R:BASE X.5

and

R:BASE X.5 Enterprise

Beginners Tutorial

The End-User's Guide to Building
R:BASE Databases and Custom Applications

R:BASE X.5 and R:BASE X.5 Enterprise

Beginners Tutorial

by R:BASE Technologies, Inc.

Welcome to R:BASE X.5 (Version 10.5)!

R:BASE X.5 (Version 10.5) is the innovative release from R:BASE Technologies offering the latest relational database elements to create function-rich and visually appealing applications.

The must-have powerful and flexible features give users an extended range of dashboard controls encompassing simplicity without sacrificing elegance.

Providing crucial data protection, that is also reliable and secure, R:BASE X.5 delivers a versatile environment for building robust systems for real time data analysis and essential business management.

R:BASE X.5 and R:BASE X.5 Enterprise Beginners Tutorial

Printed: May 2024 in Murrysville, PA

First Edition

Table of Contents

Part I Introduction 10

Part II Lesson 1 - Database Design 19

1 Introducing SkyWrite Ink ... 20
2 About this Lesson ... 20
3 Database Design? ... 21
4 Good Database Design ... 21
5 Final Design ... 28
6 Summary ... 28

Part III Lesson 2 - Defining a Database 30

1 Creating a Database ... 31
2 Defining Primary and Foreign Keys ... 36
3 Defining Indexes ... 40
4 The Database Files ... 42
5 Importing Data into the Database ... 43
6 Changing Database Structure ... 44
7 Summary ... 48

Part IV Lesson 3 - Viewing Data 49

1 Displaying Data ... 50
2 Hiding Columns ... 51
3 Redisplaying Hidden Columns ... 52
4 Changing the Column Size ... 53
5 Moving Columns ... 53
6 Locking Columns ... 53
7 Clearing Locked Columns ... 54
8 Working with Data ... 54
9 Adding a Row ... 54
10 Deleting a Row ... 54
11 Sorting Data by a Single Column ... 55
12 Sorting Data by Multiple Columns ... 55
13 Performing Calculations on Table Data ... 56
14 R:BASE BLOB Editor ... 57
15 The Query Wizard ... 57

16 Analyzing Data with Queries ... 60
17 Selecting Columns for a Query ... 61
18 Specifying a Condition for a Query ... 62
19 Modifying the Condition for a Query ... 64
20 Printing the Query Results ... 65
21 Creating Views ... 66
22 The Query Builder ... 67
23 Selecting Tables for a View ... 68
24 Linking Tables ... 68
25 Selecting Columns for a View ... 69
26 Looking at the Query Results ... 70
27 Saving the Query ... 71
28 Summary ... 71

Part V Lesson 4 - Creating and Modifying Forms 73

1 Designing a Form ... 74
2 The Form Designer ... 74
3 Creating a Quick Form with the Form Wizard ... 74
4 Understanding the Form Designer Interface ... 76
5 Field Object Properties ... 77
6 Customizing the Form ... 81
7 Editing Text Objects ... 81
8 Moving Objects ... 82
9 Changing the Size of Objects ... 83
10 Aligning Objects ... 86
11 Entering and Editing Text Objects ... 88
12 Creating a Bevel ... 89
13 Additional Objects ... 90
14 Form Properties ... 90
15 Saving, Testing and Closing Forms ... 91
16 Running the Form ... 91
17 Creating a Custom Form ... 92
18 Using Expressions and Variables ... 93
19 Placing Objects ... 95
20 Placing Text Objects ... 95
21 Placing DB Edit (column) Objects ... 96
22 Placing Variable Objects ... 97
23 Placing DB Memo (scrolling) Objects ... 98
24 Placing a DB Lookup Combo Box ... 99

25 DB User Defined Combo Box ... 102
26 Defining a Pop-Up Menu ... 102
27 Completing the Form ... 104
28 Form Hints ... 105
29 Changing Object Properties ... 105
30 Changing the Tab Order ... 106
31 Save the Custom Form ... 107
32 Running the Custom Form ... 107
33 Create Customer Form ... 108
34 Summary ... 108
35 Tips ... 108

Part VI Lesson 5 - Creating and Modifying Reports 111

1 Defining a Report ... 112
2 Creating an Employee Phone List ... 112
3 Building the Employee Phone List with the Report Wizard ... 113
4 The Report Designer ... 115
5 Modifying the Employee Phone List Report ... 117
6 Editing the Page Title ... 118
7 Changing the Column Headings ... 120
8 Resizing Objects ... 120
9 Report System Variables ... 121
10 Final Result ... 121
11 Previewing the Employee Phone List ... 122
12 Printing the Employee Phone List ... 122
13 Launching Files ... 123
14 Creating a Scheduled Flights List ... 124
15 Designing the Scheduled Flights Report ... 124
16 Building the Scheduled Flights Report ... 125
17 Report Sections ... 126
18 Removing Report Sections ... 126
19 Adding a Report Title ... 127
20 Changing the Size of the Detail Section ... 127
21 Placing Label (text) Objects ... 128
22 Aligning the Label Objects ... 128
23 Spacing the Label Objects ... 129
24 Placing DB Label (column) Objects ... 129
25 Placing a DB Memo (column) Object ... 130
26 Resizing and Aligning Column Objects ... 131

27 Saving and Previewing the Scheduled Flights Report ... 132
28 Modifying the Scheduled Flights Report ... 133
29 Adding a Format for Null Values ... 133
30 Adding a Breakpoint ... 133
31 Placing Objects in the Break Header ... 134
32 Adding a Display Format ... 135
33 Choosing DB Calc over using a computed variable in reports ... 136
34 Adding a Daily Total object using DB Calc ... 136
35 Placing Objects in the Report Footer Section ... 137
36 Adding a Grand Total DB Calc object ... 137
37 Changing Printing Styles ... 138
38 Finishing the Report ... 138
39 Saving and Closing the Report ... 139
40 Printing the Scheduled Flights Report ... 139
41 Summary ... 139
42 Tips ... 140

Part VII Lesson 6 - Creating and Modifying Labels 144

1 Building a Label ... 145
2 Designing a Mailing Label ... 145
3 Building a Label for a Customer Mailing List ... 146
4 Defining Variables for the Label ... 147
5 Placing Objects on the Label ... 148
6 Placing the Remaining Objects ... 149
7 Previewing the Label ... 149
8 Saving and Closing the Label ... 150
9 Line Compression ... 150
10 Summary ... 153

Part VIII Lesson 7 - Using R:BASE Commands 154

1 Opening the R:BASE R> Prompt ... 155
2 Using the Help at the R> Prompt ... 155
3 R:BASE Command Syntax ... 155
4 Using R:BASE Commands at the R> Prompt ... 157
5 Using Variables and Expressions ... 158
6 Summary ... 160

Part IX Lesson 8 - Writing Command Files 161

1 Using The R:BASE Editor (RBEDIT) ... 162
2 Designing And Writing A Command File ... 162

3 Testing The Command File 164
4 Enhancing The Command File 164
5 Summary 168

Part X Lesson 9 - Creating and Using an Application 169

1 Designing an Application 170
2 Identifying the Tasks Skyapp Will Perform 170
3 Organizing the Tasks Into Menus 171
4 Building an Application 171
5 Creating Menu Actions to Enter Data 173
6 Creating Menu Actions to Edit Data 177
7 Creating Menu Actions to Print Data 179
8 Assigning Actions to the Flights and Exit Menus 182
9 Defining the Main Menu 183
10 Defining the Employees Main Menus 184
11 Defining the Flights and Exit Menus 186
12 Final Steps 187
13 The Application Designer Environment 187
14 Testing an Application 188
15 Testing the Employees Menu 188
16 Testing the Customers Menu 190
17 Testing the Flights Menu 191
18 Exiting the Application 191
19 Modifying an Application 192
20 Running the Final Application 193
21 Summary 194
22 Tips 194

Part XI Lesson 10 - Maintaining a Database 196

1 Backups 197
2 What to Back Up 197
3 Checking the Database 198
4 Copying Database Files 198
5 Understanding R:BASE UNLOAD and RUN 198
6 Backing Up Database Structure 200
7 Backing Up Database Data 201
8 Backing Up a Database 201
9 Confirming the Backup File was Created 202
10 Restoring a Database 202

11 More on the UNLOAD Command 204
12 Compressing the Database 204
13 Pack In Place Versus Reload 204
14 Using the Pack In Place Option 205
15 Using the Reload Option 206
16 Summary 206

Part XII Useful Resources 207

Part XIII Feedback 209

Index 211

Part I

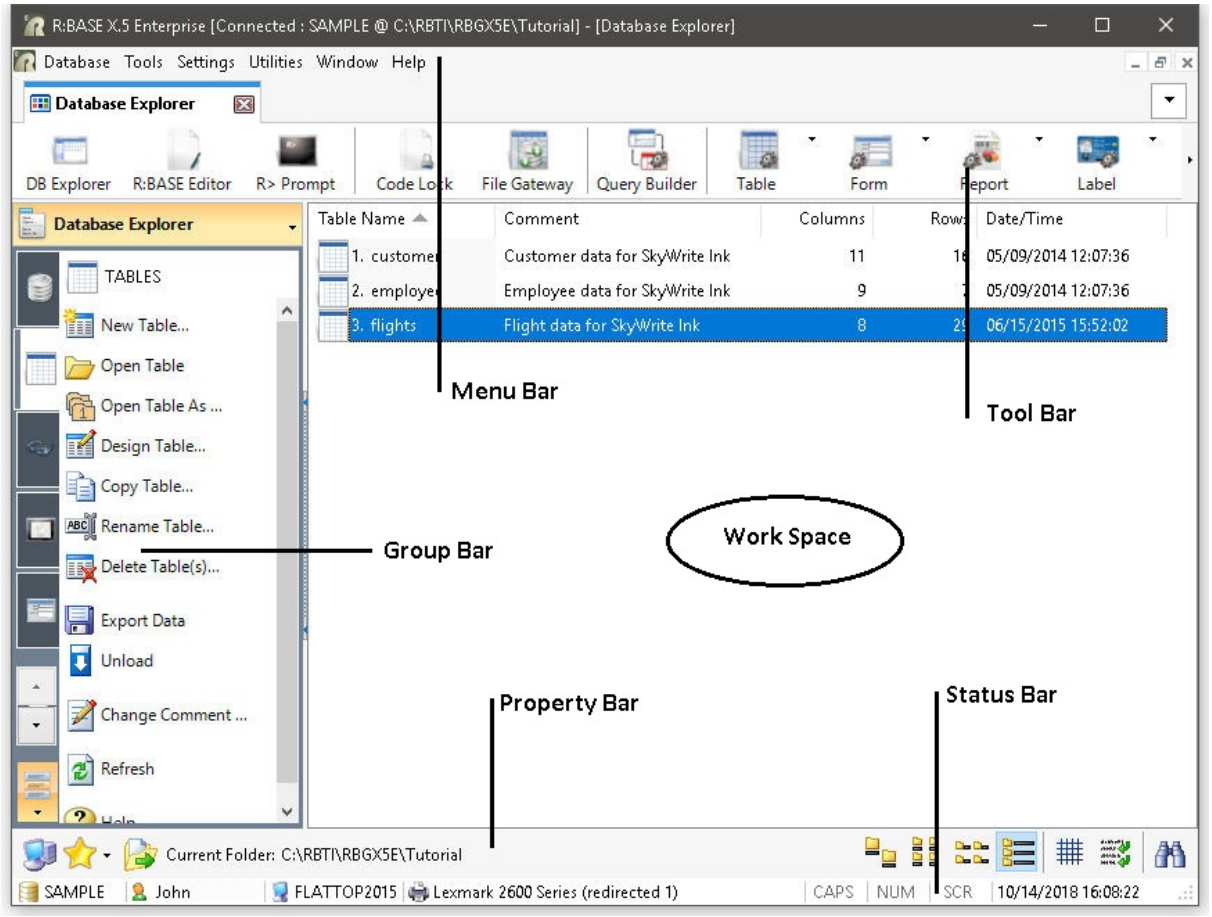
1 Introduction

Before you start working with R:BASE, take some time to get acquainted with its menus and other features. Learning the basic techniques of using R:BASE now will make it easier for you to complete the lessons that follow.

Here's what you'll learn about navigating in R:BASE:

- Starting and leaving R:BASE
- Getting help
- Using special features in R:BASE

Starting R:BASE
Start R:BASE from the Windows Operating System Desktop. Double-click the "R:BASE X.5 for Windows" icon. Before you start working on the lessons, familiarize yourself with the "R:BASE X.5" window, which has six major parts: the Menu Bar, Toolbar(s), Group Bar, Work Space, Status Bar, and Property Bar, as shown below:



The R:BASE X.5 Work Space area can be used to work in the Database Explorer (above), R:BASE Editor, R> Prompt, Data Designer, Query Builder, Report Designer, Form Designer, Label Designer, Application Designer and External Form Designer.

For a description of each module listed click on the blue underlined text. You will see these "link" references through the R:BASE Tutorial and documentation.

R:BASE Windows

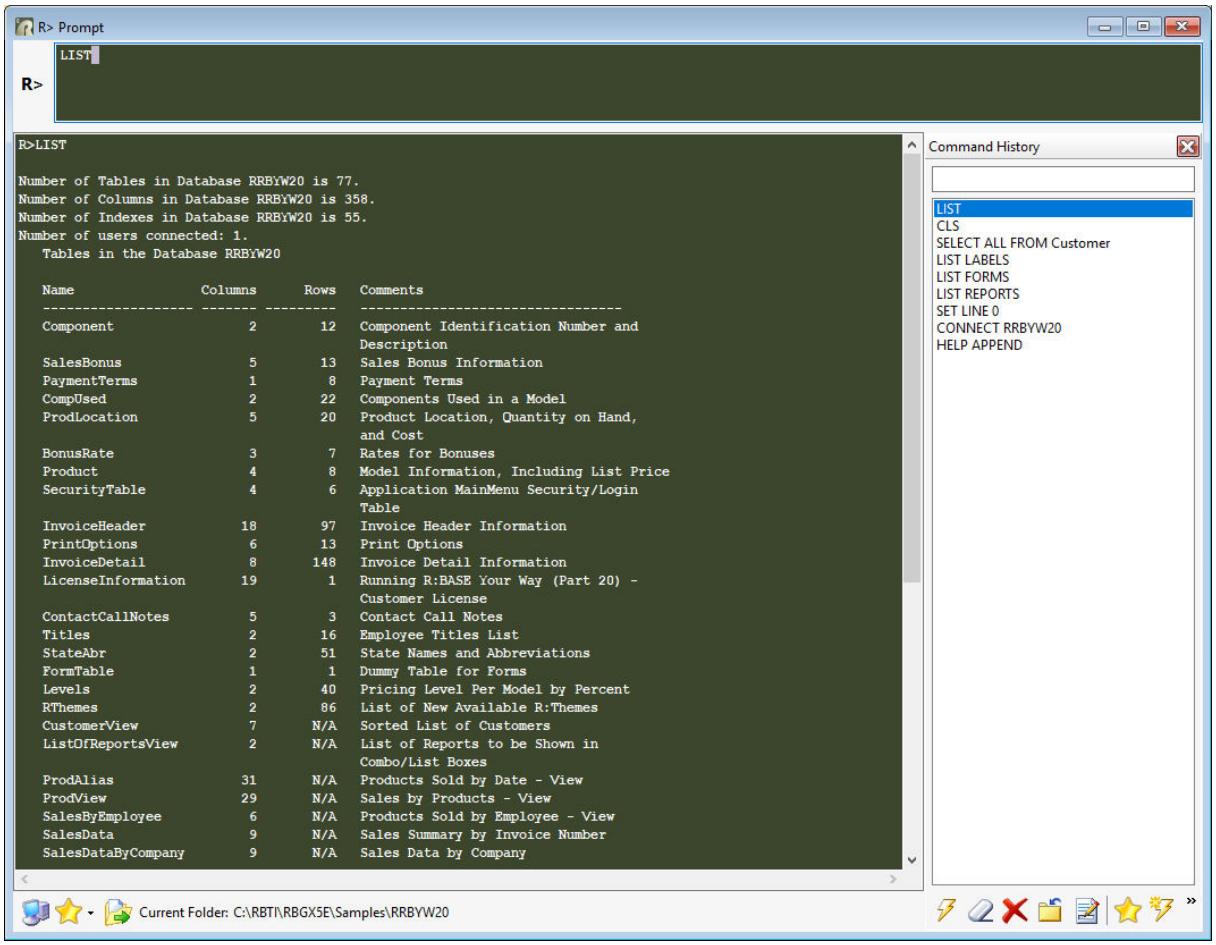
The R:BASE program starts and displays the R:BASE X.5 "window". As you work with R:BASE, other parts of the program are displayed in "windows" within the R:BASE X.5 "window". You can have many windows open at a time, and you can have many "R:BASE X.5" session windows open at a time.

As with all Windows programs, the R:BASE X.5 window can be maximized-set to encompass the entire screen; normalized-sized to be displayed on just part of the screen, or minimized-reduced to an icon. The windows displayed within the R:BASE X.5 "window" can also be maximized, normalized, and minimized. They always remain within the "R:BASE X.5" window. The title bar of the active window (the one in which you are working) is a different color than the title bars of the other windows. Refer to your Windows documentation for more information about working with windows.

The Database Explorer
The Database Explorer provides fast access to the features of R:BASE. From the Database Explorer you can create and modify databases, applications, tables, views, forms, reports, labels, and more.

The Database Explorer has three main parts: the Group Bar which is the panel on the left which shows options (e.g. Databases, Tables, Views, etc.) that correspond to the area in which you are working, the Work Space which is a list of existing objects for this area (such as a list of tables or a list of forms), and the Property Bar which displays your current folder location, folder options, and settings to alter your view of the objects in the Work Space. You'll use the Database Explorer options to start most of the lessons.

R:BASE Commands and Functions
R:BASE Commands and Functions provide access to the most powerful features of R:BASE. Rather than choosing a menu command, you enter an R:BASE command in the R> Prompt window, shown below:



The R> Prompt is an alternative way to use R:BASE-you can perform most activities using menu command and the Database Explorer. You don't have to be a programmer to use R:BASE commands,

we'll be using some of them in the lessons in this tutorial. For a complete description of R:BASE commands and their syntax, refer to the "Command Index" within the Inline Help [F1].

R:BASE built-in functions perform common programming tasks, such as finding the square root of a number or combining two strings. Specifically, a function is a predefined complex expression to which you pass one or more values and which returns a single value. R:BASE provides the following categories of commonly used functions:

- Arithmetic and Mathematical
- Conversion
- Date and Time
- Financial
- Keyboard and Operating System
- Logical
- String Manipulation
- Trigonometric
- User Defined (UDF's)

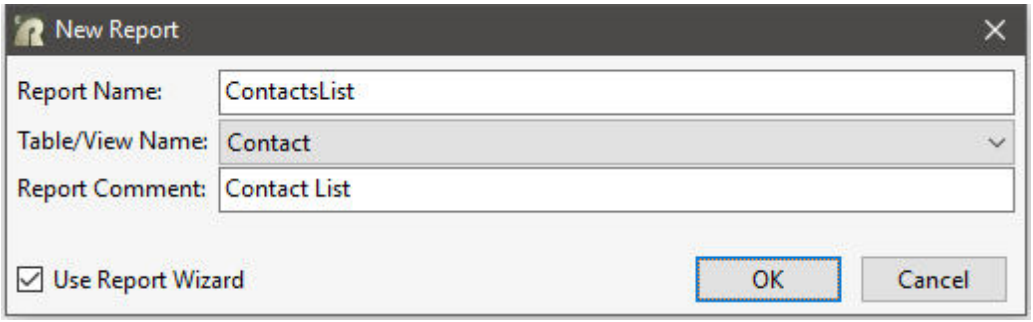
For a complete description of R:BASE functions and their syntax, refer to the "Function Index" within the Inline Help [F1].

