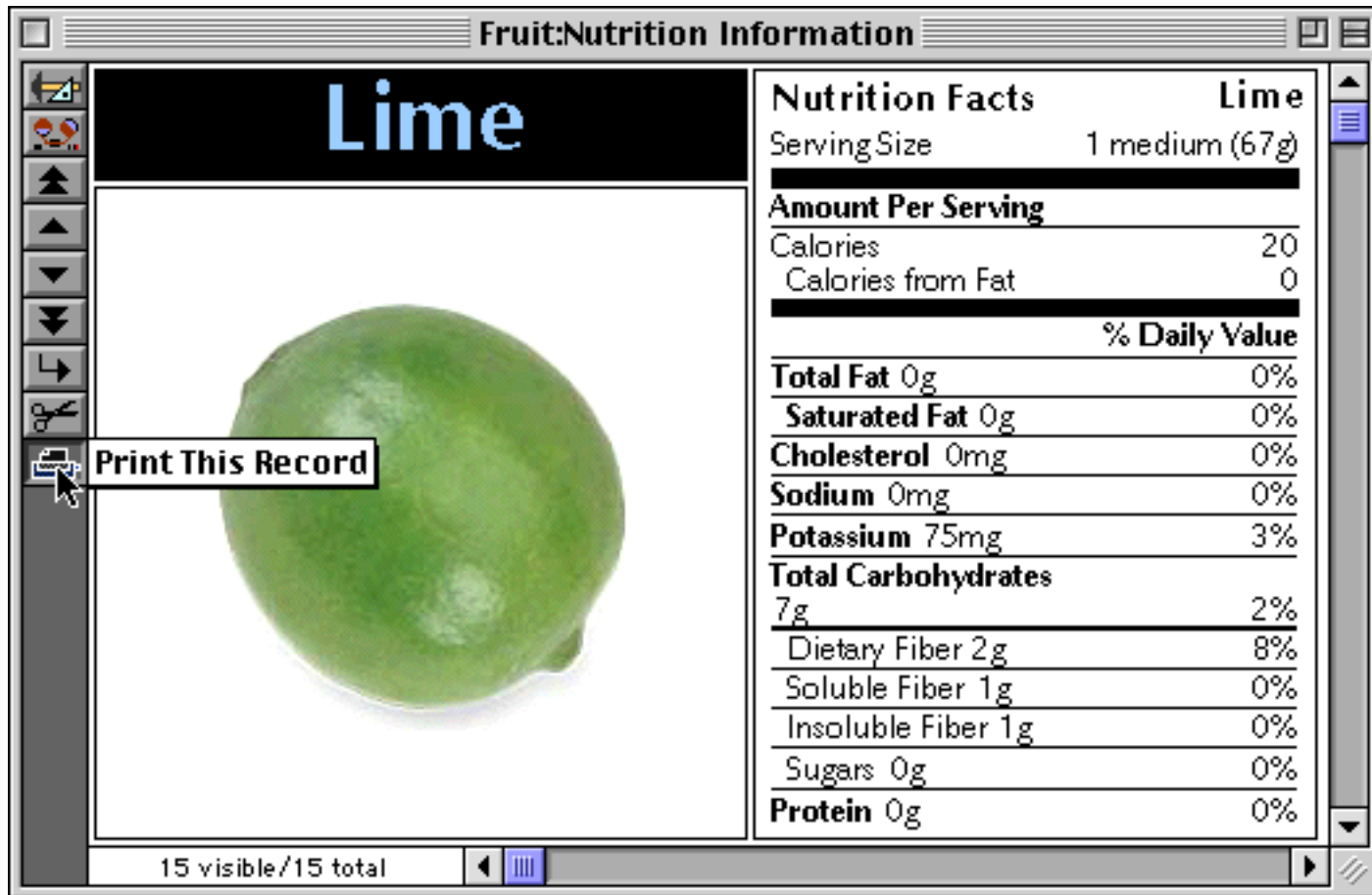
When the preview is magnified to 100%, you can use the scroll bars to shift to different locations within the page. You can also use the mouse to push the preview to a different spot. Simply press the mouse (which looks like a hand, see illustration above) on the preview window, then drag the image to a new location.

To preview the next page, click on the **Next Page** tool.



## Print One Record

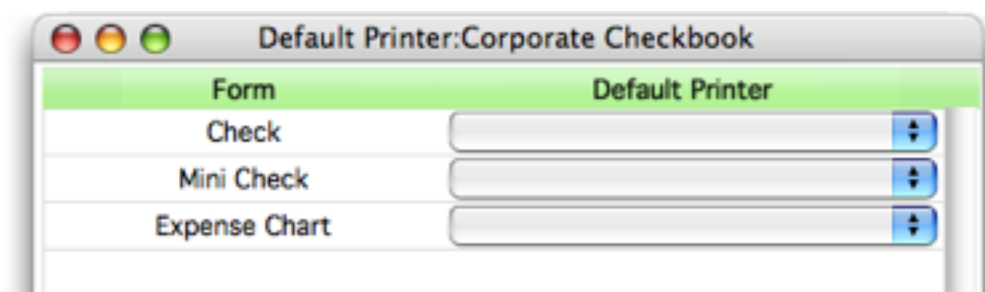
The **Print** command normally prints every visible record in a database. If you want to print just one record, use the **Print One Record** tool at the bottom of the tool palette.



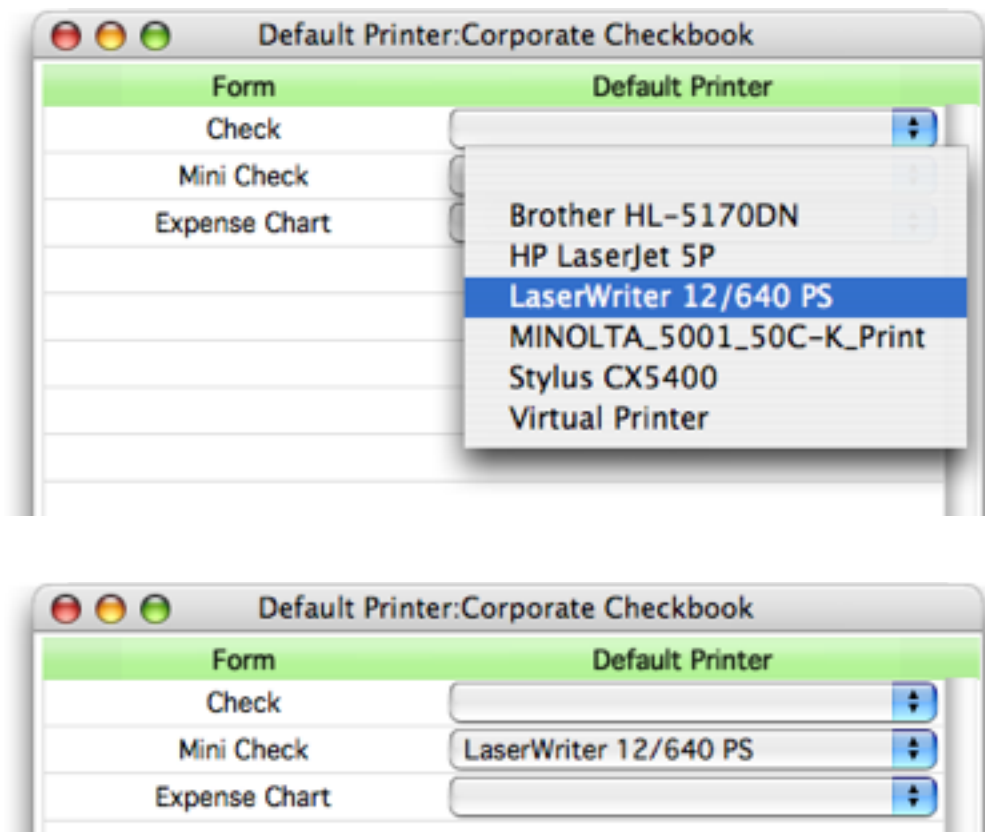
This tool only appears in the tool palette of form windows. It is not available for any other view (data sheet, design sheet, etc.).

## Setting up Default Printers

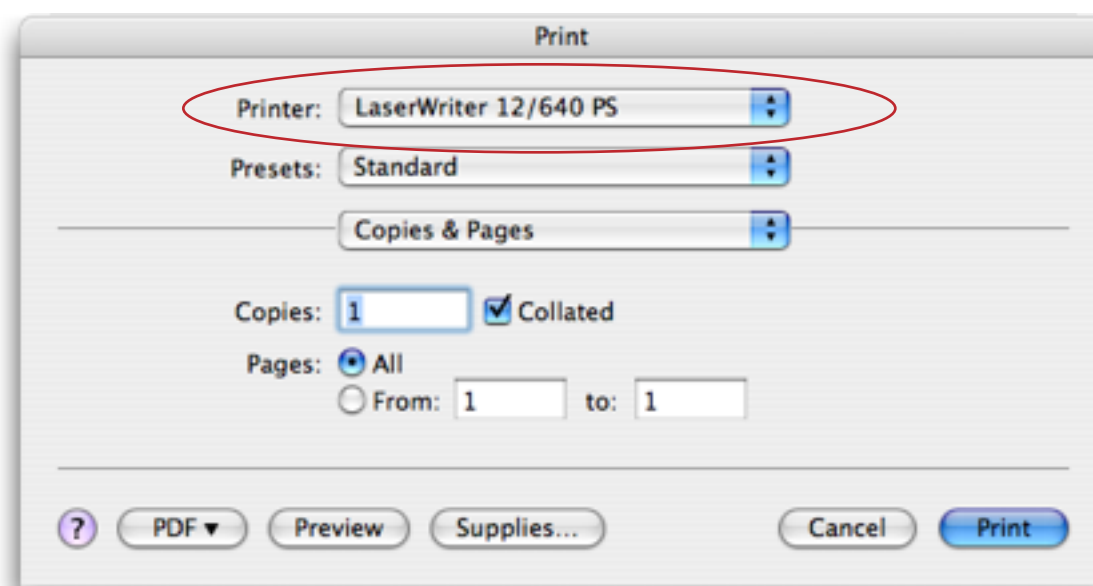
Panorama normally prints on whatever printer is currently selected in the operating system. If you are using Mac OS X you can also specify a default printer to be used with any form (this feature only works in the form view). When a default printer is set up Panorama will automatically switch to that printer when printing a form. For example you may want to print checks, envelopes or labels using a special printer. To configure this use the **Default Printer** wizard (in the **Preferences** submenu of the **Wizard** menu). Start by opening the database you want to set up, then open the wizard. The wizard will list all of the forms in the database.



Suppose you want to set up the Mini Check form so that it always defaults to printing on a specific printer, no matter what printer is currently selected. Simply click on the appropriate pop-up menu and choose the printer.



Whenever you print this form the printer will automatically be set to the printer you have specified. (Of course you can always use the pop-up menu in the Print dialog to change the printer at the last minute.)



To turn off the default printer selection simply choose the empty option from the pop-up menu.

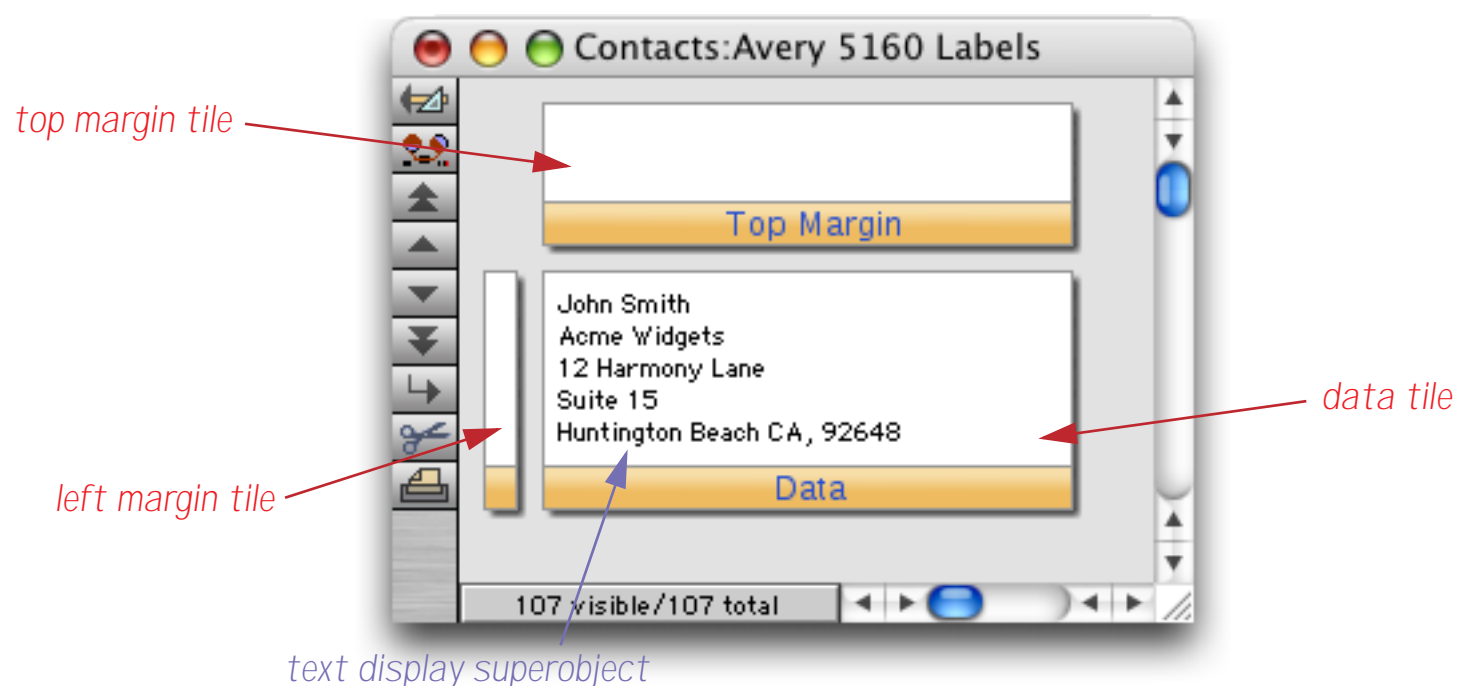
# Chapter 21: Custom Reports



Panorama has a very flexible system for printing custom reports. Panorama assembles each report page by taking pre-defined components from your form and sliding them into position on the page. Because the report components fit together like floor tiles, these components are called **report tiles**. Each tile in the form corresponds to a section of real estate on the printed page.

A form for generating a report may contain only a single report tile, or it may contain dozens of tiles. Panorama checks for the presence of each type of tile as it is building the report, and if found, uses the tile to build one section of the report. The size and shape of the tiles determines the overall layout of the report.

Report tiles are graphic objects that are part of a form. Here is an example of a typical form that contains three report tiles—top margin tile, left margin tile, and data tile. (This form also contains a Text Display SuperObject that has been placed on top of the data tile—more on that later.)



You can manipulate report tiles just as you would any other graphic object—drag, nudge, copy, etc.. Each form can be used to generate a unique report. Since a database can contain an unlimited number of different forms you can also have an unlimited number of custom reports. To print a particular custom report simply open the appropriate form and print. (You may also want to prepare the data before you print by sorting, selecting, and/or summarizing the data.)

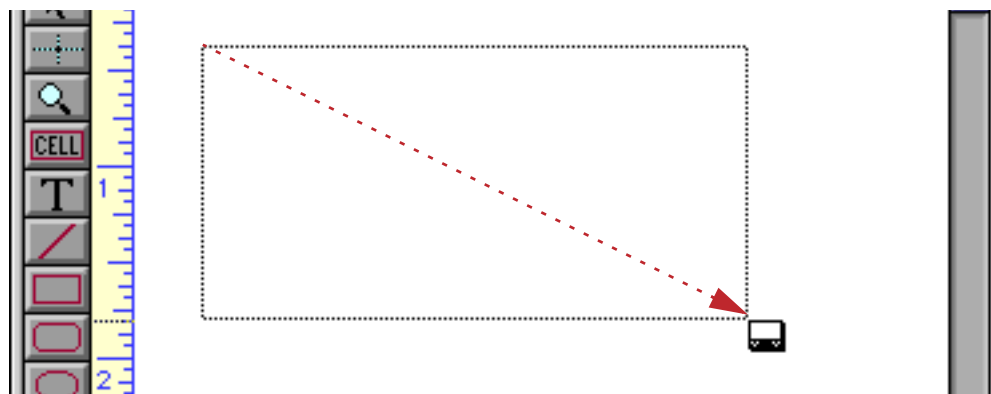
Note: This chapter only covers about half of Panorama's options for printing reports. To learn about more advanced features like variable height records, multi-page records, multi-column headers and footers, group sidebars, double sided page layout and more see Chapter 21 of the **Panorama Handbook**.

## Working with Tiles

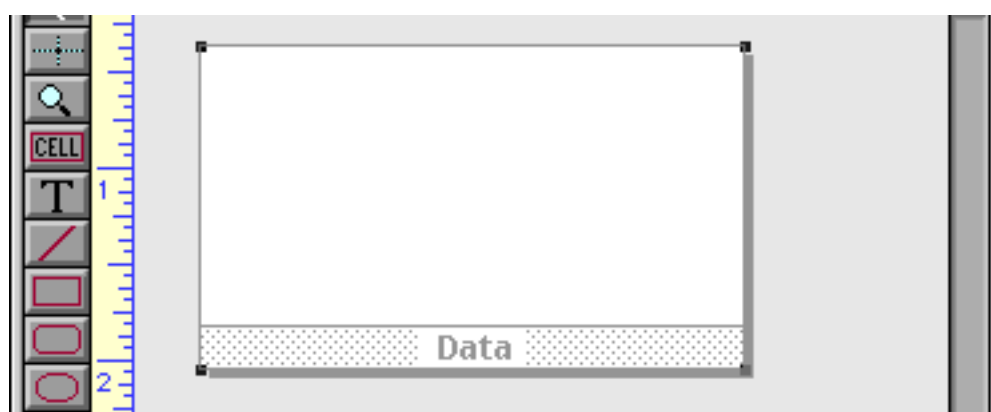
To create a new tile, start by selecting the **Tile** tool.



Next, drag the mouse across the form in the spot where you want to place the new tile. Any empty spot will do. The position of a tile on the form does not affect how it is printed—only the size and shape.

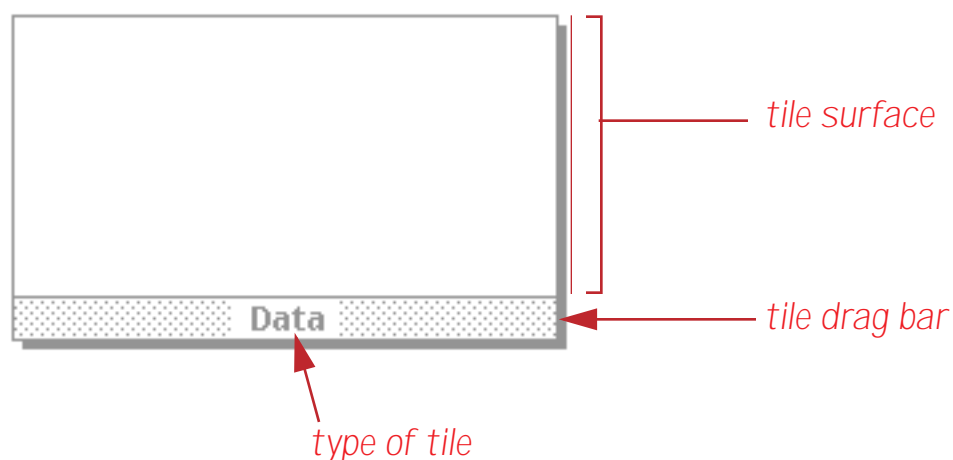


When you release the mouse, Panorama creates the new tile. If this is the first tile on the form, Panorama automatically creates a **data tile** (see “[Data Tiles](#)” on page 455). Later, Panorama will give you a choice.

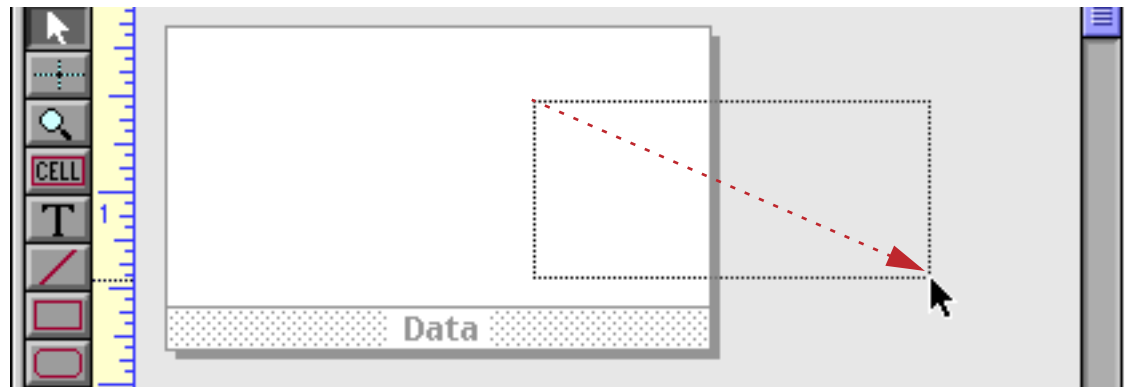


When you add the first tile to the form (the data tile), the background (outside the tile) will turn gray, as shown above. Only graphics or text that are on top of a tile will be printed — any graphics or text in the gray area will not be printed.

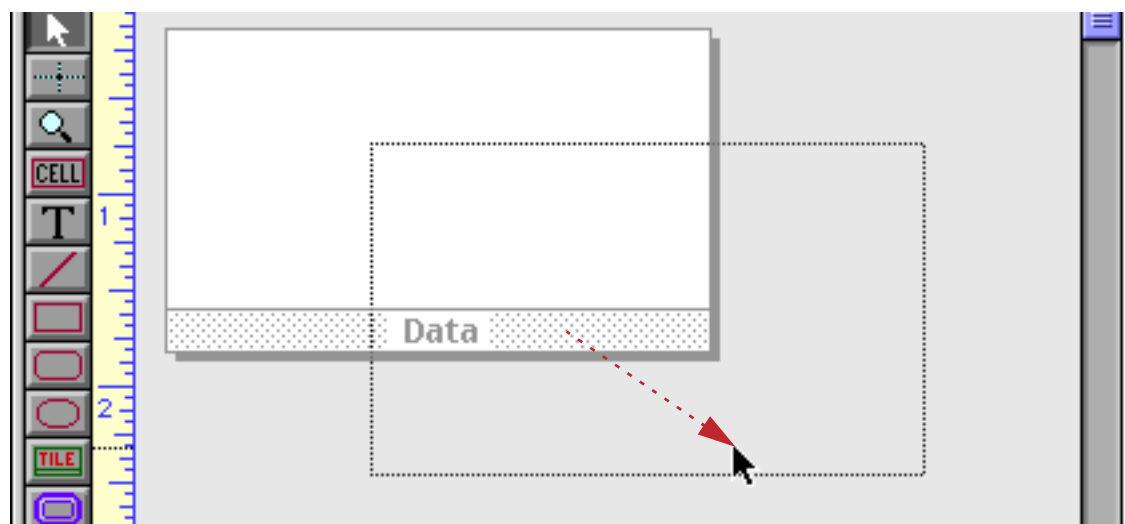
On the screen, a tile looks sort of like an upside down window. Tiles are divided into two parts: the surface and the drag bar. The surface is the actual printed area of the tile.



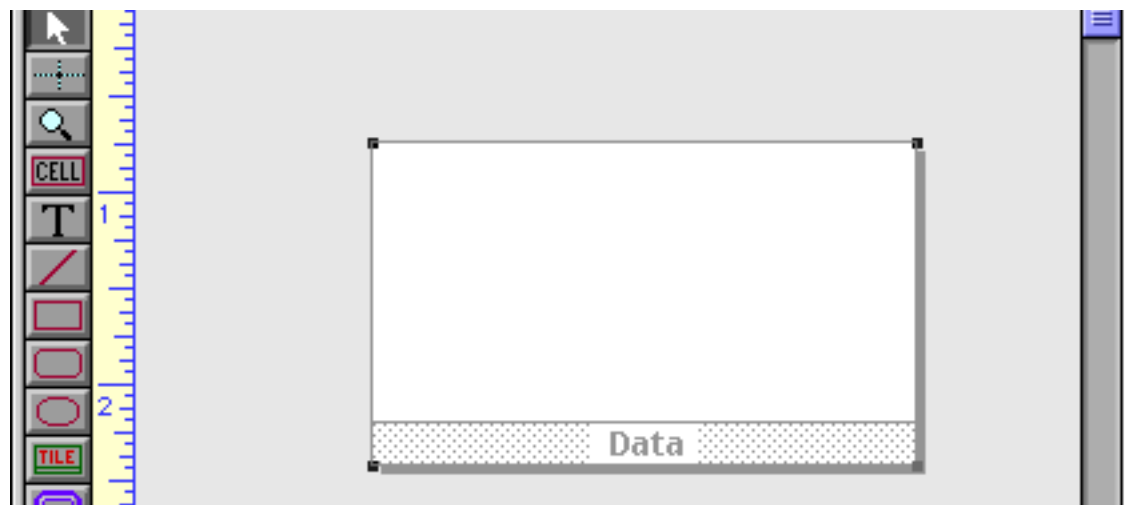
Unlike other graphic objects that can be manipulated by clicking anywhere in the object, a tile is only sensitive to clicks on its drag bar. If you click and/or drag on the surface of a tile, the object is not selected and does not move. Instead, a selection marquee appears, just as if you had dragged on an empty spot in the form.



To move a tile, press the mouse on the drag bar and drag the tile to the new position.

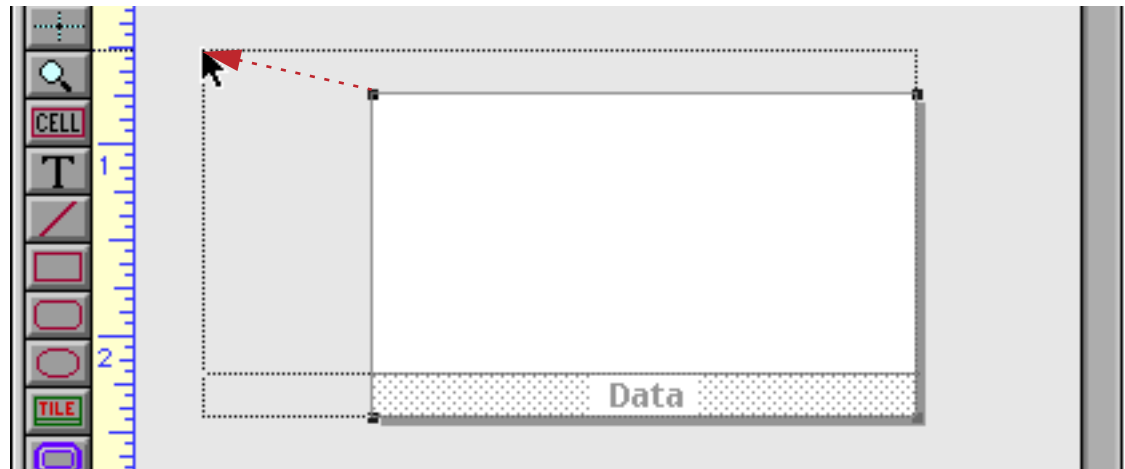


When you release the mouse, the tile moves to the new position, just as with dragging any other kind of object.

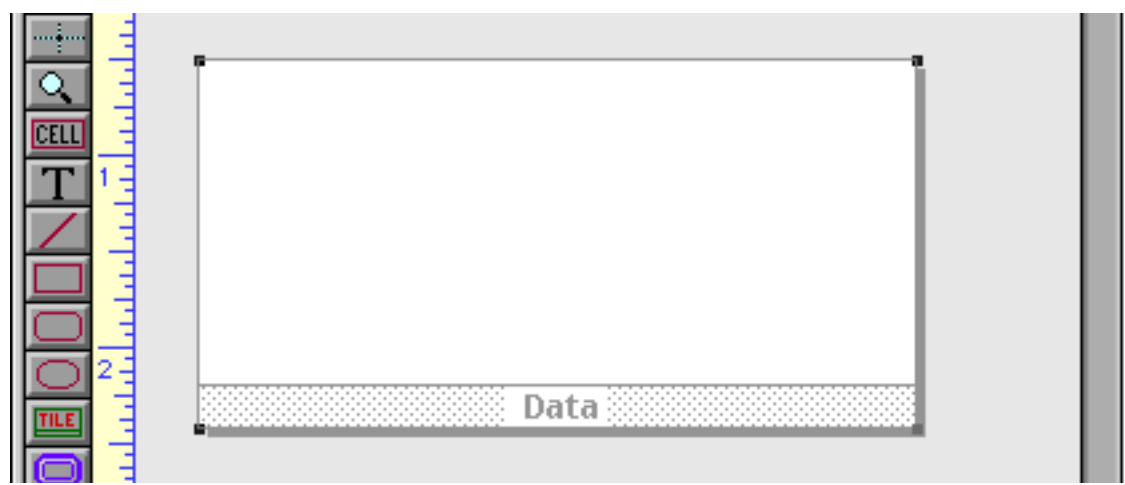




To select a tile, click on the drag bar. When the tile is selected, four handles appear around the corners of the tile, as shown above. You can use these grips to change the size of the tile, again, just like any other kind of object.

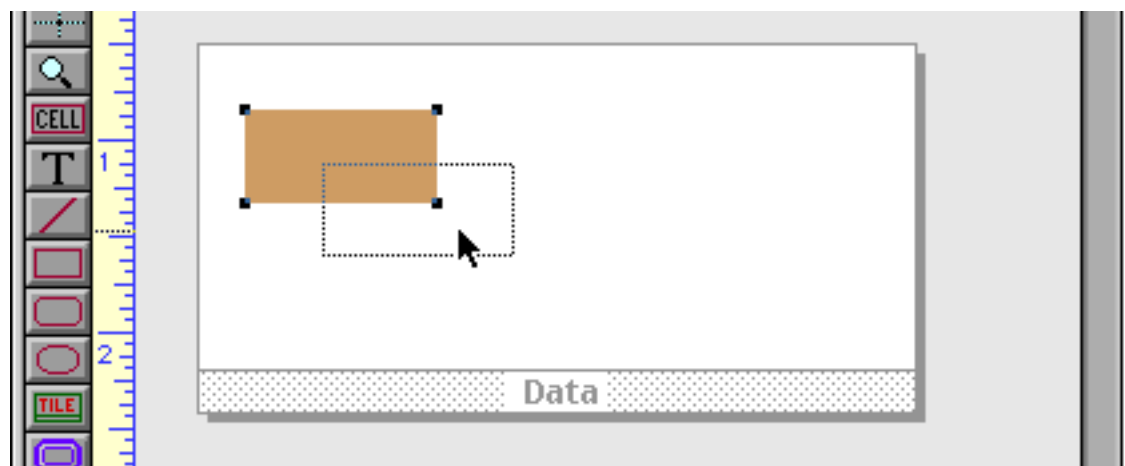


Release the mouse to see the new size.



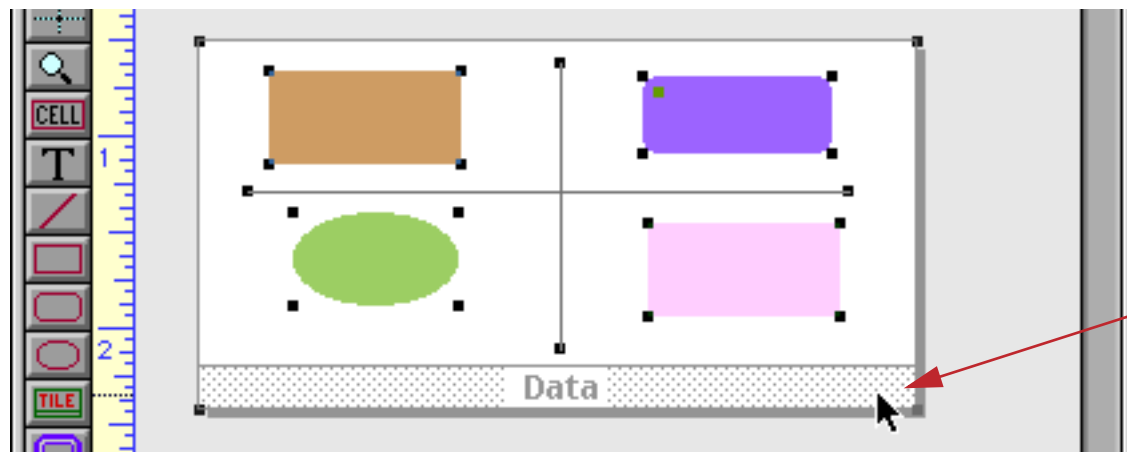
Tiles can also be moved or resized with the **Dimensions** command and by nudging with the arrow keys and. Note: When using the **Dimensions** command the dimensions are for the surface of the tile only, not including the drag bar along the bottom of the tile.

As mentioned above, the surface of the tile is not sensitive to the mouse. In other words, clicking on the surface area does not select the tile, and you cannot move the tile by dragging on the surface. However, you can place objects on top of the tile and move or select them.



In fact, as you will see later, most tiles must have other objects placed on top of them to build a complete report.

If a report tile has one or more objects placed on top of it, double clicking on the tile's drag bar will select both the tile and all of the objects on top of the tile. This is convenient if you want to move or copy the tile and the objects to a new position or to a different form.



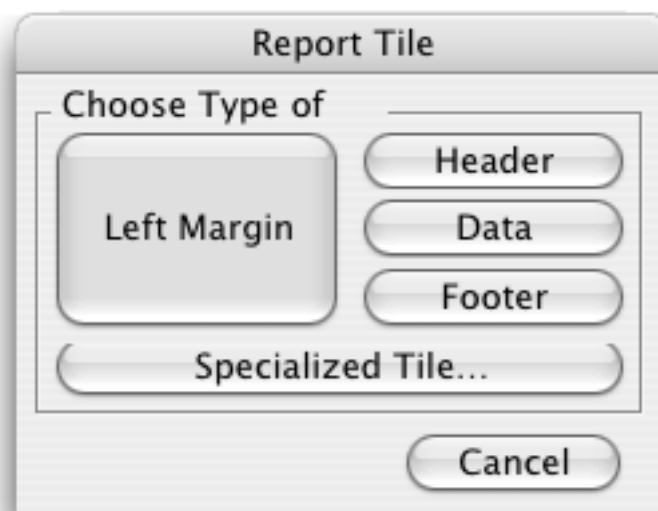
*double click drag bar to select tile and all objects on the tile*

However, double clicking on the type of tile does not select the objects.

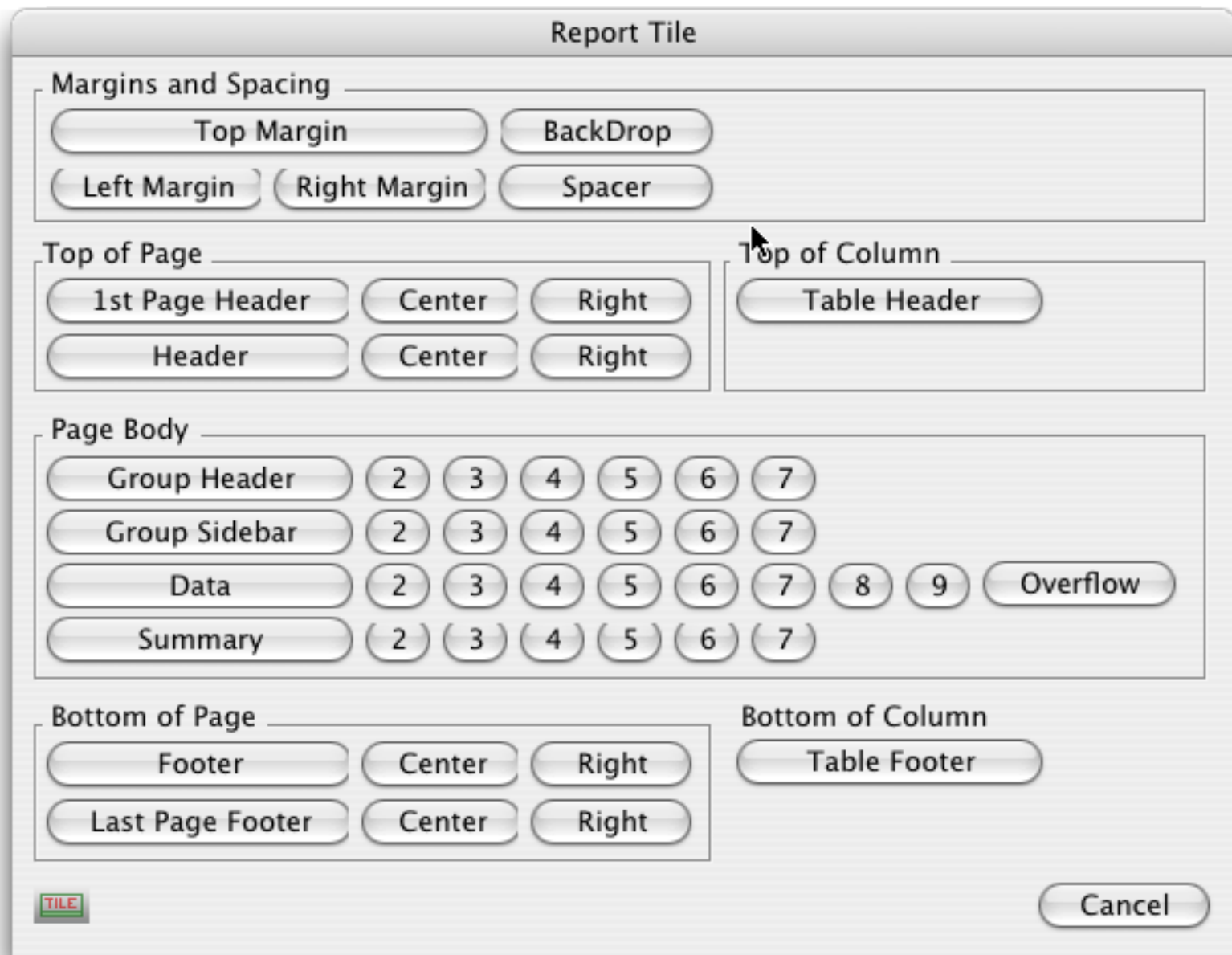


*double title to open tile configuration dialog*

Instead it opens the tile configuration dialog, allowing you to change the type of a tile. Depending on the number and type of tiles that are already on your form you may get the simple dialog shown here:



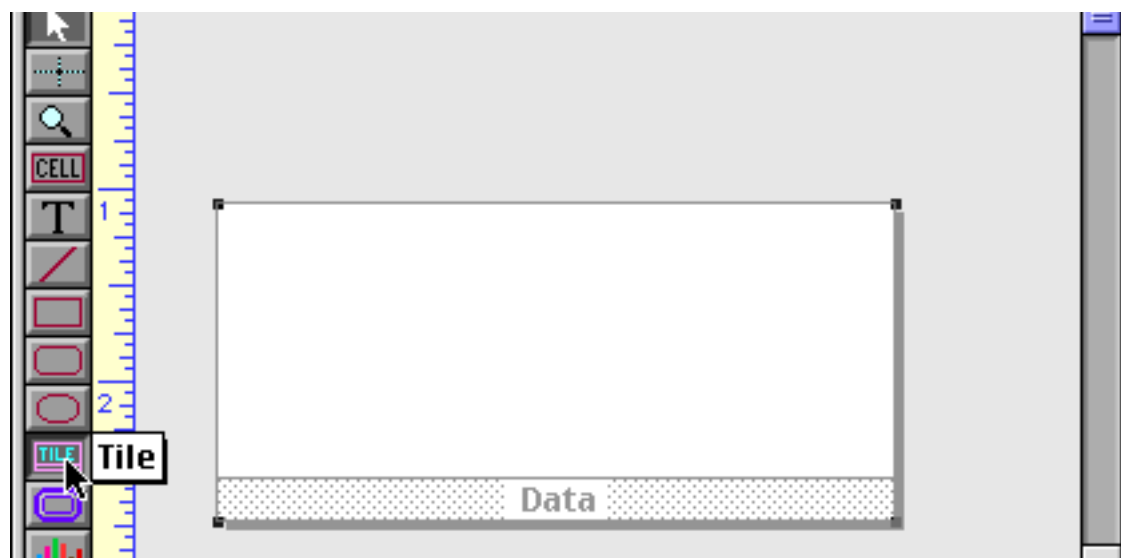
or the more complex dialog shown here (the *Specialized Tile* dialog).



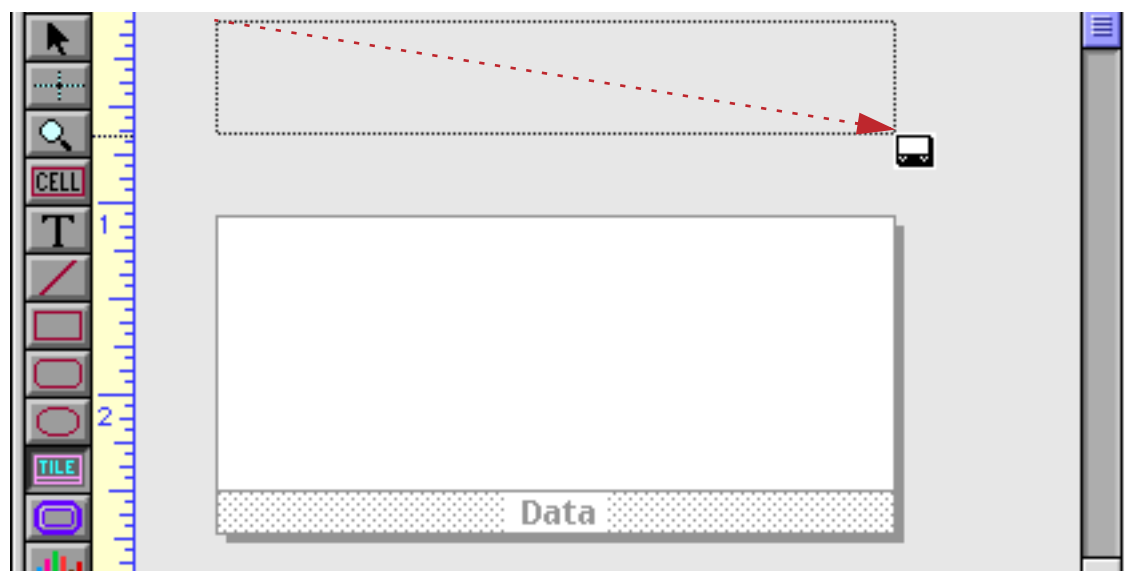
To change the tile's type simply pick the appropriate button, or press **Cancel** to keep the same type.

### Creating Additional Tiles

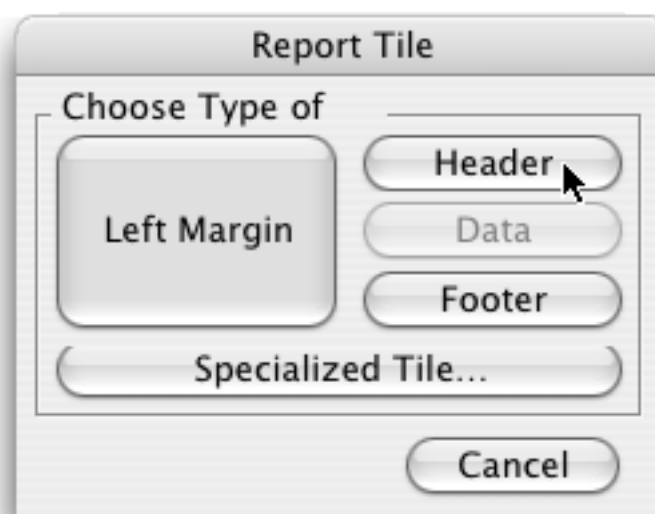
A minimum report contains at least a data tile. Many reports will contain additional tiles. To create an additional tile, start by selecting the Tile tool.



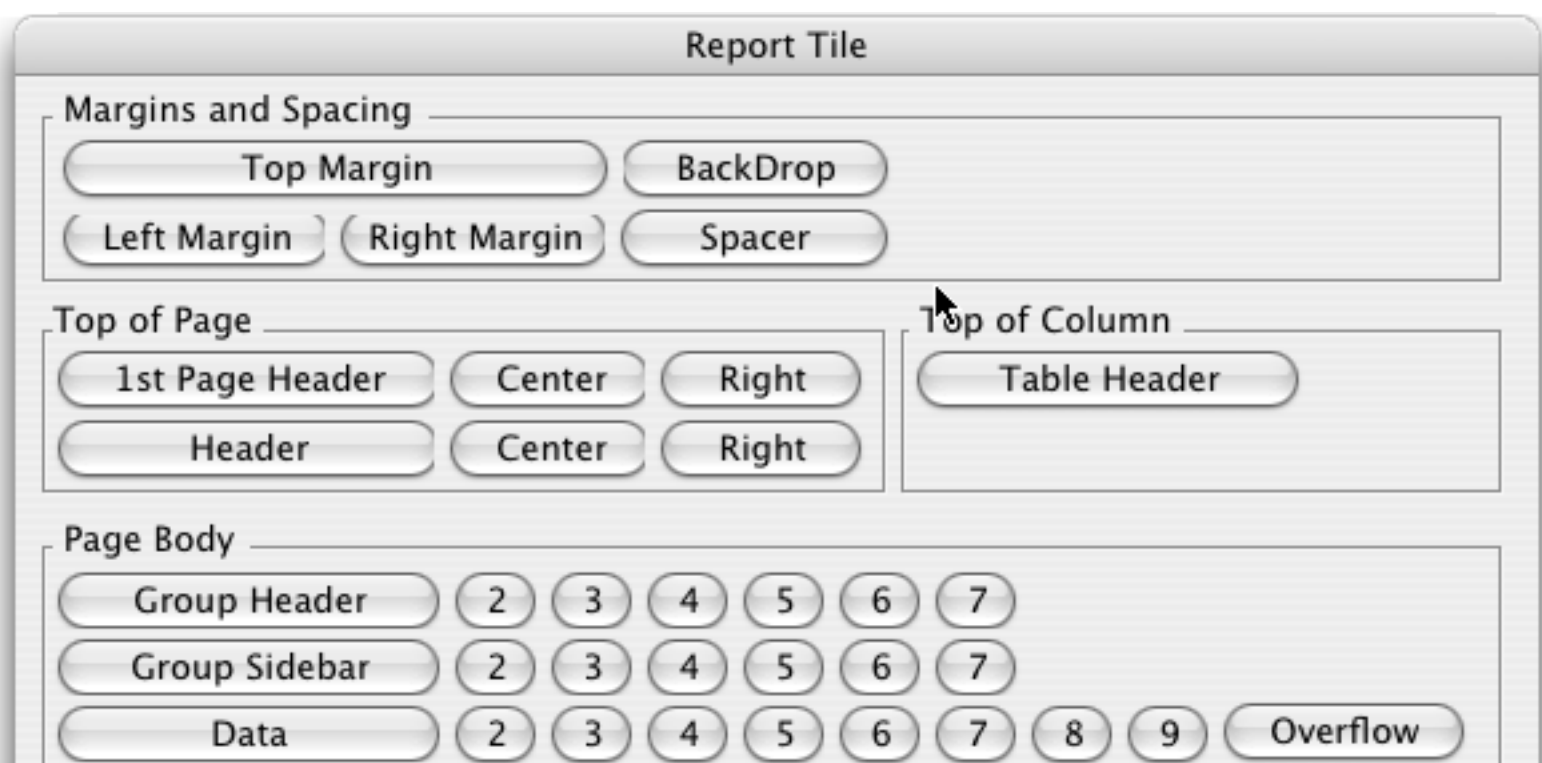
Next, drag the mouse across an empty (gray) spot on the form. Remember, the exact position of the tile is not important, only the size and shape.



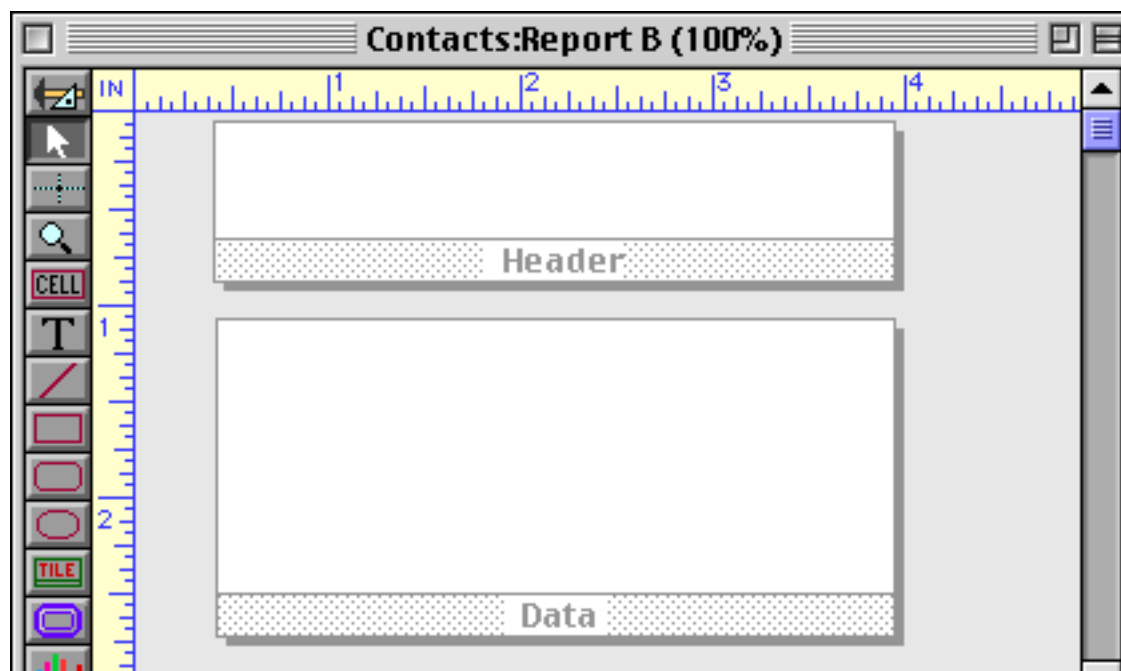
When you release the mouse the tile configuration dialog appears. Since this form currently has only one tile, the simple version of the configuration dialog will appear.



To create a **Header** tile, **Footer** tile, or **Left Margin** tile simply press the appropriate button. To create any other type of tile press the Specialized Tiles button, which makes the “long” version of the tile configuration dialog appear.

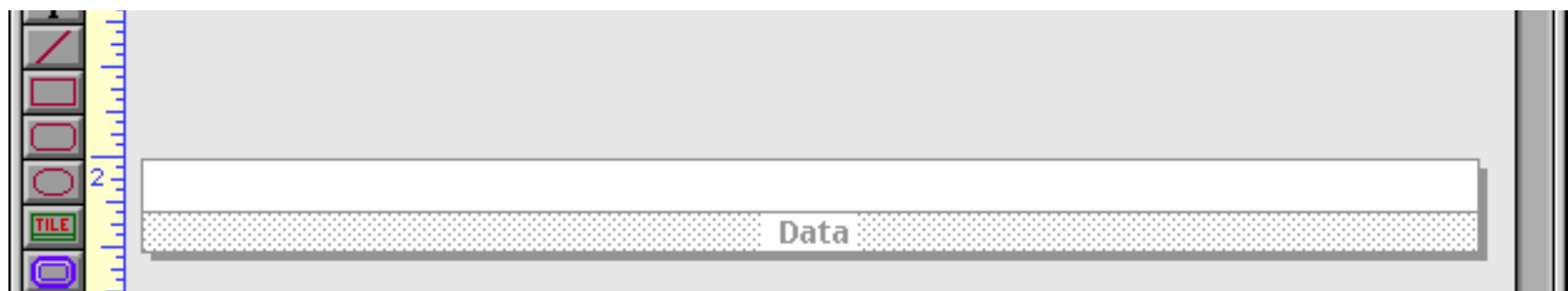


The multitude of choices in this dialog will be discussed later. For now we'll simply press the **Header** button to create a header tile.

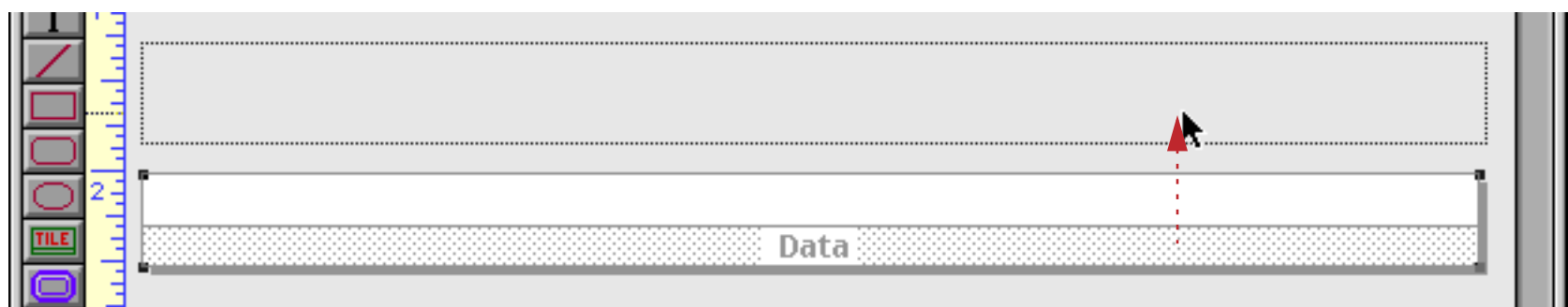


### Creating A New Tile By Duplicating

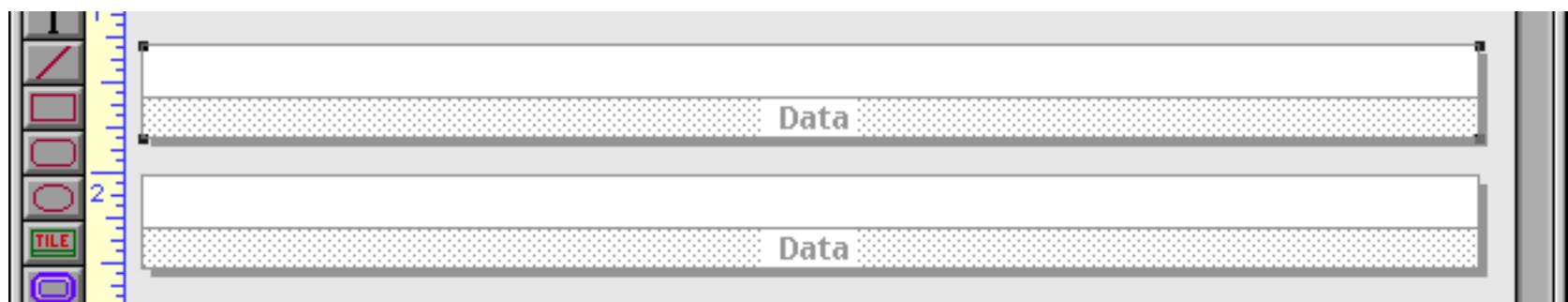
Another way to create a new tile is by duplicating an existing tile. For example, suppose you started with a data tile like this and wished to create a matching header tile.



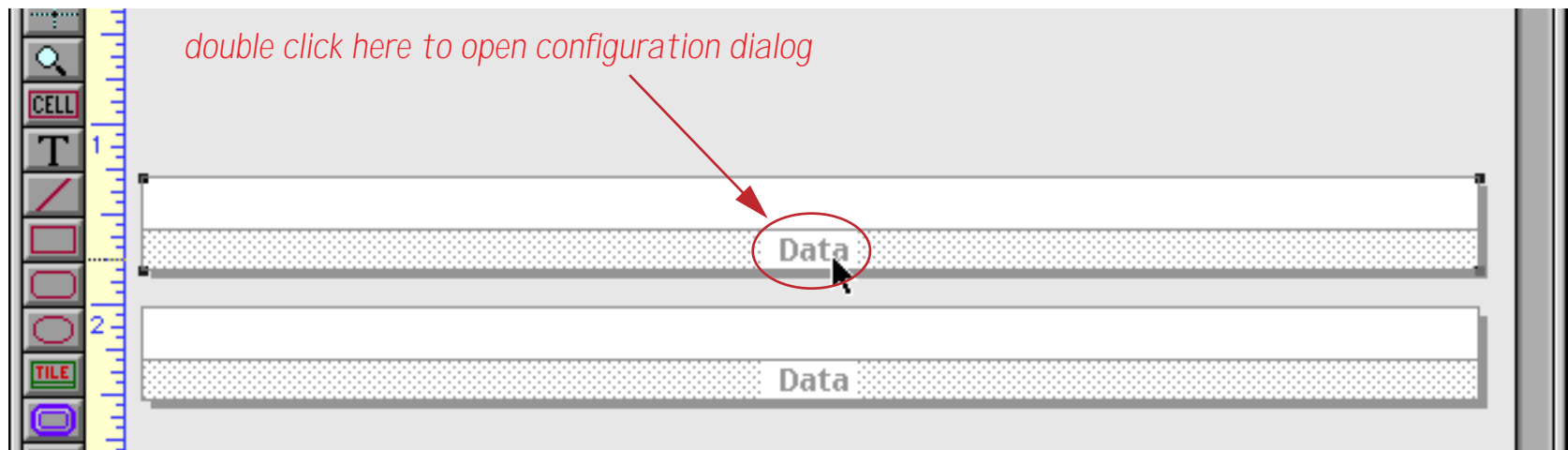
To duplicate this tile, hold down the **Option** key (Macintosh) or **Alt** key (PC), then click on the tile's drag bar and drag it. You may also want to hold down the **Shift** key to keep the new tile perfectly aligned with the original.



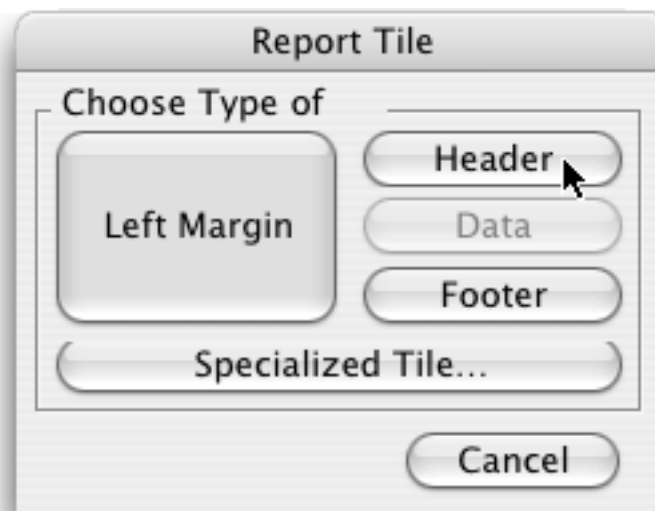
When you release the mouse a second data tile will appear.



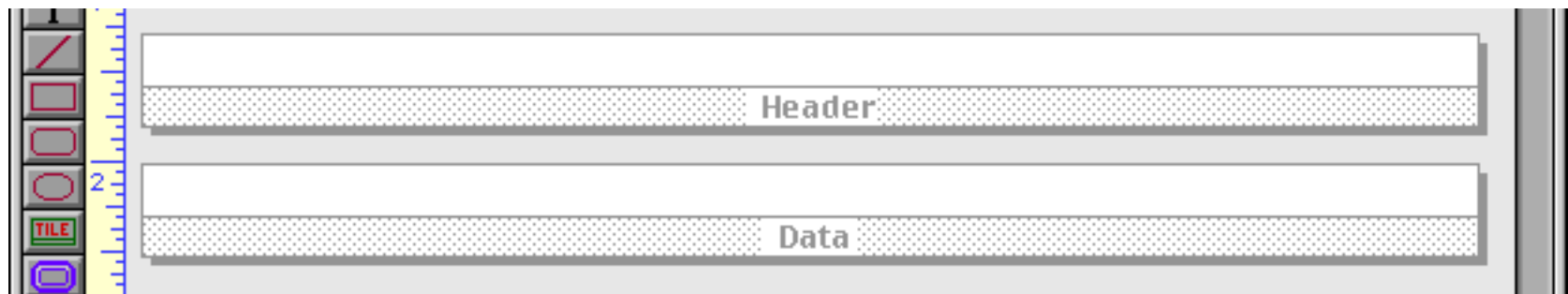
To change the type of the new tile, double click on the type of tile.



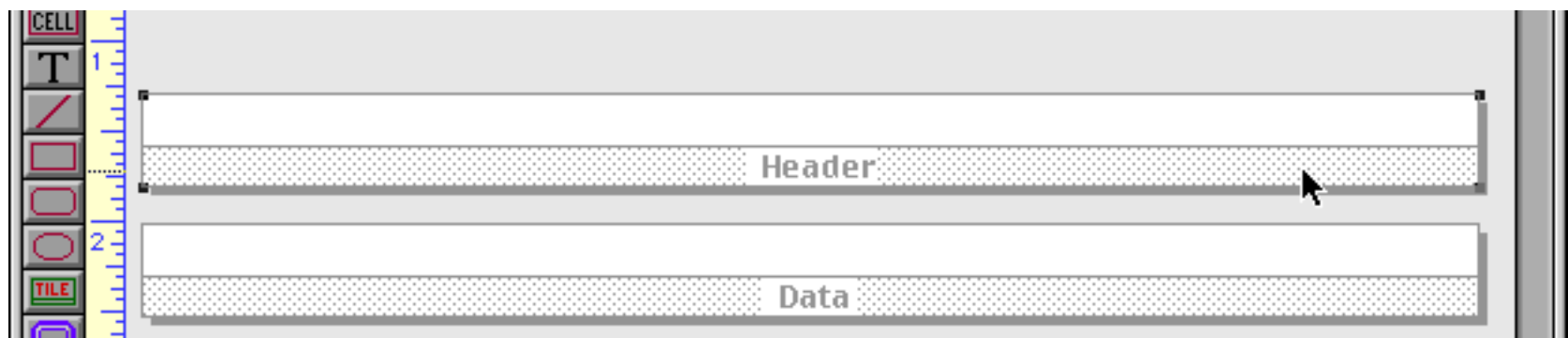
This opens the tile configuration dialog.



Press **Header** to switch the type of the new tile.

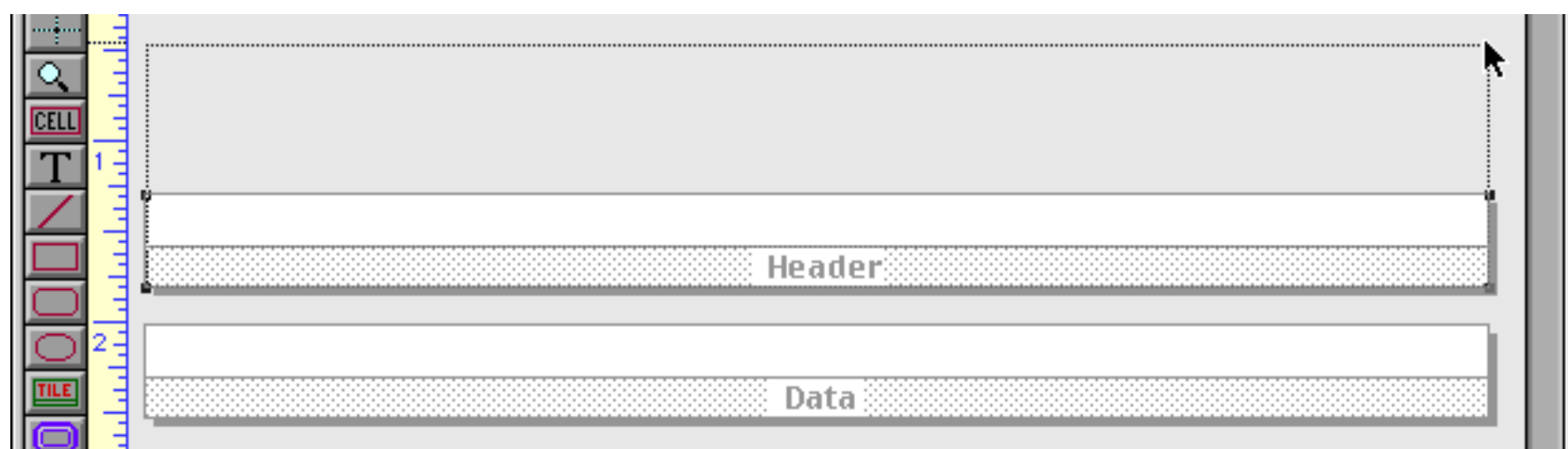


You may want to increase the height of the header tile. Here's one way to do that. Start by clicking on the drag bar to select the tile.

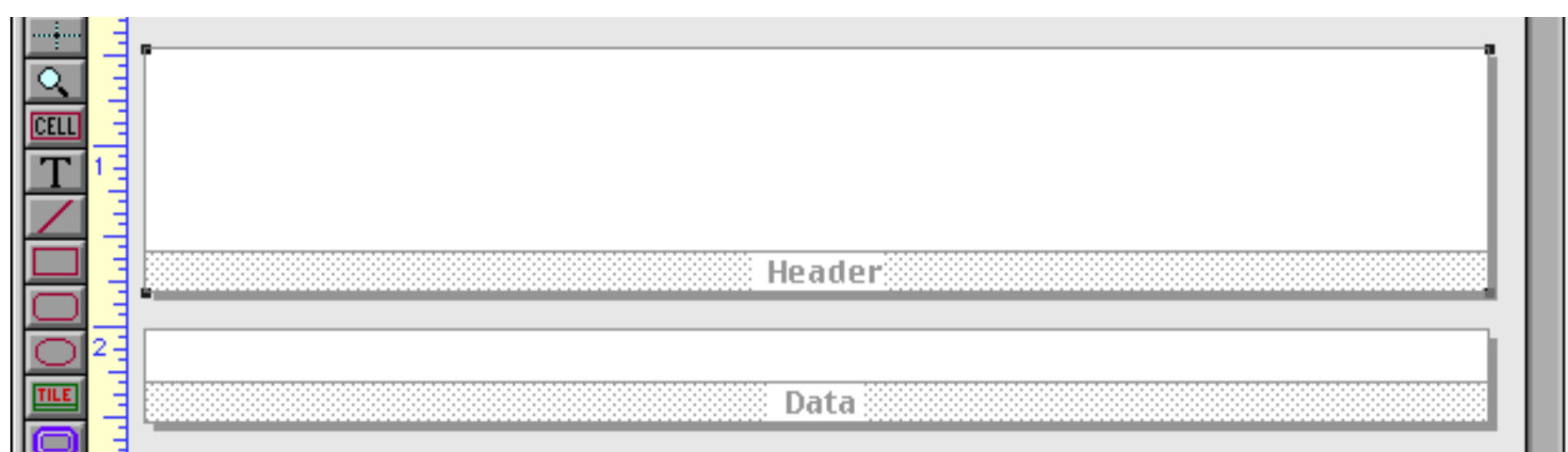




Now drag one of the handles to set the new size. Hold down the **Shift** key if you want to keep the width constant.



When you release the mouse button the tile snaps to the new height.



Of course the tile can also be moved or resized with the **Dimensions** command and by nudging with the arrow keys.

## Tiles In Action

Panorama has many different types of tiles. Each type of tile has its own rules that tell Panorama where the tile should be printed on the page. For example, a header tile is always printed at the top of the page. You can make the header bigger or smaller, add fancy graphics, even move it to a different spot in the form window—but no matter what you do, the header will always print at the top of the page.

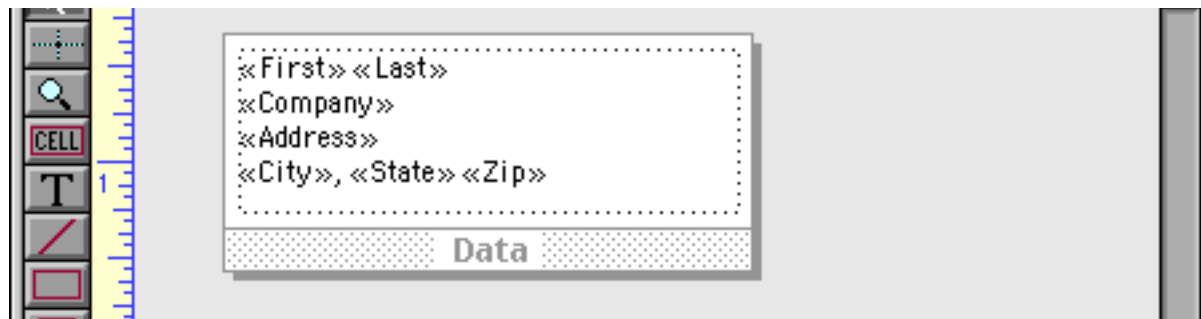
Panorama's basic rules for printing tiles are simple. First it prints the header (if any) at the top of the page, then the footer at the bottom of the page. The space left in the middle is available for data. Panorama starts in the upper left hand corner and prints a column of data—one data tile at a time. Once Panorama reaches the bottom, it checks to see if it should print another column. If another column is needed, Panorama goes back to the top and prints it.

More complex reports can have variations on this basic theme. Panorama has many different specialized types of tiles for automatically creating almost any report format. But the basic idea is the same—each tile slides into place on the page according to its rules.

A form should contain no more than one tile of each type. If a form contains two header tiles, for example, Panorama will not know which one to print at the top of the page. If you attempt to print using a form that has duplicate tiles, Panorama will display an alert.

## Data Tiles

The data tile is the cornerstone of every custom report. Because the data tile is used to print each data record, the size and shape of the data tile has a major impact on the overall look of the report. For instance, if you want to print labels, the data tile should be the same size and shape as a single label plus the gaps between the labels, like this.



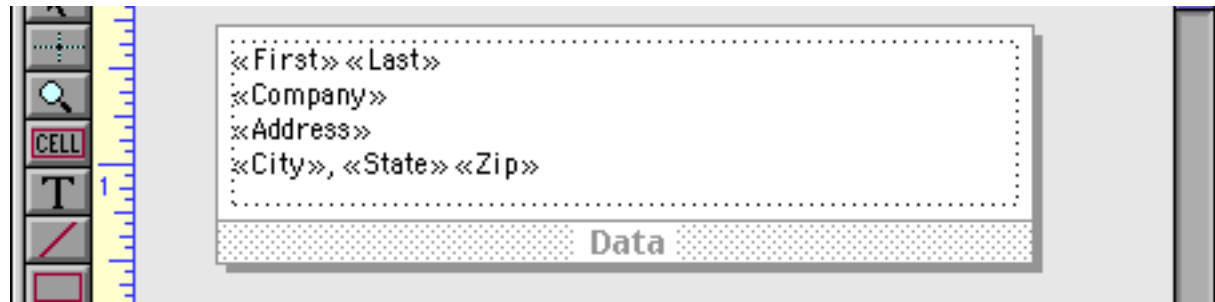
When you print this form it will look like this.

John Smith Acme Widgets 12 Harmony Lane Suite 15	Don Harmon Sudderth Video 415 Sudderth Ruidoso, NM 88345	David Blair DB Printing 869 W. Temple Lenox, IA 50851
Susan Brown 783 Algonquin Newport Beach, CA 93459	Abe Fierstein Van Nuys Lumber 1571 Haskell Van Nuys, CA 91409	Keith Baker Northgate Video 552 Northgate Lindenhurst, IL 60046
Karen Wilson Evanston Lumber 498 Noyes Evanston, IL 60201	Randy Cross Randy's Appliances 133 Hunt Rd Chelsford, MA 01824	John Sloan 79 Danube Way Olympia Fields, IL 60461
Jim Nickle Jim's Appliances 14189 8th Newhall, CA 91321	Jeffrey Rodman 2 Cary Rd Chestnut Hill, MA 02167	Guy Porter St. Louis Lumber 8702 Pershing St. Louis, MO 63107

This illustration shows how Panorama stacked the tiles together to create the report shown above. In this illustration, the tiles, which are actually invisible, are shown in light blue. As you can see, Panorama slides the tile surfaces together as closely as possible. (The tile drag bars are not counted as part of the tile when the report is printed.)

John Smith Acme Widgets 12 Harmony Lane Suite 15	Don Harmon Sudderth Video 415 Sudderth Ruidoso, NM 88345	David Blair DB Printing 869 W. Temple Lenox, IA 50851
Data	Data	Data
Susan Brown 783 Algonquin Newport Beach, CA 93459	Abe Fierstein Van Nuys Lumber 1571 Haskell Van Nuys, CA 91409	Keith Baker Northgate Video 552 Northgate Lindenhurst, IL 60046
Data	Data	Data
Karen Wilson Evanston Lumber 498 Noyes Evanston, IL 60201	Randy Cross Randy's Appliances 133 Hunt Rd Chelsford, MA 01824	John Sloan 79 Danube Way Olympia Fields, IL 60461
Data	Data	Data
Jim Nickle	Jeffrey Rodman	Guy Porter

Simply changing the size or shape of the data tile can completely change the printed report.



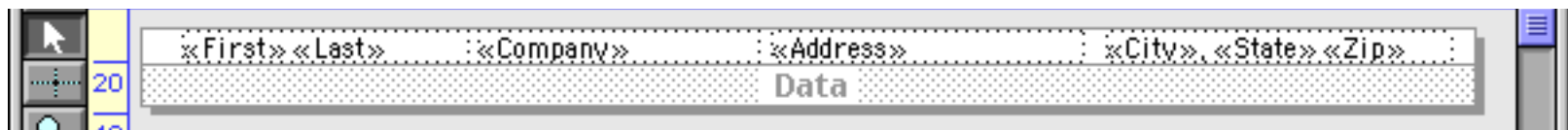
The only change is that the data tile (and the text object on it) has been made wider. Now Panorama can only stack two data tiles side by side.

John Smith Acme Widgets 12 Harmony Lane Suite 15	Don Harmon Sudderth Video 415 Sudderth Ruidoso, NM 88345
Susan Brown 783 Algonquin Newport Beach, CA 93459	Abe Fierstein Van Nuys Lumber 1571 Haskell Van Nuys, CA 91409
Karen Wilson Evanston Lumber 498 Noyes Evanston, IL 60201	Randy Cross Randy's Appliances 133 Hunt Rd Chelsford, MA 01824
Jim Nickle Jim's Appliances 14189 8th Newhall, CA 91321	Jeffrey Rodman 2 Cary Rd Chestnut Hill, MA 02167

Once again here is an illustration that shows how Panorama fits the invisible tiles together to make the finished report.

John Smith Acme Widgets 12 Harmony Lane Suite 15	Don Harmon Sudderth Video 415 Sudderth Ruidoso, NM 88345
Susan Brown 783 Algonquin Newport Beach, CA 93459	Abe Fierstein Van Nuys Lumber 1571 Haskell Van Nuys, CA 91409
Karen Wilson Evanston Lumber 498 Noyes Evanston, IL 60201	Randy Cross Randy's Appliances 133 Hunt Rd Chelsford, MA 01824
Jim Nickle Jim's Appliances 14189 8th Newhall, CA 91321	Jeffrey Rodman 2 Cary Rd Chestnut Hill, MA 02167

To make a columnar report with one line per record, make the data tile as wide as the page and one line high, like this.



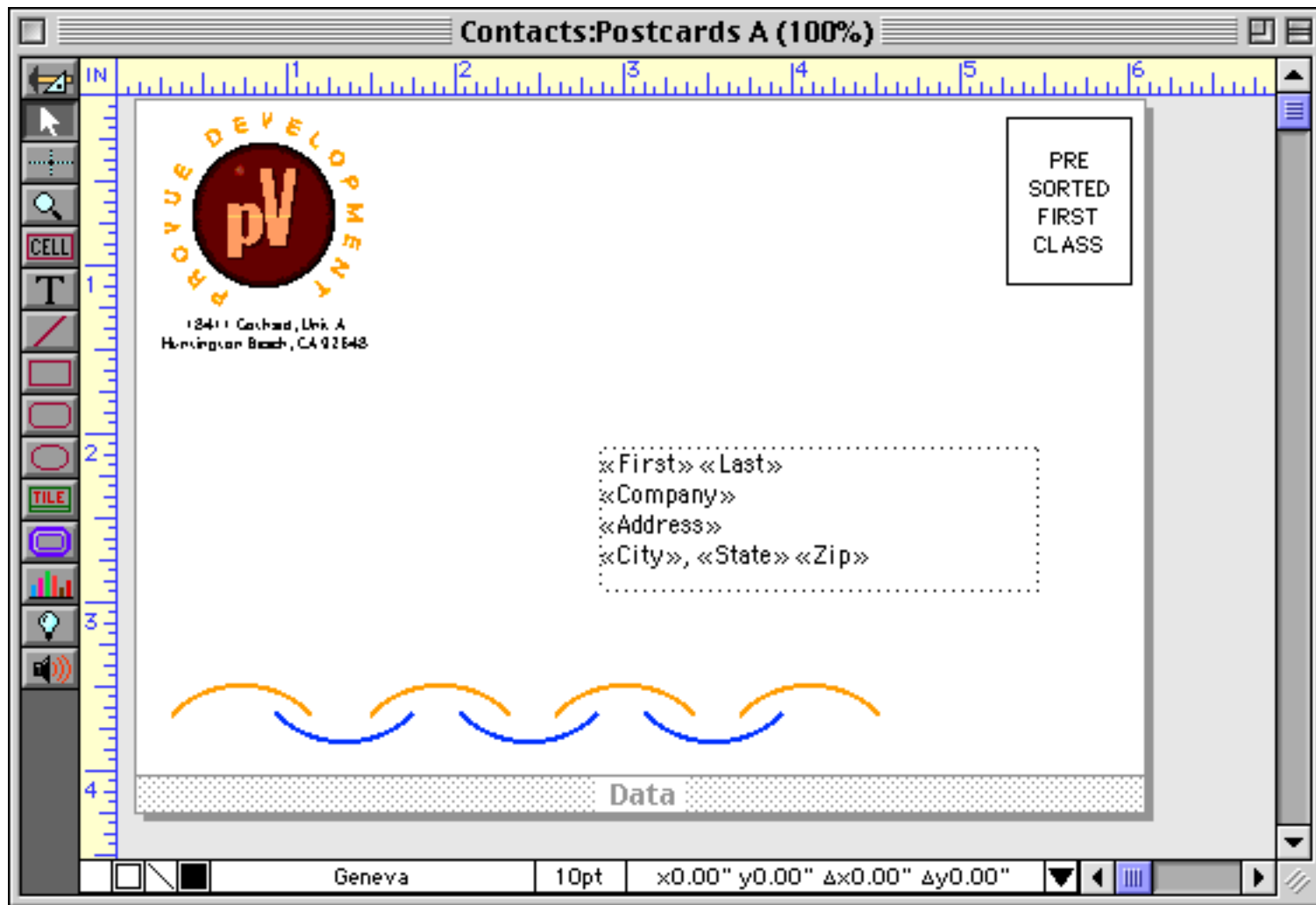
Here's the finished report printed from this tile.

John Smith	Acme Widgets	12 Harmony Lane	Huntington Beach, CA
Susan Brown		783 Algonquin	Newport Beach, CA 93459
Karen Wilson	Evanston Lumber	498 Noyes	Evanston, IL 60201
Jim Nickle	Jim's Appliances	14189 8th	Newhall, CA 91321
Brian Felty	B.F. Plumbing	118 N Wilder	Lubbock, TX 79410
Bob Hanlan	Ann Arbor Lumber	6916 Morgan	Ann Arbor, MI 48104
Tim Daniels	St. Louis Lumber	3133 Cornell	St. Louis, MO 63130
John Moses		8265 Leticia	San Clemente, CA 92672
John Fabian		3 Rose Hill	Woodstock, VT 05091
Ed Ruth	Chicago Lumber	1580 N. Oconto	Chicago, IL 60634
Don Harmon	Sudderth Video	415 Sudderth	Ruidoso, NM 88345

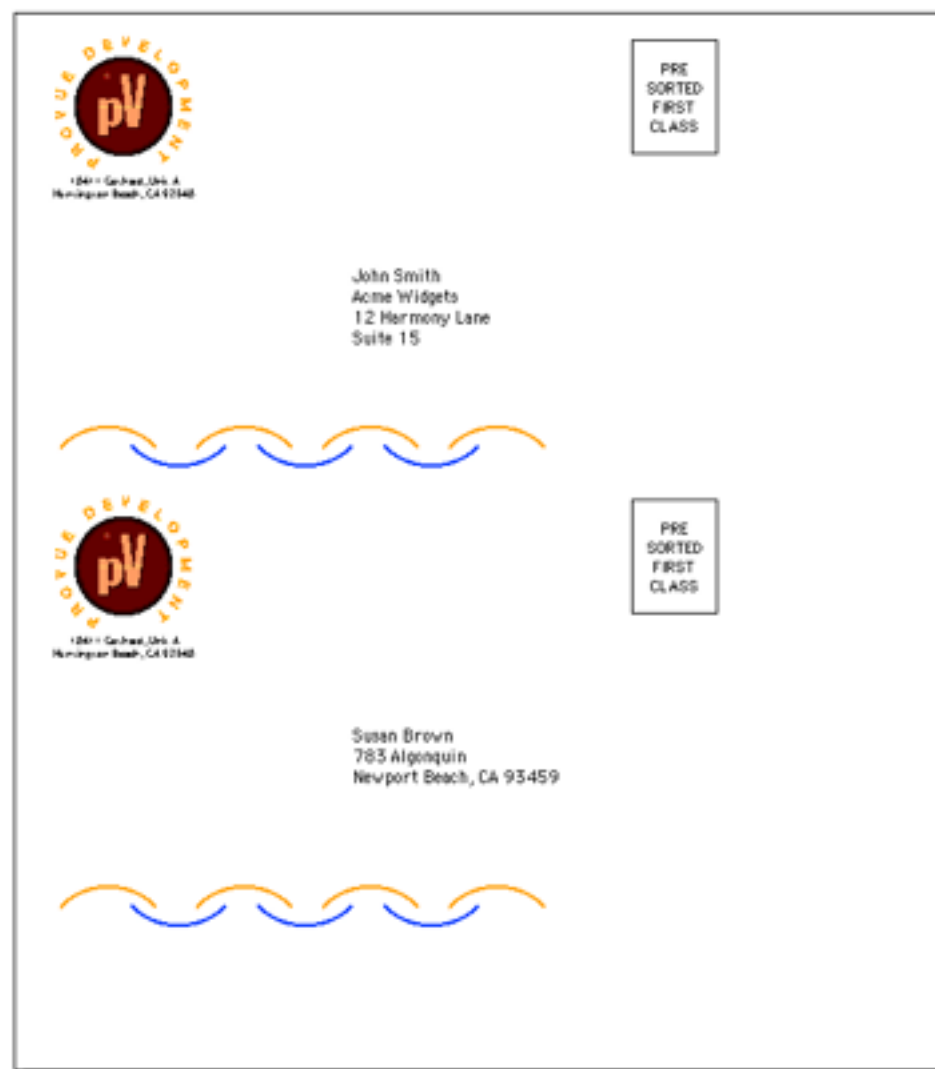
This data tile is too wide to stack side-by-side, so Panorama simply stacks them vertically.

John Smith	Acme Widgets	12 Harmony Lane	Huntington Beach, CA
Susan Brown		783 Algonquin	Newport Beach, CA 93459
Karen Wilson	Evanston Lumber	498 Noyes	Evanston, IL 60201
Jim Nickle	Jim's Appliances	14189 8th	Newhall, CA 91321
Brian Felty	B.F. Plumbing	118 N Wilder	Lubbock, TX 79410
Bob Hanlan	Ann Arbor Lumber	6916 Morgan	Ann Arbor, MI 48104
Tim Daniels	St. Louis Lumber	3133 Cornell	St. Louis, MO 63130
John Moses		8265 Leticia	San Clemente, CA 92672
John Fabian		3 Rose Hill	Woodstock, VT 05091
Ed Ruth	Chicago Lumber	1580 N. Oconto	Chicago, IL 60634
Don Harmon	Sudderth Video	415 Sudderth	Ruidoso, NM 88345

At the other extreme you can make the data tile very large so that only one or two records will be printed per page. This form is designed to print two postcards per page.



Here's what the postcards look like when printed on an 8.5 x 11 inch page.



Just as with the other reports, Panorama prints this one by stacking the data tiles as close together as possible.

PROVUE DEVELOPMENT  
pV  
12411 Cochran, Unit A  
Huntington Beach, CA 92648

PRE SORTED  
FIRST  
CLASS

John Smith  
Acme Widgets  
12 Harmony Lane  
Suite 15

Data

PROVUE DEVELOPMENT  
pV  
12411 Cochran, Unit A  
Huntington Beach, CA 92648

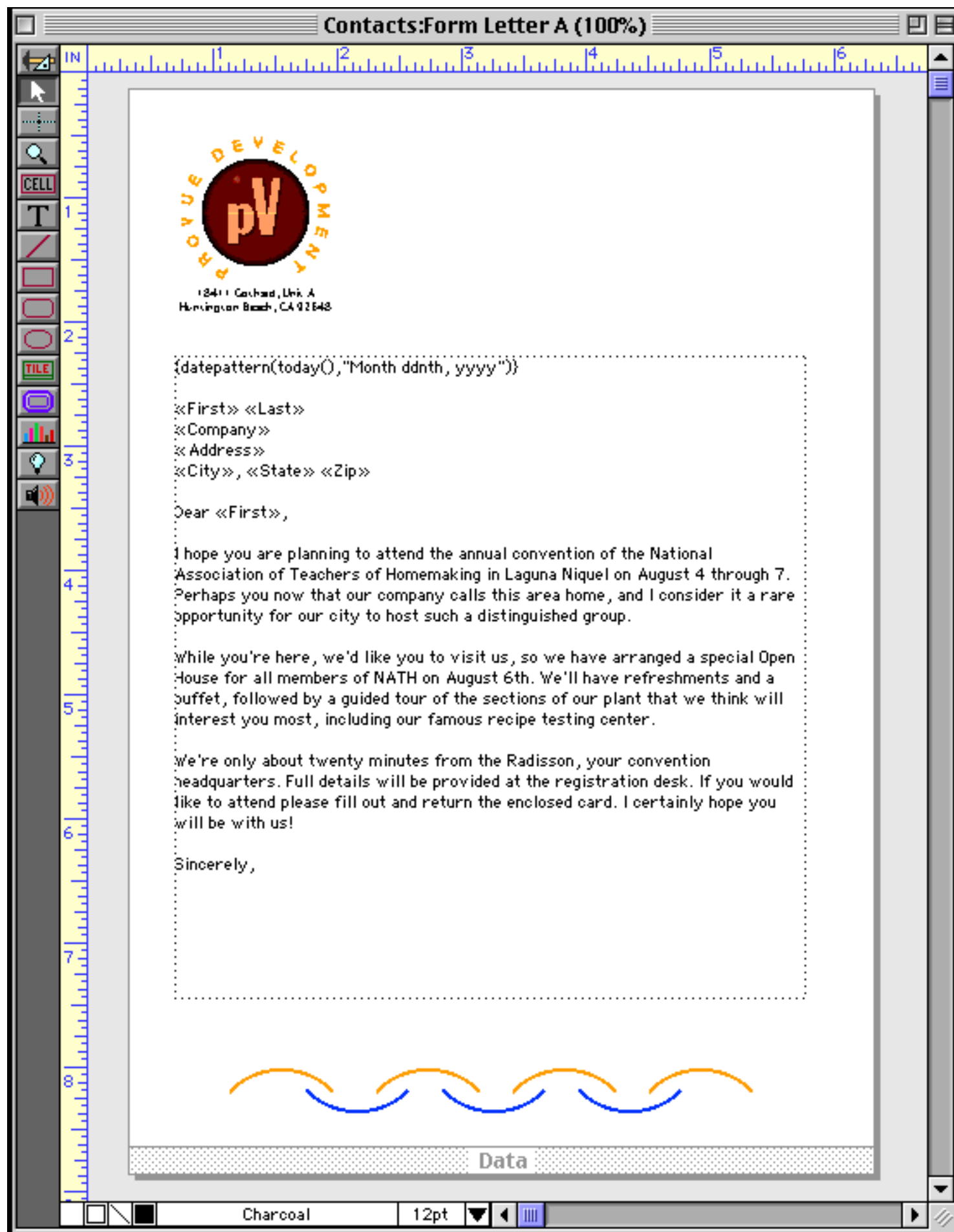
PRE SORTED  
FIRST  
CLASS

Susan Brown  
783 Algonquin  
Newport Beach, CA 93459

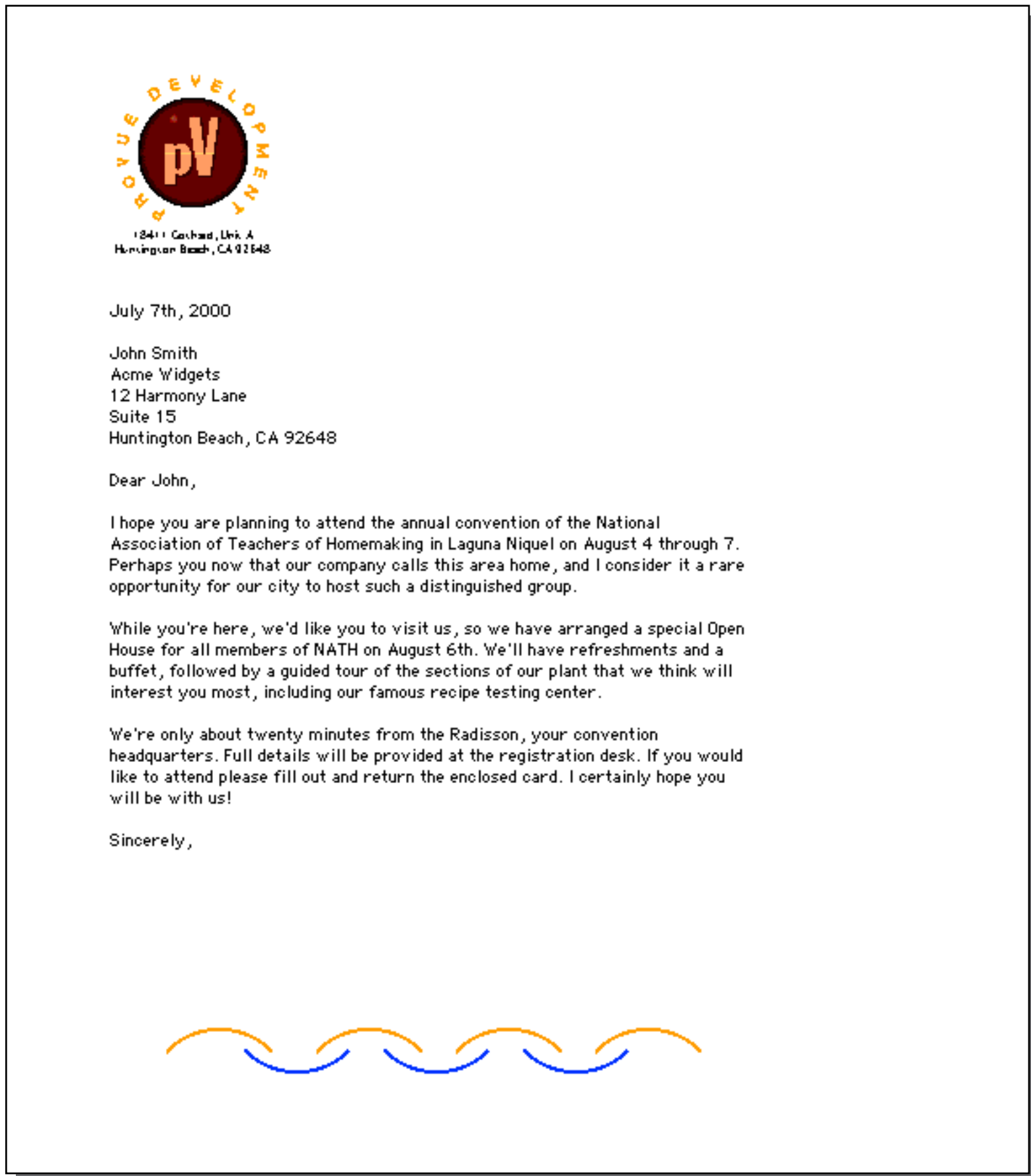
Data



If the data tile is large enough Panorama will only print one record per page. Common applications include invoices, statements, tax returns, and form letters like the one shown here.