Special Features
R:BASE displays several other features used to complete tasks. These include:

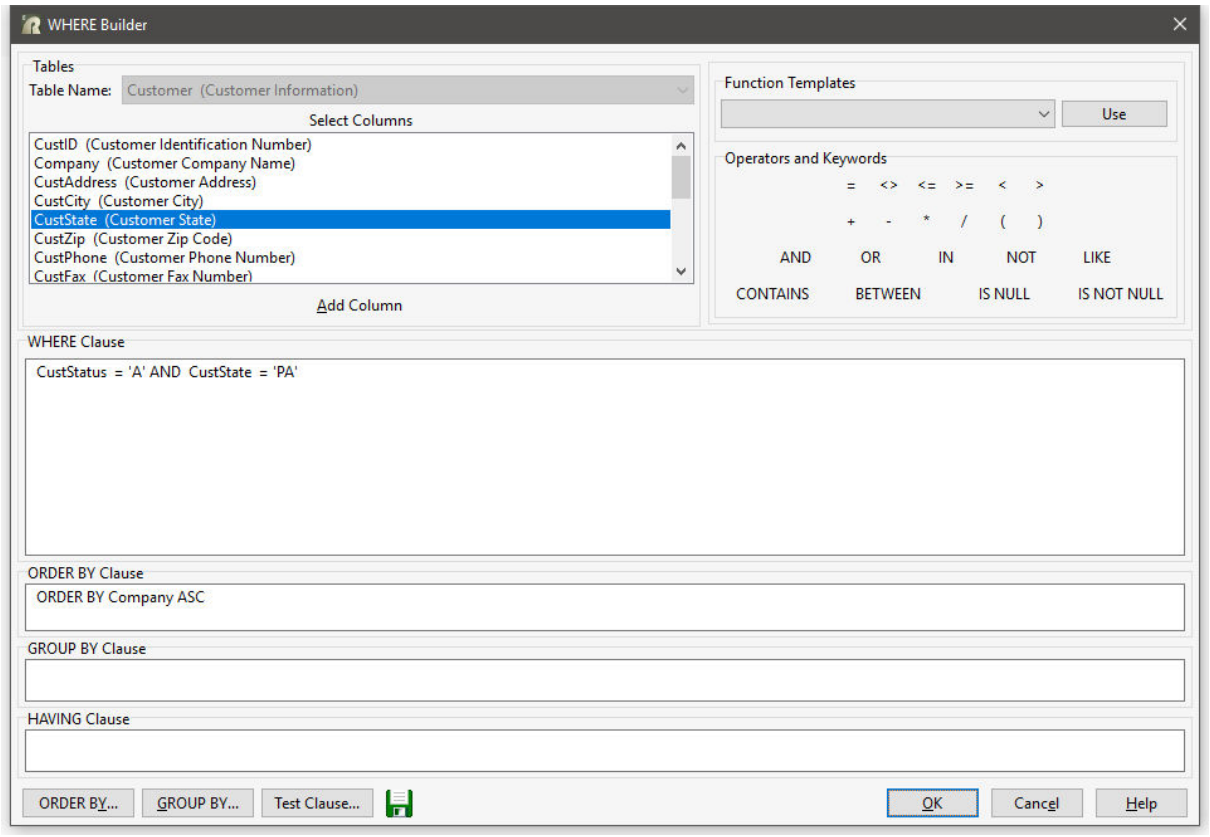
- Menus
- Dialog boxes
- Where Builder
- Hints
- Handlebars
- Lassoing

Menus
R:BASE X.5 provides a standard Windows menu. As you use R:BASE X.5, you'll notice that the available menus change depending on your current activity. For instance, if you are working with the Forms Designer, new menus appear, Database Controls, Variable Controls, and more. The options in each menu also vary accordingly. For more information about each menu and menu option, refer to online Help.

Dialog Boxes
R:BASE displays a dialog box when it needs more information about the work you are doing. For example, when building a new report you enter the necessary information about the report, such as report name and description, in a dialog box. Use the mouse or the [Tab] key to move between fields in a dialog box. You must complete the information in a dialog box before continuing on with the task.



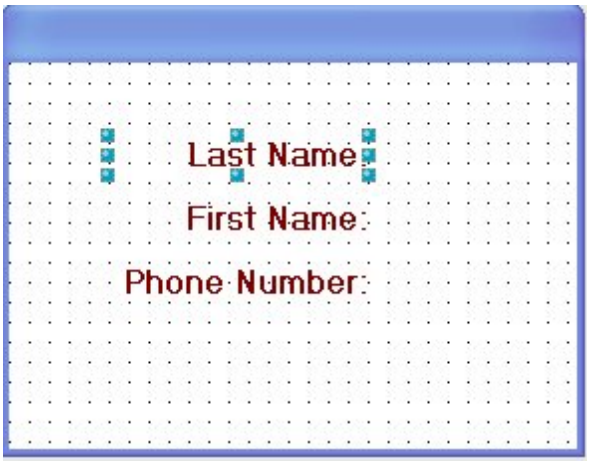
Where Builder
For several tasks, you can set conditions for the selection of rows from a table or view. When a task includes this process, R:BASE provides you with an aid called the Where Builder.

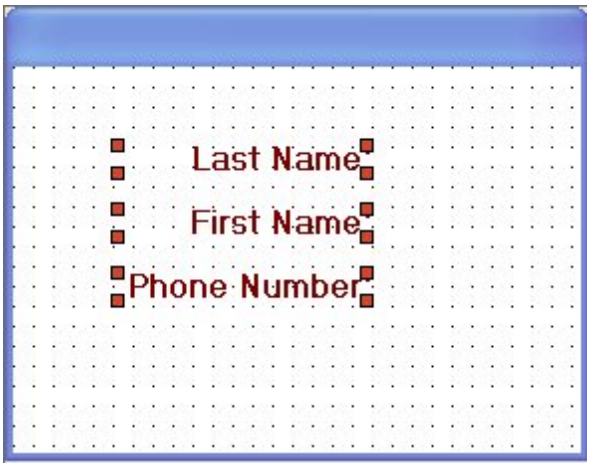


The Where Builder walks you through the process of defining one or more conditions. As you make each choice, R:BASE displays it so that you can see the condition as you're putting it together.

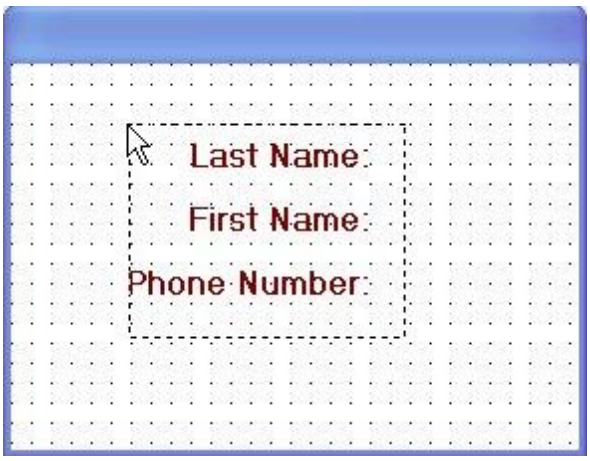
Hints
Small hint bubbles will appear above icon picture buttons throughout the R:BASE for Windows program when you hover your mouse cursor over them. These hints will inform you of the name of the button. The hints are especially helpful in the Form Designer, Report Designer, and Label Designer.

Handlebars
In the Form Designer, Report Designer, and Label Designer there are objects which can be placed. When a field object is selected, small blue circles appear around the four corners and four sides. These circles are called handlebars. When more than one object is selected, the red squares will be displayed. These handlebars are extremely useful for editing the properties of more than one object or moving more than one object. Hovering your cursor over the handlebars for an individual object changes the pointer to a double arrow. Holding down the left mouse button with the double arrow lets you stretch or shrink the field.





Lassoing
Within a form, report, or label, you can select multiple objects with the mouse cursor by lassoing objects. The lasso appears as a dotted line while you drag the mouse cursor. It works by starting in one corner of the work space, holding down the left mouse button, and dragging the cursor to the opposite corner. The lasso only needs to be touching the object in order to select it for editing. When lassoing multiple objects that are contained within a parent object, such as a panel, press and hold the [Ctrl] key to lasso the objects.



Documentation
This section discusses the documentation and conventions for this tutorial.

Manual Conventions
The following conventions identify specific types of information.

Convention	Example	Type of Information
Bold type	**File: Print**	Names of menus and menu commands
Italic type	*tblname*	Names of variables, arguments, options, names (columns, tables, databases, and so on) and documentation
All caps	RBENGINEX5.CFG	Names of commands, keywords, directories, and files
Bracketed words	[Page Up]	Names of keys on the keyboard
Courier font	`SET BELL ON`	Examples of R:BASE code
Quotation marks	"Print" dialog box	Names of dialog boxes, windows, and options

Mouse and Hot Keys
While working with R:BASE X.5, you have the choice of using a mouse or key combinations, hot keys, to choose menu commands and options. Procedures in the online Help generally describe how to perform tasks using a mouse. However, certain actions are only available by using a hot key. The various modules of R:BASE have specific hot keys that only work in that module, or have different results across the modules.

Getting Help
Getting help in R:BASE is easy: choose **Help: Contents**, press [F1], or click the Help button on the tool bar. Any of these options open the Inline Help. In addition, you can open the Inline Help specific to the task you are working on by pressing [Shift]+[F1] or by clicking the Help button in a dialog box. Inline Help opens in a separate window. Leave the Help window open for additional reference while you complete your task in R:BASE. The Inline Help is continuously being updated as new features and commands are implemented and when corrections are made.

The Inline Help includes the following types of information:

- How To section for each major portion of R:BASE
- Reference Topics for performing common tasks
- Reference commands for R:BASE commands, functions, and data types
- Description of tool bar buttons, hot keys, and menu commands
- Context-sensitive help describing R:BASE dialog boxes and windows

Additional Resources

1. PDF Documentation

- All_About_the_CHOOSE_Command.pdf
- All_About_the_DIALOG_Command.pdf
- All_About_the_PAUSE_Command.pdf
- All_About_the_PRINT_Command.pdf
- RBASE_X5_forWindows_GettingStartedGuide.pdf
- RBASE_X5_forWindows_DatabaseConversionGuide.pdf
- RBASE_X5_forWindows_QuickInstallationGuide.pdf
- Tutorial.pdf
- CommandIndex.pdf
- FunctionIndex.pdf
- ProgrammingInRBASE.pdf
- DatabaseMaintenance.pdf
- ProblemSolvingInRBASE.pdf
- GainingSpeedWithProgramming.pdf
- FormsManual.pdf
- HowToManual.pdf
- ProblemSolvingInRBASE.pdf
- ReferenceIndex.pdf
- UnderstandingDLCALL.pdf

- WhatsNewIn_RBASE_X5_forWindows.pdf

2. R:BASE X.5 for Windows - Sample Applications:

We have spent many hours to provide users with samples to demonstrate the use of each and every control and/or function in R:BASE X.5 for Windows. Please make sure to go through all the sample applications to explore the possibilities of R:BASE X.5. From your operating system desktop, you should see a RRBYW20 shortcut for running the "Running R:BASE Your Way Sample Application (Part 20)"

3. R:BASE X.5 Home Page: https://www.rbase.com/rbgx5

4. From The Edge: http://www.razzak.com/fte

5. R:BASE Support: https://www.rbase.com/support/

6. Training and Education: https://www.rbase.com/training/

7. More samples to demonstrate R:BASE features: http://www.razzak.com/sampleapplications

8. Online Syntax - General: https://www.rbase.com/support/rsyntax/

About this Tutorial
This tutorial introduces R:BASE to new users. The manual is designed for people who have a basic knowledge of their computer's operating system, but who have no experience creating R:BASE databases and applications. Those who have used R:BASE applications created by others or have worked with other database software will also find this manual useful.

What You Will Learn
Each lesson covers a single subject. Each lesson in this manual builds on previous lessons. You should complete them in order. When you have finished the lessons, you will have created a complete R:BASE database and application.

In each lesson you will learn how to save your work. It is important to save your work frequently to avoid losing changes, should your computer unexpectedly lose power.

The lessons in this manual teach you how to:

- Design a database
- Create a database and modify the structure
- Look at information in the database and create a view
- Create forms for data entry
- Create reports for printing data
- Create mailing labels
- Using R:BASE commands
- Writing command files
- Create and use an application
- Maintain the database

Sample Files

R:BASE X.5
The R:BASE X.5 installation creates a directory on your hard drive and copies some files from the disk into that directory. The directory is named "Tutorial" and is created under your C:\RBTI\RBGX5 directory. The Tutorial directory contains the following sample files:

- SAMPLE.RB1
- SAMPLE.RB2
- SAMPLE.RB3
- SAMPLE.RB4
- SKYDATA.BAK
- MyProg.rmd
- SAMPLE.RBA
- SKYWRITE.JPG

R:BASE X.5 Enterprise

The R:BASE X.5 Enterprise installation creates a directory on your hard drive and copies some files from the disk into that directory. The directory is named "Tutorial" and is created under your C:\RBTI\RBGX5E directory. The Tutorial directory contains the following sample files:

- SAMPLE.RX1
- SAMPLE.RX2
- SAMPLE.RX3
- SAMPLE.RX4
- SKYDATA.BAK
- MyProg.rmd
- SAMPLE.RBA
- SKYWRITE.JPG

Part II

2 Lesson 1 - Database Design

The lessons in this training manual will teach you the basics of R:BASE. We'll be using R:BASE to create a database and application for skyWrite Ink, our scenario for a business application. In the next few lessons, you will learn what a database is, and how to design and build one. When you have completed this course, you will have built the information system that helps skyWrite Ink.

2.1 Introducing SkyWrite Ink

In the late '90s, local businesses were looking for creative ways to advertise their products and services. Seeing this need, Amelia Lindhart and Charlie Earbergh formed skyWrite Ink, a skywriting company. skyWrite Ink provided companies with a cost-effective advertising method to reach large numbers of potential customers.

skyWrite has grown considerably from those early days. Amelia and Charlie have several salespeople who book the company's flights. Each year, skyWrite's planes log thousands of hours of flight time spelling slogans and messages above the city.

Currently, skyWrite keeps customer records and tracks flights schedules by hand. However, as the business has grown, so has the amount of paperwork that must be done.

A Business Information Problem
The problem at skyWrite is simple. The manual system that worked ten years ago can't keep track of all the company's customers and provide necessary information in a timely manner.

For example, Amelia wants to offer a discount to clients who have booked more than five flights in the previous six months. However, since she is also one of skyWrite's pilots, she doesn't have time to sort thorough the receipts from the last six months, organize them by customer, and then count the number of flights skyWrite has flown for each, which might take her more than a day to do.

Under the manual system, maintaining up the list of customer addresses is time consuming. And when Charlie wants to mail an update on skyWrite's services to customers, he can't use the list to automatically generate a set of mailing labels. He ends up hiring a temporary typist to address all the envelopes.

Amelia and Charlie have discovered that they can't get the information they want and can't easily use the information they have.

A Business Information Solution
To solve their information problem, Amelia and Charlie recently purchased R:BASE-a relational database management system. A database is a collection of information stored on a computer. A relational database, like R:BASE, organizes information so that it can be expanded, updated, and retrieved rapidly.

Using R:BASE, Amelia and Charlie can store all the information for skyWrite Ink's customers, employees, and flights. Then they can use that information in ways that will help them run the business more effectively. Amelia will be able to find out in a matter of minutes which clients used skyWrite's services most often. Charlie will be able to automatically generate a set of mailing labels based on customers' current addresses.

And because the information stored in the database can be used flexibly, they'll be able to answer other questions that arise about their business.

2.2 About this Lesson

Good database design is the cornerstone of an efficient and useful data management system. While designing a database for skyWrite, you'll become familiar with introductory level design concepts and techniques.

By the end of this lesson you'll understand:

- The need for good design.
- A process for designing a database model.
- Three basic relationships that exist among tables.

- How to properly identify relationships through the use of linking columns.

2.3 Database Design?

Information systems keep track of facts about a particular enterprise or business. These facts are used to produce a wide variety of output, providing management with vital data on which to base decisions. Therefore, it is important to implement a design that provides accurate and reliable information.

The design of a database determines the way data is stored and how different types of data relate. Regardless of the size of your database, the design should always be completed before you start building your database.

Although R:BASE is a powerful and easy-to-use tool, R:BASE can't design a database. Designing requires someone with an understanding of both the information that needs to be managed and the principles of database design that are covered in this lesson.

After you determine exactly what information needs to be stored and how it is to be retrieved, you're ready to design your database. The more carefully you design, the more effectively the database will meet your user's needs.

What can happen if a database is poorly designed?

- Data might be unreliable or inaccurate.
- Performance might be decreased.
- Flexibility might be lost.

Any of these problems can cause confusion or even inaccuracy for whomever uses your database. At the least, valuable time will be wasted as you attempt to use a system that is inefficient.

Although there are a number of rules you can follow in designing a database, the design process is only partly scientific; it also involves a degree of creativity that becomes more apparent once you have followed the design process a couple of times. You should follow these rules when at all possible, but not to the point where the database can't deliver the information and functions the user needs.

Begin by sketching a design on paper. There are advantages to creating a paper design first. Mainly, a paper design will help you see potential problems before you begin defining the database.

The first step is to identify the problem. skyWrite's problem is that the existing manual system cannot keep track of all their customers and provide necessary information in a timely manner. skyWrite needs a central location where their data can be stored and retrieved quickly.

Some of the facts they need to track are:

- How often a customer uses their services.
- Amount of income on a given day.
- Seasonal information such as the average number of flights for a given quarter.

Automating skyWrite's information will provide them with the ability to find data in a matter of minutes rather than days.

2.4 Good Database Design

The following are steps in designing a useful, efficient database, which are discussed in the following section:

- List the objects
- List the facts about the objects
- Turn the objects and facts into tables and columns
- Determine the relationships among objects
- Determine the key columns
- Determine the linking columns
- Determine the relationship constraints
- Evaluate the design

List the Objects
Make a list of all the objects, which are the themes or subjects about which you want to track information. For the skyWrite information system, there are three objects:

- Employees
- Customers
- Flights

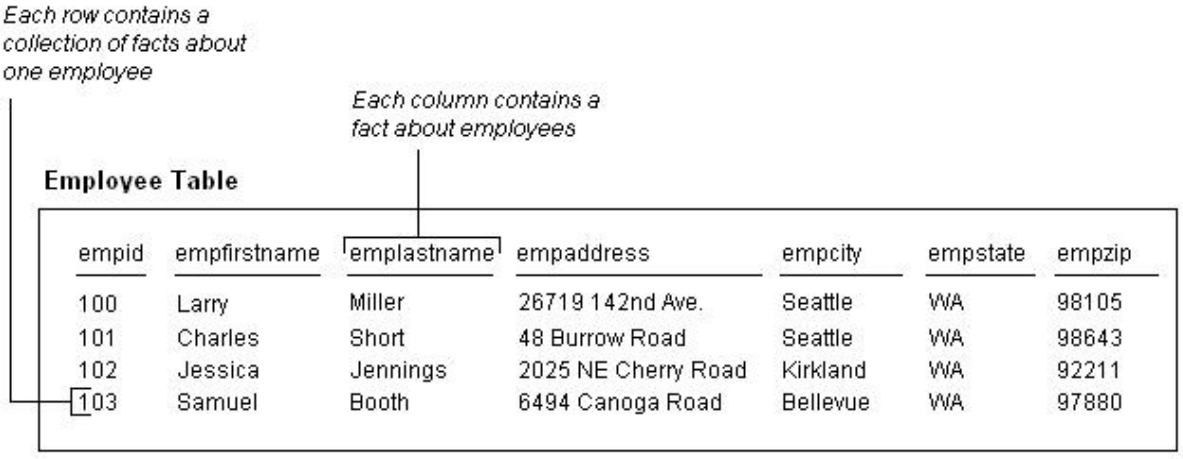
List the Facts about the Objects
Each object can have a great deal of information associated with it. List all of the facts, or information, about an object.

Objects	Important Facts about the Object
Employees	Employees first and last name, address, city, state, zip code, area code, and phone number
Customers	Customer first and last name, company, address, second address, city, state, zip code, area code, phone number
Flights	Date, slogan, number of letters, price per letter, total charge, place

Turn the Objects and Facts into Tables and Columns
In relational database design, objects become tables, and facts become columns. Each table contains information about a single object or event. Employees and customers are examples of objects. Performing a task like flying is an example of an event. Each of these objects or events becomes a separate table in the database.

Each table is made up of one or more columns. A column is a specific fact about the object or event; for example, an employee's last name is a fact about employees, or the date of flight is information about flights. Each group of facts is stored in a row, which is a collection of values that are unique to a particular instance for that table; for example, all the facts about one employee.

The object of the table shown here is employees; the facts are individual pieces of information about employees.