Here's the printed result of this form. Panorama will print one complete page for each selected record in the database, each with a customized letter. (Or, if you wish, you can use the **Print One Record** tool to print a single letter. See "[Print One Record](#)" on page 1036)



By the way, it's also possible to print multi-page letters. See Chapter 21 of the **Panorama Handbook**.

## Margins

If a form doesn't contain any margin tiles (like the examples in the previous section) Panorama will automatically start printing as far up and to the left as possible on the current printer. In this case the exact margins will depend on the printer and on the Page Setup options you have chosen.

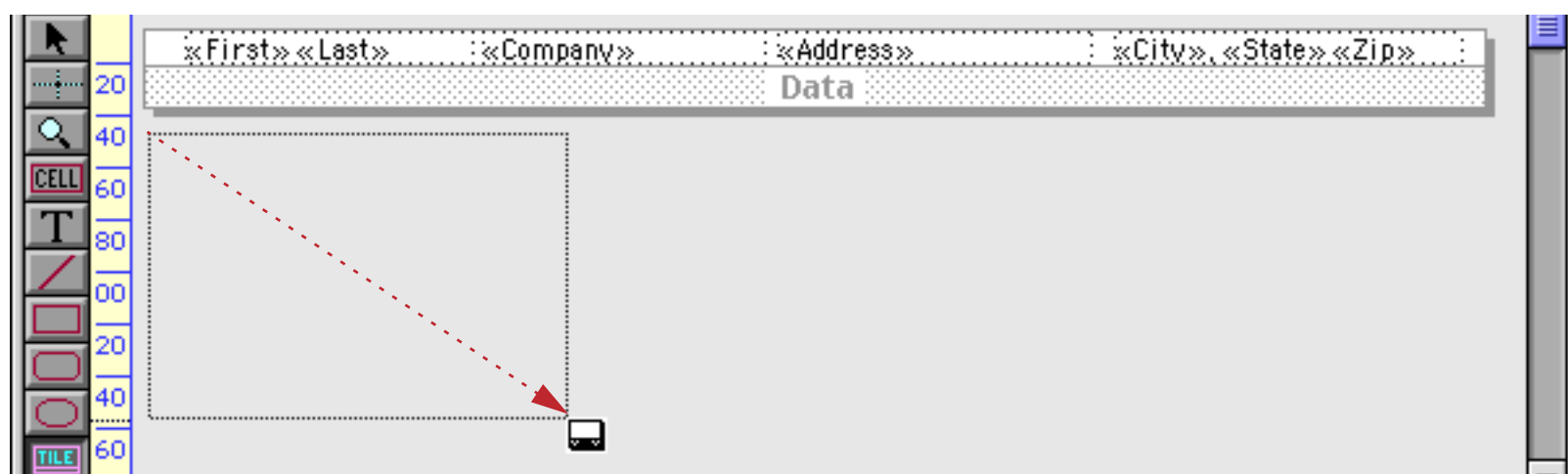
### Top Margin Tile

The top margin tile allows you to modify the distance from the top of the page to the top of the first tile printed. Unlike most other tiles where both dimensions are important, Panorama only cares about the height of the top margin tile. If the report contains a top margin tile, the height of that tile becomes the top margin of the report.

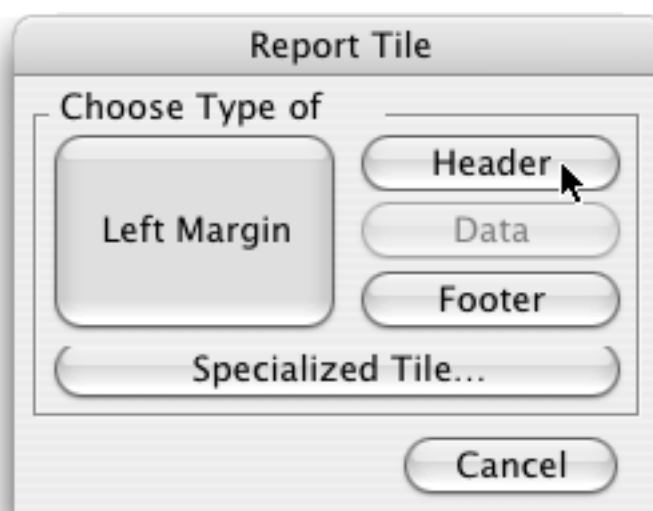
To create a top margin, start by selecting the **Tile** tool.



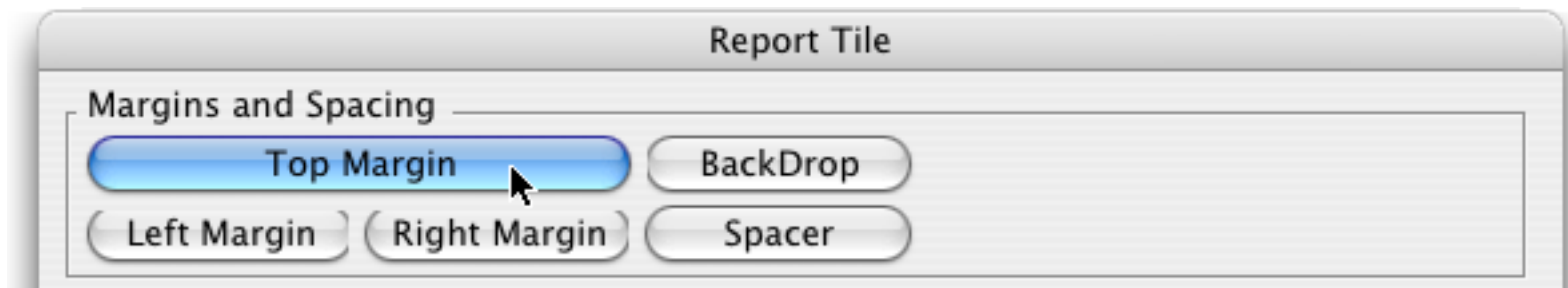
Drag the mouse over an empty spot on the form. You don't need to worry about the width, but the height of this object will become the height of the top margin. Of course you can adjust the dimensions later.



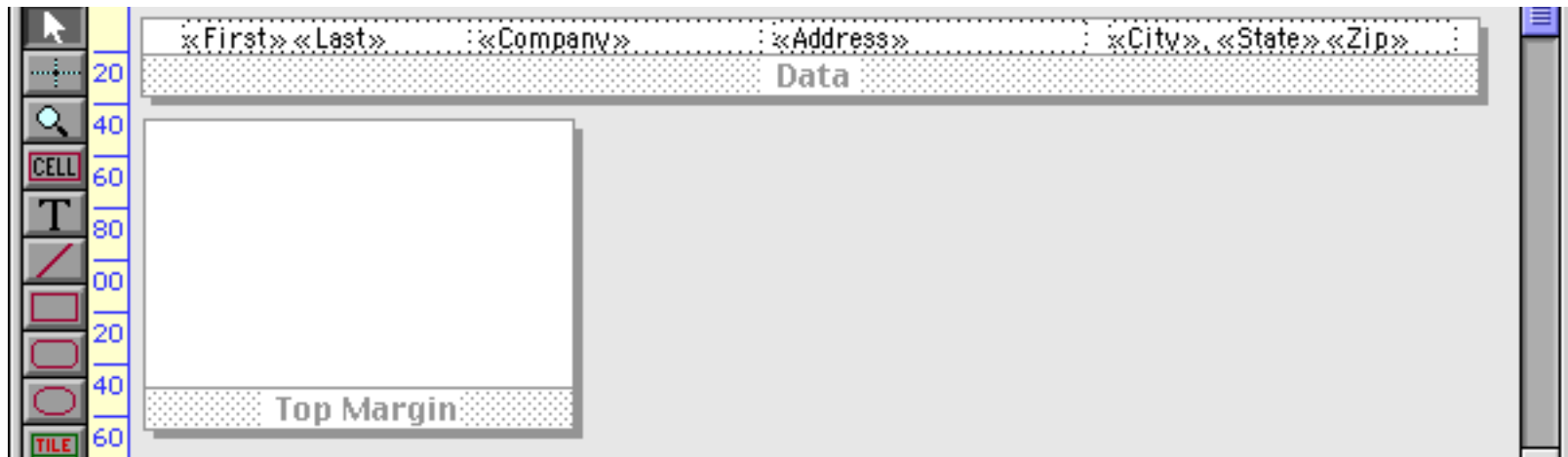
When you release the mouse the tile configuration dialog appears. If the small configuration dialog appears (as shown below) then click on the **Specialized Tiles** button.



The **Top Margin** button is near the top of the dialog.



Press this button to create the new tile.



When it's printed this report will look like this.

John Smith	Acme Widgets	12 Harmony Lane	Huntington Beach, CA
Susan Brown		783 Algonquin	Newport Beach, CA 93459
Karen Wilson	Evanston Lumber	498 Noyes	Evanston, IL 60201
Jim Nickle	Jim's Appliances	14189 8th	Newhall, CA 91321
Brian Felty	B.F. Plumbing	118 N Wilder	Lubbock, TX 79410
Bob Hanlan	Ann Arbor Lumber	6916 Morgan	Ann Arbor, MI 48104
Tim Daniels	St. Louis Lumber	3133 Cornell	St. Louis, MO 63130
John Moses		8265 Leticia	San Clemente, CA 92672
John Fabian		3 Rose Hill	Woodstock, VT 05091
Ed Ruth	Chicago Lumber	1580 N. Oconto	Chicago, IL 60634
Don Harmon	Sudderth Video	415 Sudderth	Ruidoso, NM 88345

This illustration shows how Panorama assembles the report using the top margin and data tiles. The top margin is always the top tile on the page, with any other tiles stacked below it.

John Smith	Acme Widgets	12 Harmony Lane	Huntington Beach, CA
Susan Brown		783 Algonquin	Newport Beach, CA 93459
Karen Wilson	Evanston Lumber	498 Noyes	Evanston, IL 60201
Jim Nickle	Jim's Appliances	14189 8th	Newhall, CA 91321
Brian Felty	B.F. Plumbing	118 N Wilder	Lubbock, TX 79410
Bob Hanlan	Ann Arbor Lumber	6916 Morgan	Ann Arbor, MI 48104
Tim Daniels	St. Louis Lumber	3133 Cornell	St. Louis, MO 63130
John Moses		8265 Leticia	San Clemente, CA 92672
John Fabian		3 Rose Hill	Woodstock, VT 05091
Ed Ruth	Chicago Lumber	1580 N. Oconto	Chicago, IL 60634
Don Harmon	Sudderth Video	415 Sudderth	Ruidoso, NM 88345

Unlike other tiles, Panorama will not print text or graphics that are placed on the surface of the top margin tile. The top margin tile has only one purpose—to set the top margin.

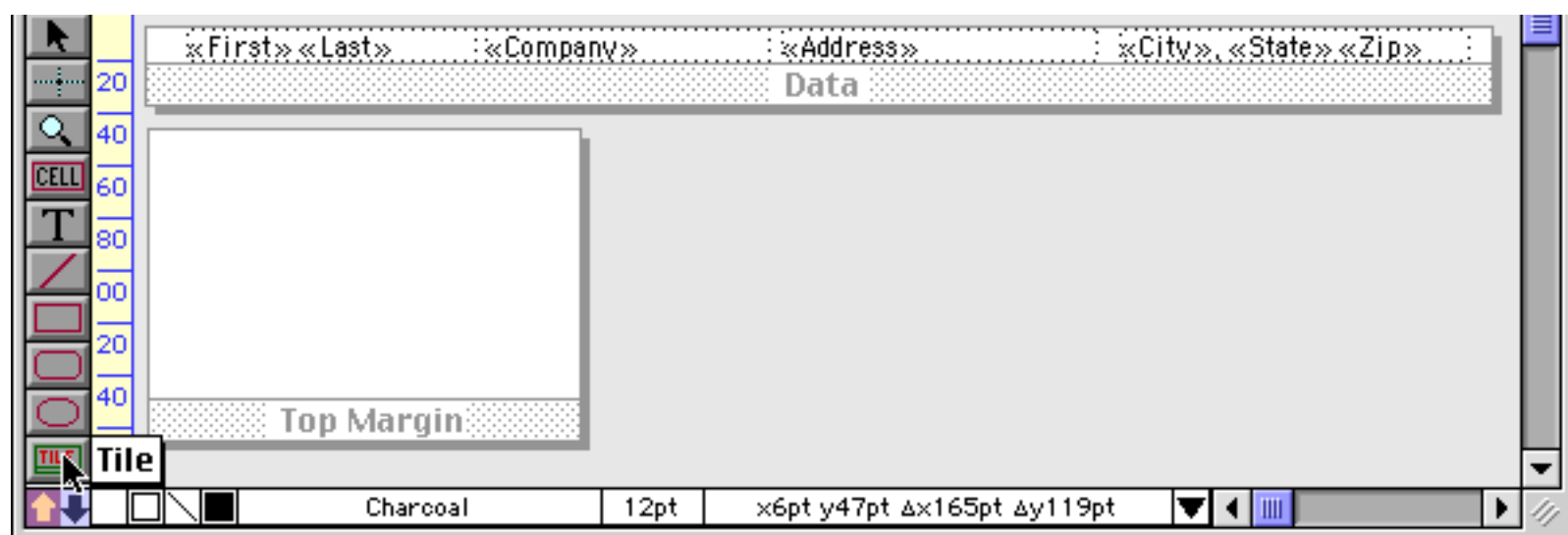
If the height of the top margin tile is less than the minimum printer margin the text may be cut off.

### Left Margin Tile

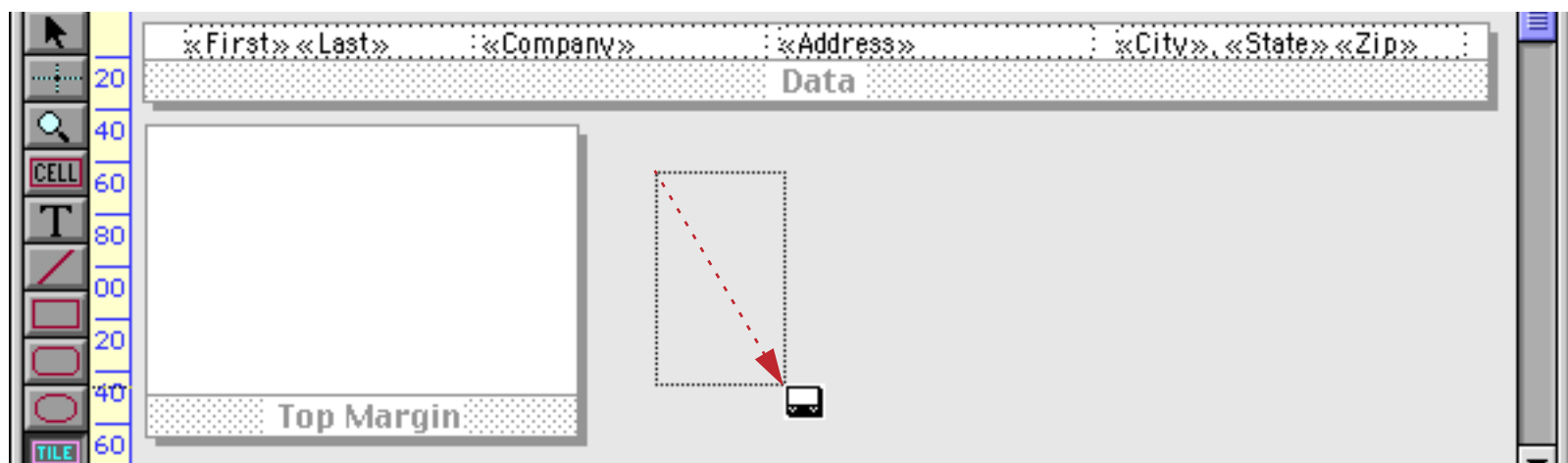
Panorama normally prints the header, footer, and data tiles with the smallest possible left margin, usually about 1/4 inch. (The actual minimum left margin depends on the type of printer you are using, and on the current **Page Setup**.)

You can use the left margin tile to change the left margin. Unlike other tiles where both dimensions are important, Panorama only cares about the width of the left margin tile. If the report contains a left margin tile, the width of that tile becomes the left margin of the report.

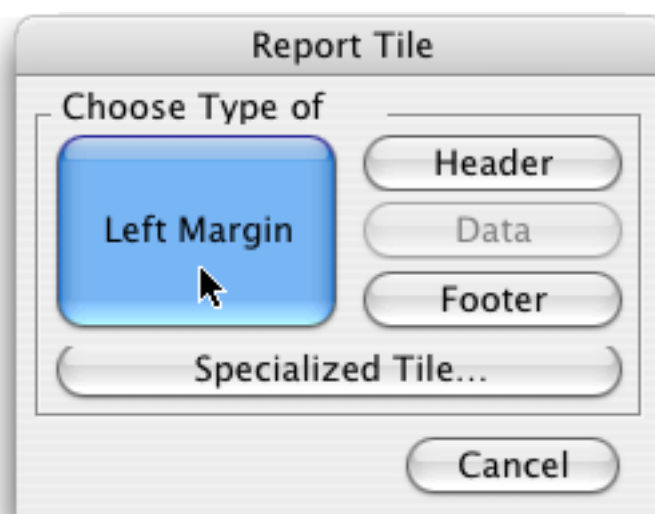
To create a left margin, start by selecting the **Tile** tool.



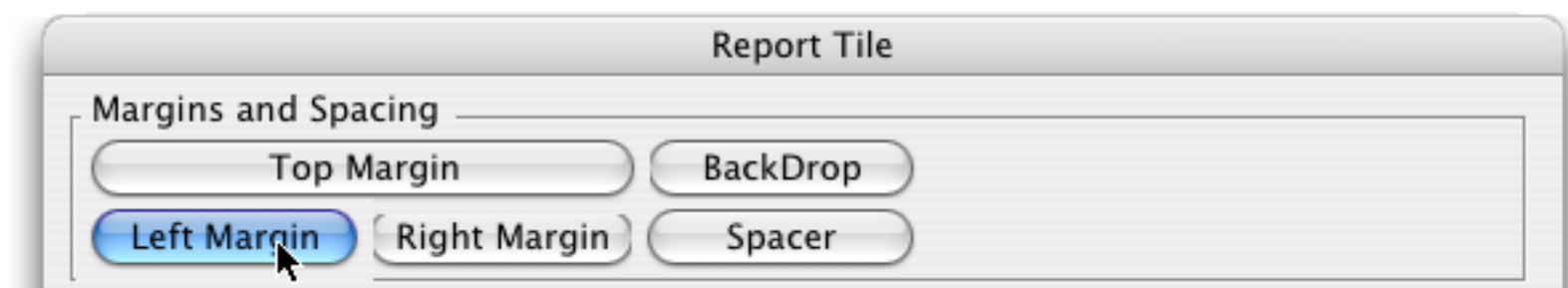
Drag the mouse over an empty spot on the form. You don't need to worry about the height, but the width of this object will become the width of the left margin. Of course you can adjust the dimensions later.



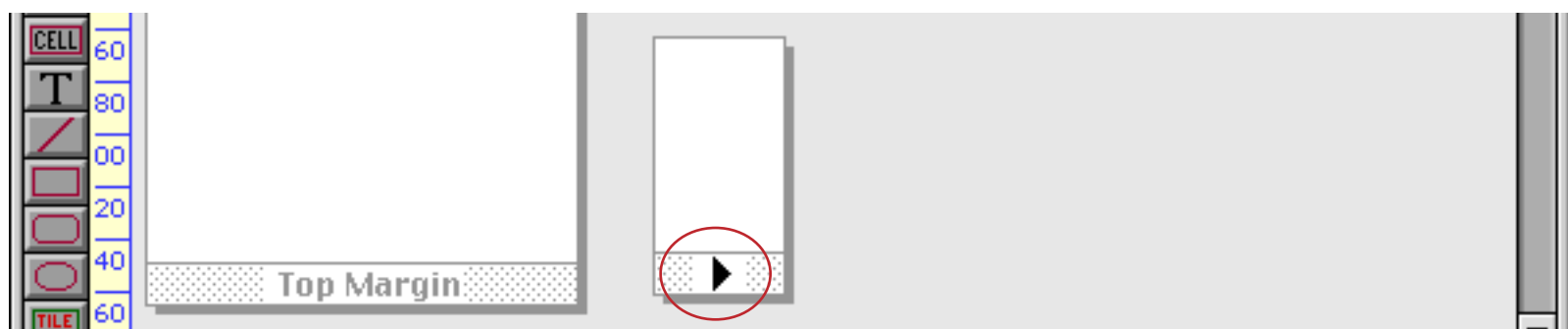
When you release the mouse the tile configuration dialog appears. If the small configuration dialog appears (as shown below) then click on the **Left Margin** button.



If the large configuration dialog appears press the **Left Margin** button, which is near the top of the dialog.



Since the Left Margin tile is often quite narrow, Panorama simply displays a triangle in the tile's drag bar.

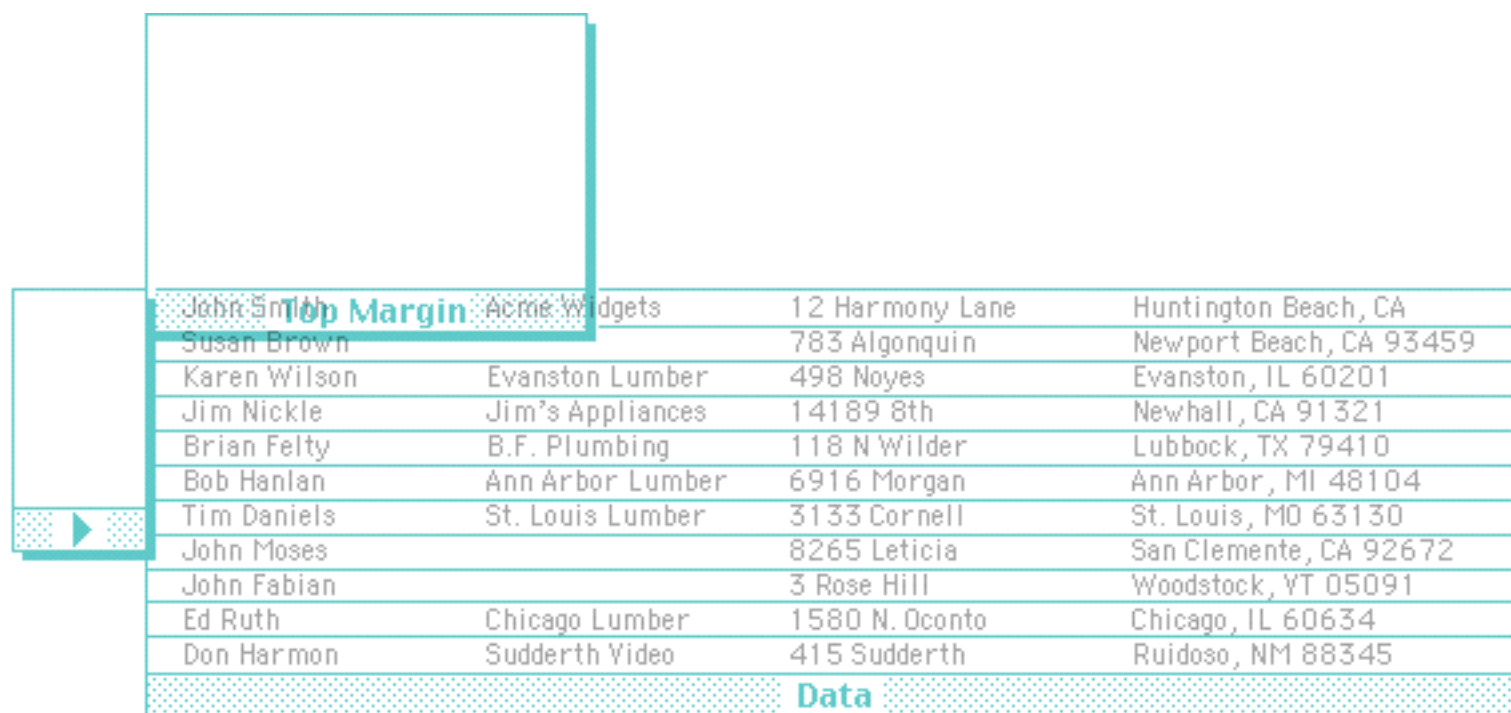




When it's printed this report will look like this.

John Smith	Acme Widgets	12 Harmony Lane	Huntington Beach, CA
Susan Brown		783 Algonquin	Newport Beach, CA 93459
Karen Wilson	Evanston Lumber	498 Noyes	Evanston, IL 60201
Jim Nickle	Jim's Appliances	14189 8th	Newhall, CA 91321
Brian Felty	B.F. Plumbing	118 N Wilder	Lubbock, TX 79410
Bob Hanlan	Ann Arbor Lumber	6916 Morgan	Ann Arbor, MI 48104
Tim Daniels	St. Louis Lumber	3133 Cornell	St. Louis, MO 63130
John Moses		8265 Leticia	San Clemente, CA 92672
John Fabian		3 Rose Hill	Woodstock, VT 05091
Ed Ruth	Chicago Lumber	1580 N. Oconto	Chicago, IL 60634
Don Harmon	Sudderth Video	415 Sudderth	Ruidoso, NM 88345

This illustration shows how Panorama assembles the report using the top margin and data tiles. The top margin is always the top tile on the page, with any other tiles stacked below it.



Unlike other tiles, Panorama will not print text or graphics that are placed on the surface of the left margin tile. The left margin tile has only one purpose—to set the left margin.

### Bottom Margin

Panorama does not have a special tile for setting the bottom margin. Instead, you should include space for the bottom margin in the footer tile (see “[Footer Tile](#)” on page 471).

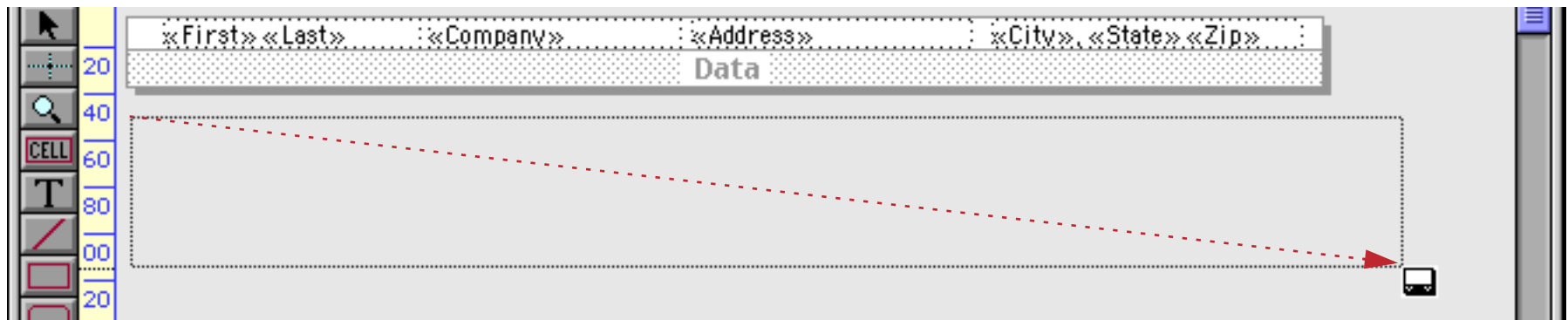
### Headers and Footers

Many reports have headers at the top of each page and footers at the bottom. As you might guess, headers and footers are set up with report tiles!

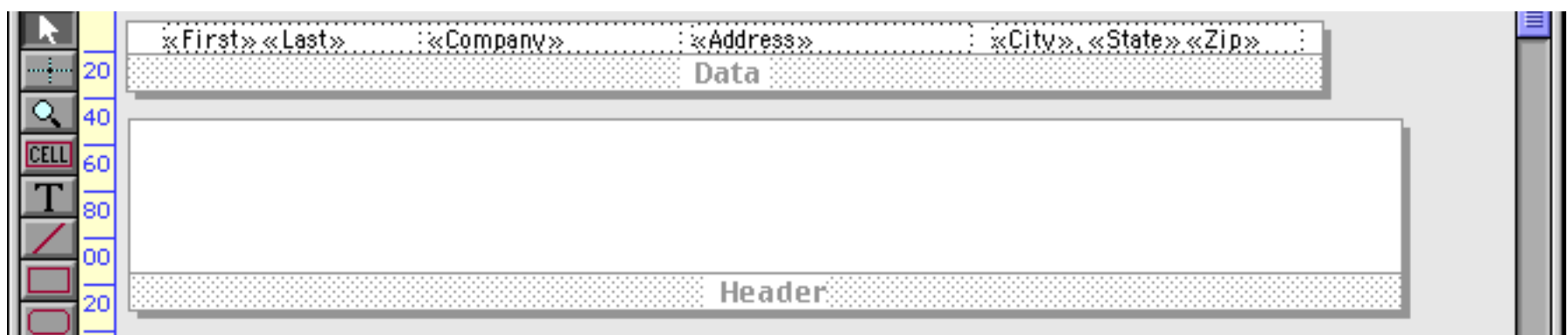
## Header Tile

The header tile is printed at the top left of each page. The header can be used to print the report title, page number (see “[Page Numbers](#)” on page 472), date (see “[Printing the Current Date and Time](#)” on page 473) or anything else you want.

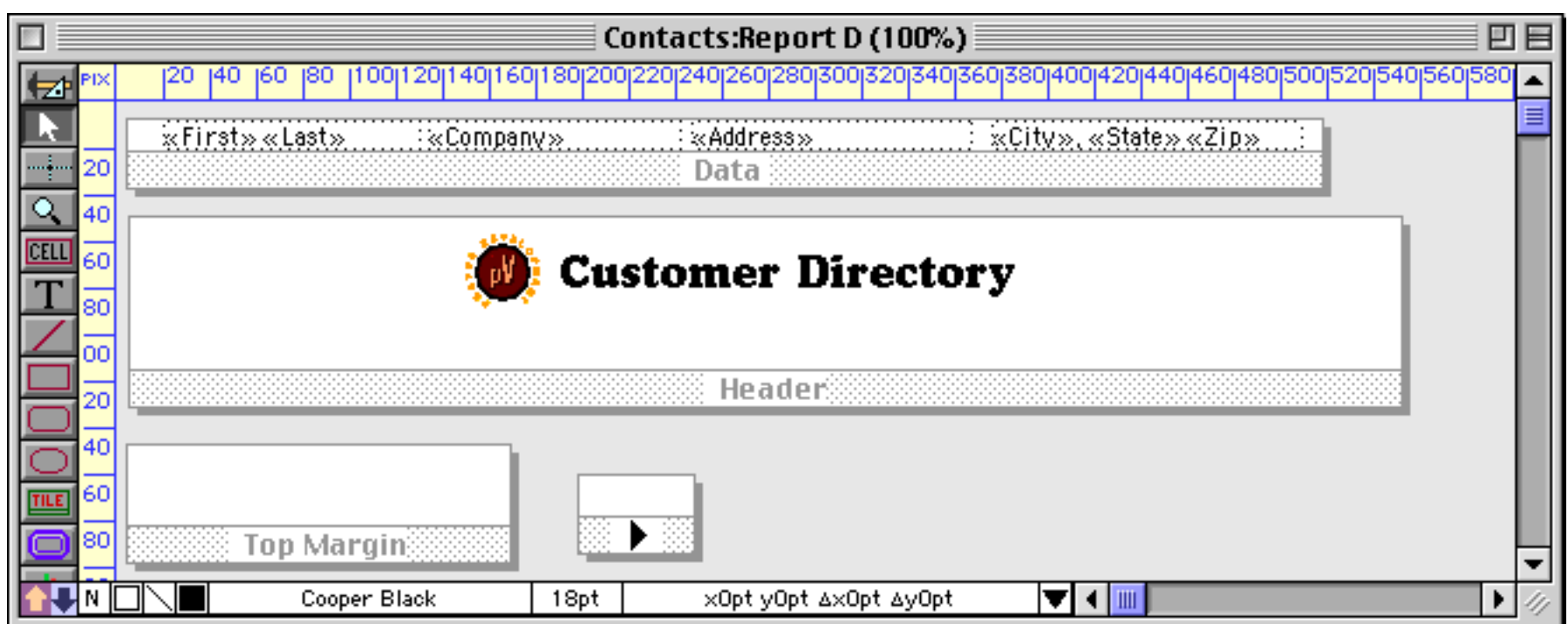
To create a header, start by selecting the **Tile** tool, just as with any other tile. Drag the mouse over an empty spot on the form, leaving the proper amount of space for the header you want to create. Usually the header is approximately the entire width of the page. (Of course you can adjust the dimensions later.) The header tile can be in any empty spot on the form. In this example we are creating the header tile below the data tile, even though the header will eventually print above the data.



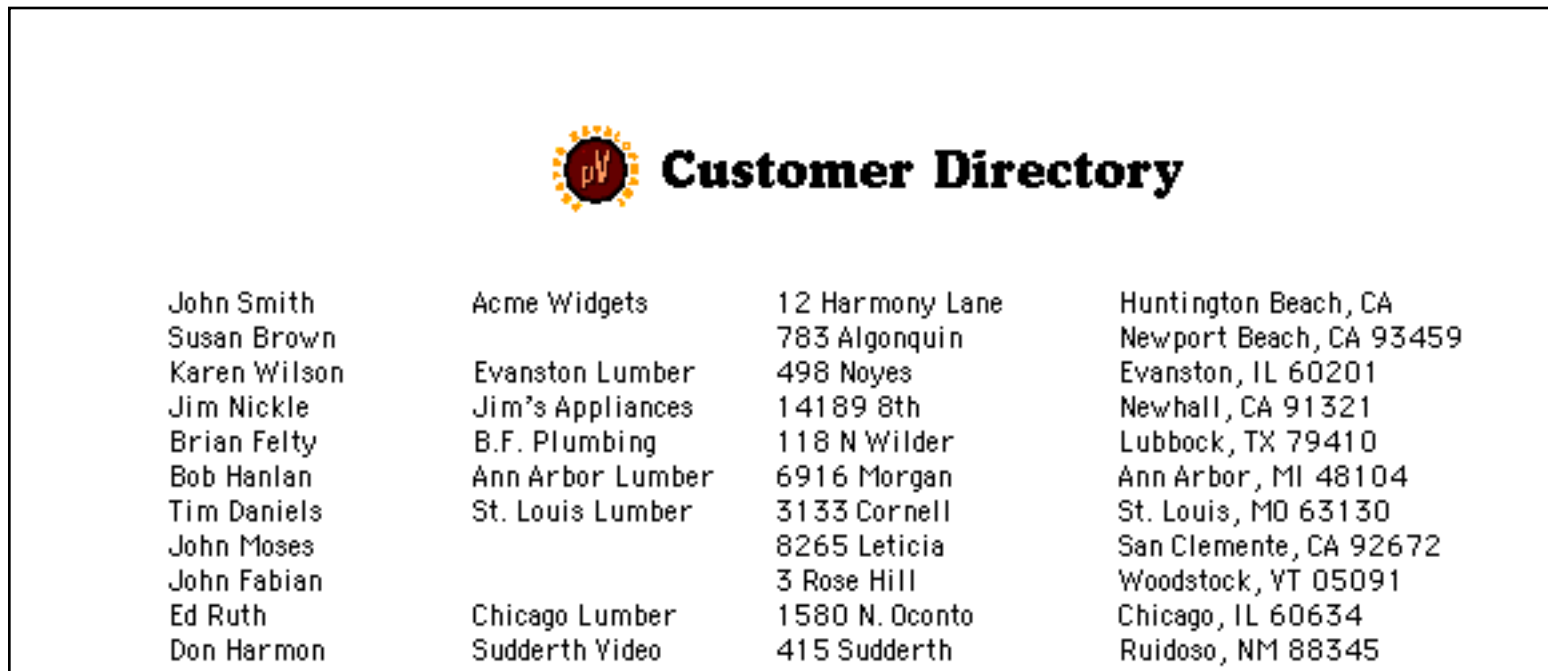
When you release the mouse the tile configuration dialog appears. If the small configuration dialog appears (as shown below) then click on the **Header** button. If the large configuration dialog appears press the **Header** button, in the *Top of Page* section. Here's the new tile.




A blank header tile isn't much use, so we'll add some text and graphics to the header.



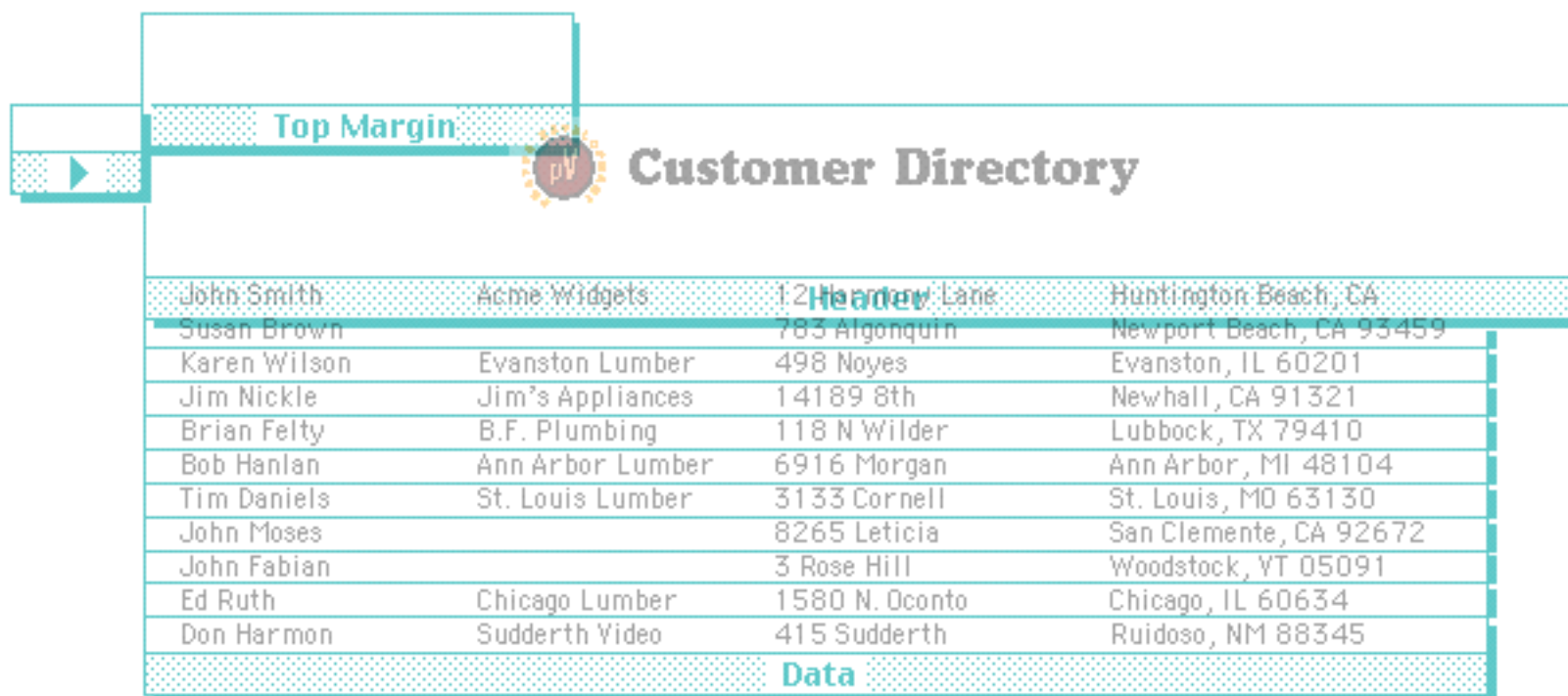
When it's printed this report will look like this.



 **Customer Directory**

John Smith	Acme Widgets	12 Harmony Lane	Huntington Beach, CA
Susan Brown		783 Algonquin	Newport Beach, CA 93459
Karen Wilson	Evanston Lumber	498 Noyes	Evanston, IL 60201
Jim Nickle	Jim's Appliances	14189 8th	Newhall, CA 91321
Brian Felty	B.F. Plumbing	118 N Wilder	Lubbock, TX 79410
Bob Hanlan	Ann Arbor Lumber	6916 Morgan	Ann Arbor, MI 48104
Tim Daniels	St. Louis Lumber	3133 Cornell	St. Louis, MO 63130
John Moses		8265 Leticia	San Clemente, CA 92672
John Fabian		3 Rose Hill	Woodstock, VT 05091
Ed Ruth	Chicago Lumber	1580 N. Oconto	Chicago, IL 60634
Don Harmon	Sudderth Video	415 Sudderth	Ruidoso, NM 88345

This illustration shows how Panorama assembles the report using the margin, header and data tiles.

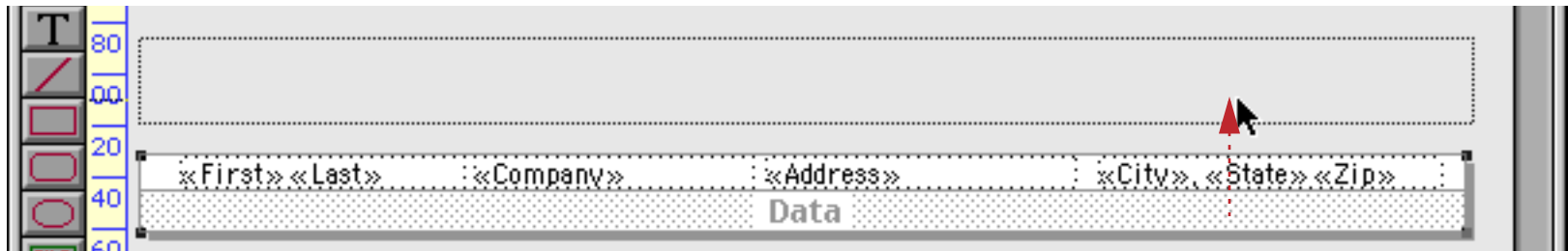


### Creating a Header Tile by Duplicating the Data Tile

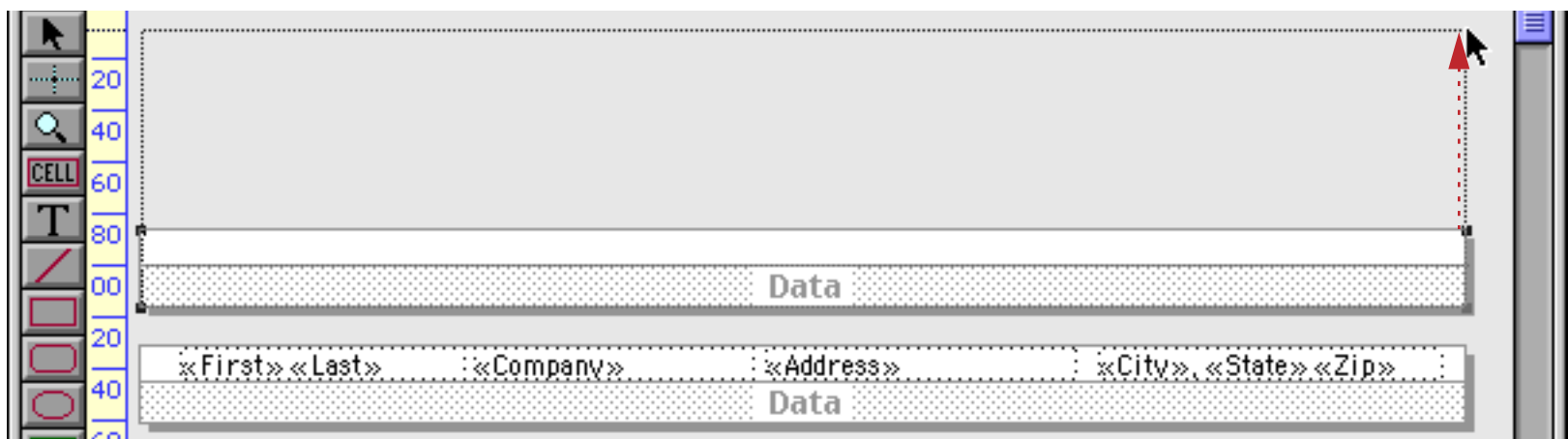
An alternate way to create the header tile is to duplicate the data tile. This makes it easy to line up items between the two tiles. Start with just a data tile.



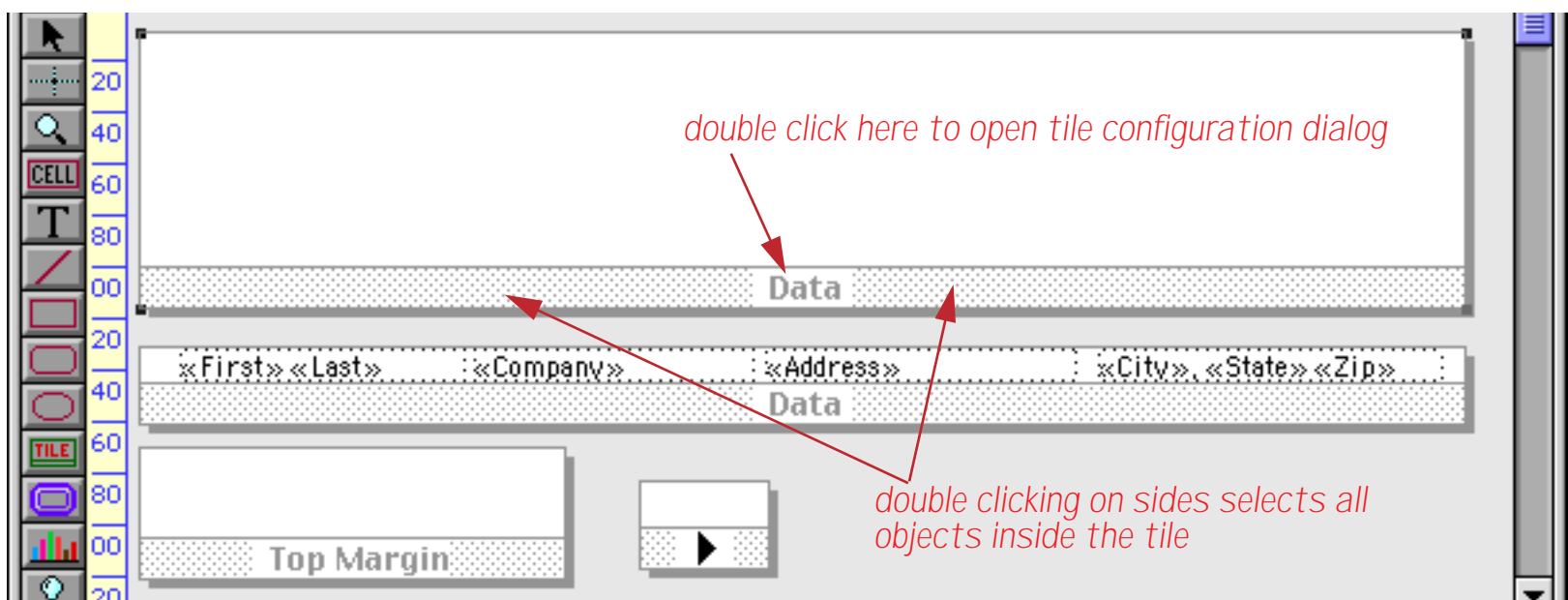
To duplicate the tile, hold down the **Option** key (Mac) or **Alt** key (Windows) and drag the tile. You probably also want to hold down the **Shift** key at the same time to make sure that the two tiles stay in perfect alignment.



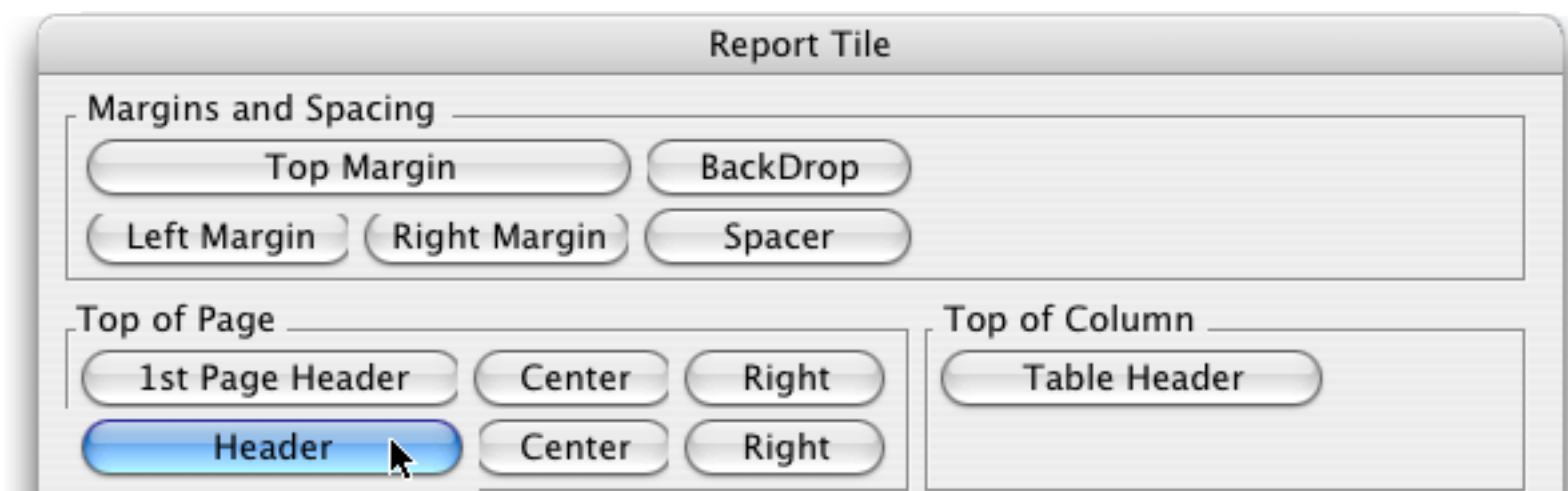
When you release the mouse your form will contain two data tiles. Grab one of the handles to adjust the height of the new tile.



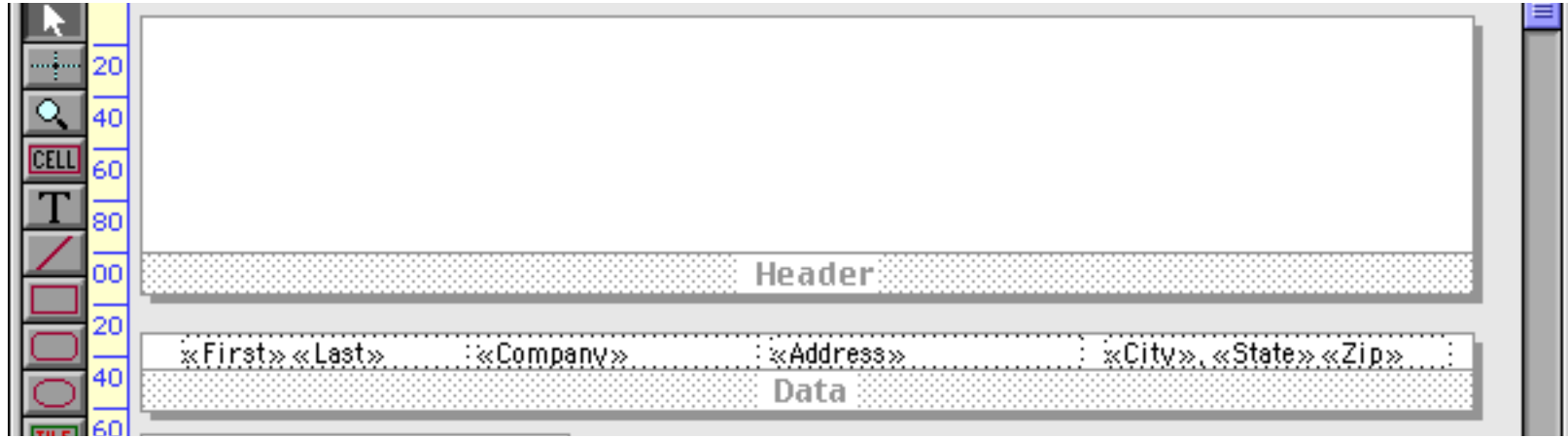
To convert the new tile from a data tile into a header tile, double click on the word **Data**.



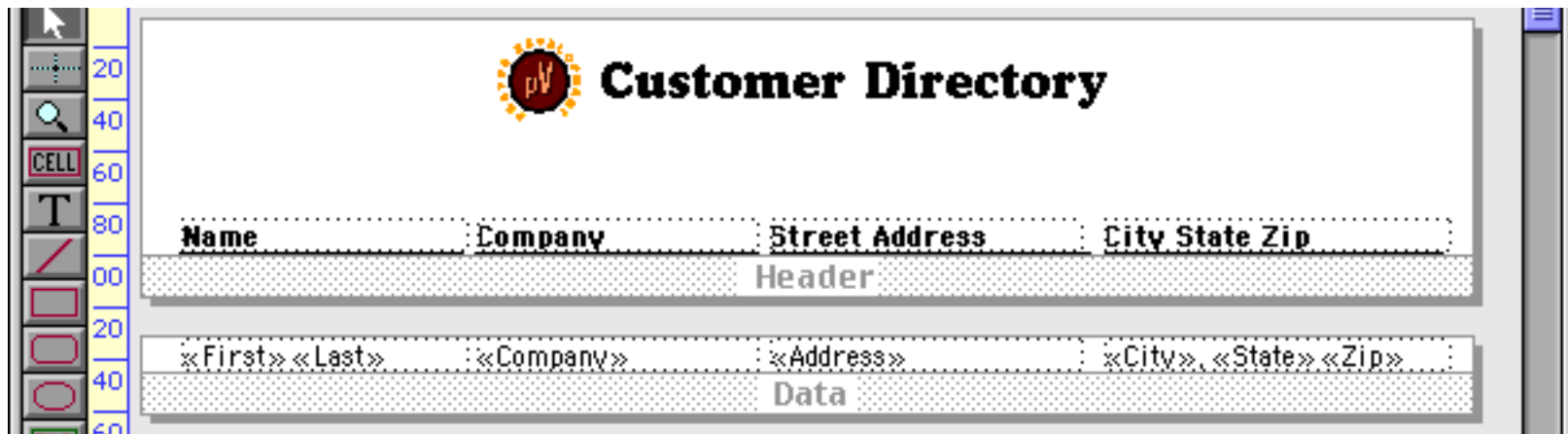
This opens the tile configuration dialog.




Press the **Header** button to convert the tile.



Now we can add graphics and text to the header. Anything placed on the header will line up vertically with objects in the same position on the data tile.



Here's the printed report.

 <b>Customer Directory</b>			
<b>Name</b>	<b>Company</b>	<b>Street Address</b>	<b>City State Zip</b>
John Smith	Acme Widgets	12 Harmony Lane	Huntington Beach, CA
Susan Brown		783 Algonquin	Newport Beach, CA 93459
Karen Wilson	Evanston Lumber	498 Noyes	Evanston, IL 60201
Jim Nickle	Jim's Appliances	14189 8th	Newhall, CA 91321
Brian Felty	B.F. Plumbing	118 N Wilder	Lubbock, TX 79410
Bob Hanlan	Ann Arbor Lumber	6916 Morgan	Ann Arbor, MI 48104
Tim Daniels	St. Louis Lumber	3133 Cornell	St. Louis, MO 63130
John Moses		8265 Leticia	San Clemente, CA 92672
John Fabian		3 Rose Hill	Woodstock, VT 05091
Ed Ruth	Chicago Lumber	1580 N. Oconto	Chicago, IL 60634
Don Harmon	Sudderth Video	415 Sudderth	Ruidoso, NM 88345

Once again Panorama builds these reports by sliding the tile's into place.

Name	Company	Street Address	City State Zip
John Smith	Acme Widgets	12 Harmony Lane	Huntington Beach, CA
Susan Brown		783 Algonquin	Newport Beach, CA 93459
Karen Wilson	Evanston Lumber	498 Noyes	Evanston, IL 60201
Jim Nickle	Jim's Appliances	14189 8th	Newhall, CA 91321
Brian Felty	B.F. Plumbing	118 N Wilder	Lubbock, TX 79410
Bob Hanlan	Ann Arbor Lumber	6916 Morgan	Ann Arbor, MI 48104
Tim Daniels	St. Louis Lumber	3133 Cornell	St. Louis, MO 63130
John Moses		8265 Leticia	San Clemente, CA 92672
John Fabian		3 Rose Hill	Woodstock, VT 05091
Ed Ruth	Chicago Lumber	1580 N. Oconto	Chicago, IL 60634
Don Harmon	Sudderth Video	415 Sudderth	Ruidoso, NM 88345

### Footer Tile

The footer tile is printed at the bottom left of each page. The footer can be used to print the report title, page number (see “[Page Numbers](#)” on page 472), date (see “[Printing the Current Date and Time](#)” on page 473) or anything else you want.

You can create a footer with the **Tile** tool (see “[Header Tile](#)” on page 467) or by duplicating another tile (see “[Creating a Header Tile by Duplicating the Data Tile](#)” on page 468). In this example the footer was created by drag duplicating the data tile.

Name	Company	Street Address	City State Zip
«First» «Last»	«Company»	«Address»	«City», «State» «Zip»
- «Page#» -			



Here's what the bottom of this page looks like.

Alan Spencer		182 Dell Rd	Northbrook, IL 60062
Raymond Wood		5420 W Crosby	Slaton, TX 79364
Lee Tucker	Latham Video	4792 Latham	Mountain View, CA 94041
Logan Nourse	Palo Alto Lumber	1828 Amaranta	Palo Alto, CA 94306
David Peters	D.P. Plumbing	191 Treg Lane	Concord, CA 94518
Tom Cane		8820 Sierra Court	Dublin, CA 94568
Pat Turner	P.T. Plumbing	1009 Secret Bay	Davis, CA 95616
Scott Lay	Portland Lumber	1278 N.E. 136th	Portland, OR 97230
John Draper	Exeter Video	446 Exeter Rd	Hampton, NH 03842

- 1 -

Notice that there is a gap between the bottom of the last line of data and the footer tile. This is because the Panorama always aligns the footer with the bottom of the page. The footer is printed as close to the bottom of the page as possible.

Alan Spencer		182 Dell Rd	Northbrook, IL 60062
Raymond Wood		5420 W Crosby	Slaton, TX 79364
Lee Tucker	Latham Video	4792 Latham	Mountain View, CA 94041
Logan Nourse	Palo Alto Lumber	1828 Amaranta	Palo Alto, CA 94306
David Peters	D.P. Plumbing	191 Treg Lane	Concord, CA 94518
Tom Cane		8820 Sierra Court	Dublin, CA 94568
Pat Turner	P.T. Plumbing	1009 Secret Bay	Davis, CA 95616
Scott Lay	Portland Lumber	1278 N.E. 136th	Portland, OR 97230
John Draper	Exeter Video	446 Exeter Rd	Hampton, NH 03842

- 1 -

Footer

If you want to create a footer that is right below the data tiles with no gap, use the **Table Footer** tile (see Chapter 21 of the **Panorama Handbook** to learn about this option).

## Page Numbers

Panorama can automatically calculate and print a page number on each page of your report. One technique for printing the page number is to use an auto-wrap text object. The page number is merged into the text by typing «page#» into the object. (On a Macintosh the « chevron is **Option-\** and the » chevron is **Shift-Option-\**. On Windows systems the « chevron is **Alt-0171** and the » chevron is **Alt-0187**.) The illustration below shows how to print page numbers on the upper right hand corner of a report. Notice that the text object has been set to right justify (Style Menu) so that the page number will be flush with the right edge of the report.

Contacts			
Name	Company	Street Address	City State Zip
«First» «Last»	«Company»	«Address»	«City», «State» «Zip»

Page «Page#»

Here is what the first three pages of this report look like.

Name	Company	Street Address	City State Zip
John Smith	Acme Widgets	12 Harmony Lane	Huntington Beach, CA
Susan Brown		783 Algonquin	Newport Beach, CA 93459
Karen Wilson	Evanston Lumber	498 Noyes	Evanston, IL 60201
Jim Nickle	Jim's Appliances	14189 8th	Newhall, CA 91321
Brian Felty	B.F. Plumbing	118 N Wilder	Lubbock, TX 79410

To center the page number at the bottom of each page, place an auto-wrap text or Text Display SuperObject in the center of the footer tile. To make sure that the page number is centered, the text object should be centered within the footer tile. You also need to check the center justify option in the Style Menu.



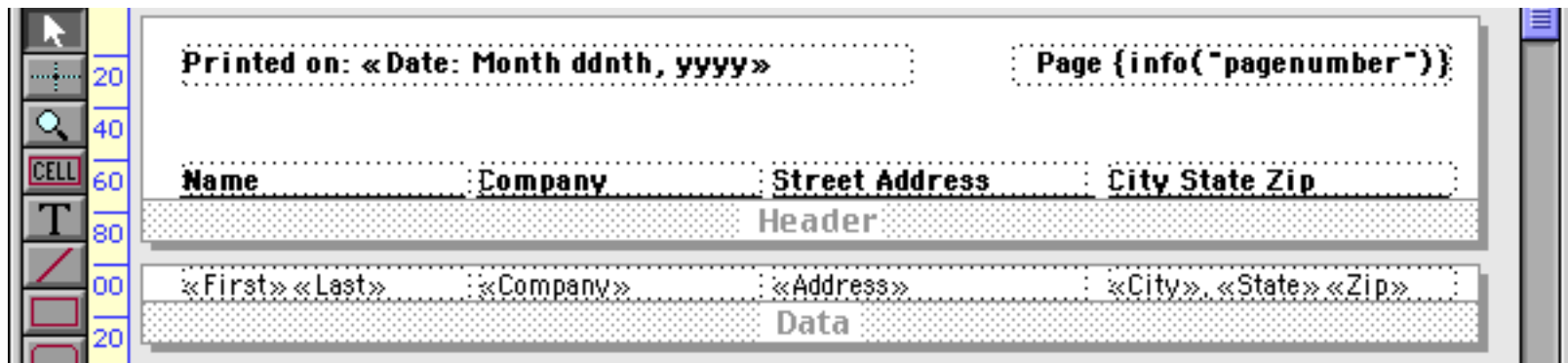
### Printing the Current Date and Time

Panorama can automatically print the current date and time on each page of your report. One technique for printing the date is to use an auto-wrap text object. The page number is merged into the text by typing «date:pattern» into the object. (On a Macintosh the « chevron is **Option-** and the » chevron is **Shift-Option-**. On Windows systems the « chevron is **Alt-0171** and the » chevron is **Alt-0187**.) Don't actually type in the word pattern, instead choose one of the patterns from the table below.

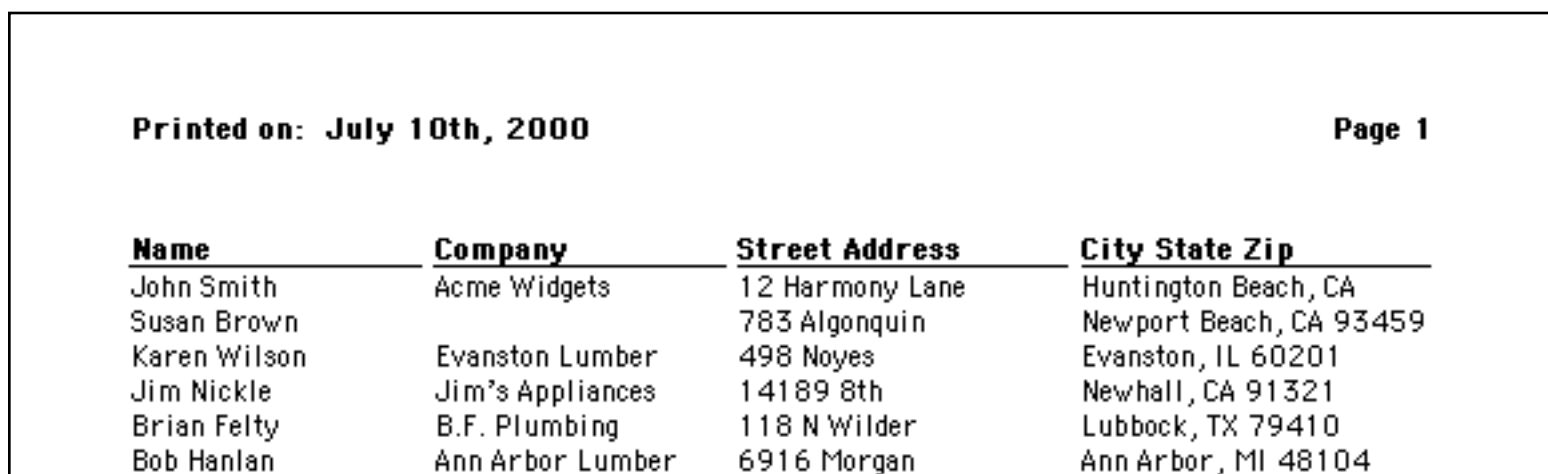
Pattern	Example
mm/dd/yy	3/9/04
MM/DD/YY	03/09/04

Pattern	Example
mm-dd-yyyy	3-9-2004
dd-MON-yy	9-MAR-04
dd-Month-yy	9-March-04
Month dd, yyyy	March 9, 2004
Month ddnth, yyyy	March 9th, 2004
DayOfWeek, Month dd	Thursday, March 9

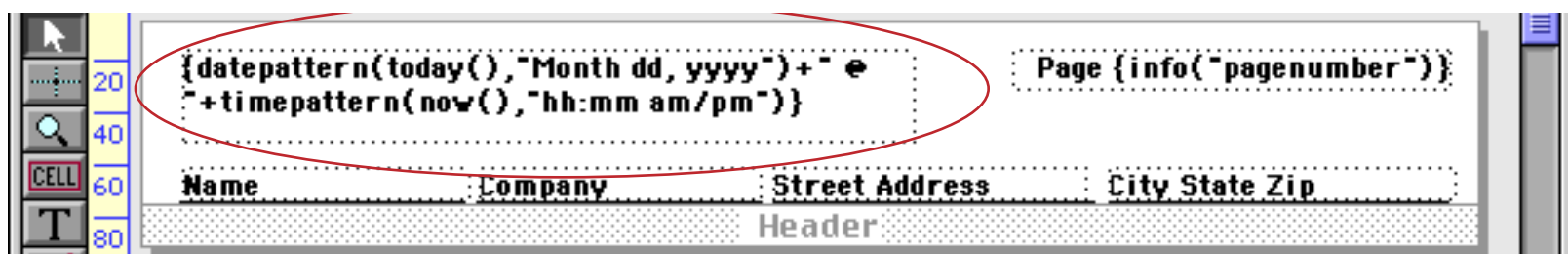
The illustration below shows how to print the current date in the upper left hand corner of a report.



Here's the top of the first page of this report.



To include the current time in a printed report you must use the `now()` and `timepattern()` functions. You can either merge these function into an auto-wrap text object or use a Text Display SuperObject. The example below prints both the date and time using a formula.



Here is the printed report.

July 10, 2000 @ 2:31 PM		Page 1	
Name	Company	Street Address	City State Zip
John Smith	Acme Widgets	12 Harmony Lane	Huntington Beach, CA
Susan Brown		783 Algonquin	Newport Beach, CA 93459
Karen Wilson	Evanston Lumber	498 Noyes	Evanston, IL 60201
Jim Nickle	Jim's Appliances	14189 8th	Newhall, CA 91321
Brian Felty	B.F. Plumbing	118 N Wilder	Lubbock, TX 79410
Bob Hanlan	Ann Arbor Lumber	6916 Morgan	Ann Arbor, MI 48104
Tim Daniels	St. Louis Lumber	3133 Cornell	St. Louis, MO 63130
John Moses		8265 Leticia	San Clemente, CA 92672
John Fabian		3 Rose Hill	Woodstock, VT 05091

You can actually merge in any formula you want into the header and/or footer.

### First Page Header Tile

Panorama normally prints the header tile (or tiles) at the top of each page (see above). However, if the report includes a first page header tile, that tile will be printed at the top of the first page. All subsequent pages will print using the normal header tile (or tiles). Here is an example of a form that contains both a first page header and a regular header that will print on the second, third, and any additional pages.

The screenshot shows a report design tool window titled "Contacts:Report G (100%)". The report content is as follows:

*My Contact File*

{datepattern(today(),"Month ddnth, yyyy")}  
{info("records")} records

**Name**      **Company**      **Street Address**      **City State Zip**

1st page Header

{datepattern(today(),"Month dd, yyyy")} + " @ " + timepattern(now(),"hh:mm am/pm")

Page {info("pagenumber")}

**Name**      **Company**      **Street Address**      **City State Zip**

Header

«First» «Last»      «Company»      «Address»      «City», «State» «Zip»

Data

Top Margin

The tool interface includes a ruler at the top (PIX 20-500), a vertical ruler on the left (20-80), and a status bar at the bottom showing "Charcoal", "12pt", and "xOpt yOpt ΔxOpt ΔyOpt".

Here are the first two pages of the printed report generated by this form. The first page includes the **First Page Header**, while the second (and subsequent pages) use the regular Header tile.

<b>July 10, 2000 @ 2:53 PM</b>			<b>Page 2</b>
<b>Name</b>	<b>Company</b>	<b>Street Address</b>	<b>City State Zip</b>
Glen Knock	South Portland	909 Wescott Rd	South Portland, ME 04106
Carl Berg	C.B. Plumbing	161 Norton St	New Haven, CT 06511
Wes Lemarr		57 Hobart Ave	Rutherford, NJ 07070
Charles Dalbert	New York Lumber	171 Broadway	New York, NY 10003
Brad Hess	Brooklyn Lumber	128 70th St	Brooklyn, NY 11209
Tim Henry	Suffolk Lumber	2375 Driver Lane	Suffolk, VA 23435

# *My Contact File*

July 10th, 2000  
104 records

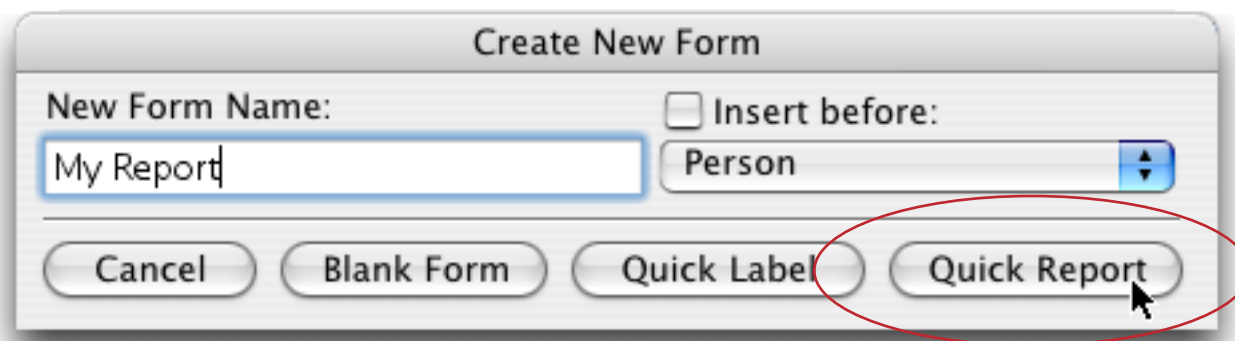
<b>Name</b>	<b>Company</b>	<b>Street Address</b>	<b>City State Zip</b>
John Smith	Acme Widgets	12 Harmony Lane	Huntington Beach, CA
Susan Brown		783 Algonquin	Newport Beach, CA 93459
Karen Wilson	Evanston Lumber	498 Noyes	Evanston, IL 60201
Jim Nickle	Jim's Appliances	14189 8th	Newhall, CA 91321
Brian Felty	B.F. Plumbing	118 N Wilder	Lubbock, TX 79410
Bob Hanlan	Ann Arbor Lumber	6916 Morgan	Ann Arbor, MI 48104
Tim Daniels	St. Louis Lumber	3133 Cornell	St. Louis, MO 63130
John Moses		8265 Leticia	San Clemente, CA 92672
John Fabian		3 Rose Hill	Woodstock, VT 05091
Ed Ruth	Chicago Lumber	1580 N. Oconto	Chicago, IL 60634
Don Harmon	Sudderth Video	415 Sudderth	Ruidoso, NM 88345
Abe Fierstein	Van Nuys Lumber	1571 Haskell	Van Nuys, CA 91409
Randy Cross	Randy's Appliances	133 Hunt Rd	Chelsford, MA 01824
Jeffrey Rodman		2 Cary Rd	Chestnut Hill, MA 02167
Steve Jackson	Ann Arbor Lumber	389 Worden	Ann Arbor, MI 48103
Dick Hardlee		4151 Polstar	Plano, TX 75075
Don Meadows	Austin Lumber	1144 A West 6th	Austin, TX 78703
Jerry Bowen	Peacock Video	2847 Peacock	Highland, CA 92346
Thom Getchell	Thom's Appliances	543 Laurel	Menlo Park, CA 94025
Brian Smith	Brian's Appliances	1844 Tiburon	Hollister, CA 95023
David Blair	DB Printing	869 W. Temple	Lenox, IA 50851
Keith Baker	Northgate Video	552 Northgate	Lindenhurst, IL 60046

22  
4  
9  
0  
4  
961

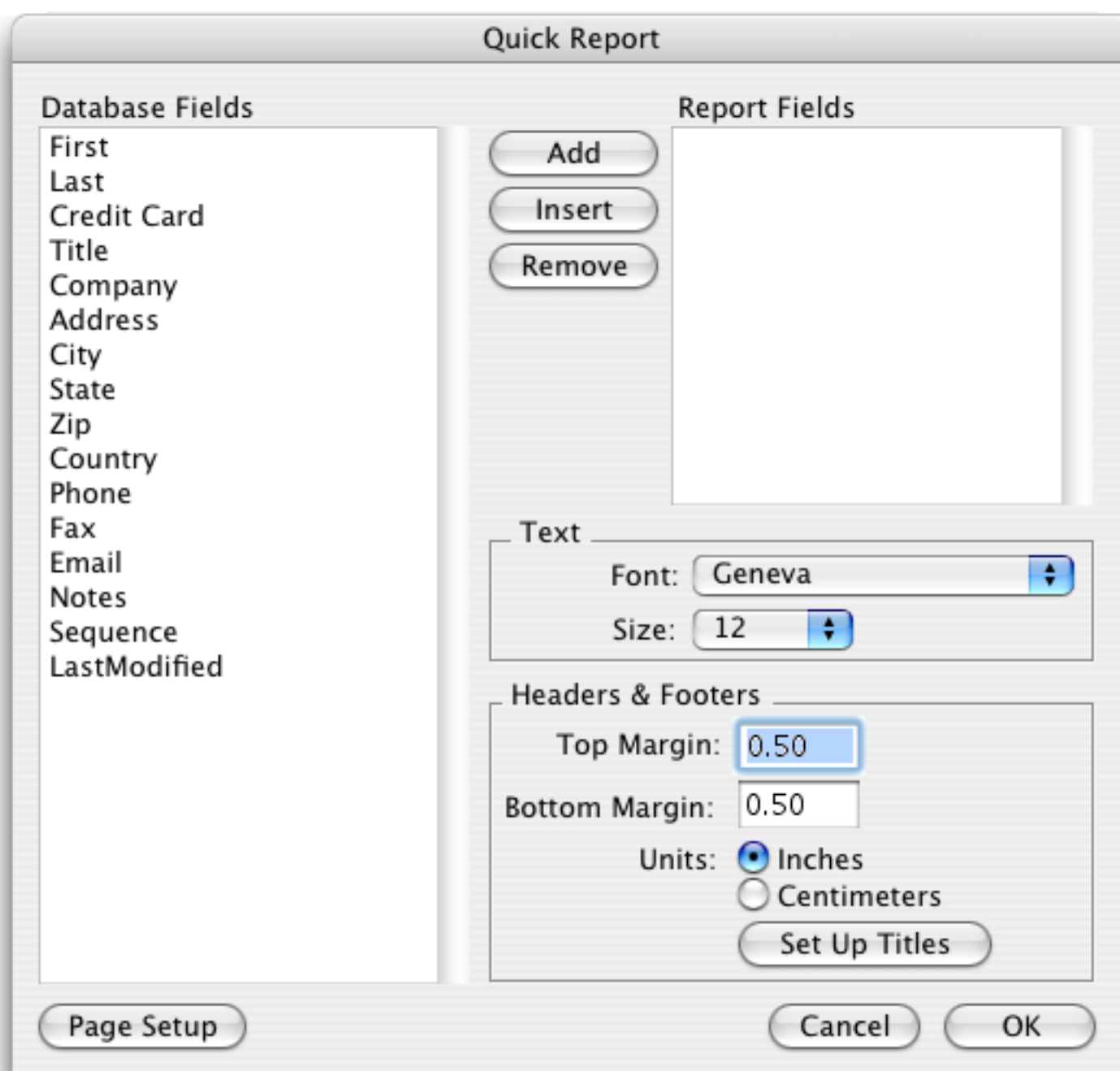
The **First Page Header** tile can also be used to create a title page for the report. To do this simply enlarge the **First Page Header** tile until it is large enough to cover an entire page. In that case Panorama will print only that tile on the first page. The regular report will begin on the second page.

## The QuickReport Dialog

When you create a new form, Panorama gives you the option of creating a blank form or automatically creating a label or report.



The **QuickReport** button opens a dialog that can do most or all of the work of setting up a report for you. The QuickReport dialog allows you to automatically create a tabular report.



On the left is a list of all the fields in the database. On the right is a list of the fields that will be included in the report. In the dialog the report fields are listed from top to bottom, in the actual report they will be printed from left to right. To set up the report, copy the fields you want to include in the report from the list of fields on the left to the report list on the right.



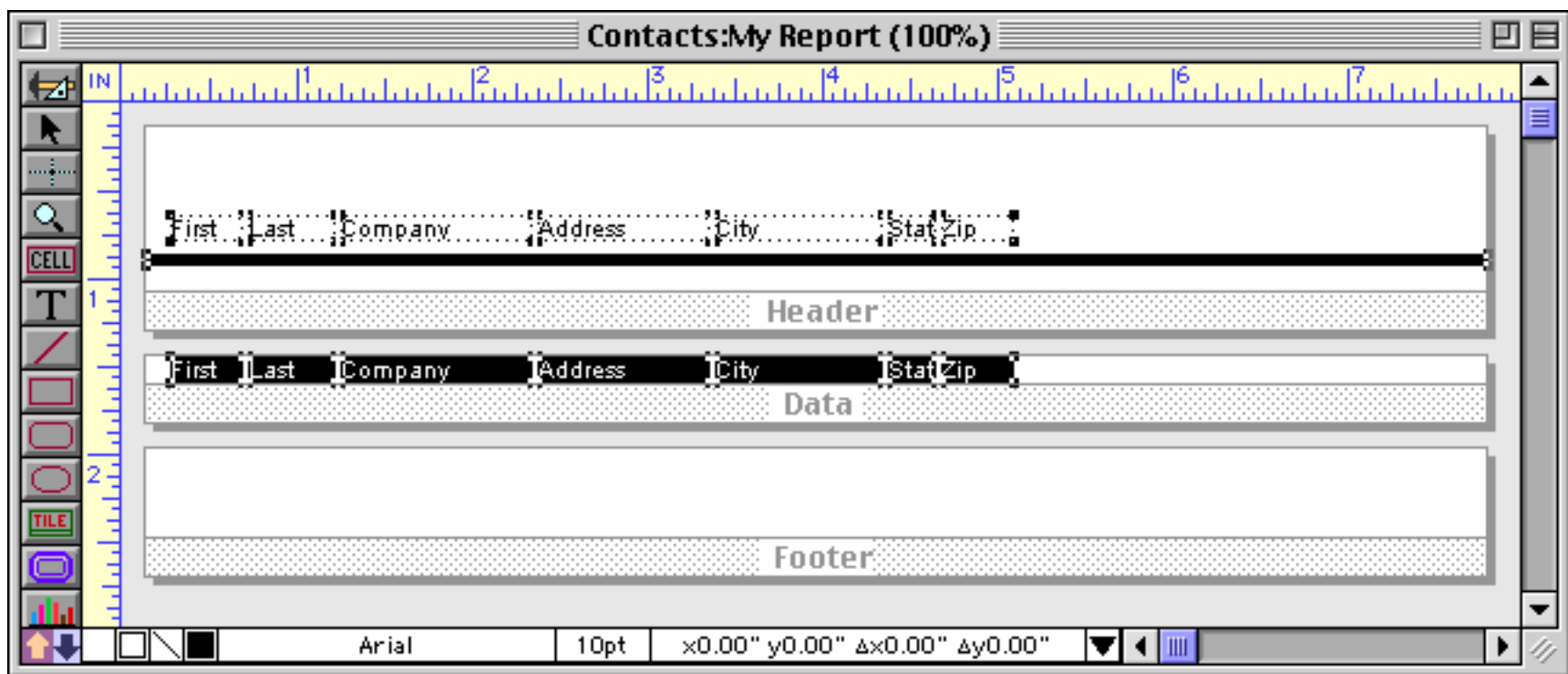
The easiest way to copy a field from the left to the right is to double click on the field name. Double clicking on a field name adds the field to the end of the list on the right. If a field name on the right is already selected, then double clicking on a field name on the left replaces that field. The **Add** button works the same as double clicking.

To insert a field into the middle of the list on the right, first select a field on the right, then select a field on the left. Press the **Insert** button to insert the new field into the list on the right.

To delete a field from the list on the right, select the field and press the **Remove** button. You can also delete a field from the list on the right by double clicking on it.

If you want to use a non-standard page size, you should use the **Page Setup** button to set it up. Be sure to set up the page size before you press the **OK** button. This lets QuickReport know how wide the page will be.

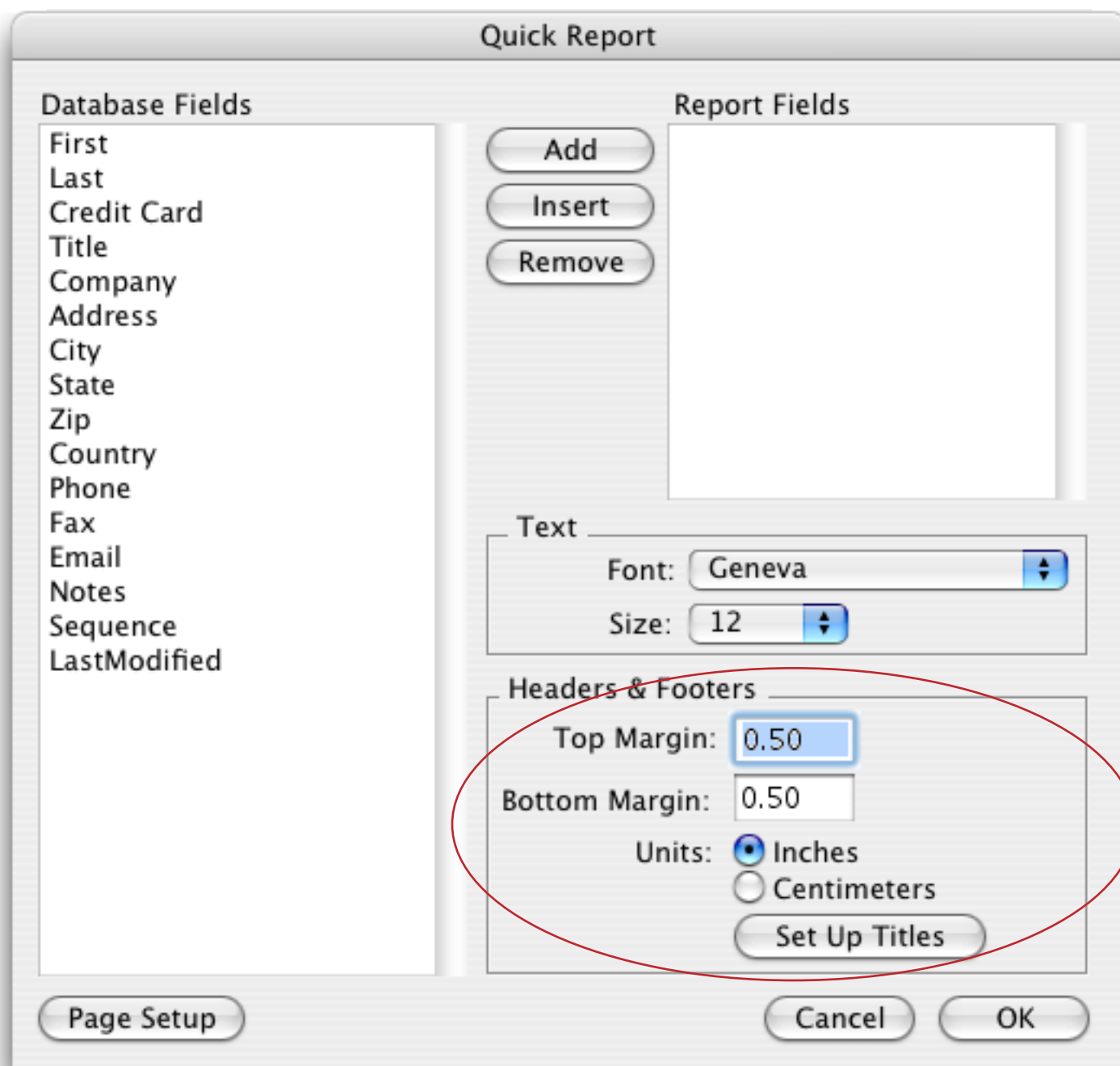
Press the **OK** button to generate the actual report. Panorama will generate all the tiles and cells needed for the report.



You can use this report as is, or you can use it as a starting point and modify it to suit your needs.

First	Last	Company	Address	City	Stat	Zip
John	Smith	Acme Widgets	12 Harmony	Huntington	CA	9264
Susan	Brown		783 Algonquin	Newport	CA	9345
Karen	Wilson	Evanston	498 Noyes	Evanston	IL	6020
Jim	Nickle	Jim's	14189 8th	Newhall	CA	9132
Brian	Felty	B.F. Plumbing	118 N Wilder	Lubbock	TX	7941
Bob	Hanlan	Ann Arbor	6916 Morgan	Ann Arbor	MI	4810
Tim	Daniels	St. Louis	3133 Cornell	St. Louis	MO	6313
John	Moses		8265 Leticia	San Clemente	CA	9267
John	Fabian		3 Rose Hill	Woodstock	VT	0509
Ed	Ruth	Chicago	1580 N.	Chicago	IL	6063
Don	Harmo	Sudderth Video	415 Sudderth	Ruidoso	NM	8834

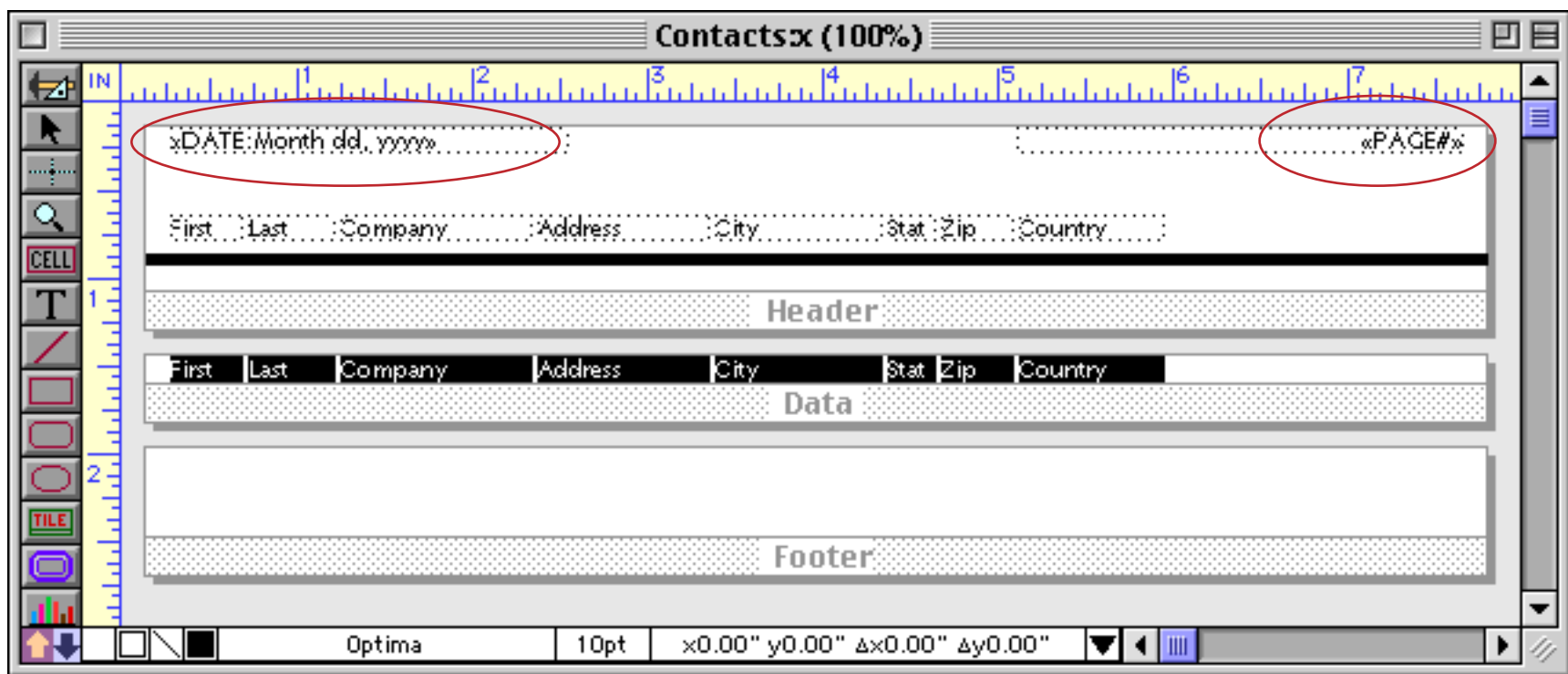
The QuickReport dialog also creates header and footer tiles. You can specify the height of the header and footer by specifying the top and bottom margins (in either inches or centimeters).



The QuickReport dialog can place page numbers and/or the date and time on the header and footer tiles. To set up the page numbers and date/time, press the **Set Up Titles** button. This button opens a small dialog with 16 checkboxes.



These checkboxes allows you to place any combination of page number, date, and time in up to four positions on the page: top left, top center, top right, and bottom center. Here is the form generated with the options shown above.

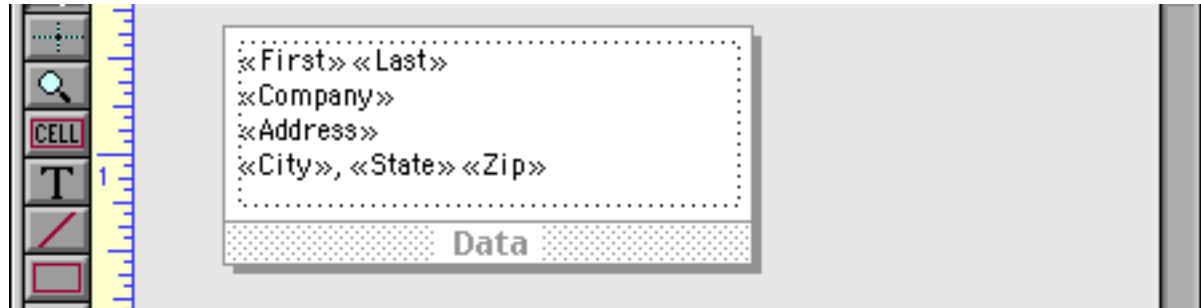


To learn more about printing the page number and date see "[Page Numbers](#)" on page 472 and "[Printing the Current Date and Time](#)" on page 473.

## Printing Multiple Column Reports

If the data tile is less than half the width of the page Panorama can print a two column report. If the data tile is less than 1/3 of the page width Panorama can print a three column report. Multiple column reports can be used to print mailing labels, or to print multiple column catalogs, lists, and directories.

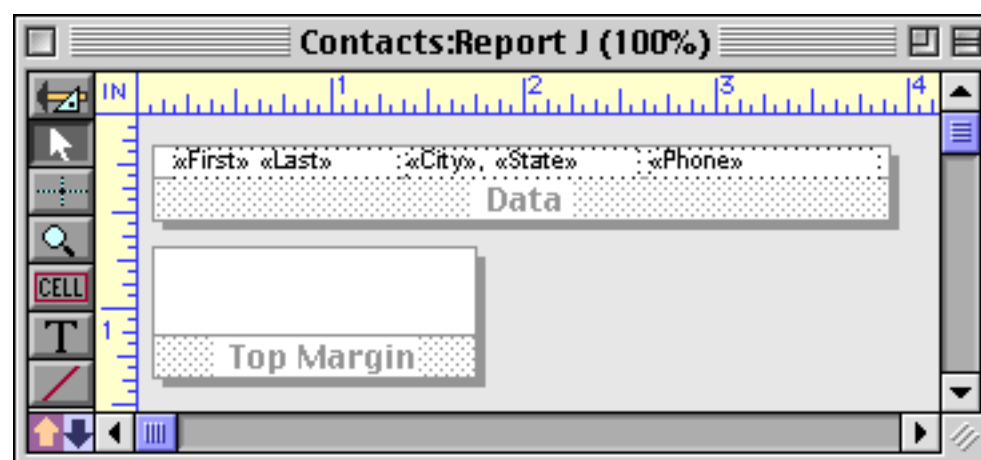
For example, the data tile on this form is just about 1/3 page.



When you print this form it will look like this.

John Smith Acme Widgets 12 Harmony Lane Suite 15	Don Harmon Sudderth Video 415 Sudderth Ruidoso, NM 88345	David Blair DB Printing 869 W. Temple Lenox, IA 50851
Susan Brown 783 Algonquin Newport Beach, CA 93459	Abe Fierstein Van Nuys Lumber 1571 Haskell Van Nuys, CA 91409	Keith Baker Northgate Video 552 Northgate Lindenhurst, IL 60046
Karen Wilson Evanston Lumber 498 Noyes Evanston, IL 60201	Randy Cross Randy's Appliances 133 Hunt Rd Chelsford, MA 01824	John Sloan 79 Danube Way Olympia Fields, IL 60461
Jim Nickle Jim's Appliances 14189 8th Newhall, CA 91321	Jeffrey Rodman 2 Cary Rd Chestnut Hill, MA 02167	Guy Porter St. Louis Lumber 8702 Pershing St. Louis, MO 63107

Here's another report where the Data tile is just less than half of the page width.

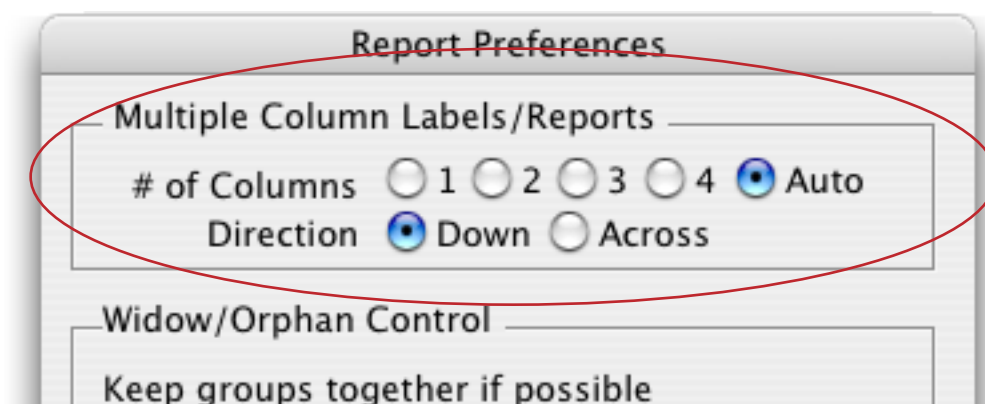


Panorama will print a two-up report, like this.

Keith Baker	Lindenhurst, IL		Jeannette Mulder	Irving, TX	
Nabil Basir	Armonk, NY		David Murray	Westport, CT	
John Bath	Mendota Heights, IL	(612) 461-1121	Jim Nickle	Newhall, CA	(805) 259-4093
Jack Beardsley	Toledo, OH		Logan Nourse	Palo Alto, CA	
Carl Berg	New Haven, CT	(203) 624-3367	Sam Pack	Inverness, IL	
Leslie Bianchi	Lexington, MA		Michael Paine	Pullman, WA	
Mary Bilbury	Beverly Hills, CA		David Peters	Concord, CA	
Joseph Bizzari	Westchester, IL		Charles Pierce	Midland, TX	
David Blair	Lenox, IA		Guy Porter	St. Louis, MO	
Al Bodner	Clifton Park, NY		Jim Pyle	Roseville, CA	
Jerry Boone	Traverse City, MI		Sari Rattner	Seattle, WA	
Jerry Bowen	Highland, CA		Bud Roble	Riverside, CA	
Yvonne Broach	Houston, TX		Jeffrey Rodman	Chestnut Hill, MA	
Susan Brown	Newport Beach, CA		Chuck Rouse	Hays, KS	
Tom Cane	Dublin, CA	(415) 833-8577	Janel Rundlett	Kansas City, KS	
David Cohn	Buffalo Grove, IL		Ed Ruth	Chicago, IL	
Michael Cox	Dallas, TX		Peter Schug	Bronx, NY	
Anne Crane	Grosse Pointe		Jules Silk	Cheltenham, PA	
Randy Cross	Chelsford, MA		Peter Silvers	New Orleans, LA	
Thomas Cupal	Ann Arbor, MI		John Sloan	Olympia Fields, IL	
Charles Dalbert	New York, NY		John Smith	Huntington Beach, IL	(999) 555-1234
Steve Dallas	Creve Coeur, MO	(314) 993-4251	Brian Smith	Hollister, CA	

### Controlling the Number of Columns

Panorama normally packs in as many columns as it can based on the width of the data tile. However, you can override the number of columns with the **Report Preferences** dialog in the Setup menu.



Picking **1**, **2**, **3**, or **4** forces the number of columns. Panorama will use the number of columns you specify even if the columns won't fit on the page. This is usually useful when the number of columns you want to fit almost fits on the page, but not quite. There's no point in printing four columns when only three will fit — the extra column will simply be invisible. Pick **Auto** to let Panorama choose the number of columns based on the width of the data tile.

## Printing Summary Information

In Chapter 10 you learned how to take a database and group it into categories and then summarize those categories. The summary information (totals, averages, etc.) appears in temporary new summary records that temporarily become part of the database, like this.

Date	CkNum	PayTo	Category	Debit
07/16/99	2185	Railroad Model Craftsma	Advertising	453.42
07/18/99	2199	New Direction	Advertising	112.48
07/19/99	2203	Model Railroader	Advertising	110.00
07/20/99	2205	Advertiser's Mailing Ser	Advertising	27.00
07/24/99	2206	Sir Speedy	Advertising	142.40
07/24/99	2209	Advertiser's Mailing Ser	Advertising	500.00
07/24/99	2211	Railroad Model Craftsma	Advertising	453.42
08/13/99	2227	Page One	Advertising	92.05
08/14/99	2237	Advertiser's Mailing Ser	Advertising	500.00
08/29/99	2257	Advertiser's Mailing Ser	Advertising	425.00
09/06/99	2266	Advertiser's Mailing Ser	Advertising	495.41
09/18/99	2271	Railroad Model Craftsma	Advertising	453.42
09/19/99	2283	Caboose Gazette	Advertising	1,990.10
09/26/99	2297	AC Label Company	Advertising	205.97
09/28/99	2298	Graphic Depot	Advertising	344.00
09/28/99	2299	Advertiser's Mailing Ser	Advertising	167.00
<b>Advertising</b>				<b>34,516.82</b>
02/01/99	1938	Unocal	Auto	182.59
02/09/99	1968	Unocal	Auto	57.62
03/16/99	2007	Unocal	Auto	33.32
05/24/99	2111	Unocal	Auto	119.05
07/16/99	2189	Unocal	Auto	38.11
07/24/99	2213	Unocal	Auto	34.44
08/20/99	2240	Unocal	Auto	89.91
<b>Auto</b>				<b>555.04</b>
01/08/99	1907	Northern Illinois Mold	Equipment Rental	96.05
02/09/99	1950	Pitney Bowes	Equipment Rental	73.14
04/23/99	2063	Pitney Bowes	Equipment Rental	79.69
05/24/99	2137	Pitney Bowes	Equipment Rental	25.75
05/24/99	2141	Pitney Bowes	Equipment Rental	79.69
08/21/99	2251	Pitney Bowes	Equipment Rental	198.00
08/21/99	2253	Pitney Bowes	Equipment Rental	79.69
<b>Equipment Rent:</b>				<b>632.01</b>
02/09/99	1952	GECC	Fixed Assets	428.39
05/02/99	2072	GECC	Fixed Assets	704.00
05/24/99	2112	GECC	Fixed Assets	74.00
06/14/99	2158	C M S	Fixed Assets	1,168.75
07/03/99	2175	GECC	Fixed Assets	250.00
07/18/99	2200	SSG LaserWorks	Fixed Assets	793.00
08/21/99	2243	GECC	Fixed Assets	725.00
09/18/99	2275	T.W. Bender Group	Fixed Assets	2,814.33
09/19/99	2280	GECC	Fixed Assets	352.00
09/26/99	2296	TesLabe	Fixed Assets	2,465.00
<b>Fixed Assets</b>				<b>9,774.47</b>

In an ordinary report these summary records are printed just like any other record – in other words, they are printed with the data tile. Here's an example of a typical form with a data tile.

July 11th, 2000 @ 10:47 PM <span style="float: right;">Page 6</span>					
Date	#	Pay To	Category	Amount	Balance
Table Header					
01/08/99	1907	Northern Illinois Mold	Equipment Rental	96.05	35,882.42
Data					



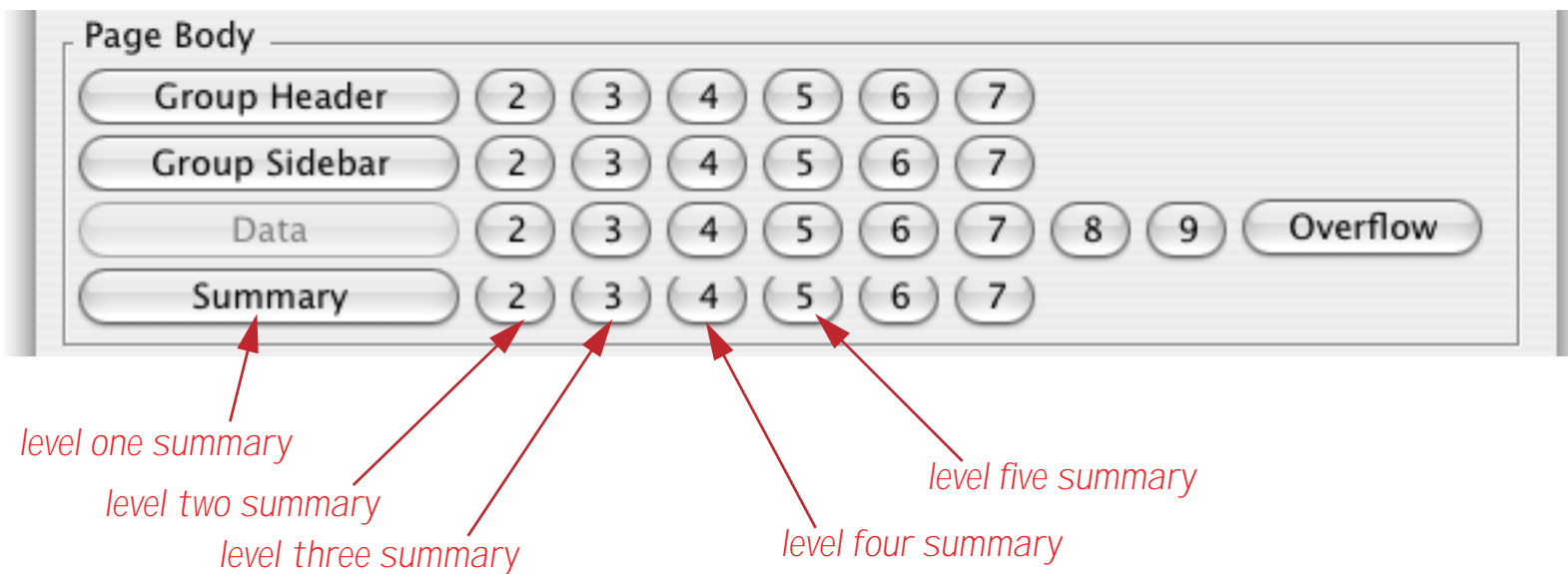
Here's a page from the printout, with the summary records circled in red. As you can see, the summary records are formatted exactly the same way as the regular data records.

<i>July 11th, 2000 @ 10:47 PM</i>					<i>Page 4</i>
Date	#	Pay To	Category	Amount	Balance
09/19/99	2285	Sherman Douglas Ins	Insurance	161.00	61,567.10
			Insurance	8,234.53	0.00
01/16/99	1910	Coudert Brothers, Attorney's	Legal Fees	223.52	35,193.10
02/09/99	1948	California Secretary Of State	Legal Fees	5.00	21,817.37
02/09/99	1949	City Of Caboose	Legal Fees	90.00	21,727.37
05/24/99	2121	Coudert Brothers/Attorney's	Legal Fees	799.55	49,799.79
06/05/99	2155	State Of California	Legal Fees	5.00	51,530.16
06/14/99	2159	Commonwealth Of	Legal Fees	10.00	58,056.21
07/16/99	2194	XL Corporation	Legal Fees	155.00	49,684.63
			Legal Fees	1,288.07	0.00
02/09/99	1964	Copierland	Maintenance	310.00	17,113.53
03/01/99	1986	Boyer & Ambrose Carpet	Maintenance	87.50	14,196.18
03/12/99	2002	Boyer & Ambrose Carpet	Maintenance	156.35	8,988.30
03/29/99	2037	Priority One Computers	Maintenance	496.40	21,349.77
03/29/99	2046	Executive Surveillance	Maintenance	132.00	19,370.06
04/24/99	2067	Boyer & Ambrose Carpet	Maintenance	132.00	31,336.38
05/08/99	2090	ServiceWorld	Maintenance	265.63	40,460.26
05/24/99	2114	E T S	Maintenance	49.00	51,641.24
05/24/99	2116	Sun Computers	Maintenance	282.00	51,091.39
05/24/99	2139	Pitney Bowes	Maintenance	140.00	43,941.69
05/29/99	2149	Sun Computers	Maintenance	276.00	52,038.36
06/05/99	2157	Sun Computers	Maintenance	101.25	51,408.96
06/21/99	2167	Dial One	Maintenance	267.13	56,654.66
07/16/99	2190	Executive Surveillance	Maintenance	168.00	50,229.71
07/24/99	2214	J & M Fire Extinguisher	Maintenance	38.19	41,952.07
08/08/99	2220	Newport Buidling &	Maintenance	120.00	49,321.22
08/21/99	2252	Vint Pest Control	Maintenance	120.00	54,797.48
09/18/99	2270	Computek Computer	Maintenance	100.00	69,777.93
09/18/99	2274	Boyer & Ambrose Carpet	Maintenance	432.54	68,392.35
			Maintenance	3,673.99	0.00
01/16/99	1911	Paramount Stationers	Office Supplies	105.84	35,087.26
01/22/99	1919	Cannon Astro	Office Supplies	145.72	24,693.23
01/25/99	1921	Nebs	Office Supplies	77.27	22,730.56
01/25/99	1922	Ramona Drinking Water	Office Supplies	98.10	22,632.46
02/01/99	1935	Copierland	Office Supplies	137.04	27,721.02

If you want the summary records to print differently from regular data records you'll need to add additional **summary report tiles** to your database (described in the next section). The use of these specialized tiles is strictly optional. If the report does not contain any summary tiles, Panorama will print summary records as if they were ordinary data records. For many reports this is just fine. If, however, you choose to use the specialized summary tiles, you can independently control how each summary level will be printed.

## Summary Tiles

A summary tile works just like a data tile, except that the summary tile is used only when Panorama prints a summary record. Since a database may have up to 7 levels of summary records, a report may contain up to 7 summary tiles. For example, you could use summary tiles to print the data in 12 point Helvetica, subtotals (level 1) in 14 point and the grand total (level 2) in 18 point. Summary tiles are created with the **Specialized Tile** configuration dialog.



When Panorama is about to print a summary record, it checks to see if the report has a summary tile for that level. If it does, that summary tile will be placed in the report instead of the data tile. If there is no corresponding summary tile, Panorama will look for summary tiles corresponding to a lower summary level. If it finds such a tile, Panorama will use it to print the summary record. If Panorama doesn't find a lower level summary tile, it gives up and uses the data tile to print the summary record.

The easiest way to create summary tiles is dragging to make a copy of the data tile (see "[Drag Duplicating](#)" on page 539). Once the new tile is created, you can double click on the tile's name to open the configuration dialog and convert it from a data tile into the appropriate level summary tile. Here is an example of a report with two summary tiles - one for subtotals and one for the grand total.

Date	#	Pay To	Category	Amount	Balance
Table Header					
			Auto	555.04	0.00
Data					
			<b>Auto</b>	<b>555.04</b>	<b>0.00</b>
Summary (1)					
<b>GRAND TOTAL</b>				<b>555.04</b>	<b>0.00</b>
Summary (2)					

Here is one page from the report that is printed by this form (assuming that the database has already been grouped by the [Category](#) field and totaled by the [Amount](#) field.

<i>July 11th, 2000 @ 10:55 PM</i>					<i>Page 4</i>
Date	#	Pay To	Category	Amount	Balance
08/21/99	2245	Maryland Casualty	Insurance	147.78	55,454.15
08/29/99	2260	Blue Cross Of Calif	Insurance	177.85	49,716.67
09/13/99	2269	Employers Health	Insurance	284.44	69,877.93
09/18/99	2272	Blue Cross Of Calif	Insurance	177.85	69,146.66
09/19/99	2285	Sherman Douglas Ins	Insurance	161.00	61,567.10
<b>Insurance</b>				<b>8,234.53</b>	<b>0.00</b>
01/16/99	1910	Coudert Brothers, Attorney's	Legal Fees	223.52	35,193.10
02/09/99	1948	California Secretary Of State	Legal Fees	5.00	21,817.37
02/09/99	1949	City Of Caboose	Legal Fees	90.00	21,727.37
05/24/99	2121	Coudert Brothers/Attorney's	Legal Fees	799.55	49,799.79
06/05/99	2155	State Of California	Legal Fees	5.00	51,530.16
06/14/99	2159	Commonwealth Of	Legal Fees	10.00	58,056.21
07/16/99	2194	XL Corporation	Legal Fees	155.00	49,684.63
<b>Legal Fees</b>				<b>1,288.07</b>	<b>0.00</b>
02/09/99	1964	Copierland	Maintenance	310.00	17,113.53
03/01/99	1986	Boyer & Ambrose Carpet	Maintenance	87.50	14,196.18
03/12/99	2002	Boyer & Ambrose Carpet	Maintenance	156.35	8,988.30
03/29/99	2037	Priority One Computers	Maintenance	496.40	21,349.77
03/29/99	2046	Executive Surveillance	Maintenance	132.00	19,370.06
04/24/99	2067	Boyer & Ambrose Carpet	Maintenance	132.00	31,336.38
05/08/99	2090	ServiceWorld	Maintenance	265.63	40,460.26
05/24/99	2114	E T S	Maintenance	49.00	51,641.24
05/24/99	2116	Sun Computers	Maintenance	282.00	51,091.39
05/24/99	2139	Pitney Bowes	Maintenance	140.00	43,941.69
05/29/99	2149	Sun Computers	Maintenance	276.00	52,038.36
06/05/99	2157	Sun Computers	Maintenance	101.25	51,408.96
06/21/99	2167	Dial One	Maintenance	267.13	56,654.66
07/16/99	2190	Executive Surveillance	Maintenance	168.00	50,229.71
07/24/99	2214	J & M Fire Extinguisher	Maintenance	38.19	41,952.07
08/08/99	2220	Newport Buidling &	Maintenance	120.00	49,321.22
08/21/99	2252	Vint Pest Control	Maintenance	120.00	54,797.48
09/18/99	2270	Computek Computer	Maintenance	100.00	69,777.93
09/18/99	2274	Boyer & Ambrose Carpet	Maintenance	432.54	68,392.35
<b>Maintenance</b>				<b>3,673.99</b>	<b>0.00</b>
01/16/99	1911	Paramount Stationers	Office Supplies	105.84	35,087.26
01/22/99	1919	Cannon Astro	Office Supplies	145.72	24,693.23
01/25/99	1921	Nebs	Office Supplies	77.27	22,730.56
01/25/99	1922	Ramona Drinking Water	Office Supplies	98.10	22,632.46

The last page of this report contains both level 1 (subtotal) and level 2 (in this case, grand total) summaries.

<i>July 11th, 2000 @ 11:08 PM</i>					<i>Page 16</i>
Date	#	Pay To	Category	Amount	Balance
02/09/99	1953	City Of Caboose	Utilities	9.64	21,151.17
02/09/99	1972	So. Calif. Gas Co.	Utilities	136.33	14,937.24
02/09/99	1973	S C E	Utilities	172.03	14,765.21
03/29/99	2041	City Of Caboose	Utilities	77.71	20,801.96
03/29/99	2042	So. Calif. Gas Co.	Utilities	217.32	20,584.64
03/29/99	2043	S C E	Utilities	89.46	20,495.18
05/07/99	2083	S C E	Utilities	96.26	41,775.05
05/07/99	2085	So. Calif. Gas Co.	Utilities	86.74	41,420.56
05/24/99	2117	So. Calif. Gas Co.	Utilities	134.99	50,956.40
05/24/99	2118	S C E	Utilities	97.00	50,859.40
05/24/99	2119	City Of Caboose	Utilities	114.77	50,744.63
06/05/99	2154	S C E	Utilities	157.31	51,535.16
06/14/99	2161	S C E	Utilities	56.27	57,897.11
07/24/99	2212	City Of Caboose	Utilities	98.52	42,024.70
08/13/99	2235	S C E	Utilities	86.53	45,764.17
09/19/99	2290	City Of Caboose	Utilities	103.15	60,793.57
09/19/99	2291	S C E	Utilities	81.13	60,712.44
09/19/99	2292	So. Calif. Gas Co.	Utilities	154.95	60,557.49
<b>Utilities</b>				<b>2,054.89</b>	<b>0.00</b>
<b>GRAND TOTAL</b>				<b>183,651</b>	<b>0.00</b>

This illustration shows how the report tiles at the end of this report are put together. Data records are printed with the Data tile, level 1 (subtotal) summary records are printed with the **Summary (1)** tile, and level 2 (grand total) summary records are printed with the **Summary (2)** tile.

05/24/99	2118	S C E	Utilities	97.00	50,859.40
05/24/99	2119	City Of Caboose	Utilities	114.77	50,744.63
06/05/99	2154	S C E	Utilities	157.31	51,535.16
06/14/99	2161	S C E	Utilities	56.27	57,897.11
07/24/99	2212	City Of Caboose	Utilities	98.52	42,024.70
08/13/99	2235	S C E	Utilities	86.53	45,764.17
09/19/99	2290	City Of Caboose	Utilities	103.15	60,793.57
09/19/99	2291	S C E	Utilities	81.13	60,712.44
09/19/99	2292	So. Calif. Gas Co.	Utilities	154.95	60,557.49
<b>Utilities</b>				<b>2,054.89</b>	<b>0.00</b>
<b>Summary (1)</b>				<b>GRAND TOTAL</b>	<b>183,651</b>
<b>Summary (2)</b>					<b>0.00</b>

### Printing Data Grouped by Month, Quarter or Year

When a database has been grouped by month, quarter or year, the dates should usually be formatted differently at each summary level. At the raw data level the entire date should be printed, while on the summary tiles only the month, quarter, or year should be printed.

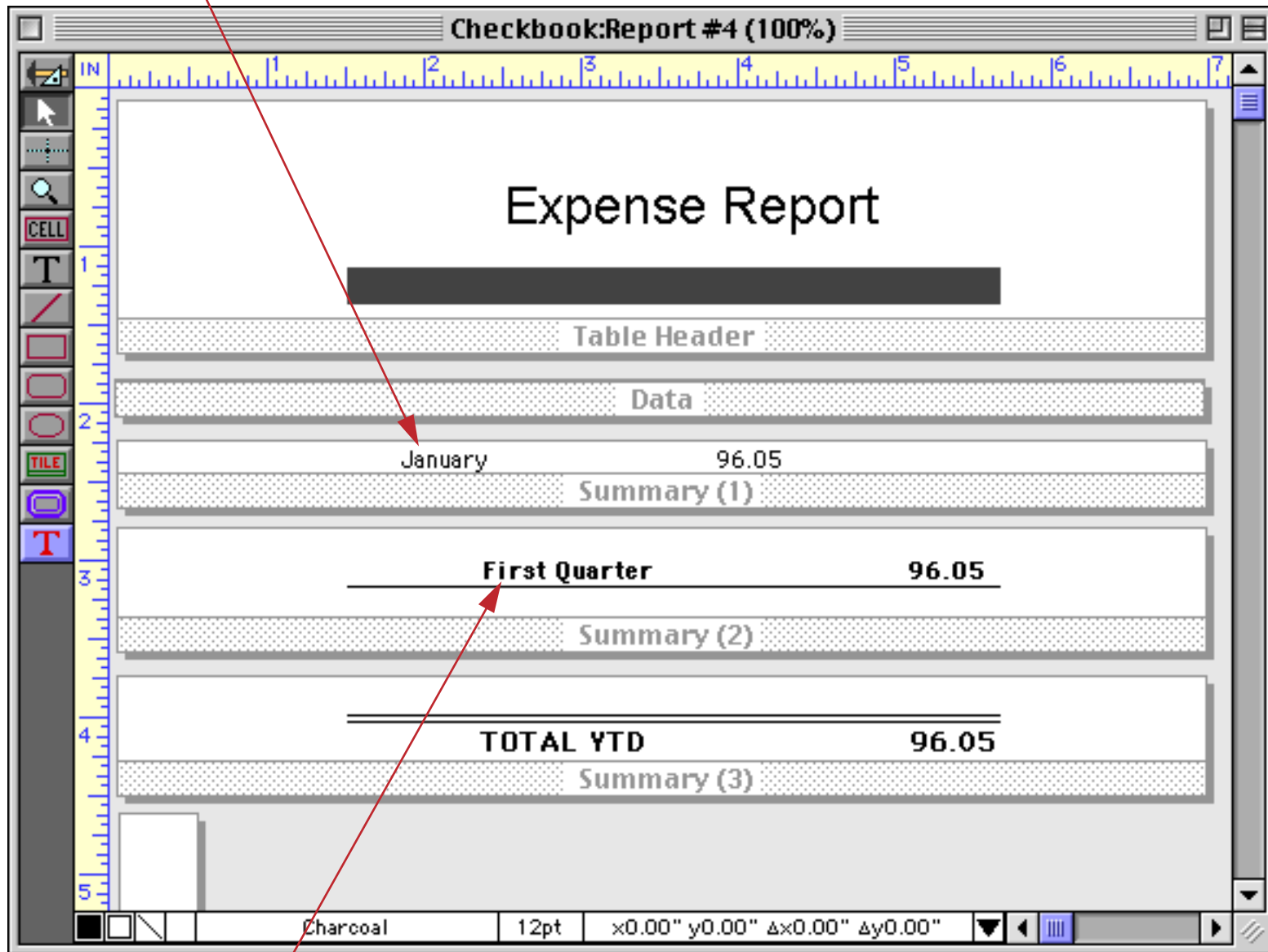
To format a Data Cell object to display only the month, quarter, or year, select the date cell object and choose the **Output Pattern** command in the Text Menu. This command sets the output pattern for just the individual selected object. The pattern for a month summary could be **Month yyyy**, **Mon-yy** or **mm-yy**. For a quarter summary use **qqyy** or **Qtr "Qtr" yy**. For a year use **yy** or **yyyy**.

To print only the month, quarter or year with an auto-wrap or Text Display object use the `datepattern()` function with one of the patterns mentioned in the previous paragraph.

Here's an example that uses Text Display SuperObjects to display the date.

**T** Text Display SuperObject™...

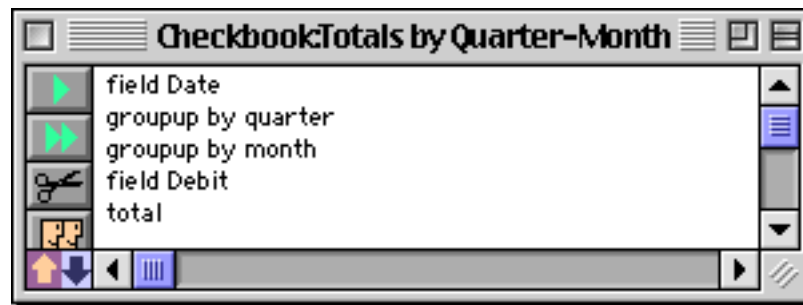
Formula: `datepattern(Date,"Month")`



**T** Text Display SuperObject™...

Formula: `datepattern(Date,"Quarter")+ " Quarter"`

To prepare this report for printing you must **Group Up** by Quarter on the **Date** field then **Group Up** by Month, then **Total** the **Debit** field. Here is a procedure that will prep the database for you (see Chapter 24).



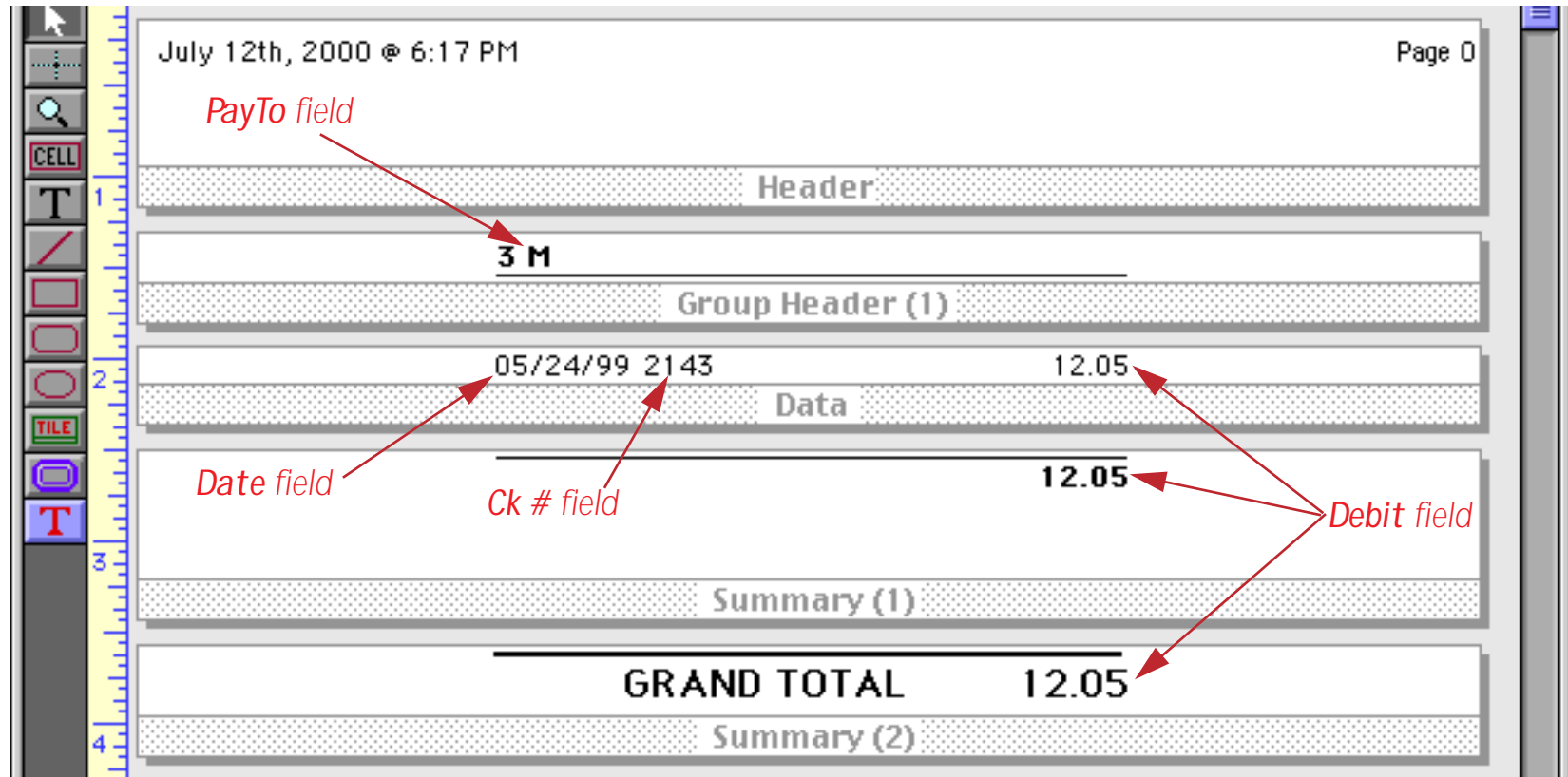
Here is the printed report.

<h2>Expense Report</h2>	
January	26,575.21
February	23,573.26
March	39,011.30
<b>First Quarter</b>	<b>89,159.77</b>
<hr/>	
April	5,852.73
May	27,659.93
June	12,742.03
<b>Second Quarter</b>	<b>46,254.69</b>
<hr/>	
July	18,860.65
August	14,842.86
September	14,533.25
<b>Third Quarter</b>	<b>48,236.76</b>
<hr/>	
<b>TOTAL YTD</b>	<b>183,651.22</b>



### Group Headers

Panorama normally prints a header at the top of each report page. The **Group Header** tiles allow an individual header to be printed at the top of each group in the report. (Of course, the database must already be arranged into groups.) The **Group Header** tile can be used to print a title at the top of each group and also to provide extra spacing between groups. Here's an example of a report with a group header. This example assumes that the database has been grouped by the **PayTo** field.



When it's printed this report looks like this.

July 12th, 2000 @ 6:17 PM		Page 1
<b>3 M</b>		
05/24/99	2143	12.05
		<b>12.05</b>
<b>A A A</b>		
03/29/99	2035	45.00
05/24/99	2148	128.65
		<b>173.65</b>
<b>A T &amp; T</b>		
03/29/99	2039	19.51
05/24/99	2129	19.51
05/24/99	2130	3.01
07/16/99	2196	7.67
08/13/99	2231	34.62
08/21/99	2241	34.62
08/21/99	2249	19.51
		<b>138.45</b>

This illustration shows how Panorama assembles the tiles to create the final report. At the beginning of each group it prints the group header, then the data tiles. At the end of the group it prints the summary tile.

July 12th, 2000 @ 6:17 PM		Page 1
<b>3 M</b>		← Header
05/24/99 2143	12.05	← Group Header (1)
		← Data
	<b>12.05</b>	← Summary (1)
<b>A A A</b>		← Group Header (1)
03/29/99 2035	45.00	← Data
05/24/99 2148	128.65	← Data
	<b>173.65</b>	← Summary (1)
<b>A T &amp; T</b>		← Group Header (1)
03/29/99 2039	19.51	← Data
05/24/99 2129	19.51	← Data
05/24/99 2130	3.01	← Data
07/16/99 2196	7.67	← Data
08/13/99 2231	34.62	← Data
08/21/99 2241	34.62	← Data
08/21/99 2249	19.51	← Data
	<b>138.45</b>	← Summary (1)
<b>Summary (1)</b>		

Sometimes two groups start at the same spot. For example, in a database grouped by **State** and **City**, the start of the California group may also be the start of the Alameda group. In this case the higher level wins, and Panorama will print the group header for California. Because of this, the higher level summary headers should include any graphics or data needed for each lower level, as shown in this illustration.

*City field is printed on both Group Header (1) and Group Header (2)*

Conference Registration: Attendees by Location (100%)	
replacemultiple(State, "CA, CT, DC, FL, GA, IL, MA, NJ, NY, VA", "CALIFORNIA, CONNETICUT, DISTRICT:	
«City»	Group Header (2)
«City»	Group Header (1)
«First Name» «Last Name», «Company Name»	Data
	Summary (1)

Here's the finished report,

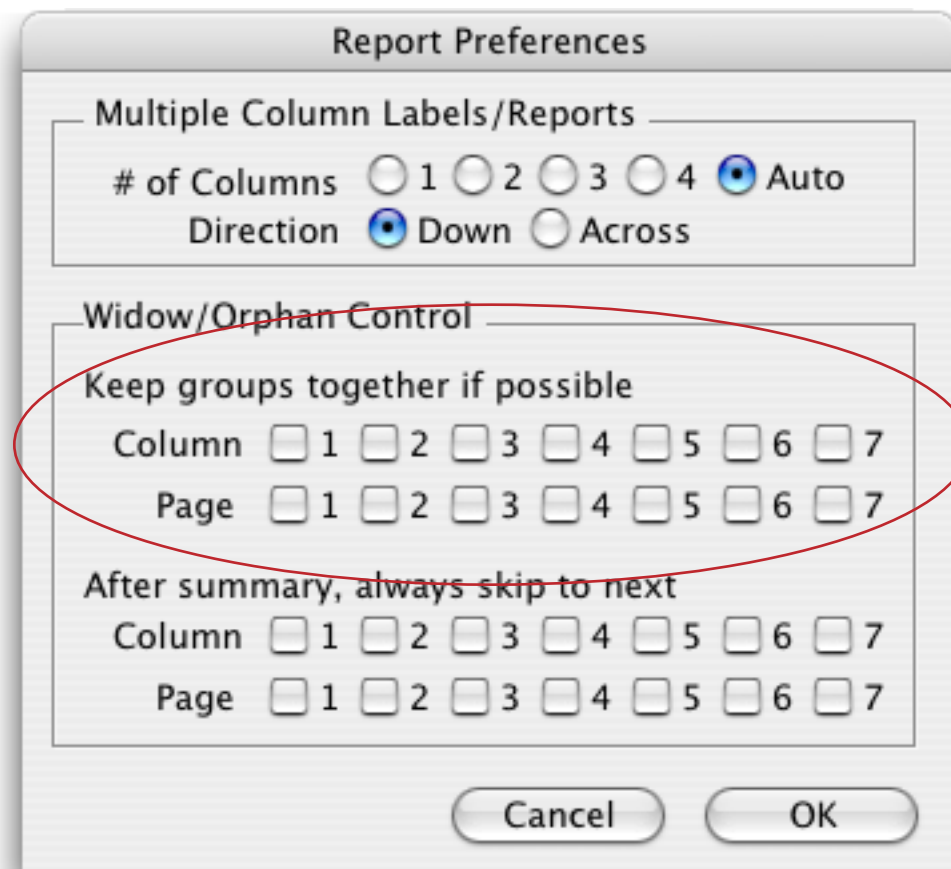
<b>CALIFORNIA</b>
<b>Alameda</b>
Jim Abrahms, International Transportation Resources
<b>Berkeley</b>
Robert Sophie, Alvarado, Johnson, & Wright
Nancy Alexander, JPSA
<b>Burbank</b>
Keith Jacobs, Bath Co.
<b>Canoga Park</b>
Ken Taylor, Saticoy Litho
<b>Compton</b>
Steve Prendergass, Pacific Photo Labs
<b>Costa Mesa</b>
Joyce Ferrell, Church Aviation
<b>Cupertino</b>
Christy Alpert, Signal Research
Arthur Clairmont, South Coast Office Products
Harold Cobb, Cobb Associates
Sherry Grossman, Pablo Distribution
Peter Parks, Hamilton Press
Michelle Adams, Sceptre
<b>Emeryville</b>
Cynthia Knight, TBS Contracting, Inc.

Here's how Panorama assembles the tiles to create this report.



## Keeping a Group Together on a Column or Page

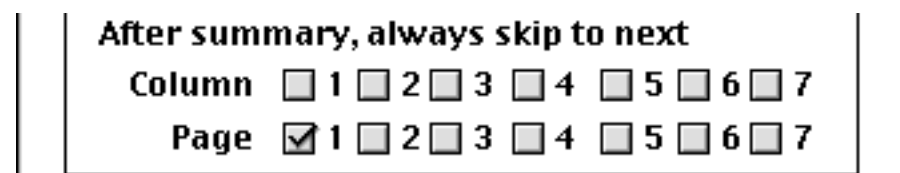
If you wish, Panorama can automatically make sure that groups are not divided across column or page boundaries. This can make a report easier to read and much more attractive. (Of course it can only do this for groups that are small enough to fit on a single column or page.) This feature is called widow/orphan control and is available in the **Report Preferences** dialog in the Setup menu.



Select the largest group level you want to keep together on a column or page (or both).

## Starting a Group on a New Column or Page

If you wish, Panorama can automatically skip to a new column or a new page at the beginning of each new group. To do this, choose **Report Preferences** from the Setup Menu, then choose the summary levels that triggers skipping to a new column or page.





# Chapter 22: Labels



Two of the most common jobs for a database program are printing mailing labels and form letters. Mailing labels can be tricky because the printing must line up with the pre-cut labels. This chapter describes tips and techniques that can help take the “trial and error” out of printing labels and form letters.

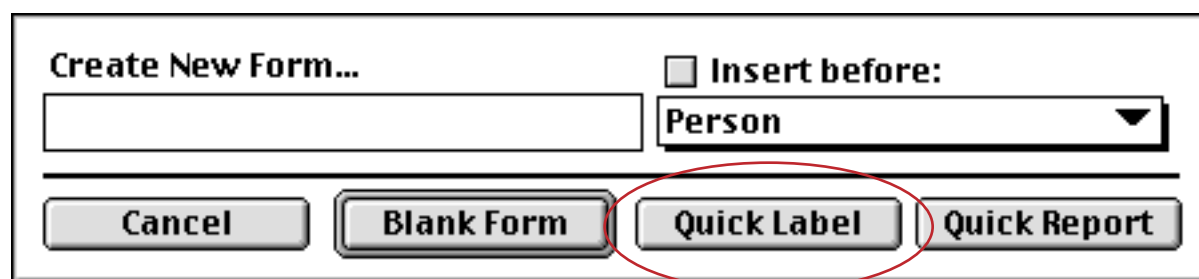
## Label Fundamentals

Labels are printed using report tiles in a form (see Chapter 21 for an introduction to report tiles). Many labels only require a single tile: the data tile. This tile should be the same size as the size of the label plus the gap between labels. In some cases you may also need margin tiles.

To actually print the names and addresses you will usually use an auto-wrap text object or a Text Display SuperObject. Either of these types of objects allow data, text, and punctuation to be mixed in the label.

## The QuickLabel Dialog

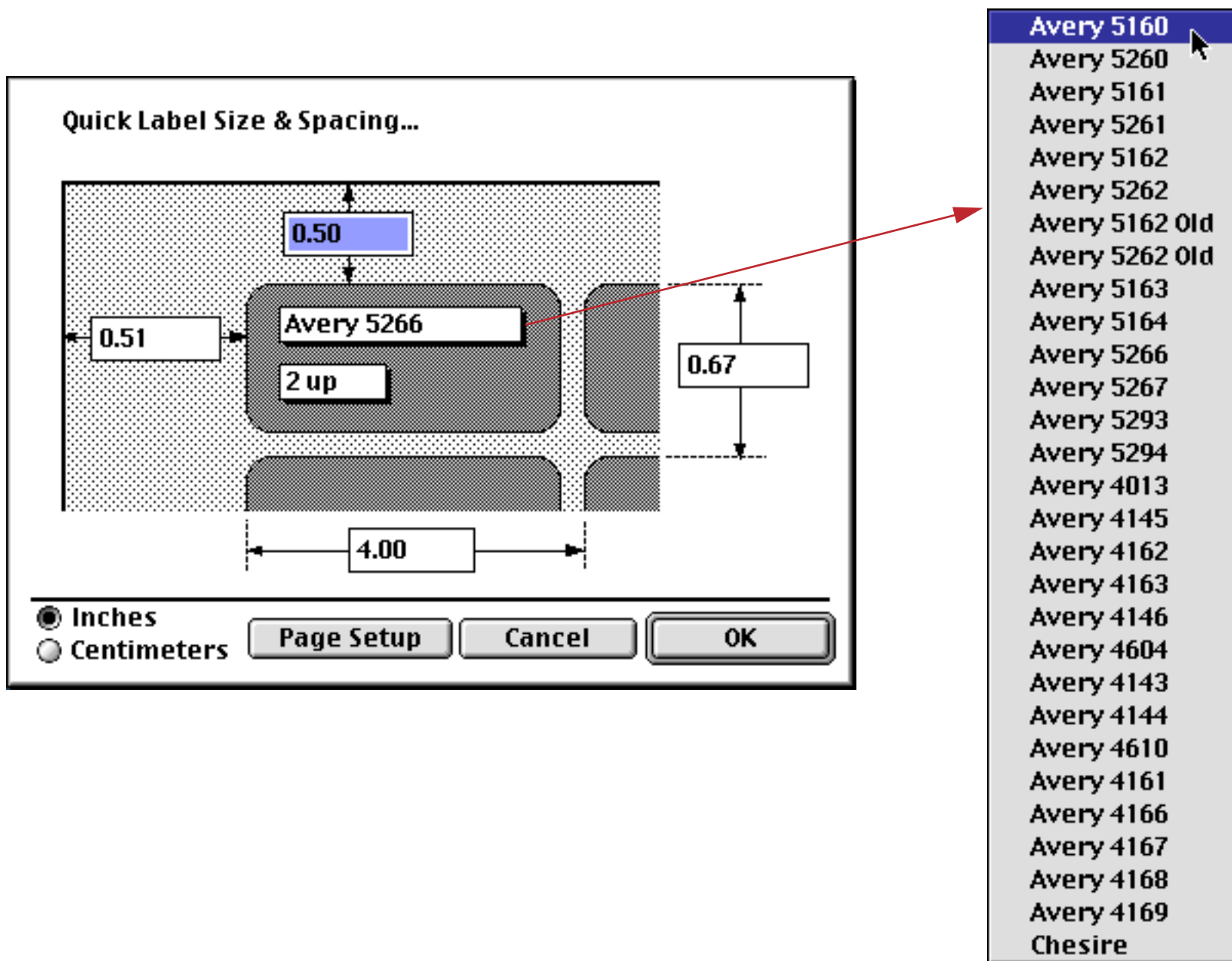
When you create a new form, Panorama gives you the option of creating a blank form or automatically creating a label or report.



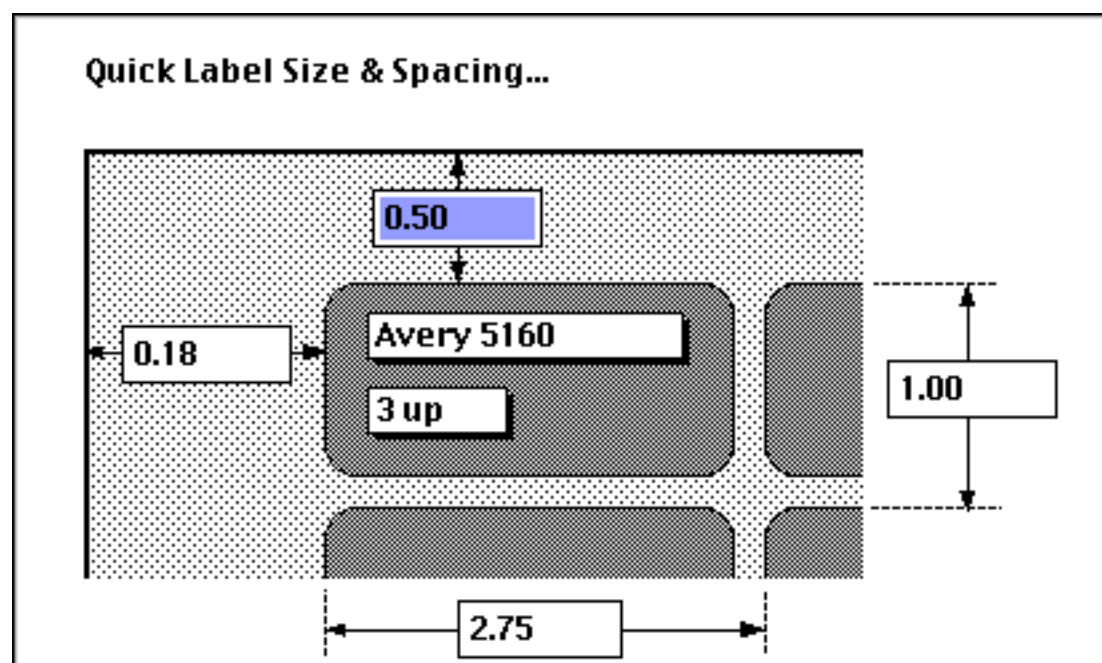
The **QuickLabel** button opens a dialog that can do most or all of the work of setting up a label for you. To use the QuickLabel dialog, select **New Form** from the **View** menu, give the new form a name, and then click on the **QuickLabel** button.



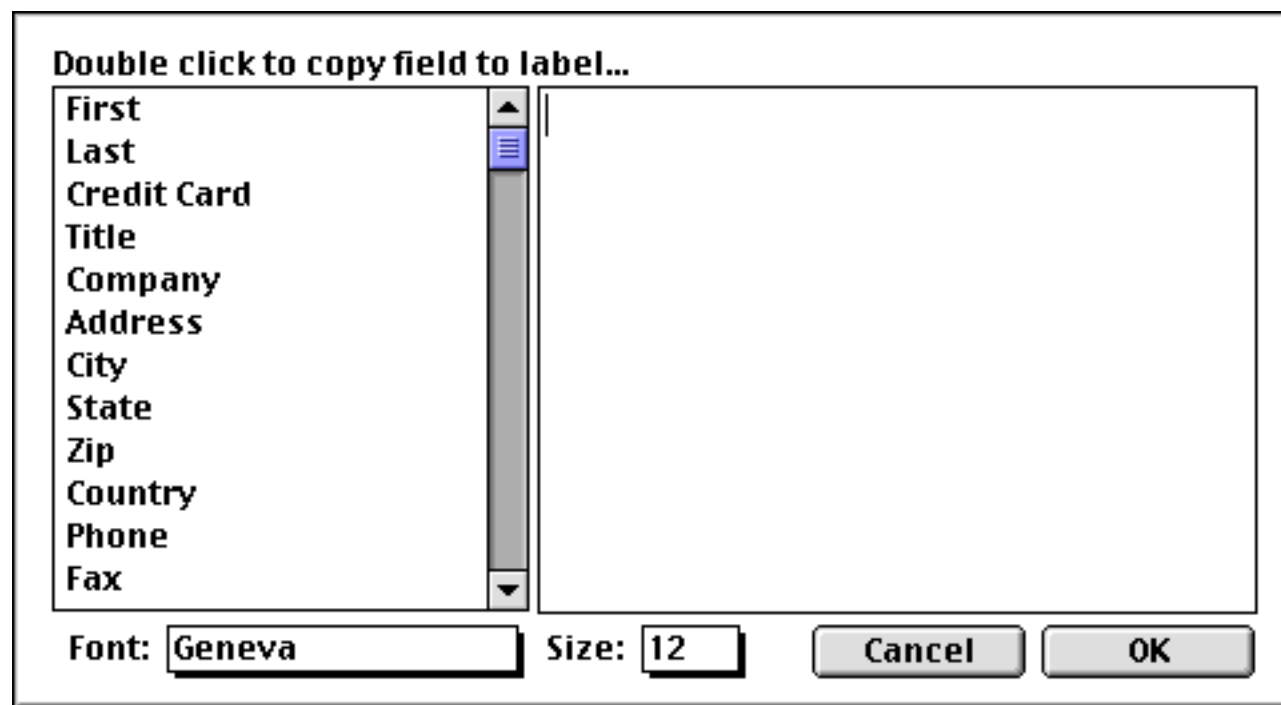
The QuickLabel dialog is really a two part dialog. The first part allows you to define the dimensions of the label. The dialog shows a picture of a label with dimensions around it. You can enter the dimensions manually, or you can pick a pre-defined label size using the pop-up menus inside the label. The top pop-up menu allows you to choose from popular label sizes. The bottom pop-up menu allows you to choose whether the label should be printed 1, 2, 3, or 4 up.



The most popular style of label is Avery 5160, which prints 30 labels on a sheet (10 rows by 3 columns).



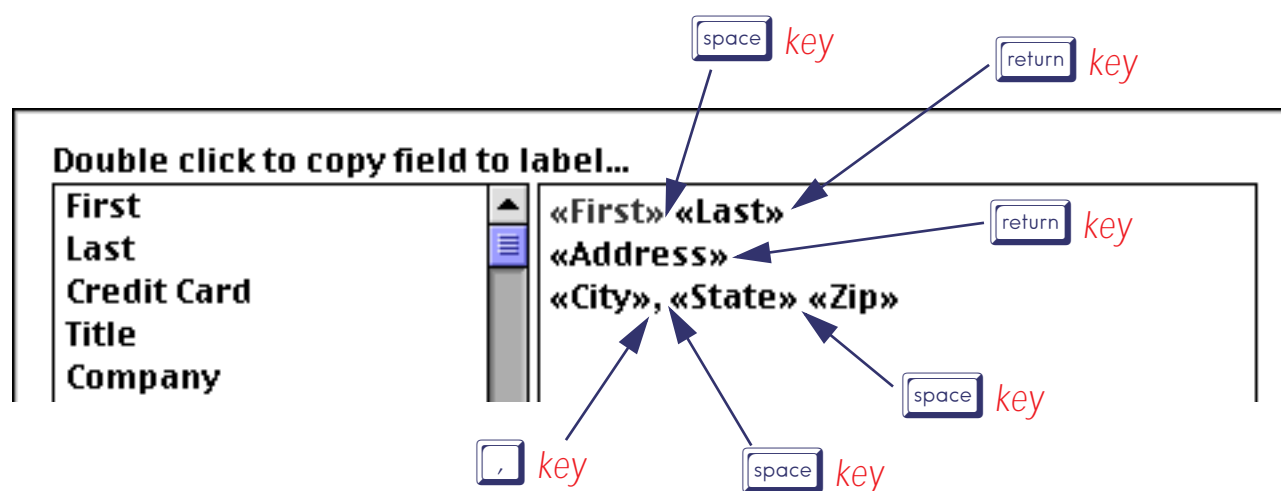
The second part of the dialog allows you to specify the data that will be printed on each label (usually name and address). The left side of the dialog lists all the database fields, while the right hand side contains an image of the actual label.



To copy a field into the label, double click on the field name. Double clicking types the field name into the area on the right, surrounded by « » characters.

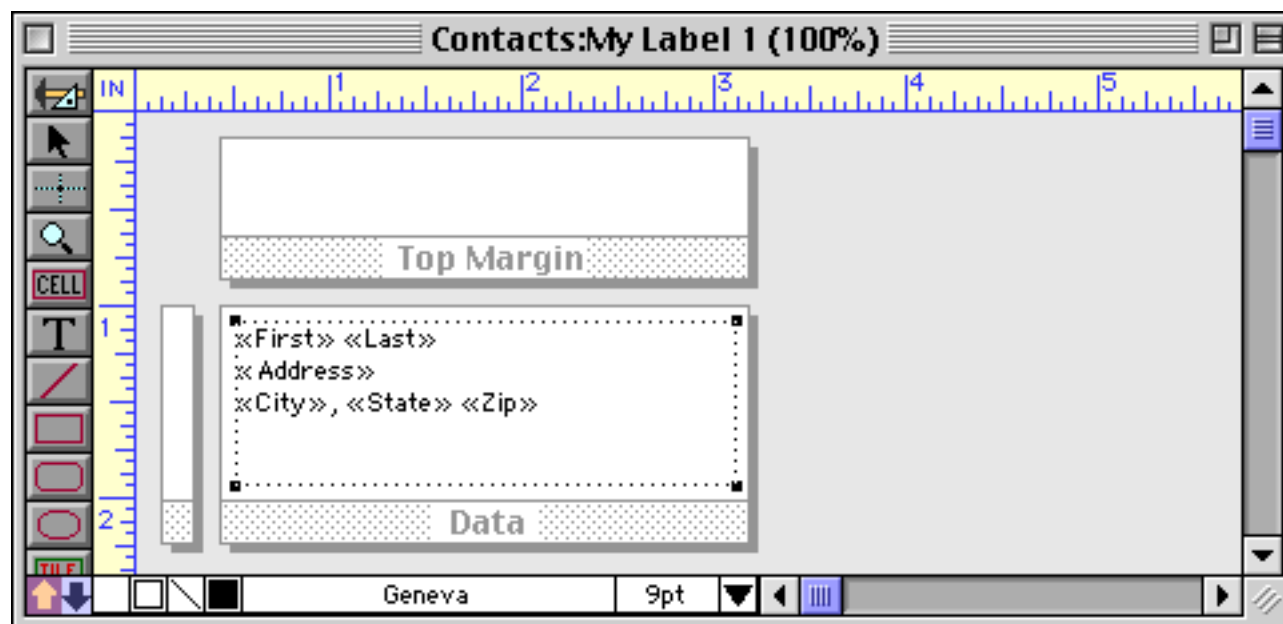


To add a space or punctuation, simply press the appropriate key. To start a new line, press the **Return** key.



You can also edit the text in the label like any other text—click to create an insertion point, drag to select one or more characters, or type to enter new text.

When the content of the label is finished, press the **OK** button. The dialog will automatically create one or more tiles and an auto-wrap text object.



The label is now finished and can be printed or modified further. Here's what the printed labels look like.

Keith Baker 552 Northgate Lindenhurst, IL 60046	Jerry Boone 6125 Park Drive Traverse City, MI 49684	Charles Dalbert 171 Broadway New York, NY 10003
Nabil Basir 12 Upland Lane Armonk, NY 10504	Jerry Bowen 2847 Peacock Highland, CA 92346	Steve Dallas 1 Chaminade Creve Coeur, MO 63141
John Bath 8864 Ave Mendota Heights, MN 55118	Yvonne Broach 9330 Poitiers Houston, TX 77071	Herb Dang 206 Phelps St San Francisco, CA 94124
Jack Beardsley 4964 Pelham Toledo, OH 43606	Susan Brown 783 Algonquin Newport Beach, CA 93459	Tim Daniels 3133 Cornell St. Louis, MO 63130

## Printing Labels on Sheets

Labels come in two styles — sheets and rolls. Most of today's laser and inkjet printers work only with sheets.

Some sheets contain labels that go right up to the edge of the sheet. Depending on your printer you may not be able to print all the way to the edge. You may be able to use the **Page Setup** command to make it possible to print closer to the edge of the sheet. Consult the documentation that came with your printer to find out if this is possible.

The height of the data tile should equal the distance from the top of one label to the top of the next label, while the width of the label should equal the distance from the left edge of one label to the left edge of the next label.

### Printing 3 by 10 1" Labels (Avery 5160)

The most common type of label sheet contains 30 1" high labels in 3 columns of 10 labels (Avery 5160). The Avery 5160 labels are spaced 1" vertically and 2.75" horizontally. There is a 1/2 inch top margin above the labels, and a 0.18 inch left margin.

You can use the QuickLabel dialog to set up the tiles for this type of label (see "[The QuickLabel Dialog](#)" on page 495). If the edges of the labels are cut off when you print, use the Page Setup dialog to adjust your printer settings to allow Panorama to print as close to the edge of the sheet as possible. Consult the documentation that came with your printer to find out how to set the printer margins (if possible).

### Aligning Labels on the Sheet

If the names and addresses are not properly aligned on the labels, you can create Left Margin and Top Margin tiles. (The QuickLabel dialog automatically creates these tiles for you.) To shift the text up or down, change the height of the **Top Margin** tile. To shift the text left or right change the width of the **Left Margin** tile (see Chapter 21 for more information on these tiles). You can make precision adjustments to the size of a tile with the **Dimensions** dialog or by nudging with the arrow keys.

Note: If you do not specify a left or top margin, Panorama will use the default printer margins.

### Printer Inaccuracy and Vertical Creep

If you try to print very small labels (less than 1/2 inch high), or if you try to print to the very edge of each label (for example a border around the label), you may run into problems due to printer inaccuracy. Due to mechanical tolerances, most printers are only accurate to about 1/8 inch over a full 11 inch page. Even if the labels line up properly at the top of the sheet, they may gradually creep out of place towards the bottom of the page. Unfortunately this problem is difficult to correct.

You may be able to reduce or eliminate the creep by adjusting for the inaccuracy in your printer. To try to compensate for this problem you can adjust the height of the data tile in 1/8th pixel increments. To do this, use the **Form Preferences** dialog to reduce the nudge increment to 1/8th pixel, then use the up/down arrows to nudge the size of the tile. Remember this adjustment may be different for different printers, or may even change for the same printer at different times.

A simpler solution is to not print anything less than 1/8th inch from the edge of a label. This leaves enough tolerance for the inaccuracy of the printer.



# Chapter 23: Formulas



The result we proceed to divide, as you see,  
by Nine Hundred and Ninety Two:  
Then subtract seventeen, and the answer must be  
Exactly and perfectly true.

- Lewis Carrol, The Hunting of the Snark

---

Panorama's primary job is storing and retrieving data. The primary job of formulas is to combine and manipulate data, both numeric and textual. Using formulas Panorama can automatically add up all the items in an invoice and calculate the sales tax. Using formulas Panorama can automatically divide all the names in a database into separate first and last names, or convert all the company names in a database to upper case. Using formulas Panorama can automatically look up the price of an item in inventory, or check the quantity on hand, or look up and display the items on a customer's previous order. As you can see, you'll need to learn how to use formulas to get the most from your Panorama investment. Fortunately, formulas are easy to learn and use (especially the most common mathematical formulas like totals, taxes and percentages). (However, we have to admit that sometimes formulas can be frustrating because they are very picky. If you get one little detail wrong, the formula won't work correctly. This isn't just a problem with Panorama, but with virtually any computer program that uses formulas. To help ease the potential frustration factor Panorama has some wizards you'll learn about later in this chapter that can help you build error free formulas.)

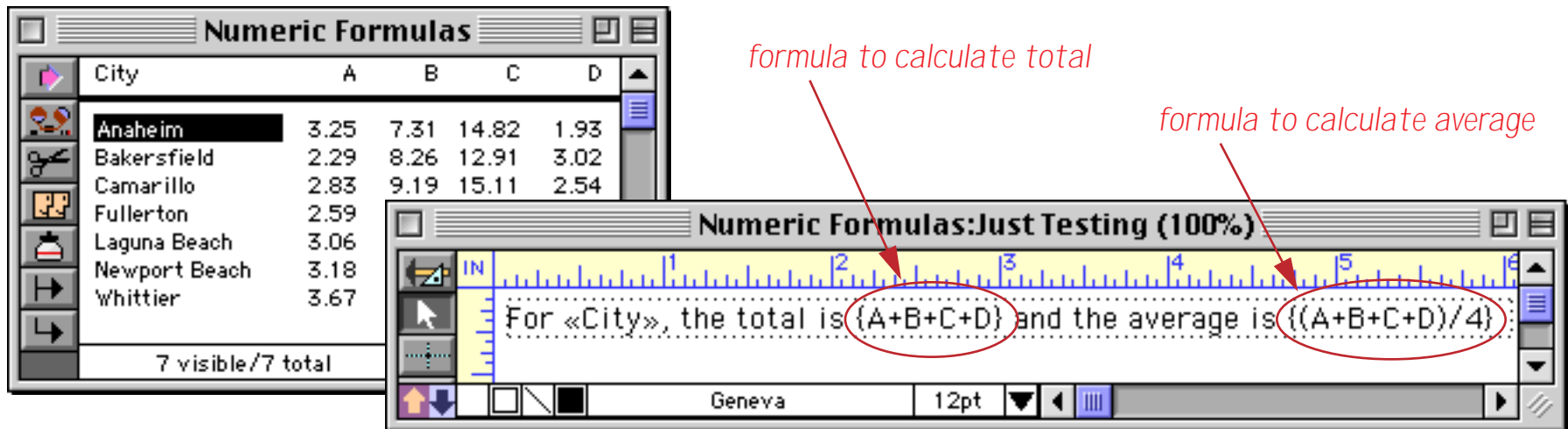
## Formulas In Action

Formulas are a general purpose tool that Panorama can use in a variety of different situations. You can display or print the result of a formula, use a formula to modify the database, or use a formula to help locate information in the database. The next few sections demonstrate each of these techniques.

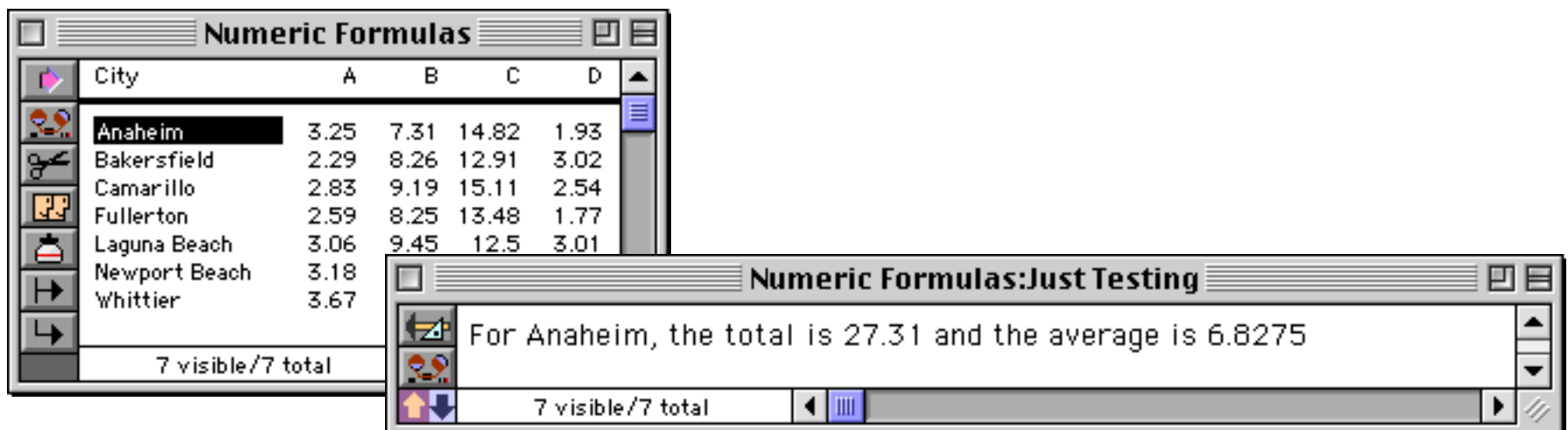


## Displaying/Printing A Formula

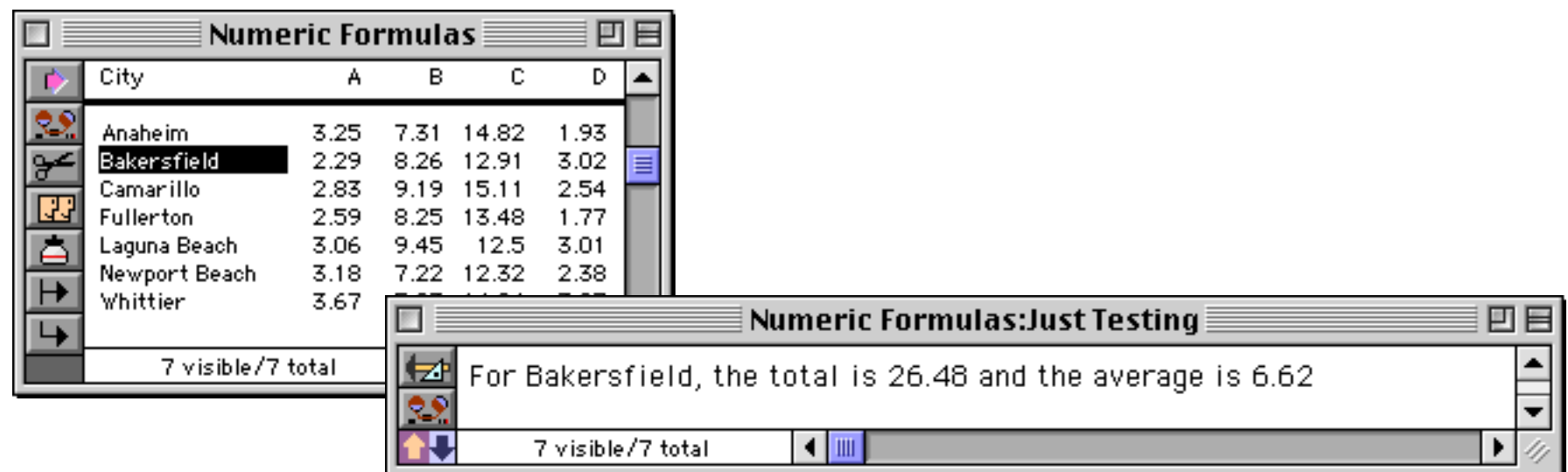
A formula can be displayed or printed anywhere on a form with an auto-wrap text object or Text Display SuperObject. For example, consider the database shown below. The auto-wrap text object contains two formulas, one which calculates the total of the four columns (A, B, C and D) and one which calculates the average.



When the form is switched to Data Access Mode, Panorama calculates the formula results and displays them.



When a formula is used this way the results are not stored anywhere in the database, they are simply calculated on the fly and displayed or printed, then thrown away. If you switch to a different record Panorama will calculate and display the new values.



Using the same technique you can even print a report using formulas calculated on the fly. Panorama's Flash Art feature allows a formula result to be displayed visually (see Chapter 16).

## Storing Formula Results in the Database

Sometimes you'll want to actually store the result of a formula in the database itself. You can do this manually after data has already been entered, or automatically as data is entered or changed. To illustrate these two techniques we'll add two new columns to the example database used in the last section, **Total** and **Avg**.

City	A	B	C	D	Total	Avg
Anaheim	3.25	7.31	14.82	1.93		
Bakersfield	2.29	8.26	12.91	3.02		
Camarillo	2.83	9.19	15.11	2.54		
Fullerton	2.59	8.25	13.48	1.77		
Laguna Beach	3.06	9.45	12.5	3.01		
Newport Beach	3.18	7.22	12.32	2.38		
Whittier	3.67	5.23	14.24	3.27		

To calculate the values for these new fields we need to use the **Formula Fill** command. To calculate the total, first click anywhere in the appropriate column.

City	A	B	C	D	Total	Avg
Anaheim	3.25	7.31	14.82	1.93		
Bakersfield	2.29	8.26	12.91	3.02		
Camarillo	2.83	9.19	15.11	2.54		
Fullerton	2.59	8.25	13.48	1.77		

Now choose **Formula Fill** from the Math menu, and enter the formula for calculating the total.

Enter the formula:

A+B+C+D

Cancel OK

When you press **OK** Panorama will calculate and store the value for every selected record in the database. In this case it performs seven calculations and stores seven values.

City	A	B	C	D	Total	Avg
Anaheim	3.25	7.31	14.82	1.93	27.31	
Bakersfield	2.29	8.26	12.91	3.02	26.48	
Camarillo	2.83	9.19	15.11	2.54	29.67	
Fullerton	2.59	8.25	13.48	1.77	26.09	
Laguna Beach	3.06	9.45	12.5	3.01	28.02	
Newport Beach	3.18	7.22	12.32	2.38	25.10	
Whittier	3.67	5.23	14.24	3.27	26.41	

Repeat the same process for the average, but of course with a different formula.

Enter the formula:

(A+B+C+D)/4

Cancel OK

Here's the end result.

City	A	B	C	D	Total	Avg
Anaheim	3.25	7.31	14.82	1.93	27.31	6.83
Bakersfield	2.29	8.26	12.91	3.02	26.48	6.62
Camarillo	2.83	9.19	15.11	2.54	29.67	7.42
Fullerton	2.59	8.25	13.48	1.77	26.09	6.52
Laguna Beach	3.06	9.45	12.5	3.01	28.02	7.00
Newport Beach	3.18	7.22	12.32	2.38	25.10	6.27
Whittier	3.67	5.23	14.24	3.27	26.41	6.60

Once the **Formula Fill** calculation is finished the formula is forgotten and the numbers are simply stored in the database just like a number that has been typed in. You can even manually edit a value to override the result of the formula calculation.

City	A	B	C	D	Total	Avg
Anaheim	3.25	7.31	14.82	1.93	27.31	6.83
Bakersfield	2.29	8.26	12.91	3.02	26.48	6.62
Camarillo	2.83	9.19	15.11	2.54	29.67	7.42
Fullerton	2.59	8.25	13.48	1.77	26.09	6.52
Laguna Beach	3.06	9.45	12.5	3.01	28.02	7.00
Newport Beach	3.18	7.22	12.32	2.38	25.7	6.27
Whittier	3.67	5.23	14.24	3.27	26.41	6.60

7 visible / 7 total

Since the **Formula Fill** formula is forgotten as soon as it is complete, Panorama does not update the values if the original numbers (in this case A, B, C or D) change or if new records are added to the database. If you want values stored in the database to update automatically as the database is updated you must set up automatic calculations in the design sheet. Here's the design sheet for our example updated to automatically calculate the total and average.

Field Name	Type	Dir	Align	Out	Inp	Range	Choi	Link	Clair	Tab	Cap	Dup	Def	Equation	Reac	Writ	Wid	Notes
City	Text	O	Left			Any		Off	Off	Off	Yes				0	0	11	
A	Nume	F	Right			Any		Off	Off	Off	Yes				0	0	4	
B	Nume	F	Right			Any		Off	Off	Off	Yes				0	0	4	
C	Nume	F	Right			Any		Off	Off	Off	Yes				0	0	4	
D	Nume	F	Right			Any		Off	Off	Off	Yes				0	0	4	
Total	Nume	F	Right	#,.	*	Any		Off	Off	Off	Yes		A+B+C+D		0	0	4	
Avg	Nume	F	Right	#,.	*	Any		Off	Off	Off	Yes		(A+B+C+D)/4		0	0	4	

*formula to calculate total*

*formula to calculate average*

To activate these formulas you need to create a new generation for this database (see Chapter 5). Once you've done this you can start entering or updating information. In this illustration a new record has been added (**Diamond Bar**) and the first number typed in (but not entered into the database yet).

Numeric Formulas							
City	A	B	C	D	Total	Avg	
Anaheim	3.25	7.31	14.82	1.93	27.31	6.83	
Bakersfield	2.29	8.26	12.91	3.02	26.48	6.62	
Camarillo	2.83	9.19	15.11	2.54	29.67	7.42	
Diamond Bar	3.13						
Fullerton	2.59	8.25	13.48	1.77	26.09	6.52	
Laguna Beach	3.06	9.45	12.5	3.01	28.02	7.00	

As soon as the data is entered by pressing the **Tab** (or **Enter**) keys the formulas update the **Total** and **Avg** fields.

Numeric Formulas							
City	A	B	C	D	Total	Avg	
Anaheim	3.25	7.31	14.82	1.93	27.31	6.83	
Bakersfield	2.29	8.26	12.91	3.02	26.48	6.62	
Camarillo	2.83	9.19	15.11	2.54	29.67	7.42	
Diamond Bar	3.13				3.13	0.78	
Fullerton	2.59	8.25	13.48	1.77	26.09	6.52	
Laguna Beach	3.06	9.45	12.5	3.01	28.02	7.00	
Newport Beach	3.18	7.22	12.32	2.38	25.70	6.27	
Whittier	3.67	5.23	14.24	3.27	26.41	6.60	

*formulas in design sheet update fields as data is entered*

As more data is entered the **Total** and **Avg** fields are updated instantaneously.

Bakersfield	2.29	8.26	12.91	3.02	26.48	6.62	
Camarillo	2.83	9.19	15.11	2.54	29.67	7.42	
Diamond Bar	3.13	7.81			10.94	2.73	
Fullerton	2.59	8.25	13.48	1.77	26.09	6.52	
Laguna Beach	3.06	9.45	12.5	3.01	28.02	7.00	
Newport Beach	3.18	7.22	12.32	2.38	25.70	6.27	

The **Total** and **Avg** fields will be updated any time the **A**, **B**, **C** or **D** fields are modified.

### Using a Formula to Locate/Select Information

The **Formula Find/Select** command and the Live Clairvoyance wizard (Chapter 9) allow you to select data based on a formula. This allows you to make selections on data that is not directly stored in the database. For example, suppose you want to select all records with an average greater than **6.8**, but without actually storing the average in the database. Here's the database.

Numeric Formulas				
City	A	B	C	D
Anaheim	3.25	7.31	14.82	1.93
Bakersfield	2.29	8.26	12.91	3.02
Camarillo	2.83	9.19	15.11	2.54
Diamond Bar	3.13	7.81	13.19	3.11
Fullerton	2.59	8.25	13.48	1.77
Laguna Beach	3.06	9.45	12.5	3.01
Newport Beach	3.18	7.22	12.32	2.38
Whittier	3.67	5.23	14.24	3.27

Choose the **Formula Find/Select** command and enter the formula. This formula calculates the average and then compares the average to **6.8**.



When the **Select** button is pressed records with averages above the threshold are selected.

City	A	B	C	D
Anaheim	3.25	7.31	14.82	1.93
Camarillo	2.83	9.19	15.11	2.54
Diamond Bar	3.13	7.81	13.19	3.11
Laguna Beach	3.06	9.45	12.5	3.01

4 visible / 8 total

Hey - I cheated (sort of)! This database already has the averages stored in the database. I can expand the window to double check that I really selected only records with averages over **6.8**.

City	A	B	C	D	Total	Avg
Anaheim	3.25	7.31	14.82	1.93	27.31	6.83
Camarillo	2.83	9.19	15.11	2.54	29.67	7.42
Diamond Bar	3.13	7.81	13.19	3.11	27.24	6.81
Laguna Beach	3.06	9.45	12.5	3.01	28.02	7.00

4 visible / 8 total

Do you forgive me? Anyway, the point is that the selection can be made even if the average is not stored in the database. Here's another example. This formula will select every record where the city name is longer than 10 characters (**11**, **12**, **13**, etc.)



Press **Select** to see only the records with long city names.

City	A	B	C	D	Total	Avg
Bakersfield	2.29	8.26	12.91	3.02	26.48	6.62
Diamond Bar	3.13	7.81	13.19	3.11	27.24	6.81
Laguna Beach	3.06	9.45	12.5	3.01	28.02	7.00
Newport Beach	3.18	7.22	12.32	2.38	25.70	6.27

With a bit of ingenuity you can almost come up with a formula to locate or select even the most obscure information.

### Formulas in Procedures

Within a procedure formulas are used to calculate values and control program flow. Most procedures contain many formulas — a typical example is shown below. The formulas in this procedure (there are 25 visible in this window) have been highlighted with a light blue box. Don't worry, I don't expect you to understand this procedure right now — the point is to show how vital formulas are to the operation of almost any procedure, and to show the wide variety of formulas possible, from very simple like a single number or text item to a complicated multi-line formula.

```

local ACTION
ACTION=parameter(1)
case ACTION contains "start"
  disableabort
  fileglobal ProgressDescription,ProgressDetail,Boiling,ProgressAbortMode,
    ThermoWidth,Mercury,Temperature,oldTemperature,ProgressCount
  local dlgHeight,dlgWidth,dlgRectangle
  ProgressAbortMode="confirm"
  if info("trigger")="Abort"
    settrigger ""
  endif
  Boiling=parameter(2)
  ProgressDescription=parameter(3)
  ProgressDetail=""
  //call .OpenDialog,"Progress",78,236,"-pause"
  dlgHeight=292 dlgWidth=207
  if (dlgHeight*dlgWidth<(rwidth(info("windowrectangle"))*rheight(info("windowrectangle"))*0.5) and
    dlgWidth<rwidth(info("windowrectangle")) and dlgHeight<rheight(info("windowrectangle")))
    /* center dialog within current window */
    dlgRectangle=rectanglecenter(
      info("WindowRectangle"),
      rectangles:ze(0,0,dlgHeight,dlgWidth))
  else
    /* center dialog within main monitor screen */
    dlgRectangle=rectanglecenter(
      info("ScreenRectangle"),
      rectangles:ze(0,0,dlgHeight,dlgWidth))
  endif
  fitwindow
  setwindowrectangle dlgRectangle,"nopalette novertscroll nohorzscroll"
  openform "Progress"

  object "Thermometer"
  ThermoWidth=rwidth(objectinfo("rectangle"))
  SelectObjects ObjectInfo("name") = "Mercury"
  Mercury=objectinfo("rectangle")

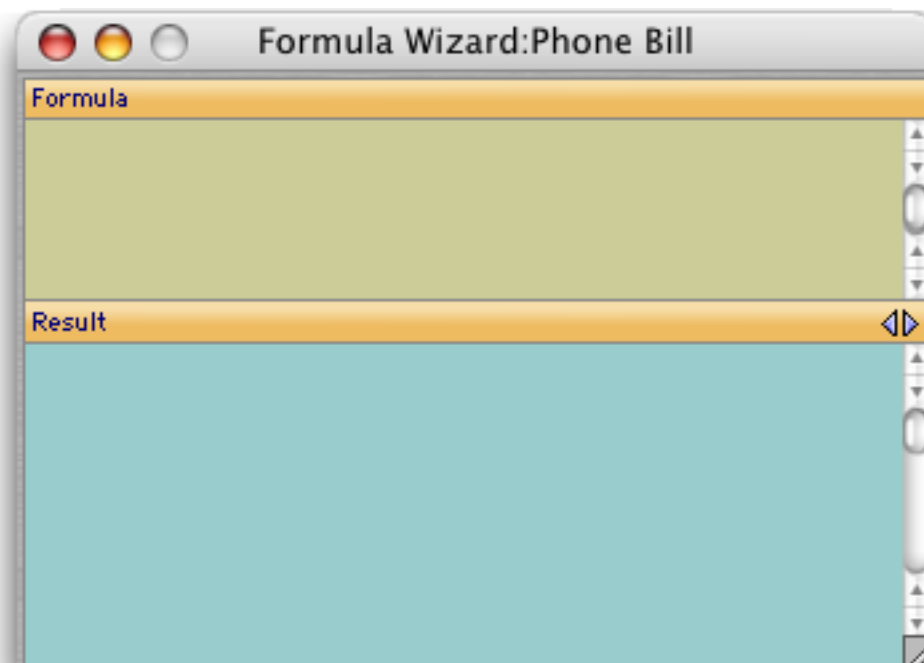
```

To learn more about creating procedures see Chapter 24.

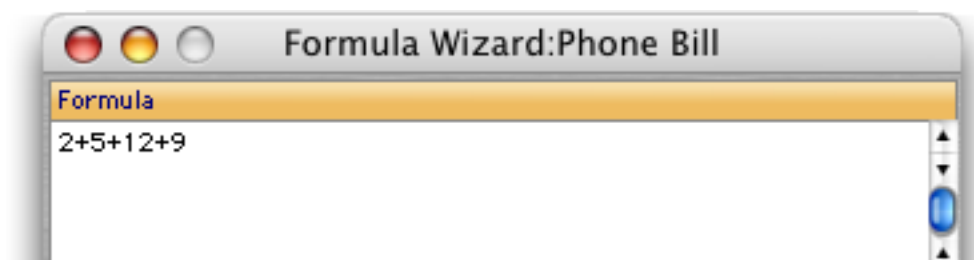


## Using the Formula Wizard

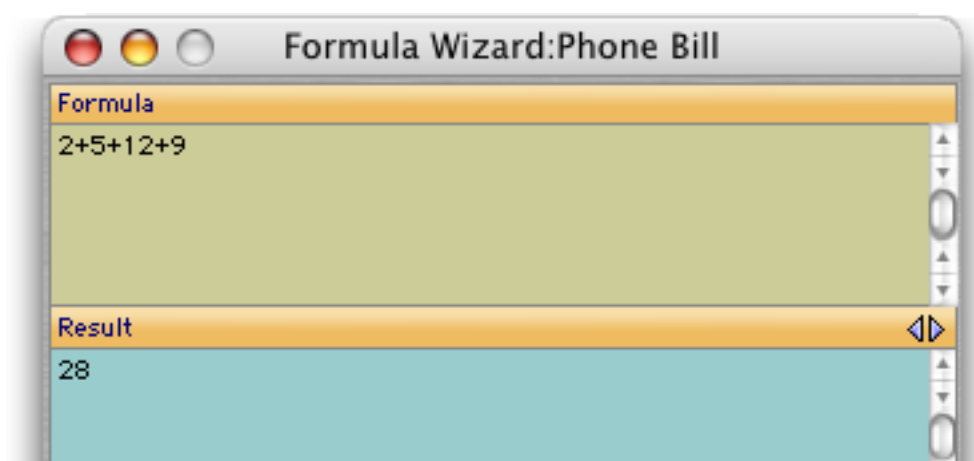
Panorama includes a **Formula Wizard** that you can use as a “sandbox” for playing with formulas. The **Formula Wizard** lets you play around with formulas and see the results immediately. It’s great for trying out a new function or technique you are not familiar with or to work the bugs out of a formula before actually incorporating it into your database. To use this wizard start with any database and select Formula Wizard from the Calculation submenu in the Wizard menu. (Tip: Unless you’ve changed this setting with the **Hotkey** wizard, you can also open the Formula Wizard by pressing **Control-1** on Macintosh systems.)



To use the **Formula Wizard** type a formula into the top section.



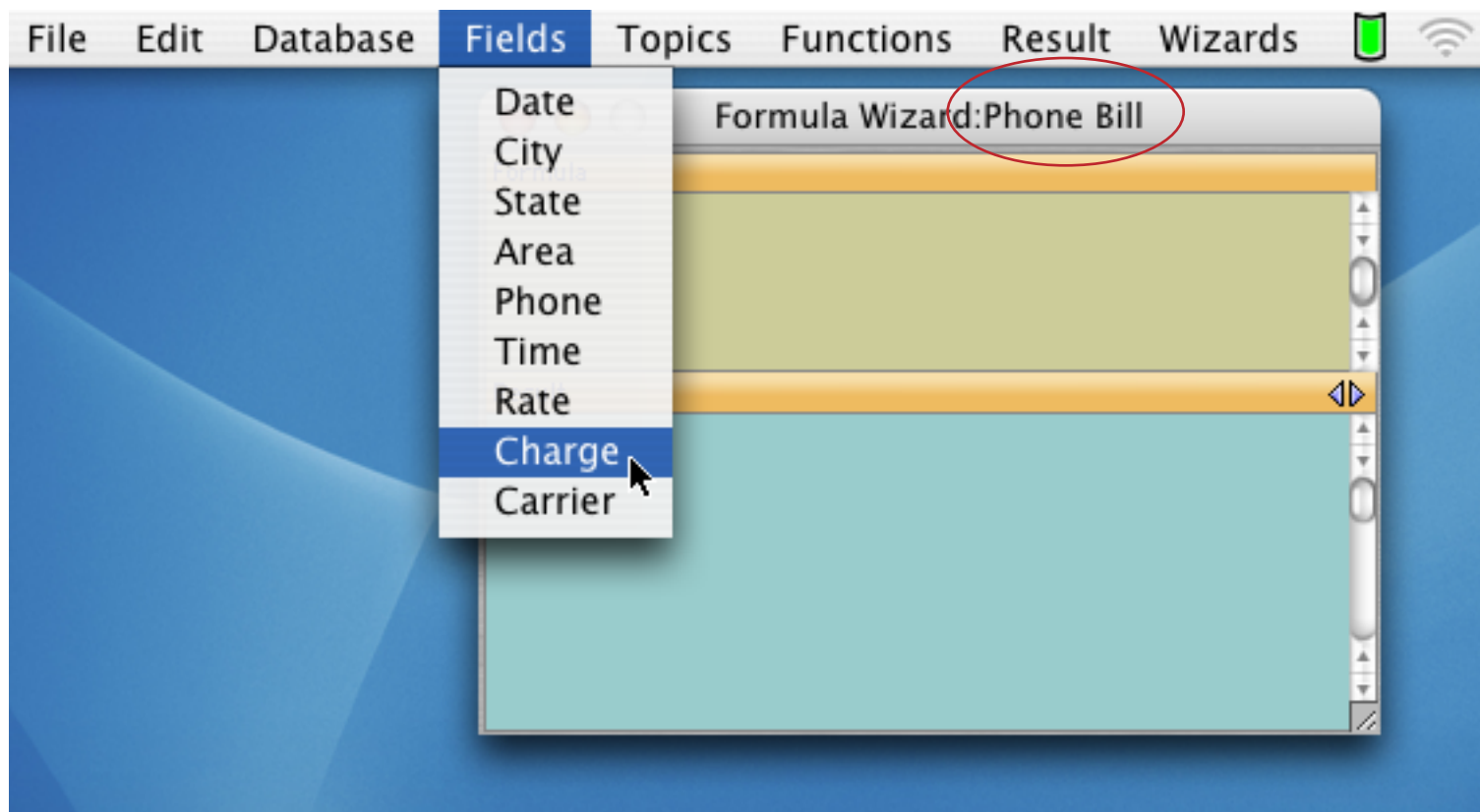
When you press the **Enter** key the result appears in the bottom section (see “[Arithmetic Formulas](#)” on page 526).



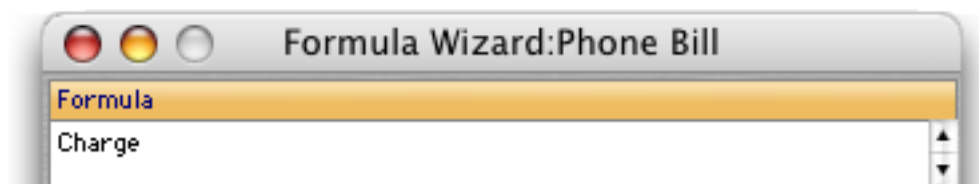
As you can see, the **Formula Wizard** can be used as a handy calculator.

### Calculations with Database Fields

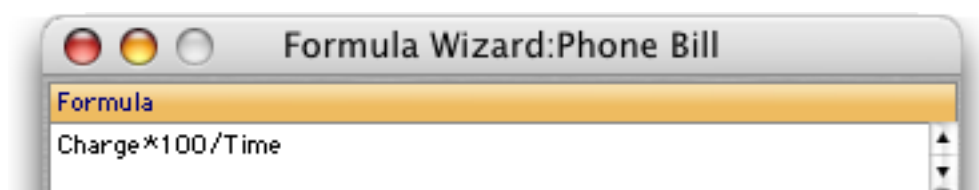
The **Formula Wizard** can include values in database fields as part of the calculation. The name of the database that is currently linked with the wizard is shown in the window's title bar, in this case **Phone Bill**. If you forget a field name you can see a list of all the fields in the **Fields** menu.



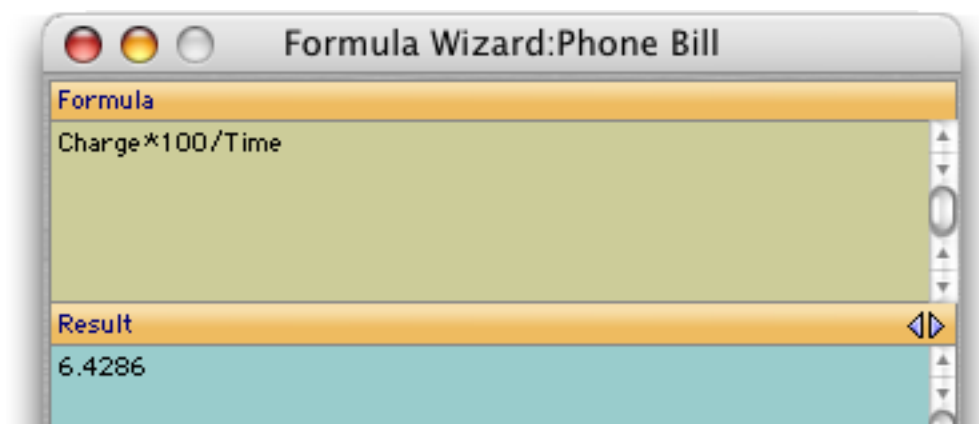
When you select a field from this menu the wizard will automatically type it in for you.