When turning facts into columns, you need to consider two things:

The column name - When naming a column, use a name that describes the data the column will hold. A column name can be up to 18 characters long in R:BASE X.5. A column name can be up to 128 characters long in R:BASE X.5 Enterprise. It must begin with a letter, and can contain letters, numbers, and the symbols # and _. R:BASE reserved words cannot be used as column names. For a complete list, see the entry "Reserved Words" in the R:BASE Help "Reference Topics" under "Conversion Issues and Considerations"

The column's data type - The type of data you'll store in a column determines the column's data type (the available data types are described below). Determining the column's data type now will make creating the database easier. Use the list below to determine the correct data types for your columns.

Data Type	Description
BIGINT	Holds a 64-bit integer value. Delimiters (such as commas) cannot be used in entry. No length is needed. Offers a range of ±999,999,999,999,999,999. Specific to R:BASE X.5 Enterprise
BIGNUM	Double precision real numbers whose precision and scale can be set. Set the precision (number of digits) from 1 to 38. Set the scale (number of decimal places) from 0 to any positive number up to the precision value. The default is precision 18, scale 0. The data type is specific to R:BASE X.5 Enterprise.
BOOLEAN	Hold true/false values. Internally stored as 0 for false and 1 for true. Accepted values for false include: 0, false, 'false'. Accepted values for true include: 1, true, 'true'. Specific to R:BASE X.5 Enterprise
BSTR	Holds binary string data. String data type that is used by COM (Component Object Model), Automation, and Interop functions. Used to support Unicode in table data. Composite data type that consists of a length prefix, a data string, and a terminator. The data type is specific to R:BASE X.5 Enterprise.
CURRENCY	Monetary values of 1 to 23 characters, with 16 digits represented in the currency format based on the database CURRENCY setting. Currency amounts are in the range of $99,999,999,999,999.99. If no decimal point is included, .00 is assumed. To view the CURRENCY settings, select "Settings: Configuration Settings" from the Menu Bar.
DATE	Date values of 1 to 30 characters represented in the month, day, and year format. Dates can range from 1/1/3999 BC to 12/31/9999 AD. The default display is MM/DD/YYYY. To view the DATE settings, select "Settings: Configuration Settings" from the Menu Bar.
DATETIME	A concatenations of the DATE and TIME data types, resulting in a sequence and display format as set by DATE and TIME. DATETIME cannot be set directly, but only through the DATETIME function.
DOUBLE	Double-precision real numbers in the range of (+or-)10(+or-)^{308} with a precision of 1 to 15 digits.
GUID	Binary global unique identifier to store unique values, which is represented as a 32-character hexadecimal string. An example of a GUID value is: **8C20005C-0E2A-47E0-B2BE-57E67961628B** As the GUID data type is a binary value, it will increase retrieval of data from tables for indexed columns. This data type is specific to R:BASE X.5 Enterprise.
INTEGER	Whole numbers in the range of (+or-)1,999,999,999. You can't use delimiters (commas) when entering these numbers.
NOTE	Variable length text of up to 4,092 characters. You can't set the length for a NOTE column. The actual length of each NOTE entry is determined by the data contained in each row of the NOTE column.
NUMERIC	Double precision real numbers whose precision and scale can be set. Set the precision (number of digits) from 1 to 15. Set the scale (number of decimal places) from 0 to any positive number up to the precision value. The default is precision 9, scale 0.
REAL	Real-number amounts in the range of (+or-)1O^{38} with 6-digit accuracy. Real numbers with up to 6 digits are displayed as a decimal number; for example, 321.414. Real numbers with more than 6 digits are represented in scientific notation; for example, 9.8E32.
SMALLINT	Holds a 16-bit integer value. Delimiters (such as commas) cannot be used in entry. No length is needed. Offers a range of ±32767. Specific

	to R:BASE X.5 Enterprise
TEXT	Alphanumeric data with a maximum length of 1,500 characters. You can set the length for a TEXT column. The default is 8 characters.
TIME	Time values of 1 to 20 characters represented in the hours, minutes, and seconds format set for the database. The default display is HH:MM:SS. Time can be specified up to thousandths of a second. Time can be displayed or entered as a 12- or 24-hour clock.
VARBIT	Holds binary data from a file. You must set the length in bits; if you set the maximum length to a value over 262,136 (32,767 bytes), the limit is ignored and defaults to 8*256 MB. You can use the LONG VARBIT data type to automatically set the maximum length to 8*256 MB.
VARCHAR	Holds alphanumeric data from a file. You must set the length in bits; if you set the maximum length to a value over 262,136 (32,767 bytes), the limit is ignored and defaults to 8*256 MB. You can use the LONG VARCHAR data type to automatically set the maximum length to 8*256 MB.
WIDENOTE	Holds Unicode data. The default length is 0, where the length is determined by the data. Holds variable length text of up to 4,092 characters. Maximum length of a WIDENOTE column can be set. Indexes and constraints are allowed on WIDENOTE data types. R:BASE reserves a minimum of four bytes of internal storage space, with 2 bytes per character. Specific to R:BASE X.5 Enterprise
WIDETEXT	Holds Unicode data. The default length is eight characters. The maximum is 1,500 characters. Maximum length of a WIDETEXT column can be set. R:BASE reserves a minimum of four bytes of internal storage space, with 2 bytes per character. Specific to R:BASE X Enterprise.

The following is a list of the preliminary tables, columns, and data types for the ***sky*** database:

Employee

Column name	**Data type**
empfirstname	TEXT 25
emplastname	TEXT 25
empaddress1	TEXT 40
empaddress2	TEXT 40
empcity	TEXT 25
empstate	TEXT 2
empzip	TEXT 10
empareacode	INTEGER
empphone	TEXT 8

Customer

Column name	**Data type**
custfirstname	TEXT 25
custlastname	TEXT 25
company	TEXT 30
custaddl1	TEXT 40
custadd2	TEXT 40
custcity	TEXT 25
custstate	TEXT 2
custzip	TEXT 10
custareacode	INTEGER
custphone	TEXT 8

Flights

Column name	**Data Type**
flightdate	DATE

slogan	TEXT 20
letters	INTEGER
price	CURRENCY
totalcharge	CURRENCY
place	NOTE

At this stage in the design, it helps to draw a simplified sketch of the database. For example, you can draw one box for each table. In later steps you can fill in key columns and draw the relationships between the tables.

Determine the Relationships Among the Objects

Using the simplified sketch above, look at the tables and determine how they might be related.

- Are employees related to flights? Yes, they are; employees schedule the skywriting flights. Therefore, draw a line connecting the *Employee* and *Flights* boxes.

- Are customers related to flights? Yes; there wouldn't be any flights if a customer did not order some skywriting. So, draw a line connecting the *Customer* box and the *Flights* box.

- Are employees related to customers? They might be, but in this case they are only related when a flight is scheduled. Therefore, you would not directly relate *Employee* with *Customer.*

You now know how each object or table is related; next determine the type of relationship. There are three types of relationships among tables: one-to-one, one-to-many, and many-to-many. They are defined as follows:

- **One-to-one relationships**

For any given row in Table A, there is only one row in Table B. For any given row in Table B, there is only one row in Table A. Usually, all the columns are in one table for this relationship.

- **One-to-many relationships**

For any given row in Table A, there are many rows in Table B. For any given row in Table B, there is only one row in Table A. Usually, two tables are used linked by a common column with values that match across the tables. The common column is known as the key column.

- **Many-to-many relationships**

For any given row in Table A, there are many rows in Table B. For any given row in Table C, there are many rows in Table B. Usually, three tables are used in this relationship with two of the tables using a common column in relation to a third. These two tables do not link directly to each other; instead, each of these link to a third table in a one-to-many relationship by a common key column with values that match.

The next step in determining the types of relationships among tables is to list every table and how it relates to any others. For example:

- *employee* is related to *flights*
- *customer* is related to *flights*

The relationship between the *employee* and *flights* tables is a one-to-many relationship. That is, if you look at a single row in the *employee* table, there should be many rows of data in the *flights* table for that employee;one employee can schedule many flights. If you then look at a single row in the *flights* table, it will point to only one employee in the *employee* table; a flight is scheduled by one employee.

Perhaps your reaction now is to say that employees are related to customers, too, and therefore the *employee* table is related to the *customer* table. The first statement--employees are related to customers-is true, but only when a flight occurs. All flights are recorded in the *flights* table, and therefore it is the *flights* table that shows the pertinent relationship.

An easy way to identify the type of relationship is to ask whether a specific row in Table A can point to (link with) one or many rows in Table B, then reverse the table order and ask the question again.

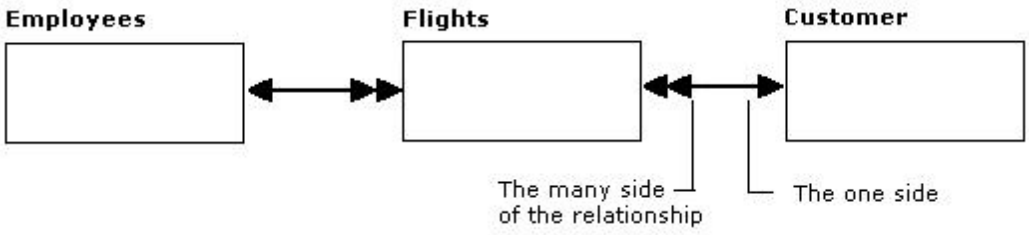
- A customer can point to one or many flights? Many.
- A flight can point to one or many customers? One.

The relationship between the *flights* table and the *customer* table is one-to-many.

- An employee can point to one or many flights? Many.
- A flight can point to one or many sales employees? One.

The relationship between the *flights* table and the *employee* table is one-to-many. There are no one-to-one or many-to-many relationships in the *sky* database.

You can represent these relationships in your sketch.



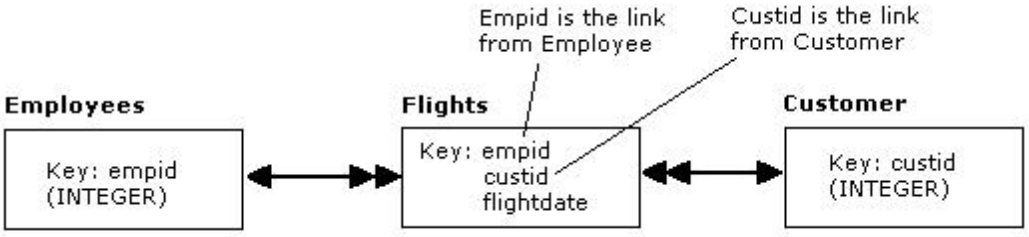
Determine the Key Columns

A key column is one that contains a unique value for every row in the table. Key columns are used to ensure the accuracy of your data and improve the speed of your database's performance. The value in a key column can be an account number, social security number, part number, license number, or any other numeric value or combination of characters that will be different for each row in a table.

One good example of a key is an employee identification number. Sometimes the number is a social security number; sometimes it is another unique number assigned by the payroll department. There is a direct relationship between an employee's identification number and the rest of the data about that employee. If you know an employee's identification number, you can easily find his or her name, address, and other information.

A column is effective as a key only if no two rows in the table have the same value in that column. You can use a company name as a key if it is universally unique. However, if there is any possibility that another company could have the same name, then it is not unique and must not be used as a key column. Do not use any column as a key when the possibility of a duplicate exists.

Most of the tables in the *sky* database use an integer value as the key value. The key to the *flights* table consists of three columns: *empid, custid,* and *flightdate.* You must include *flightdate* because *empid* and *custid* alone do not identify an unique row. For example, *customers* can book many flights. Including *flightdate* in the key allows you to always identify a particular row.



Determine the Linking Columns

Remember that relational database models deal with data in the form of tables, and each of those tables represents a single object or event. The foundation of a relational database is that these tables relate to one another through common data contained in linking columns. Linking columns allow data from two or more tables to be retrieved and manipulated at the same time.

The best linking columns are those that will point to a unique row of data in the table. Since a key column is one that contains only unique values, it makes a good linking column. Linking columns are also called common columns.

For example, by knowing that the key column for the *employee* table is *empid,* and knowing that you want to track the employee who scheduled the flight, you can include *empid* as a column in the *flights* table. Once *empid* is in the *flights* table, you can look at any row in that table and, knowing the *empid* values, determine the employee who is associated with that flight.

Where you place the links depends on the type of relationship between the tables. Once you know the type of relationship, it is easy to determine where to place the linking column.

The following rules will guide you in placing the links:

One-to-one relationships-The link should either be the most stable column or be taken from the table where the key column is created. The most stable column is the one in which the data is least likely to change.

One-to-many relationships-The linking column should come from the table on the one side of the relationship. The key column from the *employee* table (one side) should be placed in the *flights* table (many side). Similarly, one customer is associated with many flights. The key column from *customer* (one side) should be placed in the *flights* table (many side).

Determine the Relationship Constraints
Often the answers you get from a database come from more than one table. For example, if you want to know who the salesperson was for a particular flight, you determine the name by using the value in *empid* to look up the correct row in the *employee* table. The question of who the employee is can be answered only if there is a row in the *employee* table with a number in *empid* corresponding to the number in the *flights* table. Similarly, the question of who the customer is can be answered only if there is a row in the *customer* table with a number in *custid* corresponding to the number in the *flights* table.

To make sure that your database model works, you need to specify that no row can be added to the *flights* table unless there are already corresponding rows in the *employee* and *customer* tables. These requirements are known as relationship constraints.

Note: R:BASE enforces relationship constraints by implementing primary and foreign keys, form design, and rules. Primary and foreign keys are discussed in Lesson Two. Form design is discussed in Lesson Four, and rules in Lesson Two.

Evaluate the Design

The last step in the design process is evaluating the design. Look for any design flaws that could cause the data to be unreliable or redundant.

You can evaluate every table by asking the following questions:

- Does each table have a single theme?
- Does each table have at least one key column?
- Is the table easy to add data to and retrieve data from?
- Are the columns unique to the table?

To evaluate the *customer* table, for example, you might take the following approach:

- The table has a single theme: customers.
- The table has a key: *custid.*
- The table is easy to use.
- Each column name is unique to this table except for *custid,* which is the linking, or common column, in the *flights* table.

You can evaluate the *flights* table in the same way:

- The table has a single theme: scheduled flights.
- The table has a key: *empid* + *custid* + *flightdate.*
- The table is easy to use.
- Each column name is unique to this table except for the key columns used as the links from the *customer* and *employee* tables.

2.5 Final Design

The following is a list of the final tables and columns for the *sky* database, including linking columns. An asterisk (*) denotes a key column.

Employee

Column name	Data type
**empid*	INTEGER
empfirstname	TEXT 25
emplastname	TEXT 25
empaddress	TEXT 40
empcity	TEXT 25
empstate	TEXT 2
empzip	TEXT 10
empareacode	INTEGER
empphone	TEXT 8

Customer

Column name	Data type
**custid*	INTEGER
custfirstname	TEXT 25
custlastname	TEXT 25
company	TEXT 30
custaddl1	TEXT 40
custadd2	TEXT 40
custcity	TEXT 25
custstate	TEXT 2
custzip	TEXT 10
custareacode	INTEGER
custphone	TEXT 8

Flights

Column name	Data Type
**empid*	INTEGER
**custid*	INTEGER
**flightdate*	DATE
slogan	TEXT 20
letters	INTEGER
price	CURRENCY
totalcharge	CURRENCY
place	NOTE

Congratulations! Your design is now complete. In the next lesson you will learn how to implement the *sky* database design in R:BASE.

2.6 Summary

In this lesson, you have learned about database design and the important role it plays in creating a good data management system. Keep these points in mind when you design your databases:

- Relational databases store data in tables. Tables contain data about a single subject and consist of columns, which represent the facts about the subject. A single row of data pertains to a single instance in that table; for example, employee number 100 will reference only one employee in the *employee* table.

- Key columns are used to ensure the accuracy of your data and improve the speed of your database's performance. Key columns uniquely identify a single row of data in a table and establish common or linking columns with other tables.
- There are three types of relationships among tables: one-to-one, one-to-many, and many-to-many.

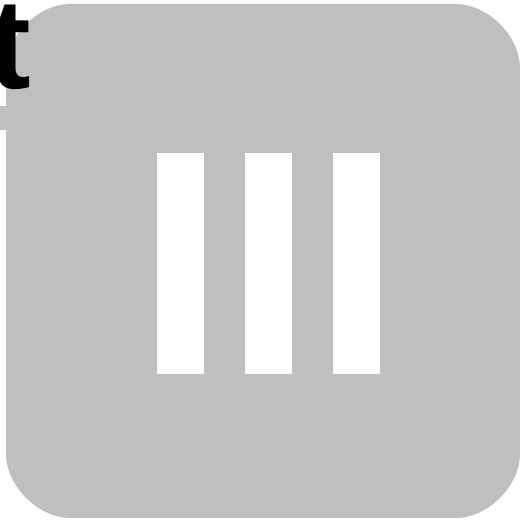
To check your knowledge, try answering the following questions:

1. What do objects become in a database model?
2. What do facts become in a database model?
3. What is a key?
4. When a one-to-many relationship exists between two tables, from which table do you take the key and place it in the other table to provide the link?

   (a) Take primary key from the table on the one side and place it in the table on the many side.

   (b) Take primary key from the table on the many side and place it in the table on the one side.
5. What effect does poor database design have on a database?
6. How many rows should you have in the *employee* table for each employee?
7. In a well designed database can tables contain data about multiple subjects? (Yes- No)

Part



3 Lesson 2 - Defining a Database

In this lesson, you will define the *sky* database and its three *tables--employee, customer,* and *flights*- which you designed in Lesson One.

At the end of this lesson you will be able to:

- Start the Database Designer
- Define a database
- Define tables and include table descriptions
- Designate auto-numbered columns
- Define columns and include column descriptions
- Choose a data type for a column
- View database information
- Defining primary and foreign keys
- Designate indexed columns
- Add a computed column to an existing table
- Define rules
- List rules
- Modify rule messages
- Test the changes you made to the database

Now that you understand what information Amelia and Charlie need to track for skyWrite Ink and have designed a database for that information, you are ready to create the database. We'll use the R:BASE Database Designer to define the database. You can define new tables or modify existing tables by adding deleting, and modifying column definitions.

3.1 Creating a Database

R:BASE stores data in tables. Each table holds data for one particular subject. Before you begin entering the data you want stored, you define the database by naming it. You then name a table and define its columns. One by one, you continue naming tables and defining columns until you have defined the complete database.

The following sections show you how to define the *sky* database with its three tables and their columns.

To build a database:

1. From the Database Explorer, choose the "Databases" option in the Group Bar and select "New Database".

R:BASE displays the "New Database" dialog box. When you create a database, the first task is to name it. Database names can be up to 8 characters long in R:BASE X.5. Database names can be up to 128 characters long in R:BASE X.5 Enterprise.

2. Enter "**sky**" in "Database Name". At this point you can also specify the Owner user name for the database to control user access to the database. For this sample, leave the Owner name option unchecked.

3. Click the OK button.

R:BASE displays the Database Designer-the "Data Designer" window and you will be prompted for a table name.