You can included as many fields as you want in the formula. However, they normally must all be from the active database, in this case **Phone Bill**.



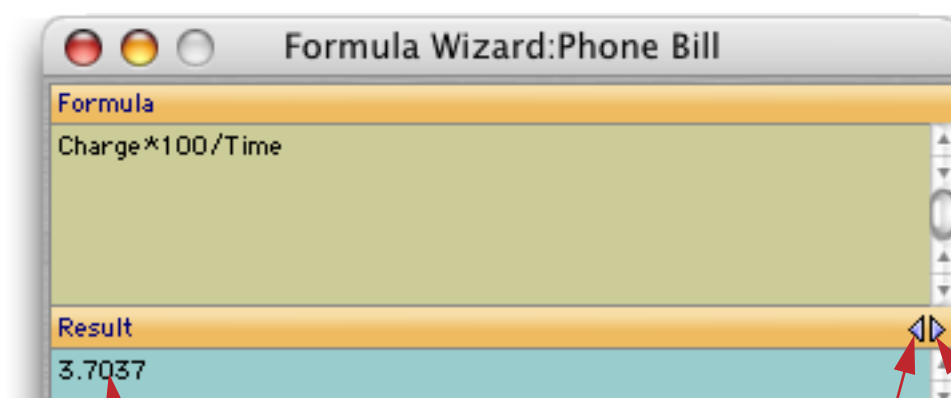
Once again when you press the **Enter** key the wizard will calculate the result, in this case the cost of the phone call in cents per minute.



When the wizard performs the calculation it does so based on the currently active record in the active database. In this example the calculation was based on a charge of 2.70 for a 42 minute phone call.

Date	City	State	Area	Phone	Time	Rate	Charge	Carrier
01/17/84	Anaheim	CA	714	991-4328	42	DE	2.70	GTE
01/29/84	Anaheim	CA	714	533-6619	27	DN	1.00	GTE
01/29/84	Anaheim	CA	714	533-6619	63	DN	2.30	GTE
02/02/84	Anaheim	CA	714	533-6619	26	DE	1.69	GTE

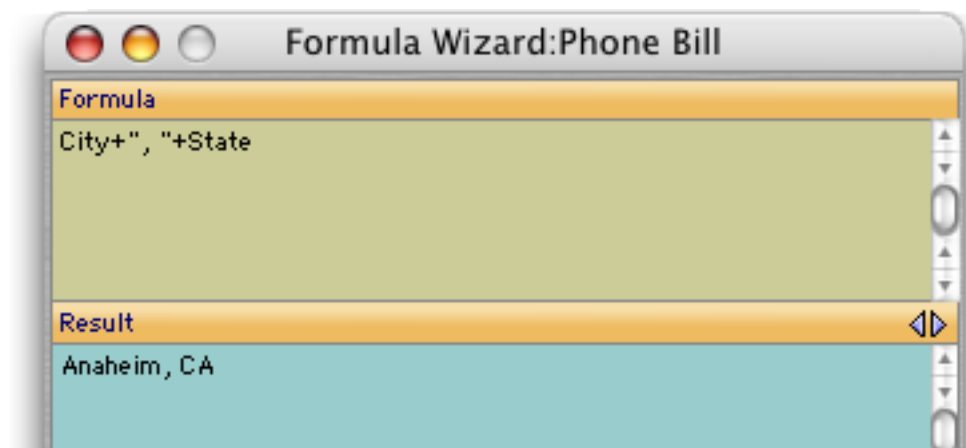
You can use the two small triangles to try out the calculation for other records in the active database. As you press the triangles the active record will move up and down.



*calculation automatically updates*

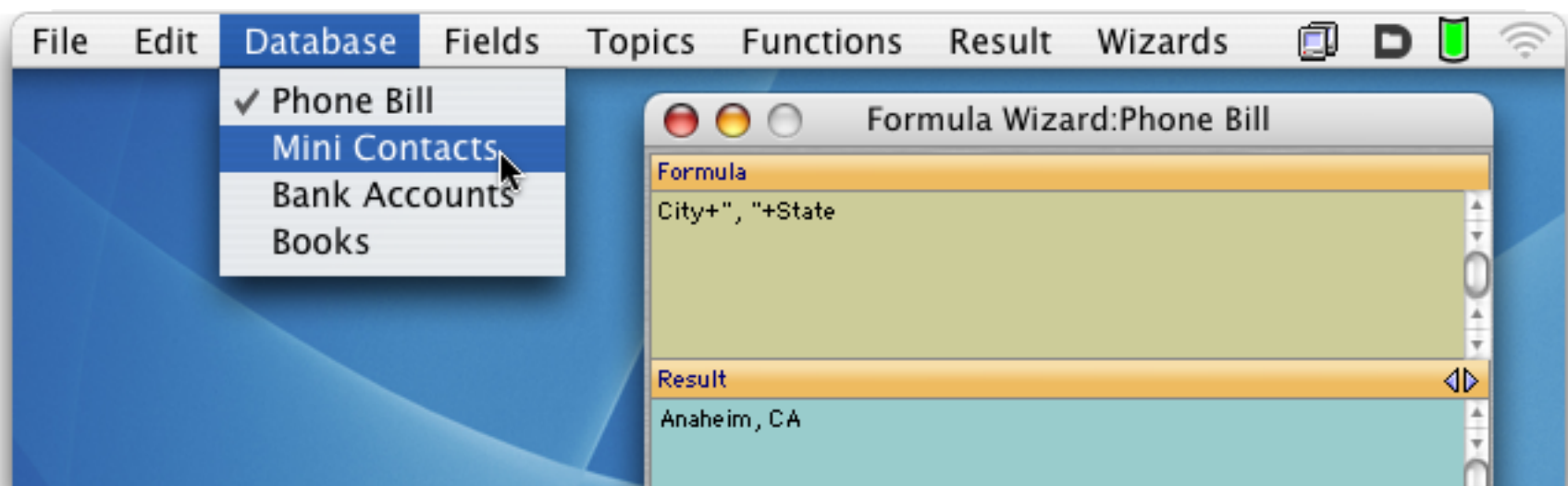
*press to move down one record*  
*press to move up one record*

A formula can also work with text fields (see [“Text Formulas”](#) on page 531). This formula glues the city and state together with a comma in between (see [“Gluing Strings Together”](#) on page 531).

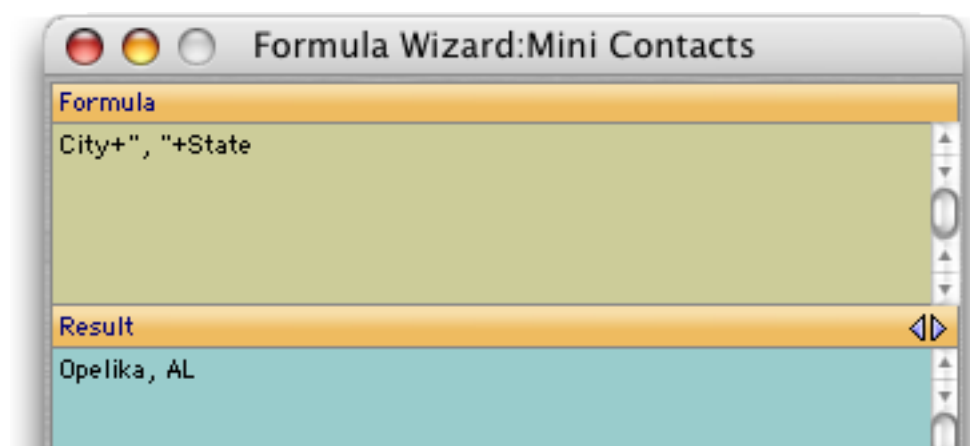


### Changing the Active Database

You can change the database that is used by the **Formula Wizard** at any time with the **Database** menu. This menu lists all of the currently open databases.



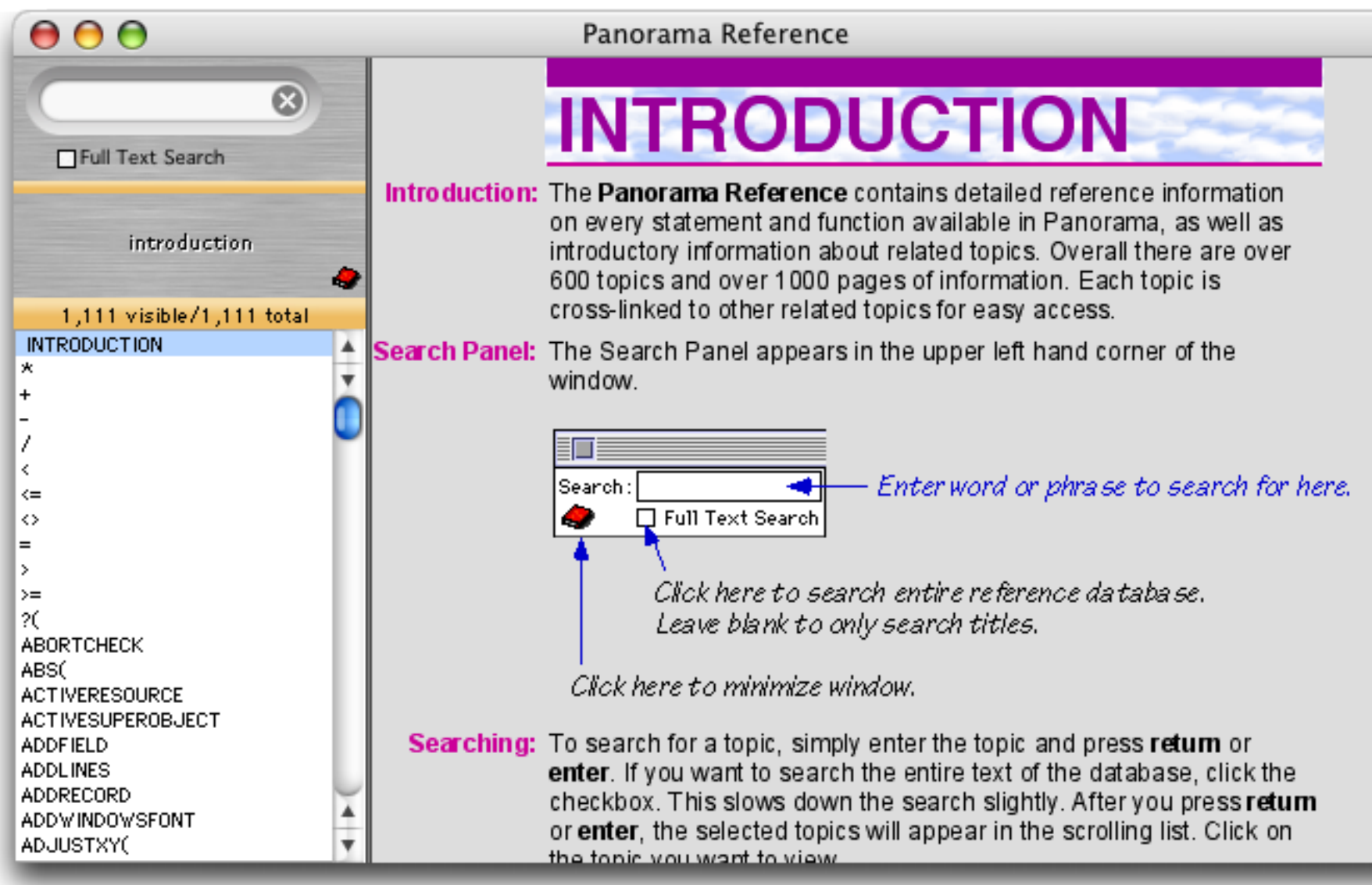
Now when we re-evaluate the same formula it comes up with a different result (**Opelika, AL**).



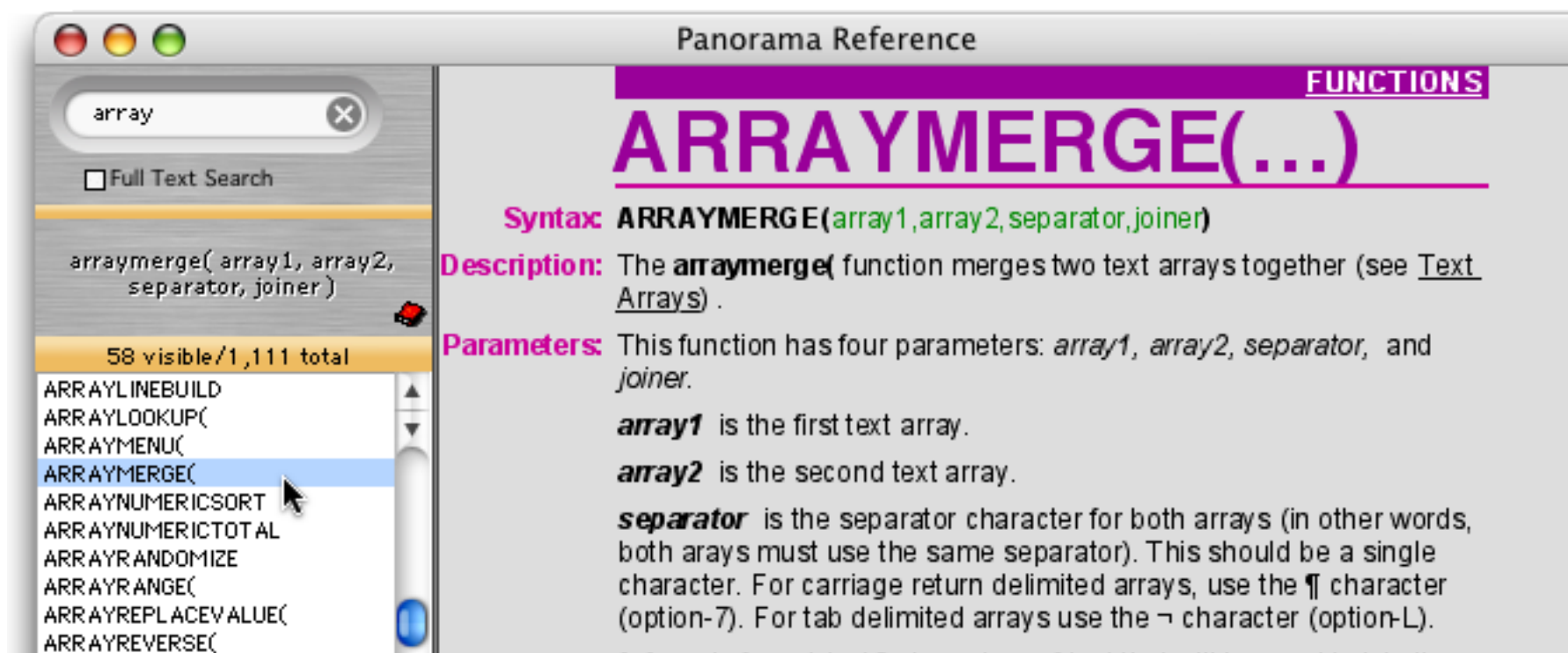
Another way to set the active database is to click on one of the database's windows and then choose **Formula Wizard** from the **Calculations** submenu of the **Wizard** menu. This brings the wizard back to the front and makes the selected database active.

## The Programming Reference Wizard

The fastest way to find complete information for any function or operator is to use the **Programming Reference** wizard. To open this wizard, select it from the Documentation submenu of the Wizard menu, or simply press **Control-R** if you are using a Macintosh computer.



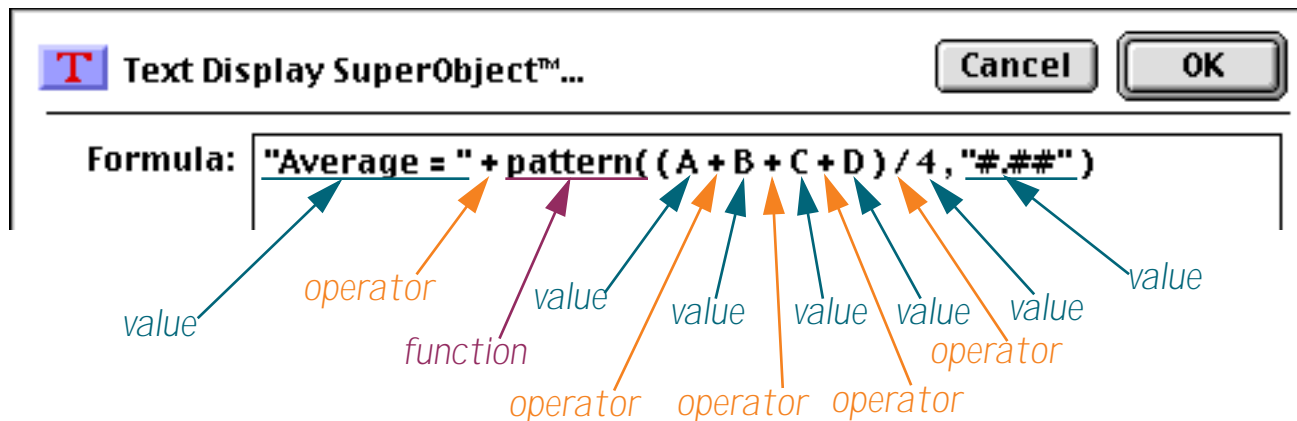
To find complete information for any function or operator simply type in the name of the function or operator in the search box in the upper left, and/or select the function or operator in the scrolling list on the left.



For additional information on this wizard see [“Programming Reference Wizard”](#) on page 1359.

## Formula Components

Just as a sentence is constructed from basic words, a formula is created by combining simple elements — **values** (also called **operands**), **operators** and **functions**. Values (operands) are roughly equivalent to nouns, while operators and functions act as verbs. This illustration shows the components that go into a typical formula.



## Formula Grammar

Panorama formulas have grammar rules just as languages like English and Spanish do. These rules tell how values, operands and functions can be combined to make a valid formula.

The simplest formula is a single data value. Here are four examples of such simple formulas.

A

47

"Origami"

ShippingMethod

Two values can be combined with an operator in between. The first example below adds two numbers together. The second example multiplies two numbers together. The third example appends two text values together (to produce a value like [Mr. Jones](#)).

2 + 2

Total \* TaxRate

"Mr. " + LastName

The values must be the appropriate type for the operator. For example, you can multiply two numbers together like this

2 \* 2

but you cannot multiply two text values together like this (see "[Grammar Errors](#)" on page 517).

"Mr. " \* LastName

You can combine three or more data values with an operator between each pair of values.

7 + 3 \* 4 / 2

FirstName + " " + MiddleInitial + " " + LastName



### Calculation Order and Parentheses

When a formula contains more than one operator, the calculations are performed from left to right unless one of the operators has a higher precedence (priority). This is the natural arithmetic order—multiply and division first, then addition and subtraction. This table lists the order of precedence for all operators.

1. Unary minus (example: -12)
2. Raise to power (example: 10^5)
3. Multiply and Divide
4. Integer Divide
5. MOD (remainder)
6. Add and Subtract
7. Comparisons (=, <>, <, >, ...)
8. NOT
9. AND
10. OR and XOR

For example, consider the formula below.

$$7 + 3 * 4 / 2$$

Panorama first multiplies  $3 * 4$  to get **12**, then divides this by **2** to get **6**. Finally it adds **7** (addition is last because of its low precedence) to get the final result, **13**.

You can override the natural calculation order with parentheses. For example, the parentheses in the formula below force the addition to be calculated first, then the multiplication and division.

$$(7 + 3) * 4 / 2$$

Now the final result is **20** instead of **13**. When in doubt you can always add parentheses to force Panorama to calculate the formula in any order you want.

### Functions

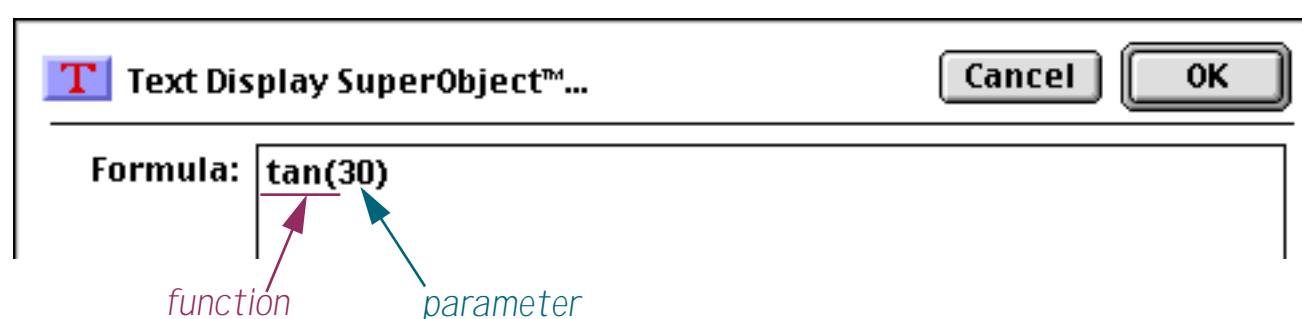
A function is a formula component that calculates a value. It may calculate the value out of thin air (for example, calculating the value of the current date or time) or it can calculate the value from other values (for example trigonometry functions calculate values from angles). Panorama has several hundred functions available. Each function has a name, and is always followed by parentheses. For example, the **tan()** function calculates the tangent (a trigonometry function) of an angle.

$$\tan(30)$$

A function can be used in a formula anywhere a regular value can be used. Just as with ordinary values, you can use operators to combine functions with other values (and functions).

$$3 + \tan(30)$$

The value operated on by the function is called a **parameter**.



A function takes the parameter value (in this case **30**) and transforms it into another value (in this case **-6.4053**, the tangent of **30**). The parameter can be a formula itself, like this.

```
tan( A + B )
```

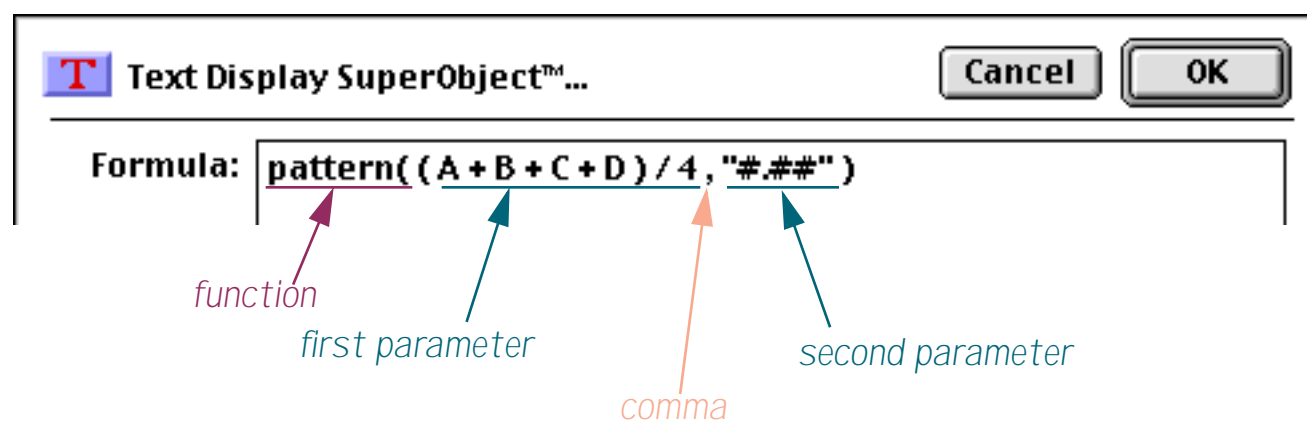
In this case Panorama first calculates the value **A+B**, then computes the tangent of that sum. A parameter may be as complex a formula as you need, with additional parentheses and even other functions nested inside the first function.

```
tan( sqr( A + B ) + 1 )
```

The parameter to the **sqr()** function is **A+B**, while the parameter to the **tan()** function is **sqr(A+B)+1**. (The **sqr()** function, by the way, calculates square roots.) Panorama will always calculate the formula from the inside out until the entire formula has been computed.

### Multi-Parameter Functions

Many functions use more than one parameter. When more than one parameter is required each parameter is separated from the next by a comma. All of the parameters are surrounded by parentheses, just as with single parameter functions. For example, the **pattern()** function (shown below) requires two parameters. The first parameter must be a numeric value (in this case a calculated average) and the second parameter must be a text value containing a pattern for formatting the number (see “[Numeric Output Patterns](#)” on page 256).



Some functions require as many as six parameters. You must always supply every parameter — you cannot leave one out (see “[Grammar Errors](#)” on page 517).

### Zero Parameter Functions

A small handful of functions don't require any parameters at all. These functions generate a value all by themselves, either by consulting the computer hardware (current date, current time), querying internal Panorama data (line number, imported data) or by generating a completely random number each time the formula is computed.

```
today()      -- current date
now()       -- current time
seq()       -- line number
import()    -- line of text from import file
rnd()       -- random number
```

As you can see, these functions simply have both parentheses next to each other, with no parameter in between. You cannot omit the parentheses — you are required to include them as shown in the examples above.

## Whitespace

Most of the examples you've seen so far have extra spaces between the components, like these.

```
7 + 3 * 4 / 2
```

```
FirstName + " " + MiddleInitial + " " + LastName
```

```
tan( sqr( A + B ) + 1 )
```

Panorama ignores spaces between components. You can leave out the spaces, like this.

```
7+3*4/2
```

```
FirstName+" "+MiddleInitial+" "+LastName
```

```
tan(sqr(A+B)+1)
```

Or you can add extra spaces between components, or even carriage returns, like this. (Note: Some dialogs do not allow you to enter carriage returns, because pressing the **Return** key closes the dialog.)

```
7 + 3 * 4 / 2
```

```
FirstName + " " +
MiddleInitial + " " +
LastName
```

```
tan(   sqr( A + B ) + 1   )
```

Spaces are only ignored **between** components, not within components. A common mistake is to place a space in between the function name and the left parenthesis. This is not allowed. The formula below will not work (see "[Grammar Errors](#)" on page 517) because of the spaces after **tan** and **sqr**.

```
tan ( sqr ( A + B ) + 1 )
```

Another common problem is spaces or other punctuation in field names. If your database has fields named **First Name**, **Middle Initial** and **Last Name** you might be tempted to try a formula like this.

```
First Name + " " + Middle Initial + " " + Last Name
```

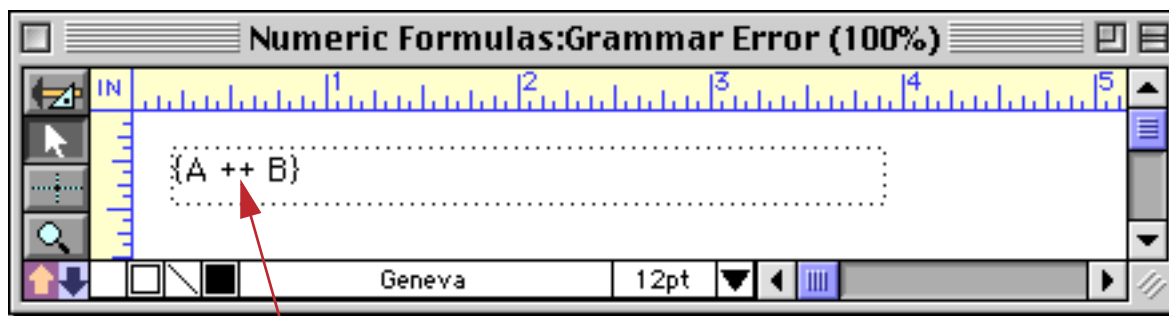
Sorry, but it won't work (see "[Grammar Errors](#)" on page 517). Because of the spaces inside the field names, Panorama will think that **First** and **Name**, **Middle** and **Initial** and **Last** and **Name** are separate components. The solution is to place chevron (« and ») characters around the field names. In many cases you can use the **Field** menu to type in the field name with chevrons for you. Otherwise, on the Macintosh press **Option-\<** to create the « chevron character and **Shift-Option-\<** to create the » chevron character. On Windows systems press **Alt-0171** to create the « chevron character and **Alt-0187** to create the » chevron character. Here's the revised formula, which will work perfectly

```
«First Name» + " " + «Middle Initial» + " " + «Last Name»
```

You'll also need to put chevrons around a field or variable name that contains punctuation, for example **«P/E Ratio»**. Without the chevrons Panorama will think that this is four separate components — **P**, **/**, **E** and **Ratio**.

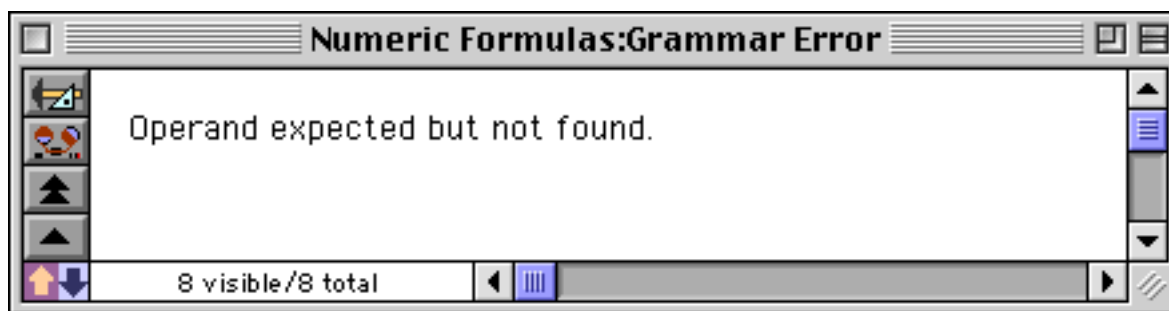
### Grammar Errors

Unlike a human listener, Panorama is not able to tolerate incorrect or sloppy grammar. If you ask Panorama to calculate a formula that has incorrect grammar it will refuse to comply until you correct the mistake. For example, consider the formula shown below in an auto-wrap text object.



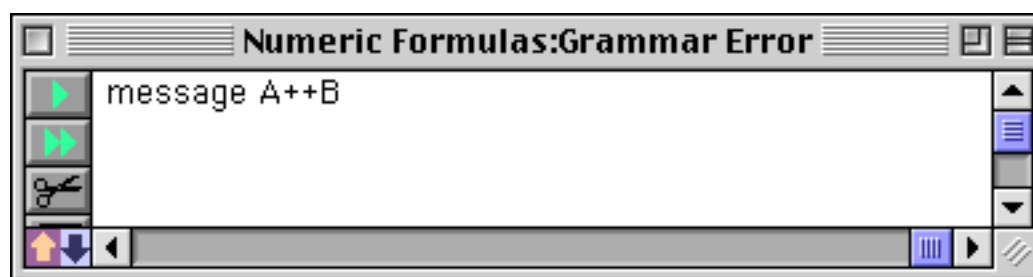
*Grammar error! Can't have two operators in a row!*

When you switch to Data Access Mode Panorama tells you about the grammar error.

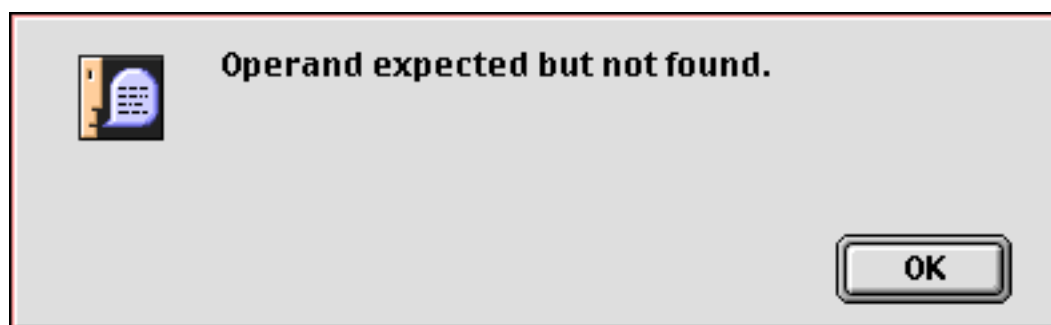


The error message tells you that Panorama expected an operand (value) after the + operator. The solution is either to remove the extra + operator or add another value in between the two + symbols.

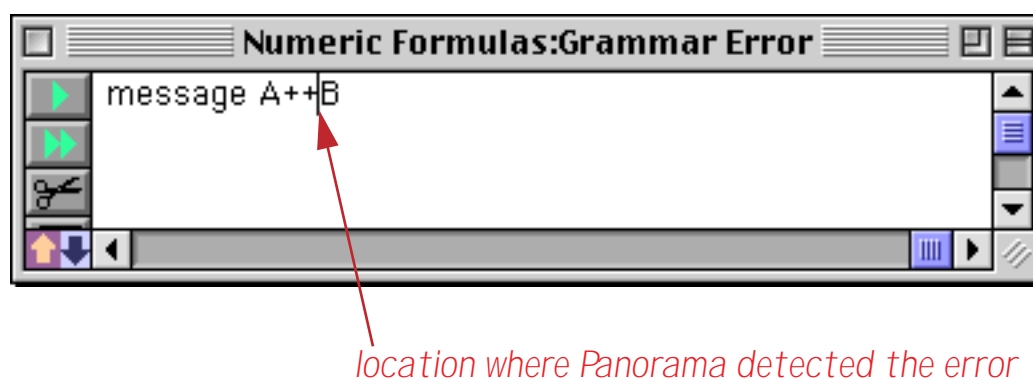
When you are editing a formula within a procedure, Panorama will attempt to point out the location of the grammatical error. For example, here is the same formula with the same error used in a procedure.



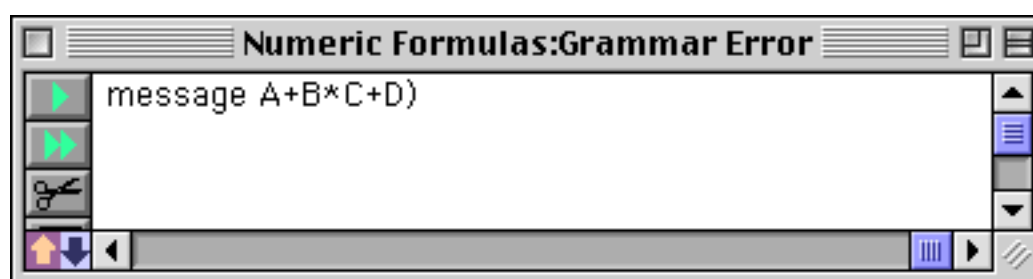
If you click to another window or use the **Check Procedure** command (in the Edit menu) Panorama will display an alert letting you know about the problem.



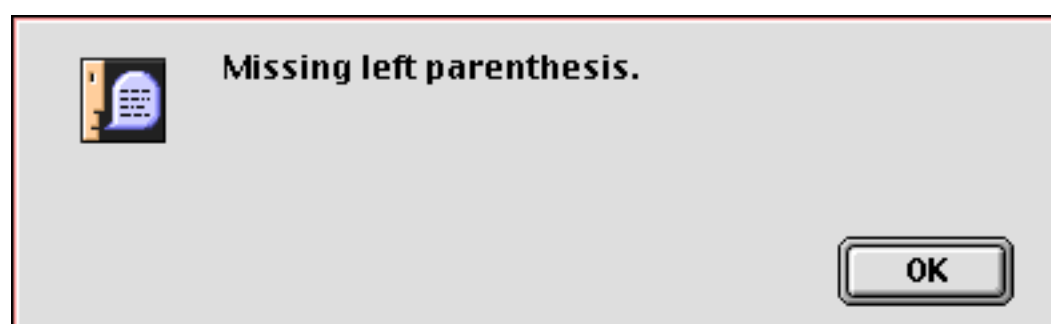
When you close the alert window Panorama will move the insertion point to the location where Panorama detected the error.



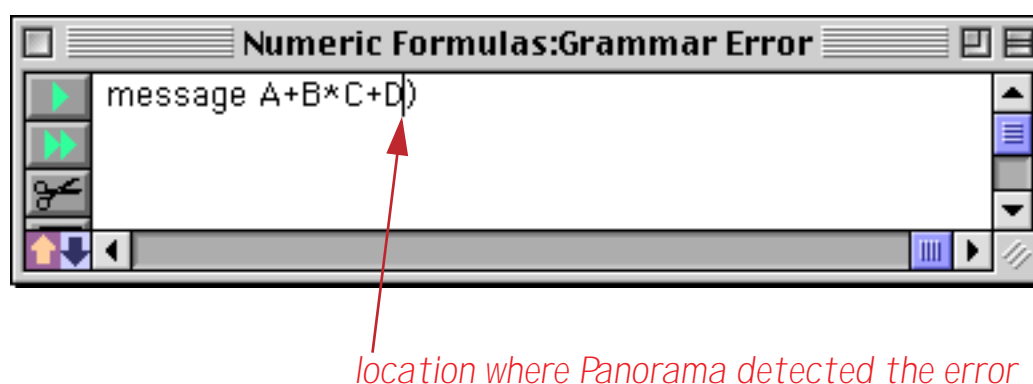
This location is usually fairly close to where the actual error is. However, in some cases Panorama is unable to determine exactly where the problem is. Consider the formula shown below, which has a missing left parenthesis.



When you click to another window or use the **Check Procedure** command (in the Edit menu) Panorama will display an alert letting you know about the problem.



When you close the alert window Panorama will move the insertion point to the location where Panorama detected the error.



But wait — is this really where the error is? No, the error actually is somewhere earlier in the formula. In this case the missing ( probably goes in front of the **B** or the **C**. Panorama has done the best job it could to locate the error for you. One thing you do know for sure, though, is that the error is always before the insertion point and not after.

## Values

Values are the raw material that formulas work with—numbers and text. A value may be embedded in the formula itself, may be stored in a database field or may be stored in a variable (see “[Variables](#)” on page 521 and “[Variables](#)” on page 1370).

## Constants

When a value is embedded in the database itself it is called a **constant**. A numeric constant may be in fixed point format, like the numbers in this example (the numeric constants are highlighted in purple).

```
x + 2
```

```
today() - 90
```

```
Total * 0.0625
```

A numeric constant may also be in floating point format, which consists of the mantissa followed by the letter **e** followed by the exponent. The example below is equivalent to the mathematical formula  $x \cdot 6.02^{23}$ .

```
x * 6.02e23
```

A formula may also contain **text constants**. A text constant is a series of characters surrounded by quotes. When writing a text constant you may choose from five different types of quotes, as shown in this table.

Type	Open	Close	Example
Double Quote	"	"	"January"
Single Quote	'	'	'Tuesday'
Curly Braces	{	}	{San Francisco}
Smart Double Quote	“	”	“Gothic”
Smart Single Quote	‘	’	‘Bohemian’
Pipes	,   ,     etc.	,   ,     etc.	abc

The primary reason for different types is to allow quotes themselves to be used in a text constant. Suppose that you needed to use the text **The shim was 6" high** in a formula. Using double quotes around the constant will cause a grammar error, because Panorama won't know what to do with the text after **6"** (shown in red below).

```
"The shim was 6" high"
```

One possible solution is to use a different quote character around the constant. Any of the examples shown below will work.

```
'The shim was 6" high'
```

```
{The shim was 6" high}
```

```
“The shim was 6" high”
```

```
‘The shim was 6" high’
```

Another solution is put two double quotes in a row (as highlighted dark blue in the example below). Panorama will convert these into a single quote and continue with the text constant.

```
"The shim was 6"" high"
```



### Build in Constants: Pi, Carriage Return and Tab

Panorama has one built in numeric constant—**pi**. Use the Greek  $\pi$  symbol to access this value. For example the area of a circle can be calculated with this formula.

```
 $\pi$  * radius^2
```

To create the  $\pi$  symbol on the Macintosh press **Option-P**. On the PC, type **Alt-0254**.

Panorama has two built in text constants—**¶** (Carriage Return) and **↵** (Tab). For example three line address can be included in a formula like this.

```
"Suzette Elliot"+¶+892 Melody Lane"+¶+"Fullerton, CA 92831"
```

To create the **¶** symbol on the Macintosh press **Option-7**. On the PC, type **Alt-0182**.

To create the **↵** symbol on the Macintosh press **Option-L**. On the PC, type **Alt-0172**.

### Fields

To use a field within a formula, type the name of the field into the formula. This formula adds up the sum of three fields.

```
SubTotal+Shipping+Tax
```

When a field is used in a formula it always refers to the value of that field in the current record in the current database (the database belonging to the topmost window). As you move from record to record the result of the computation will change depending on the values in that particular record. (The only exception to this rule is the **lookup()** and **grabdata()** functions, which may refer to fields in other records or even other databases.)

If a field name contains spaces, numbers, or punctuation marks in it, you must surround the name with chevron characters (« and »). (On the Macintosh press **Option-\** to create the « chevron character and **Shift-Option-\** to create the » chevron character. On Windows systems press **Alt-0171** to create the « chevron character and **Alt-0187** to create the » chevron character.) If the field name contains carriage returns, they must be represented with spaces. Here is a database with some unusual field names.

Price	Quantity	Zip Code	P/E Ratio
34.99	12	93221	15.67
12.99	154	50442	9.12
175.99	81	20165	45.83

The first two names can be used without chevrons, but the last two require chevrons because of spaces and punctuation in the names.

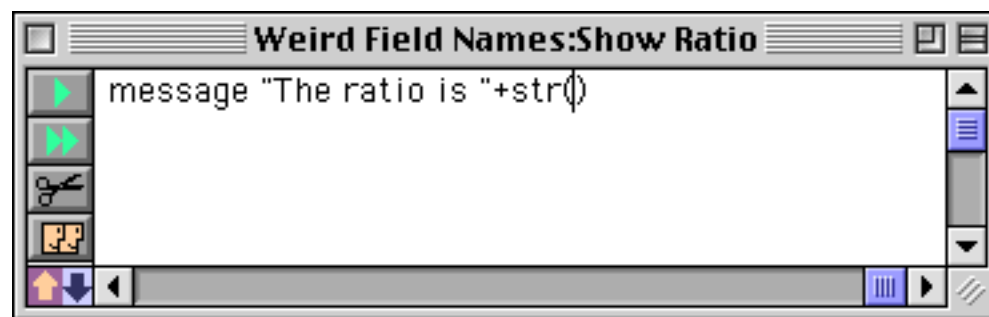
```
Price
```

```
Quantity
```

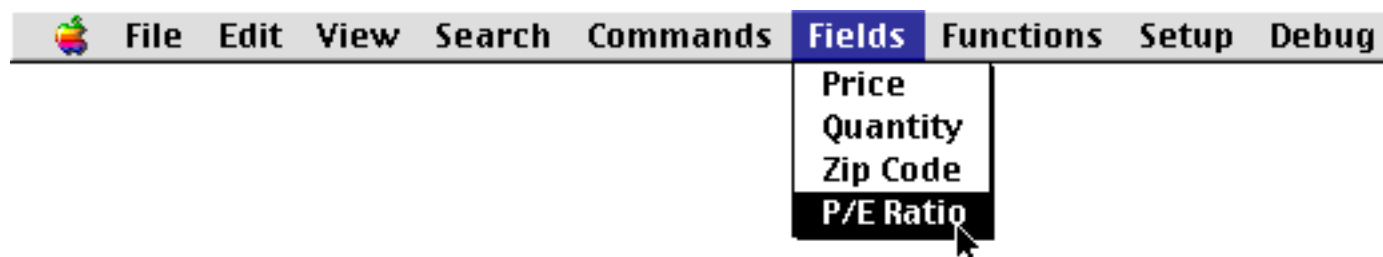
```
«Zip Code»
```

```
«P/E Ratio»
```

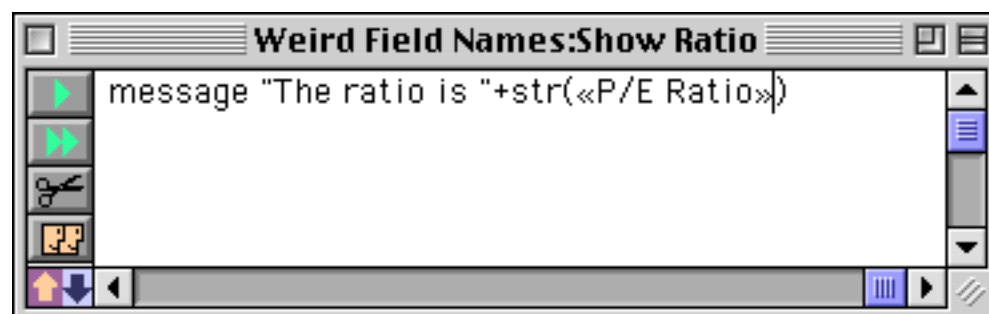
Formulas require field names to be spelled exactly as they appear in the database, with no typos allowed. Fortunately, Panorama can help you out with this. Start by positioning the insertion point where you want the field to appear.



Now pick the field from the **Field Menu**. This menu is available whenever you are editing a formula in a dialog, design sheet or procedure.



Panorama will type in the field name for you, including the chevrons if necessary (as they are in this case).



If the chevrons are not necessary (for example for **Price** or **Quantity**) Panorama will not include them.

#### Using the Current Field

A formula may use «» (see “[Special Characters](#)” on page 525) to refer to the current field without having to know what the current field is. For example, this formula converts the current cell to upper case.

```
upper («»)
```

If necessary, a formula can find out what the current field name is with the `info("fieldname")` function (see “[INFO\("FIELDNAME"\)](#)” on page 5375).

#### Variables

A variable is a place in the computer where an item of data can be stored, kind of like a storage bin for a value. Variables may be created by procedures or by SuperObjects. Most procedures will use one or more variables to hold and transfer data as the program runs (see “[Variables](#)” on page 1370 for more details on how variables can be created and used in procedures). Use a variable whenever you need to store a single data item so that you can use it later. Unlike a field, the value variable doesn’t change as you move from record to record, or, in the case of a global variable, even when you move from database to database.

## Variable Names

Just as a house is identified by its address, a variable is identified by its name. A street address tells you exactly how to find a house or business. It doesn't tell you who or what is inside the house, however. Families may come and go, but the street address remains the same. In a similar way, a variable name identifies a place where data can be stored. The data may change, but the variable name remains the same.

Panorama allows any sequence of characters to be used as a variable name. However, if the variable name contains any punctuation (including spaces) it must be surrounded by the chevron characters « and ». (On the Macintosh press **Option-** to create the « chevron character and **Shift-Option-** to create the » chevron character. On Windows systems press **Alt-0171** to create the « chevron character and **Alt-0187** to create the » chevron character.) Here are some examples of typical variable names:

`X`

`birthDay`

`Counter`

`«Tax Rate»`

`«PrimeRate%»`

A variable name must be spelled exactly the same way every time, including upper and lower case. The variable name `birthDay` is not the same as `Birthday` or `birthday`. In fact, you could create three different variables using these three different names (although this is not recommended because it would be very confusing).

By the way, it's always ok to use chevrons around a variable name, even if the name doesn't have any punctuation. `«Counter»` is exactly the same as `Counter`, and they can be used interchangeably. So if you have any doubts about whether or not chevrons are necessary, go ahead and use them. No harm, no foul.

### What's Inside A Variable?

By itself, a variable has no meaning, no value...until you put some data in it. When you use a variable in a formula or procedure, you are actually telling Panorama to use the contents of the variable.

A variable is sort of like a cup that you can pour anything into. A cup may contain water, soda, tea, or coffee. If you tell a person to drink the blue cup, what you really mean is to drink whatever liquid is in the blue cup. Each time they drink they may get a different liquid, depending on what the blue cup has been filled with.

Using a variable is similar. If you tell Panorama to calculate `X+Y` (where `X` and `Y` are variable names), what you really mean is "take whatever value is in `X` and whatever value is in `Y` and add them together."

It's important to remember that a variable name simply identifies the variable, but the name is not the variable itself. The name is like a placeholder for the real contents of the variable.

### The Life Cycle of a Variable

A variable doesn't just appear by magic. It must be created, just as you have to build a house before you can move in. Once the variable has been created it can be used for storing a data item. However, variables don't last forever. Most variables eventually disappear without a trace. You can also force a variable to disappear at any time — see "[Destroying a Variable](#)" on page 1372.

Panorama has five kinds of variables: local, window, fileglobal, global and permanent. The only difference between these three types of variables is how long they last before disappearing and when the variables are available.

**Local** variables are the most short-lived. A local variable disappears when the procedure that created the variable is finished. In addition, a local variable can only be used by the procedure that created it. If procedure A calls procedure B as a subroutine, procedure B cannot access the local variables created by procedure A. In fact, procedure B could create its own local variables with the same names as the local variables created by procedure A. Panorama keeps the local variables for each procedure completely separate from each other.

**Window** variables are associated with a particular window. A window variable is only accessible when the window it is associated with is on top, and the variable disappears completely when the window is closed. It is possible for several different windows to have window variables with the same name. In that case, each window variable may have a different value.

**FileGlobal** variables are associated with a particular database (file). A fileglobal variable is only accessible when the database it is associated with is the current database (on top), and the variable disappears completely when the file is closed. It is possible for several different files to have fileglobal variables with the same name. In that case, each fileglobal variable may have a different value. For many applications fileglobal variables are the best choice because there is no chance of an accidental conflict with a variable of the same name in another database.

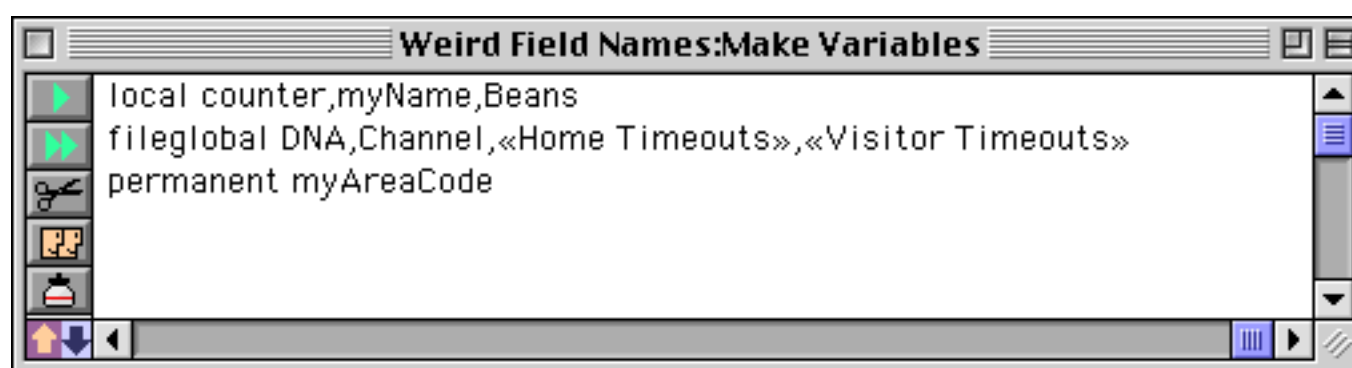
**Global** variables are relatively long-lived. A global variable doesn't disappear until you quit from Panorama. Even if you close the database, the global variable remains. Once a global variable has been created it can be accessed in any procedure, in any database or window, at any time. You should avoid using global variables unless you absolutely need universal access across databases for the value stored in the variable. If the value is only needed in one database it is much better to use a fileglobal variable to avoid the chance of an accidental conflict with another database using the same global variable name.

**Permanent** variables are almost immortal. When the database is saved, all permanent variables in that database are also saved. Like a fileglobal variable, a permanent variable is only accessible when the database it is created in is the current database, and a fileglobal variable disappears when you close the database. However, unlike a fileglobal variable, a permanent variable will re-appear like a phoenix from the ashes when you re-open the database. In fact, there are only two ways a permanent variable can permanently disappear. First, you can explicitly kill a permanent variable with the **unpermanent** statement. Secondly, you can create a permanent variable but never save the database.

### Creating Variables in a Procedure

Panorama has five statements for creating variables in a procedure: **local**, **windowglobal**, **fileglobal**, **global**, and **permanent**. Each of these statements is exactly the same except for the type of variable created. The statement must be followed by a list of one or more variables to create, with each variable name separated from the next by a comma. (Remember: if a variable name contains punctuation, it must be surrounded by chevrons « and ».)

Here are a few examples of typical statements for creating variables:



There is no limit to the number of local, global, and permanent statements you use in your programs, and no limit to the total number of variables.

## Initializing Variables

Creating a variable creates a place to store data, but it doesn't actually put any data in the variable. It's kind of like a new house that no one has moved into yet. If you try to access the variable before any data has been put into it, an error occurs.

To put data into a variable, use an assignment statement. Here's the start of a procedure that creates a variable named **Count** and initializes it to zero. The variable is now ready to use.

```
local Count
Count=0
```

Sometimes you may not be sure if a global variable has been initialized yet. If it has not been, you want to initialize it. But if it has already been initialized, you don't want to disturb the value that is already there. You can get around this problem with the if error statement, as shown in this example.

```
global AreaCode
AreaCode=AreaCode
if error
  AreaCode="714"
endif
```

This procedure starts by creating a global variable named **AreaCode**. However, it's possible that **AreaCode** has already been created and initialized by another procedure. To test this, the procedure copies the variable to itself. If the variable is already initialized, there will be no error and the contents of the variable have not been disturbed. If the variable is brand new and has not been initialized, an error occurs. This error is trapped by the if error statement and the variable is initialized. If you have a number of variables that are always initialized as a group, you don't need to test each one. Just test one, and if an error occurs initialize the entire group of variables (see "[The Define Statement](#)" on page 1367).

## Variables and Data Types

A variable can hold any kind of data: text, numbers, and secondary data types like dates, times, points, rectangles, etc. In addition, you can change the type of data in a variable at any time. One minute the **AreaCode** variable can contain text, moments later it can contain a number. The variable takes on the data type of whatever data you copy into it.

## SuperObject Variables

A number of Panorama SuperObjects™ have the option of linking to a variable or a field. These SuperObjects™ include the Text Editor, Data Button, Pop-up Menu, List, Sticky Button and Scroll Bar. If one of these objects is linked to a variable and the variable does not exist, Panorama will automatically create a global variable when it opens the form, and initialize the variable to empty text. Except for how it was created, this global variable is just like any other global variable and can be used freely in procedures and formulas.

## Variable Name Conflicts

If two database files define a global variable with the same name, you've got a conflict. It's kind of like two families trying to share the same house. This can work if the two families have an arrangement, but if they don't the result is chaos.

The best solution to this problem is to avoid it. If you can, use a fileglobal variable instead of a global variable. If this is not possible, stay away from simple global variable names like **X**, **Payment**, **Count**, etc. If possible, choose names that incorporate the database name (or an abbreviation of the name), for example **Invoice-TaxRate**, **ReceivablesTotal**, or **APLastReconcileDate**.

Variable names (even for local variables) can also conflict with fields in a database. In this battle, the variable always wins. Panorama will use the data in the variable instead of the data in the field. Avoid variable names that are the same as field names.



## Permanent Variable Tips

When the **permanent** statement creates a permanent variable, it really creates two variables: one in memory and one in the current database. The one in memory is an ordinary fileglobal variable. Whenever the database is saved, Panorama copies the contents of the fileglobal variable into the copy of the variable in the database itself, then saves the database. Just like any other data, the contents of the permanent variable are not saved unless the database itself is saved. However, if you have not made any other changes to the database, Panorama will not warn you if you attempt to close a database without saving changes to the permanent variable.

Whenever a database is opened, Panorama automatically creates fileglobal variables for any permanent variables associated with that database. Next it copies the values from the database into the fileglobal variables. The variables are now ready to use.

If you ever want to make a permanent variable un-permanent, use the **unpermanent** statement, which is followed by a list of variables you want to make unpermanent. This statement doesn't make the variables go away, but they will no longer be permanent. The **unpermanent** statement only affects variables that are permanent in the current database. The example below changes two permanent variables back into regular (non-permanent) fileglobal variables.

```
unpermanent myAreaCode,myZipCode
```

## Special Characters

Formulas are very picky about special characters. You've got to use the right special character in the right spot—no substitutes are allowed.

For example, some people mistake the bracket [] characters for the parentheses (). On your keyboard, the parentheses are created by pressing **Shift** and the **9** or **0** keys. Another common mistake is using the \ (backslash) instead of the / (slash) for divide. The table below lists all the special characters used by formulas and shows how to type them.

Character	Name	Mac	PC
(	left parenthesis	Shift-9	Shift-9
)	right parenthesis	Shift-0	Shift-0
[	left bracket	[	[
]	right bracket	]	]
{	left curly brace	Shift-[	Shift-]
}	right curly brace	Shift-]	Shift-]
«	left chevron	Option-\	Alt-0171
»	right chevron	Shift-Option-\	Alt-0187
^	caret (raise to power)	Shift-6	Shift-6
*	asterisk (multiply)	Shift-8	Shift-8
÷	divide	Option-/	not available, use /
=	equal	=	=
≠	not equal	Option-=	not available, use <>
<	less than	<	<
>	greater than	>	>
≤	less than or equal	Option-<	not available, use <=
≥	greater than or equal	Option->	not available, use >=



Character	Name	Mac	PC
¶	paragraph	Option-7	Alt-0182
⇥	export tab	Option-L	Alt-0172
§	section mark	Option-6	Alt-0167
¢	cents	Option-4	Alt-0162
‘	left smart quote	Option-]	Alt-0145
’	right smart quote	Shift-Option-]	Alt-0146
“	left smart double quote	Option-[	Alt-0147
”	right smart double quote	Shift-Option-]	Alt-0148
Ω	omega (line items)	Option-Z	Alt-0166
π	pi	Option-P	Alt-0254

To use the **Alt** key on the PC you must hold down the **Alt** key, then press the numeric digits (for example **0182**) then release the **Alt** key. When you release the **Alt** key the special symbol will appear.

## Arithmetic Formulas

Panorama formulas are very adept at performing arithmetic—from simple addition to complex financial calculations. Arithmetic formulas usually work just like the ones you learned about in high school. Panorama has seven arithmetic operators, as shown in this table.

symbol	operator
+	add
-	subtract
*	multiply
/ or ÷	divide
^	raise to power
\	integer divide
mod	modulo (remainder)

The ^ operator (press **Shift-6**) raises the operand on the left to the power specified on the right. For example the formula

$$2^3$$

means raise 2 to the third power (equivalent to the mathematical formula  $2^3$ ).

## Dividing by Zero

Dividing by zero is, of course, a no-no. If you do attempt to divide by zero, Panorama will display an alert reminding you of this arithmetical impossibility. Sometimes, however, you may want to defy mathematical reality and divide by zero without getting slapped on the wrist. For example, since formulas treat empty data cells as zeros, attempting to divide by a cell that hasn't been entered yet will result in a divide by zero error. To bypass the error message, use the `divzero()` function instead of the `/` operator. The `divzero()` function returns zero if you attempt to divide by zero. For example, using the formula

```
Price/Qty
```

can result in a divide by zero error if `Qty` field is empty, but

```
divzero(Price,Qty)
```

will not.

## Overflow/Underflow Problems

A number is a number, right? Well, not quite. You may remember that Panorama actually stores two different kinds of numbers—fixed digit and floating point, with fixed digit numbers being further divided into 0, 1, 2, 3, and 4 digit precision. In a formula these differences may be important, since some numbers are too big or too small to be represented in some of the fixed point formats.

Formulas try to perform arithmetic using the final numeric type required for the answer. For example, if the result of a formula will be placed in a fixed 2 digit field, calculations will be performed in a fixed 2 digit format unless you force the formula to use another format. If the final destination is not a numeric field, arithmetic will be performed using floating point. Floating point is also used when the answer is not going to be stored in a field—for example formulas that are merged into auto-wrap text object or Text Display SuperObject.

Since the internal format used for arithmetic can vary depending on the final destination of the answer, the same formula can give different results depending on where it is used. For example, the formula

```
1/4
```

gives the result `0.25` if the result is a floating point field, but `0` if the result is a fixed 0 digit field.

A more subtle problem can occur if an intermediate calculation causes an overflow, underflow, or loss of precision. Often this can be fixed by re-arranging the formula. For example, this formula for computing sales tax can have problems if the result will be stored in a 2-digit fixed field.

```
total*taxrate/100
```

If the tax rate is 6.5%, the intermediate result of the division is `0.065`. But since 2-digit fixed point arithmetic is being used, this intermediate result will be rounded to `0.07`, resulting in an incorrect calculation. You can fix this formula by doing the multiplication first.

```
(total*taxrate)/100
```

You can also fix this formula by forcing all the numbers to floating point using the `float()` function.

```
float(total)*float(taxrate)/float(100)
```

If all the operands are in the same numeric format, the formula will calculate the result using that format, in this case floating point.

If you don't want to worry about overflow/underflow problems one solution is simply to make all numeric fields floating point. Floating point fields take up slightly more RAM than fixed point fields, but for most databases the difference isn't critical.

## Basic Numeric Functions

These functions perform various mathematical operations. Each of these functions takes one or more numeric parameters and returns a numeric result.

Function	Description
abs(number)	This function returns the absolute (positive) value of the numeric parameter. In other words, negative numbers are converted to positive numbers while positive numbers remain positive.
divzero(numerator,denominator)	This function divides two numbers. However, unlike the / operator, the divzero( function does not care if you attempt to divide by zero. If you attempt to divide by zero, this function simply returns zero.
fix(number)	This function truncates a number to an integer. It always truncates towards zero. For example <code>fix(-4.6)</code> is <code>-4</code> , while <code>int(-4.6)</code> is <code>-5</code> . For positive numbers the <code>int(</code> and <code>fix(</code> functions are identical.  Don't confuse this function with the <code>fixed(</code> function, which converts numbers from floating to fixed point format.
fixed(number)	This function forces a number to fixed point format, using the least number of digits possible. Since formulas usually perform this conversion automatically, you probably won't ever need this function. Don't confuse this function with the <code>fix(</code> function, which truncates a number to an integer but does not change the type of the data.
float(number)	This function forces a number to a floating point format. You may need to use floating point to get around overflow, underflow, and accuracy problems that can occur when using fixed point arithmetic.
int(number)	This function truncates a number to an integer. It always truncates towards negative infinity. For example <code>int(-4.6)</code> is <code>-5</code> , while <code>fix(-4.6)</code> is <code>-4</code> . For positive numbers the <code>int(</code> and <code>fix(</code> functions are identical.
max(number,number)	This function compares two numbers and returns the larger value. If you need to compare more than two numbers, you can nest this function within itself, for example <code>max(a,max(b,c))</code> .
min(number,number)	This function compares two numbers and returns the smaller value. If you need to compare more than two numbers, you can nest this function within itself, for example <code>min(a,min(b,c))</code> .
randominteger(startnum,endnum)	Returns a random integer value greater than or equal to the startnumber and less than or equal to the end number.
round(number,step)	This function rounds a number to the nearest step. You can use any value you want for the step: <code>1</code> , <code>10</code> , <code>0.5</code> , whatever.  For example, you could use the formula <code>round(Quantity,12)</code> to round the quantity to the nearest dozen. The quantity <code>16</code> will be rounded to <code>12</code> ; the quantity <code>20</code> will be rounded to <code>24</code> .
zeroblank(number)	This function tells Panorama to store zero as an empty space. If the final formula result is not zero, this function has no effect. The <code>zeroblank(</code> function is handy when you want to leave the result of a calculation blank if one of the operands are blank. For example, if you use the formula <code>zeroblank(Qty*Price)</code> , the result will be empty if either the quantity or price is empty.

## Scientific Functions

These functions perform various log, trig, and exponential calculations. Each of these functions takes one or more numeric parameters and returns a numeric result. For more information about these functions see Chapter 23 of the Panorama Handbook.

## Financial Functions

These functions calculate financial data, including loan payments, future value, and present value. They are designed to be compatible with the same functions in Microsoft Excel®. The financial functions are based on the following formula.

$$pv(1+rate)^{periods} + payment(1+rate \times begin) \times ((1+rate)^{periods}-1)/rate + fv = 0$$

Function	Reference Page	Description
<code>pmt(rate,periods,amount,fv,begin)</code>	<a href="#">Page 5604</a>	<p>This function calculates the periodic payment required to pay off a loan. The <b>rate</b> is the interest rate of the loan per period. <b>Periods</b> is the term of the loan expressed in payment periods, for example 36 months for a three year loan that is paid monthly. <b>Amount</b> is the amount being borrowed. The <b>fv</b> (future value) and <b>begin</b> values are optional, and should usually be set to zero.</p> <p>For example, suppose you are taking out a 36 month loan of \$20,000 to buy a car. If the annual interest rate is 13.5% (1.125% compounded monthly), what would the monthly payment be?</p> <pre>pmt( 0.135/12 , 36 , 20000 , 0 , 0 )</pre> <p>The monthly payment is \$678.71.</p>
<code>fv(rate,periods,payment,pv,begin)</code>	<a href="#">Page 5283</a>	<p>This function calculates the future value of an investment. <b>Rate</b> is the interest rate per period. <b>Periods</b> is the term of the investment, for example ten years or 48 months. The <b>pv</b> is the present value of the investment, for example the starting balance in a savings account. <b>Begin</b> should be either 1 or 0; 1 if the payments occur at the beginning of the period, 0 if the payments occur at the end of the period.</p> <p>For example, to calculate the final balance in a savings plan when you invest \$500 per year for 10 years at 9% annual interest use the formula—</p> <pre>fv( 0.09 , 10 , -500 , 0 , 1 )</pre> <p>At the end of ten years you would have \$8280.15. What if this savings plan already has \$2000 in it at the time you start this 10 year savings program? The new formula would be—</p> <pre>fv( 0.09 , 10 , -500 , -2000 , 1 )</pre> <p>At the end of 10 years you would have \$13,014.87.</p>
<code>pv(rate,periods,payment,fv,begin)</code>	<a href="#">Page 5621</a>	<p>This function calculates the present value of an investment. <b>Rate</b> is the discount rate, <b>periods</b> is the periodic investment, and <b>payment</b> is the periodic payment. The <b>fv</b> is an optional lump sum at the end of the final period; use zero if there is no lump sum. <b>Begin</b> specifies whether payments are received at the beginning or end of each period—1 for beginning or 0 for end.</p> <p>Present value is a variation of the old theme that a bird in the hand is worth two...well, you know. It's better to get \$1000 now instead of \$1000 next year, but how much better? The present value computation puts a numeric value on time and money.</p> <p>For example, suppose you find an investment opportunity that promises to pay you \$1,000 per year for the next 3 years. Assuming the current interest rate is 10% per year, how much are these payments worth right now?</p> <pre>pv( 0.1 , 3 , 1000 , 0 , 0 )</pre> <p>The computation shows that \$3000 paid over 3 years is worth \$2486 right now (assuming 10% interest).</p>

## Text Formulas

Formulas can work on text as well as numbers. Formulas can combine two or more pieces of text, extract a portion of a piece of text (for example the area code or last name), or even re-arrange the text. Formulas can also convert numbers into text and back again.

Programmers call a piece of text a **string**, referring to the fact that the text is made up of a string of characters. Since this is such a handy term we'll use it ourselves. So whenever you see the word **string** think "piece of text."

Where do strings come from? Most strings come from the database itself. Any text or choice field can be used as a string. You can also store strings in a variable (see "[Variables](#)" on page 521), or put a string right into the formula itself (see "[Constants](#)" on page 519).

### Gluing Strings Together

The simplest operation that can be performed on two strings is sticking them together, also called **concatenation**. To glue strings together use the + operator. This operator attaches the string on the right to the end of the string on the left. For example the formula

```
"abc"+"def"
```

produces the result **abcdef**. To attach the word **Mr.** to the beginning of a last name field use the formula

```
"Mr. "+<Last Name>
```

(Of course, you better be sure everyone in the database is a man!).

You can use more than one + operator to stick several strings together at once. For example to combine separate first and last names into a single string using the format **Last, First** use this formula:

```
<Last Name>+", "+<First Name>
```

Another way to glue strings together is with the **sandwich()** function ([reference page 5689](#)). This function combines up to three items of text: a **prefix**, a **suffix**, and the **root** text. The **prefix** and **suffix** are slapped on the ends of the **root**, just like a sandwich. However, if the **root** is empty (sort of like a sandwich with no meat!) the **prefix** and **suffix** are also left off, just as you wouldn't bother to make a sandwich without any meat.

Let's revisit our previous example with the **sandwich()** function. The previous formula will work fine as long as there is a first name. But if the first name is empty, the formula will produce an extra comma, for example **Jones, .** The sandwich function can solve this problem:

```
<Last Name>+sandwich(", ",<First Name>,"")
```

If the **First Name** field contains a name, the **sandwich()** function will slap the prefix in front of the name (in this case the prefix is a comma and a space). But if the **First Name** field is empty, the sandwich() function will also leave off the prefix. All the formula will produce is the **Last Name**, with no extra comma and space.



## Functions for Taking Strings Apart

These functions return portions of a string. See also “[Taking Strings Apart \(Text Funnels\)](#)” on page 533, “[String Modification Functions](#)” on page 535, “[Text Arrays](#)” on page 539 and “[Date Arithmetic](#)” on page 544.

Function	Reference Page	Description
<code>after(text,tag)</code>		This function extracts all of text after a specified tag (sequence of characters). If the tag doesn't exist within the text the function returns "".
<code>before(text,tag)</code>		This function extracts all of text before a specified tag (sequence of characters). If the tag doesn't exist within the text the function returns "".
<code>firstline(string)</code>		This function extracts the first line from the text.
<code>firstword(string)</code>		This function extracts the first word from the text (the text up to the first space).
<code>lastline(string)</code>		This function extracts the last line from the text.
<code>lastword(string)</code>		This function extracts the last word from the text (the text from the last space to the end).
<code>left(string,len)</code>		Extracts characters from the left edge of the text. For example <code>left(text,2)</code> extracts the leftmost two characters.
<code>mid(string,len)</code>		Extracts characters from the middle of the text. For example <code>mid(text,6,4)</code> extracts four characters starting with the sixth character.
<code>nthline(string,num)</code>		This function extracts the nth line from the text. For example <code>nthline(text,4)</code> extracts fourth line.
<code>nthword(string,num)</code>		This function extracts the nth word from the text. For example <code>nthword(text,7)</code> extracts seventh word.
<code>removeprefix(text,prefix)</code>		This function checks to see if a text item starts with a prefix. If it does, the prefix is removed.
<code>removesuffix(text,suffix)</code>		This function checks to see if a text item starts with a suffix. If it does, the suffix is removed .
<code>right(string,len)</code>		Extracts characters from the right edge of the text. For example <code>right(text,7)</code> extracts the rightmost seven characters from the text.
<code>snip(string,startposition,count)</code>		This function removes (snips!) one or more characters from the middle of an item of text. The startposition specifies the first character removed, the count is the number of characters to remove. (Note: This function requires the startposition to be a positive number.) If count is -1 then all the text from the start position to the end of the text is snipped, otherwise the count must be a positive number.
<code>textafter(string,tag)</code>		This function extracts the text after the tag. The tag many be one or more characters long. If the tag doesn't occur in the text then the entire original string is returned. For example <code>textafter("someone@isp.net","@")</code> will return <code>isp.net</code> .
<code>textbefore(string,tag)</code>		This function extracts the text before the tag. The tag many be one or more characters long. For example <code>textbefore("someone@isp.net","@")</code> will return <code>someone</code> . If the tag doesn't occur in the text then the entire original string is returned.
<code>trim(string,len)</code>		This function removes characters from the right edge of the text. For example <code>trim(text,4)</code> removes the last four characters from the text.
<code>trimleft(string,len)</code>		This function removes characters from the left edge of the text. For example <code>trimleft(text,2)</code> removes the first two characters from the text.

## Taking Strings Apart (Text Funnels)

Sometimes you may have an item of text where you only need a portion of the text and want to strip off the beginning and or the end of the text. In addition to the functions in the previous section Panorama has a special tool for stripping off the ends of a text item. This tool is called a **text funnel**. Text funnels are powerful tools, however, many users find them a bit difficult to figure out. In recent years we've added many functions that can perform most of the operations that a text tool can perform. Before deciding to use a text funnel you may want to check out "[Functions for Taking Strings Apart](#)" on page 532, "[String Modification Functions](#)" on page 535, "[Text Arrays](#)" on page 539 and "[Date Arithmetic](#)" on page 544.

A text funnel is used a bit differently than other Panorama functions and operators. The text funnel always follows the text item that is being "stripped." In a sense a text funnel has three parameters, the text item, start, and end. But as you can see below, these parameters are arranged quite differently than they are for other functions:

```
<text item>[<start>,<end>]
```

The first parameter, **text item**, is the item of text which will be stripped to get the final result. This may be a field, a variable, or an entire formula (as long as it produces a text item as its final result). If you use an entire formula you should put parentheses around the formula.

The second parameter, **start**, specifies the first character you want to include in the final output. For example if you want to strip off the first three characters the start should be 4 (because the 4th character is the first one we want to keep). If the starting position is past the end of the text all the text will be stripped out and the formula is left with an empty text item.

The third parameter, **end**, specifies the last character you want to include. For example, if you want to strip off everything after the 12th character, the end should be 12. If the starting position is after the ending position, all the text will be stripped and the formula is left with an empty text item.

The real trick in setting up text funnels is deciding what the start and end parameters should be. The following sections will describe several techniques for setting up these parameters.

### Numeric Start and End Positions

The simplest way to specify starting and ending positions is with a number. Positive numbers are counted from the beginning of the original text item (1 is the first character in the original text item). Negative numbers are counted from the last character of the original text item (-1 is the last character).

Our first example removes the first character from the **Notes** field.

```
Notes[2,-1]
```

The next example does the exact opposite—it removes the last character from the **Notes** field.

```
Notes[1,-2]
```

By using the same number for the start and end a text funnel can strip out a single character. The procedure below uses the text funnel **[1,1]** to check to see if the first character of the phone number is a (. If so, it uses another text funnel to strip out the area code.

```
if Phone[1,1]="("
  AreaCode=Phone[2,4]
endif
```

A procedure can use a variable to pre-load the start and end positions. The procedure below will strip out everything starting with the phrase **Private Notes Below ---**.

```
local X
X=search(Notes,"Private Notes Below ---")
if X≠0
  PublicNotes=Notes[1,X-1]
else
  PublicNotes=Notes
endif
```

### Specifying Numeric Length Instead of Position

An alternate form of text funnel allows you to specify the length of the text to be stripped out, instead of the ending position. This alternate form simply uses a semicolon instead of a comma:

```
<text item>[<start>;<length>]
```

The **length** specifies the number of characters from the starting position. A positive length means that the stripped text begins at the starting position and extends to the right. A negative length means that the stripped text begins at the starting position and extends to the left. The character at the starting position is always included (unless the length is zero).

Let's look at two examples of this technique. The first extracts the area code from a long distance phone number.

```
Phone[2;3]
```

The next example strips out the local phone number (the last 8 characters).

```
Phone[-1;-8]
```

If the original text item is too short to fulfill the request the text funnel will take whatever it can get. For example, if the phone number is only 3 characters long, the value in **LocalNumber** will be 3 characters long.

### String Testing Functions

These functions return information about the content of a string.

Function	Description
length(string)	This function counts the number of characters in a string. The result is an integer. If the string is empty, the result will be zero.
linecount(string)	This function counts the number of lines in the text.
rangecontains(thetext,therange)	This function checks to see if the text contains any characters in the specified range. The range must be a series of character pairs, for example <b>AZ</b> for upper case alphabetic characters, <b>AZaz</b> for upper and lower case, <b>09</b> for numeric digits, etc. If the text contains any characters in the specified range the function returns true, otherwise it returns false. For example, <b>rangecontains(Company,"09")</b> will return true if the company name contains any numeric digits, false if it doesn't.
rangematch(string,range)	This function checks text to see if the text matches the specified range. The range must be a series of character pairs, for example <b>AZ</b> for upper case alphabetic characters, <b>AZaz</b> for upper and lower case, <b>09</b> for numeric digits, etc. If it matches the function returns true, if it doesn't match, it returns false. For example, <b>rangematch(Address,"AZaz09 ")</b> will return true if the address contains only letters, numbers and spaces, false if it contains any other characters.

Function	Description
search(string,phrase)	This function searches through a string looking for a word or phrase. If the search is successful, the function returns the position of the phrase within the string, otherwise the function returns zero. For example, the formula <code>search(Name,"Dr.")</code> will return a non-zero value (usually 1) if the name contains <code>Dr.</code> , or zero if it does not.
sizeof(name)	<p>This function calculates the amount of memory used by a field cell or a variable. <code>Name</code> is the name of the field or variable that you want to calculate the size of. The function returns the number of bytes of memory used by the variable or field cell.</p> <p>The <code>sizeof()</code> function can be used to decide if a numeric or date field is empty or not. The example procedure shown below selects all the records with no price (not the same as records with a price of zero).</p> <pre>select sizeof(Price)=0</pre> <p>Another use for the <code>sizeof()</code> function is to check if a variable is taking up too much scratch memory (see “<a href="#">Changing Scratch Memory Size (Macintosh OS 9)</a>” on page 158). This example checks to see if the variable <code>importLetter</code> is more than 500 bytes long. If it is, the procedure clears the variable.</p> <pre>if sizeof(importLetter)&gt;500   importLetter="" endif</pre>
wordcount(string)	This function counts the number of words in the text.

### String Modification Functions

These functions modify the contents of a string. Usually the string is actually a database field. Remember, to use a database field as a string parameter simply use the name of the field, for example `upper(Name)`. You'll often want to use these functions to modify the existing data in a field. For example, you might want to convert all company names to upper case. To convert existing data use the **Formula Fill** command in the Math Menu (see “[Filling a Field with a Formula](#)” on page 431). This command calculates the formula over and over again—once for each selected record.. Note: In addition to the functions listed here you will also find methods for modifying strings in “[Functions for Taking Strings Apart](#)” on page 532, “[Taking Strings Apart \(Text Funnels\)](#)” on page 533, “[Text Arrays](#)” on page 539 and “[Date Arithmetic](#)” on page 544.

Function	Description
connect(prefix,connector,suffix)	This function appends a prefix and suffix together with a connector in between. If either the prefix or the suffix is missing then the connector will also be left out. For example, <code>connect(City," ",State)</code> combines the city and state with a comma and space in between, but if either the city or state is missing then the comma and space will also be left out. See also the <code>sandwich()</code> and <code>yoke()</code> functions in this table.
defaulttext(text,default)	This function returns the text value supplied in the first parameter. However, if this text value is empty (" ") the function will return the specified default value.
fixedwidth(string,width)	This function makes the text a fixed width. If the text is shorter than the specified width, it is padded with spaces. If it is longer than the specified width, it is cut off.
fixedwidthright(string,width)	This function makes the text a fixed width. If the text is shorter than the specified width, it is padded with spaces on the left (i.e. the text is right justified). If it is longer than the specified width, it is cut off on the left.
linestrip(text)	This function removes any blank lines from the text.



Function	Description
<code>lower(string)</code>	This function converts all of the letters in the string to lower case. For example, the formula <code>lower(Terms)</code> will convert <code>NET 30</code> to <code>net 30</code> , or <code>C.O.D.</code> to <code>c.o.d.</code> See also the upper and upperword functions.
<code>obscuredigits(number,count)</code>	This function obscures digits (usually a credit card number) with X's. The first parameter is the text that contains the digits. The second parameter is the number of digits on the end that will NOT be obscured. For example the formula <code>obscuredigits("1234-5678-9876-5432",4)</code> will produce the value <code>XXXX-XXXX-XXXX-5432</code> . Notice that the function retains any additional formatting in the text, in this case dashes.
<code>onespace(string)</code>	This function removes any extra spaces between words, so that there is exactly one and only one space between each word.
<code>onewhitespace(string)</code>	This function removes any extra whitespace between words, making sure that there is one and only one space between each word. Other whitespace characters (carriage returns, tabs) are converted to spaces and removed if there is more than one between words.
<code>padzero(text,width)</code>	This function makes the text a fixed width. If the text is shorter than the specified width, it is padded with 0's on the left (i.e. the text is right justified). If it is longer than the specified width, it is cut off on the left..
<code>rep(string,count)</code>	This function replicates a string over and over. The number of replications is specified by the count (a number). This function is handy for creating a long repeating string. For example to create a string containing twenty asterisks in a row, use the formula <code>rep("*",20)</code> . The count does not have to be a constant, but it must be an integer.
<code>replace(string,search,replace)</code>	This function searches for a word or phrase within a string and if found, replaces it with a new word or phrase. The first parameter is the string that may contain the word or phrase. Usually this parameter is a database field. The second parameter is the word or phrase to search for. The third parameter is the new word or phrase.  For example, to replace <code>Corporation</code> with <code>Corp.</code> in the <code>Client</code> field, use the formula <code>replace(Client,"Corporation","Corp.")</code> . To use this formula to replace the data in the database, use the <b>Formula Fill</b> command. (For a simple replace case like this, however, it is easier to use the <b>Change</b> command. The <code>replace(</code> function is useful when you want to perform other transformations in addition to the <code>replace.</code> )
<code>sandwich(prefix,root,suffix)</code>	The <code>sandwich(</code> function assembles a text item from three smaller text items. The <code>prefix</code> and <code>suffix</code> are slapped on the ends of the <code>root</code> , just like a sandwich. However, if the root is empty, the prefix and suffix are also left off (the result is an empty text item), just as you wouldn't make a sandwich without any meat.  Suppose you have a database with names and titles, and you want to display this information in a report with the titles surrounded by parentheses. The formula below could be used with an auto-wrap text object or Text Display SuperObject.  <code>Name+sandwich(" (",Title," ")</code>  If the person has a title it will appear in parentheses like this: <code>Steve Johnson (Sales Mgr)</code> . If they don't have a title then no parentheses will appear. The <code>sandwich(</code> function is useful any time you have optional data items combined together with punctuation in between. See also the <code>connect(</code> and <code>yoke(</code> functions in this table.
<code>strip(text)</code>	This function strips off leading and trailing blanks and other whitespace (carriage returns, tabs, etc.) This function has one parameter, the item of text that you want to strip. The function removes blanks at the beginning or end of the text, but does not affect blanks in the middle of the text. It also removes carriage returns, tabs, or any character with an ASCII value less than 32.

Function	Description
stripchar(text,range)	<p>This function removes characters you don't want from a text item. You specify exactly what kinds of characters you want and don't want included in the final output. <b>Text</b> is the item of text that you want to strip. <b>Range</b> specifies what kinds of characters you want to keep and what kinds of characters you want to strip away. The <b>range</b> consists of one or more pairs of characters. Each pair specifies a set of characters you want to keep. For example, the pair <b>AZ</b> means that you want to keep the characters from <b>A</b> to <b>Z</b>. For alphanumeric characters the set is pretty obvious. For other types of characters you should check an ASCII chart (see "<a href="#">ASCII Character Constant Functions</a>" on page 538). For example the pair <b>#&amp;</b> specifies a set of four characters: <b>#</b>, <b>\$</b>, <b>%</b> and <b>&amp;</b>. You can use the <b>ASCII Chart</b> wizard to try out your character ranges.</p> <p>If a pair consists of the same character repeated twice in a row, the set is just that single character. For instance the pair <b>##</b> means you want to keep one character: <b>#</b>.</p> <p>The range may consist of several pairs put together. For example the range <b>AZaz09..</b> consists of four pairs, and specifies that all letters, numbers, and periods will be kept, with all other characters stripped away.</p> <p>One handy use for this function is to quickly check if a field or variable contains any inappropriate characters. If a field or variable changes when you run it through the stripchar( function it must contain characters that are not part of the specified range.</p>
striptmltags(text)	This function removes all HTML tags from the text.
stripprintable(text)	This function removes any non-displayable characters from the text.
striptoalpha(text)	<p>This function removes everything but alphabetic letters from a text item. Everything else (numbers, spaces, punctuation, non-English letters, etc.) will be removed from the text.</p> <p>One handy use for this function is to quickly check if a field or variable contains all alphabetic characters. If a field or variable changes when you run it through the striptoalpha( function it must contain non-alphabetic characters.</p>
striptonum(text)	<p>This function removes everything but numeric digits from a text item. Everything else (letters, spaces, punctuation, non-English letters, etc.) will be removed from the text.</p> <p>One handy use for this function is to quickly check if a field or variable contains all numeric digits. If a field or variable changes when you run it through the striptonum( function it must contain non-numeric characters.</p>
upper(string)	This function converts all of the letters in the string to upper case. For example, the formula <b>upper(Terms)</b> will convert <b>net 30</b> to <b>NET 30</b> , or <b>c.o.d.</b> to <b>C.O.D.</b> See also the lower( and upperword( functions.
upperword(string)	The upperword( function converts the first letter of each word in the string to upper case, and all other letters to lower case. For example the formula <b>upperword(State)</b> will convert <b>new york</b> to <b>New York</b> , or will convert <b>VERMONT</b> to <b>Vermont</b> . See also the lower and upper functions.
yoke(prefix,joiner,suffix)	This function appends two text items (prefix and suffix) together. If both are non-blank, a joiner is placed in between. If either (or both) is blank, the joiner is not used. In some ways this is the reverse of the <b>sandwich(</b> function.



## Converting Between Numbers and Strings

These functions convert numbers into strings and strings into numbers.

Function	Description
<code>asc(string)</code>	This function converts the first character of the string into a number based on the ASCII value of the character. For example the formula <code>asc("Y")</code> returns the value 89, while <code>asc("Z")</code> returns the value 90. See also the <code>chr()</code> function.
<code>chr(number)</code>	This function converts a number into a single character of text based on the ASCII value of the number. The number should be an integer between 0 and 255. For example, the letter A has an ASCII value of 65, while the letter B is 66. You can create special characters with this function; TAB is 9 and RETURN is 13. See also the <code>asc()</code> function.
<code>dollarsandcents(number)</code>	This function converts a number to text formatted as dollars and cents (for example 98123.45 becomes <b>Ninety eight thousand one hundred twenty three dollars and 45 cents</b> ).
<code>money(number)</code>	Converts a number to text, formatted with commas every three digits and two digits after the decimal point (for example <b>98,123.45</b> ).
<code>nth(number)</code>	This function converts a number into an ordinal, i.e. 1=1st, 2=2nd, 3=3rd, 4=4th, etc.
<code>pattern(number,string)</code>	This function converts a number into text, using the string as an output pattern. For example the formula <code>pattern(Price,"\$#.,##")</code> will convert the price 3458.23 into the string <b>\$3,458.23</b> . The pattern adds the \$ and the comma. For more information on numeric output patterns see " <a href="#">Numeric Output Patterns</a> " on page 256.
<code>str(number)</code>	This function converts a number into text without any special formatting. If you want to format the number (add commas, set # of digits, etc.) use the <code>pattern()</code> function.
<code>val(string)</code>	This function converts a string into a number. The string must start with one or more numeric digits. Everything after the first non-numeric character will be ignored. For example, the formula <code>val(Address)</code> will return the number 731 if the address is <b>731 N. Miller St.</b>
<code>zbpattern(number,pattern)</code>	This function displays a number using a pattern. Unlike the normal <code>pattern()</code> function, the <code>zbpattern()</code> function will output "" if the number is zero. (Note: zb is short for zeroblank.)

## ASCII Character Constant Functions

These functions return common ASCII characters.

Function	Description
<code>info("lineseparator")</code>	This function returns the line separator character on the current platform. On Macintosh systems this is a carriage return. On Windows PC systems this is a carriage return followed by a linefeed (CR-LF).
<code>cr()</code>	This function generates a carriage return. This is equivalent to <code>chr(13)</code> and is also the same as ¶.
<code>crlf()</code>	This function generates a carriage return line feed. This is equivalent to <code>chr(13)+chr(10)</code> .
<code>lf()</code>	This function generates a line feed. This is equivalent to <code>chr(10)</code> .
<code>tab()</code>	This function generates a tab character. This is equivalent to <code>chr(9)</code> and is also the same as ↹.
<code>vtab()</code>	This function generates a vertical tab character. This is equivalent to <code>chr(11)</code> .

## Text Arrays

An array is a numbered collection of data items. Panorama includes a number of functions and statements that treat a single text data item as if it were a numbered collection of smaller items. The smaller text data items must be separated from each other by a delimiter, for instance a comma or carriage return.

Consider the text data item shown below. Panorama would normally treat this as a single item with a length of 40 characters. The functions described in this section, however, can treat this text as a collection of 7 elements separated by semicolons.

```
white;red;orange;yellow;green;blue;black
```

In this example, the ; is the separator character. You can use any character you want for a separator character, in fact, you can use different separator characters at different times. You could even build a multi-level array by using two different separator characters.

Using the array functions and statements provided by Panorama you can extract elements from an array, change array elements, even sort an array. Since arrays are really text, they can be stored in any variable or any text field, and they can be edited with the data sheet, a data cell, or a Text Editor SuperObject.

There are many statements and user interface elements that work with text arrays, including lists and pop-up menus. There are also a number of functions that generate text arrays, including functions for building lists of files, windows, fields, choices, and data. Most of these statements, user interface elements, and functions require that carriage returns be used as separators, so that each array element is on a separate line.

It is up to you to keep track of the fact that you are using an array and what the separator character is. Panorama won't stop you from trying to access the array of colors above as if it were delimited with commas instead of semicolons, but you probably won't get the results you wanted unless you use the correct separator character.

(If you are familiar with the arrays in C or Pascal, Panorama text arrays are quite a bit different, although both are a numbered collection of items. As with anything unfamiliar, Panorama text arrays probably won't look as good as the ones you are used to at first. Panorama arrays do have some significant advantages though: they don't have to be declared in advance, each array element can be of unlimited length without wasting space, and Panorama arrays can be directly edited. It's also very easy to "pre-fill" a Panorama array with a list of values.)

### Picking a Separator Character

Any ASCII character can be used as a separator character, so you have 256 possible choices. Common separators include comas, semicolons, slashes, carriage returns, spaces and tabs.

It's important to pick a separator character that will not occur in the data elements of your array. If your data may include commas, don't use the comma as a separator character. If the data might include carriage returns, don't use a carriage return. If you want to be extra sure to avoid conflicts, pick a non-printing character. You can use the `chr(` function ([reference page 5099](#)) to generate non-printing characters, for example `chr(1)`, `chr(2)`, `chr(3)`. Most `chr(` values below 32 are non-printing except for `chr(9)` and `chr(13)`, which correspond to tab and carriage return.

Some Panorama user interface elements and functions use text arrays as parameters or to hold a list of values. For these applications the separator character is usually required to be a carriage return. For example, the Pop-Up Menu SuperObject uses a carriage return delimited array to define the list of pop-up menu choices. The `lookupall(` function extracts information from another database and places it into an array with whatever separator you specify. Consult the documentation for each individual statement, function or SuperObject to see the exact specifications for any arrays they may use.

## Working With Arrays

Panorama has about a dozen functions and procedure statements for working with arrays. These functions are described in this table.

Function	Description
array(text,item,sep)	<p>This function extracts a single data item from a text array. <b>Text</b> is the item of text that contains the data you want to extract. <b>Item</b> is the number of the data item you want to extract. The first item is item 1, the second is item 2, the third item is 3, etc. <b>Separator</b> is the separator character for this array. This should be a single character. For carriage return delimited arrays, use the ¶ character (see “<a href="#">Special Characters</a>” on page 525). For tab delimited arrays use the – character.</p> <p>The array( function returns a single item of text from the array. Only the item itself is returned, the separator characters on each end are not included. If the item does not exist (for example if you ask for item 12 from a 7 item array) the function will return empty text (“”).</p> <p>There are 7 VHF television stations in Los Angeles. The example procedure below will convert channel numbers into the names of the stations. For example, the procedure converts Channel 7 into <b>KABC</b>.</p> <pre>Stations=" ,KCBS, ,KNBC,KTLA, ,KABC, ,KCAL, ,KTTV, ,KCOP" «Channel Name»=array(Stations,7," ,")</pre> <p>The example uses an array called <b>Stations</b>. This array uses commas as a separator character.</p>
arrayboth(a1,a2,sep)	<p>This statement compares two arrays. The result is a list of elements that are included in both arrays. Note: Empty array elements, if any, will be ignored. Both arrays must use the same separator.</p>
arraychange(text,value,item,sep)	<p>This function changes a single value inside a text array. Only the one item is changed, all the other items in the array remain the same. <b>Text</b> is the text array that contains the data you want to change. <b>Value</b> is the new value of the data item. <b>Item</b> is the number of the data item you want to change. Items are numbered starting from 1 (1,2, 3,...). This item must already exist in the array. The arraychange( function will not add the item if it does not exist. <b>Sep</b> is the separator character for this array. This should be a single character. For carriage return delimited arrays, use the ¶ character (see “<a href="#">Special Characters</a>” on page 525). For tab delimited arrays use the – character. This function returns a copy of the text array, with the data item changed. If you want to change the original array you should use an assignment statement (see below).</p> <p>The example procedure below will change the 5th item of the array to Navajo White.</p> <pre>Colors=arraychange(Colors,"Navajo White",5,";")</pre> <p>This example assumes that a field or variable named <b>Colors</b> already exists.</p>
arraycontains(text,item,sep)	<p>This function checks to see if any element of an array matches the specified text. For the result to be true, the array element must match the specified text exactly, including upper and lower case. Otherwise the function will return false. So checking for “<b>Green</b>” will only match that exact array element, not “<b>green</b>” or “<b>Olive Green</b>”. (Note that this is quite different from the <b>contains</b> operator, which ignores upper and lower case and allows a submatch.)</p>
arraydeduplicate(text,separator)	<p>This function removes duplicate values from an array. As a byproduct it also sorts the array.</p>

Function	Description
arraydelete(text,item,count,sep)	<p>This function deletes one or more elements from the middle of a text array. <b>Text</b> is the text array that you want to insert elements into. <b>Item</b> is the spot where you want the elements to be deleted. <b>Count</b> is the number of elements you want to delete from the array. <b>Sep</b> is the separator character for this array. This should be a single character. For carriage return delimited arrays, use the ¶ character (see “<a href="#">Special Characters</a>” on page 525). For tab delimited arrays use the ␣ character. This function returns a copy of the original text array, with the specified elements deleted from the middle. The example procedure below will delete the 3rd item from the <b>SpeedDial</b> array:</p> <pre style="text-align: center;">SpeedDial=arraydelete(SpeedDial),3,1,¶)</pre>
arraydeletevalue(text,value,sep)	<p>This function deletes any array elements that match the value parameter. This must be an exact match, including upper and lower case. If the value occurs multiple times in the array, every occurrence of the value will be removed, with one exception: If the value occurs in two consecutive array elements, only the first occurrence will be deleted.</p>
arraydifference(a1,a2,sep)	<p>This statement compares two arrays. The result is a list of elements that are in the first array but not the second. (Note: Empty array elements, if any, will be ignored.) Both arrays must use the same separator.</p>
arrayfirst(text,sep)	<p>This function extracts the first element of an array.</p>
arrayinsert(text,item,count,sep)	<p>This function inserts one or more elements into the middle of a text array. <b>Text</b> is the text array that you want to insert elements into. <b>Item</b> is the spot where you want the new elements to be inserted. <b>Count</b> is the number of blank elements you want to insert into the array. <b>Sep</b> is the separator character for this array. This should be a single character. For carriage return delimited arrays, use the ¶ character (see “<a href="#">Special Characters</a>” on page 525). For tab delimited arrays use the ␣ character.</p> <p>This function returns a copy of the original text array, with the new blank array elements inserted into the middle. The example procedure below will add 5 new array items to the <b>SpeedDial</b> array between the 2nd and 3rd array items:</p> <pre style="text-align: center;">SpeedDial=arrayinsert(SpeedDial),¶,3,5)</pre> <p>The new array items created by arrayinsert( are blank (empty). You can fill them in with the arraychange( function.</p>
arraylast(text,sep)	<p>This function extracts the last element of an array.</p>
arraylefttrim(text,count,sep)	<p>Removes the first elements of an array. For example <b>arraylefttrim(text,2,",")</b> removes the first two elements from a comma separated array.</p>
arraynotcontains(text,item,sep)	<p>This function is the reverse of the arraycontains( function.</p>

Function	Description
arrayrange(text,start,end,sep)	<p>This function extracts a series of data item from a text array. <b>Text</b> is the item of text that contains the data you want to extract. <b>Start</b> is the number of the first data item you want to extract. Items are numbered starting from 1 (1, 2, 3,...). <b>End</b> is the number of the last data item you want to extract. Items are numbered starting from 1 (1, 2, 3,...). <b>Sep</b> is the separator character for this array. This should be a single character. For carriage return delimited arrays, use the ¶ character (see “<a href="#">Special Characters</a>” on page 525). For tab delimited arrays use the ␣ character.</p> <p>This function returns a series of items from the array. It returns the first item, the last item, and everything in between (including any separators that are in between). If the last item does not exist (for example if you ask for item 12 from a 7 item array) the function will return up to the actual last item in the array. If both requested items do not exist, the function will return empty text (“”).</p> <p>This example procedure will fill the variable <b>WeekDays</b> with the text <b>Mon,Tue,Wed,Thu,Fri</b>.</p> <pre>Days="Sun,Mon,Tue,Wed,Thu,Fri,Sat" WeekDays=arrayrange(Days,2,6,"")</pre>
arrayreplacevalue(text,oldvalue,newvalue,sep)	<p>This function replaces any array elements that match the value parameter. This must be an exact match, including upper and lower case. If the value occurs multiple times in the array, every occurrence of the value will be replaced, with one exception. If the value occurs in two consecutive array elements, only the first occurrence will be replaced.</p>
arrayreverse(text,sep)	<p>This function reverses the order of the elements in a text array. In other words, the first element becomes the last element, the second element becomes the second to last, etc. <b>Text</b> is the text array that you want to modify. <b>Sep</b> is the separator character for this array. This should be a single character. For carriage return delimited arrays, use the ¶ character (see “<a href="#">Special Characters</a>” on page 525). For tab delimited arrays use the ␣ character.</p> <p>The arrayreverse( function reverses the order of the elements of an array. For example, the formula:</p> <pre>arrayreverse("1;2;3;4",";")</pre> <p>will produce the array <b>4;3;2;1</b>.</p>
arraysearch(array,text,start,sep)	<p>This function searches a text array to see if it contains a specific value. <b>Array</b> is the text array that you want to search. <b>Text</b> is the text that you want to search for. This parameter may contain the wildcard characters ? and * . For example, to search for array items that start with John use <b>John*</b> . To search for any array item containing Pacific use <b>*Pacific*</b> . The array item must match the text exactly, including upper/lower case. For more information on wildcard characters, see “<a href="#">A match B</a>” on page 553. <b>Start</b> is the spot in the array where you want the search to begin from. If you want to search the entire array, this parameter should be one. <b>Sep</b> is the separator character for this array. This should be a single character. For carriage return delimited arrays, use the ¶ character (see “<a href="#">Special Characters</a>” on page 525). For tab delimited arrays use the ␣ character.</p> <p>If the arraysearch( function finds an array element that matches what you are searching for it returns the number of that array element (1, 2, 3, etc.). If there is no matching element, the function returns 0.</p>



Function	Description
arraysize(text,sep)	<p>This function counts the number of items in a text array. <b>Text</b> is the text array that you want to count. <b>Sep</b> is the separator character for this array. This should be a single character. For carriage return delimited arrays, use the ¶ character (see “<a href="#">Special Characters</a>” on page 525). For tab delimited arrays use the ␣ character.</p> <p>This function returns a number. This is the number of elements in the array. If there is no text in the array, the function will return one. If you need a function that returns zero if there is no text you can use the extract function with the last parameter set to -1 (see “<a href="#">String Modification Functions</a>” on page 535).</p> <p>This example uses the arraysize( function to display the number of forms in the current database. (The dbinfo("forms","") function creates an array listing all the forms in the current database, separated by carriage returns.)</p> <pre>message "This database contains "+ str(arraysize(dbinfo("forms",""),¶))+" forms"</pre>
arraysort(text,separator)	This function sorts an array in alphabetical order.
arraystrip(text,sep)	<p>This function removes any blank elements from a text array. <b>Text</b> is the text array that you want to strip the blank elements from. <b>Sep</b> is the separator character for this array. This should be a single character. For carriage return delimited arrays, use the ¶ character (see “<a href="#">Special Characters</a>” on page 525). For tab delimited arrays use the ␣ character. This function returns a copy of the original text array, with any blank array elements removed from the array.</p>
arraytrim(text,count,sep)	This function removes the last elements of an array. For example <code>arraytrim(text,2,")</code> removes the last two elements from a comma separated array.
makenumberedarray(sep,start,end)	This function generates a numeric sequenced array, for example 1, 2, 3, 4, 5. You can specify the starting and ending number of the sequence.



## Date Arithmetic

Formulas can perform several useful calculations on dates. For example, you can calculate the number of days between two dates, or you can add or subtract a certain number of days to a date. You can also convert a date to text using a wide variety of formats.

Usually we think of a date in terms of years, months, and days. Formulas, however, treat dates as a certain number of days—specifically, the number of days between that date and January 1, 4713 B.C., adjusted for the Gregorian calendar correction in October 1582. (The date 4713 B.C. is chosen for obscure astronomical reasons). For example, to a formula the date August 7, 1991 is day number 2,448,476.

Fortunately you should never have to worry about numbers like 2,448,476. The formula will automatically convert a date field into the number of days, perform the calculation, and then convert back into a regular date again.

Since formulas handle dates as numbers, you can use any numeric operator or function to manipulate dates. However it doesn't make much sense to take the square root of a date (although Panorama will let you). There are really only two numeric operations that make sense on dates—subtracting two dates to find the number of days in between and adding or subtracting a number of days to a date.

To calculate the number of days between two dates, just subtract one from the other. For example, the formula

```
<Ship Date>-<Order Date>
```

will calculate the number of days required to process an order.

To calculate an offset from a given date, just add the number to the date. For example the formula

```
<Ship Date>+30
```

calculates the normal due date 30 days after the ship date.

## Today's Date

The `today()` function ([reference page 5862](#)) returns the number corresponding to today's date, allowing you to use today's date in a formula. For example, to calculate the age of an invoice use a formula like this.

```
today()-<Ship Date>
```

To calculate the due date for a library book, use the formula like this.

```
today()+14
```

This formula assumes that books are checked out for two weeks.

## Converting Between Dates and Text

These functions allow you to convert a date into text, or text into a date. You should only use these functions if you want to store the result of a date calculation in a text field instead of a date field, or if you want to access a date that has been stored as text.

Note: Remember, formulas handle dates as numbers, so these functions actually convert numbers into text and vice versa. It's up to you to make sure that these numbers actually represent the correct dates.

Function	Description
completedatestr(number)	Convert a date to text, including the day of the week (for example <b>Sunday, April 20th, 2003</b> ).
date(text)	<p>This function converts a text string in a date format into the number representing that date. Use this function to include a constant date in your formula, for example <code>date("12/9/1979")</code>. You should also use this function to access dates that have been stored in text fields (but why are you doing that in the first place?).</p> <p>Several formats are supported, including <b>mm/dd/yy</b>, <b>mm/dd/yyyy</b>, <b>Month dd, yyyy</b>, and <b>Mon dd, yyyy</b>. Dates in the current week can be represented by the name of the day, for instance <b>Tuesday</b> or <b>Fri</b>. Dates in the previous or upcoming week can be represented by adding the words <b>last</b> or <b>next</b>, for example <b>last friday</b> or <b>next wed</b>.</p>
datepattern(number,pattern)	<p>This function converts a number representing a date into a formatted text string. The <b>pattern</b> parameter is an output pattern telling the function how to format the date. For more information on date output patterns, see "<a href="#">Date Output Patterns</a>" on page 261.</p> <p>Use the <code>datepattern(</code> function to store a date in a text field, or to display a formatted date in an auto-wrap text object or Text Display SuperObject. For example, the formula:</p> <pre>datepattern(«Ship Date», "Month ddnth, yyyy")</pre> <p>can be used to display the date an order was shipped in the format <b>May 12th, 2003</b>.</p>
datestr(number)	Convert a date to text using format <b>mm/dd/yy</b> (for example <b>4/20/03</b> ).
daystr(number)	Convert a date to the day of the week (for example <b>Sunday</b> ).
eurodatestr(number)	Convert a date to text in European format (for example <b>20-APR-2003</b> ).
exportcell(field)	<p>This function takes any database field and converts it to text, using the appropriate pattern if one has been defined in the design sheet. <b>Field</b> is the name of the field to be converted to text.</p> <p>The function always returns a text type data item. The power of the <code>exportcell(</code> function is that it does not require you to know what type of data you are exporting. It simply takes whatever kind of data is in the field (text, number, date, whatever) and converts it into text.</p>
longdatestr(number)	Convert a date to text with format <b>Month ddnth, yyyy</b> (for example <b>April 20th, 2003</b> ).
naturaldata(date)	This function converts a date to text in a natural format similar to how people would refer to the date, for example Today, Tue, Apr 4. If the date is more than 180 days in the past or is in the future the pattern <b>mm/dd/yy</b> is used. This is similar to how the Apple Finder displays dates..

## Date Functions

These functions perform various calculations and conversions on date values. Unless specified otherwise the date is always processed as a numeric value.

Function	Description
<code>datevalue(year,month,day)</code>	This function converts three integers into a date. The three integers are the year, month and date. This function provides a way to create a date that is independent of the system date settings (the <code>date()</code> function, which can also create dates, will produce different values in different countries depending on the date formats used in those countries).
<code>dayofweek(date)</code>	<p>This function computes the day of the week (0-6) of a date, with Sunday being 0, Monday 1, etc. The function returns a number from 0 to 6. The days of the week are:</p> <pre> 0  Sunday 1  Monday 2  Tuesday 3  Wednesday 4  Thursday 5  Friday 6  Saturday </pre> <p>The procedure below uses the <code>dayofweek()</code> function to select all weekday records (monday through friday).</p> <pre> select dayofweek(Date)≥1 and dayofweek(Date)≤5 </pre>
<code>dayvalue(date)</code>	This function extracts the day of the month from a date as a numeric value (1 to 31).
<code>month1st(date)</code>	<p>This function computes the first day of a month. For example, if the date passed to this function is <b>October 18, 1997</b>, this function will return the date <b>October 1, 1997</b>. The date is returned as a number.</p> <p>The example procedure below uses this function to select the orders placed this month, then displays the count.</p> <pre> select OrderDate≥month1st( today() ) and        OrderDate&lt;month1st(today()+monthlength(today() ) message str(info("records"))+" orders this month" </pre>
<code>monthlength(date)</code>	<p>This function computes the length (number of days) of a month. For example, if the date passed to this function is <b>October 18, 1997</b>, this function will return <b>31</b>, the number of days in October. This function knows about leap years and adjusts the length of February accordingly.</p> <p>The example procedure below uses this function to select the orders placed this month, then displays the count.</p> <pre> select OrderDate≥month1st( today() ) and        OrderDate&lt;month1st(today()+monthlength(today() ) message str(info("records"))+" orders this month" </pre>

Function	Description
monthmath(date,offset)	<p>This function takes a date and computes another date that is one or months before or after the original date. <b>Date</b> is a number representing the original date. <b>Offset</b> is the number of months that you want to add or subtract to the original date. Use a positive number to move forward in time, a negative number to go backwards. For example, if you offset the date <b>May 12, 1997</b> by <b>2</b> (two months forward) the result is <b>July 12, 1997</b>. If you offset the same original date by <b>-2</b> (two months backward) the result is <b>March 12, 1997</b>.</p> <p>If the new date does not exist because a month does not have enough days in it, the monthmath( function will pick the last day of the month. For example, if you offset <b>March 31</b> by 1 month the result is <b>April 30</b>. If the new month lands in February the function knows about leap years and adjusts accordingly.</p> <p>This example calculates a renewal date exactly one year from today.</p> <pre>monthmath(today(),12)</pre>
monthvalue(date)	This function extracts the month from a date as a numeric value (1 to 12).
quarter1st(date)	This function computes the first day of a quarter. For example, if the date passed to this function is <b>August 18, 1997</b> , this function will return the date <b>July 1, 1997</b> . The date is returned as a number.
quartervalue(date)	This function extracts the quarter within a year from a date as a numeric value (1 to 4).
today()	This function returns today's date (assuming, of course, that your computer clock has been set correctly).
week1st(date)	This function computes the first day of a week (Sunday). For example, if the date passed to this function is <b>July 12, 1995</b> (a Wednesday), this function will return the date <b>July 9, 1995</b> (a Sunday). The date is returned as a number.
year1st(date)	<p>This function computes the first day of a year. For example, if the date passed to this function is <b>July 12, 1995</b>, this function will return the date <b>January 1, 1995</b>. The date is returned as a number.</p> <p>The example below calculates the number of days remaining in the current year.</p> <pre>yearfirst(year1st( today( ))+366)-today()</pre>
weekvalue(date)	This function extracts the week from a date as a numeric value (this is the number of weeks since the start of the year, 1 to 52).
yearvalue(date)	This function extracts the year from a date as a numeric value.

## Time Arithmetic

To Panorama, time is not hours, minutes, and seconds, but simply seconds. To be precise, a time is the number of seconds since midnight. For example, the time **4:32 AM** is **16,320** seconds after midnight. As you can see, a Panorama time is really a number in disguise. Since times are numbers, it's easy to compare them, sort them, or find the difference between them (number of seconds).

### Converting Between Times and Text

Unlike dates, Panorama does not automatically provide a time data type that automatically converts a date in text format into a number. You must use a function to convert time in text format into seconds before you can do math calculations with the time, and use another function to convert back.

Function	Description								
<code>now()</code>	This function returns the current time (number of seconds since midnight). Of course the clock on your computer must be set correctly!								
<code>seconds(text)</code>	<p>This function converts text into a number representing a time. The function has one parameter — the text that you want to convert to a number representing a time. If the text includes an AM or PM suffix, the number of seconds is calculated from midnight (12 A.M.), otherwise it is calculated from 0:00:00 (elapsed time). The text must contain a valid time. Here are some examples of valid times:</p> <pre>4:13 PM 11:00 AM 2:30 18:45</pre> <p>This function returns a number representing the time. The number is the number of seconds since midnight. For example, if the time is <b>10:23 AM</b> this function will return the number <b>37,380</b>.</p>								
<code>timepattern(number,pattern)</code>	<p>This function converts a number representing a time into text. The function uses a pattern to control how the date is formatted.</p> <p>The function has two parameters: <b>number</b> and <b>pattern</b>. <b>Number</b> is the number that you want to convert to text. This number must be the number of seconds since midnight. <b>Pattern</b> is text that contains a pattern for formatting the date. The pattern is assembled from four components: <b>hh</b> (hours), <b>mm</b> (minutes) <b>ss</b> (seconds), and <b>am/pm</b>. Some of the more common time patterns are listed here:</p> <table border="1"> <thead> <tr> <th>Pattern</th> <th>Converted Text</th> </tr> </thead> <tbody> <tr> <td>"hh:mm:ss am/pm"</td> <td>4:32:17 pm</td> </tr> <tr> <td>"hh:mm am/pm"</td> <td>4:32 pm</td> </tr> <tr> <td>"hh:mm:ss"</td> <td>16:32:17</td> </tr> </tbody> </table> <p>If <b>am/pm</b> is left off the pattern the time will be formatted in 24 hour format, as shown on the last line of the table above. You should also leave off <b>am/pm</b> for converting elapsed times.</p>	Pattern	Converted Text	"hh:mm:ss am/pm"	4:32:17 pm	"hh:mm am/pm"	4:32 pm	"hh:mm:ss"	16:32:17
Pattern	Converted Text								
"hh:mm:ss am/pm"	4:32:17 pm								
"hh:mm am/pm"	4:32 pm								
"hh:mm:ss"	16:32:17								

Function	Description
time(text)	<p>This function converts text into a number representing a time. The function has one parameter — the text that you want to convert to a number representing a time. The time function allows you to leave out the colons in the time, and also allows you to leave off the am/pm. Here are some examples of valid times:</p> <pre>4:13 PM 11:00 AM 2:30 18:45 230 4p midnight noon afternoon evening night nite</pre> <p>The time( function is very lenient about the format you use to enter the time. It will accept a time without colons, for example <b>425 pm</b> instead of <b>4:25 pm</b>. If there is no am or pm the time function will try to make an intelligent guess. For example, <b>230</b> is almost certainly <b>2:30 pm</b>, not <b>2:30 am</b>. By default, the time( function assumes that any time from 6:00 to 11:59 is AM, and any time from 12:00 to 5:59 is PM, but you can change these assumptions with the timedefaults statement (<a href="#">reference page 5858</a>).</p> <p>The time( function will also convert “named” times: <b>noon</b>, <b>midnight</b>, <b>morning</b>, <b>afternoon</b>, <b>evening</b>, and <b>night</b>. This function assumes that <b>morning</b> is <b>9:00 am</b>, <b>afternoon</b> is <b>1:00 pm</b>, <b>evening</b> is <b>6:00 pm</b>, and <b>night</b> is <b>10:00 pm</b>. These assumptions can be changed with the timedefaults statement (<a href="#">reference page 5858</a>).</p>
timestr(number)	Convert a number to text in am/pm time format (for example <b>9:34 AM</b> ).

## Time Calculations

Once time has been converted into seconds you can perform arithmetic on it. For example, to calculate the number of hours worked from a time card use a formula like this (this formula assumes that **In** and **Out** are text fields containing times).

```
(seconds(Out)-seconds(In))/3600
```

(The division by 3600 converts the result into hours.)

To find out when a task will be finished that takes 2 1/2 hours to complete, use the formula

```
seconds(«Start Time»)+seconds("2:30")
```



Simple addition and subtraction does not compensate for time wrapping around midnight. For example, if you want to calculate the length of a shift that begins at 11 P.M. and ends at 7 A.M., you must add 24 hours to 7AM before subtracting the times. To solve this problem you can use one of the functions described below, or you can use a SuperDate, which combines time and date into a single number (see “[True/False Formulas](#)” on page 552).

Function	Description
time24(time)	<p>This function takes a time and makes sure it falls within a 24 hour period. If the time is less than 24 hours, it is unchanged. If the time is greater than 24 hours, it is converted to the equivalent time in a 24 hour period (for example 30:00:00 is converted to 6:00:00).</p> <p>The time24( function can help with calculations of an ending time from a start time and duration. The basic formula for such a calculation is shown here.</p> $\text{EndTime}=\text{StartTime}+\text{Duration}$ <p>This formula works fine unless the interval extends over midnight. The time24( function adjusts the result to make sure it starts over at zero as it crosses midnight.</p> $\text{EndTime}=\text{time24}(\text{StartTime}+\text{Duration})$ <p>This formula will correctly calculate that 10:30 PM + 4 hours is 2:30 AM.</p>
timedifference(start,end)	<p>This function calculates the difference between two times. It works correctly even if the interval between the two times crosses over midnight. This function returns a time interval between -12 and +12 hours. See also the timeinterval( function, which returns a time interval between 0 and 24 hours.</p> <p>There are two parameters, <b>start</b> and <b>end</b>. <b>Start</b> is a number (number of seconds) representing the starting point of the time interval. <b>End</b> is a number (number of seconds) representing the ending point of the time interval. This function returns the number of seconds between the two times. For example, if the start time is 9:30 PM and the end time is 2:05 AM, the difference would be 4:35. But if the parameters are reversed and the start is 2:05 AM and the end is 9:30 PM, the difference is -4:35. If the result is positive, the end is after the start. But if the result is negative, the start is after the end.</p>
timeinterval(start,end)	<p>This function calculates the time interval between two times. It works correctly even if the interval between the two times crosses over midnight. This function returns a time interval between 0 and 24 hours. See also the timedifference( function, which returns a time interval between -12 and +12 hours.</p> <p>There are two parameters, <b>start</b> and <b>end</b>. <b>Start</b> is a number (number of seconds) representing the starting point of the time interval. <b>End</b> is a number (number of seconds) representing the ending point of the time interval. This function returns the number of seconds between the two times. For example, if the start time is 9:30 PM and the end time is 2:05 AM, the interval would be 4:35. But if the parameters are reversed and the start time is 2:05 AM and the end time is 9:30 PM, the interval is 19:25.</p>

### Time Calculations with Text

Unlike the functions in the previous sections, these functions operate with time values in strings. There aren't as many functions available as for times expressed as numbers, but if your input and output values will be in strings using these function saves the intermediate conversion steps.

Function	Reference Page	Description
texttimeinterval(start,end)		<p>This function calculates the time interval between two times. It works correctly even if the interval between the two times crosses over midnight. This function returns a time interval between 0 and 24 hours. See also the <code>timedifference()</code> function, which returns a time interval between -12 and +12 hours.</p> <p>There are two parameters, <b>start</b> and <b>end</b>. <b>Start</b> is a string representing the starting point of the time interval. <b>End</b> is a string representing the ending point of the time interval. This function returns the time between the start and end. For example, if the start time is 9:30 PM and the end time is 2:05 AM, the interval would be 4:35. But if the parameters are reversed and the start time is 2:05 AM and the end time is 9:30 PM, the interval is 19:25.</p>
texttimedifference(start,end)		<p>This function calculates the difference between two times. Instead of being expressed as numbers, the input output times are expressed as text (for example 12:45 pm). This function works correctly even if the interval between the two times crosses over midnight. This function returns a time interval between -12 and +12 hours. See also the <code>timeinterval()</code> function, which returns a time interval between 0 and 24 hours.</p> <p>There are two parameters, <b>start</b> and <b>end</b>. <b>Start</b> is a string representing the starting point of the time interval. <b>End</b> is a string representing the ending point of the time interval. This function returns the time difference between the start and end. For example, if the start time is 9:30 PM and the end time is 2:05 AM, the difference would be 4:35. But if the parameters are reversed and the start is 2:05 AM and the end is 9:30 PM, the difference is -4:35. If the result is positive, the end is after the start. But if the result is negative, the start is after the end.</p>

## True/False Formulas

In Panorama as in most programming languages, control flow decisions are made on the basis of formulas that are either true or false. The most basic true/false formula compares two values to see if they are equal.

```
PaymentMethod="C.O.D."
```

This formula will compare the value in the field `PaymentMethod` with `C.O.D.` The result will be true if `PaymentMethod` is `C.O.D.`, and false if it contains anything else (for example `Check`, `Cash`, `Visa`, etc.).

## Comparison Operators

Panorama has about a dozen different operators that can compare two values and produce a true false result. You can type these operators in yourself (see “[Special Characters](#)” on page 525), or you can use the **Operator** sub-menu in the Function menu to type in the symbols for you. The table below lists the universal comparison operators. These comparison operators will work with any type of data: text, numeric, or date.

Operator	Example	True/False Meaning	Notes
=	A=B	is A equal to B?	
≠	A≠B	is A not equal to B?	Not available on PC
<>	A<>B	is A not equal to B?	
>	A>B	is A greater than B?	
≥	A≥B	is A greater than or equal to B?	Not available on PC
>=	A>=B	is A greater than or equal to B?	
<	A<B	is A less than B?	
≤	A≤B	is A less than or equal to B?	Not available on PC
<=	A<=B	is A less than or equal to B?	

All of the above operators require that A and B be the same data type. In other words, you cannot directly compare numbers to text, or text to dates. If A and B are different types you must convert them to the same type before comparing them, using the `str()`, `val()`, `pattern()`, `date()` or `datepattern()` functions. See “[Converting Between Numbers and Strings](#)” on page 538 and “[Converting Between Dates and Text](#)” on page 545 for more information on these functions.

Panorama also has a number of specialized comparison operators that work only with the text data type.

Operator	Example	True/False Meaning
beginswith	A beginswith B	does A begins with B?
endswith	A endswith B	does A end with B?
contains	A contains B	does A contain B?
notcontains	A notcontains B	does A not contain B?
soundlike	A soundlike B	does A sound like B (phonetically)?
match	A match B	does A match the wildcard pattern in B (disregarding upper/lower case)?
matchexact	A matchexact B	does A exactly match the wildcard pattern in B?
notmatch	A notmatch B	does A not match the wildcard pattern in B (disregarding upper/lower case)?
notmatchexact	A notmatch B	does A not exactly match the wildcard pattern in B?

Some of these operators deserves a more complete explanation:.

**A soundslike B**

This operator checks to see if the text in A “sounds like” the text in B. For example, the formula below will determine if the `LastName` sounds like the name `Smith`.

```
LastName soundslike "Smith"
```

This formula will be true if the name is `Smith`, `Smyth` or `Smythe`, and false if the name is `Jones` or `Williams`.

The method Panorama uses to determine whether two values sound alike is called “soundex.” This technique is not very exact, and often will produce extra matches that you might not think really sound similar. However, it almost never fails to match on names that do sound similar, so it is a good starting point when you are not sure of an exact spelling.

The soundex technique does require that the first letter of the two values match. For example even though we think they sound alike, `Christy` and `Kristy` will not match because the first letter is different.

**A match B**

This operator checks to see if the text in A matches a pattern you specify in B. The pattern allows you to set up very flexible “wildcard” matches where some characters must match and some don’t have to.

The pattern should combine normal characters, which must match the text in A, and wildcard characters: `?` and `*`. The `?` wildcard character will match any character. The `*` wildcard character (asterisk) will match a variable number of characters. The best way to understand wildcard matches is probably to look at a few examples.

Our first example uses the pattern `j*johnson`. With this pattern the name must begin with `j` (or `J`) and end with `johnson` (or `Johnson`, etc.) The characters in between don’t matter.

```
Name match "j*johnson"
```

This formula will produce a true result for names like `Jim Johnson`, `Jack Johnson`, `Joe Johnson`, etc. The formula will also be true for names like `J346 Ujohnson` or `J@#opcjohnson`.

The second example uses the pattern `926???`. With this pattern the zip code must begin with `926` and must be 5 digits long. (Our example assumes that `ZipCode` is a text field, not a numeric field.)

```
ZipCode match "926???"
```

This formula will produce a true result for zip codes like `92631` or `92685` but a false result for zip codes like `89324` or `92685-0301`. Here’s a variation that will work with 5 or 9 digit zip codes. The `??` characters mean that there must be at least five digits, while the `*` means that any extra characters are ok.

```
ZipCode match "926???"
```

This formula will produce a true result for zip codes like `92631`, `92685` or `92685-0301`, but a false result for `926` or `9262`.

Don’t forget that a space is a normal character. The example below checks for people with a middle initial. The pattern looks for any number of characters followed by a space, followed by a single character, followed by a period, followed by another space, followed by any number of characters.

```
Name match "* ? . *"
```

This formula will produce a true result for `Robert E. Lee` or `Winston O. Link`, but a false result for `Frank Tesh`, `Billy Martin`, or `Sara Jessica Parkman`.

The match operator can be used to simulate the beginswith, endswith and contains operators. The table below shows the equivalent match formulas for each of these operators.

These formulas...	are the same as these.
<code>A match B+"*"</code>	<code>A beginswith B</code>
<code>A match "*" + B</code>	<code>A endswith B</code>
<code>A match "*" + B + "*"</code>	<code>A contains B</code>

Note: The match operator does not worry about upper or lower case. If upper and lower case are important to you, use the matchexact operator.

#### A matchexact B

This operator checks to see if the text in A matches a pattern you specify in B. This operator works exactly the same as the match operator, except that the normal characters must match exactly, including upper and lower case. For example, the formula below

```
Name matchexact "J*Johnson"
```

will produce a true result for **Jeff Johnson**, but a false result for **JEFF JOHNSON**. (However, **JEFF Johnson** would produce a true result.)

You can use the matchexact operator instead of beginswith, endswith, or contains if you need an exact upper and lower case match.

#### A notmatch B

#### A notmatchexact B

These operators are the exact opposite of match and matchexact.

### Combining Comparisons

The basic comparisons described in the previous section can be combined together for more complicated decisions. There are four basic operators that can combine or modify decisions: **and**, **or**, **xor**, and **not**.

Operator	Sample	Description
and	A and B	true if both A and B are true
or	A or B	true if either A or B are true
xor	A xor B	true if A and B are different
not	not A	true if A is false

#### A and B

The **and** operator combines two true/false formulas together so that the result is only true if both formulas are true. The example procedure below determines if a person is a teenager.

```
if Age≥13 and Age<20
  Status="Teenager"
endif
```

The result of the formula is only true if the person is 13 or older and less than 20.

**A or B**

The **or** operator combines two true/false formulas together so that the result is true if either one of the two formulas are true. The example below determines if a transaction is being paid with a credit card.

```
if PaymentMethod="Visa" or PaymentMethod="MasterCard"
  Terms="Credit Card"
endif
```

The result of the formula is only true if the payment method is **Visa** or **MasterCard**.

Notice that each side of the **or** operator must contain a complete formula. The formula below looks right in English, but will not work in Panorama. The example below is WRONG:

```
if PaymentMethod="Visa" or "MasterCard"          /* WILL NOT WORK !! */
```

There must be a comparison on both sides of the **or**, as shown in the first example.

**A xor B**

The **xor** (short for exclusive-or) operator is a bit tricky. **Xor** combines two true/false formulas together so that the result is true if one of the two formulas is true, but false if both are true or both are false. Another way to put it is that the result will be true if A and B are different, but false if they are the same. The example below determines if two shoes are a pair.

```
if Shoe1="Left" xor Shoe2="Left"
  message "These shoes are a pair"
endif
```

The result of the formula is only true if one shoe is **Left** and the other shoe is **Right** (or to be more precise, not **Left**).

**not A**

The not operator reverses a true-false formula. If the result was true, now it will be false. If it was false, now it will be true.

```
if (not (Shoe1="Left" xor Shoe2="Left"))
  message "These shoes are not a pair!"
endif
```

Note: This example shows that if **not** is used as the very first operator in a formula in a procedure, you must surround the entire formula with an extra pair of parentheses. If **not** is in the middle of the formula the extra parentheses are not necessary. The parentheses are also not necessary if the formula is not in a procedure (in the Design Sheet or a **Formula Fill** dialog, for example).

**Equals Comparison vs. Assignment**

If you have skipped ahead to read about procedures you know that the equals sign is used to assign a value to a field or variable. The example formula we used earlier to compare two values:

```
PaymentMethod="C.O.D."
```

would also be the same formula used to assign the value C.O.D. to the field or variable PaymentMethod. At first glance this may appear ambiguous...the same formula is used to compare two values and to assign a value. How do we know when we are assigning and when we are comparing? The answer lies in the context in which the formula is found.



In a procedure, an assignment is always by itself, not part of a larger statement. A true-false formula is always part of another statement, for example **if**, **case**, **until**, **while**, **stoploopif**, **repeatloopif**, **find**, **select**. Here's an example that shows two formulas that look almost the same, but one is a true-false formula and one is an assignment.

```
if PaymentMethod="C.O.D."
    ShippingMethod="UPS"
endif
```

The first formula, **PaymentMethod="C.O.D."**, is part of the **if** statement. This formula means: Is the field (or variable) **PaymentMethod** equal to **C.O.D.** (true/false)?

The second formula, **ShippingMethod="UPS"**, is not part of any statement, but stands alone, so this is an assignment. The statement means: Take the value **UPS** and copy it into the field or variable named **Shipping-Method**.

If an assignment has more than one equals sign, the first equals sign is for the assignment and the rest are for comparisons. The example assignment below compares **B** and **C**. If they are equal (true) the value -1 will be copied into **A**. If they are not equal (false) the value 0 will be copied into **A**.

```
A=B=C
```

In other words, **A** becomes the result of the comparison between **B=C**, or **A = (B=C)**.

### True/False Values

For purposes of calculation, Panorama treats true and false as numbers: true is -1 and false is zero. Panorama also has two functions that directly generate these values.

Function	Reference Page	Description
true()		This function always returns true (-1).
false()		This function always returns false (0).

Like any other number, you can store a true/false value in a field or variable and then use it later. The example below calculates whether a person is a teenager, then uses that information later.

```
local Teenager
Teenager=Age≥13 and Age<20
...
if Teenager
    Price=4.50
else
    Price=6.00
endif
```

Notice that the **if** statement doesn't need to compare, it simply uses the result of the comparison that was calculated earlier. In fact, the **if** statement (and all other statements that use true/false logic) can use any formula that produces a numeric integer result. The value 0 will be regarded as false, and any non-zero value will be regarded as true. The example below will be true if the length of the name is non-zero.

```
if length(Name)
    yesno "Is this a home address?"
    ...
endif
```

The first line of this example could also have been written **if length(Name)≠0**. The result is the same either way.

## The ? Function

The ?( function allows a formula to make a decision. Will it be door number 1 or door number 2? The function uses a true-false value to pick from one of two values. The syntax for this function is like this.

```
?(decision-value,true-value,false-value)
```

The first parameter, **decision-value**, is used to pick which of the two choices will be returned as the final value, the **true-value** or the **false-value**.

For example, the formula below can be used to calculate a 10% discount if the quantity is 100 or more—

```
?( Qty<100 , Price , Price*0.9 )
```

The decision is based on the comparison **Qty<100**. If Qty is less than 100, the ? function picks the second parameter, **Price**. But if the quantity is 100 or more, the ? function will pick the third parameter, **Price\*0.9**, for a 10% discount.

If you need to pick from three or more choices you can nest several ? functions together. For example, this formula shows how you can add a third discount level (20% for quantities of 500 or more)—

```
?( Qty<100 , Price , ?(Qty<500 , Price*0.9 , Price*0.8 ) )
```

Although these examples have used numeric data, text can also be used for either the true-false logic or the choices. The formula below, for example, could be used by a movie theater to check if a person is a child or an adult.

```
?( Age≤12 , "Child" , "Adult" )
```

Note: The ? function always evaluates all three parameters you give it, even though it really uses only two of the parameters. This means that you cannot use the ? function to avoid errors (for example divide by zero errors) because the error will happen before the ? function decides which parameter to use (use the `divzero()` function to avoid divide by zero problems).

## Converting a Boolean Value to Text

The `boolstr()` function converts a boolean value to text, either **true** or **false**. For example

```
message boolstr(Qty<100)
```

will display **true** if the Qty is less than 100, or **false** if it is greater or equal to 100.

## Linking With Another Database

Many database applications require multiple database files working together. For example, organizing a company's order entry operations usually requires an invoice file, an inventory/price list file, and possibly a customer file. The primary method for accessing information in other databases is the **lookup()** function (and other related functions). This function can search for and retrieve information from any open database. Need to look up a price or a customer's credit limit? Chances are the lookup() function is the tool for the job.

When you look up information manually (for example, looking up someone's number in the phone book), you are actually performing a multi-step process. You start with one piece of information—a person's name, for example. The first step is to locate the correct phone book. Once you've located the correct book, you must search through it to find the name of the person you are looking for. When you find the name, the final step is to copy down the person's phone number.

Panorama's lookup() function follows a similar process when it looks up data. For example, suppose you want to find out the number of calories in an orange using the database shown here.

Groceries											
Fruit	Serving Size	Calories	Fat (	Total Fat	Saturated	Cholesterol	Sodium	Potassium	Carbc	Dietary Fiber	
Grapes	1-1/2 cups grapes	90	10	1g	0g	0mg	0mg	270mg	24g	1g	
Lemon	1 med lemon (58g)	15	0	0g	0g	0mg	0mg	0mg	5g	1g	
Lime	1 medium (67g)	20	0	0g	0g	0mg	0mg	75mg	7g	2g	
Cantaloupe	1/4 melon (134g)	50	0	0g	0g	0mg	25mg	280mg	12g	1g	
Honeydew	1/10 melon	50	0	0g	0g	0mg	35mg	310mg	13g	1g	
Orange	1 med orange (154g)	70	0	0g	0g	0mg	0mg	260mg	21g	7g	
Tomato	1 med tomato (148g)	35	0	1g	0g	0mg	5mg	360mg	7g	1g	
Pear	1 medium (166g)	100	10	1g	0g	0mg	0mg	210mg	25g	4g	
Kiwifruit	2 med kiwi (148g)	100	10	1g	0g	0mg	0mg	0mg	24g	4g	
Grapefruit	1/2 grapefruit	60	0	0g	0g	0mg	0mg	230mg	16g	6g	

Here is the formula for looking up the number of calories in an orange. The parameters to the lookup contain all the information necessary to locate the information.

*lookup database* → `lookup("Groceries",Fruit,"Orange",Calories,0,0)`

*lookup data field* → `Calories`

*lookup key field* → `Fruit`

*lookup key value* → `"Orange"`

*lookup default value* → `0,0`

*lookup summary level (almost always zero)* → `0,0`

The first parameter is called the **lookup database**. It tells Panorama what database to look in for the information, in this case **Groceries**.

The second and third database tell Panorama how to search for the data you want. In this case Panorama is being told to "search through the Fruit column until you find **Orange**." The field to look in (in this case **Fruit**) is called the **lookup key field**. The data to look for (in this case **Orange**) is called the **lookup data value**. By the way, Panorama is very picky about the lookup data value. It must exactly match the value in the database, or Panorama won't find a match. In this case only Orange will work — not **orange** or **ORANGE** or even **oRaNGe**!

At this point we come to a fork in the road. Perhaps Panorama found **Orange** in the database, perhaps not. If it did the fourth parameter tells Panorama what to do next. This fourth parameter is called the **lookup data field**, and it may be any field in the lookup database. In this case it is **Calories**, so Panorama will lookup the value in the **Calories** field (**70**) and return it as the result of the function.

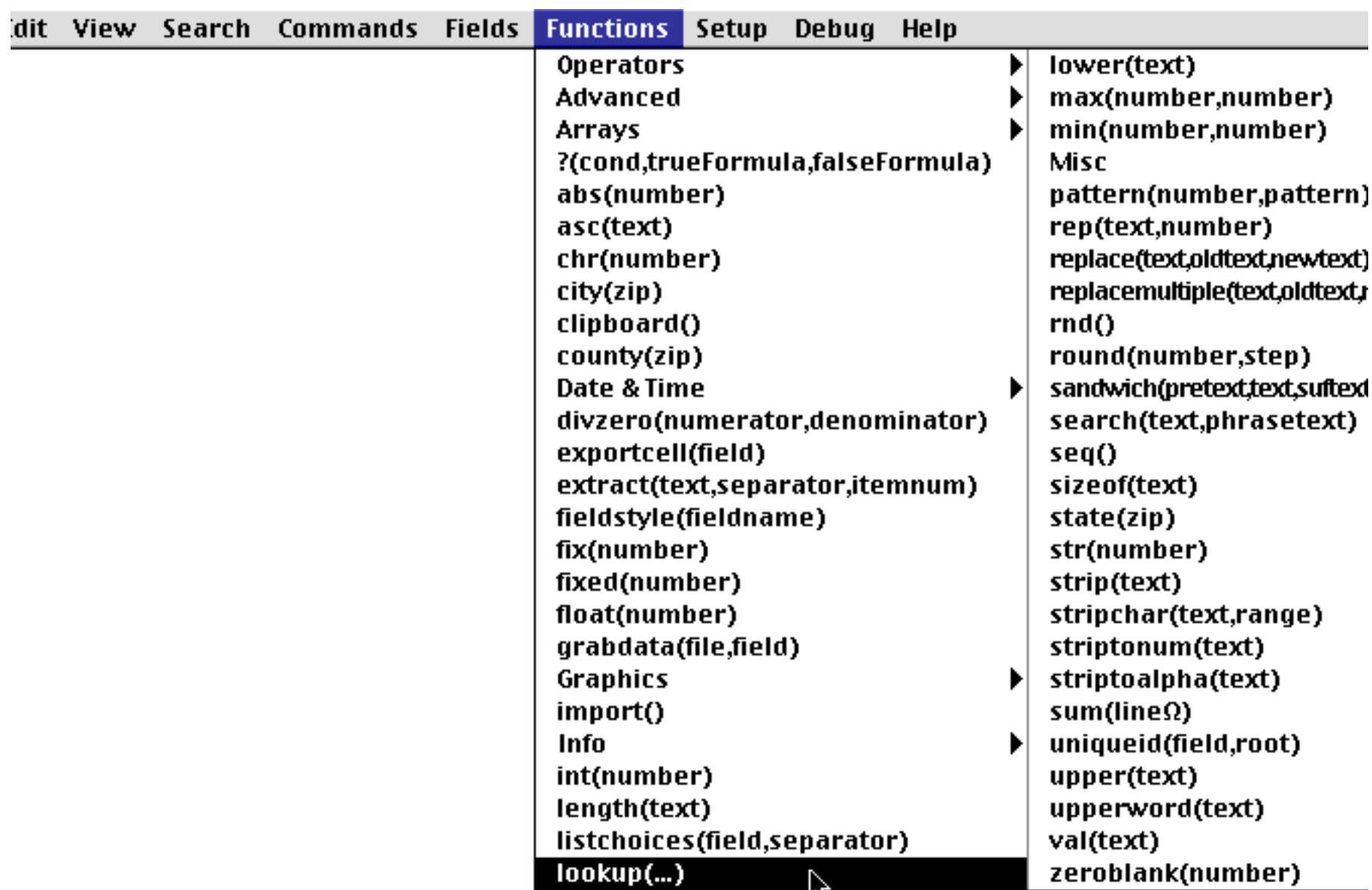
What if Panorama didn't find **Orange** in the database? In that case Panorama simply returns the value of the fifth parameter, the **lookup default value**. In this case the default value is **0**. The default value should match the data type of the lookup data field. Since **Calories** is a numeric field, the default is also numeric. If the lookup data field had been a text field (for instance **Serving Size**) the default would need to be text (for example "").

The sixth and final parameter to the lookup function is the **lookup summary level**. This is the minimum summary level to be searched within the lookup database. Usually the lookup summary level is zero so that the entire lookup database will be searched. If the level is set to 1 through 7, only summary records will be searched. This is useful if you want to look up summary information (see "[3-Step Summarizing](#)" on page 371) while ignoring the raw data.

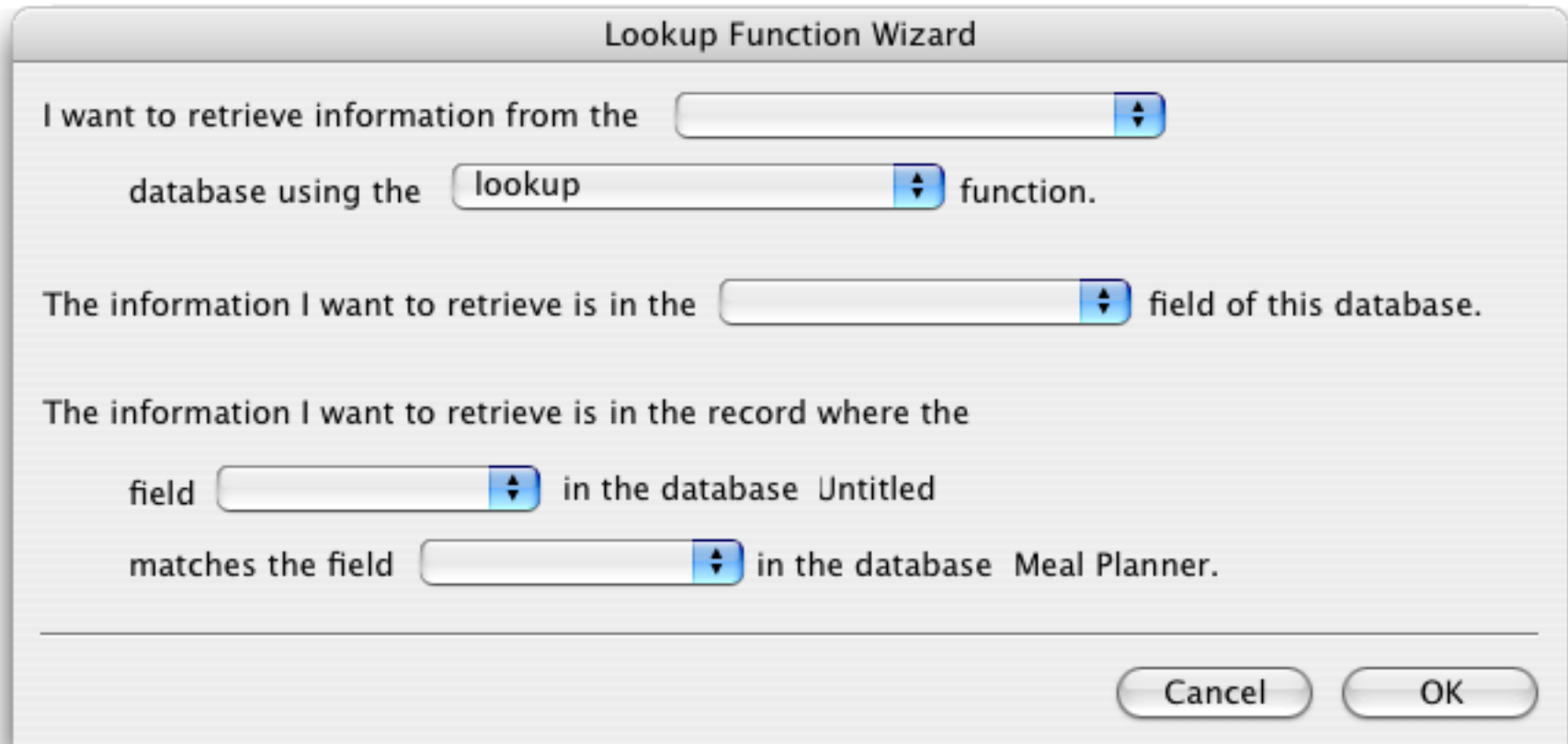
In this example the end result of the lookup is the value **70**. The lookup() function is often used by itself, but a more complicated formula can take this value and perform additional computations. If the result of the lookup is a text value then all of the text functions described earlier in this chapter can be used to modify the result.

### The Lookup Wizard

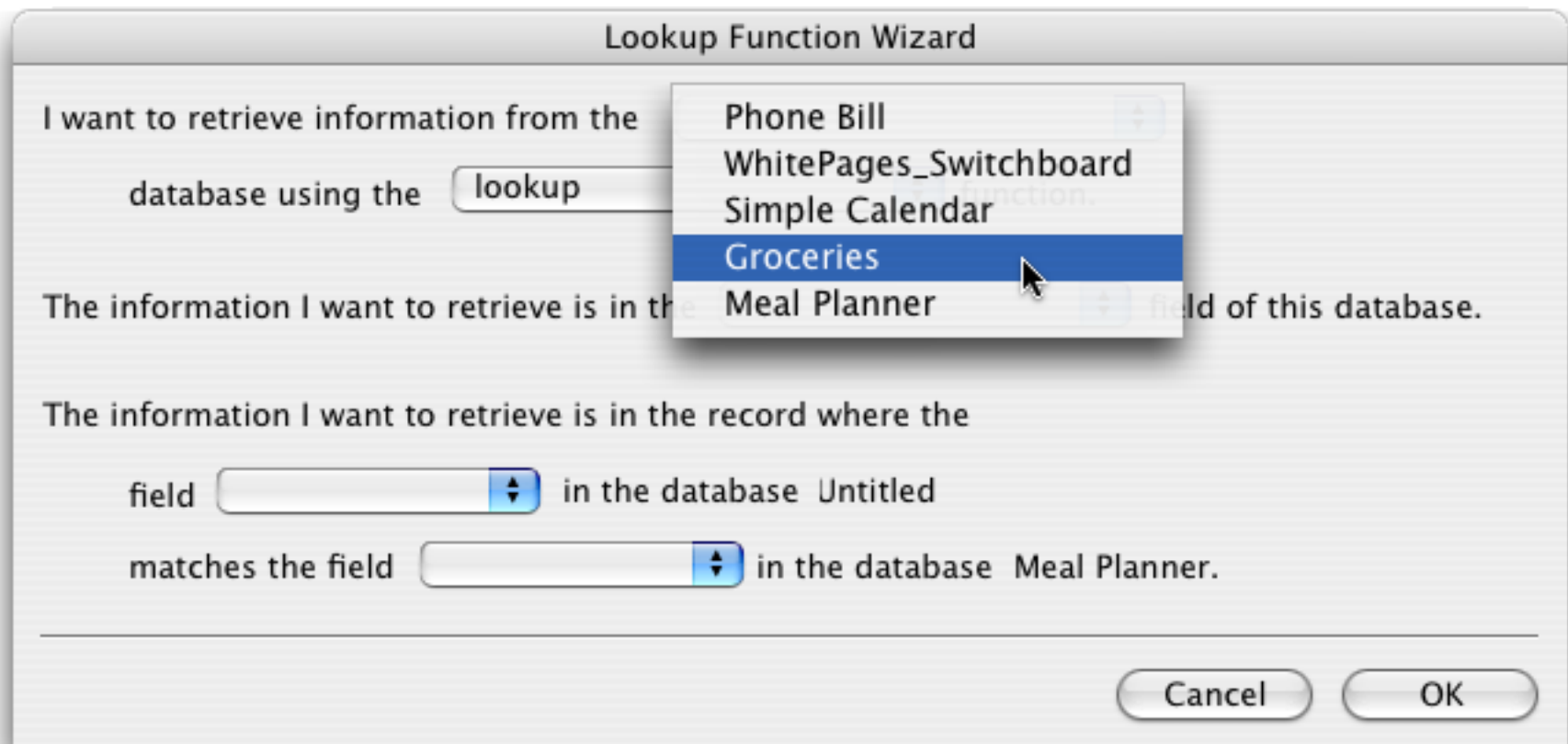
Since the lookup() function is kind of picky about all of its parameters we've provided a "fill-in-the-blanks" dialog to help build the function. To open this dialog simply pull down the **Functions** Menu and choose **lookup(...)**.



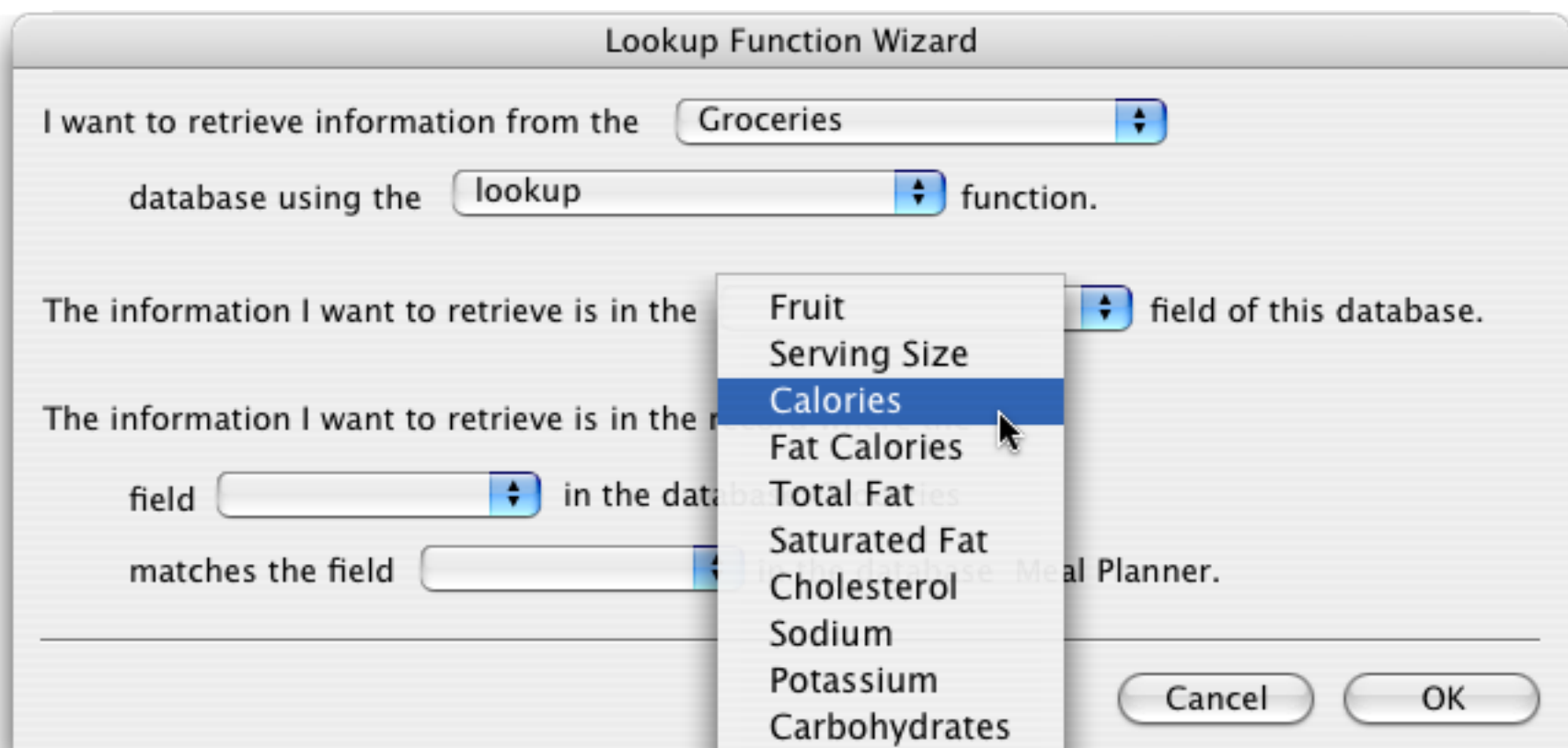
Now the lookup wizard dialog appears.



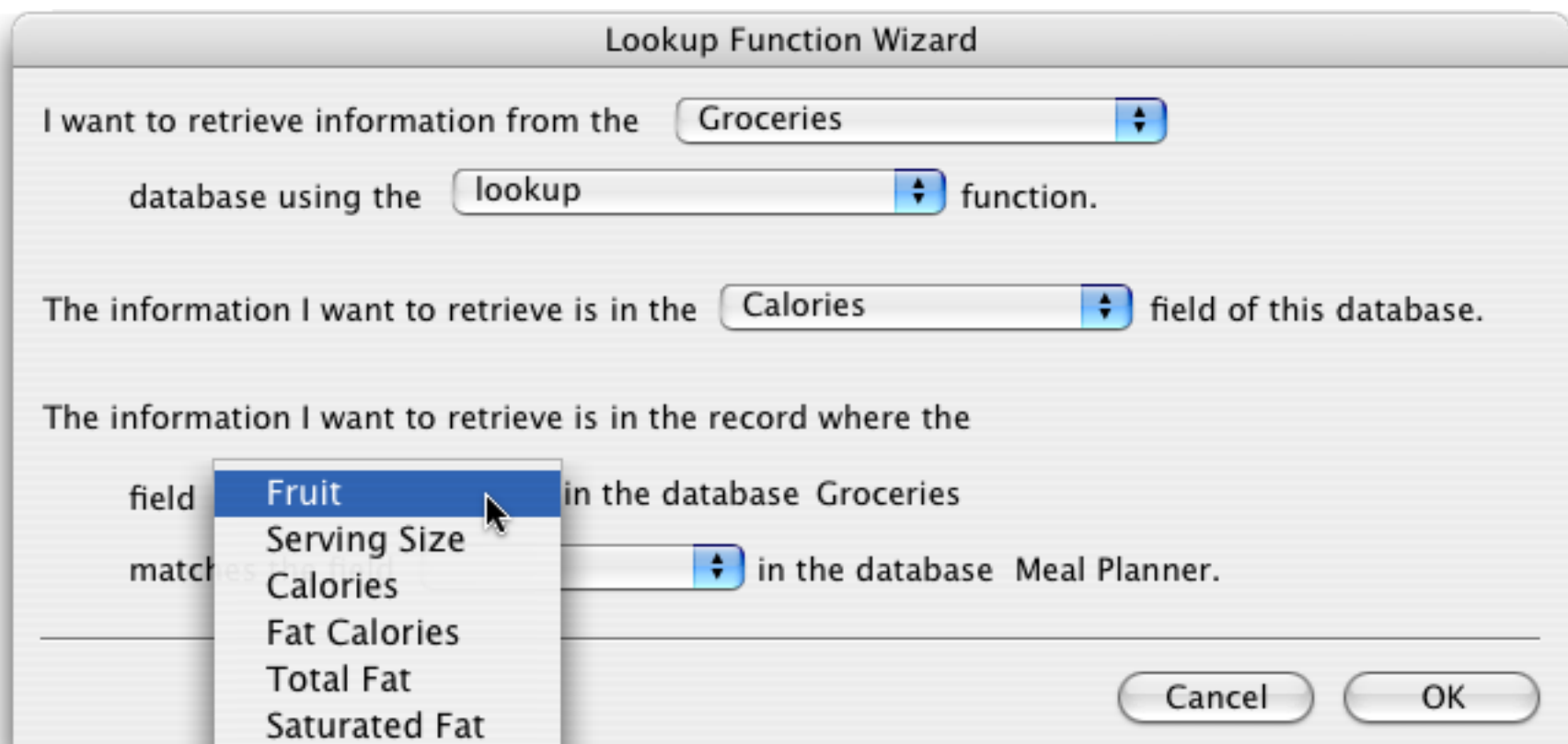
To create the lookup function, start at the top of the dialog and work your way down. Start by selecting the database to lookup from (in this case [Groceries](#)).



Next, choose the data you want to retrieve (the lookup data field, which will become the fourth parameter to the lookup function). In this case we want to retrieve the number of [Calories](#).

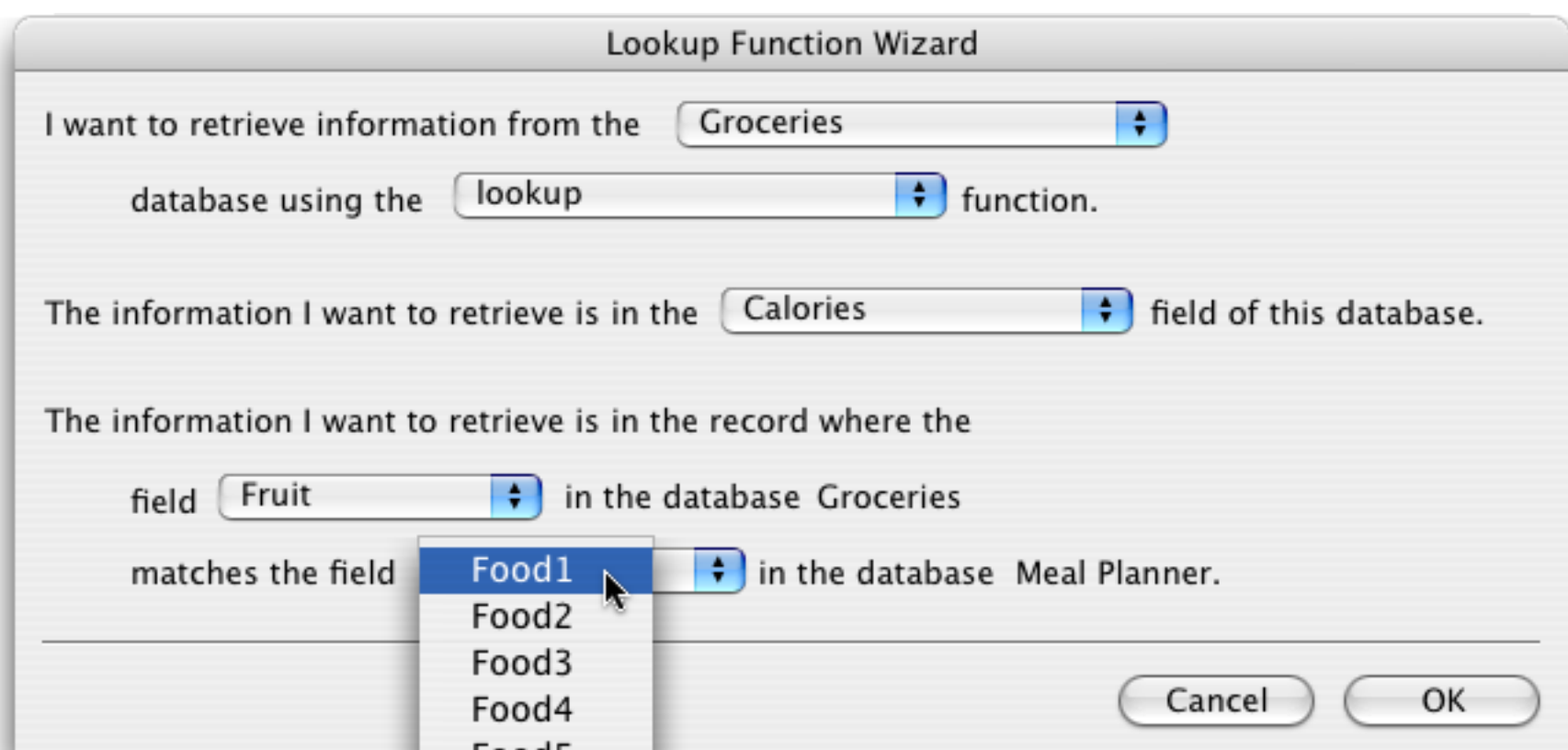


The next step is to choose the lookup key field, which in this case is [Fruit](#).

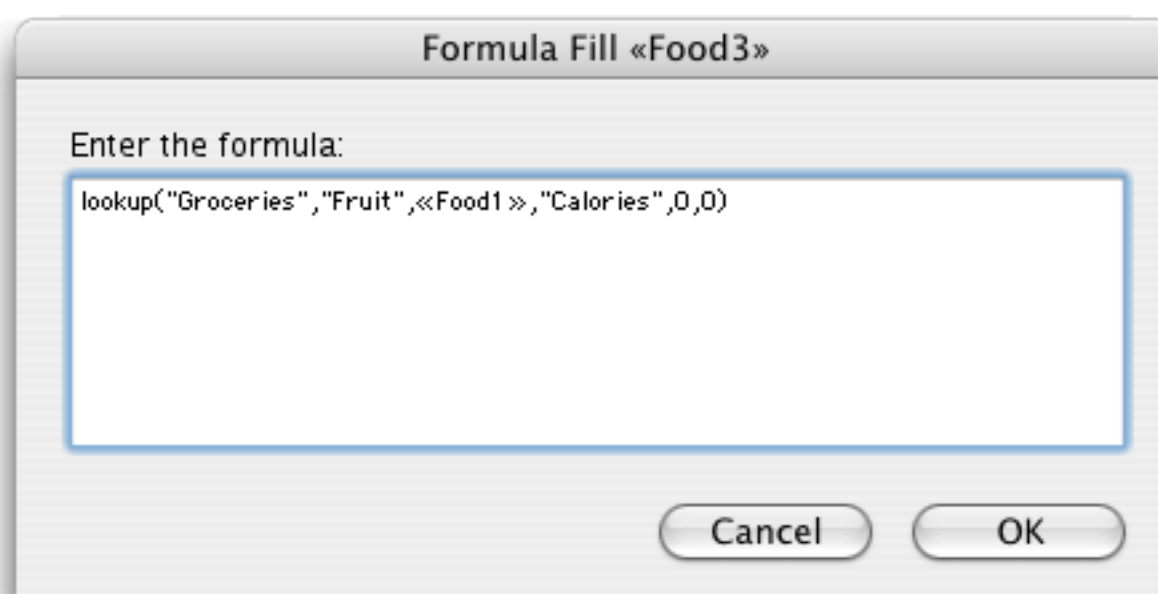




Finally, choose the field containing the lookup key value. If there is no such field (perhaps the value is in a variable) then just choose any field and adjust the formula once the wizard is finished.



Once you've made all of the selections press **OK** to generate the finished formula. For example, if you were using the Formula Fill command the formula would look like this:



### Type Mismatch Problems

One of the most common problems when setting up a lookup function is type mismatches. With some careful thought, however, you can avoid these problems.

The first source of type mismatch problems is the lookup key field and the lookup key value. The field and value must be the same type of data. In other words, if the lookup key field is numeric, the lookup key value must be numeric also. If necessary, you can convert a text key value into numeric with the `val()` function, or you can convert a numeric key value into text with the `str()` function (see [“Converting Between Numbers and Strings”](#) on page 538 for details on both of these functions).

```
lookup("Catalog","Part#",val(Item),"Price",0,0)
```

Another source of type mismatch problems is the lookup data field. This field must have the same type of data as the field you want to store the result in. For example if you look up a price, the result must be stored in a numeric field.

If you need to store a numeric value in a text field, use the `str()` function to convert the value. The `str()` function should go outside the entire lookup function, for example

```
str(lookup("Catalog","Item",Desc,"Price",0,0))
```

Another source for type mismatch problems is the lookup default value. The default value should be the same type as the lookup data field. If the lookup data field is numeric, the default should be numeric (for example 0 or 100). If the lookup data field is text, the default should also be text (for example "" or "n/a").

## Lookup Variations

There are actually several different variations of the lookup function. All of the variations have the same six parameters. The standard lookup function locates the first occurrence of the key (nearest to the beginning of the file).

Function	Description
<code>lookup()</code>	This function searches for the first occurrence of the value within the lookup database. If there is more than one copy of the value in the database this function will find the one closest to the top.
<code>lookuplast()</code>	This function searches for the last occurrence of the value within the lookup database. If there is more than one copy of the value in the database this function will find the one closest to the bottom. However, there is one exception. If you are looking up within the current database Panorama will skip the current record. If the current record matches the key value then Panorama will skip backwards to the next matching record.
<code>lookupselected()</code>	This function searches for the first occurrence of the value within the selected records in the lookup database. Unselected (invisible) records are ignored. If there is more than one copy of the value within the selected records this function will find the one closest to the top.
<code>lookuplastselected()</code>	This function searches for the last occurrence of the value within the selected records in the lookup database. Unselected (invisible) records are ignored. If there is more than one copy of the value within the selected records this function will find the one closest to the bottom.
<code>table()</code>	The table function allows you to lookup data by an approximate match instead of exact match. If the table function does not find an exact match, it uses the next lower value. A common example is a shipping rate table. Rate tables do not have an entry for every possible weight. Instead, the table only lists weights where the shipping rate changes. For example, suppose a rate table contains entries for 100 pounds and 250 pounds, and you have a 158 pound package. The table function will return the rate for the next lower value, in this case the 100 pound rate.

## Looking Up Rates in a Rate Table

The `table()` function is designed for looking up rates from a table. For example, this function can be used to look up shipping rates, tax rates, discount rates, or any kind of stepped rate where the rate changes according to a sliding scale. To illustrate this function, consider this shipping rate database.

Weight	Rate Per Pound
0	2.50
50	2.35
100	2.25
250	2.12
500	2.03
1000	1.94

For packages from 0 to 49.99 pounds the rate is 2.50 per pound. For packages from 50 to 99.99 pounds the rate is 2.35 per pound, from 100 to 249.99 the rate is 2.25 etc. Suppose we use a regular lookup function to look up the weight, like this.

```
lookup("Shipping Rates",Weight,PackageWeight,«Rate Per Pound»,0,0)
```

This formula will work fine for weights that appear in the table like 50, 100 and 250. But for other weights like 47 or 182 the formula will return the default value, zero. To fix this, use the `table()` function instead of the `lookup()` function.

```
table("Shipping Rates",Weight,PackageWeight,«Rate Per Pound»,0,0)
```

The `table()` function will return the closest lower match. This means that if the `PackageWeight` is 3, 17 or 42 the formula will return 2.50. If the `PackageWeight` is 110 or 246 the formula will return 2.25, etc. Here is a complete formula that calculates the shipping cost for any package.

```
PackageWeight*table("Shipping Rates",Weight,PackageWeight,«Rate Per Pound»,0,0)
```

The formula looks up the rate per pound and then multiplies that rate by the package weight.

## Looking Up Multiple Fields From One Record

Sometimes you may need to lookup several fields in the same record. For example, when you lookup someone's address you may also want to lookup their city, state, zip code, phone number and recent purchasing history. In a procedure one way to do this is with multiple `lookup()` functions, like this.

```
Address=lookup("Customers",Company,Company,Address,"",0)
City=lookup("Customers",Company,Company,City,"",0)
State=lookup("Customers",Company,Company,State,"",0)
Zip=lookup("Customers",Company,Company,ZipCode,"",0)
Phone=lookup("Customers",Company,Company,"Phone#","",0)
```

When a procedure contains several `lookup()` in a row for the same thing like this Panorama doesn't actually search the database over and over again. Instead it notices that it is searching for the same item and simply grabs the data from the record it has already found.

## The GrabData Function

The `grabdata()` function grabs the contents of a field in the current record of any open database. You can grab data from the current database, or from another database. The function has two parameters — the name of the database to grab from and the name of the field within that database. For example here is the formula to look up the number of calories of the currently selected fruit.

```
grabdata("Groceries",Calories)
```

The value returned by this function will change depending on what record is active in the `Groceries` database.

## Looking Up Data in the Current File

You can use the `lookuplast()` function to look up the previous entry, with the same value, in the same database. For example, in a checkbook database you can automate repetitive payments by looking up the previous payment to the same company. By using the `info("database")` function to look up the database name you can make sure that the formula will continue to work even if the database is renamed.

```
lookuplast(info("database"),PayTo,PayTo,Amount,0,0)
```

Suppose that your last check to **Pacific Mutual** was **\$178.34**. Using the formula above you could automatically enter this value the next time you write a check to this company.

Another application for looking up data in the current file is locating summary information further down in the database. To do this, set the lookup summary level to a non-zero value so that only summary records will be located.

## Zip Code Lookup

If you have purchased Panorama's optional zip code dictionary you can lookup the city, county and state of a zip code using the functions listed in the table below.

Function	Description
<code>city(zip)</code>	This function looks up a zip code and returns the name of the city for that zip code. The zip code may be either a number or text. For example the formula <code>city(92831)</code> will return the city name <b>Fullerton</b> , while the formula <code>city("92648")</code> will return <b>Huntington Beach</b> . If there is more than one possible name, the function returns the primary zip code name as defined by the US Post Office. If the zip code is not a valid zip code the function will return an empty string (""). If the zip code dictionary has not been installed the function will return --.
<code>county(zip)</code>	This function looks up a zip code and returns the name of the county for that zip code. The zip code may be either a number or text. For example the formula <code>county(92831)</code> will return the county name <b>Orange</b> , while the formula <code>county("95234")</code> will return <b>San Joaquin</b> . If the zip code is not a valid zip code the function will return an empty string (""). If the zip code dictionary has not been installed the function will return --.
<code>state(zip)</code>	This function looks up a zip code and returns the two letter abbreviation for the name of the state the zip code is in. The zip code may be either a number or text. For example the formula <code>state(92831)</code> will return the state abbreviation <b>CA</b> , while the formula <code>state("15234")</code> will return <b>PA</b> . If the zip code dictionary has not been installed the function will return --.

For more information about purchasing this optional package please visit our web site, [www.provue.com](http://www.provue.com).

## US Post Office Abbreviation Functions

These functions return text arrays that contain lists of official US Post Office abbreviations. These functions are designed to be used with the [arraylookup\(\)](#) and [arrayreverselookup\(\)](#) functions.

Function	Description
stateabbreviations()	This function returns a list of state abbreviations in this format: <a href="#">AL:ALABAMA</a> ; <a href="#">AK:ALASKA</a> ; ... This table is designed to be used with the <a href="#">arraylookup()</a> and <a href="#">arraylookupreverse()</a> functions.
statelookup(state)	This function looks up the name of a state from the state abbreviation (for example <a href="#">CALIFORNIA</a> from <a href="#">CA</a> ). If the parameter does not match any state then the original value is returned.
uspssecondaryunits()	This function returns a list of USPS secondary suffix designation abbreviations in this format: <a href="#">APT:APARTMENT</a> ; <a href="#">RM:ROOM</a> ; ... This table is designed to be used with the <a href="#">arraylookup()</a> and <a href="#">arraylookupreverse()</a> functions.
uspsstreetsuffixes()	This function returns a list of USPS street suffix abbreviations in this format: <a href="#">ALY:ALLEY</a> ; <a href="#">AVE:AVENUE</a> ; ... This table is designed to be used with the <a href="#">arraylookup()</a> and <a href="#">arraylookupreverse()</a> functions.

## Import/Export Functions

These functions may be used to customize the import and export of data.

Function	Description
import()	<p>This function returns a line of imported data. This function only works in conjunction with the <a href="#">importusing</a> (see “<a href="#">IMPORTUSING</a>” on page 5355) and <a href="#">arrayfilter</a> statements (see “<a href="#">ARRAYFILTER</a>” on page 5045).</p> <p>The <a href="#">import()</a> function returns a line of text. When it is used with the <a href="#">importusing</a> statement, the <a href="#">import()</a> function returns the contents of the line that is currently being imported. Using this function you can process and re-arrange the data as it is being imported.</p> <p>When it is used with the <a href="#">arrayfilter</a> statement, the <a href="#">import()</a> function returns the individual array element currently being processed. Using this function you can process the data in each array element.</p> <p>When it is used at any other time, the <a href="#">import()</a> function returns empty text.</p>
importcell(colNumber)	<p>This function returns one cell of imported data. This function only works in conjunction with the <a href="#">importusing</a> statement (see “<a href="#">IMPORTUSING</a>” on page 5355). <a href="#">ColNumber</a> is the column of data from the imported text that you want to return. The text being imported is separated into columns by either tabs or commas. The first column is column 0, the next is column 1, etc.</p> <p>The <a href="#">importcell()</a> function always returns text. When it is used with the <a href="#">importusing</a> statement, the <a href="#">importcell()</a> function returns the contents of the specified column from the line that is currently being imported. If the text being imported is comma delimited, the <a href="#">importcell()</a> function will strip off any quotes around the data before returning it. Using this function you can process and re-arrange the data as it is being imported.</p> <p>When it is used at any other time, the <a href="#">importcell()</a> function returns empty text. It will also return empty text if you specify a column number that does not exist in the text being imported.</p>



Function	Description
exportline()	<p>This function generates a tab delimited line of data containing all the fields in the current record. This function is designed to be used with the <code>export</code> (see “<a href="#">EXPORT</a>” on page 5207) and <code>arraybuild</code> (see “<a href="#">ARRAY-BUILD</a>” on page 5038) statements, but may actually be used anywhere.</p> <p>The function returns a a tab delimited line of data containing all the fields in the current record. Any non-text fields (numeric, date) will be converted to text as they are placed into the tab delimited line. The tab delimited line does NOT include a carriage return on the end.</p>
exportcell(field)	<p>This function takes any database field and converts it to text, using the appropriate pattern if one has been defined in the design sheet. <code>Field</code> is the name of the field to be converted to text.</p> <p>The function always returns a text type data item. The power of the <code>exportcell()</code> function is that it does not require you to know what type of data you are exporting. It simply takes whatever kind of data is in the field (text, number, date, whatever) and converts it into text.</p>

## System and Database Information Functions

The functions described in this section allow a formula to access information about the computer system and about the current database.

### System Information

These functions return information about the computer system — the keyboard, mouse, memory, clipboard and Panorama itself.

Function	Description
currentprinter()	This function returns the name of the current printer (OS X only).
info("computername")	This function returns the short name of the computer (OS X only). This is the computer name that needs to be used in terminal sessions.
info("freememory")	This function calculates how much free database memory is available.
info("modifiers")	<p>This function returns the status of the modifier keys. The five different possible modifier keys are:</p> <pre>"shift" "capslock" "option" (returned when <b>ALT</b> key pressed on PC) "command" (returned when <b>Control</b> key pressed on PC) "control" (returned when <b>Right Mouse button</b> clicked on PC)</pre> <p>The <code>info("modifiers")</code> function also returns the status of the last mouse click. If the last mouse click was a double click, <code>info("modifiers")</code> will return "<code>doubleclick</code>". If the last mouse click was a triple click, <code>info("modifiers")</code> will return "<code>tripleclick</code>".</p> <p>If more than one modifier key is active the function will return all of them strung together like this:</p> <pre>"shift option"</pre> <p>You should check for a specific modifier with the <code>contains</code> operator. This example opens the <code>Status</code> form if the user double clicks on a button.</p> <pre>if info("modifiers") contains "double"   openform "Status" endif</pre>



Function	Description
info("parameters")	This function returns the number of parameters passed to a procedure.
info("trigger")	<p>This function returns information about how the current procedure was triggered. If the procedure was triggered by data entry this function will return the word <b>Key</b> followed by a period and then the key that actually triggered the procedure:</p> <pre>Key.return Key.enter Key.tab</pre> <p>If the procedure was triggered by a button, the function will return the word <b>Button</b> followed by a period and then the name of the button, for example:</p> <pre>Button.Save Button.Calculate Tax Button.Show Chart</pre> <p>If the procedure was triggered by a custom menu, the function will return the word <b>Menu</b> followed by a period, the name of the menu, another period, and then the menu item. Here are some examples:</p> <pre>Menu.Accounting.Aging Menu.Letter.New</pre>
info("version")	<p>This function returns the version of the currently running copy of Panorama. The version number is returned as text, for example <b>3.5.1</b>. If you want to extract only a portion of the version number you can use the array or array functions. This example will extract the first two numbers of the version. The result will be something like <b>3.5</b> or <b>4.0</b>.</p> <pre>arrayrange(info("version"),1,2,".")</pre>
info("unixusername")	This function returns the short user name the user has logged in under (Mac OS X only). This is the user name that needs to be used in terminal sessions.
listprinters()	This function returns a carriage return delimited list of the printers available on the system (Mac OS X only).
parameter(number)	<p>This function is used to transfer data between a main procedure and a subroutine. The main procedure can set up one or more data item parameters as part of the call statement (see "<a href="#">CALL</a>" on page 5086). The subroutine can retrieve and use these data items using the parameter( function. <b>Number</b> is a number specifying what parameter you want to retrieve. All parameters are numbered, starting with 1 (1, 2,3, 4, etc.). The function returns a data item. This data item may be text or numeric, depending on what kind of data is passed to the subroutin</p> <p>New in Panorama 5.5: The parameter( function now works with negative parameter numbers. When a negative parameter # is passed it returns information about the parameter data type. This can be used in conjunction with the setparameter statement to modify the data type of the data being passed back by setparameter. Possible data types are: 0 = parameter cannot be modified (might not exist), so has no data type, 1 = text, 2 = compressed text, 3 = compressed text, 4 = picture, 5 = date, 6 = floating point, 7 = integer,8 = fixed 1 digit, 9 = fixed 2 digits, 10 = fixed 3 digits, 11 = fixed 4 digits, -1 = variable (has no data type).</p>

## Database Information

These functions return information about the currently active database.

Function	Description
countsummaries(level)	This function counts the number of summary records in the current database. The level parameter should be from 0 to 7. If 0, all summary records will be counted. If 1 to 7 then only that specific level will be counted.
dbname()	This function returns the name of the current database.
dbpath()	This function returns the path of the folder the current database is located in.
dbsubfolder(subpath)	This function returns the folder ID of a subfolder within the same folder as the current database. For example if the current database is in the folder "MyDrive:My Stuff", the function <code>dbsubfolder("Images")</code> returns the folder ID of the "MyDrive:My Stuff:Images" folder.
emptydatabase()	This function returns true if the current database is empty, false if it is not.
emptyline()	This function returns true if the current line (all fields) is empty, false if it is not.
getautonumber()	This function returns the automatically generated number for the next record that will be added to the database.
listchoices(field,separator)	<p>This function builds a text array containing a list of all the values stored in a specified field. (Note: this function is not related to the choices data type.) There are two parameters: <b>field</b> and <b>separator</b>. <b>Field</b> is the name of the field that contains the values you want to build a list of. <b>Separator</b> is the separator character for the text array you are building (see "<a href="#">Text Arrays</a>" on page 539).</p> <p>The <code>listchoices()</code> function scans the specified field and builds a list of all the values stored in that field. The list is returned in the format of a text array. Here is a formula that builds a list of the states in the current database.</p> <pre>listchoices(State,¶)</pre>
seq()	<p>This function returns a sequential numbers (1, 2, 3, etc.). This function only works in conjunction with the <code>formulafill</code>, <code>select</code>, <code>find</code> and <code>arrayfilter</code> statements.</p> <p>When it is used with the <code>formulafill</code>, <code>find</code> or <code>select</code> statements the <code>seq()</code> function return a sequential number for each record (the first selected record is 1, the second is 2, etc.).</p> <p>When it is used with the <code>arrayfilter</code> statement, the <code>seq()</code> function returns a sequential number for each element in the array being processed (the first array element is 1, the second is 2, the third is 3, etc.).</p> <p>When it is used at any other time, the <code>seq()</code> function returns the number 1.</p> <p>This procedure uses the <code>seq()</code> function to select the first 10 records in the database:</p> <pre>select seq()≤10</pre>
info("bof")	This function returns true if the database is currently on the first visible record. (Note: "bof" stands for "beginning of file".)
info("changes")	This function returns the number of changes that have been made to the current database since the last time it was saved.
info("databasename")	This function returns the name of the current database. If the database name has a <code>.pan</code> suffix, that suffix is not included.

Function	Description
info("empty")	<p>This function returns true or false depending on the result of the last select operation. If no records were selected the function will return true, otherwise it will return false. The procedure below selects all records that are "Ready", whatever that means. If there are any ready records, the procedure prints them.</p> <pre> select Status="Ready" if info("empty")     message "Nothing ready today!"     stop endif print dialog field Status formulafill "Printed" </pre>
info("eof")	This function returns true if the database is currently on the last visible record. (Note: "eof" stands for "end of file".)
info("fieldname")	This function returns the name of the current field. See also the <a href="#">info("modifiedfield")</a> function below.
info("found")	This function returns true or false depending on the result of the last <b>find</b> or <b>next</b> statement (see " <a href="#">FIND</a> " on page 5245). If the last find or next found something, this function will return true. Otherwise it will return false.
info("records")	<p>This function returns the total number of records in the current database. To find out the number of selected records, use <a href="#">info("selected")</a> (see below).</p> <p>This example checks to see if all records are selected. If some records are not selected, the procedure does a selectall statement.</p> <pre> if info("selected") &lt;info("records")     selectall endif </pre>
info("selected")	This function returns the number of selected records in the current database. To find out the total number of records, use <a href="#">info("records")</a> (see above).
info("stopped")	This function returns true or false depending on the result of the last <b>uprecord</b> , <b>downrecord</b> , <b>left</b> or <b>right</b> statement. If the statement could not move the active cell because the active cell was already as far as it could go, the function will return true. Otherwise it will return false.
info("summary")	This function returns the summary level of the current record, from 0 (data record) to 7 (see " <a href="#">3-Step Summarizing</a> " on page 371).
info("visible")	This function returns true/false result based on whether or not the current record is visible. Useful for situations where Panorama may scan invisible records, for example the <a href="#">arraybuild</a> and <a href="#">select</a> statements, also the Scrolling List SuperObject.

## Custom Functions

Panorama doesn't limit you to the built-in functions that are supplied with a Panorama. In fact, you can actually create your own user defined functions that can be used in any formula. To learn how to build a custom function see Chapter 23 of the [Panorama Handbook](#).

# Chapter 24: Procedures



Right out of the box, Panorama is a very flexible program. Its built in menus and tools bring incredible power to your fingertips. In spite of this power, however, Panorama is a general purpose tool. To get the most out of Panorama you'll want to customize it to meet the specific needs of your business or industry. Doing this takes an up front investment of time and/or money. But if done properly, the payoff can be huge—a tool optimized specifically for running and organizing your business or life, not someone else's idea of how things should be done. You'll save time, reduce errors, and look more professional to your customers, vendors, employees and/or supervisors.

## **Programming Isn't Magic!**

If you've never programmed before, the idea of programming may seem like magic. But really there's nothing magic about it at all. Programming doesn't really add any new features or capabilities to Panorama. Anything that can be done with programming can also be done manually with Panorama's standard menus and tools.

If programming doesn't add any new features, what is it for? Many database tasks take more than one step to complete. To set up a report, for example, you may need to select certain information, sort the database, and perform calculations. A program allows you to define such a sequence of steps in advance. Once the sequence of steps is defined, you don't have to perform that sequence of steps manually any more. Simply ask the computer and it will perform the steps for you, flawlessly and at the highest possible speed. This lets the computer do what it does best, remember things accurately, and frees your mind for more important tasks. Eventually you can teach Panorama all the everyday tasks you need for running your organization. Of course every journey begins with a single step, so let's get started!

## **Introduction to (Panorama) Programming**

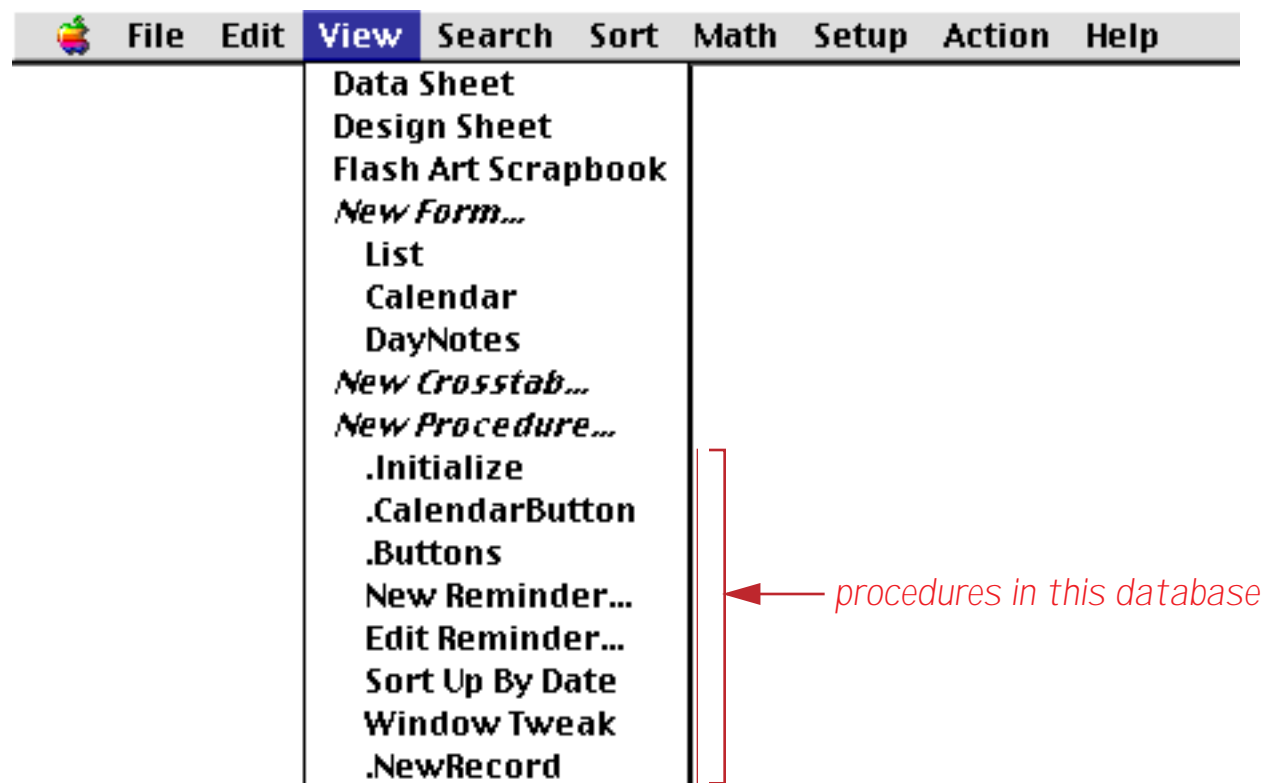
The next few pages introduce the fundamentals of programming with Panorama. You'll learn how complex programs are assembled from a series of small steps, and you'll learn the nuts and bolts of actually creating and using programs. If you are experienced with other programming languages like C or Basic, much of this material will be familiar to you already. If not, welcome to the exciting world of computer programming!



## Procedures

A complete program is assembled by combining a series of steps together so that they perform a complete task. In Panorama this complete series of steps is called a procedure. Each procedure performs a complete task from start to finish.

Each database can contain many procedures—one for each task that needs to get done in that database. To help keep all these procedures straight, Panorama requires you to give each procedure a unique name. The procedure name is used whenever you need to refer to the procedure—in a menu, a button, etc. All of the procedures in a database are listed in the **View** menu.



You can view a procedure by selecting it from this menu. You can also open the procedure in a new window by holding down either the **Control** key (Macintosh) or the **Alt** key (Windows) while you select the procedure from the **View** menu.