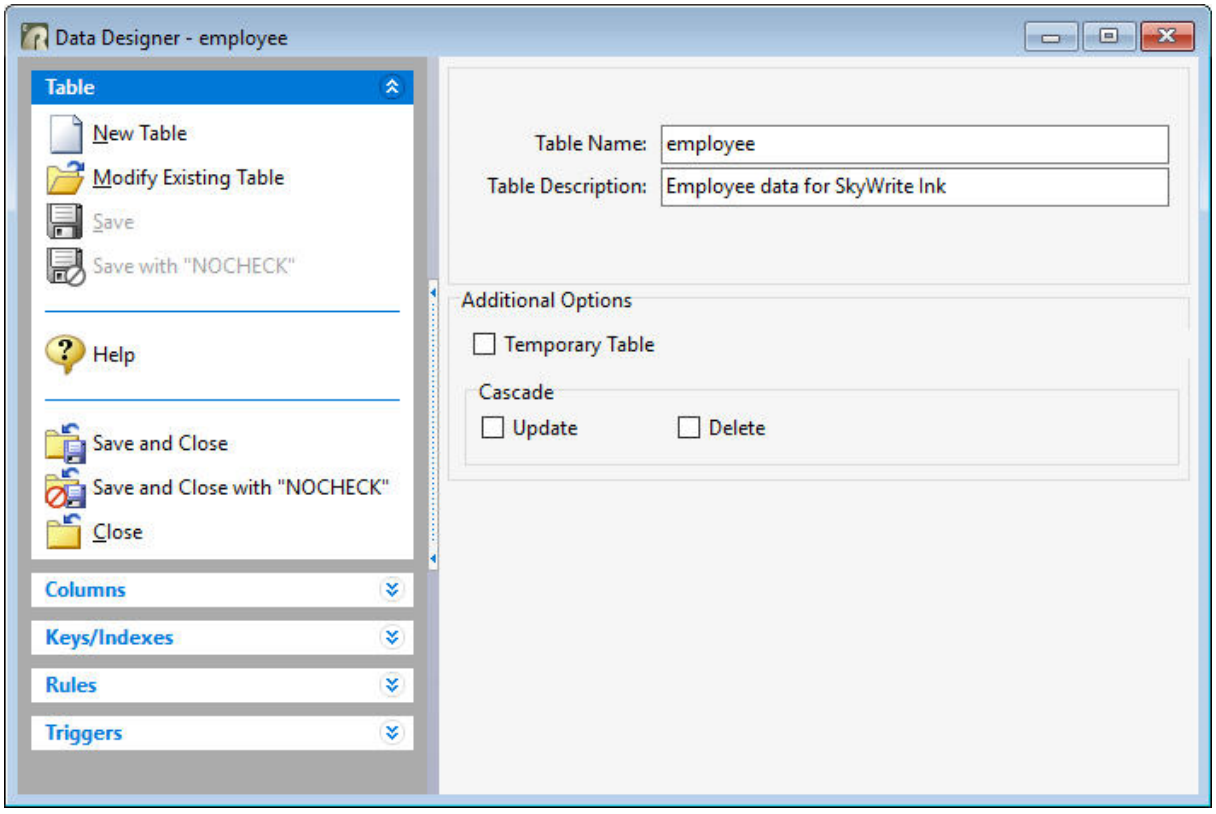
Defining Tables
Next, we'll define the three tables that are in the *sky* database. The actual number of tables and columns that can be defined in a database may vary based on the amount of available memory. When a database is created, R:BASE automatically creates some tables and columns to hold system information called System Tables.

The Employee Table

The first table we'll define is the *Employee* table, the table we designed to hold the data for employees. When defining a table, fill in the table name and description, and then define the columns that belong in the table.

To define the employee table:

1. If this is the first table in a new database, the Data Designer will launch automatically for a new table name. Otherwise, from the Database Explorer, you would choose "Tables" and "New".
2. Enter "employee" in "Table Name:". A table name can be as many as 18 characters long in R:BASE X.5. A table name can be as many as 128 characters long in R:BASE X.5 Enterprise.



3. Enter "Employee data for skyWrite Ink" in "Table Description."
4. Click the "Columns" option.

Here you enter the name, data type, description, and other column attributes. You first name the column.

5. When adding a column to a table for the first time, the "Enter New Column Name" dialog will appear. Otherwise, for an existing table, you would select the "Add Column" button. In the "Data Designer" window, define the columns for the table:

- For the first column, enter "empid".
- Select a data type for the column. We determined the data type for the *empid* column in Lesson One. Since the column contains whole numbers only (no fractions or decimal values), we decided on an INTEGER data type for the column. Select "INTEGER". By pressing the [Enter] key while the data type column is highlighted or double-clicking on the data type option will allow a drop down box to appear.

- Select "AutoNum". When a column is auto-numbered, R:BASE will assign a new, higher number to the column for each row that is added to the table. R:BASE allows you to assign the first number in the auto-numbering sequence and to set the amount by which the number will increase.

- Enter "107" in "Initial Value". We will be loading the tables in the database with data at the end of this lesson. Because there are already seven employees whose id numbers run from 100 through 106, we want the next employee added to the table to have the employee id 107 in the *empid* column.

- Leave "Increment Value" as 1. Each new row added to the table will have a number that is one higher than the previous number.

- In "Description", enter "Employee identification number" as the description for the *empid* column.

  This completes the definition of the first column in the table.

6. Select the "Add Column" option and enter "empfirstname" in the next row as the name of the second column.

- Select "TEXT" in the "Data Type" column.

- Enter 25 to set the column length. For TEXT data types, R:BASE asks you to define the length of the column. The default length for the TEXT data type is 8 characters. However, an employee's first name might be longer than this, so we'll allow as many as 25 characters to be entered into the column. Press the [Enter] key to edit the 8 to 25. The press [Enter] again to save the change.

- Enter "Employee first name" in "Description".

Now define the remaining eight columns for the *employee* table. The information you need to define them is shown below. Column descriptions and default values are optional in R:BASE. If you don't want to include them, leave "Description" and/or "Default" blank.

Note: The Data Designer interface is set up to allow the insertion of multiple columns while defining the column name, data type and, if applies, the length. Using the [Tab] key, you can move left to right and top to bottom adding new columns. After leaving the length field using [Tab], R:BASE will prompt you for a new column name. The values for default value and description can be added after all the column values are stored.

Column Name	Data Type	Length	Default	Column Description
emplastname	TEXT	25		Employee last name
empaddress	TEXT	40		Employee address
empcity	TEXT	25		Employee city
empstate	TEXT	2	WA	Employee state
empzip	TEXT	10		Employee zip code
empareacode	INTEGER	N/A		Employee area code
empphone	TEXT	8		Employee phone number

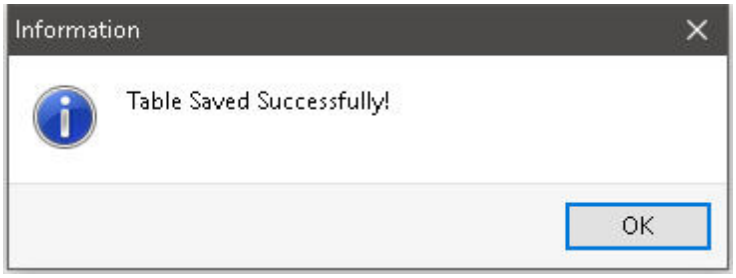
Based on your screen's resolution settings, R:BASE will display around 20 column definitions at a time. A scroll bar is displayed on the right side when more columns are defined than fit on the screen. You can also use the arrow keys to move the cursor within the "Column Name", "Data Type", and "Length" column lists. In the column lists, another column can be added or a defined column can be modified. Use the mouse to select a column under "Column Name" to view or modify previously defined columns. To

delete the selected column, click the "Delete Column" button. To insert a column, click the "Insert Column" button. R:BASE inserts the new column above the highlighted column.

Caution: If an entry contains an error, you can correct it by typing over the error.

7. Click the "Save" button to store the *Employee* table in the database.

R:BASE displays a message confirming that the table was saved successfully.



8. Click the "Close" option to close the Data Designer.

You have finished defining the first table in the *sky* database.

The Customer Table

Next, we'll define the *customer* table, the table we designed to hold data for customers. We follow the same process as for the *employee* table.

To define the customer table:

1. From the Database Explorer, choose "Tables", and select "New".
2. Enter "Customer" in "Table Name."
3. Enter "Customer data for skyWrite Ink" as the table description.
4. Click the Columns button.

You are ready to define the columns in the *customer* table. Define the columns just as you did when you defined the columns for the *employee* table.

5. Enter "custid" in the first row under "Column Name" to start defining columns.

Custid is also an autonumbered INTEGER column. Define it using the same procedure as in Step 5 in the previous procedure for *empid* in the *employee* table. For *custid,* make the initial value 16. The increment is 1.

6. Enter a description, "Customer Identification number".
7. Define the rest of the columns in this table.

Column name	Data type	Length	Default	Description
custfirstname	TEXT	25		Customer first name
custlastname	TEXT	25		Customer last name
company	TEXT	30		Customer company
custadd1	TEXT	40		Customer first address
custadd2	TEXT	40		Customer second address
custcity	TEXT	25		Customer city
coststate	TEXT	2	WA	Customer state
custzip	TEXT	10		Customer zip code
custareacode	INTEGER	N/A	206	Customer area code
custphone	TEXT	8		Customer phone number

8. Once you have defined all the columns in customer, click the the "Save" option.
9. Then, confirm that the changes were saved, and select the "Close" option.

You've finished defining the second table in the *sky* database. Now we'll define the *flights* table.

The Flights Table

The last table we'll define for *sky* contains information on the flights made by skyWrite Ink. The table is linked to the two tables we've already created by including the columns *empid* and *custid.*

To define the flights table:

1. From the Database Explorer, choose "Tables", and select "New".
2. Enter "flights" in "Table Name."
3. Enter "Flight data for skyWrite Ink" as the table description.
4. Click the Columns option.

You are ready to define the columns in the *flights* table. Define the columns just as you did when you defined the *employee* and *customer* tables.

5. Enter "custid" in the first row under "Column Name" and press [Enter].

R:BASE displays the INTEGER data type for the column. R:BASE knows that the column already exists in the *customer* table, and automatically supplies the correct data type. This is part of R:BASE's relational capability; columns cannot have the same name and different data types.

6. Enter "Customer booking flight" in "Description."

7. Now define the rest of the columns that make up the table just as you did when you defined columns for the previous two tables.

Column name	Data	Length	Default	Compute	Column Description
empid	INTEGER	N/A			Employee selling the flight
flightdate	DATE	N/A			Date of the flight
slogan	TEXT	20			Slogan to be written
letters	INTEGER	N/A			Number of letters in slogan
price	CURRENCY	N/A			Price per letter
place	NOTE	N/A			Where to write slogan

The *place* column has a NOTE data type. NOTE columns allow you to enter information whose length can be quite long, up to 4,096 characters. We've made *place* a NOTE column so that the employee who booked the flight has plenty of room to enter special instructions or other important information concerning the flight.

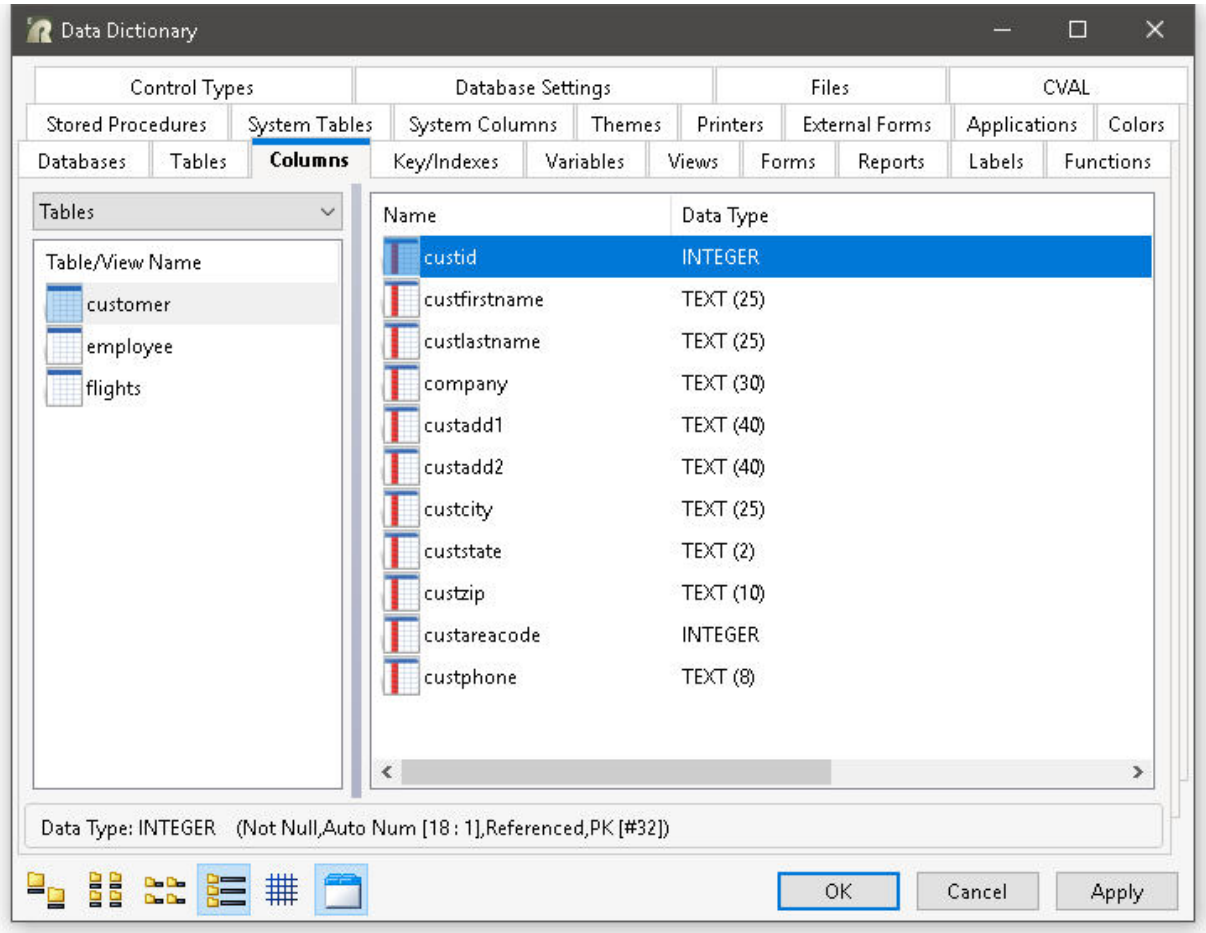
8. Once you have defined all the columns in flights, click the the "Save" option.
9. Then, confirm that the changes were saved, and select the "Close" option

You've finished defining all the tables in the *sky* database.

Viewing Database Information
You now have a database named *sky* with three tables: *employee, customer,* and *flights.* The database contains 27 columns. When you are working with your database, you can view information in the tables and columns in the database at any time using the Data Dictionary.

To look at information in the sky database:

1. From any area of R:BASE, press [F3] or from the Menu Bar select "Tools", then Data Dictionary.



R:BASE displays the "Data Dictionary" dialog box. Use the dialog box to show information about your database. The tabs across the top lists the types of database information you can display. When the "Tables" tab is selected and shows three tables in the middle box. The tables are listed in alphabetical order. The display in the Data Dictionary can be changed on your preferences using the four buttons on the bottom left corner. Choose the fourth button on the right.

2. Select "customer".

The table description is displayed in the right panel of the dialog box.

Using the Data Dictionary, you can also see the names of the columns that are in each table.

3. Select "Columns" tab and select the specific table.

The *customer* table is automatically highlighted and R:BASE displays a list of the columns in that table and column the description. The columns are displayed in the order they were created.

4. Select "custlastname".

The data type and column length are all displayed when you select a column name.

5. Click the Cancel button to return to the "Database Explorer" window.

3.2 Defining Primary and Foreign Keys

When we designed the database in Lesson One, we decided that the *empid* and *custid* columns would be key columns. They are the columns that enforce the relationship constraints between the tables. When flights are entered, we want to make sure that the *empid* and *custid* values exist in the *employee* and

customer tables. We use R:BASE's built-in constraints to enforce the relationships between our tables. In addition to enforcing the relationships between tables, by defining constraints, we automatically create indexes on the columns to speed data retrieval. There can be multiple primary keys in one table as well as multiple foreign keys in a table.

The *employee* table has a one-to-many relationship with the *flights* table, using the column *empid,* as there can be only one unique employee in the employee table, and there can be many flights entered in the *flights* table by an employee. The *empid* column would be the primary key for the *employee* table and *empid* would be the foreign key in the *flights* table.

The *customer* table also has a one-to-many relationship with the *flights* table, using the column *custid,* as there can be only one unique customer in the *customer* table, and there can be many flights entered in the *flights* table for a customer. The *custid* column would be the primary key for the *customer* table and *custid* would be the foreign key in the *flights* table.

First, we'll make the *empid* column in *employee* a primary key. A primary key is used to identify the columns that uniquely identify a row in a table. A primary key must be a unique value. The difference between a primary key and a foreign key is that the primary key data must be unique to that table and the foreign key can occur more than once.

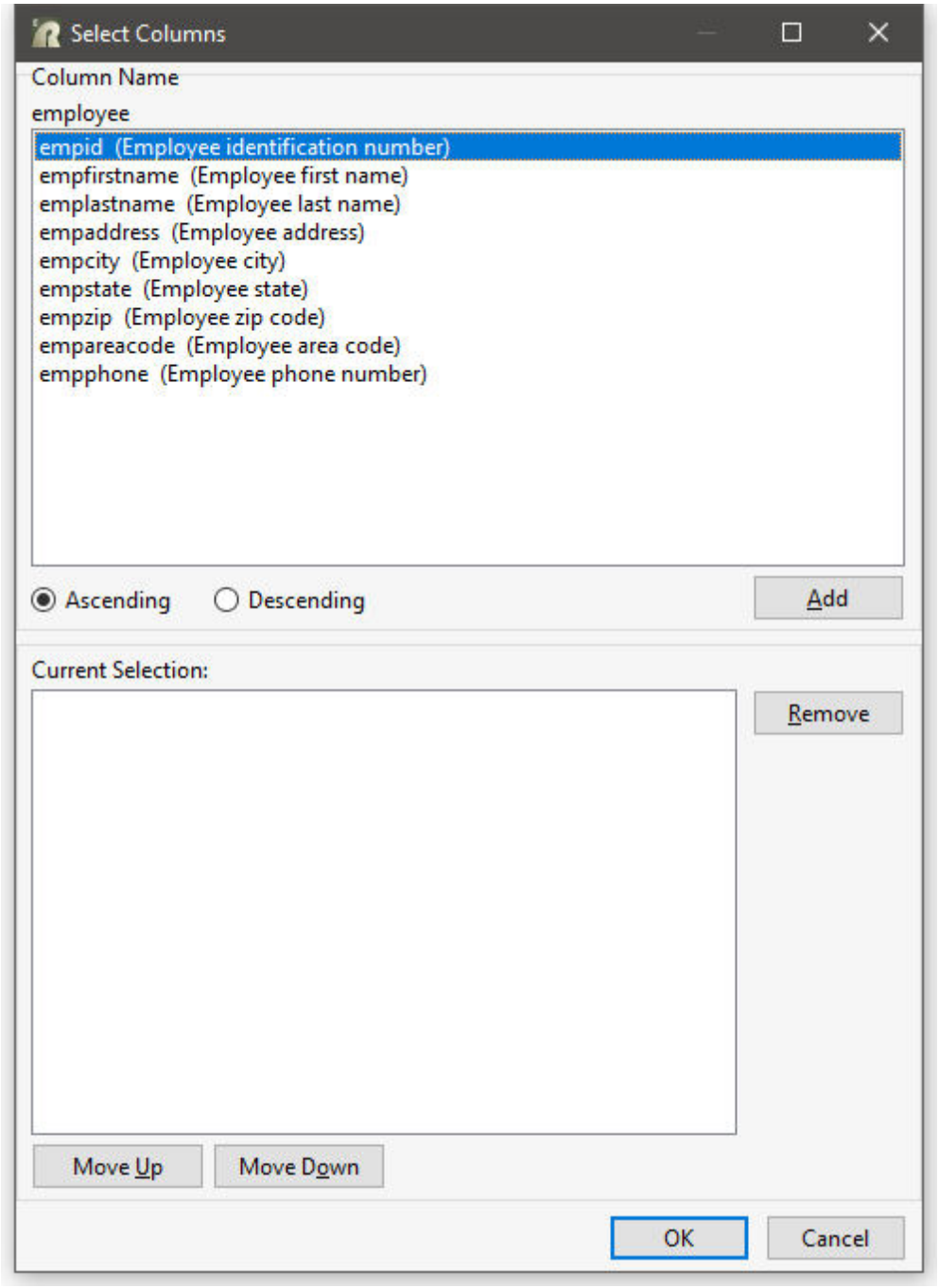
To define a primary key:

1. In the "Database Explorer" window, click the "Tables" option from the Group Bar.
2. Highlight the *employee* table.
3. Select "Design Table".
4. Click the "Keys/Indexes" option.

R:BASE displays currently defined indexes and constraints for the selected table. The window is blank, as none have been defined.

5. Click the "New Key" option.

R:BASE displays a sorting menu. The columns in the *employee* table are listed. Sometimes, tables have multiple column keys. Our primary key is a single column key.



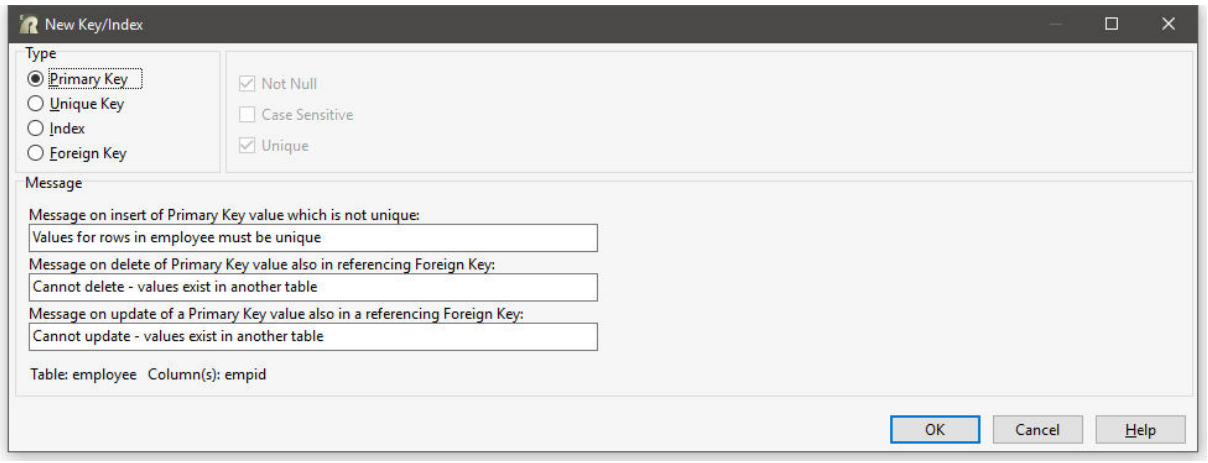
6. Select "empid".

You must choose the sort order for the constraint index.

7. Click the "Ascending" radio button to select ascending order and select "Add".

8. Click the OK button.

The "New Key/Index" dialog box is displayed.



9. Select "Primary Key".

The dialog box automatically displays options applicable to a primary key. We won't edit the default messages.

10. Click the "OK" button.

The primary key you just defined is now listed as a key/index for the employee table.

11. Click the "Save" option.
12. After confirming that the changes were saved, select the "Close" option.

Now, repeat this procedure to make the *custid* column in the *customer* table a primary key.

Next, we'll define the foreign keys for the *flights* table. When we defined the table relationships, we noted that we could have many rows for each employee and many rows for each customer in the *flights* table. By defining these columns as foreign keys, we ensure that matching values exist in the *employee* and *customer* tables.

To define a foreign key:

1. In the "Database Explorer" window, click the "Tables" option from the Group Bar.
2. Highlight the *flights* table.
3. Select "Design Table".
4. Click the "Keys/Indexes" option.

R:BASE displays currently defined indexes and constraints for the selected table.

5. Click the New Key button.

R:BASE displays a sorting menu listing the columns in the *flights* table. Sometimes, tables have multiple column keys. Our foreign keys are single column keys to match the defined primary keys.

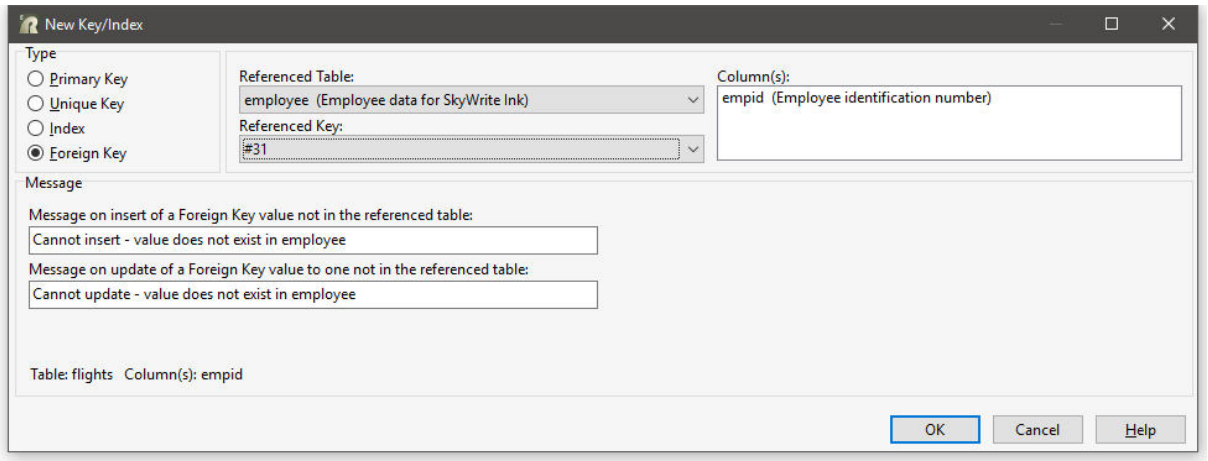
6. Select "empid".
7. Select the "Ascending" radio button to select ascending order and select "Add".
8. Click the OK button.

The "New Key/Index" dialog box is displayed.

9. Select "Foreign Key".

The dialog box automatically displays options applicable to a foreign key.

R:BASE requires you to select a "Referenced Table," the table with the corresponding primary key for this foreign key.



10. Select the "employee" table within the "Referenced Table" drop down menu.
11. Select the reference key number within the "Referenced Key" drop down menu.

Normally, there is only one number to choose in the "Referenced Key:" drop down box, unless more than one primary key is defined for a table.

12. Click the OK button.

The foreign key you just defined is now listed as a key/index for the *flights* table.

Now make the *custid* column in the *flights* table a foreign key. Repeat the steps above, and select "customer" as the matching (referenced) primary key table.

13. Click the "Save" option.
14. After confirming that the changes were saved, select the "Close" option.

3.3 Defining Indexes

When you index a column, R:BASE records the location of every row in the column. When you use the column to search for or sort data, R:BASE can find values in the column more quickly. The columns *empid* and *custid* are key columns and will be used to look up data from the *customer* and *employee* tables as well as the *flights* table. We automatically indexed these columns when we defined them as primary and foreign keys (constraints). We don't need to define additional indexes for them as R:BASE uses the constraint index to speed data retrieval and to link tables.

In addition to looking up data by employee and by customer, we might want to look up all the flights scheduled for a particular date or range of dates. To speed up this process, we'll index the *flightdate* column in the *flights* table.

To define an index:

1. In the "Database Explorer" window, click the "Tables" option from the Group Bar.
2. Highlight the *flights* table.
3. Select "Design Table".
4. Click the "Keys/Indexes" option.

R:BASE displays currently defined indexes and constraints for the selected table. The two foreign keys defined for the *flights* table are displayed.

5. Click the "New Key" option.

R:BASE displays a sorting menu listing the columns in the *flights* table. Indexes can be defined to use multiple columns. The multi column index works better when your where clauses have columns in the

same order as the index. If you plan to use single column where clauses then single column indexes would be return faster results. We are building a single column index.

6. Select "flightdate".

You are prompted to select the sort order for the index.

7. Click the "Descending" radio button and select "Add".

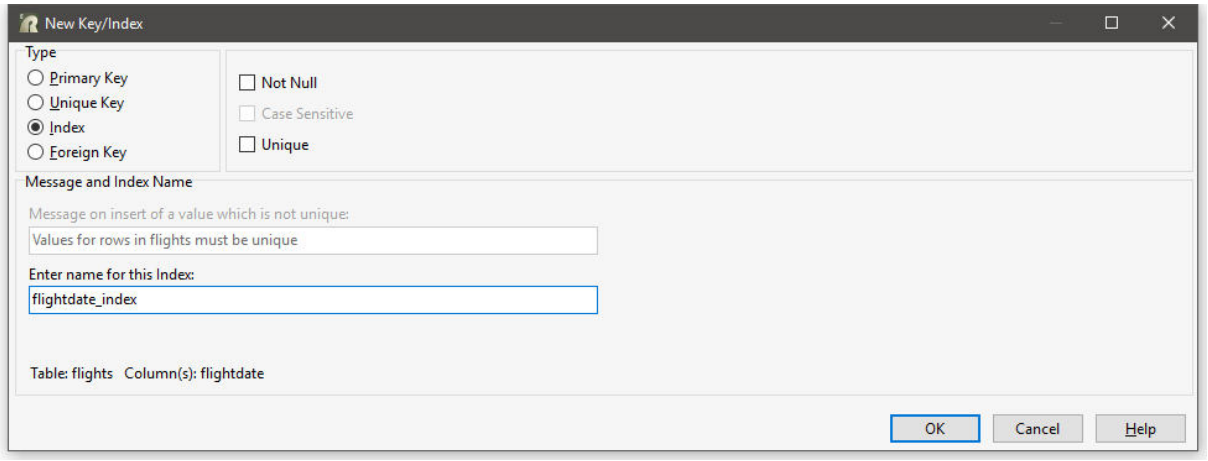
Descending was chosen as users are more likely to search newer records as the database grows, thus speeding up and searches on the flightdate column.

8. Click the OK button.

The "New Key/Index" dialog box is displayed.

9. Select "Index".

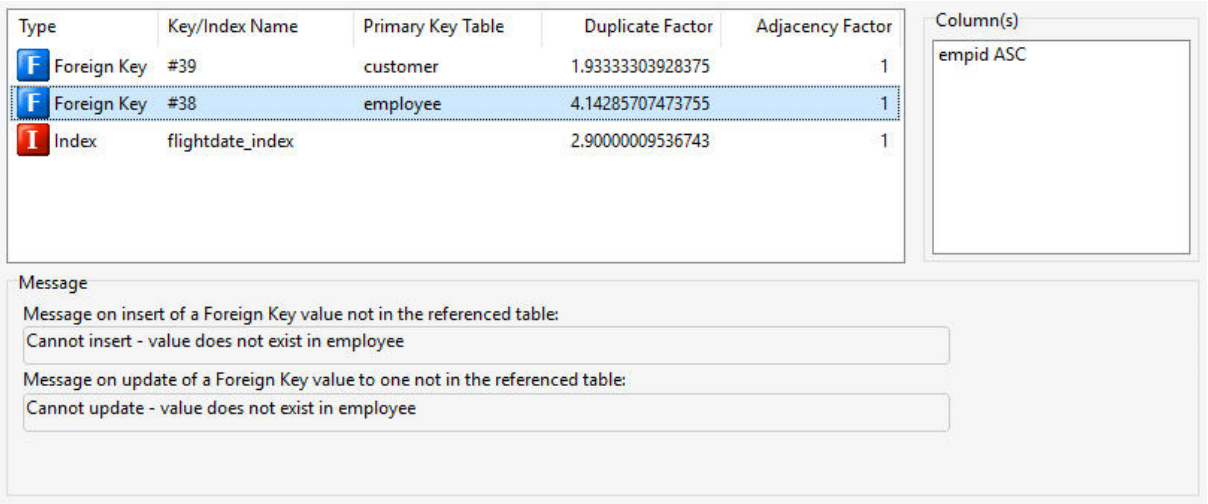
The dialog box displays options applicable to an index. Indexes must be named.



10. Enter "flightdate_index" as the name for the index within the "Enter name for this Index" field.

11. Click the OK button.

The index definition is complete. You should the following keys and index defined for the *flights* table.



12. Click the "Save" option.
13. After confirming that the changes were saved, select the "Close" option.

Well done! You have just defined a relational database using RBASE. Now we'll look at the database files and structure.

3.4 The Database Files

R:BASE databases are stored in four files. When you created the *SKY* database in R:BASE X.5, R:BASE created four files: SKY.RB1, SKY.RB2, SKY.RB3, and SKY.RB4. When you created the *SKY* database in R:BASE X.5 Enterprise, R:BASE created four files: SKY.RX1, SKY.RX2, SKY.RX3, and SKY.RX4. These four files together make up a database.

Using the RB1-RB4 files as an example, the first file, SKY.RB1, holds the information on the structure of the database including the table and column definitions. The second file, SKY.RB2, holds the data. SKY.RB3 holds the values of all indexed columns. The fourth file of the database, SKY.RB4, holds the large object data. When you defined the database, R:BASE stored the four database files in the current directory.

Viewing the Database Structure

We used the [F3] key earlier in this lesson to look at database information. We can also look at database information using the R:BASE command mode-the "R:BASE R> Prompt" window. There, we'll use the LIST command to see information about the structure of the database-the columns and tables.

To list the tables in the sky database:

1. Choose Tools: "R> Prompt..." or select the R> Prompt button on the toolbar.

The "R:BASE R> Prompt" window opens. Whenever you are at the R:BASE command line, R:BASE displays an "R>" prompt.

2. Enter "LIST" at the "R> Prompt and press [Enter].

A list of the tables in the database is displayed.

To edit these settings, choose "Settings" from the Menu Bar, and choose "R> Prompt..." from the list of options.

The tables you created are listed. The screen shows the number of columns and rows in each. The total number of tables and columns in the database is more than you defined. These numbers include the R:BASE system tables. The system tables are not displayed with this LIST command.

To list the system tables in the sky database:

At the R> Prompt, enter "LIST SYS%" and press [Enter].

A list of the system tables in the database is displayed. Each system table begins with "SYS_" R:BASE takes care of updating and maintaining the system tables.

In addition to listing information about the tables in a database, you can also list information about the columns in a database. First, display the columns defined for a specific table.

To list information about a specific table:

1. At the R> Prompt, enter "LIST TABLE customer" and press [Enter].

The "More output follows" dialog box is displayed in the center of the screen. You can move the dialog box so it doesn't cover the data on the screen. The dialog box will be displayed in the new location until you move it again.

Information about the *customer* table is displayed. The *custid* column is marked as a primary key, autonumbered column.

2. Click the "Yes" button to display the rest of the columns in *customer.*

Now, we'll look at all the columns defined for the *sky* database.

To list all the columns in a database:

1. At the R> Prompt, enter "LIST COLUMNS" and press [Enter].

R:BASE displays all of the columns in the database in alphabetical order. You can see which columns appear in more than one table. For example, the *custid* column is listed in the *customer* and *flights* tables.

2. Click the "Yes" button to display more output or click the "No" button to quit.

Since we are done looking at the list of columns, let's clear the screen.

To clear the screen:

At the R> Prompt, enter "CLS" and press [Enter].

R:BASE clears the screen of all data.

3.5 Importing Data into the Database

In the next lesson, we will be viewing information in the database. So now we need to add data to the tables that we created.

In the "Tutorial" directory located in the R:BASE X.5 default installation folder, either C:\RBTI\RBGX5\Tutorial\ or C:\RBTI\RBGX5E\Tutorial\, there is a file called SKYDATA.BAK. The file contains data for all three tables in *sky,* as well as the R:BASE commands needed to load the data into the correct tables. We'll use a command to load the data from the file.

IMPOTANT NOTE: It is likely that the folder where you created the *sky* database is not the same folder location where the SKYDATA.BAK file is located. In this case, you are required to copy the SKYDATA.BAK file to your directory where the *sky* database exists.

To load the tables in the sky database:

Choose Tools: R> Prompt... if the "R:BASE R> Prompt" window is not open.
Enter "RUN SKYDATA.BAK" and press [Enter].

R:BASE displays a series of messages to tell you that data is being loaded into the tables in the *sky* database. If you receive an error stating that the file does not exist, refer to the "IMPORTANT NOTE" above.

Now let's check to see that the rows were loaded into their tables.

Enter "LIST" and press [Enter].

R:BASE lists the tables in the database. Check to make sure that there are 7 rows of data in the *employee* table, 15 rows in *customer,* and 28 rows in *flights.*

To close the R> Prompt window:

Right-click the Status Bar menu for the "R> Prompt" window and choose Close or select the "X" (lower X) in the Menu Bar, which is the default for most Window's programs.

3.6 Changing Database Structure

The structure of a database includes table and column names, and column characteristics such as data type, length, and index status. As you develop a new database, it's quite common to modify the database structure. For example, add a computed column to automate a calculation you frequently perform, or define rules that make sure only valid information can be entered into the tables.

Modifying a Table

Columns can be added or modified anywhere in a table. The table might or might not contain data. In this section, you will modify the *flights* table by adding a new column that calculates the total cost of the advertising slogan for each flight.

The number of letters in the advertising slogan determines the skywriting fee. The price per letter is contained in the *price* column of the *flights* table. In order to see the total cost of the slogan for a flight, the number of letters in the *letters* column needs to be multiplied by the rate in the *price* column. You can have R:BASE calculate the total cost of a slogan by defining a computed column.

To open the flights table for modification:

1. If the Database Explorer is not open, choose Tools: Database Explorer...
2. Click the Databases option.
3. Highlight "sky."
4. Click the "Connect Database" option to open the *sky* database.

If you were not already connected to the database, you will see the options for tables, views, forms, reports, labels, and stored procedures appear.

5. Click the "Tables" option for the table list to appear.
6. Highlight "flights" from the table list and enter the Data Designer. You can enter the Data Designer by selecting "Design Table" from the "Group Bar" or by right clicking on the table name and selecting "Design Table".
7. Click the "Columns" option.

R:BASE displays the columns in the *flights* table.

About Computed Columns

A computed column uses an expression or function to calculate a value from the values of other columns in a table or from constant values you specify. An expression is a calculation used to determine a value and is made up of operators and operands. The operators are listed below:

Operator	Definition
+	Addition
-	Subtraction
*	Multiplication
/	Division
**	Exponentiation
+	Concatenation (with no space)
&	Concatenation (with one space)

Operands, which are the part of the expression the operators are acting on, can be variables, constant values, or column names.

Like a regular column, a computed column has a data type and can be indexed. Unlike a regular column, a column that does not contain a value until you enter one, you can't enter or update a value in a computed column. R:BASE enters or updates the value for you.

To calculate the total charge for a slogan, you'll define this expression:

operator
|

letters * price
\ /
operands

Note: For more information about expressions, see the "Expressions" chapter within the "Reference Topics" of the R:BASE X.5 Help.

Adding a Computed Column

You can change any part of the column definitions for a table. For this lesson, you won't change any of the existing columns; you'll just add a new column. When using computed columns, it is important that a computed column always follow any columns used in its expression.

The first column in the table, *custid,* is highlighted. Columns are inserted above the highlighted column, so you need to first scroll the column list display past the *letters* and *price* columns.

To add a computed column to the flights table:
1. In the "Data Designer" window under the "Columns" tab, click in the *place* column.
2. Click the "Insert Column" option.

R:BASE displays a dialog box for the new column name.

3. Name the column "totalcharge". You will notice the column was inserted above *place.*
4. Choose "CURRENCY" from "Data Type".
5. Check the "Compute" radio button.

R:BASE will change the screen for you to enter an expression.

6. Enter "(letters * price)" in "Expression".

The computed column *totalcharge* has now been added to the flights table.

7. Click the "Save" option.
8. After confirming that the changes were saved, select the "Close" option.

Data Entry Rules

Data entry rules help ensure that the information in your database is accurate and reliable. You can use rules to ensure that the data entered into a column meets the criteria you specify.

There are two methods for defining rules. You can create quick rules-rules with conditions and messages supplied by R:BASE-or define custom rules. When you define custom rules, you must define all parts of the rule. In this lesson, you'll use a quick rule option to create two rules for the *flights* table that say:

- The *price* column must contain a value between $40 and $50.
- The *totalcharge* column must be greater than or equal to $400.

These rules ensure that the prices charged for slogans meet expenses. Thus, at the price of $40 per letter, a customer displays a slogan of up to 10 letters in order to meet the minimum total charge requirement of $400.

Defining a Quick Rule

The easiest way to define a rule is to use one of R:BASE's quick rule options.

To define a Quick rule:

1. Enter the "Data Designer" window for the *flights* table.
2. Click the "Rules" option.

R:BASE displays rules that have been defined for the selected table, *flights.* No rules have been defined yet so the screen is empty.