## Statements

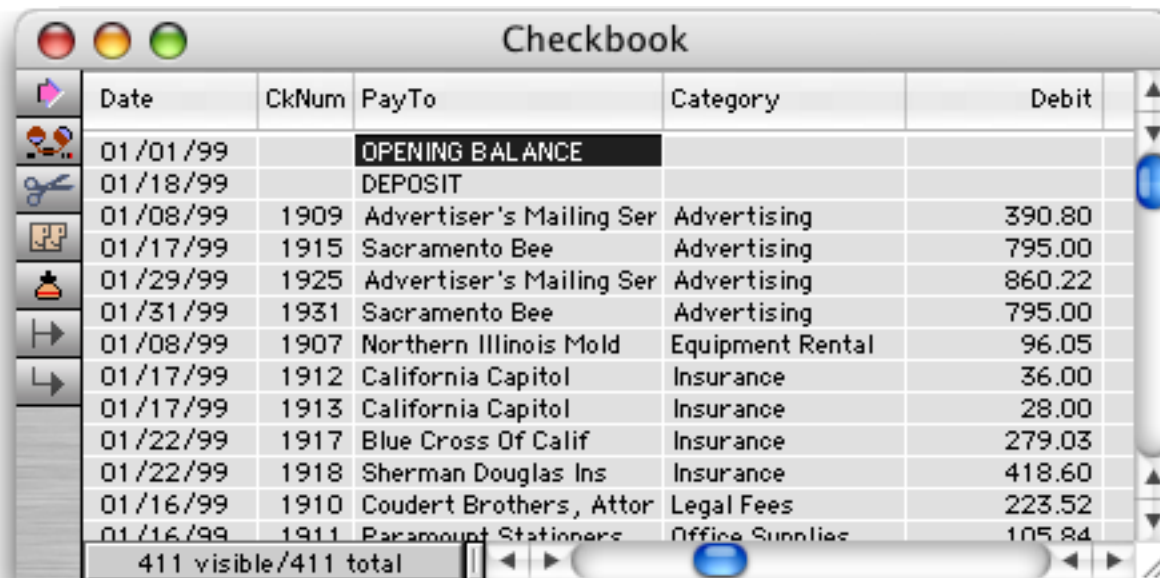
As mentioned in the previous section, a procedure is simply a series of steps. Programmers have a special name for these steps—they call them statements. Each statement is simply a single step to be performed by the computer. Most statements start with a special word (sometimes called a command or keyword) that tells Panorama what the statement should do, for example **SortUp**, **Select**, **Print**, **Open**. Panorama understands several hundred different keywords that perform a wide variety of operations.

A statement may consist of a keyword all by itself, for example **SortUp** or **CloseWindow**. However, many keywords also require additional options, for example **Open "MyDatabase"** or **Select Price>200**. These additional options are called parameters. If a keyword uses parameters, they must follow the keyword in the program.

(Note to experienced programmers: Unlike many programming languages, Panorama's keywords or commands are not reserved words. This means, for example, that you can use a database column named **Print** or create a variable named **Open**. These names are perfectly OK in Panorama's programming language. However, using keywords in this way could easily result in programs that are very confusing to read, so we recommend that you avoid keywords when you are defining fields and variables.)

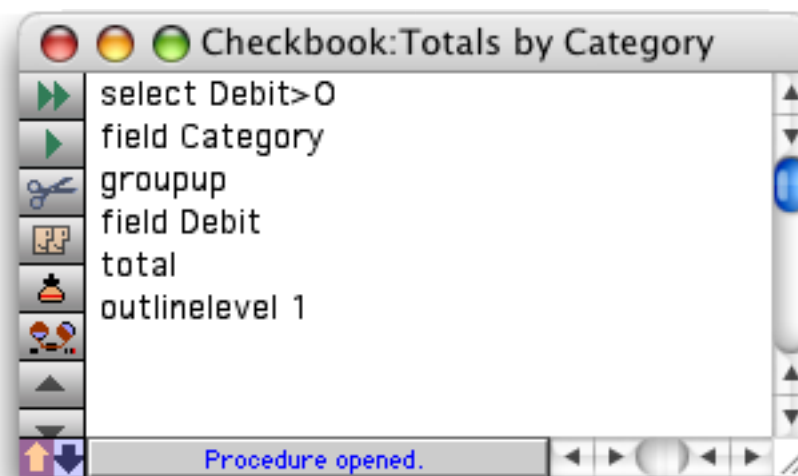
## A Simple Procedure in Action

Let's take a look at a simple procedure and see how it works. This procedure is part of a **Checkbook** database that looks like this:



Date	CkNum	PayTo	Category	Debit
01/01/99		OPENING BALANCE		
01/18/99		DEPOSIT		
01/08/99	1909	Advertiser's Mailing Ser	Advertising	390.80
01/17/99	1915	Sacramento Bee	Advertising	795.00
01/29/99	1925	Advertiser's Mailing Ser	Advertising	860.22
01/31/99	1931	Sacramento Bee	Advertising	795.00
01/08/99	1907	Northern Illinois Mold	Equipment Rental	96.05
01/17/99	1912	California Capitol	Insurance	36.00
01/17/99	1913	California Capitol	Insurance	28.00
01/22/99	1917	Blue Cross Of Calif	Insurance	279.03
01/22/99	1918	Sherman Douglas Ins	Insurance	418.60
01/16/99	1910	Coudert Brothers, Attor	Legal Fees	223.52
01/16/99	1911	Paramount Stationers	Office Supplies	105.84

The procedure itself contains six statements. In this procedure each statement is on its own line, but as you'll see later this is not necessary.



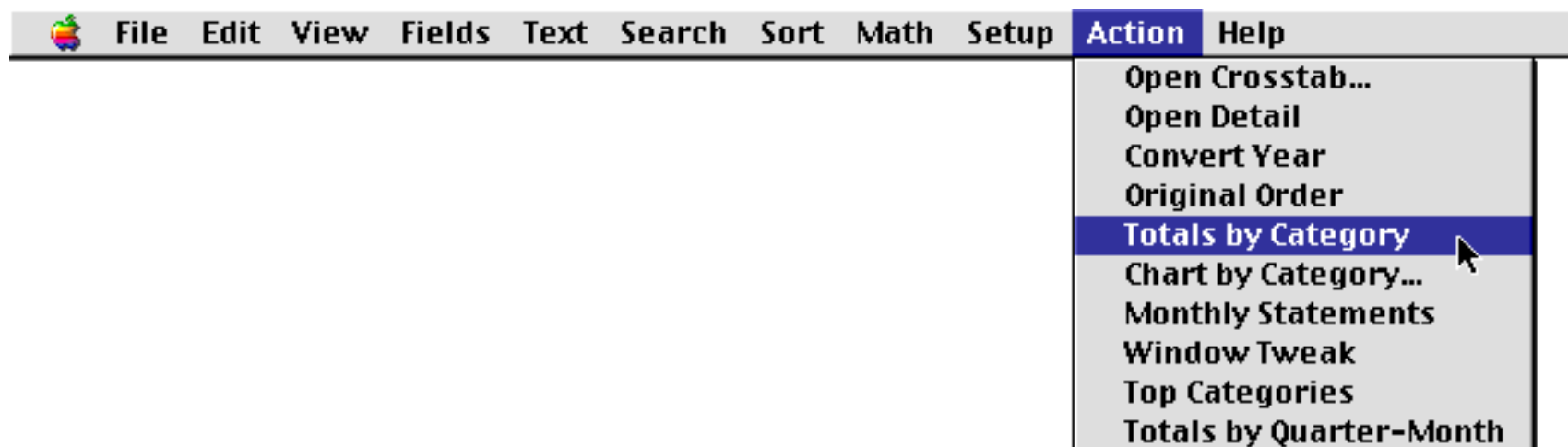
```

select Debit>0
field Category
groupup
field Debit
total
outlinelevel 1

```

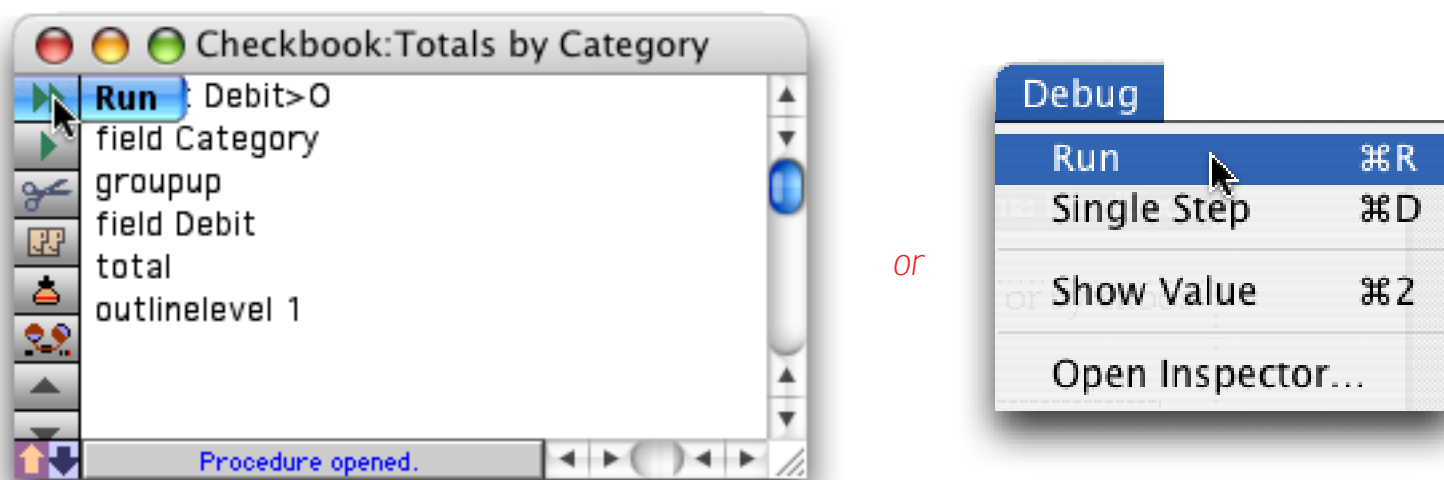
Procedure opened.

To use a procedure you need to start it somehow. (This is called “triggering” the procedure.) There are two easy ways to do this. If the data sheet is currently the top window you can trigger the procedure by choosing it from the **Action** menu.





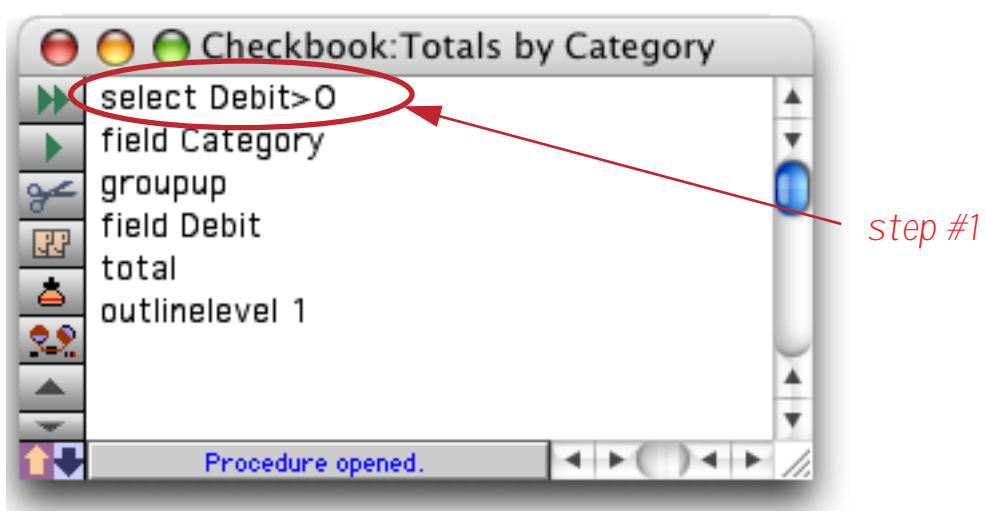
If the procedure itself is currently the top window you can trigger it by clicking on the **Run** tool, or by choosing **Run** from the **Debug** menu.



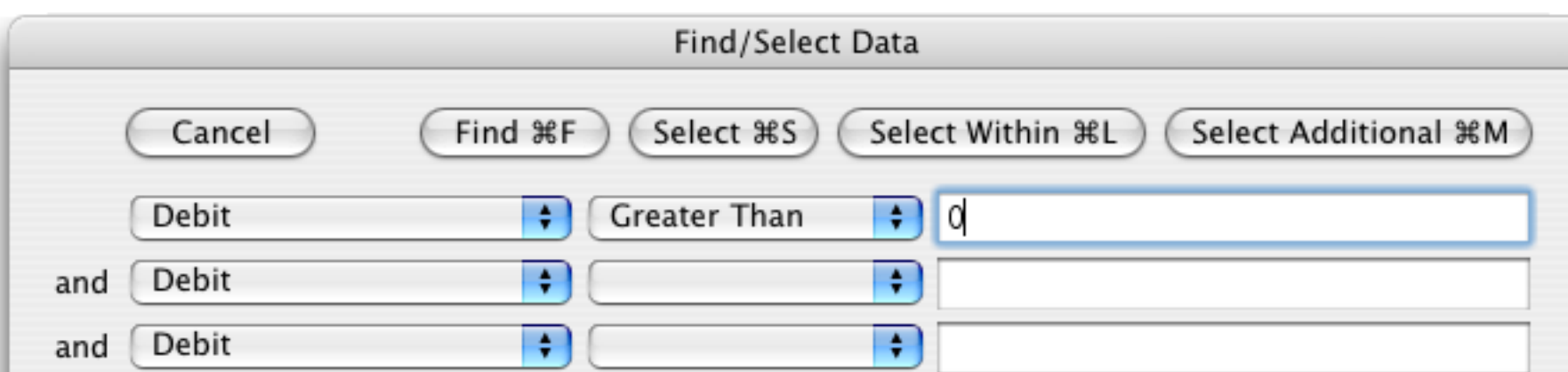
Once the procedure is triggered Panorama begins performing the steps in the procedure. You can watch as Panorama rapidly performs each step, kind of like a fast action “Keystone Kops” movie.

Let’s take a look at how Panorama performs each of the steps in our sample procedure, starting with step number 1.

```
select Debit>0
```



This first statement selects a subset of the data. Panorama performs this step exactly as if you had chosen the **Find/Select** command (see “[The Find/Select Dialog](#)” on page 335) and filled in the dialog like this. (However since Panorama already knows what to do it doesn’t actually display this dialog.)



The result of this step is that a subset of the database is now selected.

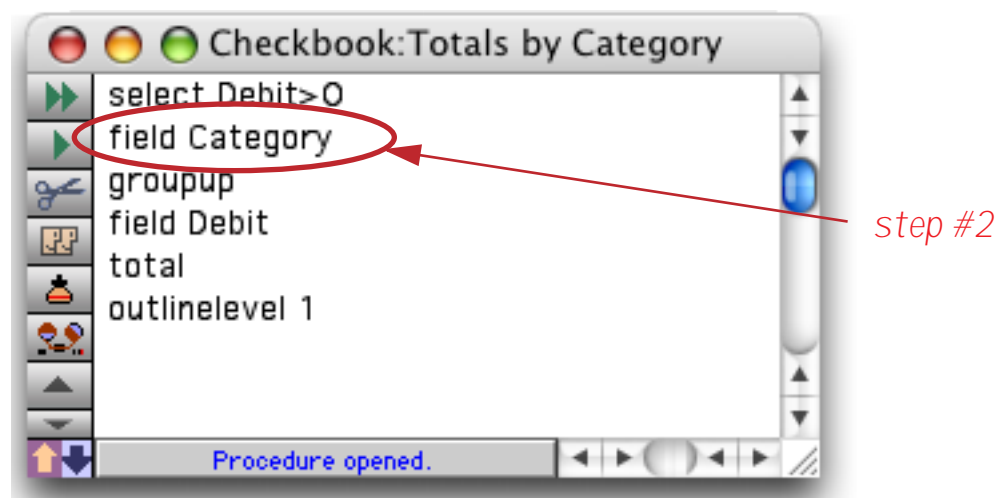


Date	CkNum	PayTo	Category	Debit
01/08/99	1909	Advertiser's Mailing Ser	Advertising	390.80
01/17/99	1915	Sacramento Bee	Advertising	795.00
01/29/99	1925	Advertiser's Mailing Ser	Advertising	860.22
01/31/99	1931	Sacramento Bee	Advertising	795.00
01/08/99	1907	Northern Illinois Mold	Equipment Rental	96.05
01/17/99	1912	California Capitol	Insurance	36.00
01/17/99	1913	California Capitol	Insurance	28.00
01/22/99	1917	Blue Cross Of Calif	Insurance	279.03
01/22/99	1918	Sherman Douglas Ins	Insurance	418.60
01/16/99	1910	Coudert Brothers, Attor	Legal Fees	223.52
01/16/99	1911	Paramount Stationers	Office Supplies	105.84
01/22/99	1919	Cannon Astro	Office Supplies	145.72
01/25/99	1921	Nebs	Office Supplies	77.27

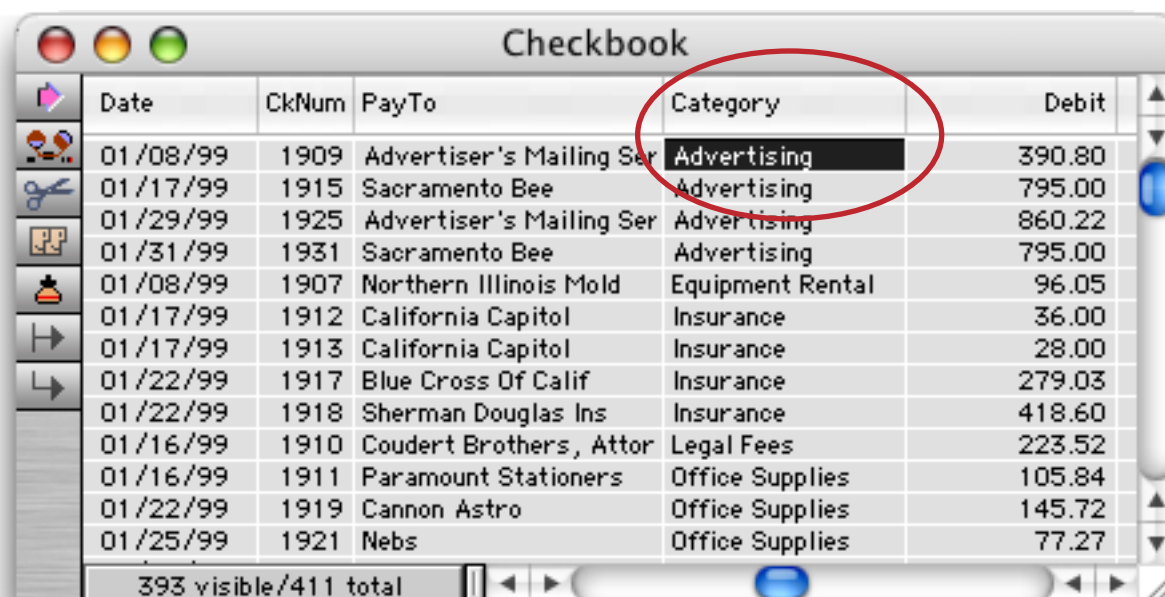
393 visible/411 total

On to step number 2.

field Category



Many Panorama operations require you to click on a field before you perform an operation. For example, to sort or group by a particular column you would first click anywhere in the column. In a procedure this is accomplished with the **field** statement. If you watch quickly you'll see the cursor jump over to the **Category** column as Panorama performs this step.

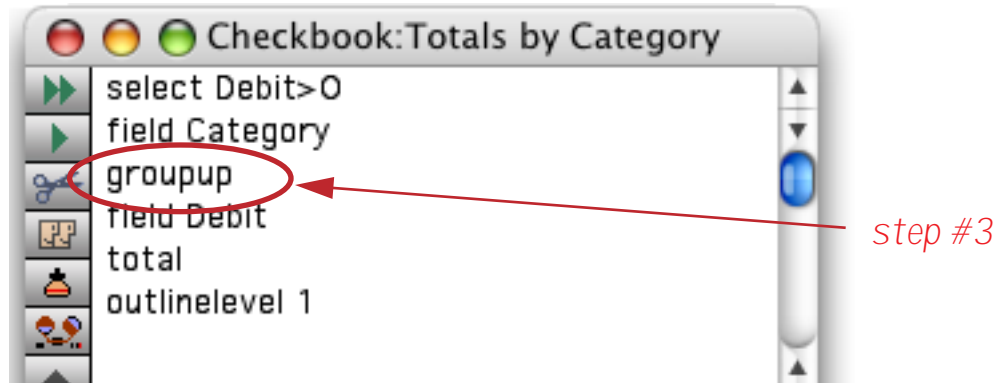


Date	CkNum	PayTo	Category	Debit
01/08/99	1909	Advertiser's Mailing Ser	Advertising	390.80
01/17/99	1915	Sacramento Bee	Advertising	795.00
01/29/99	1925	Advertiser's Mailing Ser	Advertising	860.22
01/31/99	1931	Sacramento Bee	Advertising	795.00
01/08/99	1907	Northern Illinois Mold	Equipment Rental	96.05
01/17/99	1912	California Capitol	Insurance	36.00
01/17/99	1913	California Capitol	Insurance	28.00
01/22/99	1917	Blue Cross Of Calif	Insurance	279.03
01/22/99	1918	Sherman Douglas Ins	Insurance	418.60
01/16/99	1910	Coudert Brothers, Attor	Legal Fees	223.52
01/16/99	1911	Paramount Stationers	Office Supplies	105.84
01/22/99	1919	Cannon Astro	Office Supplies	145.72
01/25/99	1921	Nebs	Office Supplies	77.27

393 visible/411 total

Now Panorama is ready for step number 3.

groupup



This statement tells Panorama to group the database, just as if you had selected **Group Up** from the Math menu (see "[STEP 1 - GROUP](#)" on page 377).

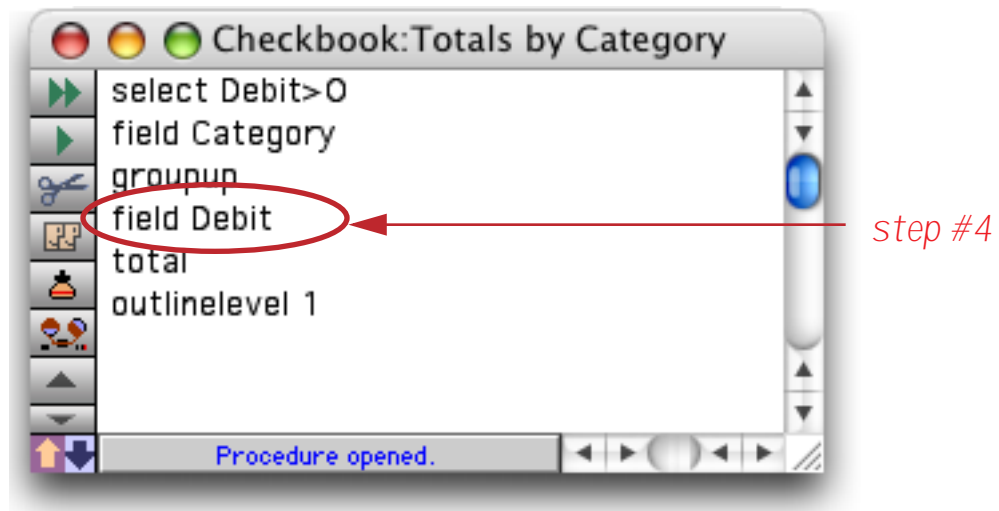
The screenshot shows the "Checkbook" window with a table of transactions. The table has columns for Date, CkNum, PayTo, Category, and Debit. The 'Advertising' category is highlighted in blue.

Date	CkNum	PayTo	Category	Debit
09/28/99	2299	Advertiser's Mailing Ser	Advertising	167.00
09/28/99	2298	Graphic Depot	Advertising	344.00
			<b>Advertising</b>	
02/01/99	1938	Unocal	Auto	182.59
02/09/99	1968	Unocal	Auto	57.62
03/16/99	2007	Unocal	Auto	33.32
05/24/99	2111	Unocal	Auto	119.05
07/16/99	2189	Unocal	Auto	38.11
07/24/99	2213	Unocal	Auto	34.44
08/20/99	2240	Unocal	Auto	89.91
			<b>Auto</b>	
01/08/99	1907	Northern Illinois Mold	Equipment Rental	96.05
02/09/99	1950	Pitney Bowes	Equipment Rental	73.14

410 visible/429 total

On to step number 4.

field Debit

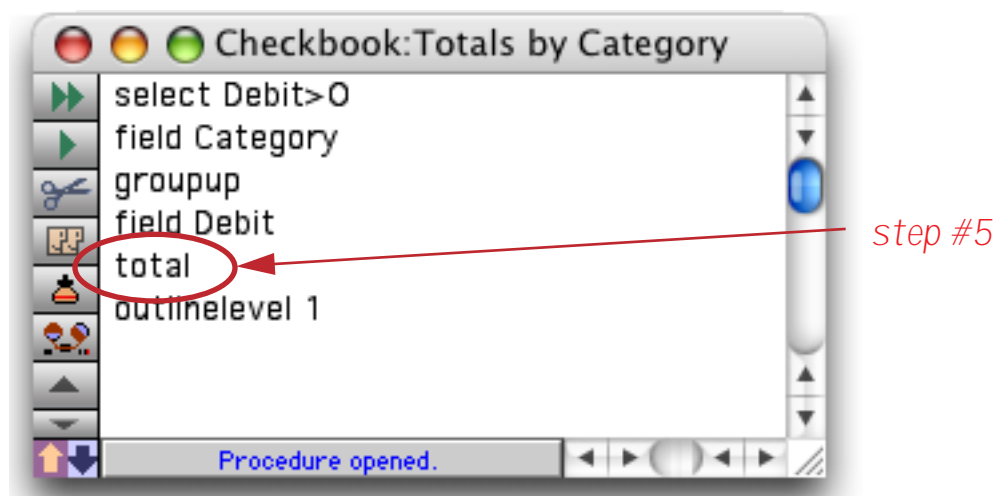


If we were performing this sequence of steps manually we would now click on the **Debit** column.

Date	CkNum	PayTo	Category	Debit
09/28/99	2299	Advertiser's Mailing Ser	Advertising	167.00
09/28/99	2298	Graphic Depot	Advertising	344.00
<b>Advertising</b>				
02/01/99	1938	Unocal	Auto	182.59
02/09/99	1968	Unocal	Auto	57.62
03/16/99	2007	Unocal	Auto	33.32
05/24/99	2111	Unocal	Auto	119.05
07/16/99	2189	Unocal	Auto	38.11
07/24/99	2213	Unocal	Auto	34.44
08/20/99	2240	Unocal	Auto	89.91
<b>Auto</b>				
01/08/99	1907	Northern Illinois Mold	Equipment Rental	96.05
02/09/99	1950	Pitney Bowes	Equipment Rental	73.14

Next is step number 5.

total

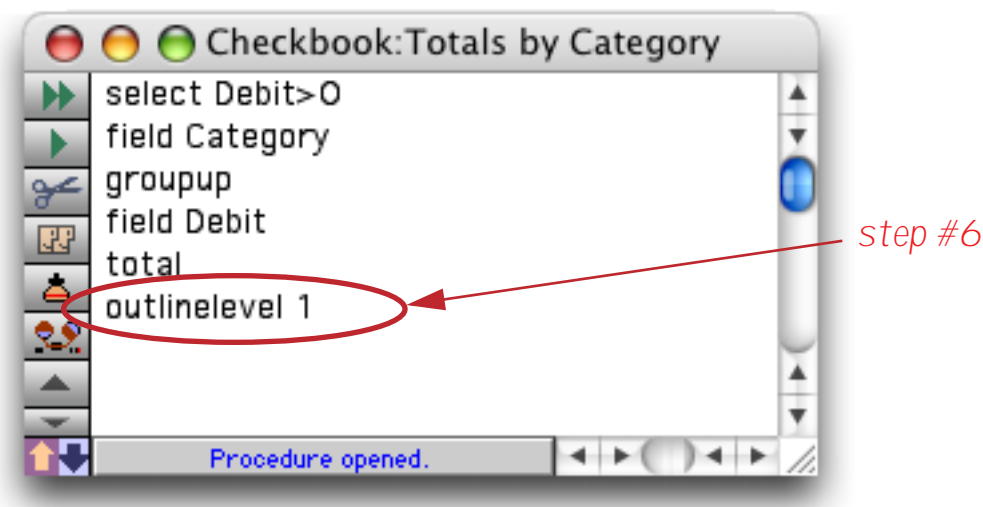


This statement is the same as choosing the **Total** command from the Math menu.

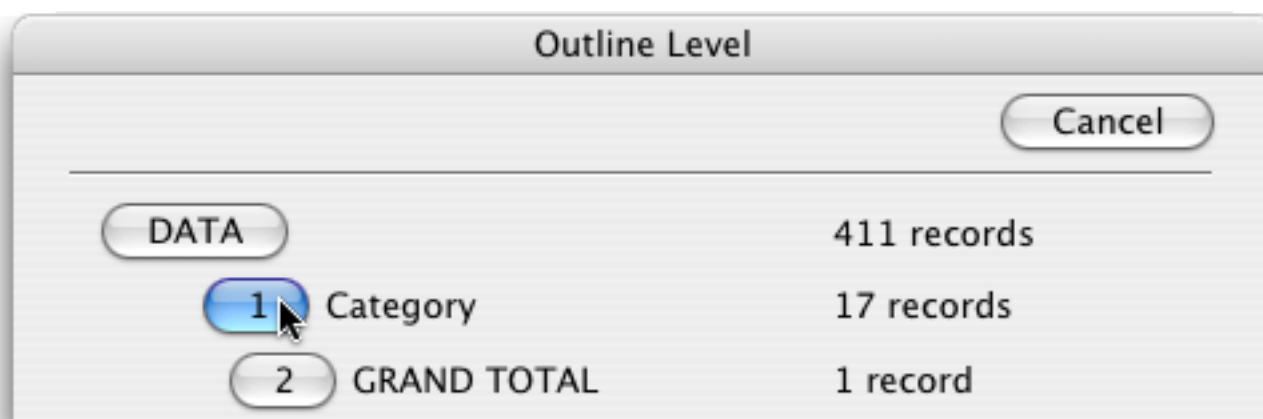
Date	CkNum	PayTo	Category	Debit
09/28/99	2299	Advertiser's Mailing Ser	Advertising	167.00
09/28/99	2298	Graphic Depot	Advertising	344.00
			<b>Advertising</b>	<b>34,516.82</b>
02/01/99	1938	Unocal	Auto	182.59
02/09/99	1968	Unocal	Auto	57.62
03/16/99	2007	Unocal	Auto	33.32
05/24/99	2111	Unocal	Auto	119.05
07/16/99	2189	Unocal	Auto	38.11
07/24/99	2213	Unocal	Auto	34.44
08/20/99	2240	Unocal	Auto	89.91
			<b>Auto</b>	<b>555.04</b>
01/08/99	1907	Northern Illinois Mold	Equipment Rental	96.05
02/09/99	1950	Pitney Bowes	Equipment Rental	73.14

And now for the final step, number 6.

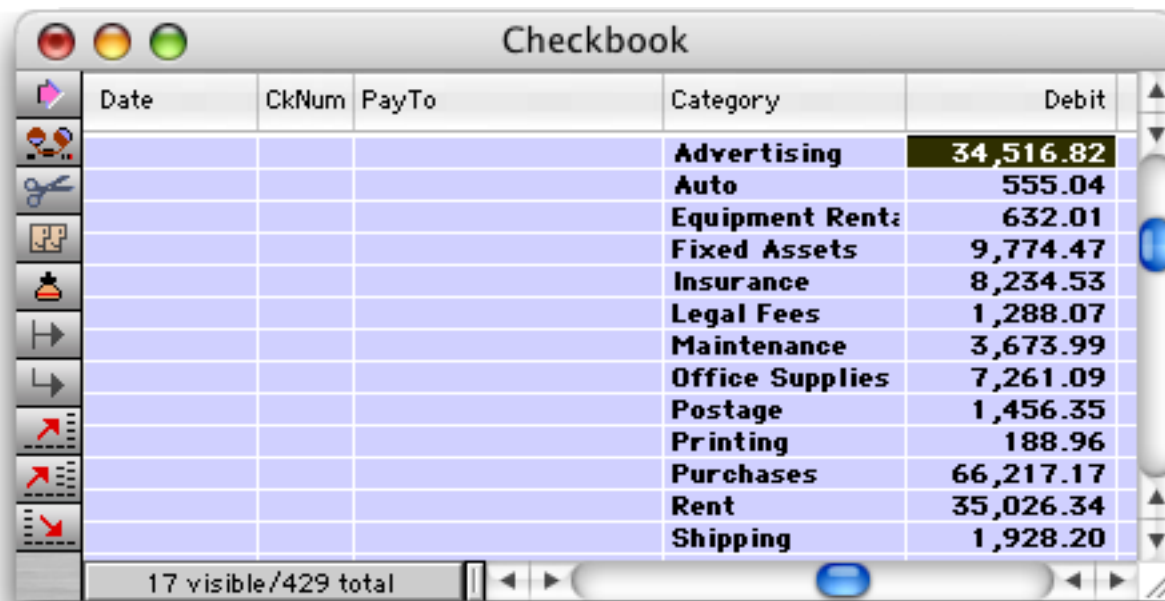
```
outlinelevel 1
```



This statement is the same as choosing the **Outline Level** command from the Sort menu. The parameter on this statement, **1**, tells Panorama to simulate pressing the **1** button in the dialog. (Once again, since Panorama already knows what to do it doesn't actually display the dialog.)



Here's the final result after all six statements have completed.



Date	CkNum	PayTo	Category	Debit
			<b>Advertising</b>	<b>34,516.82</b>
			<b>Auto</b>	<b>555.04</b>
			<b>Equipment Rent:</b>	<b>632.01</b>
			<b>Fixed Assets</b>	<b>9,774.47</b>
			<b>Insurance</b>	<b>8,234.53</b>
			<b>Legal Fees</b>	<b>1,288.07</b>
			<b>Maintenance</b>	<b>3,673.99</b>
			<b>Office Supplies</b>	<b>7,261.09</b>
			<b>Postage</b>	<b>1,456.35</b>
			<b>Printing</b>	<b>188.96</b>
			<b>Purchases</b>	<b>66,217.17</b>
			<b>Rent</b>	<b>35,026.34</b>
			<b>Shipping</b>	<b>1,928.20</b>

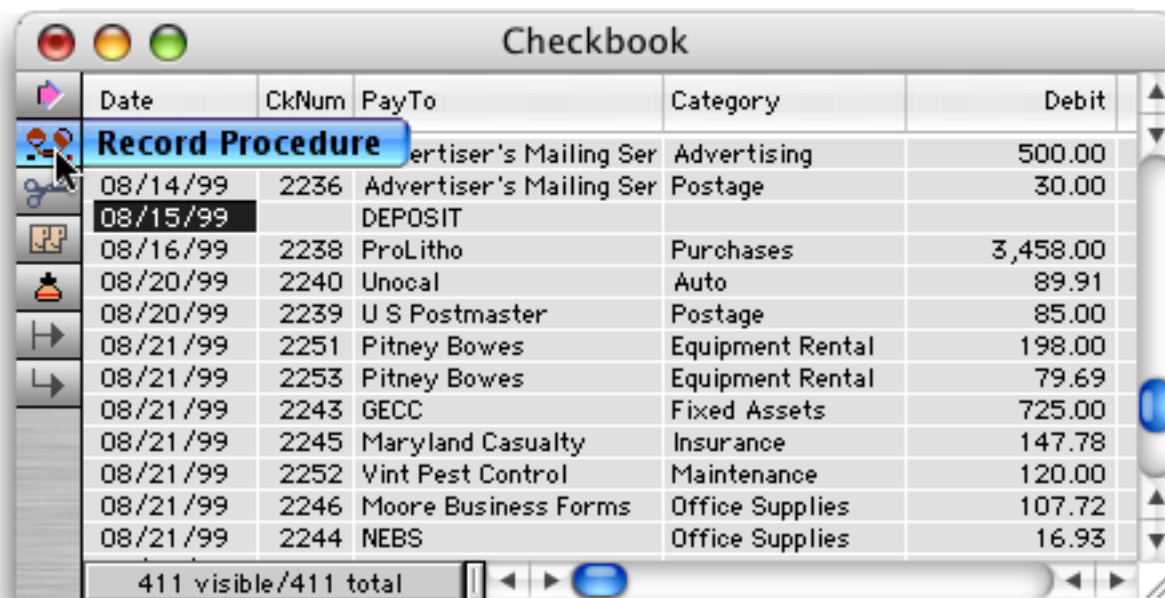
17 visible/429 total

Pretty simple, is it not? This is the basic operation of any procedure — first the procedure is triggered, then Panorama performs each statement from top to bottom. (Later you'll learn several ways to change the top to bottom order when it is necessary, see "[Control Flow](#)" on page 601.)

### Creating a Procedure with the Recorder

A basic procedure like the one in the last section is very easy to create with Panorama's built in procedure recorder. The procedure recorder is like a tape recorder that records your actions as you work. Recording a procedure is a four step process— 1) start the recorder, 2) perform the steps while Panorama records, 3) stop the recorder, and 4) give the new procedure a name.

To start the procedure recorder, click the **Record Procedure** tool (available in the Data Sheet and Form tool palettes (unless you are in Graphics Mode)).

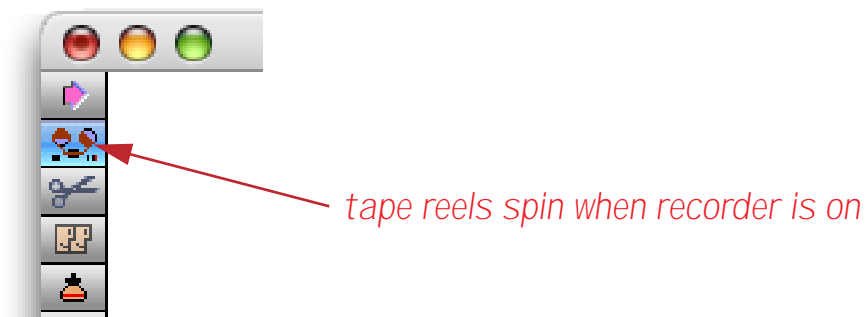


Date	CkNum	PayTo	Category	Debit
		Advertiser's Mailing Ser	Advertising	500.00
08/14/99	2236	Advertiser's Mailing Ser	Postage	30.00
08/15/99		DEPOSIT		
08/16/99	2238	ProLitho	Purchases	3,458.00
08/20/99	2240	Unocal	Auto	89.91
08/20/99	2239	U S Postmaster	Postage	85.00
08/21/99	2251	Pitney Bowes	Equipment Rental	198.00
08/21/99	2253	Pitney Bowes	Equipment Rental	79.69
08/21/99	2243	GECC	Fixed Assets	725.00
08/21/99	2245	Maryland Casualty	Insurance	147.78
08/21/99	2252	Vint Pest Control	Maintenance	120.00
08/21/99	2246	Moore Business Forms	Office Supplies	107.72
08/21/99	2244	NEBS	Office Supplies	16.93

411 visible/411 total



The “reels” on the recorder tool will begin to spin. This lets you know that Panorama is recording your actions.



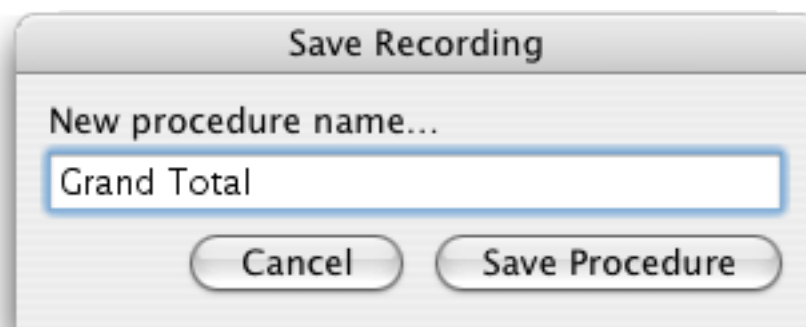
Once the recorder is running just continue to use Panorama normally. Panorama will record every menu command or tool that you use. To demonstrate the recorder we'll create a short procedure that calculates the grand total. Start by clicking anywhere in the **Debit** field.

Date	CkNum	PayTo	Category	Debit
08/14/99	2237	Advertiser's Mailing Ser	Advertising	500.00
08/14/99	2236	Advertiser's Mailing Ser	Postage	30.00
08/15/99		DEPOSIT		
08/16/99	2238	ProLitho	Purchases	3,458.00
08/20/99	2240	Unocal	Auto	89.91
08/20/99	2239	U S Postmaster	Postage	85.00
08/21/99	2251	Pitney Bowes	Equipment Rental	198.00
08/21/99	2253	Pitney Bowes	Equipment Rental	79.69
08/21/99	2243	GECC	Fixed Assets	725.00
08/21/99	2245	Maryland Casualty	Insurance	147.78
08/21/99	2252	Vint Pest Control	Maintenance	120.00
08/21/99	2246	Moore Business Forms	Office Supplies	107.72
08/21/99	2244	NEBS	Office Supplies	16.93

Next choose **Total** from the Math menu. Panorama calculates the total.

Date	CkNum	PayTo	Category	Debit
09/19/99	2288	MCI	Telephone	67.59
09/19/99	2290	City Of Caboose	Utilities	103.15
09/19/99	2291	S C E	Utilities	81.13
09/19/99	2292	So. Calif. Gas Co.	Utilities	154.95
09/21/99	2294	Advertiser's Mailing Ser	Postage	167.00
09/21/99	2295	Advertiser's Mailing Ser	Postage	67.00
09/26/99	2297	AC Label Company	Advertising	205.97
09/26/99	2296	TesLabe	Fixed Assets	2,465.00
09/28/99	2299	Advertiser's Mailing Ser	Advertising	167.00
09/28/99	2298	Graphic Depot	Advertising	344.00
<b>08/23/04</b>	<b>2310</b>			<b>183,651.22</b>

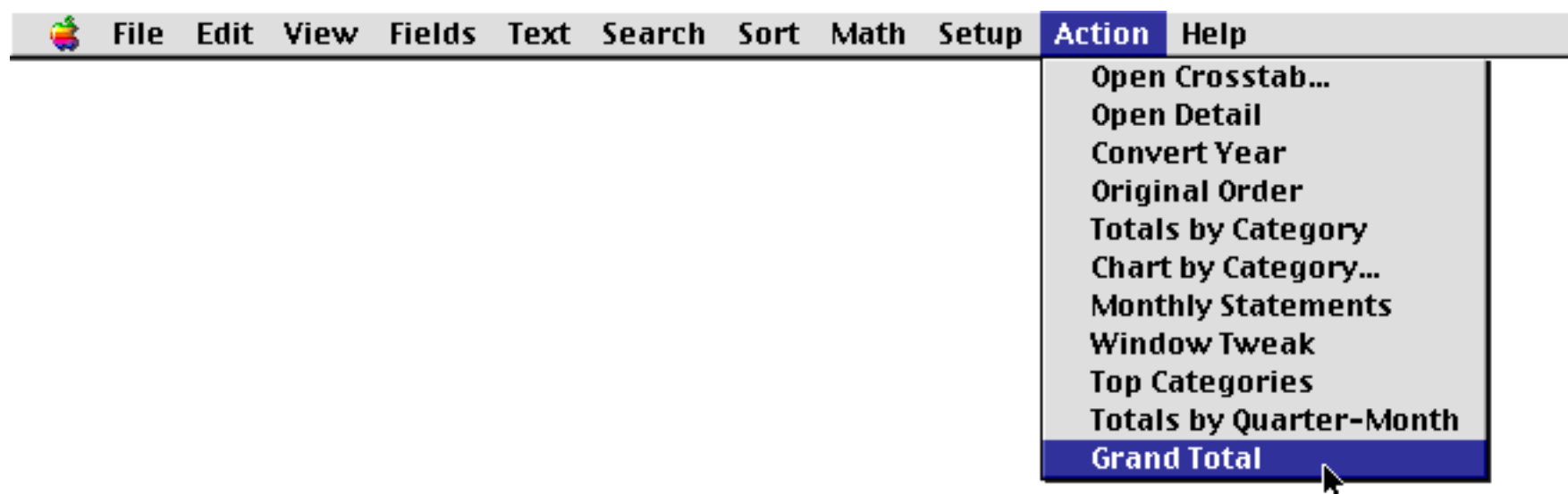
Our simple procedure is complete. To stop the recorder, click on the **Record Procedure** tool again. The “reels” will rewind and stop and a dialog box appears. This dialog box allows you to give your new procedure a name, up to 25 characters. Pick a name that will help you remember what the procedure does, for example **Print Invoices** or **Balance Checkbook**.



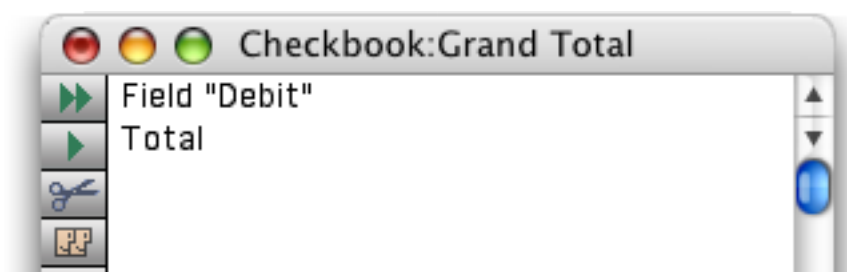
Once you have entered the name, press **Save Procedure**.

Tip: There are five characters you should not use in a procedure name unless you know what you are doing. The five characters are: ^ ; < ( /. Later in this manual you'll learn how these characters can be used to create special effects in the **Action Menu** (see “[Action Menu Options](#)” on page 637).

Once you have given the recording a name, the new procedure is added to the end of the **Action Menu**. (If the database doesn't already have a **Action Menu**, it will be created.)



You can play back your new procedure at any time by selecting it from the **Action Menu**. To view the statements in the new procedure you can open it with the **View** menu (hold down the **Control** key (Mac) or **Alt** key (PC) to open the procedure in its own new window.)



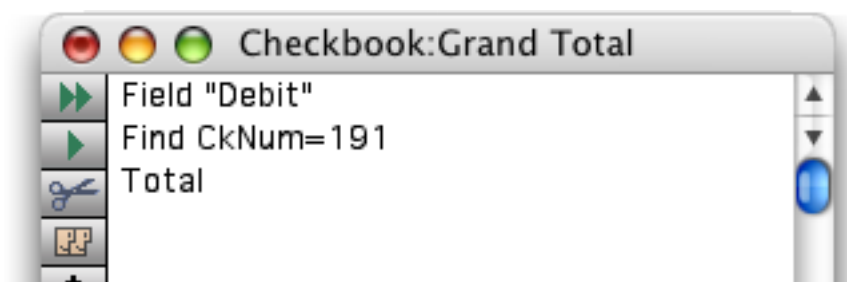
Our simple recording contains two steps. You can use the procedure “as is” or you can customize it further.

### Recording Mouse Clicks

Panorama does not normally consider clicking the mouse to be a step—clicking the mouse is not a menu command or tool. However, if you click on a different window or a different field within the current window, Panorama will record that step. The recorder will ignore all other mouse clicks (for example, clicking on the scroll bar, dragging a window to a new position or changing the size of a window, etc.).

Panorama never records the row position of a click. If you look back at the previous section you'll notice that I clicked on check number 1915 when I recorded the procedure. However, this information was not recorded. When the procedure is played back Panorama will not move to check number 1915, but will stay on whatever check it is already on. The **total** statement doesn't care as long as the column (field) is correct.

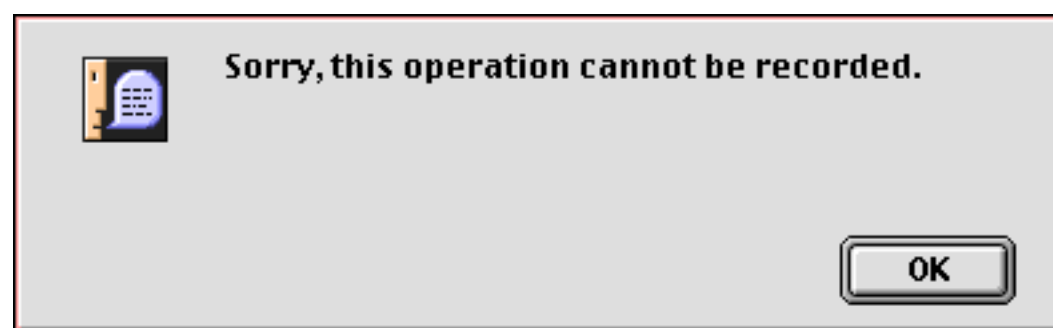
If you do want Panorama to move to a specific record you'll need to use the **find** statement (see "**FIND**" on page 5245). You can either create this during recording using the **Find/Select** dialog (see "**The Find/Select Dialog**" on page 335) or simply by typing it into the procedure window. Here's a modified version of the procedure that moves to check number 1915.



Of course this example doesn't make much sense because Panorama will only be on check number 1915 for a fraction of a second before the **total** statement makes Panorama jump to the end of the database.

### Non Recordable Menus and Tools

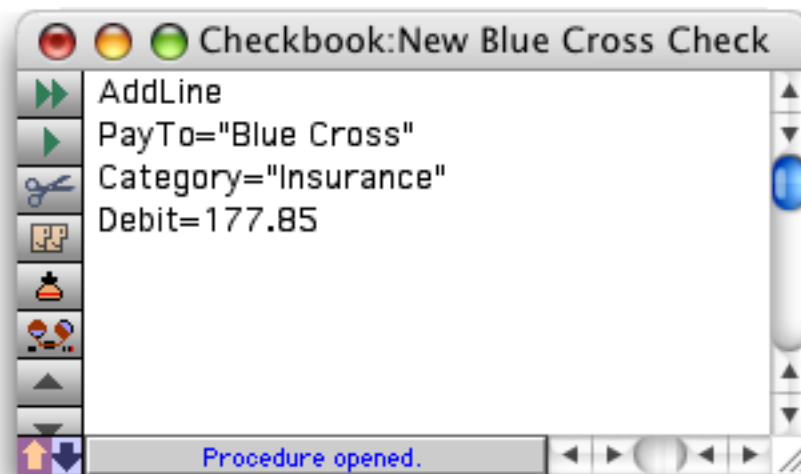
A couple of pages ago we told you that Panorama records every menu and tool when the recorder is on. Sorry, but that was a lie. Some menus and tools are not recordable. Most commands and tools that work with graphics cannot be used. For example, **Sort Up**, **Insert Record**, and **Print** can be used as steps in a procedure, but **Bring to Front**, **Oval**, and **Align** cannot. In general, only actions that affect data can be included in a procedure. If the recorder is on and you attempt to use a menu command or tool that cannot be used as a step in a procedure, Panorama will alert you.



As long as you stick with the Data Sheet and Data Access Mode Forms you'll usually be fine.

### Recording Data Entry

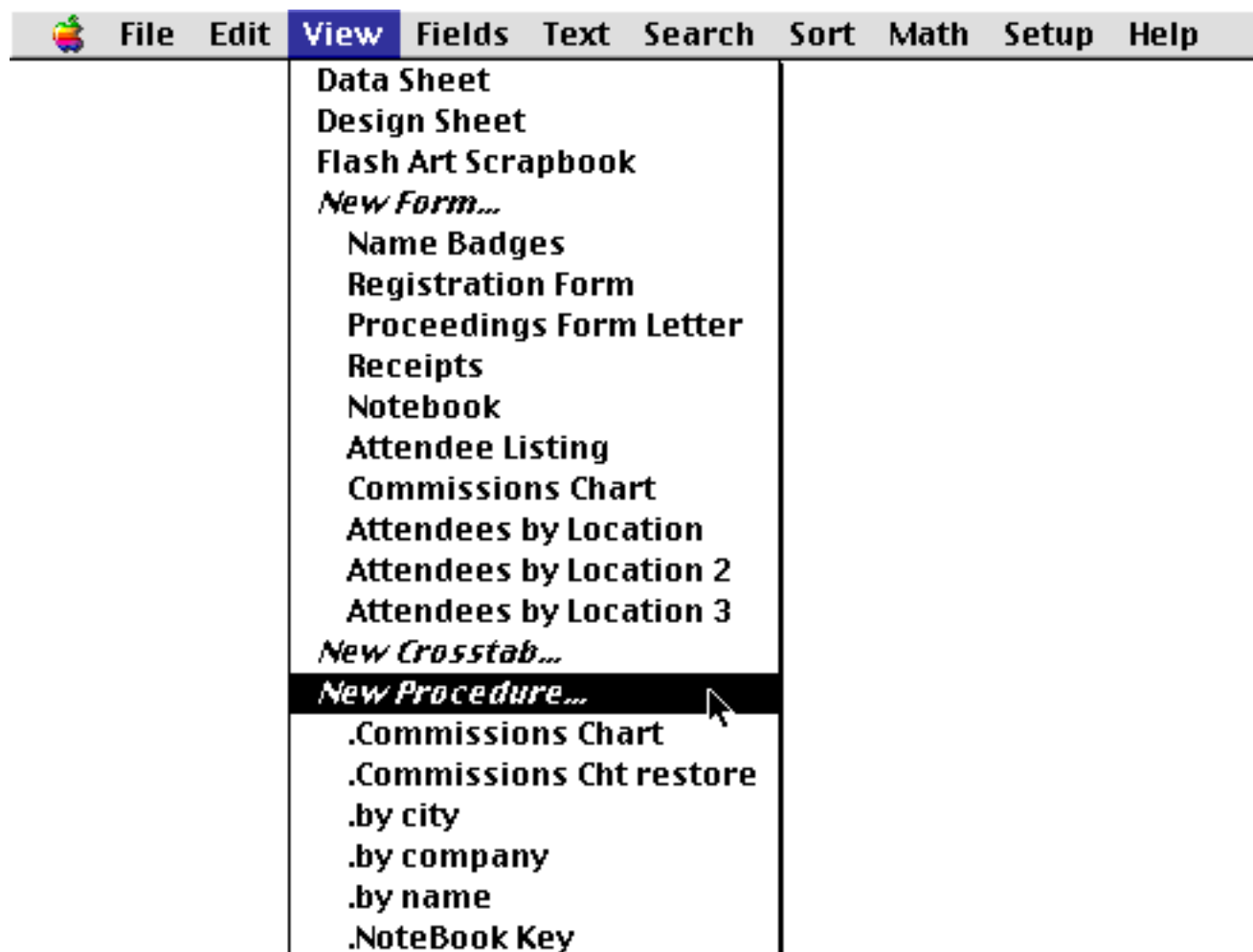
Panorama doesn't do a very good job of recording data entry. Instead of using the recorder we recommend that you use assignment statements to perform data entry, as shown in this example.



You cannot record these assignment statements, you have to type them in manually. See “[Assignment Statements](#)” on page 596 for more information on assignment statements.

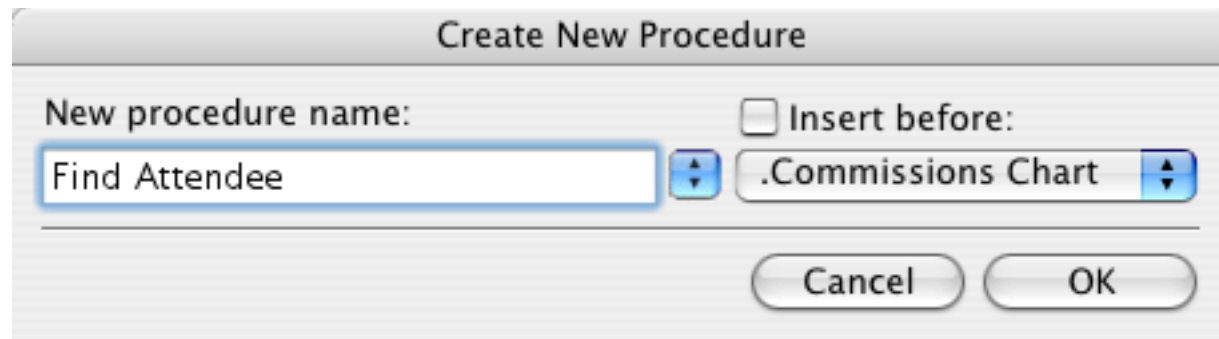
### Writing a Procedure from Scratch

If you want to write a procedure from scratch (instead of using the recorder), the first step is to create a new, empty procedure. To do this select **New Procedure** from the **View** menu.



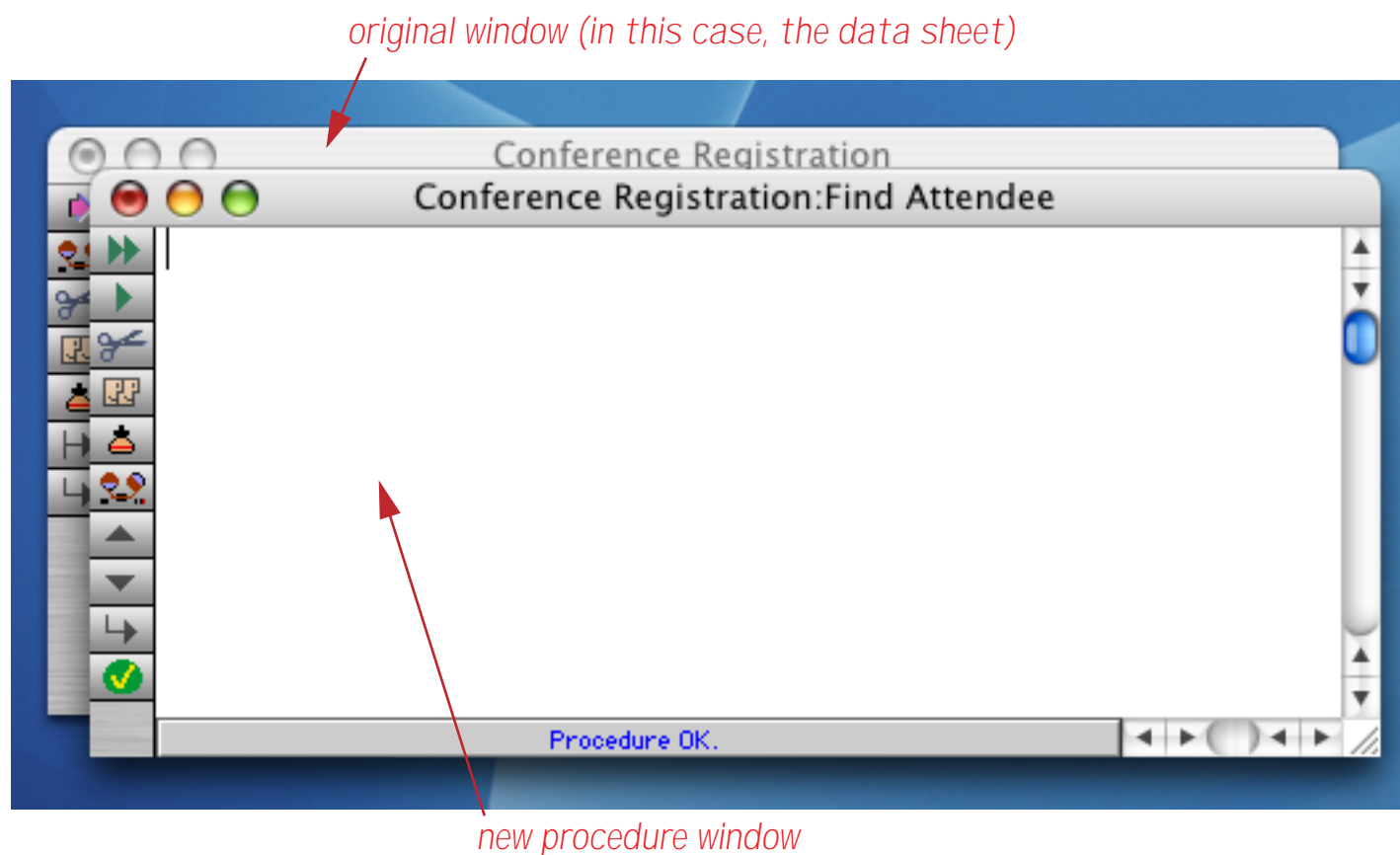
Selecting **New Procedure** normally creates the new procedure in the same window you are currently in. It's often convenient to create the new procedure in a new, separate window, leaving the original window open. That way you can easily flip back and forth between your database window (data sheet or form) and the procedure window. To open the new procedure in a separate window hold down the **Control** key (Macintosh) or **Alt** key (Windows) as you click on the **View** menu.

After you select **New Procedure** this dialog appears.



Type in the name of the new procedure, then press **OK**. The name may be up to 25 characters long, and must be unique within this database. Tip: There are five characters you should not use in a procedure name unless you know what you are doing. The five characters are: ^ ; < ( /. Later in this manual you'll learn how these characters can be used to create special effects in the **Action Menu** (see "[Action Menu Options](#)" on page 637).

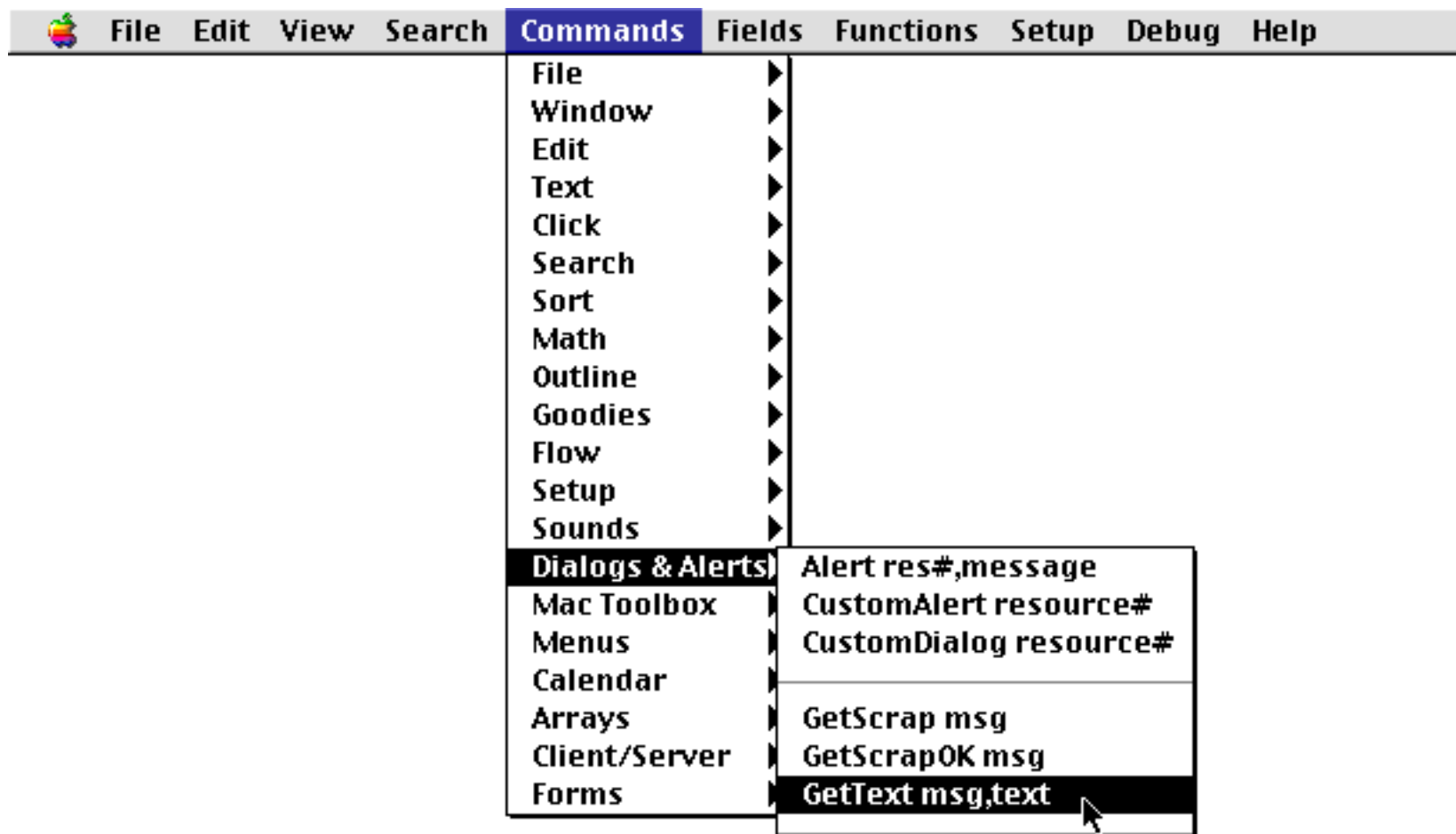
After you press the **OK** button a new, empty procedure is created, and the window switches to show you this new procedure. If you held down the **Control** key (Mac) or **Alt** key (Windows) the new procedure will open in a new window just below and to the right of the original window, like this.



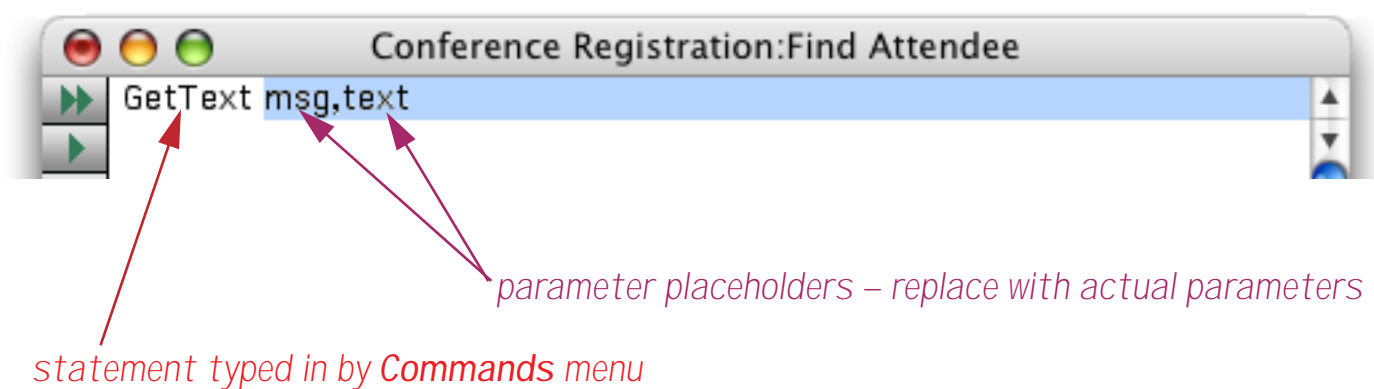
## Writing Statements

Once you have created an empty procedure you can start adding statements to the procedure. If you know the keywords for the statements you want to use, just type them in. For example, if you want to sort the database in ascending order just type in the statement `SortUp`. By the way, the capitalization of statements doesn't matter, so you could also type `sortup`, `SORTUP`, or even `SoRTuP`. However, keywords are always a single word with no blanks, so `sort up` will not work.

If you don't know the exact keywords for the operations you want the procedure to perform, you have a several choices. You could look up the keyword in this manual, then type it in from the keyboard. Or you could locate the keyword in the **Commands** menu and let Panorama type in the keyword for you. (A third choice is to use the **Programming Reference** wizard, see "[Programming Reference Wizard](#)" on page 591.)



You'll still have to type in any options and parameters yourself, but the **Commands** menu will type in placeholders for these values to help remind you that they are necessary (we'll talk more about options and parameters later).

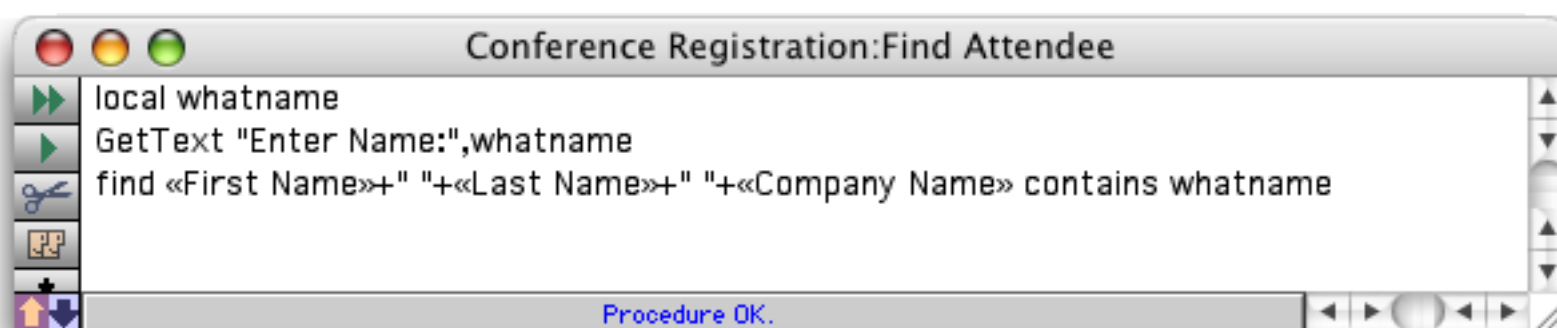


Here is the finished statement with the actual parameters typed in.





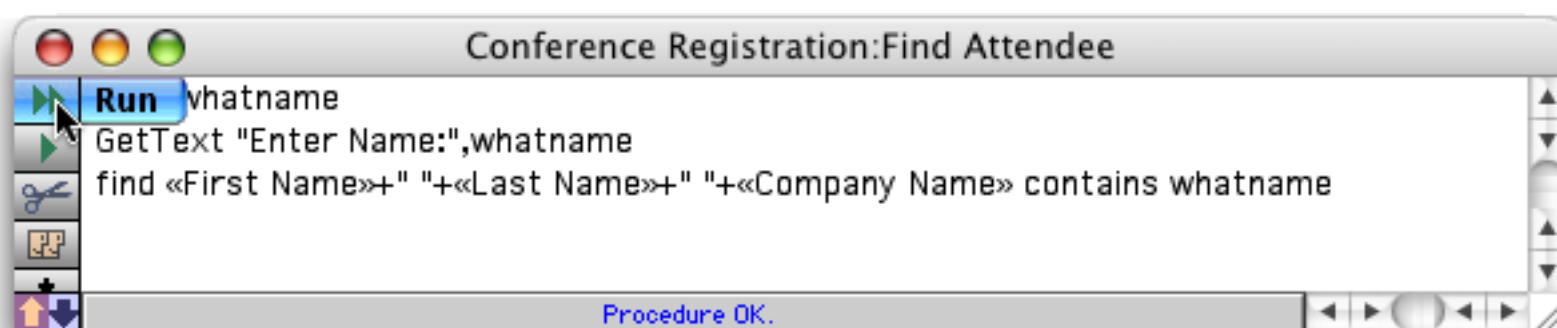
Here is the completed procedure.



This procedure has three statements. The `local` statement creates a local variable named `whatname`. The `gettext` statement displays a dialog asking to enter a name. Whatever is typed in is placed in the `whatname` variable. The `find` statement searches the database to locate the requested name.

### Trying Out a Procedure

The easiest way to try out a procedure you are working on is to press the **Run** button (you can also press **Command-R** on a Mac or **Control-R** on a PC).



When you run a procedure this way Panorama will automatically switch to a window that contains data. Any window that allows you to display and edit data will do (as long as it is in the same database), so Panorama will pick the topmost one that will work. If there is no window that will work (for example if the procedure is in the only window for this database), Panorama will display an error message and the procedure will not run.

In this case Panorama will switch to the data sheet window and begin performing the steps in the procedure, starting with the topmost statement.

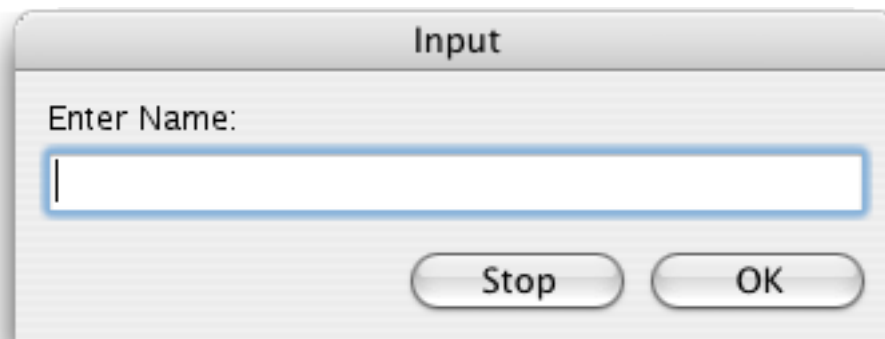
```
local whatname
```

This statement allocates a local variable named `whatname` for temporary storage (see “[Variables](#)” on page 1189). This operation is completely invisible.

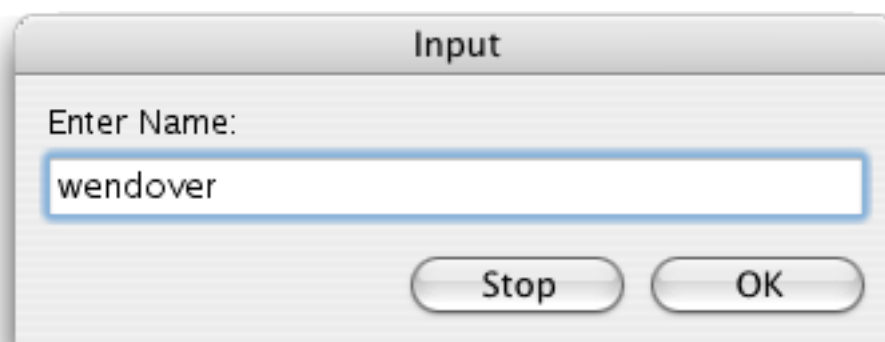
On to step 2 —

```
GetText "Enter Name:",whatname
```

This statement displays a dialog asking the user to type in some text.



Enter the name you want to search for.



When you press **OK** the dialog disappears and the procedure continues on with the next statement. Before it does, however, it copies the text that was typed in into the variable named `whatname`.

```
find «First Name»+" "+"«Last Name»+" "+"«Company Name» contains whatname
```

This final statement searches three fields in the database to see if they contain whatever text was typed in. This database does contain the name `wendover`, so Panorama jumps to that record. The name is circled in the illustration below to make it more clear.

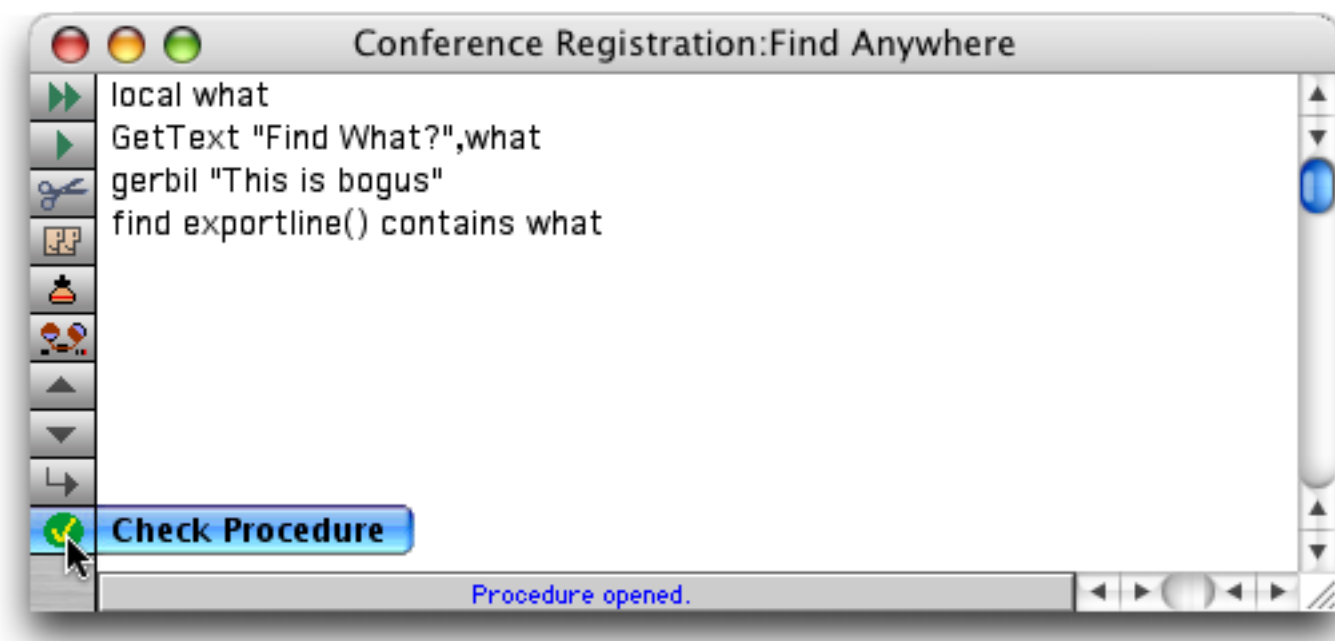
T	First Name	Last Name	Company Name	Street Address	Suite Box	City	Stat
	Ms. Christy	Alpert	Signal Research	1120 Sharon Pa		Cupertino	CA
	Mr. Arthur	Clairmont	South Coast Office Produ	4390 Kaiser Dr		Cupertino	CA
	Mr. Harold	Cobb	Cobb Associates	3912 Phillip St.		Cupertino	CA
	Mrs. Sherry	Grossman	Pablo Distribution	1400 Valley Riv		Cupertino	CA
	Mr. Peter	Parks	Hamilton Press	41 Kenosia Ave		Cupertino	CA
	Mrs. Michelle	Adams	Sceptre	10159 Alliance		Cupertino	CA
	Mrs. Kathy	Schwartz	Wendover Insurance Grou	814 Castro St.		Long Beach	CA
	Mr. Charles	Arrow	Arrow, Inc.	390 Davis St.		Los Angeles	CA
	Mr. Dave	Elko	First Row Group	547 Jocom Way		Los Angeles	CA
	Mrs. Cindy	Blunden	Hot Lines, Inc.	#6 Hoover Pk		Palo Alto	CA
	Mr. Robert	Dorn	Valley Services	33 Cambridge P		Palo Alto	CA
	Mrs. Roxie	Jacobsen	Alpha Pic	174 Bellevue A		Palo Alto	CA
	Mr. Russ	Quade	Challenger Air Comm	220 Diver Road		Palo Alto	CA

Now you have an easy way to locate any person in this database without having to bother with the **Find/Select** dialog.

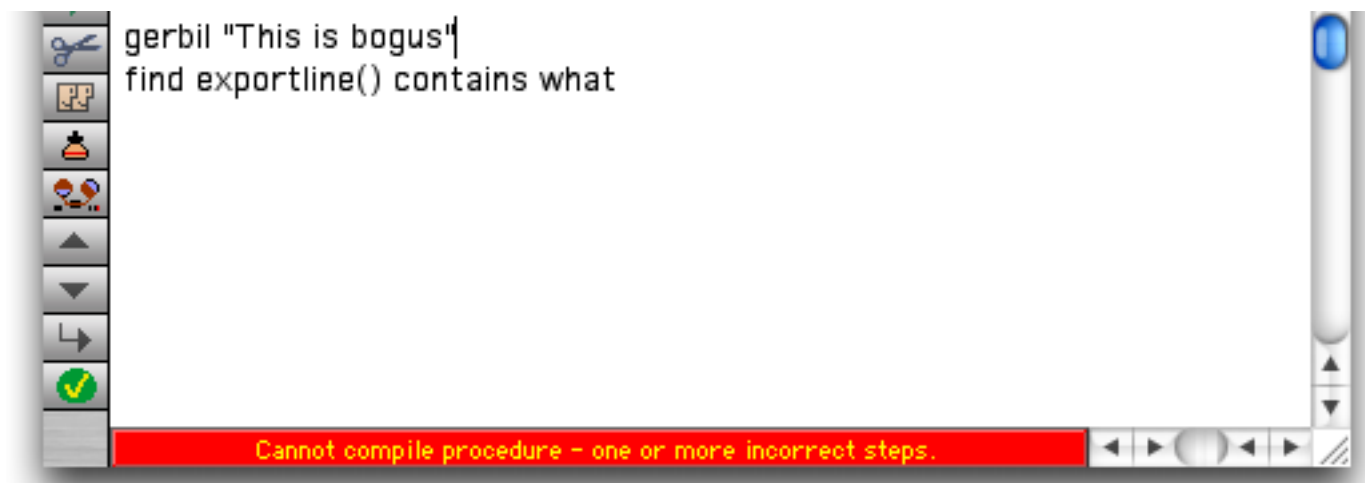
### Checking for Mistakes

If you write a procedure yourself without using the recorder, you may make a mistake. You might misspell a keyword, forget to include a parameter, leave off a closing parenthesis, etc. Panorama will not let you use a procedure until you find and fix all of these mistakes.

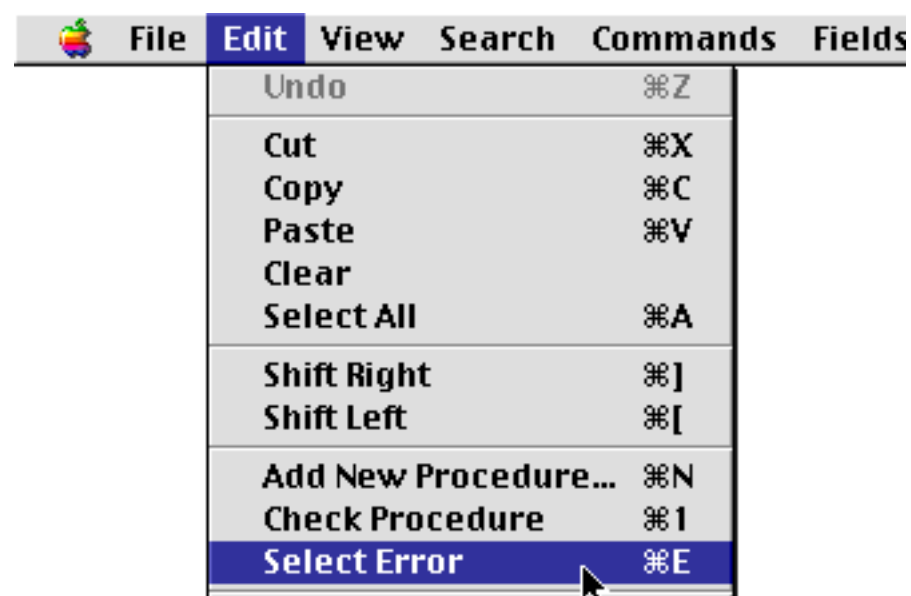
To help you find these mistakes Panorama provides the **Check Procedure** tool or menu command (Edit menu).



This tool scans the procedure looking for misspelled keywords and other mistakes. If it finds an error the status bar turns red and contains a description of the error.



Choose **Select Error** from the Edit menu if you would like Panorama to attempt to highlight the location of the error for you.



Panorama will identify the spot where it thinks the error occurred (see “[Mysterious Errors](#)” on page 589).



Correct the error, then use the **Check Procedure** tool again to see if there are any more mistakes. Repeat this process until the **Check Procedure** tool no longer finds any mistakes in your procedure. (Note: Panorama also automatically checks your procedure whenever you click on another window, save the database, switch to another view, or close the window containing the procedure.)

### Mysterious Errors

Usually the **Check Procedure** tool in combination with **Select Error** is able to pinpoint the exact spot where the mistake in your procedure is located. Some types of mistakes, however, are not detectable right away, so Panorama actually will highlight the wrong spot in the procedure. Usually this is caused by a missing **endif** statement, mismatched parentheses, or mismatched quotes. If Panorama tells you that there is an error but the spot highlighted looks ok to you, check carefully above the spot where the error was flagged. The actual error may be many lines above the spot Panorama has flagged. If you still can't find the error, try splitting the procedure into several smaller procedures and checking each piece separately until you find the section containing the error.

### Closing the Window When a Procedure is Finished

When you have finished writing a procedure you'll probably want to close the window for that procedure. If it is in a separate window, just click on the close box. If it is the only window for the database use the **View** Menu to flip to another view. You don't have to close the procedure window to use the procedure, but when you are sure it is working properly you will probably want to close it just to cut down on window clutter.

### Re-Opening a Procedure

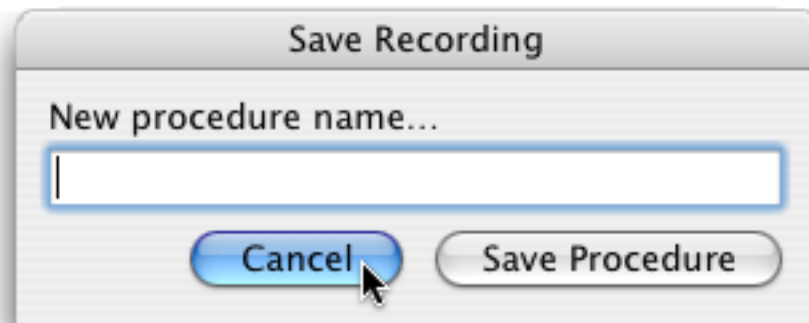
You can change any procedure at any time. Simply open the procedure with the **View** Menu, then make the changes. Don't forget that you can hold down the **Control** key (Macintosh) or **Alt** key (Windows) to make the procedure open into its own separate window.

### Font and Size

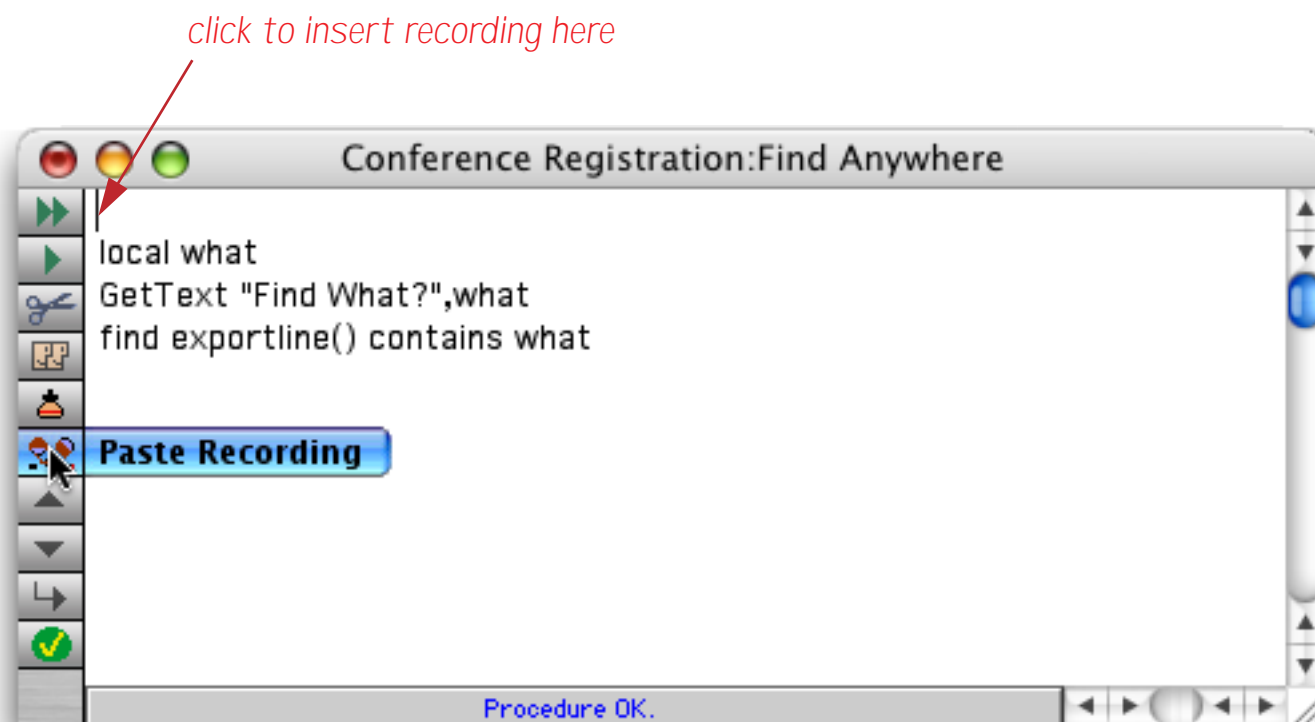
The **Font** and **Size** submenus (Edit Menu) allow you to change font and size of the procedures in this database. The font and size are always the same for every procedure in the database—you cannot set different procedures in the same database to different fonts or sizes.) If you are using a Windows system we recommend that you stick with one of the four fonts installed with Panorama: Alpine, Block, City or Yankee (the default is Alpine 12). These fonts are designed to be able to display some of the special characters used by Panorama that are not normally available on Windows systems ( $\neq$ ,  $\leq$ ,  $\geq$  etc.)

### Adding a Recording to an Existing Procedure

Earlier you learned how to create a new procedure by recording (see “[Creating a Procedure with the Recorder](#)” on page 579). It’s also possible to use the recorder to add statements to an existing procedure. To do this, start the recorder normally, and record the steps you want to include. When the steps have been completed click on the recorder again to stop the recording. When Panorama asks you what name you want to give the new procedure, click the **Cancel** button.



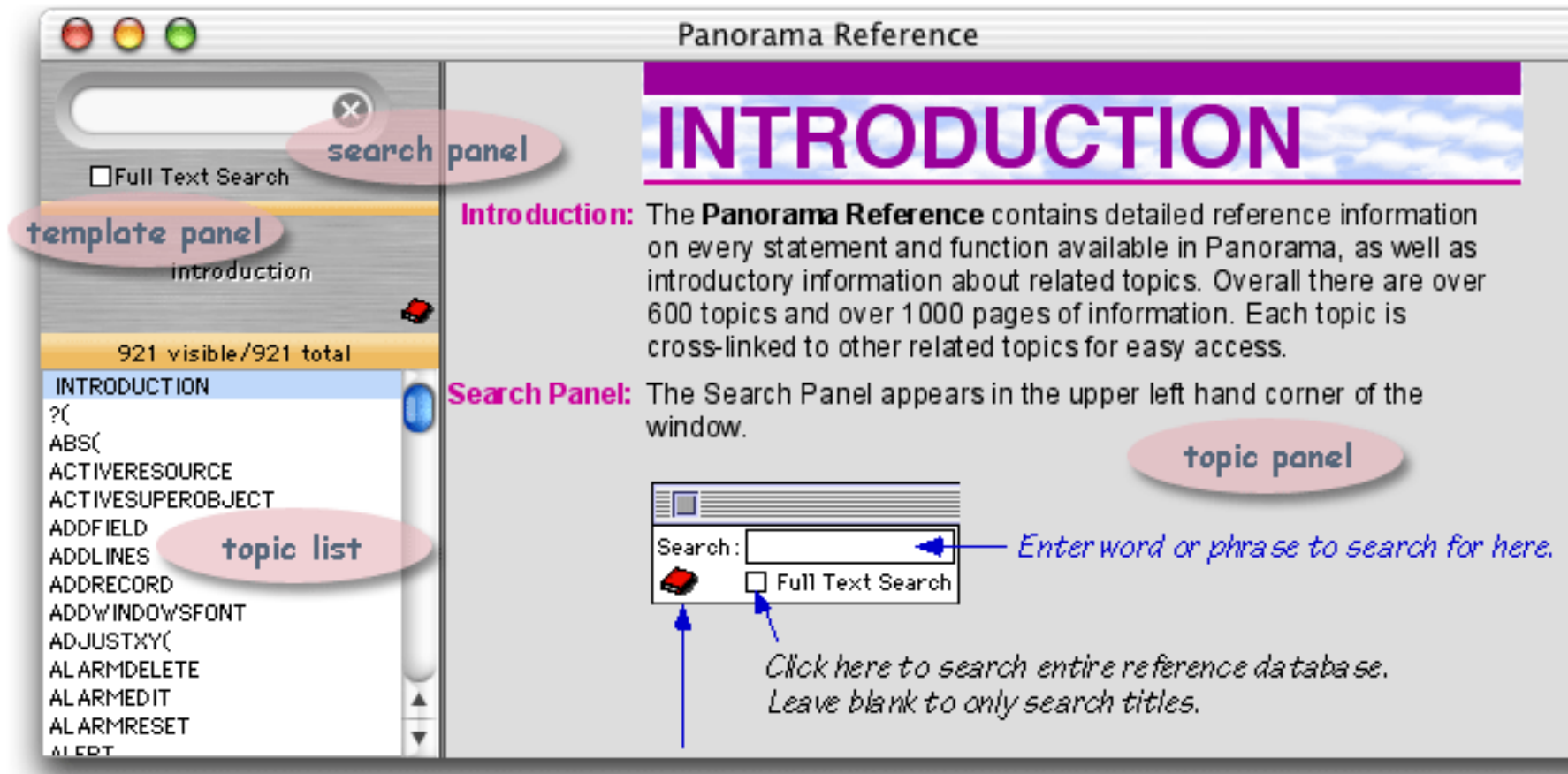
Now go to your procedure window and click on the spot where you want the recording to be inserted. Then choose the **Paste Recording** tool.



Panorama will insert the recording into the procedure. If necessary you can edit the recorded statements or use them “as is.”

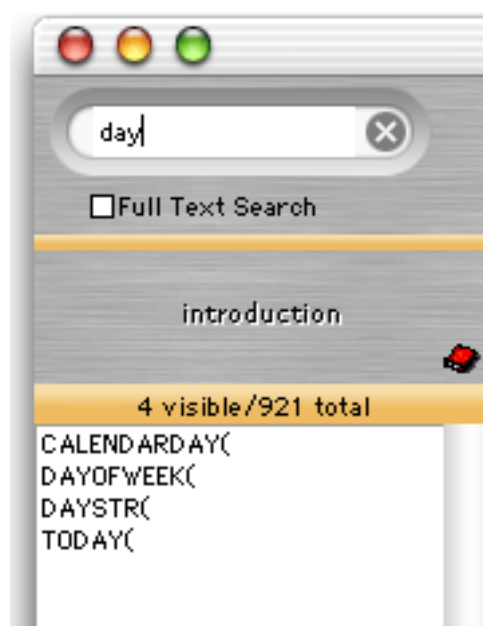
## Programming Reference Wizard

Panorama has over one thousand functions and statements that you can use in formulas and procedures. Most of these have one or more parameters that must be used correctly. Even here at ProVUE Development we can't keep all of this memorized. To help, we created the **Programming Reference** wizard, an instant, on-line reference to what's what in Panorama formulas and programs. You can open this wizard from the Documentation submenu of the Wizard menu, or by pressing **Control-R** (Macintosh only). Once the wizard opens the reference window is divided into four sections: search panel, template panel, topic list and topic panel.



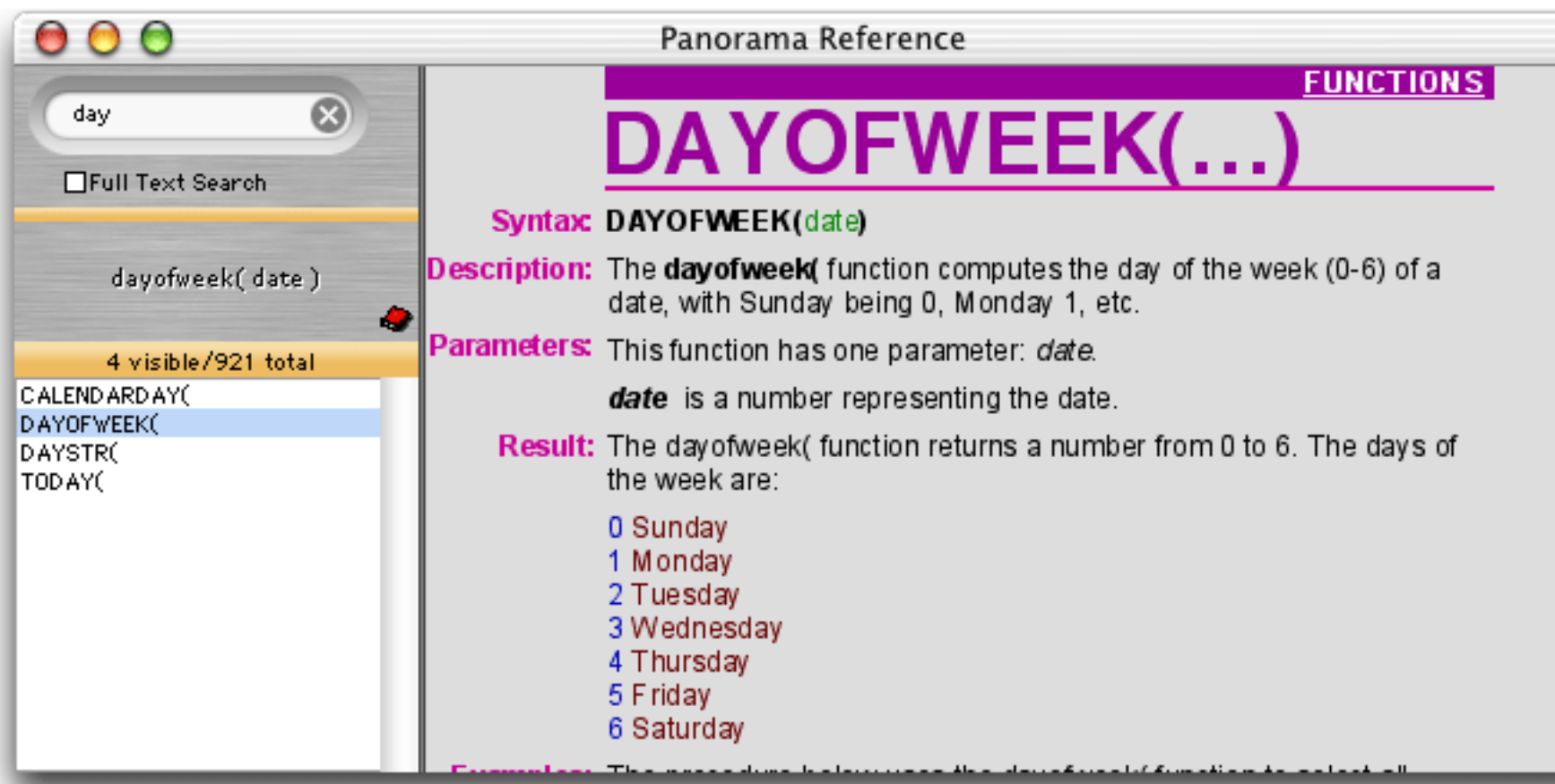
### Navigation Using the Search Panel and Topic List

The search panel and topic list work together to help you locate a specific topic. As you type into the search panel, the topic list updates to show topics that match.

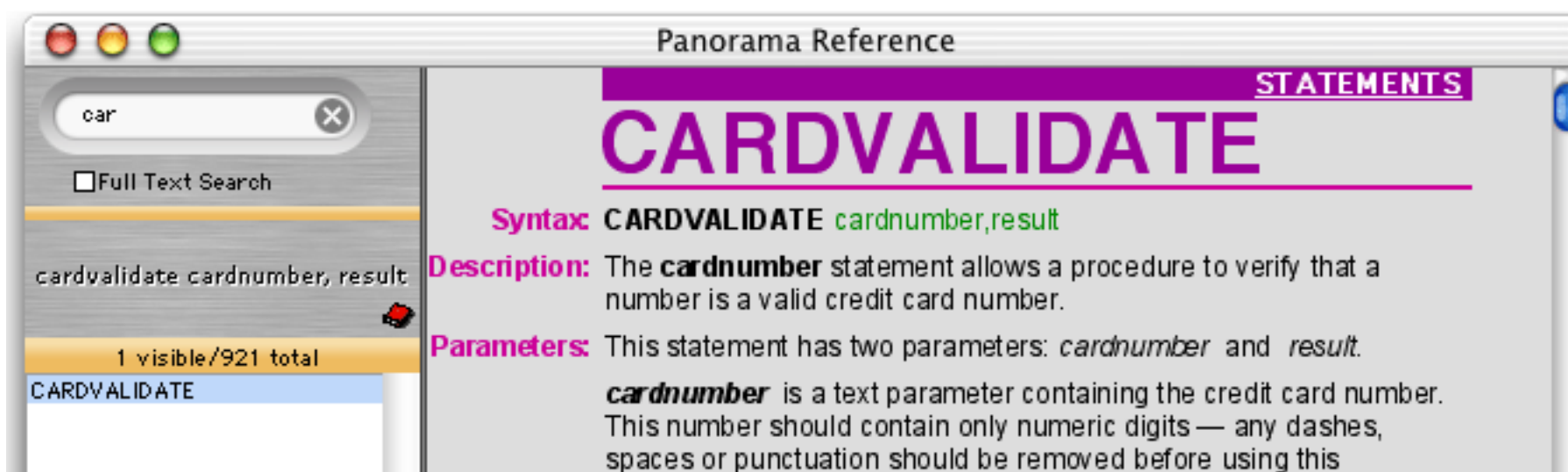




When you see the topic you want, click on it to display the topic in the topic panel.



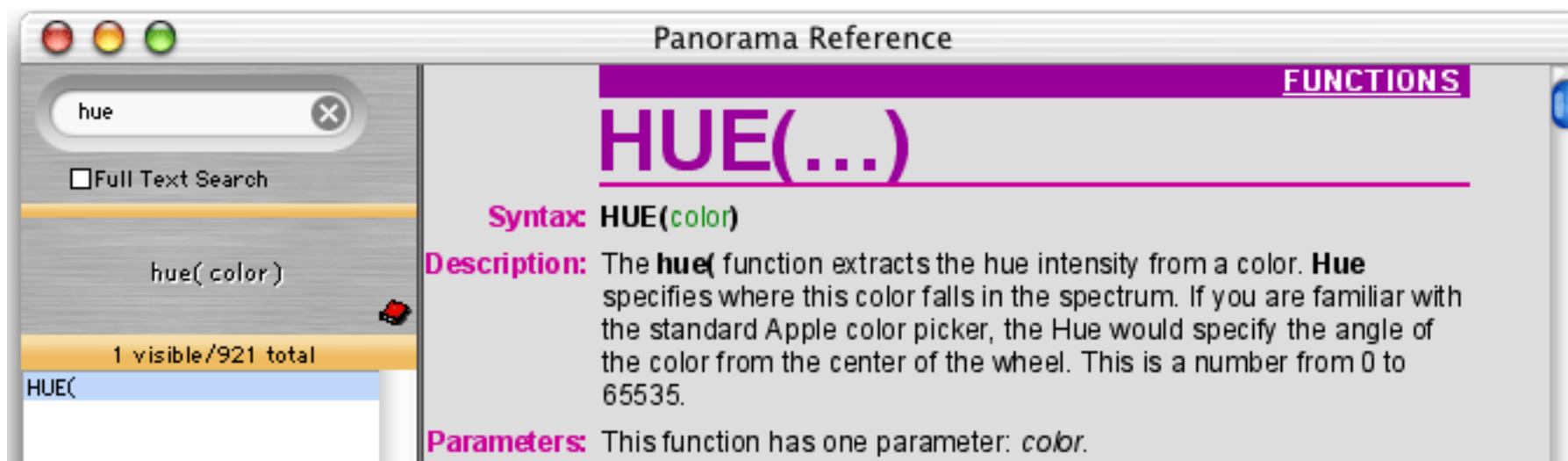
If there is only one topic in the topic list, the wizard will display the topic automatically (without having to click on the list).



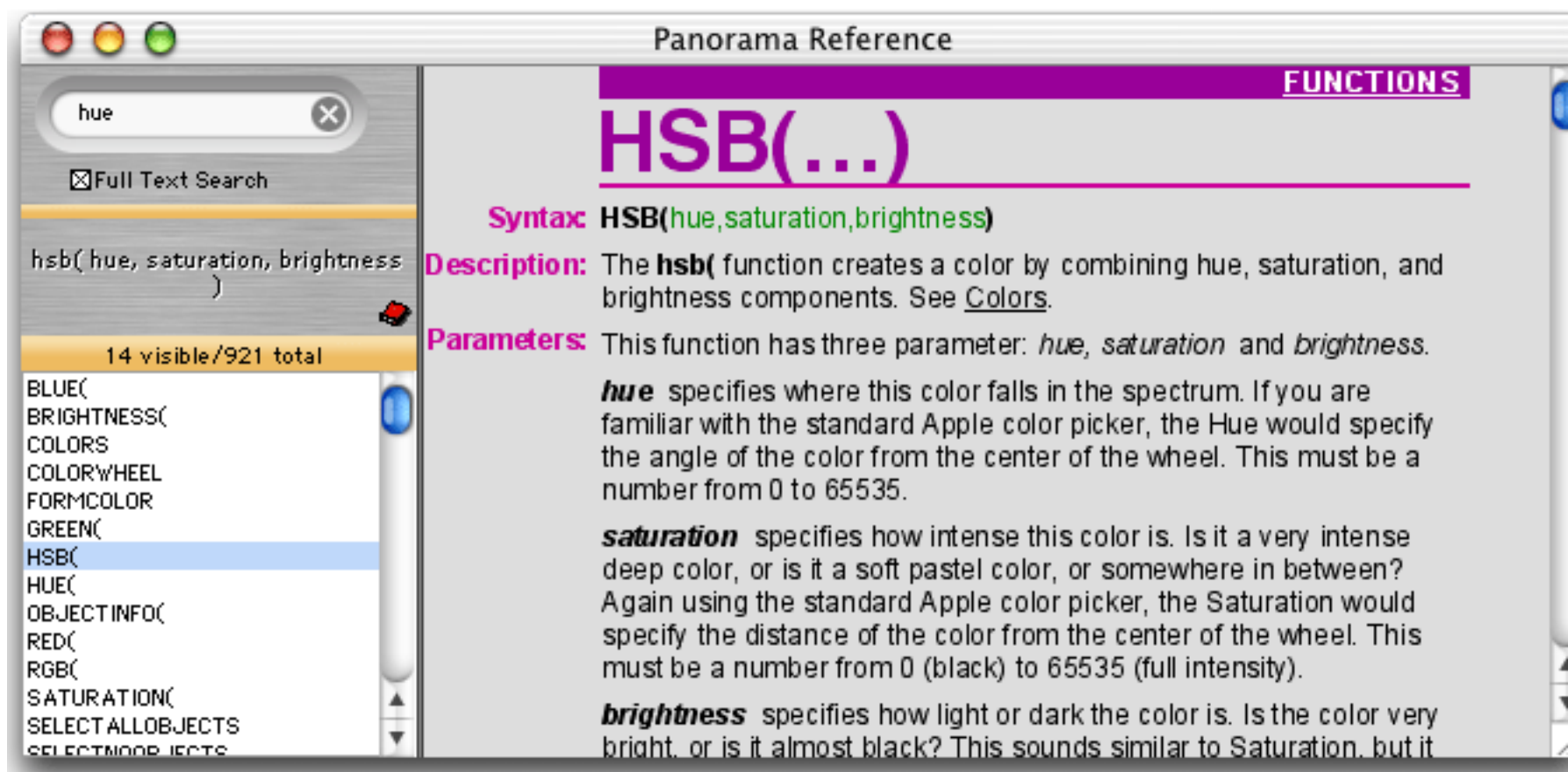
To quickly erase the query in the search panel, click on the  button.

## The Full Text Search Option

The search panel normally searches only the name and category of each topic. When the **Full Text Search** option is checked the wizard will also search the complete text of each topic. This makes it possible to quickly find every topic that references a particular function or statement, as well as the function or statement itself. For example, a normal search for the word **hue** will turn up only one match.



Repeating the search with the **Full Text Search** option turned on yields 14 matches. You can click on the match you are interested in.

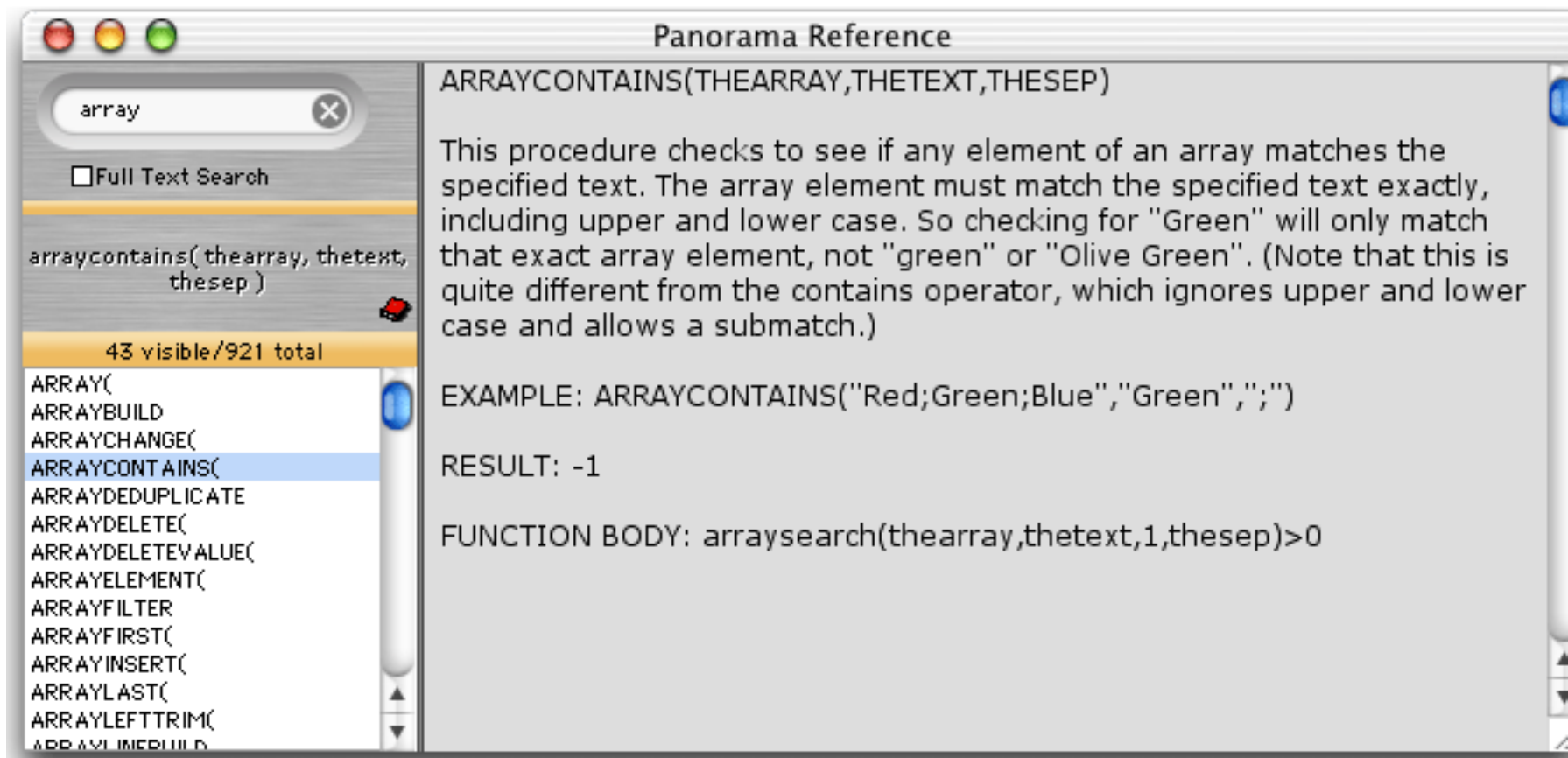


## Navigation Using HyperLinks

Like a web browser, the **Programming Reference** contains links from one page to other related topics. These links are underlined in the text. To jump any linked topic simply click on the underlined text.

## Built In vs. Custom Statements and Functions

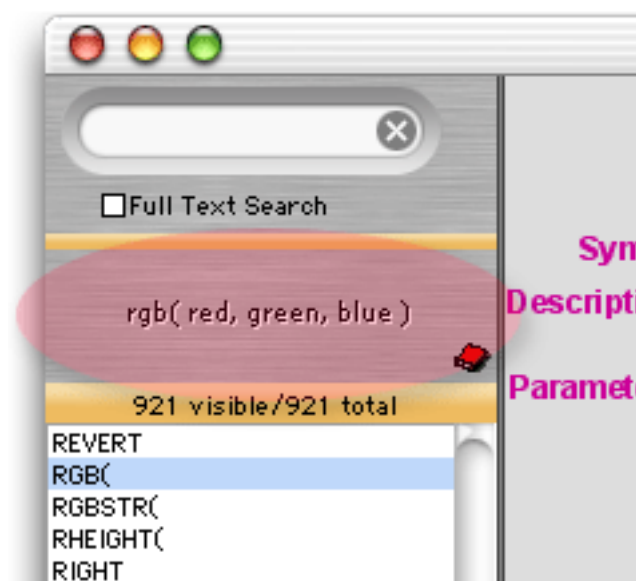
Panorama supports both built in and custom statements and functions. Several hundred custom statements and functions are included with Panorama, and these are also included as topics in the **Programming Reference** wizard. (You can also create your own custom statements and functions, but these are not included in the **Programming Reference** wizard.) For most custom statements and functions the topic panel uses a basic "plain text" format instead of the more graphical format used for built-in statements and functions.



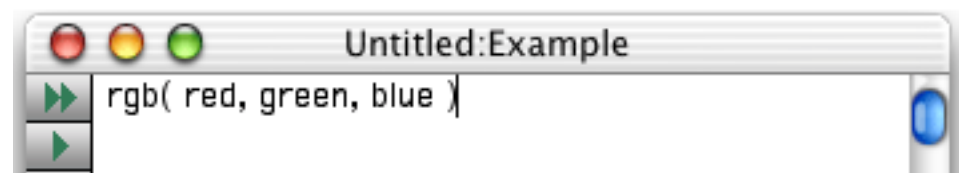
Don't adjust your set -- this plain text view is normal for custom statements and functions.

## Using the Template Panel

The template panel displays a sample that illustrates how this statement or function would be used in a formula or procedure. In this case the panel shows an example of the `rgb()` function, which has three parameters.




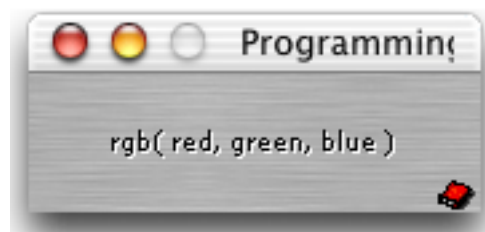
To copy the template into the topmost procedure window, hold down the **Control** key and click on the template panel. (If you are using a PC system you should right-click on the template panel.) The template will be pasted into the procedure at the current insertion point, and the procedure window will be brought to the front so that you can edit it further (for example, filling in the actual parameters).




You can also copy the template into the procedure window using the **Reference** menu. The **Copy to Procedure** command copies the template into the procedure and brings the procedure window forward (exactly like control-clicking on the template panel). The **Copy to Proc & Close** command does the same, but also closes the **Programming Reference** wizard.

### Minimizing the Programming Reference Wizard

In addition to its normal "wide-screen" view, the wizard can also be used in a minimized view. To minimize the wizard, either choose **Minimize Window** from the **Reference** menu or click on the  button in the template panel. As shown here the minimized wizard hides the search panel, topic list, and topic display panel.



Although you can't search for topics when the window is minimized, you can still use the **Reference**, **Topics**, **Statements** and **Function** menus. When you want to maximize the window again choose either the **Maximize Window** from the **Reference** menu or click on the  button.

## Data Flow

The purpose of almost any program is to organize and channel data. Since Panorama is a database, this is even more true for programs written in Panorama. This section discusses the techniques for storing and manipulating data within a procedure.

### Assignment Statements

An assignment statement computes a value (text or numeric) and stores that value somewhere. Unlike every other statement, an assignment statement has no specific keyword that identifies the statement. Assignment statements always have the format shown below:

```
<data storage location> = <formula>
```

The first part of the assignment statement is the **data storage location**. This is the final destination for the data that is being moved. In fact, sometimes the data storage location is simply called the **destination** of the assignment. The data storage location may be a variable, a field in the currently active record, or the clipboard.

The next part of the assignment statement is the equals symbol. This identifies this statement as an assignment statement.

After the equals symbol is the formula. The formula produces the data that will be stored in the data storage location. The formula may simply take a variable or field and pass it along, or it may process, calculate or filter the data before it passes it along to be stored in the data storage location.

Here's a simple assignment statement that takes the contents of B and moves it into A. After this statement is finished both A and B will contain the same value.

```
A=B
```

More complicated assignment statements may combine multiple fields or variables, and they may process the data in some way. An assignment statement may also take a constant value and store it. Here are some examples:

```
A=B*C
```

```
Name=upper(myName)
```

```
City="San Francisco"
```

In each case, the process is the same. First Panorama calculates the formula to produce a data value. Then it stores the data value in a data storage location.



## Variables

A variable is a place in the computer where an item of data can be stored, kind of like a storage bin for a value. Variables may be created by procedures or by SuperObjects. Most procedures will use one or more variables to hold and transfer data as the program runs. Use a variable whenever you need to store a single data item so that you can use it later. Unlike a field, the value variable doesn't change as you move from record to record, or, in the case of a global variable, even when you move from database to database.

### Creating a Variable

Panorama has five different statements for creating variables within a procedure. The most common one is the `local` statement (see “`LOCAL`” on page 5489), which generates temporary variables that only last until the procedure is finished. This statement should be followed by a list of the variables to be created, with each name separated from the next by a comma. This example creates four variables. Because these variables were created with the `local` statement they are called **local variables**.

```
local alpha,gamma,delta,sigma
```

Creating a variable is kind of like surveying a lot on empty land. Once the land is surveyed you know where it is, but the land is still empty until something is built on it. A new variable is like an empty plot of land that has just been surveyed — it has an address (the name) but it doesn't have any data yet.

By the way, Panorama allows any sequence of characters to be used as a variable name. However, if the variable name contains any punctuation (including spaces) it must be surrounded by the chevron characters « and ». (On the Macintosh press **Option-\`<`** to create the « chevron character and **Shift-Option-\`>`** to create the » chevron character. On Windows systems press **Alt-0171** to create the « chevron character and **Alt-0187** to create the » chevron character.) Here are some examples of typical variable names:

```
x
```

```
birthDay
```

```
Counter
```

```
<Tax Rate>
```

```
<PrimeRate%>
```

A variable name must be spelled exactly the same way every time, including upper and lower case. The variable name `birthDay` is not the same as `Birthday` or `birthday`. In fact, you could create three different variables using these three different names (although this is not recommended because it would be very confusing).

By the way, it's always ok to use chevrons around a variable name, even if the name doesn't have any punctuation. «`Counter`» is exactly the same as `Counter`, and they can be used interchangeably. So if you have any doubts about whether or not chevrons are necessary, go ahead and use them. No harm, no foul.

Note: Some programming languages require you to create all variables first, before any other statements. Panorama isn't that picky. You can create new variables anywhere in the program. Here is a procedure that creates four variables, does some work, then creates two more variables.

```
local alpha,gamma,delta,sigma
  alpha=4
  gamma="blue"
  delta=alpha*3
  sigma=delta/alpha
local epsilon,omega
  epsilon=0.01
  omega="z"
```

By the way, the indentation is not necessary, it's simply to make the local statements easier to see.



### Assigning a Value to a Variable

Once a variable has been created you can assign a value to it. This is done by putting the variable on the left side of an assignment statement (see “[Assignment Statements](#)” on page 596) like this.

```
alpha=4  
  
gamma="blue"  
  
delta=alpha*3  
  
sigma=delta/alpha
```

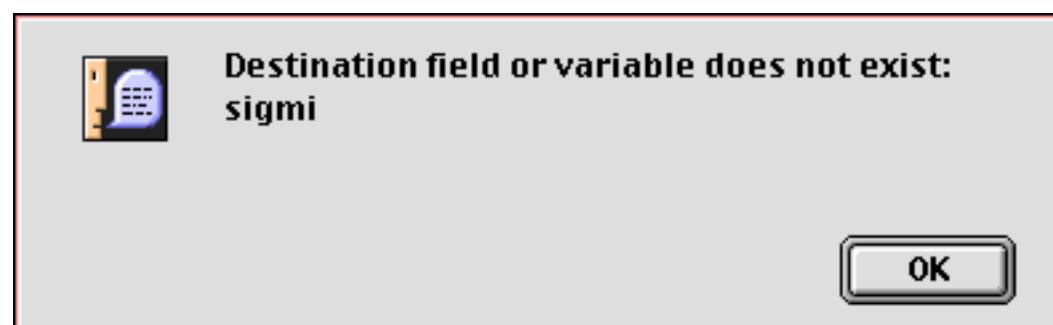
A new value can be assigned to a variable at any time. You can assign a value to a variable once or over and over again a million times. The new value does not have to have anything to do with the old value — you can store a number in a variable that originally held text or vice versa — Panorama doesn’t care. This line of text assigns the number **400** to the **gamma** variable, which originally had a text value stored in it.

```
gamma=400
```

Some programming languages allow you to assign a value to a variable without having to create the variable first. Panorama’s programming language does not allow this. For example, this procedure creates four variables, then attempts to store a value into a fifth variable, **sigmi**.

```
local alpha,gamma,delta,sigma  
sigmi=200
```

When you run this procedure Panorama will complain. Picky picky picky!



In this case the problem is probably a typo, and the variable name in the assignment needs to be corrected. If this really is a separate variable you must create it first.

### Using a Variable in a Formula

Once a variable has been assigned a value you can use it in a formula. A variable may be used anywhere a field or constant may be used. Here are some typical examples.

```
alpha*sigma  
  
gamma+" action"  
  
4*alpha/(delta+20)
```

See “[Formula Grammar](#)” on page 1178 to learn more about using variables in a formula.

### The Birth and Death of a Local Variable

When a lot is surveyed, that lot usually exists more or less forever (barring wars or natural disasters). A local variable, however, is not nearly that permanent. In fact, local variables only exist until the end of the procedure that they are created in. When the procedure finishes, Panorama checks to see if it created any local variables. If it did, the values in these variables are dumped out and all record of the variables are destroyed, just as if they had never existed in the first place. It's kind of as if you went down to the county recorder's office and burned all the survey records for a tract of land.

Why are the local variables destroyed? Two reasons. First, they take up memory that is no longer needed and can be used for other things. Secondly, this allows different procedures to use the same variable names without having to worry about conflicting with each other. Suppose procedure A and procedure B both have a local variable named **gamma**. Since the variable is destroyed when each procedure finishes, neither procedure needs to worry about the other. Each happily creates and uses its own copy of **gamma**, which is then destroyed before it can interfere with any other procedure.

### Long Life Variables

Sometimes, of course, you won't want a variable to be destroyed when the procedure is finished. Panorama actually has five different kinds of variables, each with different life cycles (local, window, fileglobal, global and permanent). You've already learned about local variables. The next most common type of variable is a **fileglobal variable**, which I'm sure you'll be surprised to learn is created with the **fileglobal** statement. Just as with the local statement, the fileglobal statement is followed by a list of variables to create. This statement creates two variables.

```
fileglobal Speed,Direction
```

Unlike a local variable, a fileglobal variable isn't destroyed when the procedure is finished. It hangs around and can be used over and over again. However, fileglobal variables don't last forever. When the database is closed, the values in these variables are dumped and the variables themselves are destroyed.

As you might guess, a **permanent variable** has a very long life. However, the life of a permanent variable is not continuous but interrupted. When the database is closed any permanent variables associated with that database are destroyed. However, before the variables are destroyed the values are stored in the database itself. When the database is re-opened later Panorama automatically re-creates the permanent variables again. (Note: You must save the database. Just as with data in database fields, Panorama only saves permanent variables when the database is saved.)

Another type of long life variable is a **global variable**. A global variable is not destroyed even when the database that created it is closed. It's almost immortal. However, when you **Quit** from Panorama, that's the end of the road for global variables. They are not re-created automatically the next time Panorama opens.

### Variable Accessibility

Just because a variable exists doesn't mean you can access it. Many types of variables are "attached" to a file or a window and are only available when that file or window is active. When the file or window isn't active, these variables are "dormant." They still exist (and take up memory), but you cannot access or modify them until the file or window they are attached to becomes active again.

**Fileglobal** and **permanent** variables are attached to the file that was active when they were created. When this file is open and on top, the variables are accessible and can be used in a formula or modified with an assignment statement. When some other file is on top these variables are dormant and cannot be used.

The beauty of this system is that it allows you to create variables without worrying about conflicting with other databases. Consider the two fileglobal variables created in the previous section, **Speed** and **Direction**. What if some other open database also has variables with these names? As long as both databases use fileglobal variables instead of global variables (see below) they'll both be all right. Each will have their own separate **Speed** and **Direction** variables. When database A is active its variables will also be active while B's are dormant. When database B is active A's variables become dormant. Essentially Panorama will keep two completely separate sets of **Speed** and **Direction** variables, each with their own values.

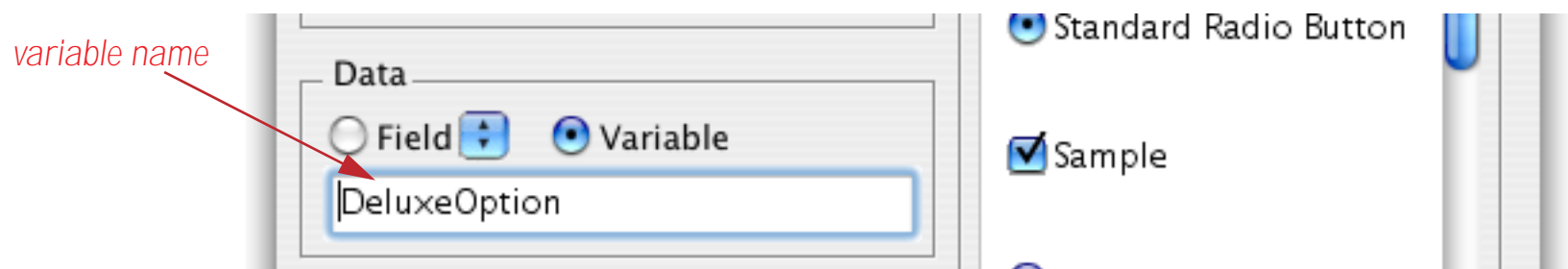
**Windowglobal** variables are attached to the window that was active when they were created. You can have multiple cloned windows that use the same variable name but actually each has its own separate variable that is only active when the window is on top (see “[Window Clones](#)” on page 1607).

**Global** variables are always active, no matter what file or window is on top. This is great if you need to have data that is accessible anywhere at any time. But be careful! If two different databases use the same global variable they had better be co-operating with each other! Remember that you may open databases that were created by other people. Who knows what global variable names they will use? If you do need to use global variables we recommend that you use very long descriptive names like **ProTechSpeed** or **AcmeSalesTaxRate**. Names like this are more likely to be unique and not conflict with anyone else's global variable names.

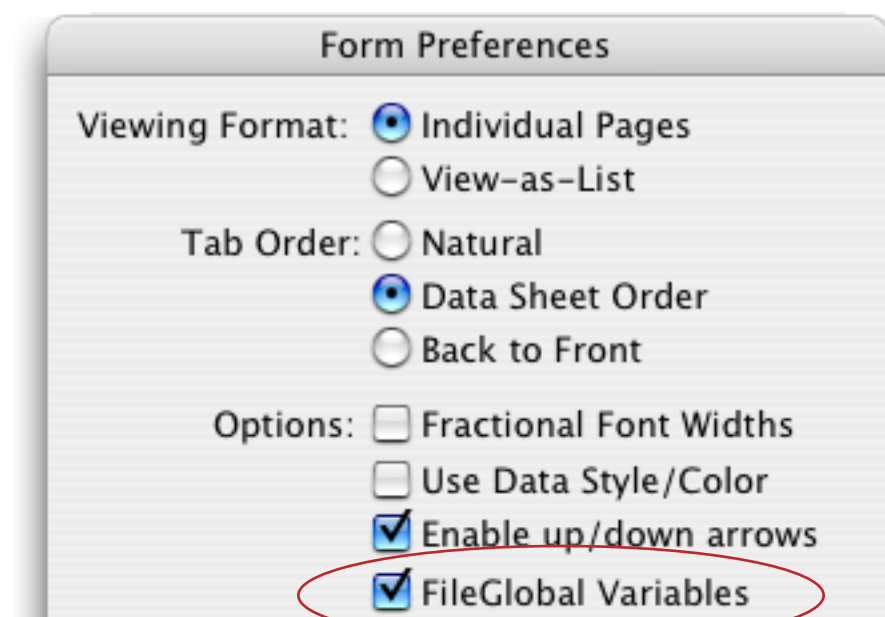
### Creating Variables with a SuperObject

Variables are usually created in a procedure (see “[Creating a Variable](#)” on page 597). However, several different types of SuperObjects have the option of linking to a variable or a field, and these objects will automatically create the variable if it does not exist. When a variable is created by a SuperObject it is always a global or fileglobal variable and is initialized to empty text. SuperObjects that can create variables include the Text Editor, Word Processor, Data Button, Pop-up Menu, List, Sticky Button and Scroll Bar.

For example, suppose you are working on a form and create a checkbox using the **Data Button** tool (see “[Data Buttons](#)” on page 815). You select the **Variable** option and type in the variable name **DeluxeOption**, as shown in this illustration.



When the **OK** button is pressed, Panorama checks to see if you have already created a variable named **DeluxeOption**. This may be a global variable, a fileglobal or permanent variable (in this database) or a windowglobal variable (in this window). If the variable has already been created, Panorama will simply use it. But if there is no such variable, Panorama will create it as a global or fileglobal variable. The default is a global variable unless the **FileGlobal Variables** option is set in the **Form Preferences** dialog (Setup menu). Except for how it was created, this variable is just like any other variable and can be used freely in procedures and formulas.



## Control Flow

As it runs a procedure, Panorama usually starts with the first statement in a procedure and works its way down. However, Panorama is not limited to this kind of linear approach. The procedure can be designed to make comparisons or decisions and take different steps depending on the result. For example, the procedure might decide to skip one or more steps, or it might decide to repeat a sequence of steps more than once. The procedure may even decide to trigger another procedure to help it complete its job. Programmers call this decision making process control flow, because it controls and possibly alters the flow of statements dynamically as the program is running.

The ability to change the flow of steps “on-the-fly” is the key to programming. A simple example may help make the utility and power of this concept more clear. Suppose you want to create a procedure that adds fifty new records to the end of the database. You could simply create a procedure with the `AddRecord` statement repeated fifty times. Of course this is inconvenient, and what if you wanted to create 2500 new records? Instead of repeating the `AddRecord` statement over and over, you can write a program that repeats a single `AddRecord` statement over and over until the proper number of records have been added.

```
loop
  addrecord
until 50
```

The program is much shorter than if you had literally repeated the `AddRecord` statement 50 times, and can be changed easily if you want to add a different number of new records.

Now suppose you want to change this program so that as it adds new records, it alternately puts the word `Black` or `Gold` into the `Color` field of each record. This requires the program to make a decision for each record—is this record even or odd? There are several ways this could be programmed in Panorama, The example below shows just one of them. In this example you’ll notice that all seven steps between loop and until are being repeated 50 times.

```
local nextColor
nextColor="Black"
loop
  addrecord
  Color=nextColor
  if nextColor="Black"
    nextColor="Gold"
  else
    nextColor="Black"
  endif
until 50
```

Although this program is quite simple, it illustrates the basic elements of control flow.

### True/False Formulas

In Panorama as in most programming languages, control flow decisions are made on the basis of formulas that are either true or false. The most basic true/false formula compares two values to see if they are equal.

```
PaymentMethod="C.O.D."
```

This formula will compare the value in the field `PaymentMethod` with `C.O.D.` The result will be true if `PaymentMethod` is `C.O.D.`, and false if it contains anything else (for example `Check`, `Cash`, `Visa`, etc.). To learn more about true/false formulas and how to create them see “[True/False Formulas](#)” on page 1260.

### Equals Comparison vs. Assignment

If you have been paying attention you undoubtedly noticed that the formula in the previous section looks exactly like an assignment. Why doesn't this formula

```
PaymentMethod="C.O.D."
```

assign the value **C.O.D.** to the field **PaymentMethod**? At first glance this may appear ambiguous...the same formula is used to compare two values and to assign a value. How do we know when we are assigning and when we are comparing? The answer lies in the context in which the formula is found.

In a procedure, an assignment is always by itself, not part of a larger statement. A true-false formula is always part of another statement, for example **if**, **case**, **until**, **while**, **stoploopif**, **repeatloopif**, **find**, **select**. Here's an example that shows two formulas that look almost the same, but one is a true-false formula and one is an assignment.

```
if PaymentMethod="C.O.D."
    ShippingMethod="UPS"
endif
```

The first formula, **PaymentMethod="C.O.D."**, is part of the **if** statement. Because it is part of the **if** statement this formula means: Is the field (or variable) **PaymentMethod** equal to **C.O.D.** (true/false)?

The second formula, **ShippingMethod="UPS"**, is not part of any statement, but stands alone, so this is an assignment. The statement means: Take the value **UPS** and copy it into the field or variable named **ShippingMethod**.

If an assignment has more than one equals sign, the first equals sign is for the assignment and the rest are for comparisons. The example assignment below compares B and C. If they are equal (true) the value -1 will be copied into A. If they are not equal (false) the value 0 will be copied into A.

```
A=B=C
```

In other words, **A** becomes the result of the comparison between **B=C**.

### True/False Values

For purposes of calculation, Panorama treats true and false as numbers: true is -1 and false is zero. Like any other number, you can store a true/false value in a field or variable and then use it later. The example below calculates whether a person is a teenager, then uses that information later.

```
local Teenager
Teenager=Age≥13 and Age<20
...
if Teenager
    Price=4.50
else
    Price=6.00
endif
```

Notice that the **if** statement doesn't need to compare, it simply uses the result of the comparison that was calculated earlier. In fact, the **if** statement (and all other statements that use true/false logic) can use any formula that produces a numeric integer result. The value 0 will be regarded as false, and any non-zero value will be regarded as true. The example below will be true if the length of the name is non-zero.

```
if length(Name)
    yesno "Is this a home address?"
...
endif
```

The first line of this example could also have been written **if length(Name)<>0**. The result is the same either way.



## IF Statements

The basic building block for making decisions in a Panorama program is the **if statement**. The **if** statement will skip over the next few statements (up to the next **endif** statement) if the true/false formula is false. Here's a simple example.

```
if City=""
  City="Pismo Beach"
  State="CA"
endif
message City+", "+State
```

Depending on what's in the **City** field, this procedure can work one of two ways. If the **City** field is empty, the true/false formula **City=""** will be true, so the procedure will perform the assignments **City="Pismo Beach"** and **State="CA"**. But if the **City** field is not empty, the true/false formula will be false, and Panorama will skip past the **endif** to the message statement.

In this example there are two statements between the **if** and **endif** statements. These two statements will be skipped if the formula is false. However, there is no limit to the number of statements that may be between the **if** and **endif**. Just make sure that there is always a matching **endif** for every **if**. Although it is not required, indenting the statements between the **if** and the **endif** usually makes the procedure easier to read and understand.

## ELSE Statements

The **else statement** turns the **if** statement into a two way operation: if true, do this, otherwise, do that. You could do this with two **if** statements in a row, but the **else** is simpler.

To use the **else** statement, place it between the **if** and **endif** statements. If the true/false formula is true, Panorama will perform the statements from the **if** up to the **else** and skip the statements from the **else** up to the **endif**. If the true/false formula is false, Panorama will skip the statements from the **if** up to the **else** and perform the statements from the **else** up to the **endif**.

The example below calculates sales tax and shipping for both in-state and out-of-state purchases.

```
if State="CA"
  SalesTax=0.08
  Shipping=2.50
else
  SalesTax=0
  Shipping=5.00
endif
```

If the state is **California**, the sales tax is 8% and shipping is \$2.50. But if the purchase is from any other state, the sales tax is zero and shipping is \$5.00. In this example more or less the same statements are used in both halves of the **if/else**, but this is not necessary. The two sections could be completely different.

## Nested if Statements

Panorama is not limited to one **if** at a time. Panorama can make a decision, execute some more statements, and then make a subdecision. Since the inner **if endif** pair is completely surrounded by the outer pair, this is called nesting.

```
local CardLength
if PaymentMethod="Credit Card"
  CardLength=length(CardNumber)
  if CardLength<13 or CardLength>16
    message "Sorry, invalid credit card number."
  endif
endif
```



If the `PaymentMethod` is not `Credit Card`, the procedure will skip all the following statements and do nothing. But if the `PaymentMethod` is `Credit Card`, the procedure will continue and calculate the `CardLength` variable. The second `if` statement checks the card length. The `message` statement will only be performed if both `if` statements are true.

### Error Handling with `if error`

There are literally hundreds of different errors that can occur while a procedure is running. Of course you'll want to eliminate all of the errors in the procedure itself, but many errors are the result of circumstances beyond the programmers control. A file can fail to open because it was placed into the wrong folder, the user can enter the wrong data type into a formula, the list is endless. When such an error occurs, Panorama's normal response is to display an error message and stop the procedure immediately. (In addition, if the procedure window is open, Panorama will attempt to highlight the location of the error.)

If you want to create a database that operates professionally, simply stopping the procedure half finished if there is an error may not be acceptable. Instead, you may want your procedure itself to trap the error and try to correct it, if possible. At a minimum, you may be able to display an error message that is more relevant to an untrained operator than Panorama's general purpose error messages.

To trap errors, use the `if error` statement (two words - there **must** be a space). This statement must be placed immediately after the statement that you are worried might cause an error. For example, suppose you have a procedure that appends a file with the `openfile` statement. If the file is missing or has been moved an error will occur. This example checks for that error, and if the error occurs, asks the user to enter a new file name. The procedure will keep trying until the file is opened successfully or the user gives up and enters an empty name.

```
local txFileName
txFileName="New Transactions"
loop
  openfile "+" + txFileName
  if error
    gettext "Enter the file name", txFileName
    reloopif txFileName=""
  endif
while 1≠1
  if txFileName="" stop endif
/* further processing of the new transactions, below */
...
```

### CASE Statements

If a program needs to select one (and only one) option out of many, the `case statement` is the way to go. Like the `if` statement, the `case` statement uses a true-false formula to decide whether or not to perform the following statements. But unlike the `if` statement, which is used alone, the `case` statement is always used in groups. Panorama checks the true-false formula for each `case` statement. If it is false, it skips to the next `case` statement. If it is true, it performs the statements until the next `case` statement. Then it skips past all the rest of the `case` statement to the `endcase` statement.

After all the `case` statements, you may optionally add a `defaultcase` statement. This will pick up any left-overs that weren't included in any of the other cases.

The example below shows how the case statement can be used to divide people up into five age groups.

```
case Age<5
  AgeGroup="Pre-School"
case Age<13
  AgeGroup="Youth"
case Age<20
  AgeGroup="Teen"
case Age≥65
  AgeGroup="Senior"
defaultcase
  AgeGroup="Adult"
endcase
```

This example has included one statement for each **case** statement, but there is no limit to the number of statements that may be included in each section. However, there is a maximum limit of 75 **case** statements per **endcase** statement.

## LOOP Statements

A **loop** allows Panorama to repeat a sequence of statements over and over again. The loop can be repeated a fixed number of times, or until a special condition is fulfilled.

All loops begin with the **loop** statement, and end with either **until** or **while**. The statements in between these two statements are said to be “inside the loop.” These are the statements that will be repeated over and over again. Although it is not required, your procedures will usually be easier to read and understand if the statements inside the loop are indented.

To repeat the statements inside the **loop** a fixed number of times, use the **until** statement with a number after it. This number may be a fixed number, or a variable or formula that calculates a number. For example, this procedure will add a dozen shiny new records to the database:

```
loop
  addrecord
until 12
```

To repeat the statements inside a loop until a specific condition is met, put a true/false formula after the **until** statement. Like the previous example, this example adds new records to the database. In this case, however, the number of new records is determined by asking the user (with the **gettext** statement).

```
local NewCount
NewCount="1"
gettext "How many new records?",NewCount
NewCount=val(NewCount)
loop
  NewCount=NewCount-1
  AddRecord
until NewCount=0
```

The **while** statement is the exact opposite of the **until** statement; it repeats the loop as long as the formula remains true. Here is the previous example rewritten to use the **while** statement.

```
local NewCount
NewCount="1"
gettext "How many new records?",NewCount
NewCount=val(NewCount)
loop
  NewCount=NewCount-1
  AddRecord
while NewCount>0
```

These two examples are exactly the same except for the last line.

Note: The **while** statement can also be followed by the special word **forever**, which tells Panorama to repeat the loop forever. Usually this is used with a **stoploopif** statement to break the loop, otherwise your procedure won't ever stop!

### Stopping a Loop in the Middle

The **stoploopif** statement allows Panorama to break out of the loop in the middle (or even at the top), instead of at the bottom. Panorama will break out of the loop if the true-false formula is true.

The example below finds every record where the field **PrintDuplicate** contains **Yes**. Each of these records is duplicated. But what if there were no such record? The **stoploopif** statement will stop the loop before it ever begins. The **stoploopif** statement also checks each time the loop is repeated to see if the next statement has found another record to duplicate, or if the loop is done.

```
toprecord
find PrintDuplicate="Yes"
loop
  stoploopif info("notfound")
  copyrecord
  pasterecord
  downrecord
  next
while forever
```

Notice that this sample uses **while forever**. This means that the **while** statement will never stop the loop.

### Restarting a Loop in the Middle

The **repeatloopif** statement tells Panorama to restart the loop from the top. The example procedure below tries to extract a phone number from the clipboard.

```
local X,theChar,aPhone
X=1
aPhone=""
loop
  theChar=clipboard()[X;1]
  X=X+1
  stoploopif theChar=""
  repeatloopif theChar≠ "(" and aPhone=""
  aPhone=aPhone+theChar
until aPhone match "(???) ???-????"
```

Each time the loop goes around it copies the next character from the clipboard into the variable **theChar**. If there are no more characters, the loop stops. Each character is checked to see if it is a left parenthesis. Until a ( is found, the **repeatloopif** statement stops the loop short, repeating only the top portion of the loop. Once the ( is found the loop starts collecting the following data into **aPhone**. The loop finally stops when the entire phone number is collected or the clipboard runs out of data.

### Subroutines

Sometimes you may need to use the exact same series of steps in several places in your program. Wouldn't it be nice if Panorama had a special statement that performed this series of steps for you, so you wouldn't have to type those same steps over and over again? Your programs would be smaller, easier to create, and easier to modify. You can't create your own statements, but a subroutine is the next best thing.

A subroutine is used by "calling" it. It's sort of like calling someone to dinner. When a subroutine is called, Panorama temporarily stops performing the steps in the current procedure. It marks its place in the current procedure, and then starts performing the steps in the subroutine. When it has completed all the steps in the subroutine Panorama goes back to the original procedure and starts off right where it left off. The net effect is as if the statements from the subroutine were copied into the middle of the original procedure.

When the same steps are used in different places, a subroutine has many advantages. First of all, using a subroutine makes the database smaller, because these statements appear only once. An even bigger advantage is that if the statements in the subroutine ever need to be changed, they will only have to be changed in one place, instead of over and over again.

It's possible for the main procedure and the subroutine to pass values back and forth between them. These are called parameters. Parameters allow very general subroutines to be written that can handle a wide variety of situations.

### CALL Statement

The call statement allows any procedure in the current database to be called as a subroutine. The basic format is simple:

```
call <procedure name>
```

For example, suppose you have created a procedure called [DuplicateRecord](#). The procedure looks like this:



We can use this procedure in another procedure by calling it.

```
toprecord
find PrintDuplicate="Yes"
loop
  stoploopif info("notfound")
  call DuplicateRecord
  next
while forever
```

Each time Panorama repeats the loop it will call the [DuplicateRecord](#) procedure. The three steps in that procedure will be performed, then it will return to the loop and perform the next statement. As far as Panorama is concerned, this is exactly the same as if you had written the procedure this way.

```
toprecord
find PrintDuplicate="Yes"
loop
  stoploopif info("notfound")
  copyrecord
  pasterecord
  downrecord
  next
while forever
```

Although there is no difference as far as running the procedure is concerned, there is a big difference for writing procedures. Suppose you have many procedures that need to duplicate a record. If you create a subroutine to duplicate the record, you can save two lines of typing each time you need to duplicate a record. Most subroutines have more than three lines, so the savings are even more substantial.

An even more important advantage is that using subroutines allows you to “modularize” your code. You’ll probably never have to modify the simple code needed to duplicate a record, but more complicated subroutines often need to be adjusted from time to time. If you had simply typed the statements of the subroutine into each location where they were needed (as shown in orange above) then making the adjustment would be very time consuming because you would have to locate and modify each copy of the statements. By collecting these statements together in a subroutine you can make any adjustments necessary to the code in a single location. Every procedure that calls the subroutine will automatically get the benefit of the adjustments.

### Calling Procedures With Unusual Names

If a procedure has a space or other punctuation inside the procedure name, you must enclose the procedure name in quotes, like this:

```
call "Calculate P/E Ratio"
```

Quotes are not necessary for a procedure name that contains a period, even if the period is the first character of the name.

```
call .DialNumber
```

It is even possible to calculate the procedure name with a formula. The formula must be surrounded with parentheses. The example below assumes that there is a dialing procedure for several different fields in the database, **Dial Name**, **Dial Company**, etc. If there is such a procedure for the current field, this procedure will call it.

```
call ("Dial "+info("fieldname"))
if error
  message "Sorry, can't dial the "+info("fieldname")
endif
```

If there is no such procedure, the error message will appear.

### Passing Values to and from a Subroutine (Parameters) and Results

A subroutine can get values (parameters) from the program that calls it, and send values back to that program (results). To learn more about parameters and results see Chapter 24 of the **Panorama Handbook**.

### Terminating a Subroutine in the Middle

The **rtn** statement (short for **return**) allows a subroutine to stop short in the middle and return to the original procedure. Usually when a subroutine is called, all the statements in the subroutine are performed from top to bottom. But if the **rtn** statement is encountered, the subroutine stops and immediately goes back to the original procedure. The **rtn** statement is almost always used in combination with the **if** or **case** statement.

The simple example below dials a local phone number, which is passed to the subroutine in parameter 1. If no phone number is passed, or if a long distance phone number is passed, the subroutine returns without doing anything.

```
local DialNumber
DialNumber=parameter(1)
if error
  rtn
endif
if DialNumber="" or length(DialNumber)>8
  rtn
endif
dial DialNumber
```

If the **rtn** statement is encountered in a procedure that has not been called as a subroutine (i.e. an original procedure) the procedure will simply stop.

### Mini Subroutines within a Procedure

Sometimes you may want to use a short subroutine, perhaps two or three lines. It just seems like too much hassle to create a separate procedure. For these situations, Panorama allows you to create a subroutine right inside the current procedure. This special subroutine within a procedure is called a **short subroutine**.

A short subroutine always begins with a **label**. A label is a unique series of letters and numbers that identifies a location within the procedure. The label may not contain any spaces or punctuation except for . and \$, and must always end with a colon. The colon is not actually part of the label, it simply identifies the series of letters and numbers as a label, as opposed to a field or variable. Here are some examples of labels:

```
diamond:
```

```
blue:
```

```
mailAction7:
```

```
Dispatch.Route:
```

A short subroutine ends with the end of the procedure, or with a **rtn** statement. A single procedure may contain many short subroutines, each starting with a label and ending with a **rtn** statement.

Short subroutines are called with the **shortcall** statement. This statement is always followed by the name of the short subroutine (the label). Don't include the colon here. You must type the label exactly as it appears at the top of the short subroutine (except for the colon), no quotes, and unlike a regular call statement the subroutine name cannot be calculated. The **shortcall** statement also does not allow parameter passing. Here are examples of how to call short subroutines.

```
shortcall diamond
```

```
shortcall blue
```

```
shortcall mailAction7
```

```
shortcall Dispatch.Route
```

The example below contains a short subroutine called **GroupTotal**. The short subroutine starts on line 7, with the label **GroupTotal:**. This short subroutine performs three steps and then returns to the main section of the program. The main section of the program calls the subroutine twice, then stops.

```
field City
shortcall GroupTotal
field State
shortcall GroupTotal
stop
```

```
GroupTotal:
  groupup
  field "Amount"
  total
  rtn
```



If the `stop` statement was not included, the program would continue down and perform the steps in the short subroutine a third time. In fact, we do this on purpose to produce a shorter version of this program.

```
field City
shortcall GroupTotal
field State

GroupTotal:
  groupup
  field "Amount"
  total
```

We've also removed the `rtm` statement at the end of the short subroutine. It's redundant in this case because the short subroutine ends at the end of the entire procedure. However, if there were additional short subroutines after this one, all but the last one would require a `rtm` statement at the end.

### Subroutines and Local Variables

Earlier in this chapter you learned that local variables are destroyed at the end of the procedure that created them (see "[The Birth and Death of a Local Variable](#)" on page 599). Local variables also become dormant when a subroutine is called (except for short subroutines, see "[Mini Subroutines within a Procedure](#)" on page 609). At the end of the subroutine the local variables come out of hibernation. Because of this, local variables created in one procedure cannot be accessed in another procedure. Local variables are always completely separate.

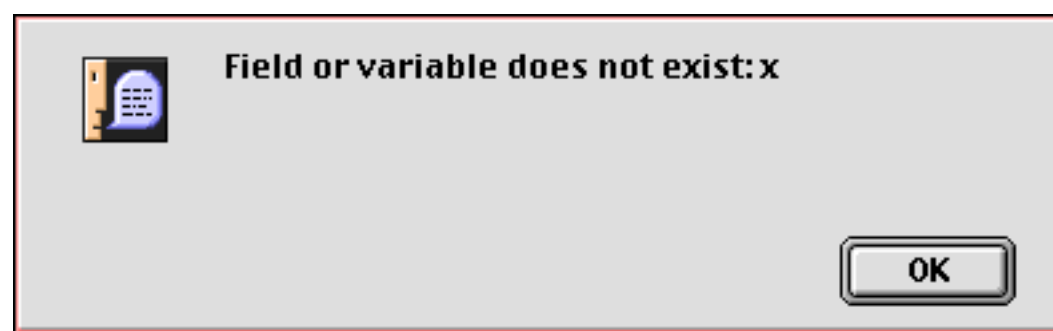
To illustrate this, consider this procedure which creates a local variable named `x`.

```
local x
x=2
call test
message x
```

The procedure `test` contains only one line which displays the `x` variable.

```
message x
```

However, this subroutine does not work! The `x` variable is now dormant and is not accessible to the `test` procedure. Instead of displaying the value 2, this error message appears.



As a matter of fact, Panorama allows the `test` subroutine to have its own separate `x` variable, like this.

```
local x
x=9
message x
```

Now when the original procedure (see above) is run two alerts appear. The first displays the value 9 (the value of `x` set in the `test` subroutine). The second displays the value 2 (the original value of `x` which was dormant but re-appeared when the `test` subroutine was finished).

## Other Control Flow Statements

There are a few other control flow statements that don't fit into any neat categories. These statements are described in the following sections.

### Jumping to an Another Location in the Program

The `goto` statement tells Panorama to jump immediately to another spot in the program (see "[GOTO](#)" on page 5326). The format of this command is simple:

```
goto <label>
```

A **label** is a unique series of letters and numbers that identifies a location within the procedure. The label may not contain any spaces or punctuation except for `.` and `$`, and must always end with a colon. The colon is not actually part of the label, it simply identifies the series of letters and numbers as a label, as opposed to a field or variable. Here are some examples of labels:

```
diamond:
```

```
tryAgain:
```

```
accessRoute3:
```

```
Start.Over:
```

The example subroutine below asks the user to enter an angle. If the angle is not between `0` and `360`, Panorama will jump back to the label `TryAgain`.

```
fileglobal GoAngle
GoAngle="0"

TryAgain:
  gettext "Enter direction (0-360)",GoAngle
  if GoAngle>360 or GoAngle<0
    goto TryAgain
  endif
  setparameter 1,val(GoAngle)
```

You may wonder why the `goto` statement is buried here at the end of the chapter. It's simple: we don't want you to use it! At least, not much. For years professional programmers have known that too many `goto`'s quickly create "spaghetti" code that is usually confusing. In fact, it has been mathematically proven that any program can be written with `if` and `loop` statements, without any `goto`'s at all. Here's how our example could be rewritten without a `goto`.

```
global GoAngle
GoAngle="0"
loop
  gettext "Enter direction (0-360)",GoAngle
  while GoAngle>360 or GoAngle<0
    setparameter 1,val(GoAngle)
```

As you can see, this example is actually simpler without the `goto` statement. Occasionally the `goto` statement does make it easier to write a program. Frankly we couldn't come up with an example of this, but if you do, feel free to use the `goto` statement. That's what it's there for.

### Stopping the Program

The **stop** statement tells Panorama to stop the procedure immediately (see “[STOP](#)” on page 5796). If the current procedure is a subroutine, the original procedure is also stopped. If you want the current subroutine to stop but the original procedure to continue, use the **rtn** statement (see “[Terminating a Subroutine in the Middle](#)” on page 608). The **rtn** statement also acts to stop the procedure if it is not being used as a subroutine.

### Doing Nothing for a While

The **nop** statement (short for no operation) tells Panorama to do absolutely nothing! You can use the **nop** statement as a placeholder, or to delay for a short time. Here’s an example of a procedure that will delay for a short time (probably less than a second).

```
loop
  nop
until 20
```

The exact amount of time delay depends on the speed of your computer. Here's an example that will delay exactly 10 seconds.

```
local startTime
startTime=now()
loop
  nop
until now(>startTime+10
```

Another use for the **nop** statement is to fool Panorama into not displaying a warning dialog. When used as the last statement in a procedure, or just before a **stop** statement, statements like **quit** and **close** will ask the user if they want to save changes. By adding a **nop** statement you can prevent this dialog from appearing.

```
if info("trigger") contains "Close w/o Save"
  close
  nop
  stop
endif
```

## Program Formatting

The way a program is formatted can make a big difference in how understandable it is. Panorama is very flexible in letting you format a program; you can have multiple statements on a single line, or split a single statement over multiple lines. Statements can be flush on the left or indented; it's all up to you.

For example, here's a sample procedure from earlier in this chapter with one line per statement.

```
select Debit>0
field Category
groupup
field Debit
total
outlinelevel 1
```

Here's the same procedure squished onto a single line. Panorama will understand this just fine, although you and I might have a more difficult time.

```
select Debit>0 field Category groupup field Debit total outlinelevel 1
```

You can even split individual statements across multiple lines, as long as you don't split a single word or constant in the middle. Here's another version of this same program.

```
select
Debit>0
field
Category
groupup
field
Debit
total
outlinelevel
1
```

Anyplace you can have a single blank or carriage return you can have more than one. Here's one final example.

```
select      Debit>0
field      Category
groupup
field      Debit
total
outlinelevel 1
```

In general, we recommend using one statement per line for readability. We also recommend indenting the statements between **if** and **endif**, **case** and **endcase**, and between **loop** and **until** or **while** (as seen in most of the examples throughout this manual). If you have multiple levels of nested **if** statements, each level should be indented further. This makes it easy to see which statements are associated with each **if/endif** pair.

Here's a program example with no indenting:

```
local CardLength
if PaymentMethod="Credit Card"
CardLength=length(CardNumber)
if CardLength<13 or CardLength>16
message "Sorry, invalid credit card number."
endif endif
```

Now the same procedure with the recommended indenting:

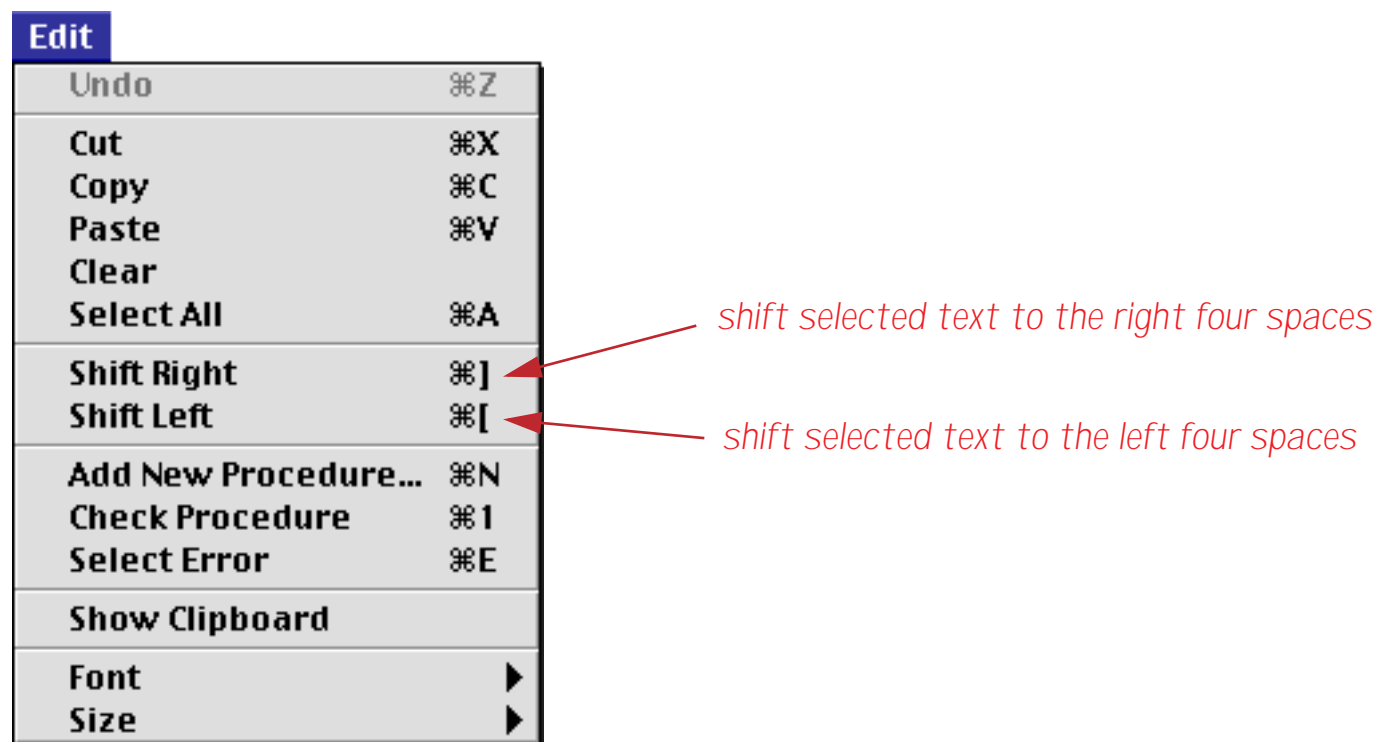
```

local CardLength
if PaymentMethod="Credit Card"
  CardLength=length(CardNumber)
  if CardLength<13 or CardLength>16
    message "Sorry, invalid credit card number."
  endif
endif
endif

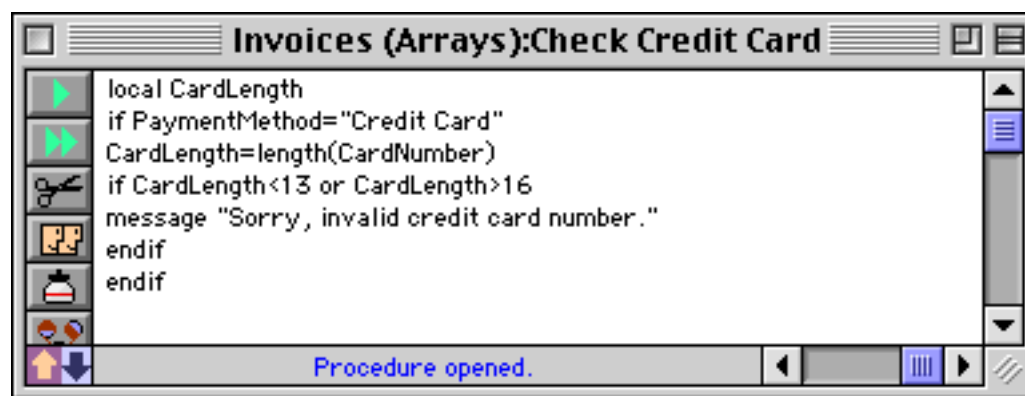
```

A lot easier to understand, isn't it?

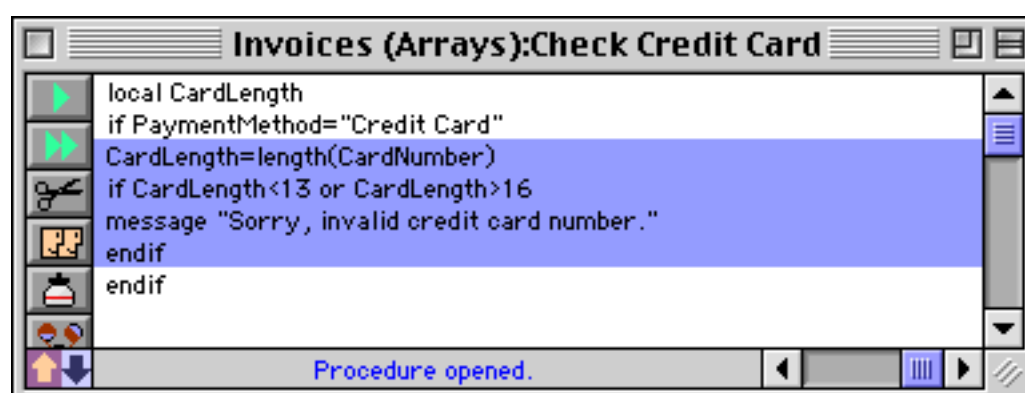
The **Edit** menu has two commands that can help you shift a section of text to the left or right.



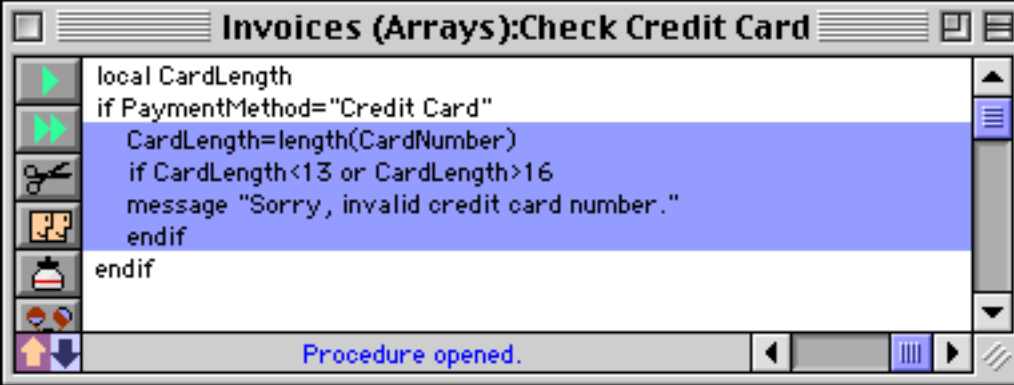
For example, suppose you start with this procedure, which is not indented at all (it's the same procedure listed earlier in this section).



To indent the text, start by selecting the text between the **if** and **endif** statements.



Now use the **Shift Right** command to indent the selected text.



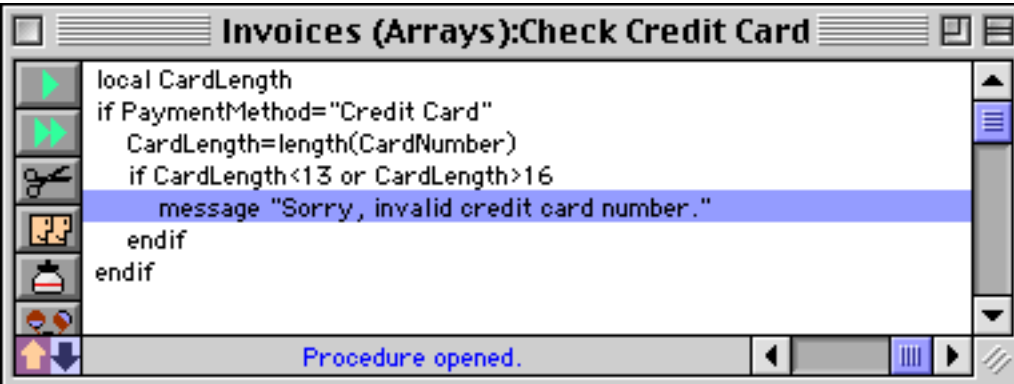
```

local CardLength
if PaymentMethod="Credit Card"
  CardLength=length(CardNumber)
  if CardLength<13 or CardLength>16
    message "Sorry, invalid credit card number."
  endif
endif

```

Procedure opened.

Repeat as necessary to indent any other text.



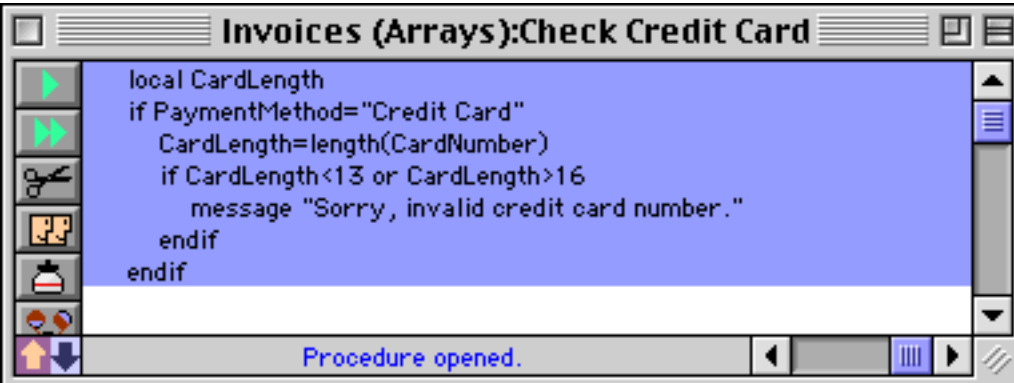
```

local CardLength
if PaymentMethod="Credit Card"
  CardLength=length(CardNumber)
  if CardLength<13 or CardLength>16
    message "Sorry, invalid credit card number."
  endif
endif

```

Procedure opened.

You can even use these commands to change the indentation of the entire procedure.



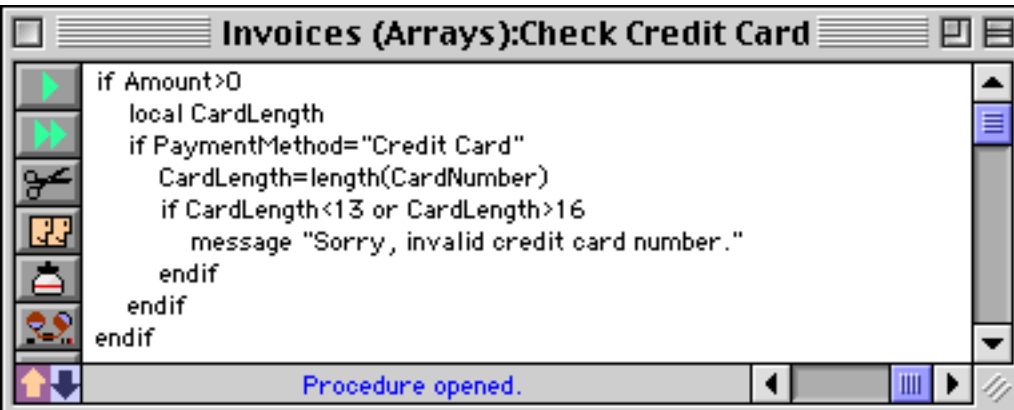
```

local CardLength
if PaymentMethod="Credit Card"
  CardLength=length(CardNumber)
  if CardLength<13 or CardLength>16
    message "Sorry, invalid credit card number."
  endif
endif

```

Procedure opened.

Now we can add another if statement around the entire procedure.



```

if Amount>0
  local CardLength
  if PaymentMethod="Credit Card"
    CardLength=length(CardNumber)
    if CardLength<13 or CardLength>16
      message "Sorry, invalid credit card number."
    endif
  endif
endif

```

Procedure opened.

Consistent indentation can go a long way towards making your programs more readable and bug free.



## Notes To Yourself

A **comment** is a note inside the program. Comments are very useful for documenting how a procedure works, what the variables are for, what the procedure parameters are, etc. Comments are totally optional, but you should use them to record anything you think you might forget about the operation of a procedure.

Panorama has three different comment styles: `/* ... */`, `//`, and `;`.

`/* ... */` comments begin with `/*` and end with `*/`. The advantage of this type of comment is that a single comment may be many lines long. You can also use this type of comment within a formula.

`//` comments begin with `//` and continue to the end of the line.

`;` comments begin with a semicolon and continue to the end of the line.

A comment can appear almost anywhere in a procedure. The only restriction on comments is that they cannot be inside the middle of a statement or formula; they must be between statements.

This example shows a procedure with lots of comments. If anyone comes back and takes a look at this procedure next year, they will have no problem telling what the procedure does and how it does it. To emphasize the comments they are shown in purple below. However, when actually editing a procedure the comments are black just like everything else.

```

/* Procedure: .Delay

This procedure delays for a fairly precise time.
The procedure has one parameter, the number of seconds to delay.

Example:
  call .Delay,12; delay for 12 seconds
*/
local startTime
startTime=now()// record the time we started
loop
  nop ; short delay
until now(>startTime+parameter(1)

```

## “Commenting Out” Statements

One handy use for `/* ... */` comments is to temporarily remove one or more statements from your program (usually for testing purposes). Simply put `/*` and `*/` around the statement or statements you want to remove, and those statements are effectively removed from your program without actually erasing them. (Programmers call this “commenting out” the statements, because they are temporarily “out” of the program.) To re-enable the statements simply remove the `/*` and `*/`.

## Suppressing Display of Text and Graphics

As a program executes the windows belonging to the database will often flicker or even redisplay over and over again as the statements are performed. Often this redisplay serves no purpose except to slow the program down and annoy you. To disable display while a sequence of steps is performed you must bracket the steps with the `noshow` and `endnoshow` statements, like this.

```
noshow
  statement
  statement
  statement
  ...
endnoshow
```

The `noshow` statement tells Panorama to suppress all display of text and graphics by the following statements. For example the `sort` and `formulafill` statements normally cause some or all of the window to be redisplayed. But if these statements follow a `noshow` statement the window will not redisplay. The `endnoshow` statement cancels the effect of the `noshow` statement and resumes normal display operation.

Note: The `noshow` statement suppresses all display that results from changes to the database. It does not, however, suppress display that is caused by changes in the configuration of database windows. For example if a procedure moves a window to the front with the `window` statement the newly visible section of the window will always be displayed, with or without a `noshow` statement. This is true for any statement that changes the window configuration: opening or closing a window, changing the size of a window, or changing the stacking order of the windows.

### Updating the Display After (or Within) a NoShow Block

The `noshow` statement is great for making procedures run faster without unnecessary window updating. There's just one problem though - since the window is not updated, it winds up being wrong! Consider the procedure below.

```
noshow
  field Date
  groupup by month
  field Category
  groupup
  field Amount
  total
  outlinelevel 2
endnoshow
```

Without the `noshow` statement this procedure will cause the window to update four times. But with the `noshow` statement the final result is not displayed! To fix this you must add one of the seven statements in the table below.

Statement	Parameters	Description
<code>showpage</code>	none	Displays the entire database.
<code>showline</code>	none	Displays the current record.
<code>showfields</code>	list of fields	Displays the specified fields in the current record
<code>showvariables</code>	list of variables	Displays the specified variables
<code>showcolumns</code>	list of fields	Displays the specified fields in all visible records
<code>showrecordcounter</code>	none	Displays the number of records
<code>showother</code>	field,option	Depends on option, see documentation below

Using the **showpage** statement we can fix the program listed earlier so that it displays the final result at the end of the procedure.

```
noshow
  field Date
  groupup by month
  field Category
  groupup
  field Amount
  total
  outlinelevel 2
  showpage
endnoshow
```

The seven display statements are described in the following sections.

### ShowPage

The **showpage** statement forces Panorama to redisplay all windows in the current database. Here is an example that performs several operations on the current database, but only updates the display once:

```
noshow
  field Date
  groupup by month
  field Category
  groupup
  field Amount
  total
  outlinelevel 2
  showpage
endnoshow
```

### ShowLine

The **showline** statement forces Panorama to redisplay the current record in all windows in the current database. The example below clears the current record in the database, but doesn't display anything until it is completely finished.

```
noshow
  field array(dbinf("fields",""),1,1) /* go to first field */
  loop
    clearcell
    right
  until stopped
  showline
endnoshow
```

Without the **noshow** statement you would be able to watch as Panorama cleared each cell in the line. With the **noshow** statement all of the cells will appear to disappear simultaneously.

### ShowFields field,field,...,field

The **showfields** statement forces Panorama to redisplay the specified fields in all windows in the current database. You may list as many fields as you want to display, with each field separated by a comma. (If you want to display all the fields it is easier to use the **showline** statement.) The example below modifies three fields but only displays the change made to the **Balance** field.

```
noshow
  Date=today()
  Time=now()
  Balance=Credit-Debit
  showfields Balance
endnoshow
```

**ShowColumns field,field,...,field**

The **showcolumns** statement forces Panorama to redisplay the specified fields in all windows in the current database. In a data sheet or view-as-list window the entire column is re-displayed, not just the current record. You may list as many fields as you want to display, with each field separated by a comma. (If you want to display all the fields it is easier to use the **showpage** statement.) The example below performs two calculations on the **Balance** field, but only redisplay the column a single time.

```
noshow
  field Balance
  Balance=Credit-Debit
  RunningTotal
  showcolumns Balance
endnoshow
```

**ShowVariables var,var,...,var**

The **showvariables** statement forces Panorama to redisplay the specified variables in all windows in the current database. You may list as many variables as you want to display, with each variables separated by a comma. The example below adds a new record to the database without changing the display, but does show the new record count.

```
noshow
  global myCount
  addrecord
  myCount=info("total")
  showvariables myCount
endnoshow
```

**Warning:** The **showvariables** statement is **always** required if you want to display changes to one or more variables, even if you are not using the **noshow** statement.

**ShowRecordCounter**

The **showrecordcounter** statement forces Panorama to redisplay the record count in all windows in the current database. The example below adds three new records to the database but only updates the display once.

```
noshow
  addrecord
  addrecord
  addrecord
  showpage
  showrecordcounter
endnoshow
```

### ShowOther field,code

The `showother` statement forces Panorama to redisplay all windows in the current database. You specify a code that tells Panorama what portion of the window to update. This is the code that Panorama uses internally, so if Panorama becomes capable of a new mode of redisplay it will automatically become available. Some codes allow you to specify a specific field to update, you can use the field name or use `All` to specify all fields. The available codes are listed in this table.

Code	Action
0	Display current cell (should use <code>showfields</code> instead)
1	Display entire page (should use <code>showpage</code> instead)
2	Cursor moved, update data sheet
3	Cursor moved, data sheet already updated
4	Update window after insertline (data sheet or view-as-list)
5	Move cursor up/down (for example after a search)
6	New line with cursor move
96	Display the tool palette
97	Display record count (use <code>showrecordcounter</code> instead)
98	Display data sheet field header (after changing field name)
99	Display after database redesign (insert field, etc.)

We recommend that you avoid this command if one of the other show commands will do the job for you.

### Checking NoShow Status

Sometimes a procedure may need to check whether the display is currently disabled with `noshow`. This can be done with the `info("noshow")` function. This function returns true if `noshow` is currently in effect, false if display is normal.

```




local nstate
nstate=info("noshow")
noshow
sortup
if nstate=false()
  showpage
  endnoshow
endif
rtn

```

The example above is a subroutine that sorts the database without updating the display. It checks to see if `noshow` was already on (perhaps the procedure that called this subroutine had already turned it on), and if so, leaves it on when it finishes. The calling procedure can then continue to perform other operations without updating the display. But if the calling procedure had not enabled `noshow` before calling this subroutine the subroutine turns the display back to normal before returning (with `endnoshow`).

### Disabling the Watch Cursor

As a procedure runs the cursor often flips from the arrow into a watch, or sometimes a pie chart. This helps let the user know that they may need to wait.

Arrow	Watch	Pie Chart
		

In some cases Panorama flips to the watch or pie chart cursor when it is not really necessary (especially on today's faster machines). If you want the mouse cursor to remain as an arrow while your procedure runs you can use the `nowatchcursor` statement (see "[NOWATCHCURSOR](#)" on page 5549). Here is a procedure that opens a database. This would normally cause the watch cursor to be displayed, but in this case the arrow remains active.

```
nowatchcursor  
openfile "Reference Data"  
watchcursor
```

The final statement re-enables the watch and pie chart mouse cursors. In this example the `watchcursor` statement isn't really necessary because Panorama always automatically re-enables these cursors at the end of any procedure.



## Debugging a Procedure

In the real world, programs often don't work correctly the first time. (Sometimes they don't work the second or the third time, either!) Panorama has a number of tools that you can use to help locate and correct the problem in a procedure. In addition to the methods described below see Chapter 24 of the **Panorama Handbook** for more advanced debugging techniques.

### The Panorama Interactive Debugger

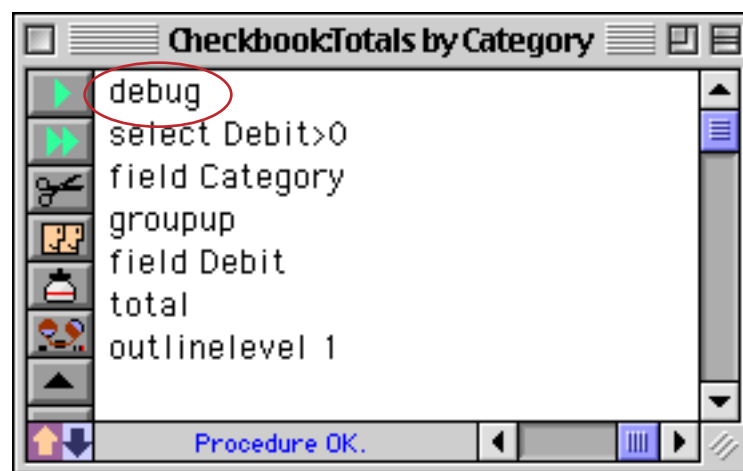
To help solve more stubborn problems Panorama includes a built in debugger. The debugger allows you to stop a procedure in the middle and execute statements one at a time (called **single-stepping**). You can actually watch as your program executes each statement, and you can check the value of fields and variables at any time.

#### The Debug Statement

The **debug** statement pauses the procedure so that you can examine fields and variables. You can insert a **debug** statement anywhere in a procedure, and a procedure may contain more than one **debug** statement. Usually **debug** statements are inserted into the procedure temporarily while you are getting the program running, and then removed when the procedure is operating properly.

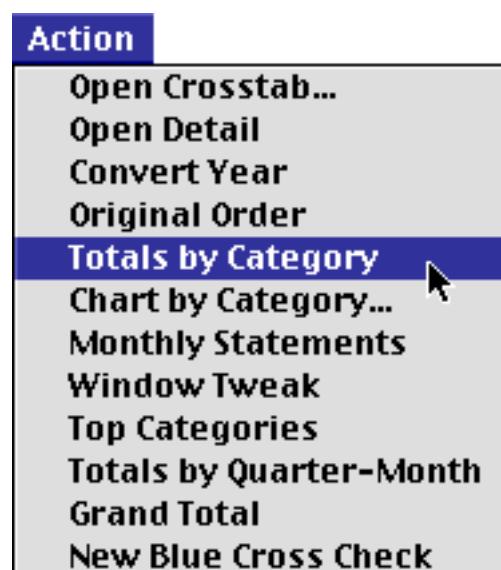
#### Using the Debugger

The first step in using the debugger is to add one or more **debug** statements to a procedure. In this example the **debug** statement has been inserted at the very top of the procedure, but it can be inserted anywhere.

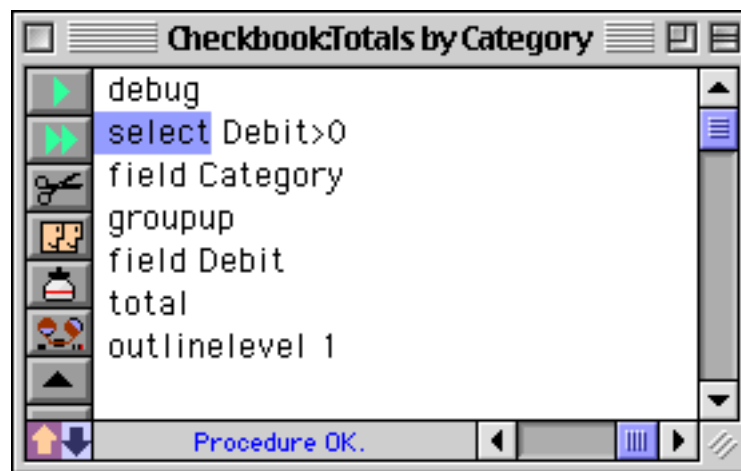


Once this is done, go to a form or data sheet window where you can test the procedure. Be sure to leave the procedure window open! If necessary, you can open an additional window in the same database.

Now start the procedure normally. Usually you will click on a button or pull down a menu item. If this is a "hidden trigger" procedure you should perform whatever action triggers the procedure.



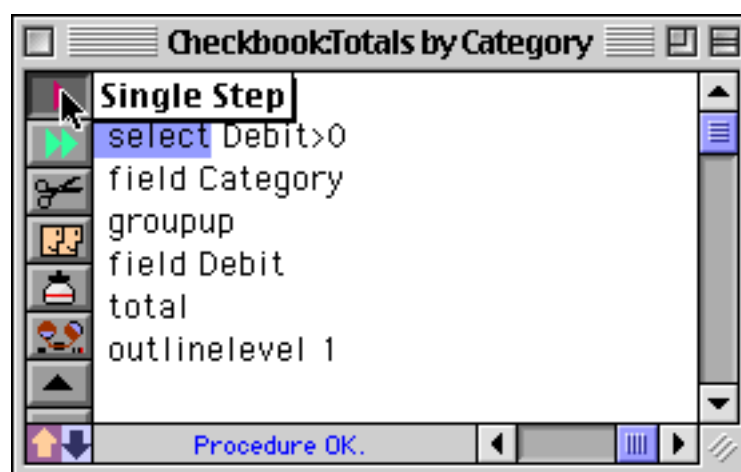
The procedure will run normally until it gets to the `debug` statement. At that point the procedure will stop and Panorama will bring the procedure window back to the front. The statement following the `debug` statement will be highlighted, indicating that it is the next statement to be performed when the procedure resumes.



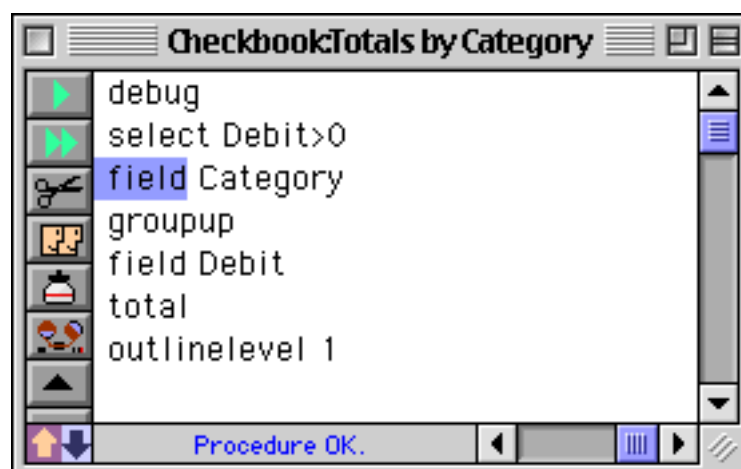
**Important:** If the procedure window is not open, the procedure will not stop. That's why it was so important to leave the procedure window open in the last paragraph. If you wish, you can leave `debug` statements permanently in a procedure. They won't affect the procedure unless the procedure window is open.