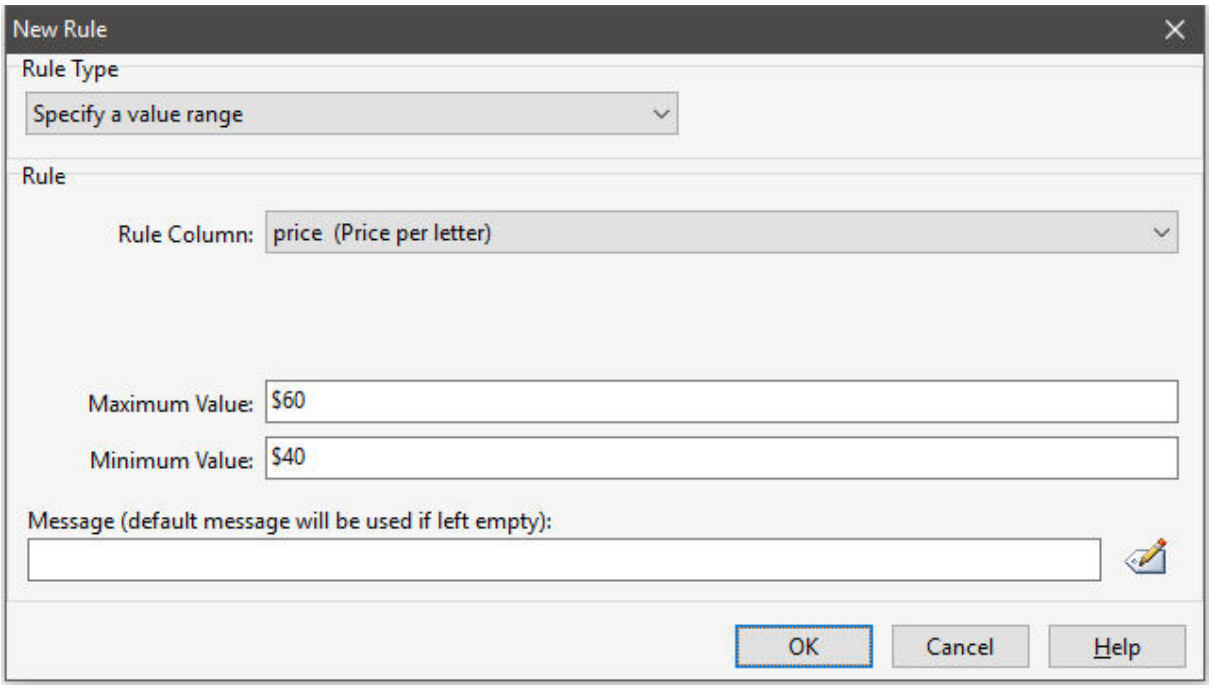
3. Click the "Add Rule" option.

R:BASE displays the "New Rule" dialog box.

4. Click the arrow on the "Rule Type" drop down menu to display the rule types. The first seven rule types define quick rules. The last allows you to define a custom rule.
5. Select "Specify a value range".

R:BASE will display a drop down box to select the Rule Column-the column this rule will apply to.

6. From "Rule Column" select "price".
7. Enter "$50", the maximum value for the range, in "Maximum Value".
8. Enter "$40," the minimum value for the range, in "Minimum Value".



9. Click the OK button.

R:BASE returns to the "Data Designer" window and lists rules defined for the flights table.

10. Click the "Save" option.
11 After confirming that the changes were saved, select the "Close" option.

R:BASE saves the table and the rule definition.

Listing Rules

When you define a quick rule, you choose only the rule table and column. R:BASE supplies the rest of the rule definition for you. When listing rules, you see all parts of the rule definition.

To list rules for the flights table:

In the "Data Designer" window, click the Rules button. R:BASE displays the rule you just defined.

Rules are displayed for the selected table only. You can view all the rules defined for a database, by using the LIST command at the "R:BASE R> Prompt" window.

To list rules for all the tables:

1. Choose **Tools: R> Prompt...**

The "R:BASE R> Prompt" window opens.

2. Enter "LIST RULES" at the "R>" prompt and press [Enter].

A list of all rules defined for the database is displayed.

Parts of a Rule

Whether you define quick rules or custom rules, all rules have four parts:

- Rule Table-The table to which a rule applies.
- WHERE clause-The WHERE clause specifies the exact conditions a row must satisfy or not satisfy (depending on whether you choose Succeeds or Fails) in order to be added to the table.
- Succeeds/Fails-A row must meet or must not meet the conditions specified in the WHERE clause. Succeeds specifies that a row is added to the table only if it satisfies the conditions in the WHERE clause. Fails specifies that a row is added only if it does not satisfy the conditions in the WHERE clause.
- Rule message-The rule message is what you want displayed if the conditions of the rule are violated.

You can change any or all of these parts of a rule.

For the rule you just defined, R:BASE used a standard message: "Value for price cannot be outside allowed range." This message is easy for you to understand because you designed the database and created the rule. But other users entering data probably won't know the allowed range, so you should change this message to say what the allowed range is.

Modifying a Rule

Once you have created a rule, you can edit the rule to change the message, the rule condition, or any other part of the rule.

To change the rule message:

1. In the "Data Designer" window for the "flights" table, click the "Rules" tab.

R:BASE displays the rules defined for the *flights* table.

2. Select the rule you just defined
3. Click the "Modify Existing Rule..." option.

R:BASE displays the "Rule" dialog box. You can change any part of the rule definition here. Let's change the message.

4. Highlight the rule message in "Message".
5. Edit the text to "Price per letter must be between $40 and $50." With the radio button on " Add/Modify" the message will be updated.
6. Click the OK button.

R:BASE writes the change to the rule.

Creating a Rule on Your Own

Following the steps used previously, create a rule that says the total charge for a flight must be at least $400.00 before it can be entered in the *flights* table. After you create the rule, change the rule message.

Here's the information you'll need to create the rule and modify its message:

- The rule table is *flights*
- The quick rule option is "Specify a minimum value"

- The rule column is *totalcharge*
- The minimum value is $400.00
- The modified rule message is "Flights cannot be ordered for less than $400.00"

After you are done with the modifications to the *sky* database and have saved your changes, close the "Data Designer" window.

To close the "Data Designer" window:

7. Click the "Save" option.
8. After confirming that the changes were saved, select the "Close" option.

3.7 Summary

In this lesson you created the database that you designed in Lesson One. Remember, always design your database on paper before building it. This will save you hours of time and frustration. You did the following:

- Named the *sky* database
- Defined the *employee, customer,* and *flights* tables
- Defined the columns for each table and selected data types for each column
- Designated *empid* and *custid* as auto-numbered columns
- Designated *empid* and *custid* as primary and foreign keys to ensure data integrity
- Designated *flightdate* as an indexed column for faster processing
- Looked at the structure of the *sky* database
- Added the computed column *totalcharge* to the *flights* table
- Created two quick rules
- Listed rules
- Modified the rule messages

To check your knowledge, try answering the following questions:

1. How do you start the Database Designer?
2. How many tables can be defined for one database?
3. How many columns can be defined for one table?
4. What must you do to designate a column as indexed for faster processing by R:BASE?
5. What is an example of data that would match the INTEGER data type?
6. How do you designate a column as an autonumbered column?
7. What function key displays the tables and columns in the database?
8. What is the command from the R> Prompt that displays database information?
9. What is a computed column?
10. Can you enter or update values in a computed column? (Yes- No)
11. Why do you define rules for a database?

Part IV

4 Lesson 3 - Viewing Data

Once tables are constructed and contain data, you can return to them to look at the data, change the way the data is displayed, modify the data, and query the data. A query is a question you construct to get information from your database tables and columns. This lesson teaches you how to look at your data and how to build a query.

At the end of this lesson you will be able to:

- Hide, move, and lock columns
- Change the display size of a column
- Add and delete rows of data
- Perform calculations on your data
- Sort your data
- Build single and multiple condition queries
- Print the results of your query
- Create a multi-table query
- Save the query as a view
- Select the rows R:BASE displays in the view

You have designed the *sky* database and created its tables and columns. Now that the database contains skyWrite Ink's employee, customer, and flight information, you are ready to use R:BASE features to view this data. You'll be able to see important information easily and find answers quickly.

The two features of R:BASE you'll use to display data or build a query from a single table are the Data Browser and the Query Builder.

- **Data Browser -** The data from the table is displayed in the Data Browser window. You can browse the data, change the way the data is displayed, sort the data, perform calculations, and also edit the data. To open a table in the "Data Browser" window, in the Database Explorer, click the Tables option. Then, select a table and click the "Open" option. You can also choose **Tools: Browse/Edit Table** to open the "Data Browser" window. Any of these options will display your data in the "Data Browser" window.

- **Query Builder -** The Query Builder is used to ask questions of your data such as "Which flights were flown in April?" or "Which companies are located in Olympia?" You create a query by selecting the tables, columns, and conditions you want to use-setting specifications that tell R:BASE which rows of data to retrieve. To build a query in R:BASE from the Database Explorer window, click the "Views" tab within the "Group Bar" and click "New". When creating a new query, the Query Builder will always ask you if you want use the Query Wizard, which is a utility to built queries on the fly!

4.1 Displaying Data

In this section we'll display data from the *customer* table. Then, working in the "Data Browser" window, we'll customize the appearance of the displayed customer information.

To display data:

1. If the *sky* database is not open, click the "Databases" option in the Database Explorer, select *sky,* then click the Connect option.

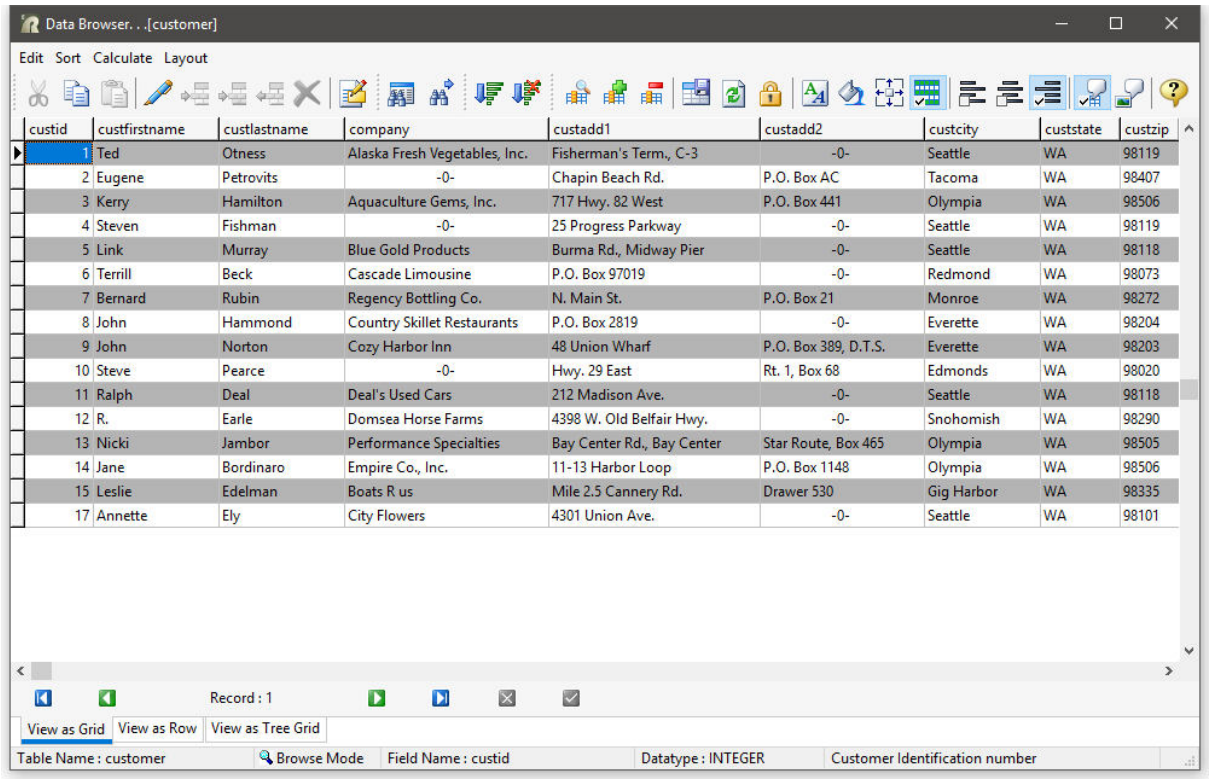
2. In the Group Bar, click the Tables option.

   The tables in the database are listed in alphabetical order.

3. Highlight the *customer* table.

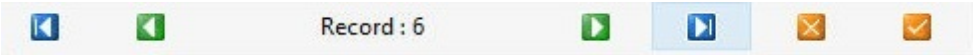
4. To open the table, select "Open Table" or double-click on the table name.

The "Data Browser" window opens with the data from the *customer* table.



The Menu Bar across the top of the R:BASE window, which changes depending on your current activity, has new options, "Edit", "Sort", "Calculate", and "Layout", which are used by the "Data Browser" window.

When you created the *customer* table in Lesson Two you created 11 columns. You can use the scroll bars or the Navigation Bar to display other columns and rows. The Navigation Bar contains arrow buttons to move to certain rows of the table and the ability to view an entire row of data vertically.



Note: When you maximize a window, the "R:BASE X.5" window title bar changes to include the title of the maximized window. The control-menu and the normalize button move to the left and right sides respectively of the menu bar. When you maximize a window, all the other open windows are also maximized.

For more information about navigating in Windows, refer to your Windows documentation.

4.2 Hiding Columns

You can customize the way data is displayed on your screen. Using the options on the **Layout** menu, you can move columns, hide or show columns, lock the column position, change the cell size, change the column color, change the cell font, hide duplicate rows, add zebra stripes, and adjust the data justification. We'll hide three columns from view.

Let's say you want to see the company and customer names and addresses, but do not need to see the customer IDs or phone numbers. You can temporarily remove these columns from the display.

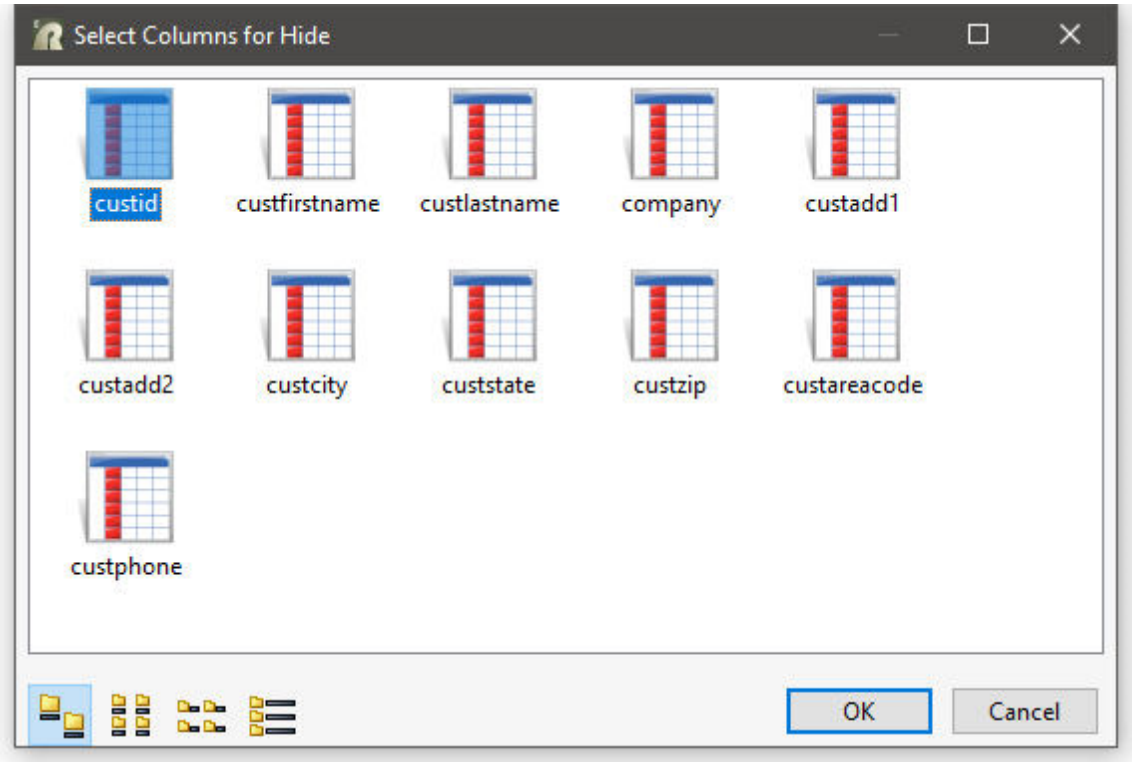
To hide the custid column:

1. In the "Data Browser" window, choose **Layout: Hide Column...**

The "Hide Column" dialog box displays a listing of the columns in the *customer* table.

2. Select "custid".

"Custid" is highlighted. More than one column can be selected.



3. Click the OK button.

The *custid* column disappears. Remember, customizing the layout does not change the table structure. The *custid* column and data are still in the table.

Repeat this procedure for the *custareacode* and *custphone* columns.

4.3 Redisplaying Hidden Columns

If you later decide you want to see the data in a hidden column, you can redisplay the column.

To redisplay the custareacode and custphone columns:

1. In the "Data Browser" window, choose **Layout: Show Column...**

The "Show Column" dialog box displays the hidden columns in the *customer* table.

2. Select "custareacode" and "custphone".
3. Click the OK button.

The columns reappear in their previous order.

Use the same procedure to redisplay the *custid* column.

Note: Changes made to the data display using the options on the Layout menu are not automatically saved. The default setting for "Layout" is off. To save changes, choose **Settings: Configuration Settings: Operating Condition** and set "LAYOUT" to on. This setting switches saving layouts on or off. When LAYOUT is set on, R:BASE saves the layout of data displayed in the Data Browser when you exit. The next time you display the table with the Data Browser, the layout of data will be as you previously

arranged it. When LAYOUT is set off, R:BASE does not save the layout of data and ignores any saved layouts. If you want to see a table displayed in its default format without changing the saved layout for it, set LAYOUT off before displaying the table.

4.4 Changing the Column Size

Notice the extra spaces to the right in the *custfirstname* and *custlastname* columns. The cell size shown on the screen is determined by the column length you specified when you created the *customer* table. We can display more columns on the screen if we decrease the column size.

To change the column size:

1. In the "Data Browser" window, click on the *custfirstname* column.
2. Choose **Layout: Change Cell Size...**

The "Cell Properties" dialog box appears. You can change the column width and/or the row height. Currently the column width is 184 and the height is 20. We'll change the column width to 150.

3. In the "Cell Properties" dialog box, enter "150" in the "Width" field.
4. Click the OK button.

The column width for *custfirstname* is extended. Now, we'll decrease the width of the *custlastname* and *custstate* columns.

Follow Steps 1 through 4. In Step 3, enter "160" in "Width" for the *custlastname* column and "35" for the *custstate* column.

5. Press the [Home] key to return to the first cell.

4.5 Moving Columns

The screen now displays the columns we are most interested in. However, it would be easier to find information about the companies if their names were listed first, on the left side of the screen.

To move the company column:

1. In the "Data Browser" window, click and hold the mouse button down with the cursor on top title of the *company* column.

   You will notice the thick black line running vertically down the left side of the column.

2. While holding down the mouse button and dragging cursor, you can move the column to any place within the table. Drag the *company* column to the far left.

4.6 Locking Columns

We have the columns in the order we would like to have the data displayed. We would like to keep the company name displayed on the screen when we move to the right to view the other columns. Currently the columns scroll off the screen to the left.

You can lock one or more columns along the left margin of the "Data Browser" window.

To lock the company column:

1. In the "Data Browser" window, choose **Layout: Lock Column...**

The "Lock Column" dialog box is displayed to enter the number of columns to lock. The number represents the number of columns from left to right.

2. Enter a 1 to lock the first column listed, which is *company*.
3. Click the OK button.

You will notice that the *company* column cells have a gray background like the actual column names.

4. Scroll through the columns. The *company* column remains locked in place. Now you are able to see the company address while still displaying the company name on the screen.

4.7 Clearing Locked Columns

Just as you can lock a column, you can unlock it as well.

To unlock the company column:

1. In the "Data Browser" window, choose **Layout: Lock Column...**
2. In the "Lock Columns" dialog box, click the OK button.

R:BASE unlocks all columns.

4.8 Working with Data

So far, we've seen how to change the way R:BASE displays the data in a table. Now let's see how to change the data. Switch from the "Data Browser" to the "Data Editor" by pressing the [F4] key or choosing **Edit: Edit Mode** from the Menu Bar. In the "Data Editor" window, you can add, change, or remove information. We'll learn how to sort the rows of data and how to perform calculations.

4.9 Adding a Row

You can add rows of data or edit existing information in the "Data Editor" window. Let's add a new customer.

To add a row to the customer table:

1. Press the [Ctrl] + [Home] keys to return to the first cell.
2. In the "Data Browser" window with the *customer* table open, choose **Edit: Edit Mode**.

The Table Status Bar has changed from "Browse Mode" to "Edit Mode".

Before you can enter the new customer information, you must insert a blank row.

3. Choose **Edit: Insert Row** or press the [F2] key.

R:BASE adds a blank row to the table. When a row is added, R:BASE displays it on the line above the currently selected row. However, R:BASE stores the row at the end of the table. The next time you open the table, the row will be at the bottom of the table.

When the new row is added, R:BASE automatically numbers the *custid* column; it contains the next highest number, "16." We defined default values for the *coststate* and *custareacode* columns in Lesson 2. R:BASE fills in those values. The other columns in the row we've added contain a null *(-0-)* value.