### Single Stepping

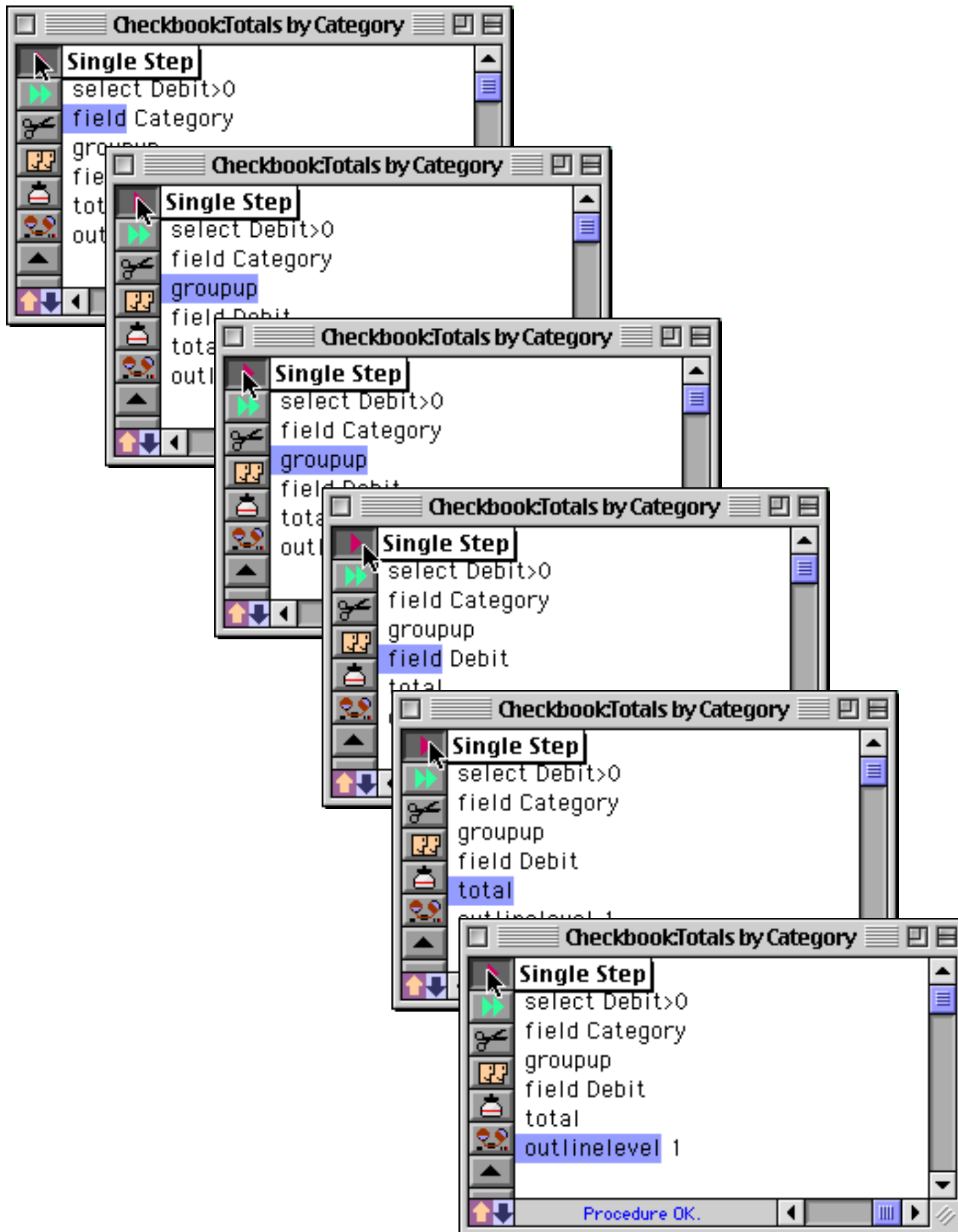
After the procedure has been stopped by a `debug` statement, you have the option of continuing the procedure one step at a time. You can watch to see what happens as each step is performed. To perform the next step, press the **Single Step** tool, or select **Single Step** from the Debug menu. (Note: you can also use the **Single Step** command without the `debug` statement simply by clicking on the tool. Panorama will start single stepping from the first line of the procedure.)



Before it performs the next statement, Panorama will move the procedure window to the back again, so that the form or data sheet (or whatever the current window was) is on top again. Panorama performs the statement, then brings the procedure window back on top again. The following statement is highlighted.



By single stepping again and again you can watch the program run. You can see as the procedure makes decisions at `if` statements, watch as a loop runs over and over again—everything your procedure does is instantly visible.



If the procedure uses `call` or `farcall` statement to trigger a subroutine (see “[Subroutines](#)” on page 606), single stepping usually considers the subroutine to be a single step. In other words, in one step Panorama will perform the entire subroutine. However, if the window containing the subroutine procedure is open, Panorama will single step through the subroutine, letting you see each step in it.

### Resuming Full Speed Execution

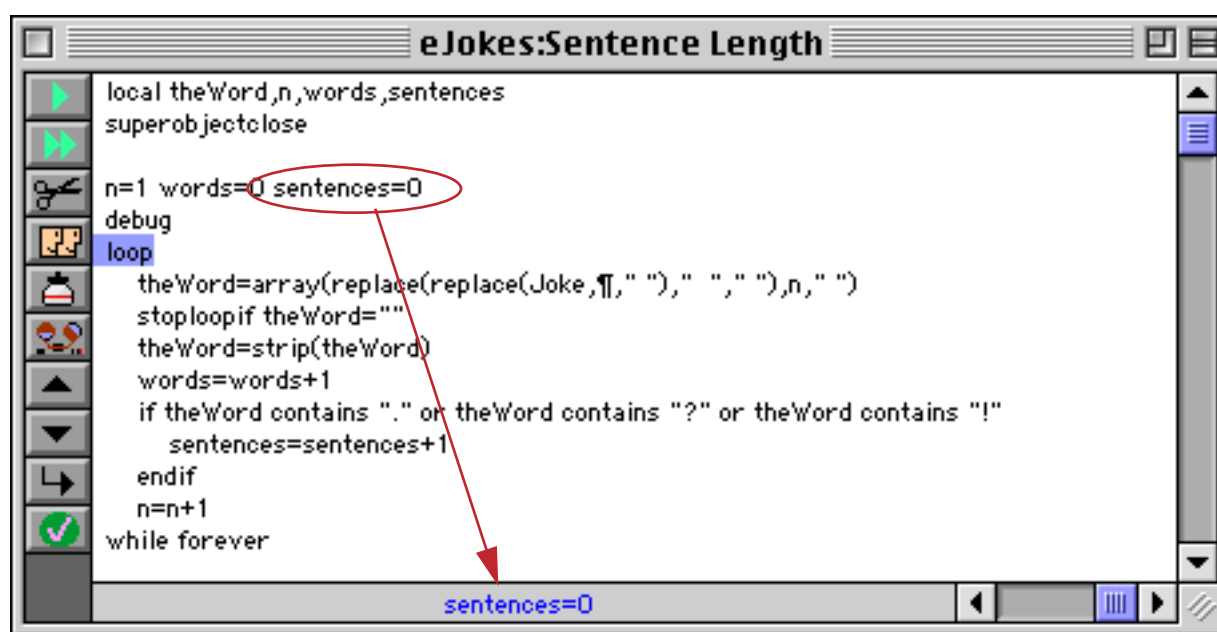
If you want the procedure to start up again at full speed, press the **Run** tool or select **Run** from the Debug menu. The procedure will start up again at full speed from the current spot. It will continue at full speed until it either reaches the end of the procedure, or it comes to another **debug** statement in an open procedure window.

### Making Corrections to a Procedure

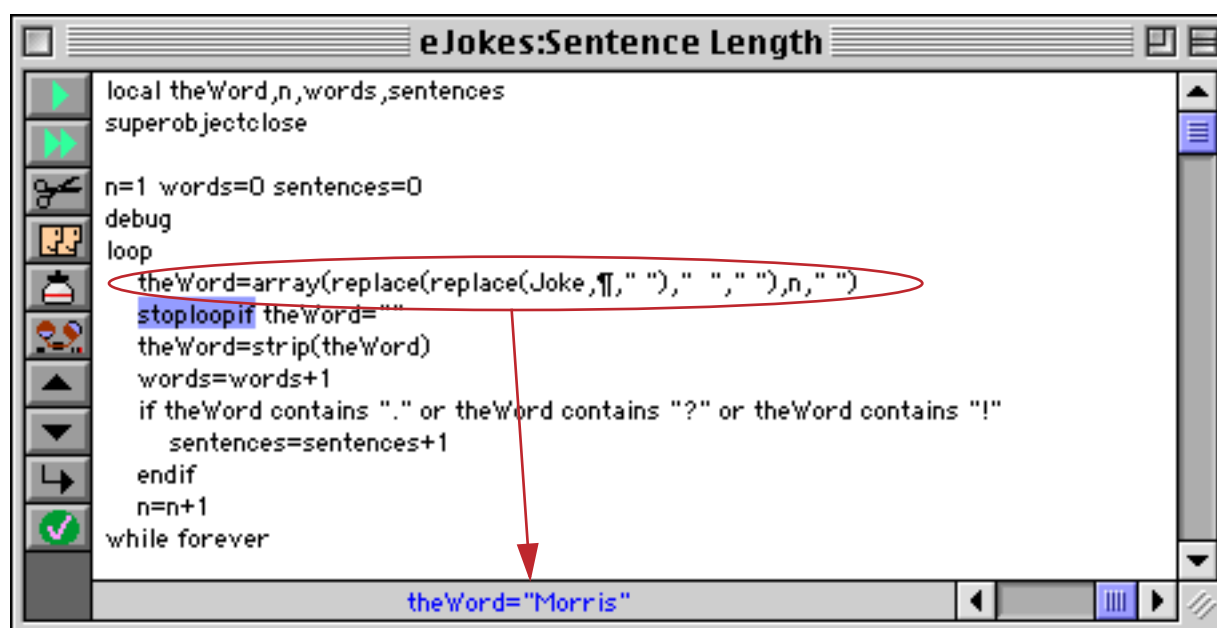
In the course of debugging you may find a problem with your procedure. To fix the problem, just edit the procedure. However, after you change the procedure you can no longer single step or continue the procedure. You must start over again from the top after any kind of change.

### Watching Computations

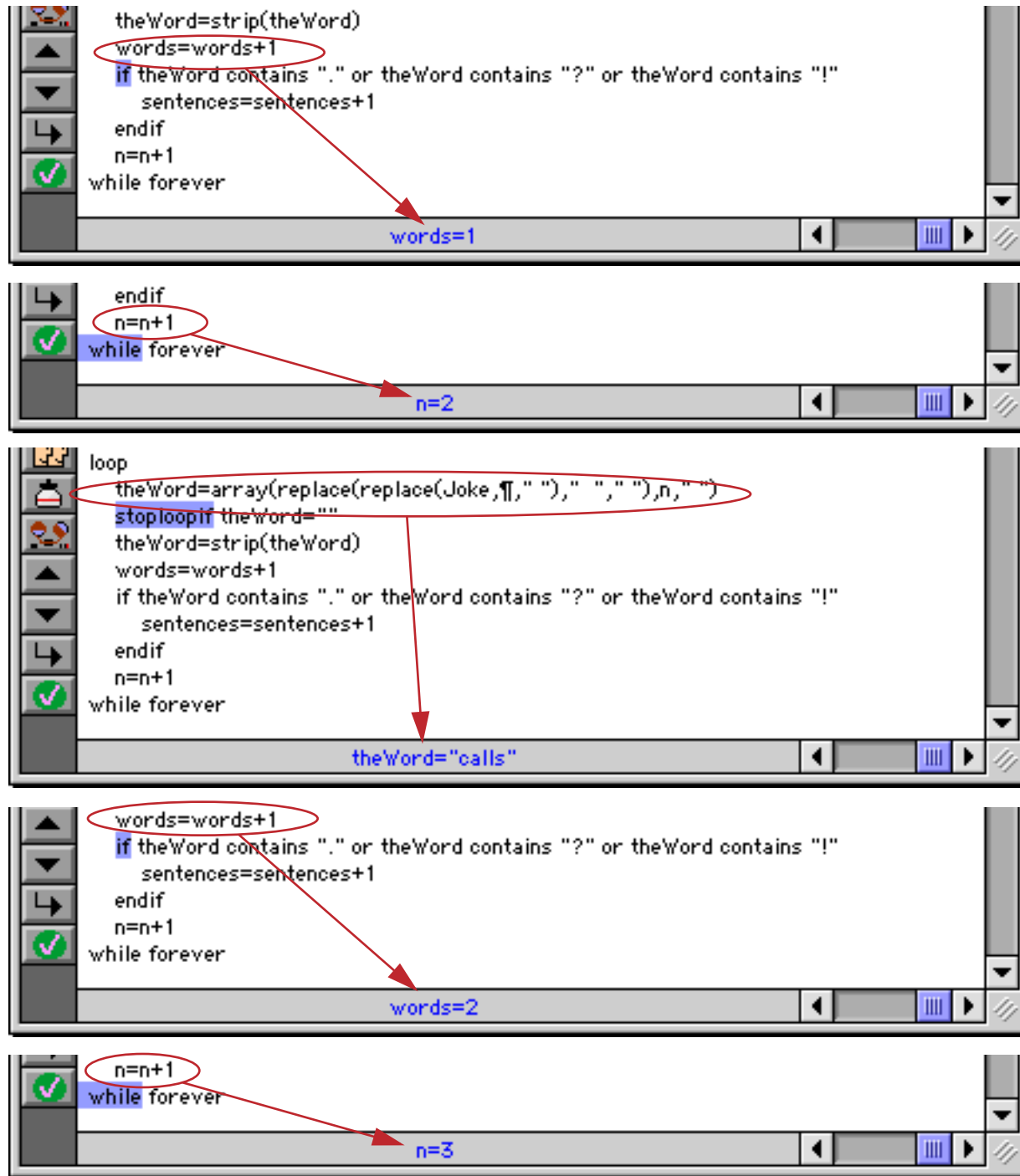
When you single step through a procedure Panorama updates the status bar to show the result of each assignment statement (see “[Assignment Statements](#)” on page 596). This makes it easier to follow along with what is going on in the procedure. For example, the procedure shown below has just stopped after the **debug** statement. The status bar shows the result of the last assignment statement, **sentences=0**.



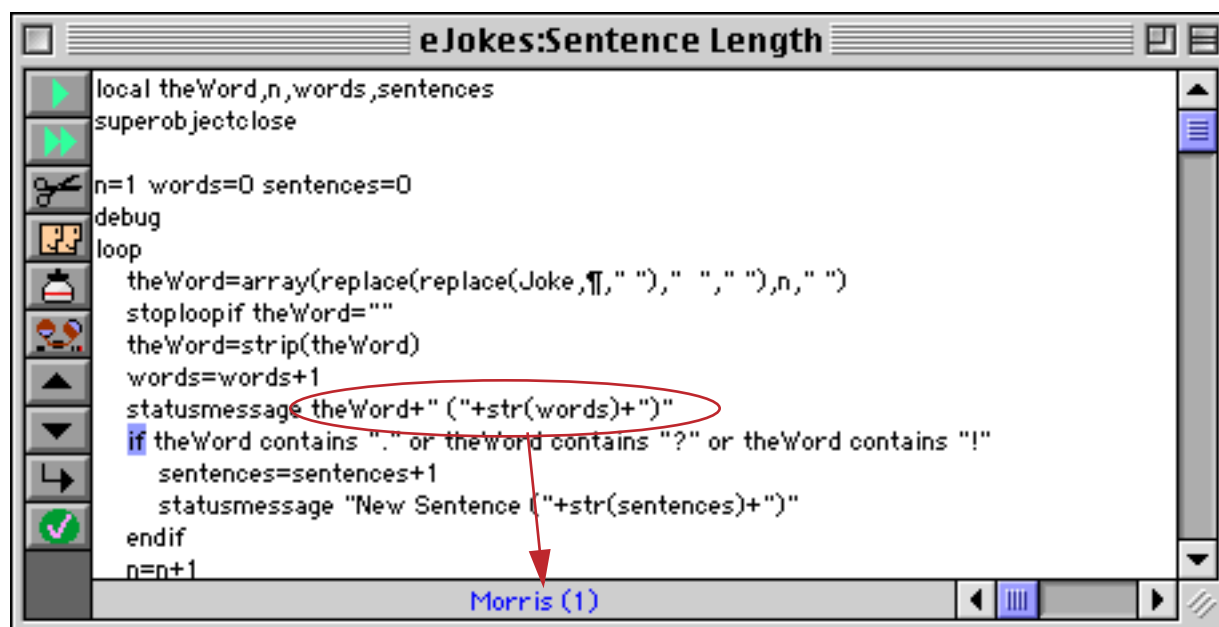
Single stepping twice executes the loop and assignment statements. Again, the status bar shows the result of the assignment, in this case the word **Morris**.



Each time you single step through an assignment statement the result is shown in the status bar.



You can use the `statusmessage` statement to display any formula in the status bar. This statement is similar to the `message` statement, but instead of displaying the message in an alert it displays it in the status bar, as shown below.

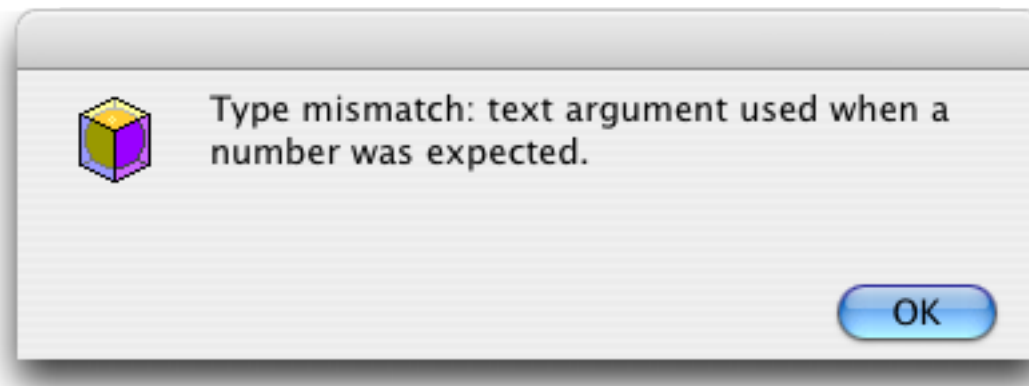


The `statusmessage` statement works any time the procedure window is open, even when the procedure is running at full speed. A strategically placed `statusmessage` statement can be ideal for watching the progress of a loop. For example the `statusmessage` statement in the procedure above will let you watch as the procedure counts the words in each sentence — 1, 2, 3 ... . If the procedure window is closed the `statusmessage` statement is simply ignored, so it is safe to leave this statement in your final procedure in case you need it later. (When the window is open the procedure may run slower than normal due to the time taken to update the status bar.)

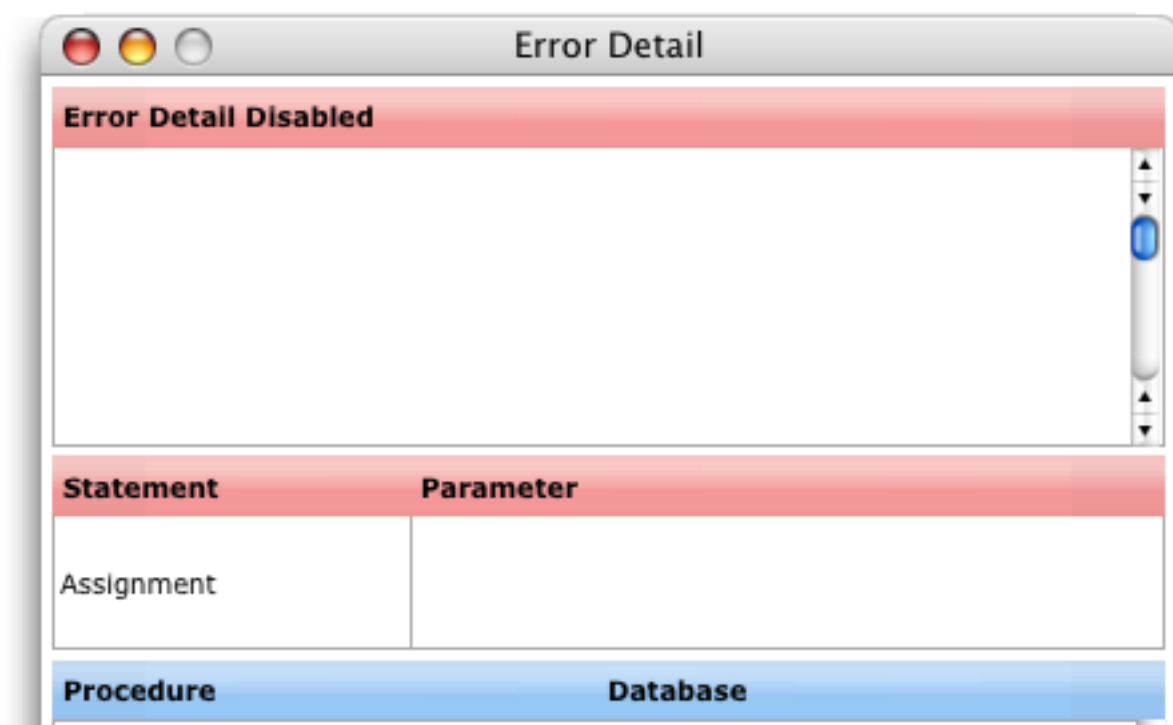


## Error Detail Wizard

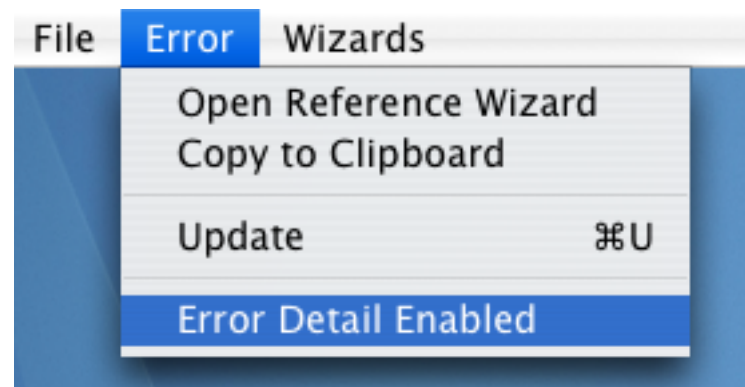
The **Error Detail** wizard can help track down the source of an error in a procedure or formula. When an error occurs, Panorama normally displays an alert, like this:



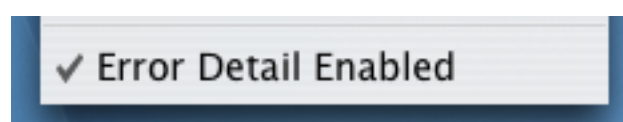
Once it is enabled the **Error Detail** wizard can give you more help in tracking down errors like this. Start by opening the wizard. As you can see, it is initially disabled.



To enable the wizard choose the **Error Detail Enabled** command in the Error menu.



The menu always shows the current status.

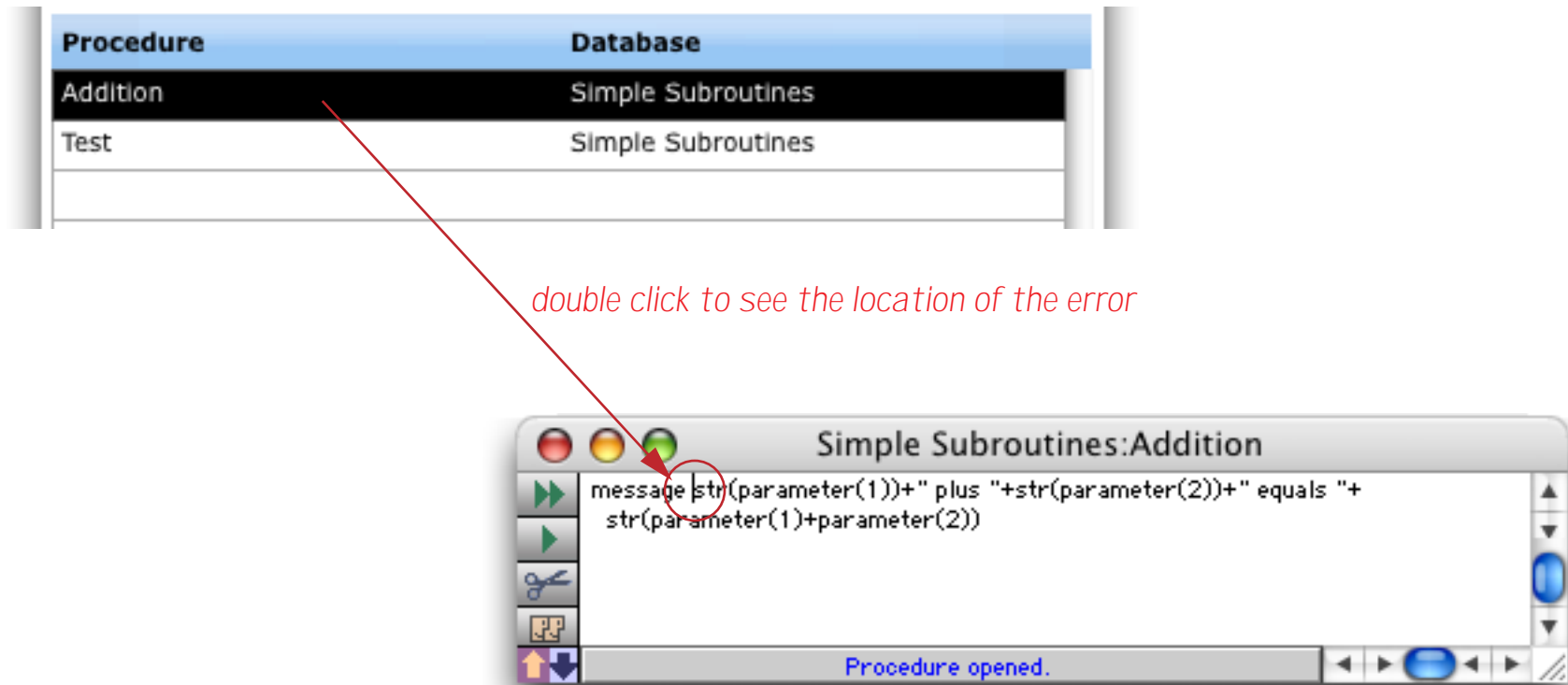


Once you enable **Error Detail** it will remain on until you explicitly turn it off (even if you close the wizard or completely quit and relaunch Panorama). Note: Sometimes Panorama becomes so confused by an error that this wizard doesn't display the correct information. Fortunately this happens quite rarely.

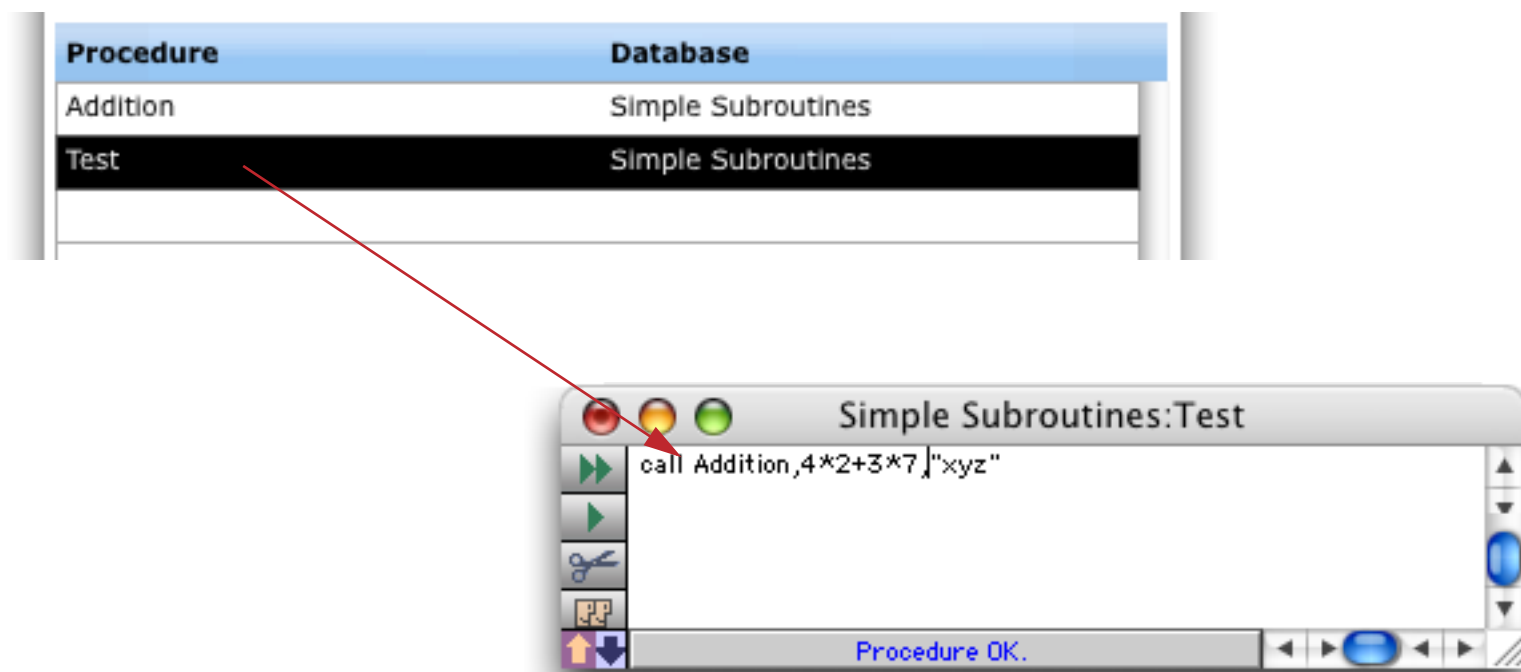


### Finding the Source of the Error

In addition to providing more information about an error the **Error Detail** wizard can also pinpoint the exact location where the error occurred. To find the exact location double click on the procedure name.

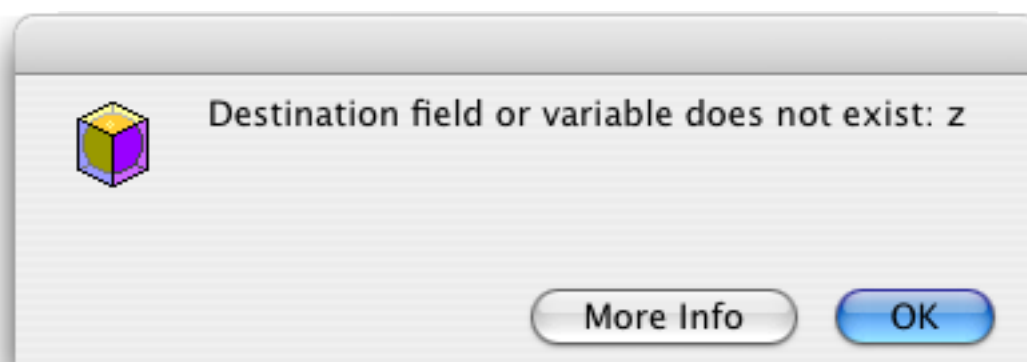


In some cases (like the example above) the actual problem isn't at the location where the error occurred, but further up the "call chain", where the procedure was called (see "[Subroutines](#)" on page 606). You can double any procedure in the "call chain" to see where the procedure containing the error was called.

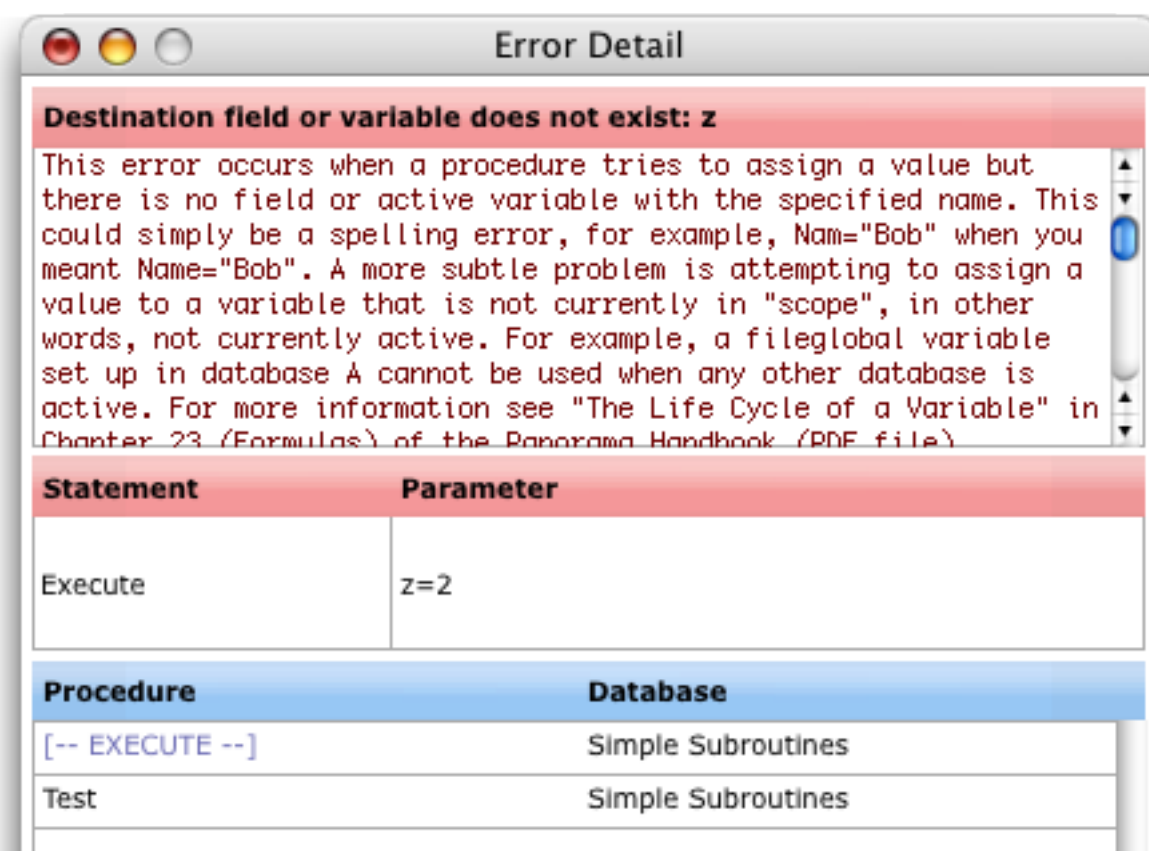


Now we can see the problem — the call statement is passing the text "xyz" when it needs to be passing a number like 123. Changing "xyz" to a number will fix the problem.

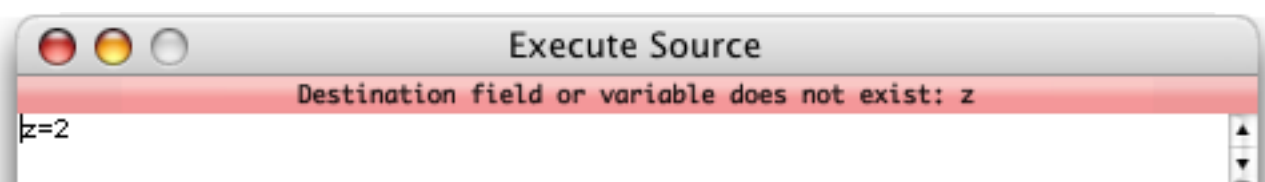
The wizard can also help track down problems that occur in an `execute` statement (see “” on page 612). Suppose you see an error message like this:



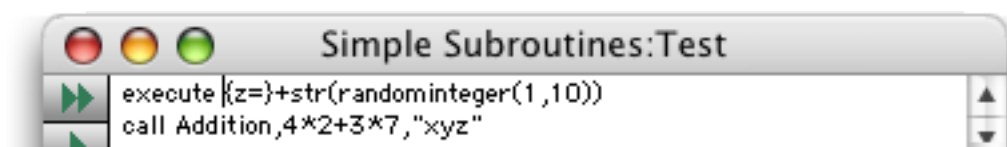
Press the **More Info** button to see the additional detail.



As you can see, the procedure containing the error has no name because it was built on the fly by an `execute` statement. Double click on this line to see the statement itself.



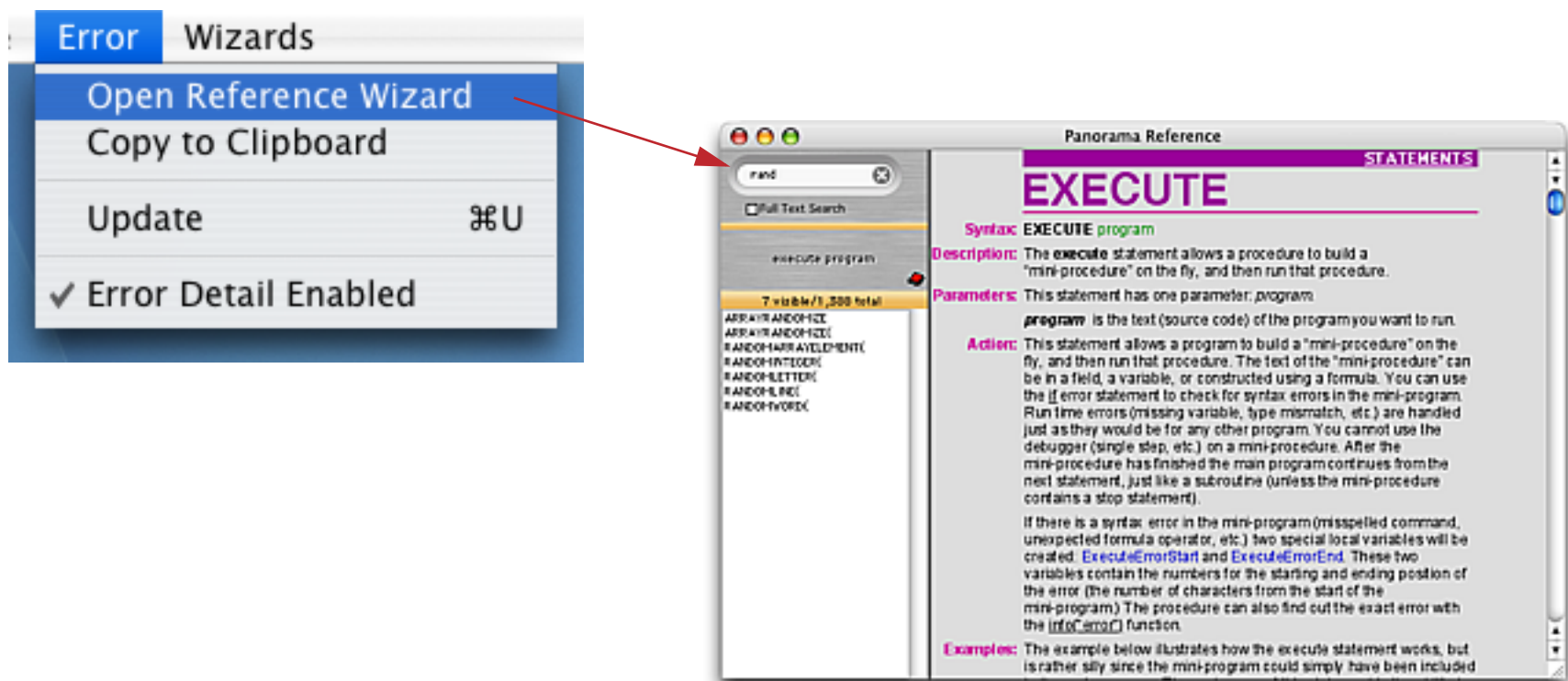
But where is this in the original program? Double click on the second line to see the procedure that contains the `execute` statement.



Ok, now the problem should be easy to fix. Notice that the actual statement in the two windows does not match. This is because the **Execute Source** window shows the statement after the formula has been evaluated. This can be very useful if the formula used to build the statement on the fly contains an error.

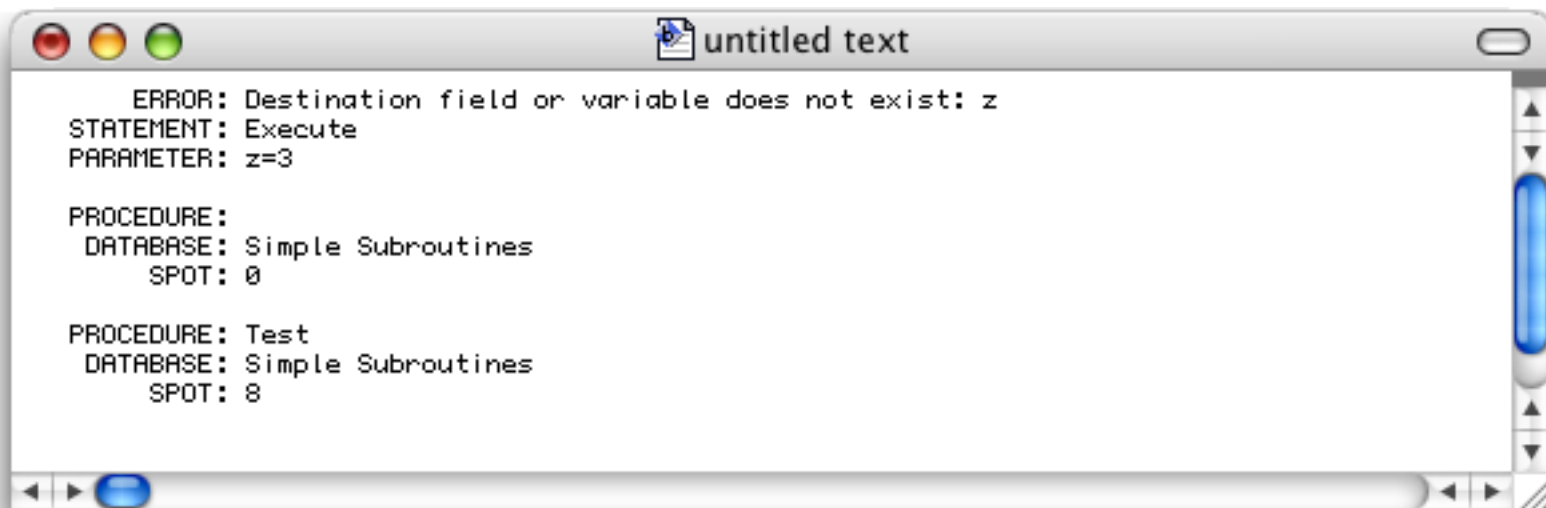
## Open Reference Wizard

Need more information about the statement that the error occurred in? Simply choose **Open Reference Wizard** from the **Error** menu. The **Programming Reference** wizard (see “[Programming Reference Wizard](#)” on page 591) will automatically open and display the page for the statement in question.



## Copy to Clipboard

This command copies the error detail so that it can be pasted into an e-mail, allowing it to be sent to someone else. Here's what the error detail looks like in text format.



If the PROCEDURE name is blank this code is in an `execute` statement. The SPOT indicates the location of the error within the source code. The spot is in characters, so for example the `call` statement in the test procedure is 8 characters from the start of that procedure (or in this case, the start of the statement defined by the `execute` statement).

## Error Detail Problems

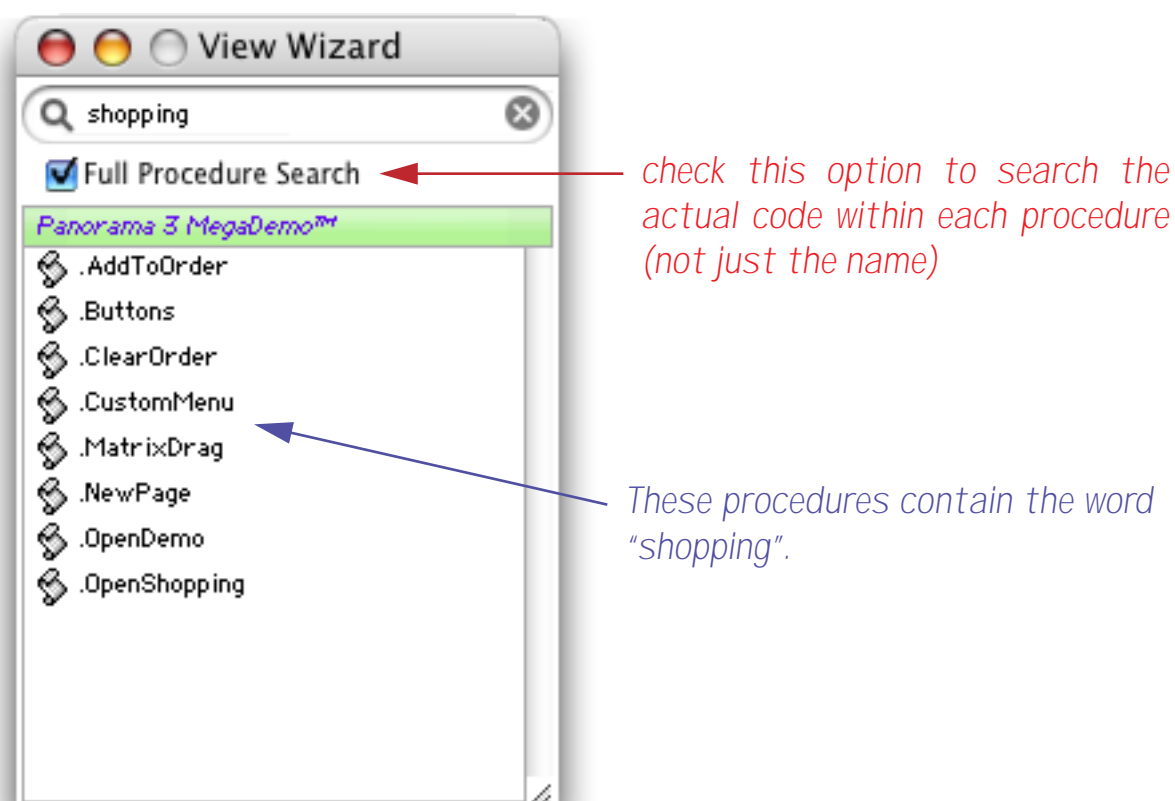
The **Error Detail** wizard works well in almost all situations, but there are a few advanced programming techniques can trip it up and prevent it from providing accurate information. Panorama was not originally designed to support this wizard, and in some situations we were simply unable to retrofit it to do so. The good news is that it will be immediately obvious when this happens, so you won't waste time tracking down bogus information. However in these cases you'll have to resort to more old-fashioned methods for tracking down the problem, for example inserting `message` statements into your code.

## Using the View Wizard with Procedures

The **View wizard** has some special features for working with procedures. You can search all procedures within a database, list information about procedures, even export procedure source and transfer procedures from one database to another. See “[The View Wizard](#)” on page 193 to learn the basics of working with this wizard.

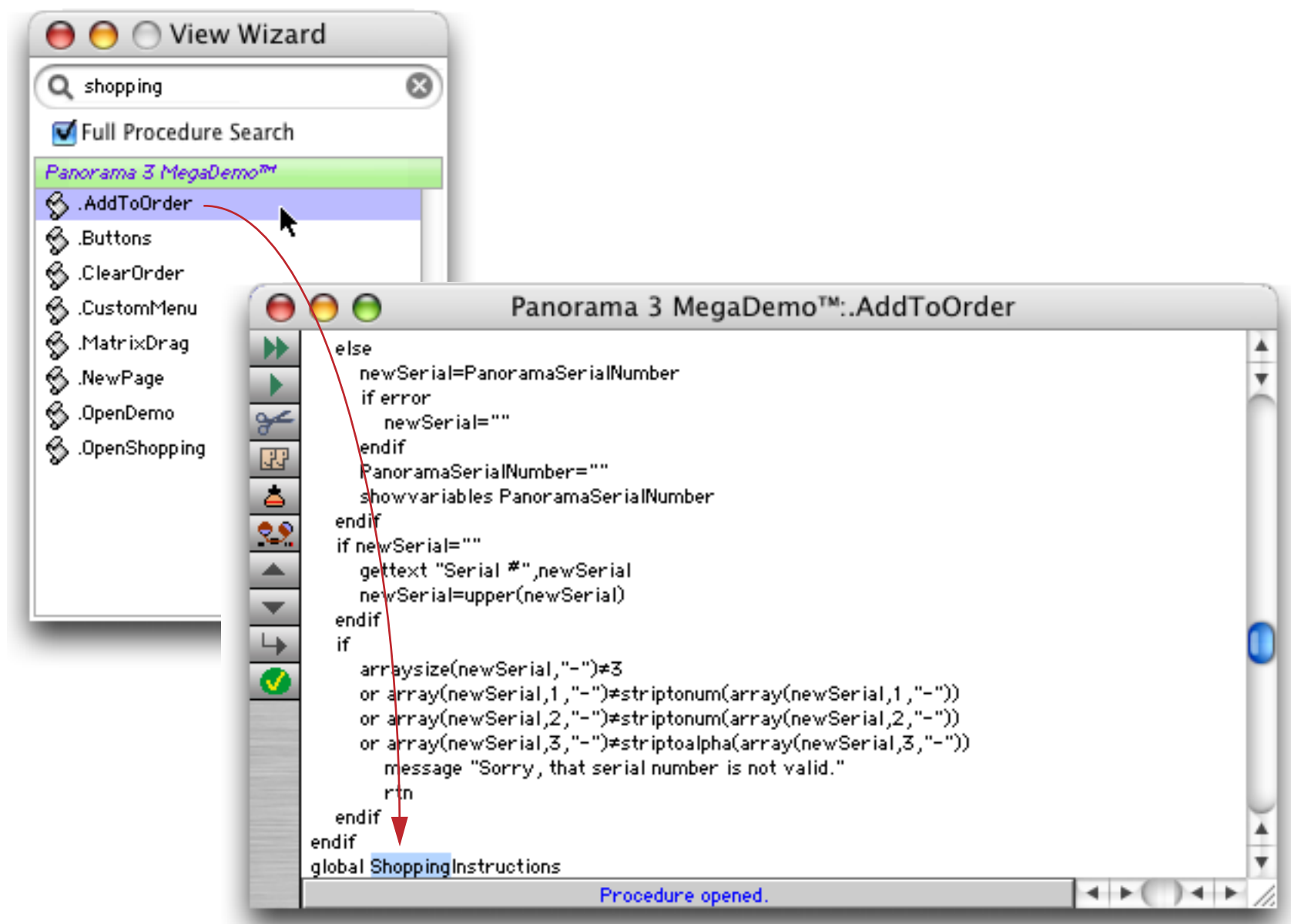
### Searching All Procedures

The **View Wizard** has the capability of searching the text of all procedures in a database. Simply check the **Full Procedure Search** option and type in the word or phrase you want to search for. The list will update as you type each key.

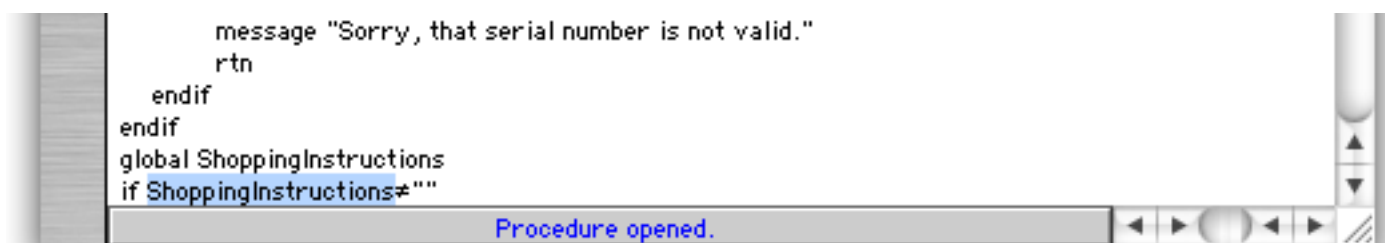




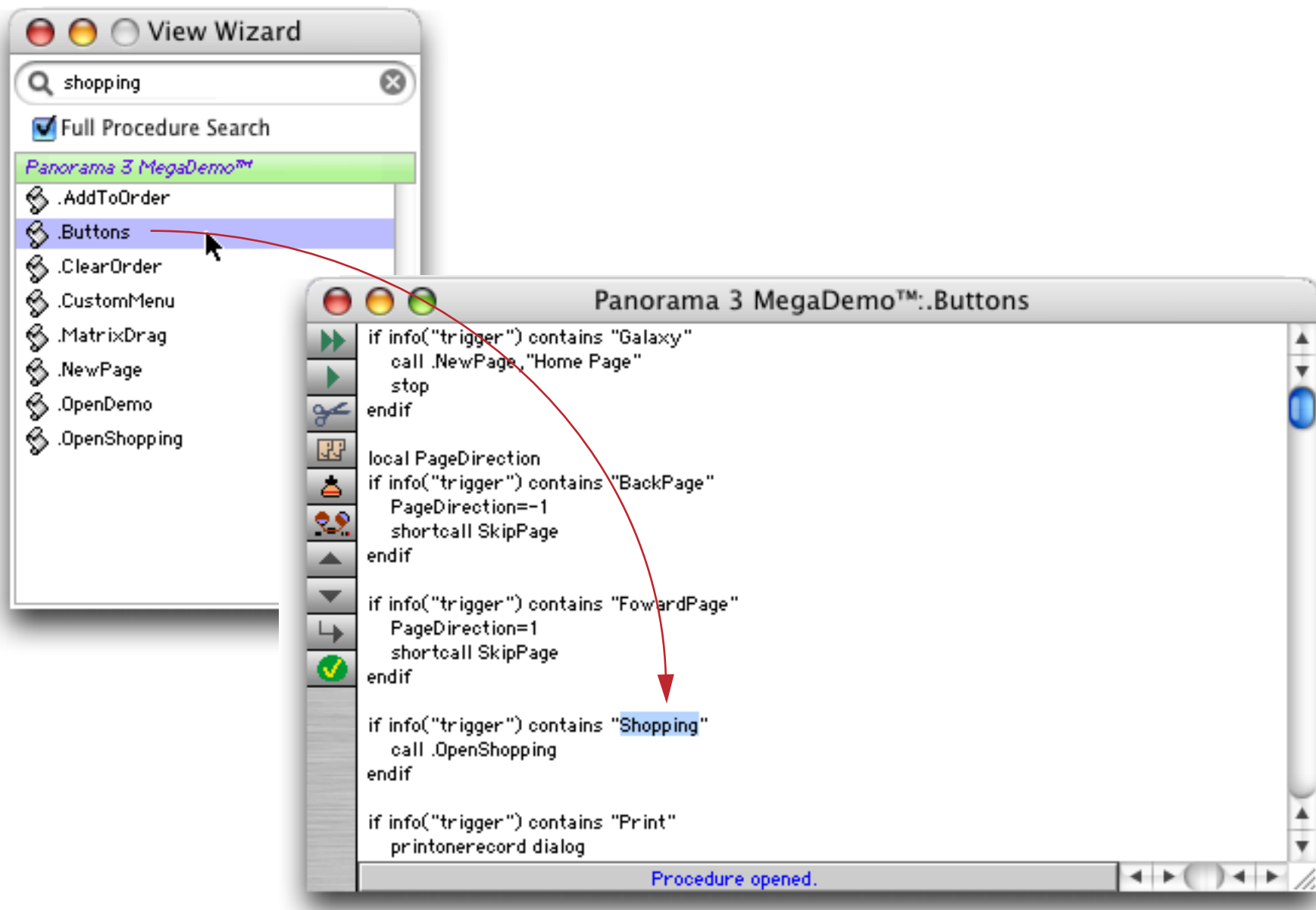
When you double click on one of these procedures the wizard will open the procedure window and automatically locate the first occurrence of the word or phrase.



Choose Find Next from the Search menu to find the next occurrence of this word or phrase within the procedure (if any).



You can repeat using the **Find Next** command until you have located every occurrence of the word or phrase in this procedure. At that point you'll need to go back to the **View Wizard** to continue with the next procedure.



You can continue this process until you have located every occurrence of the word or phrase in the database.

## 50 Ways to Trigger a Procedure

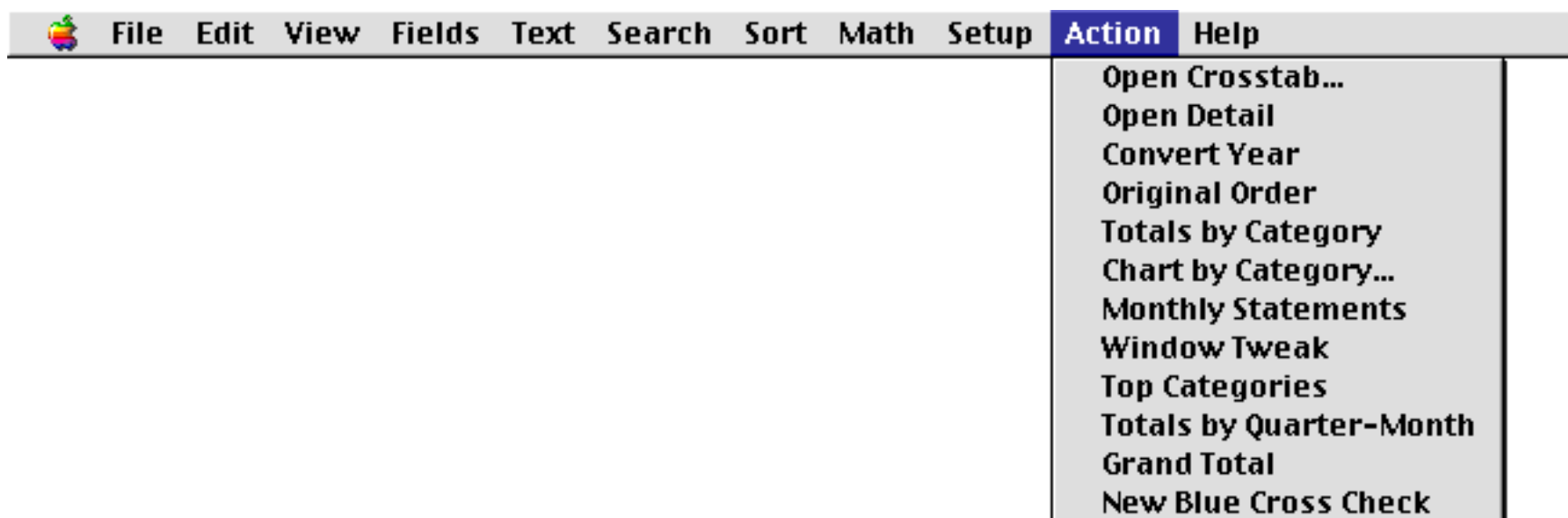
Procedures don't start up on their own — they must be triggered somehow. There aren't really 50 ways to trigger a procedure, but there are quite a few.

There are basically two types of triggers that can activate a procedure: **explicit triggers** and **hidden triggers** (implicit). Explicit triggers allow the user to deliberately trigger a procedure, for example by pressing a button or choosing an item from a menu. Hidden triggers activate a procedure automatically when the user performs some normal Panorama action. Examples of user actions that can cause a hidden trigger to activate include adding new records to a database, deleting records, opening a file, closing a window and many more. Procedures that are activated by hidden triggers can customize the way Panorama responds to these user actions, giving the programmer tremendous flexibility in creating a user interface that is appropriate for the task at hand.

The same procedure can be triggered different ways at different times. For example, the same procedure could be triggered both by a menu command or a button. If necessary, a procedure can use the `info("trigger")` function to find out how it was triggered.

### The Action Menu

The **Action** menu is the simplest way to allow a procedure to be triggered. All you have to do is create the procedure, and it is automatically listed in the **Action** menu. The **Action** menu is added to the end of the standard menus, and the user can activate any procedure simply by selecting its name from the **Action** menu. (Advanced tip: If a procedure may be triggered other means in addition to the **Action** menu (for example, by a button), you can use the `info("trigger")` function to find out which way the procedure was triggered. If the procedure was triggered by the menu this function will return **Action Menu**.)



Panorama allows some variations from the basic one-size-fits-all **Action** menu. The programmer can give the **Action** menu a different name, or even split the **Action** menu into multiple menus. The programmer can also exempt some procedures so that they are not listed in the **Action** menu (a procedure that is not listed can only be triggered some other way, for example by a button).

The **Action** menu does have some significant limitations. **Action** menus can only be added to the standard menus, they cannot replace the standard menus. **Action** menus cannot have any submenus. The **Action** menu cannot change when the user switches from form to form—it always contains the same items (unless you switch to a different database). In addition, there must be a separate procedure for each menu item in the **Action** menu. It is not possible to have multiple menu items handled by a single procedure. With these restrictions in mind, the **Action** menu is by far the easiest way to set up your own menus in a Panorama database. For many custom Panorama databases, the **Action** menu is the only user interface. (Note: Prior to version 3.0 the **Action** menu was called the **Macro** menu.)

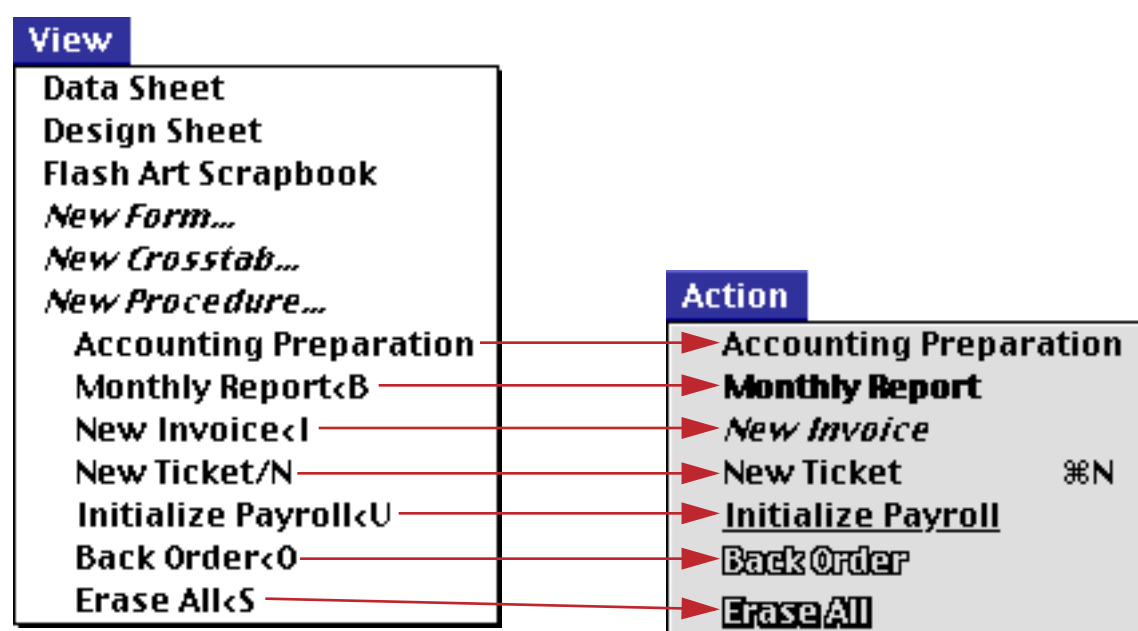
## Action Menu Options

By adding special characters to a procedure's name, you can change the way the procedure is displayed in the **Action** menu, or even remove the procedure from the menu completely.

There are two special characters that should never be used in a procedure name that is listed in the **Action** menu: **^** and **;**.

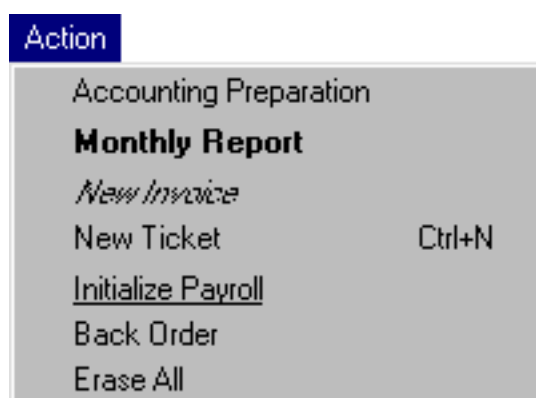
### Setting Different Menu Item Styles (Bold, Italic, etc.)

You can make a procedure name appear in several different styles in the **Action** menu—bold, italic, underline, outline, shadow, or a combination of these styles. To change the style of a menu item you must add a special suffix to the end of the procedure name. The suffix consists of the **<** character followed by the letter **B** (bold), **I** (italic), **U** (underline), **O** (outline) or **S** (shadow). The action menu below show all six different styles (including plain) and the procedure names for creating those styles.



You can also combine styles with multiple suffixes, for example **Initialize Payroll<B<I** for both italic and bold. You can also combine a style with a command key equivalent, for example **Back Order<I/B**.

Here's the same menu on a Windows based system.



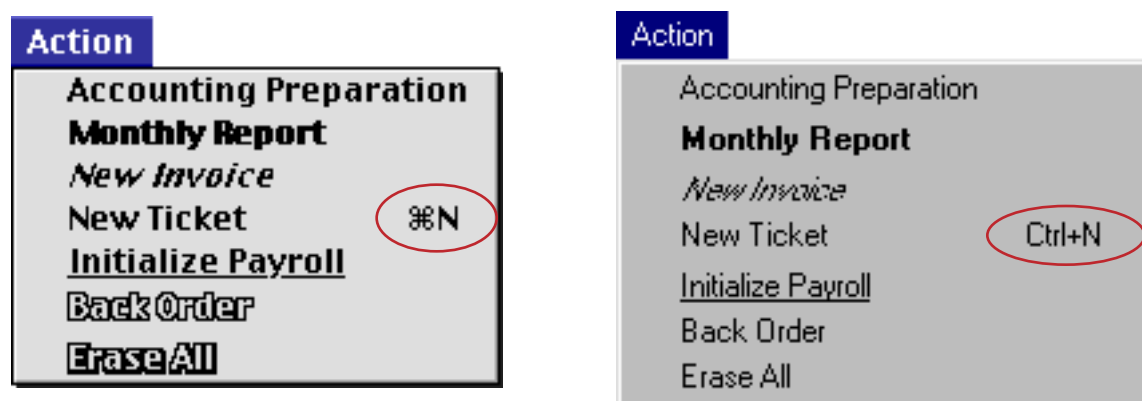
As you can see, the Outline and Shadow styles do not appear on Windows systems.

### Shortcuts/Command Key Equivalents

Like other menu items, procedures in the **Action** menu can have keys on the keyboard assigned to them. On the Macintosh these are called **Command Key Equivalents**, on the PC (Windows) they are called **shortcuts**. To assign a key to a procedure you add a suffix consisting of a **/** character followed by the key you want to assign to the procedure. For example, a procedure named

`New Ticket/N`

will show up in the **Action** menu assigned to the **N** key. Here is what this menu looks like on both the Mac (left) and Windows (right).



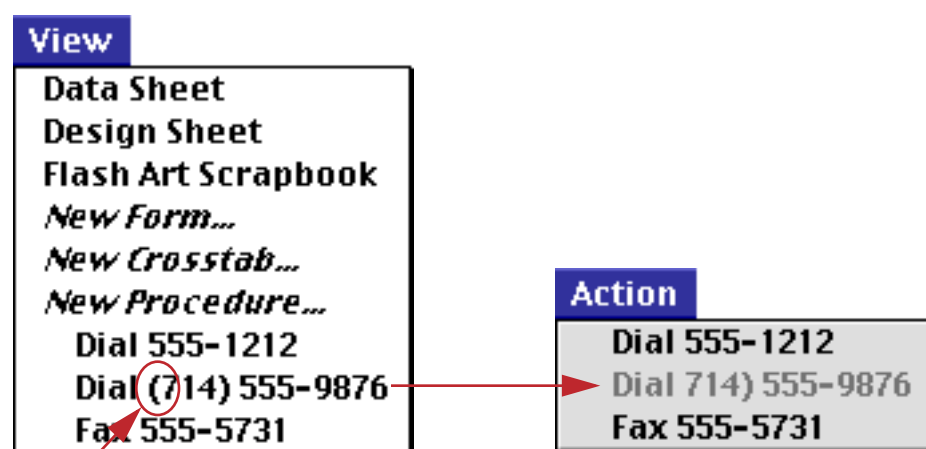
You can run this procedure by choosing it from the menu, or by pressing **Command-N** on the Macintosh or **Control-N** on a Windows based computer.

If a procedure's key assignment conflicts with one of Panorama's standard key assignments, the procedure will override the standard equivalent. For example, **Command/Control-P** is normally a command key equivalent for **Print**, but if you add a procedure called **Post Checks/P**, pressing **Command/Control-P** will trigger the procedure instead of printing.

Note: You cannot assign a command key equivalent to an "unlisted" procedure (one that begins with a period). Only procedures that appear in the **Action** menu can have command key equivalents.

#### Disabled Menu Items

If a procedure name contains the ( character, the procedure name will appear in the menu but will be disabled (gray).



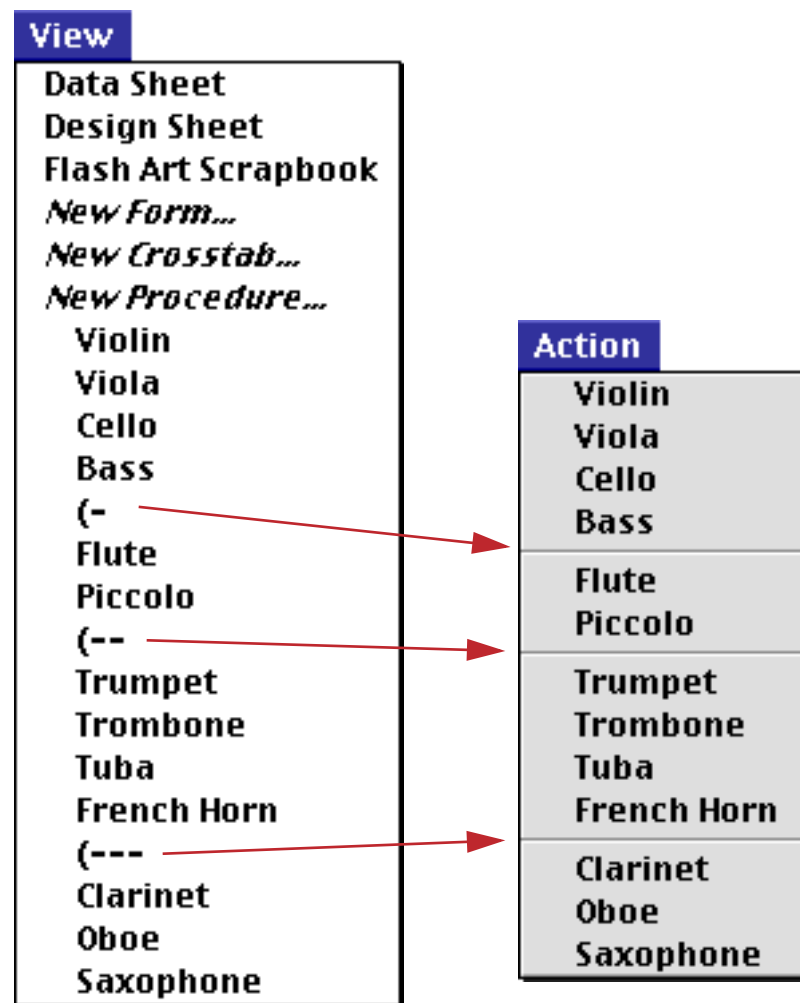
( in procedure name causes menu item to be disabled

Don't use parenthesis in a procedure name unless you want the procedure to be disabled.

#### Separator Lines in a Menu

Many menus contain one or more gray lines separating different sections in the menu. To add a gray line to the **Action** menu, create a procedure with a name that start with ( -. Use the **New Procedure** command in the **View** menu to create the procedure. Since the procedure will be disabled in the menu, it should not contain any statements.

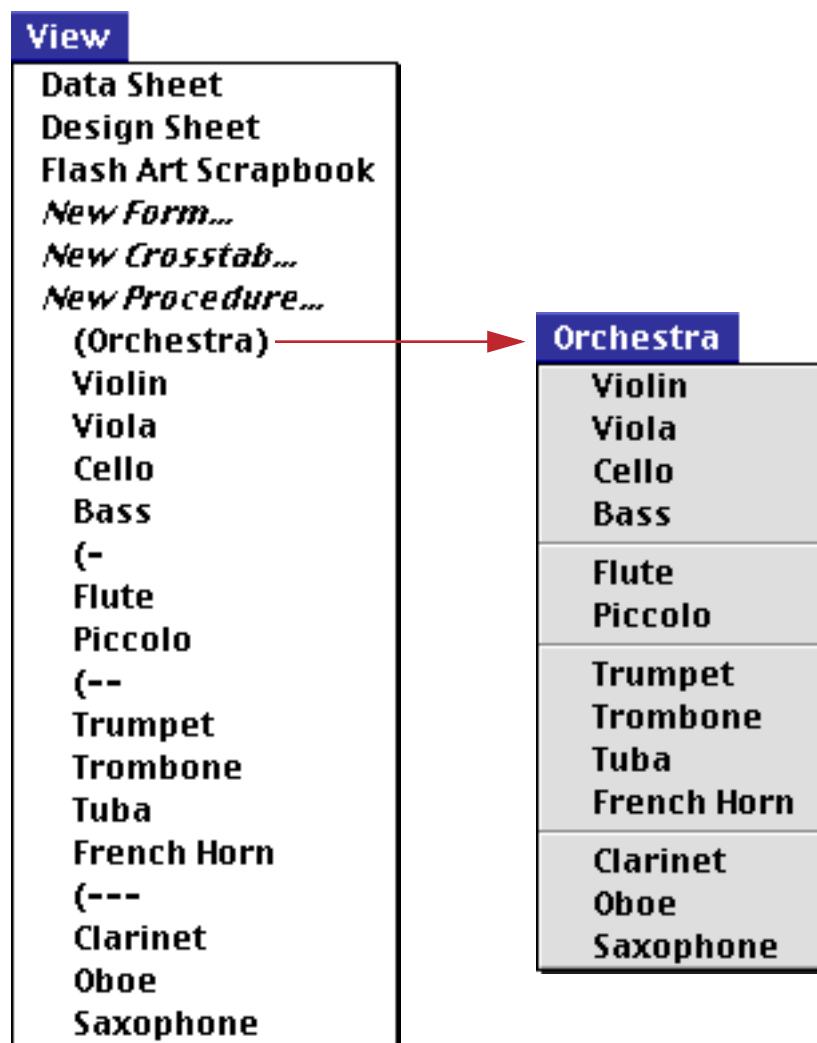
To add a second gray line to the menu, create a procedure named (- -. The third gray line should be named (- --, the fourth (- ---, etc. (Each procedure name must contain a different number of dashes because Panorama does not allow duplicate procedure names.) Here's an example of an **Action** menu divided into four sections.





### Renaming the Action Menu

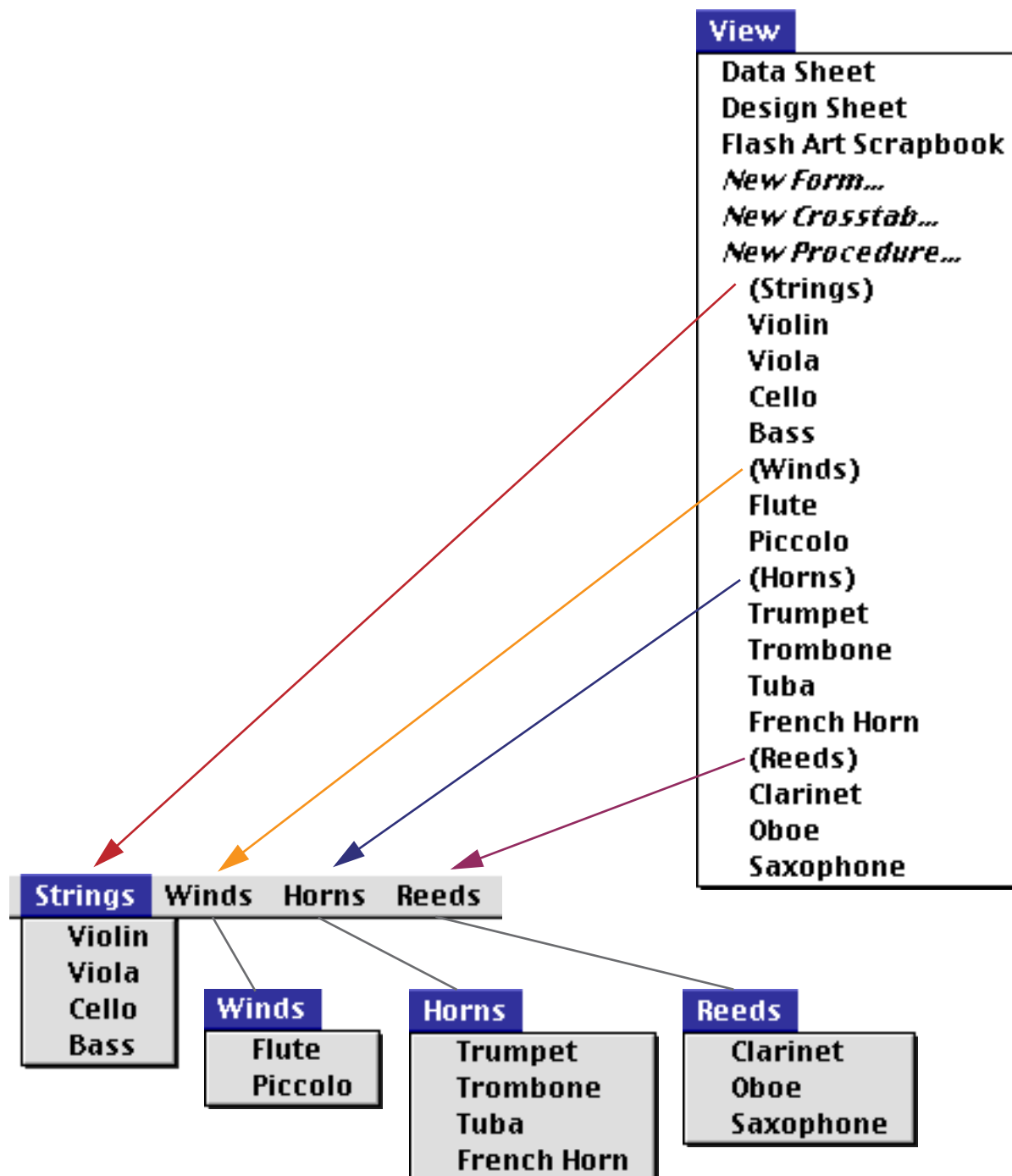
To give the **Action** menu a different name, insert an empty procedure with a name that begins and ends with a parenthesis as the very first procedure in the database. For example, inserting a procedure named **(Orchestra)** before the first procedure causes the **Action** menu to become the **Orchestra** menu.



See "[Creating a New Form, Crosstab or Procedure](#)" on page 202 to learn how to insert a procedure in any position.

### Dividing the Action Menu into Multiple Menus

If your database contains lots of procedures you may want to split the **Action** menu into two or more separate menus. To split the **Action** menu into separate pieces, insert an empty procedure with a name that begins and ends with parentheses. For example, to start a new menu named **People** add a new procedure named **(People)**. All of the procedures below this point will be listed in the **People** menu. You may split the **Action** menu into up to 12 separate menus. The example below shows an action menu split into four different menus.

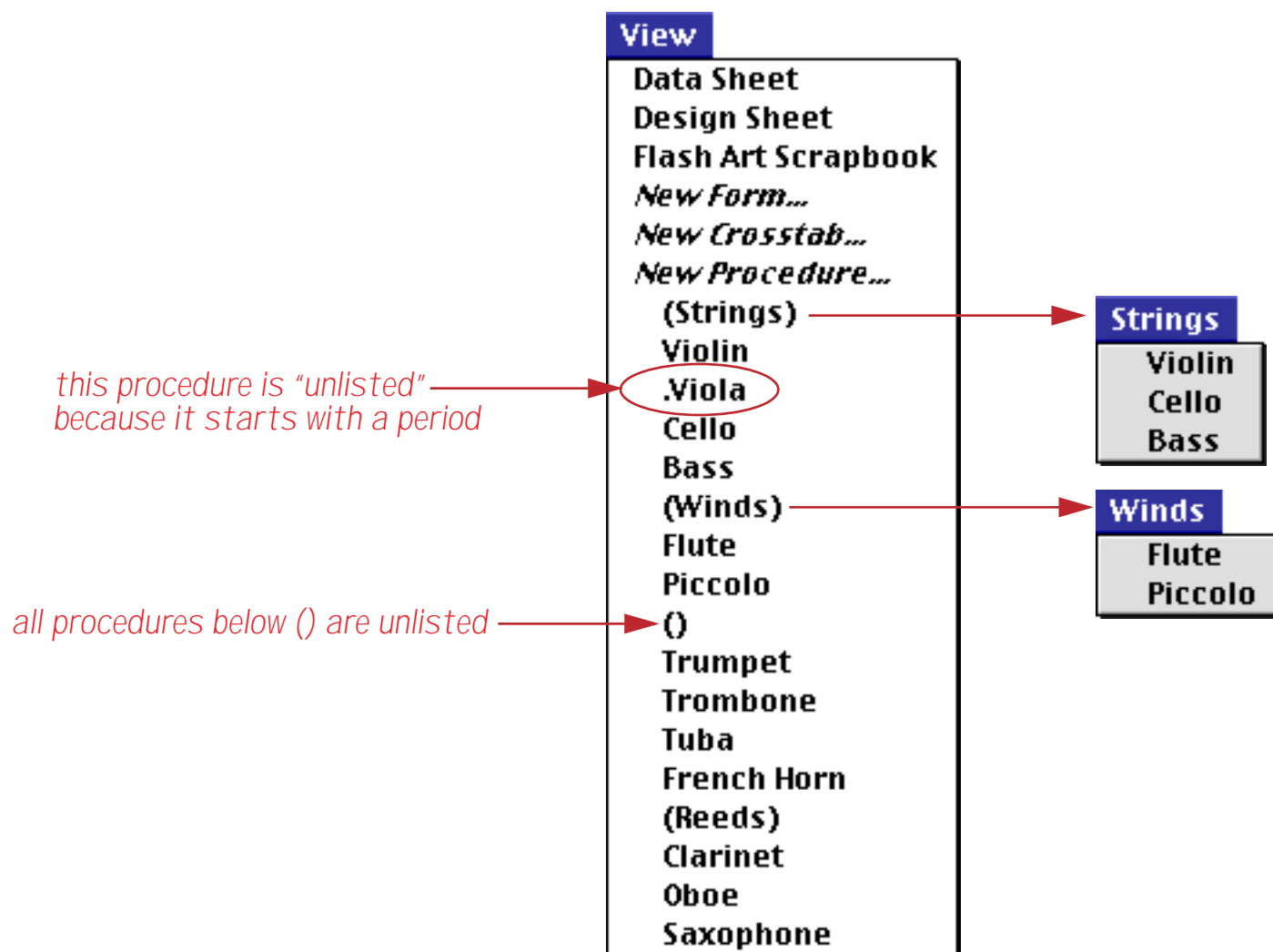


If necessary you can re-arrange the procedures to organize them into menus. See “[Changing the Order of Forms, Crosstabs or Procedures](#)” on page 203.

### “Unlisted” Procedures

You may not want the **Action** menu to list the special procedures you create for buttons, automatic events, custom menus, or subroutines. To keep a single procedure out of the menu, add a period to the beginning of the procedure name. Any procedure name that begins with a period will be “unlisted,” for example **.Balance** or **.Prepare Chart**.

To keep an entire group of procedures out of the menu insert a menu named **()**. Any procedure below a procedure named **()** will not appear in any menu.



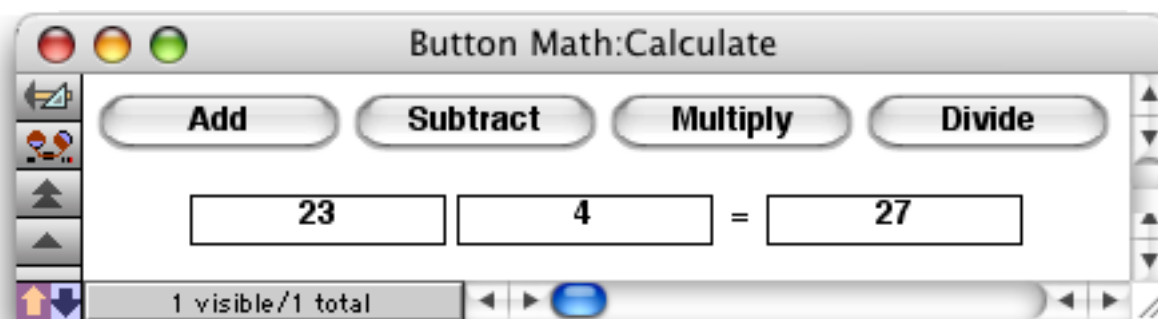
## Live Menus

The **Action Menu** is a very simple method for adding menus to your database. If you need more flexibility, however, you can use Panorama's "Live" menu feature. Live menus give you complete control over the content and appearance of each menu. To learn all about this advanced feature see Chapter 24 of the **Panorama Handbook**.

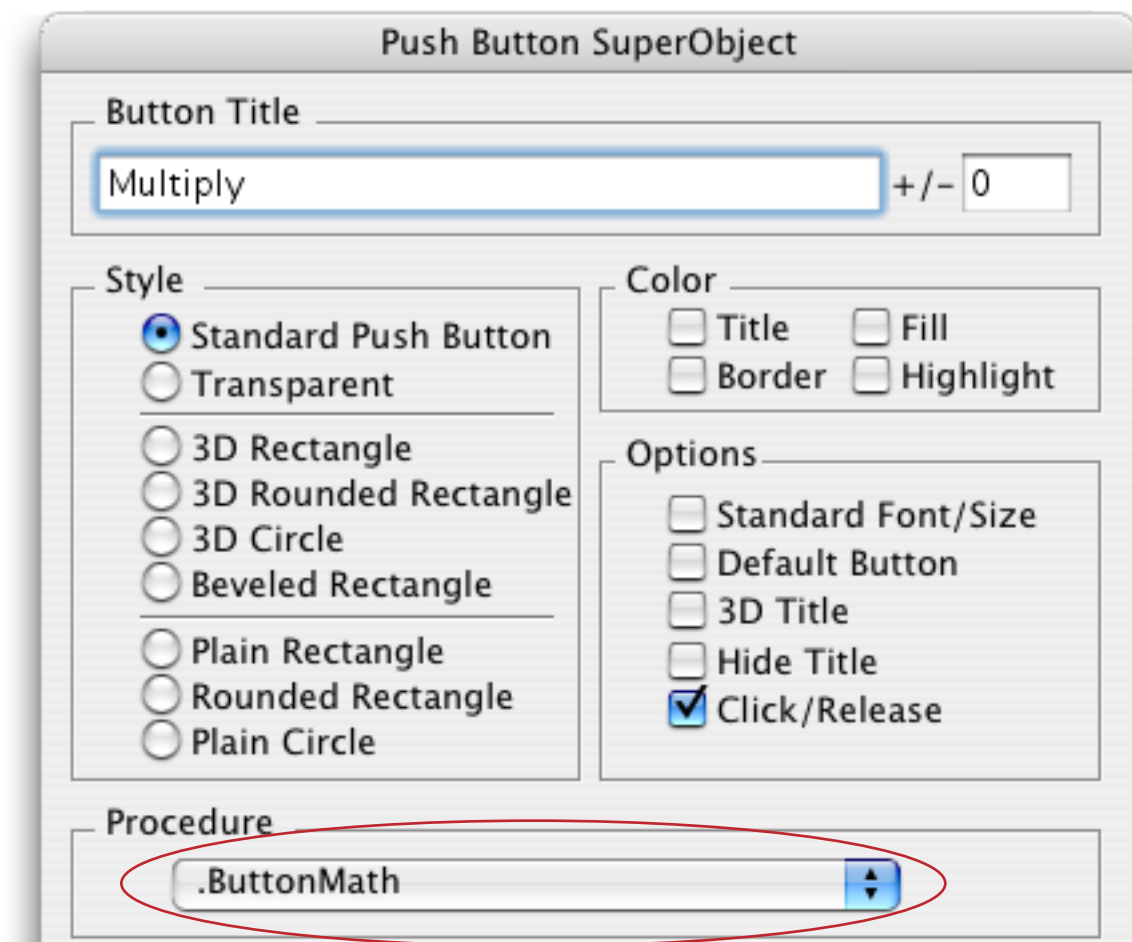
## Buttons

Buttons are an important part of the today's modern graphic user interfaces. You can use a wide variety of buttons in any Panorama form. Panorama buttons come in three basic varieties: push buttons, data buttons (checkboxes and radio buttons), and pop-up menu buttons. All of these types of buttons can trigger a procedure. Use the configuration dialog for the button to select which button will be triggered when the button is pressed (see Chapter 17).

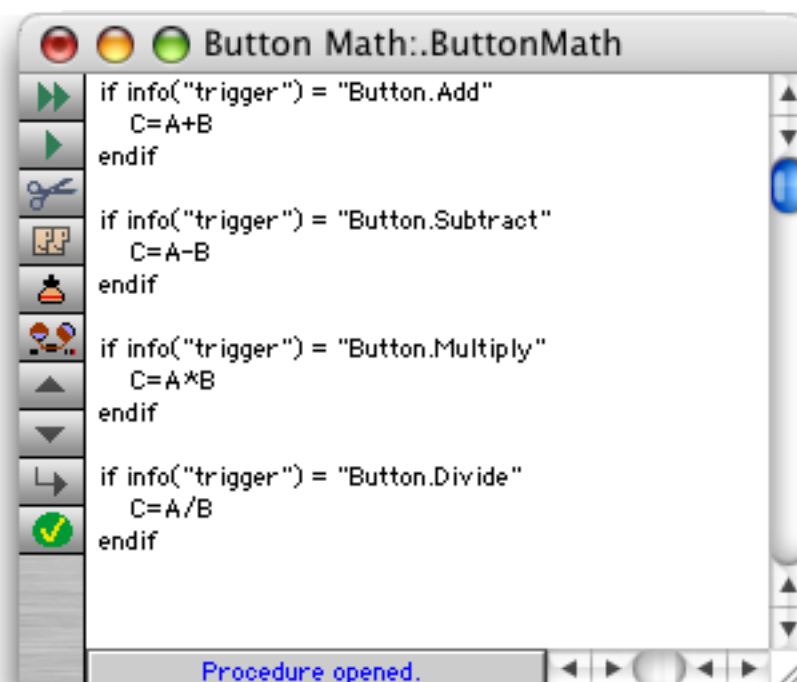
When a button is triggered by a procedure the `info("trigger")` function will return the title of the button. If you wish you may use a single procedure with many different buttons. For example, consider this form, which has four different buttons.



All four of these buttons trigger the **.ButtonMath** procedure. In fact, the only difference between these buttons is their titles. Here is the configuration dialog for one of these buttons.



The `.ButtonMath` procedure uses the `info("trigger")` function to decide which button was pressed.



Notice that the name of this procedure starts with a period. This makes this an “unlisted” procedure that does not appear in the **Action** menu (see See “[“Unlisted” Procedures](#)” on page 642). It wouldn’t make any sense to trigger this procedure from the menu, so it’s best to make it unlisted.

## Hidden Triggers

Hidden triggers activate a procedure automatically when the user performs some normal Panorama action. Examples of user actions that can cause a hidden trigger to activate include adding new records to a database, deleting records, opening a file, closing a window and many more. The trigger is “hidden” because the user is not explicitly asking Panorama to activate a procedure by pressing a button or selecting a menu choice.

Procedures that are activated by hidden triggers can modify (or even override) the way Panorama reacts to many standard user actions. For example, when a user clicks on a window’s close box, Panorama normally responds by closing the window. But with a hidden trigger the programmer can activate a procedure whenever the close box is clicked. This procedure can do anything the programmer wants. For example, the programmer may want to save the window position before the window is closed. Or the programmer may not want to let the user even close the window until all the data on a form is filled in. Of course this kind of flexibility comes with a price. The user expects the window to close—so any other action must be carefully designed so that it doesn’t confuse or frustrate the user.

To learn how to set up hidden trigger procedures see Chapter 24 of the **Panorama Handbook**.