4. Press [Tab] to see the default values.

Let's enter the first and last name of a new customer in the new row. Enter "Laurel" in *custfirstname* and "Tranowski" in *custlastname.*

4.10 Deleting a Row

You can also use the "Data Editor" to delete rows. Let's delete the row we just added.

To delete a row from the customer table:

1. In the "Data Editor" window, click in the row we just added.
2. Choose **Edit: Delete Row** or press the [F9] key.

R:BASE asks if you are sure you want to delete this row.

3. Click the Yes button.

The row is deleted from the screen and from the customer table.

4.11 Sorting Data by a Single Column

It is often useful to see data in a specific order, either to get a quick answer on the screen or for reporting purposes. R:BASE provides an easy way to sort rows using a single column or multiple columns. You can specify whether to sort in ascending or descending order.

We will use the *flights* table. Before sorting data, close the Data Browser for the *customer* table.

To close the "Data Editor... [customer]" window:

Make sure the "Data Editor...[customer]" window is active. Right click the Status Bar for the window and click Close or use the "X" on the Data Editor window.

We'll open the "Data Browser" window for the *flights* table.

To open the flights table:

1. In the "Database Explorer" window, highlight the "flights" table.
2. Select "Open Table".

The data from the *flights* table is displayed in a "Data Browser" window.

Now that we've displayed the data from the *flights* table, suppose you would like to see the flight information sorted in alphabetical order by the slogan. To sort the data by slogan:

1. In the "Data Browser" window, click on any data for the *slogan* column for focus.
2. Click on the column name "*slogan*".
3. The column background will turn green for "Ascending" order, unless the Data Browser column titles are displaying a gradient.
4. Click on the column name "*slogan*" again.
5. The column background will turn yellow for "Descending" order, unless the Data Browser column titles are displaying a gradient.

To remove the column title gradient in order to see the green/yellow colors when sorting columns in ascending and descending order, select "**Settings: Data Browser**" from the main Menu Bar. There will a check box you can remove.

4.12 Sorting Data by Multiple Columns

Now let's sort by flight date and customer ID. This is a multi-column sort. If you want to return the data to the original sort order, choose **Sort: Clear Sort** from the Menu Bar.

To sort the data by flightdate and custid:

1. In the "Data Browser" window, choose **Sort: Sort** from the Menu Bar.

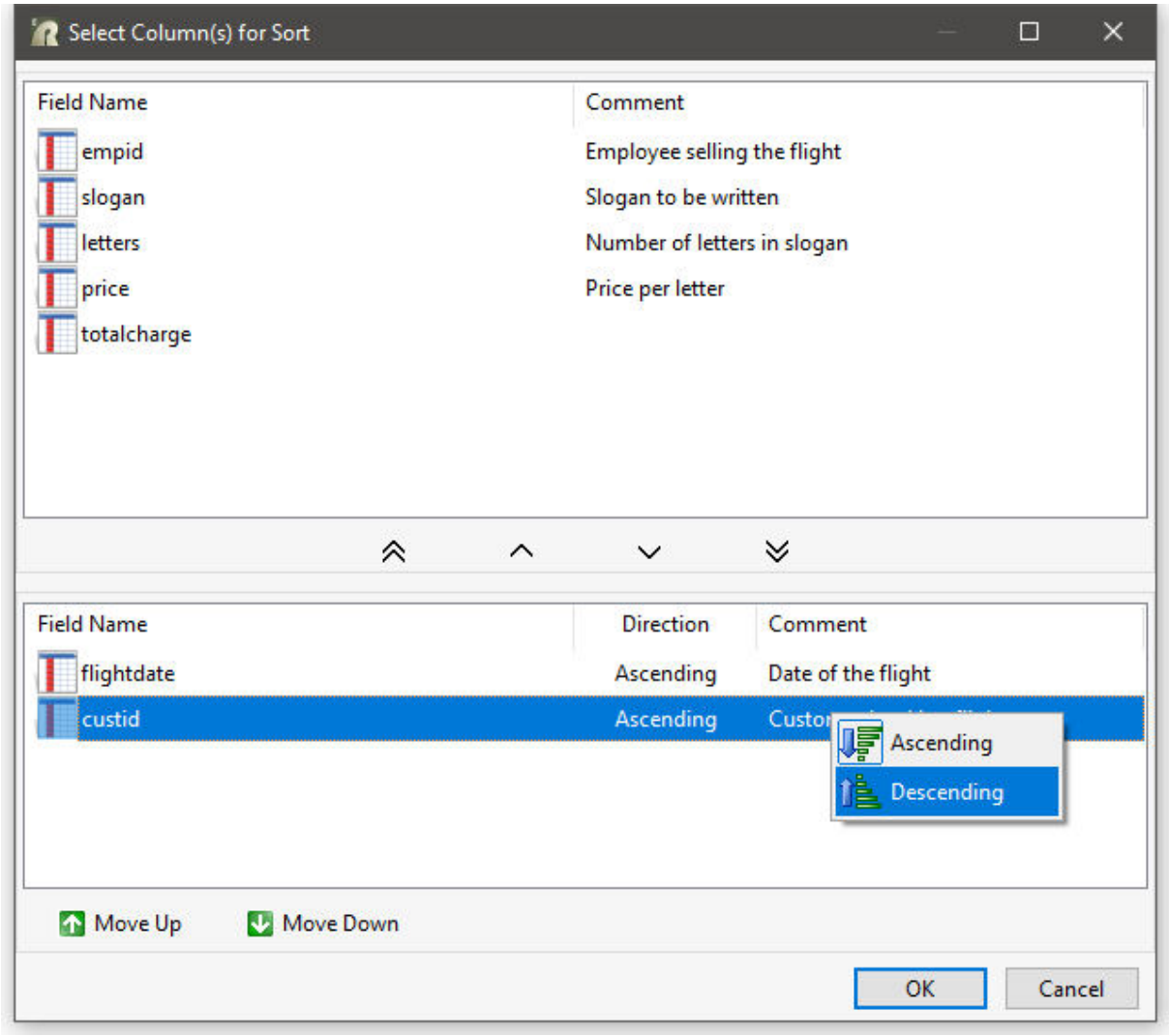
The "Select Column(s) for Sort" dialog box displays the columns in the flights table.

2. Highlight the column "flightdate" and click on the single arrow pointing to the bottom panel.

You will notice the column moved to the bottom panel.

3. Highlight "custid" and click on the single arrow pointing to the bottom panel.

Right-click on the field name "custid" within the bottom panel and a sorting option will appear.



4. Click the Descending option.
5. Click the OK button.

R:BASE displays the data sorted first by flight date and then by customer ID.

4.13 Performing Calculations on Table Data

Suppose you would like to see if skyWrite Ink has many repeat customers, and if so, who those customers are. You can perform a variety of calculations on your data using the four options, and the seven aggregate calculations:

- Compute
- Tally
- Group
- Crosstab
- Aggregate
  - Count
  - Minimum
  - Maximum
  - Sum
  - Average
  - Standard Deviation

- Variance

To get the information we want about our repeat business we will tally it. A tally counts the number of occurrences of each value in a column.

To count how many times a customer ordered flights:

1. In the "Data Browser" window, choose **Calculate: Tally: Select Column**.

The "Tally" dialog box displays the columns in the *flights* table. We want to count the number of times each customer ID occurs in the table.

2. Highlight "custid", and click the OK button.

The tally is displayed in the "Tally" window.

The values in the *custid* column are sorted in ascending order and the number of occurrences for each value is displayed. You can see that customer number "1," who has booked four flights, does the highest volume of business. Close the "Tally" window when you are finished.

From time to time, skyWrite Ink gives bonuses to its salespeople. You would like to know the number of sales each employee has made. Use the **Tally** option to display the number of sales for each employee.

4.14 R:BASE BLOB Editor

To view the NOTE data and images stored in the database, R:BASE includes the R:BASE BLOB Editor, an integrated display screen and editor for these specific data types. When a column is defined with a NOTE data type only some of the text is displayed in the "Data Browser" window. If there is an image stored in the table, then a small icon is displayed. These data types are stored in the fourth database file.

To open the R:BASE BLOB Editor:

1. In the "Data Browser" window, [Tab] the cell focus to the right, until you are on the *place* column.
2. Using the mouse cursor, double click the *place* column.

The "R:BASE BLOB Editor" will launch in a separate window. Only a small amount of data is stored in the column, but this tool allows the ability to add much more.

3. Close the window to return to the Data Browser.

4.15 The Query Wizard

A query is a question to the database that you construct, allowing you to specify the information you want to see. When we opened the *flights* table and displayed the data in the "Data Browser" window, we actually made a query. We told R:BASE to display all the data in all of the columns in the *flights* table. You can look at your data any way you want with a query. Creating a new query is very simple in R:BASE X.5 using the Query Wizard, which displays a graphical representation of your query.

Before working on the query, we'll close the "Data Browser" window and open the "Query Wizard" window. Close the Data Browser.

To build a query with the Query Wizard:

1. If the *sky* database is not open, click the Databases tab in the Database Explorer and select "sky".
2. Click the Connect option.
3. To open the Query Wizard, select Views from the Group Bar followed by "New View...".

A pop up will appear asking if you would like to use the Query Wizard.

4. Select Yes.

The "Query Wizard" window is displayed.

5. To start, select "Tables" from the "List of Tables/Views" to the right.

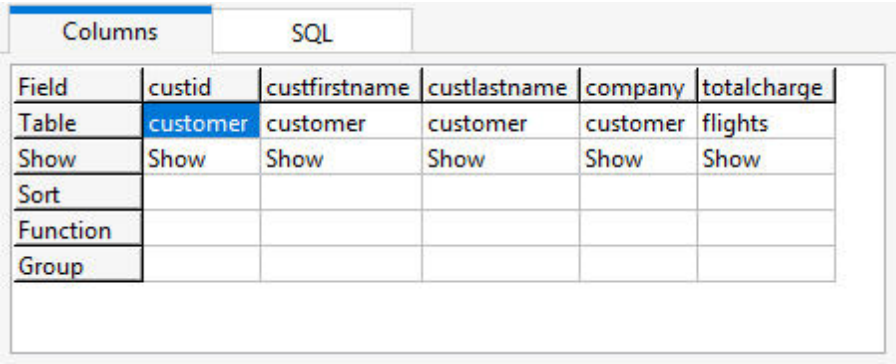
The list of defined tables for the database will appear. To add a table to your Query, select a table from the "List of Tables/Views" to the right, and drag the table onto the workspace by holding down your left mouse button. Wherever you let go of the mouse button, the table with the column list will appear.

6. Drag the *flights* and *customer* tables into the workspace.

To select your columns, either right click in the table area and use the speed menu to "Select All" or "Unselect All" columns, or click the check box next to the column name to select individual columns.

7. Select the *custid, custfirstname, custlastname,* and *company* columns for the *customer* table.
8. Select the *totalcharge* column from the *flights* table.

Notice that when each column is selected, the column grid below id being populated with your selections. What you are basically doing is constructing a SELECT clause that your query is based upon. Next, you will add a WHERE Clause to limit the amount of data returned.

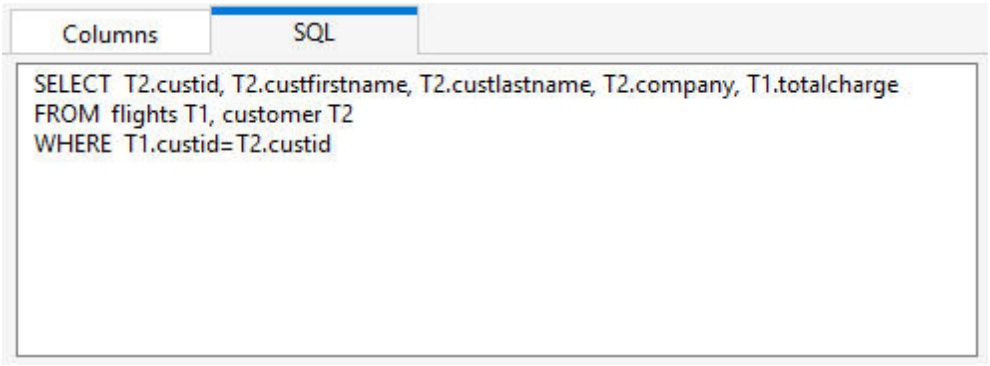


To add a WHERE clause to your query, you need to drag the linking column from one table to the linking column in the other table. The direction that you drag the link is relative to create the WHERE clause syntax. In the example below, if *custid* from "customer : T2" is dragged to "flights : T3", the WHERE clause would look like: WHERE t2.custid=t3.custid. However, if *custid* from "flights : T3" is dragged to "customer : T2", the WHERE clause would look like: WHERE t3.custid=t2.custid. This may be relevant in some cases when the selected columns will limit the results from either table.

9. For this view drag the mouse from "customer : T2" *custid* to the "flights : T3" *custid*.

The link, in the form of a line from column to column, will appear in the workspace joining the tables.

As you edit the query, the "SQL" syntax for the SELECT command in the bottom panel will change. Select the "SQL" tab to see the syntax. To refresh the changes, select the "Build SQL" button on the toolbar. It should read as follows:



The Query Wizard also gives you the option of sorting the data in an ascending or descending order. This will add an ORDER BY clause at the end of the WHERE clause.

To sort your query with an ORDER BY clause:

1. Click the "Columns" tab in the lower panel of the Query Wizard to display the selected columns.
2. Right click the "Sort" row of the column *custid*.
3. From the speed-menu list, choose "Ascending".

The text "ASC" should appear in the column.

The Query Wizard also gives you the option of adding an aggregate function to the query. An aggregate function calculates the sum, count, minimum, maximum, or average for a selected column.

To add an aggregate function to a column in your query:

1. Click the "Columns" tab in the lower pane of the Query Wizard.
2. Right click the "Function" row of the column *totalcharge*.
3. From speed-menu list, choose "Sum".

The text "SUM" should appear in the column.

The Query Wizard gives you the ability to Group columns so that only distinct values for a column appear.

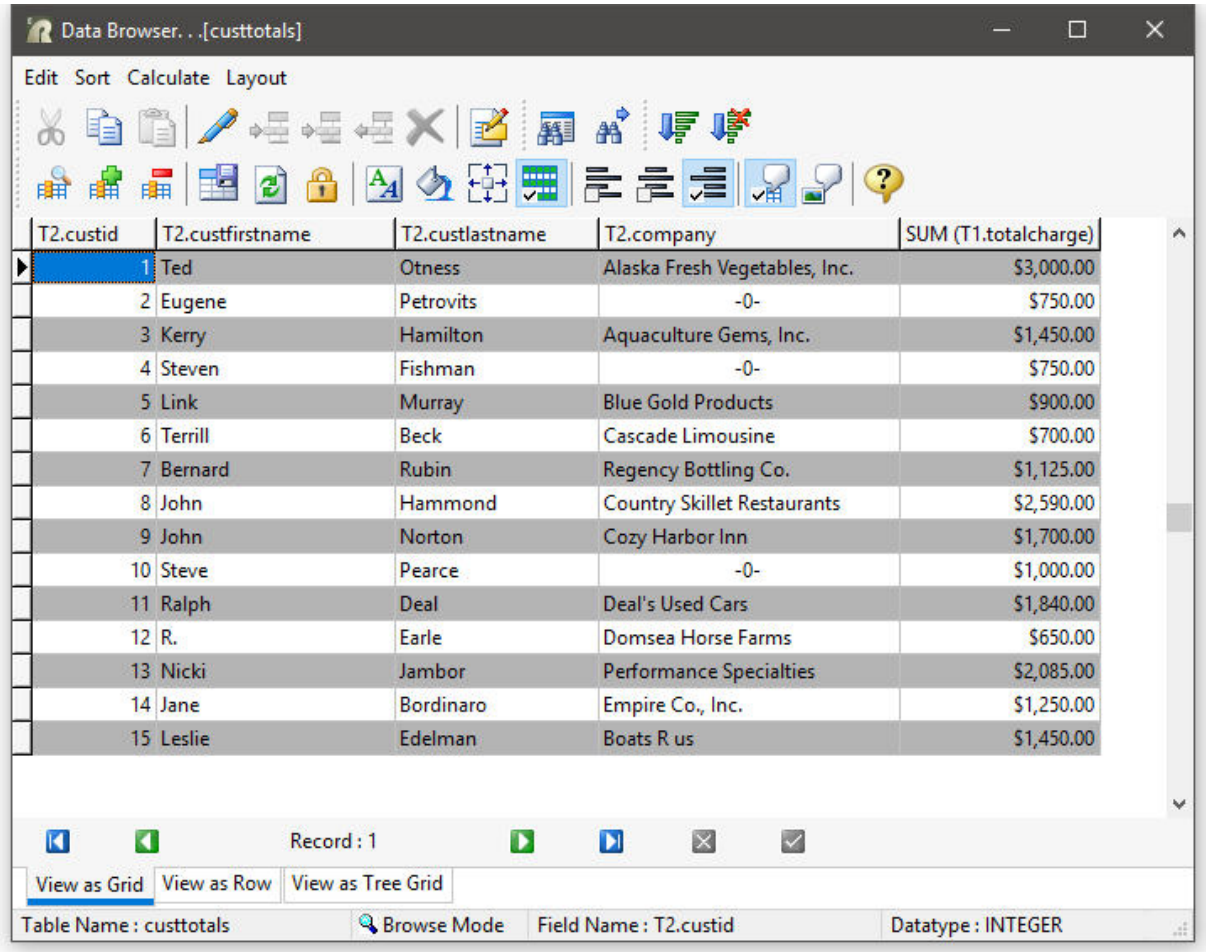
To Group a column or columns in your query:

1. Click the "Columns" tab in the lower pane of the Query Wizard.
2. Right click the "Group" row of the column *custid.*
3. Choose "Group" from the speed-menu.
4. Repeat this step for the *custfirstname, custlastname,* and *company* columns.

The text "Group" should appear in the columns.

The query that is geared to display customers grouped together with the sum of their total charges. To preview your query click the "Browse Query" button from the toolbar.

Your query will then be displayed in the Data Browser window. You may have to drag the window wider or drag the column lengths shorter to fit in the screen, but it should look like the picture below.



Close the Data Browser to return to the Query Wizard.

You can print the data to the printer by choosing any of the "Output Data" options.

To view the complete SQL statement for the new query, select the "Build SQL" button on the toolbar.

You can save the query as a view to review at a later time.

To save the query as a view:

1. Choose the "Save Query" button from the toolbar.

The "Save Query as" dialog will appear.

2. For the "View Name:" field, enter "CustTotals".
3. Enter "Customer Totals" in the "Comment" field.
4. Close the Query Wizard using the "Close" option.

The view will appear in the Database Explorer under Views.

4.16 Analyzing Data with Queries

In this section, we will learn how to make a more specific query by choosing the columns and limiting the rows that R:BASE retrieves from the database using a more defined WHERE Clause. Let's use Query Builder to build this query.

Building a query with the Query Builder:

1. From the Group Bar, select the "Views" option and select "New View".

A pop up will appear asking if you would like to use the Query Wizard.

2. Select No.

The "Query Builder" window is displayed.

On the left, the "List of Tables/Views" drop down menu provides option to displays tables, views, system tables, or all for the connected database. The right and bottom panels display portions of the query that make up what is being asked from the database. These panel contents will change as you build and modify the query. The "Tables/Views in Use" panel will display any tables/views that are added to the query. The bottom panel is the SELECT command syntax that is automatically generated during the query creation.

3. From the "List of Tables/Views" section, limit the list to "Tables", then select *flights* and click the "Add Table/View" button to add the table to the query.

"Flights T1" appears in the "Tables/Views in Use" panel. The "T1" is an alias name for the table *flights.* Each time a table is selected in the Query Builder, R:BASE gives the table an alias name. The alias number on your screen may not be the same as in this lesson. Also notice that the SELECT syntax has changed to include "FROM Flights.T1."

4.17 Selecting Columns for a Query

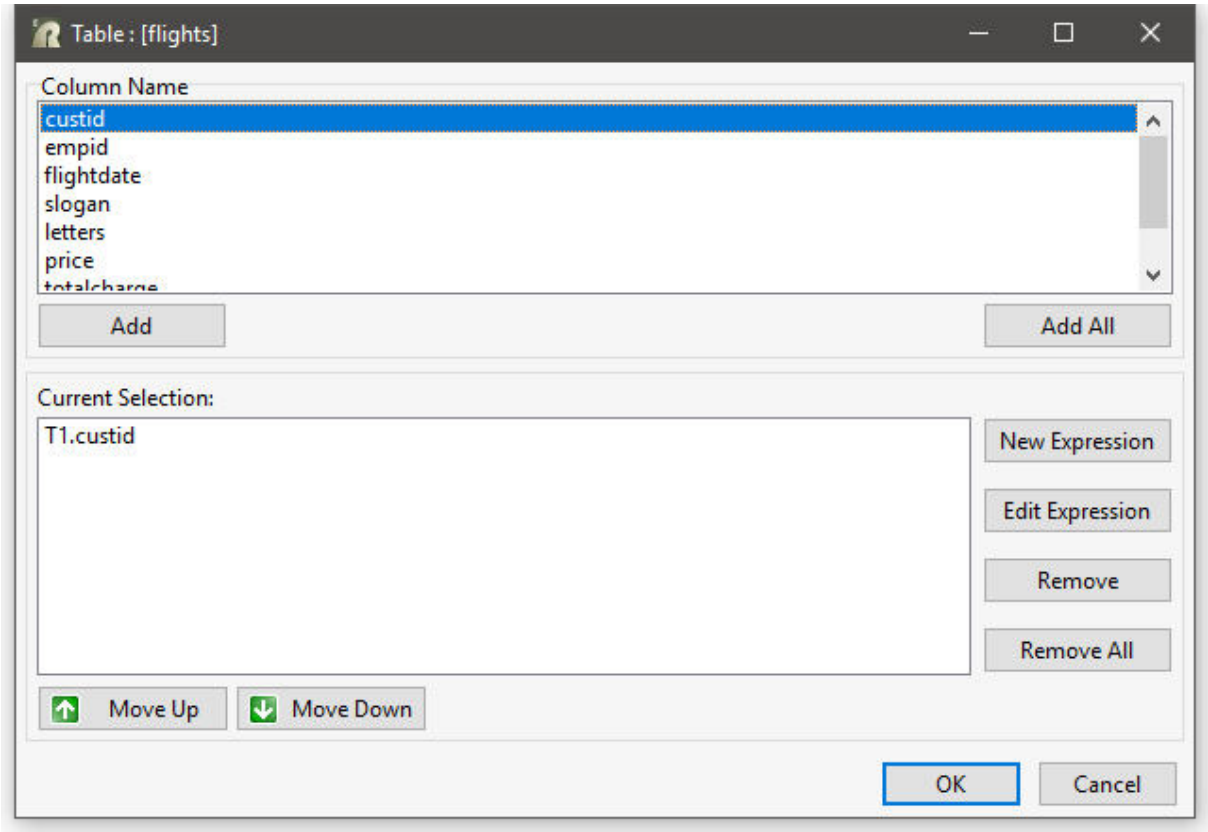
Because we are only interested in seeing the scheduled flights and not who booked the flight, let's select all the columns except the one that contains employee ID's.

To select the columns for the query:

1. In the "Table/Views In Use" panel, select the table "Flights.T1" and click the right mouse button.
2. From the speed menu, choose "Select Columns"

The "Table: [flights]" dialog box now displays the columns from the *flights* table.

3. Select "custid" from the "Column Name" section, then click the "Add" button. You will be asked to create an alias for the column name. Select "OK" and R:BASE will assign the column name for you. The column will appear under "Current Selection:".



The column is referred to as "T1.custid." The "T1" is the alias name for the *flights* table and explicitly identifies the column. Each column is referenced by the table's alias name.

Repeat this step for the following columns: *flightdate, slogan, letters, price, place,* and *totalcharge.* As you select and add each column, it is listed in the "Current Selections" section. The order in which the columns are selected is the order the columns will be displayed in the query.

4. Click the OK button.

The selected columns now appear in the SELECT syntax panel of the Query Builder window.

4.18 Specifying a Condition for a Query

Let's say that we would like to see a list of the flights scheduled for June 3, 2008. Since we don't want to see all of the data in the table, we need to limit the data that R:BASE retrieves. We'll, specify a condition to limit the data to a flight date of June 3, 2008.

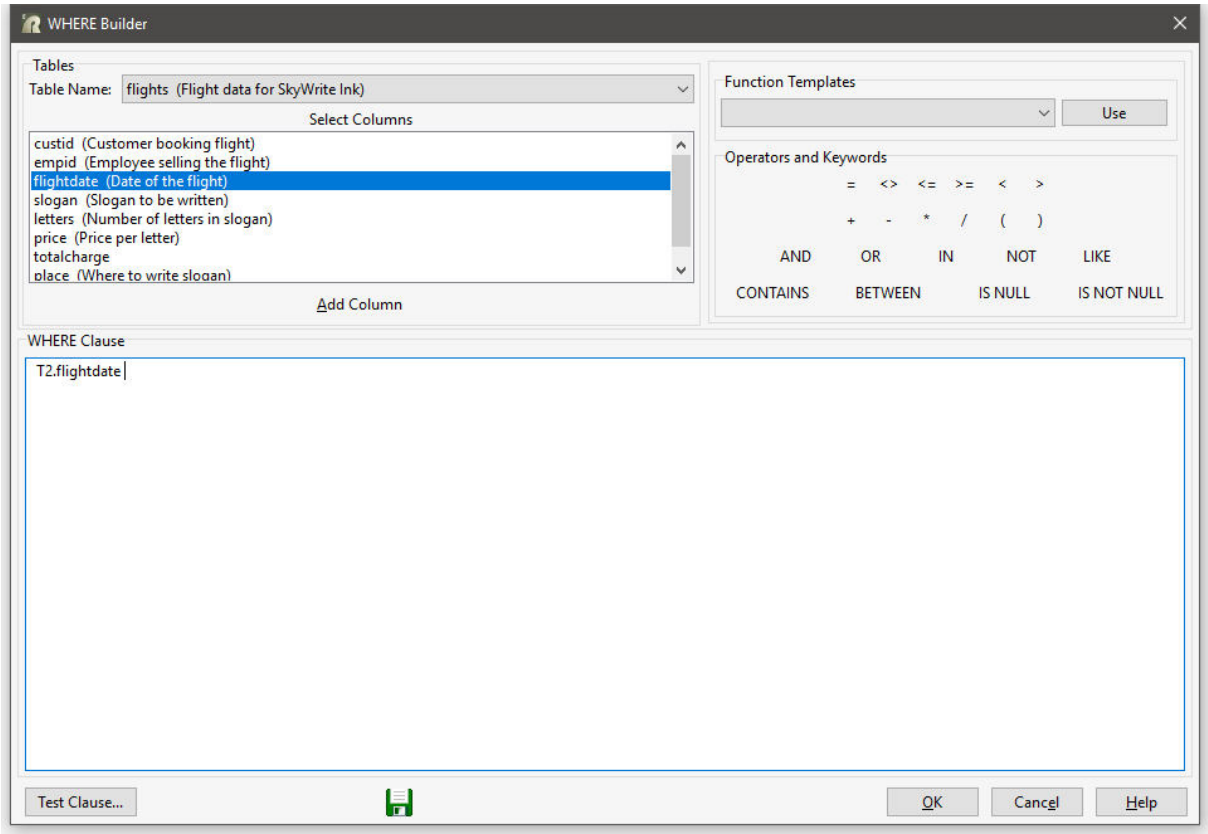
To specify a condition for the query:

1. In the "Query Builder" window, right-click on "flights T1" from the list of "Tables/Views In Use".
2. From the pop-up menu, choose WHERE Clause.

The Where Builder dialog box opens. The "WHERE Builder" lets you work with conditions. "Flights" is selected in "Table Name" and the columns from the *flights* table are listed in "Select Column."

3. Select "flight date" from "Select Columns" panel.
4. Click the "Add Column" button.

"T1.flightdate" is displayed in "WHERE Clause" panel.



Now we need to choose an operator. The table below describes each of the operators.

Operator	Description
=	Equal to
<>	Not equal to
>	Greater than
>=	Greater than or equal to
<	Less than
<=	Less than or equal to
BETWEEN	Greater than or equal to *value]* and less than or equal to *value2*
+	Addition, concatenation
-	Subtraction
*	Multiplication
/	Division
(	Left parentheses
)	Right parentheses
IN	Value of column exists in selected rows
NOT	Negation
IS NULL	Column contains no value (null)
IS NOT NULL	Column contains any value
LIKE	Column equals *string*
AND	The condition following AND must be met in addition to the condition before it
CONTAINS	Column contains a string called *string*

OR	Either the condition before OR or the condition following it must be met

We want to display only those rows that have the exact value of "06/03/2008" in the *flightdate* column.

5. In "Operators and Keywords", click the "=" button.

The operator "=" (equals) is added to the "WHERE Clause" text.

6. Enter "06/03/2008" in "WHERE Clause" text.
7. To test the syntax of your WHERE clause, click the "Test Clause..." button.

A dialog box displays the message, "Your Query tested OK!"

8. Click the OK button.

If an error message is displayed instead, click the OK button and correct the WHERE clause.

9. Click the OK button to close the "WHERE Builder" dialog box.

The WHERE clause you built in the WHERE Builder has now been added to the SELECT syntax panel.

To see the results of the query:

In the "Query Builder" window, choose **Query: Browse Query** from the main Menu Bar or select the "Browse Query" button on the toolbar.

R:BASE searches the *flights* table for all the rows that meet the condition and then displays the data in the "Data Browser" window.

There are five flights. The *empid* column is not displayed; only the selected columns are displayed in the order they were selected.

10. Close the Data Browser by selecting the "X" button or pressing the [Esc] key.

4.19 Modifying the Condition for a Query

Let's say you would like to see the flights for March 3, 2008 only. You can easily change the condition.

To change the rows retrieved by the query:

1. In the "Query Builder" window, right-click over "flights T1" from the list of "Tables/Views In Use" and choose WHERE Clause.
2. In "WHERE Builder" text, change the date to "03/03/2008".
3. Click the OK button.

To see the results of the query:

In the "Query Builder" window, choose **Query: Browse Query** from the main Menu Bar or select the "Browse Query" button on the toolbar.

R:BASE searches the *flights* table for all the rows that meet the condition and then displays the data in the "Data Browser" window. Close the window when finished.

Specifying More Than One Condition

Now we would like to see the flights scheduled for May that have a price per letter greater than $45.00.

You can create a query that places multiple conditions on your data. We will change the condition in the *flightdate* column and add a condition in the *price* column.

To use multiple conditions to retrieve data:

1. In the "Query Builder" window, right-click over "flights T1" and choose WHERE Clause.
2. In "Where Clause" text, delete "= 03/03/2008".
3. Click the BETWEEN button within "Operators and Keywords".

The "BETWEEN" operator is added to the WHERE Clause panel.

4. Enter "05/01/2008" in the "WHERE Clause" text.
5. Click the "AND" button within "Operators and Keywords".

The connector AND is added to the "WHERE Clause" text.

6. Enter "05/31/2008" in the "WHERE Clause" text.

By using the BETWEEN operator, we will display those flights that were scheduled from the 1st to 31st of May.

Now, we specify the second condition.

7. Click the "AND" button in "Operators and Keywords".

The connector AND is added to the "WHERE Clause" panel.

8. In the "Select Column" panel, select the "price" column and click the "Add Column" button.

The column *t1.price* is added to the WHERE Clause panel.

9. Click the ">" button within "Operators and Keywords"

The operator ">" (greater than) is added to the "WHERE Clause" panel.

10. Enter "45" in the "WHERE Clause" text.

You have finished building the WHERE clause. You can test the WHERE clause if you want by clicking the "Test Clause..." button. The query should test correctly.

11. Click the OK button to close the WHERE Builder.

To see the results of the new query:

In the "Query Builder" window, choose **Query: Browse Query** from the main Menu Bar or select the "Browse Query" button on the toolbar.

R:BASE displays the information you specified in the "Data Browser" window. As you browse the rows you should see only those flights scheduled in May with rates higher than $45.00 per letter. Close the Data Browser when you are finished.

4.20 Printing the Query Results

You can print the results of your query to a printer. Be sure you are connected to the printer and that your printer is online.

From the Menu Bar, select **Query: Output Data: Printer**

R:BASE prints the data. You now have a copy of your data showing the flights for May that have a price per letter greater than $45.00. The data is sorted by the flight date and the customer ID.

You can save the query as a view just as done previously. Otherwise, close the Query Builder and do not save the query.

4.21 Creating Views

A view is a collection of columns from the available tables. Although you can think of a view as a temporary table, a view does not store data. A view is a stored query. The query specifies the tables, columns, and linking columns that make up the view. Each time you use the view, R:BASE reconstructs it from the query.

You've learned how to look at data from individual tables in the *sky* database and how to find the information you want in those tables. But what if the answer to a question requires information from more than one table in *sky?* To answer a question like "Which salesperson booked flights for the Cascade Limousine company?" you'll need to query multiple tables. By learning to use multi-table queries and views, Amelia, Charlie, or anyone in skyWrite Ink can answer specific questions quickly.

While designing and building the database in Lessons One and Two, we stressed that tables should share common columns. Each of the three tables we created has a separate theme: *employees*, *customers*, and the *flights* made for customers. Each table is related to another table by at least one common column: the *empid* column or the *custid* column. Now you will see the benefits of these linking columns.

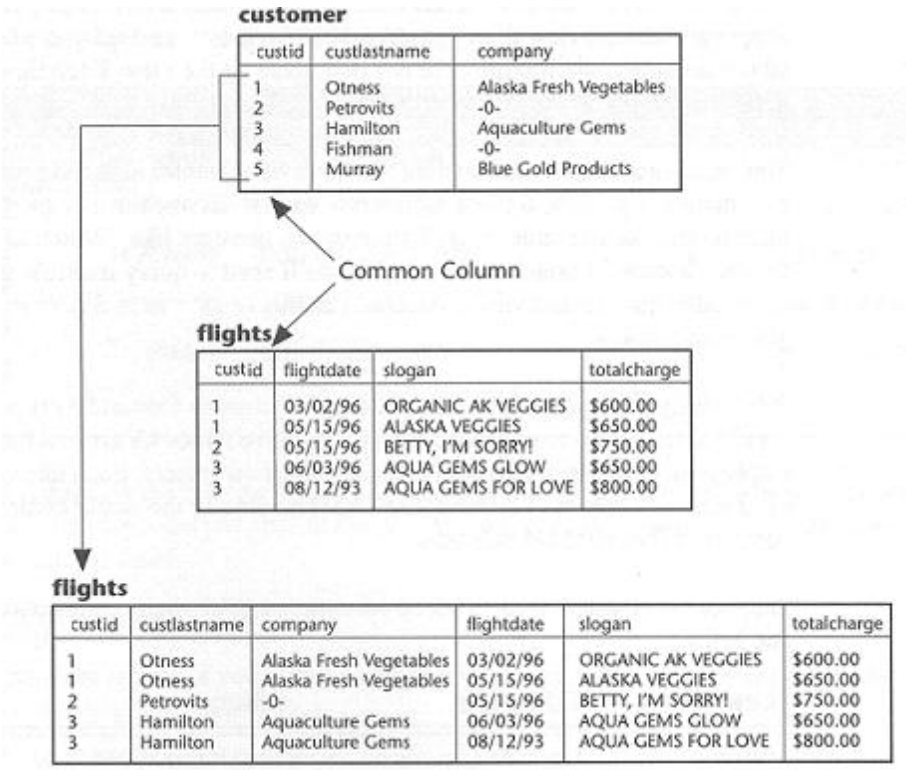
Suppose you wanted to see a list of customers and all their corresponding flights like the one below:

Last Name	Company	Slogan	Total Charge
Beck	Cascade Limousine	LIMOS 555-9987	$700.00
Earle	Domsea Horse Farms	HORSE 'n AROUND	$650.00
Rubin	Regency Bottling Co.	R POP HITS THE SPOT	$675.00

The *flights* table contains all the flight information but shows only the customer number, not the customer's name or company. That information is contained in the *customer* table.

We can gather the information we want from each table by creating a query that combines the two tables. Once the query is defined, we can save it as a view. Then we can use the view as much as we would use a table in the database.

The illustration below shows how the rows and columns in the *flights* and *customer* tables combine to make *flightview,* the view we'll create in this lesson.



The two tables combine based on their common column, *custid.* When you link the two tables in the query, R:BASE compares each row in the *customer* table with each row in the *flights* table. When rows in the *custid* column match, R:BASE creates a row in the view that contains the columns from both tables. For example, when *custid* contains a "1" in both tables, R:BASE adds a row to the view. Because the relationship between the *customer* and *flights* tables is one-to-many, the view will contain multiple rows for customers who have more than one flight.

To create a view, first build a query that defines the view. Creating a multi-table query requires three steps:

- Choosing the tables containing the information you want to see
- Linking the tables through common columns
- Choosing the columns containing the information you want to see

Once you have created the query, you can save it as a view. You build the query and save the view in the "Query Builder" window. Remember, a view is just a stored query.

4.22 The Query Builder

The Query Builder is used to create a new view or modify an existing view definition.

To build a new view:

1. Begin at the Database Explorer. If the *sky* database is not connected, click the Databases option and select *sky.*
2. Click the Connect option.
3. Click the "Views" option from the Group Bar.
4. Click the "New View" option.

R:BASE displays the "Do You Want to Use the Query Wizard?" dialog box.

5. Select No.

The "Query Builder" window opens.

4.23 Selecting Tables for a View

The first step in the "Query Builder" window is to select the tables to include in the query. Each table you select is displayed in the "Tables/Views in Use" panel. Remember, this is the same window you used to create a condition for your query earlier in Lesson 3.

To select the tables for the view:

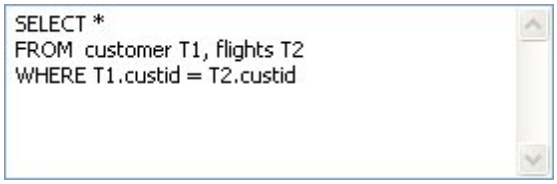
1. In the "Query Builder" window, select "customer" from "List of Tables/Views"
2. Click the "Add Table/View" button.

R:BASE displays "customer T1" in the "Table/Views In Use" panel. The syntax "FROM Customer T1" is also added to the SELECT syntax panel. "T1" is the alias name R:BASE assigned to the *customer* table. The number on your table alias may not match to the "T1" in this example.

Next, we'll add the second table to the query.

3. Select "flights" in "List of Tables/Views" and click the "Add Table/View" button.

R:BASE displays "flights T2" in a second box in the workspace. R:BASE adds "flights T2" to the FROM part of the query. "T2" is the alias name R:BASE assigned to the *flights* table. The WHERE part of the query has also changed to display the link between the tables. The number on your table alias may not match to the "T2" in this example.



4.24 Linking Tables

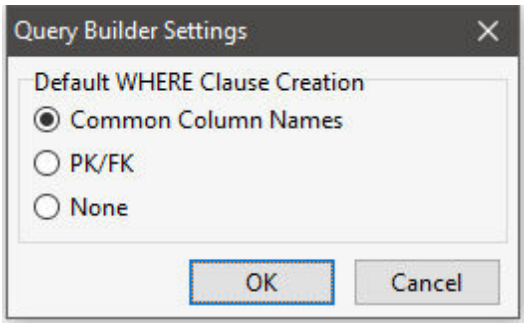
When you retrieve information from more than one table, the information must be related in some way, so you must specify the link between the tables.

You have three options when specifying links between tables:

- R:BASE can automatically link the tables' common columns, or columns with the same name. Using the example above, the common column between the *customer* table and the *flights* table is the *custid* column. This method is the default.
- R:BASE can automatically link tables with primary key and foreign key constraints. To use this option, constraints must be defined. However, columns do not have to have the same name. For more information about constraints, see Defining Primary and Foreign Keys.
- You can link the tables yourself when you specify the conditions for the query. For more information about specifying query conditions, see "Specifying a Condition for a Query".

The table link is displayed in the "WHERE Clause" portion of the SELECT syntax panel.

To change between the above options, choose "Query" from the Main Menu bar, and then select "Query Builder Settings". The following dialog will appear allowing you to edit the table linking properties.



4.25 Selecting Columns for a View

Next, select the columns from each table to include in the view. We'll select the *custid, custlastname,* and *company* columns from the *customer* table first.

To select the columns from the *customer* table:

1. In the "Query Builder" window, select the table "customer" from the list of "Tables/Views In Use" and click the right mouse button.
2. From the speed menu, choose "Select Columns".

The "Table: [Customer]" dialog box is displayed. The columns from the *customer* table are listed in the "Column Name" panel.

3. Select "custid" in "Column Name" and click the Add button.

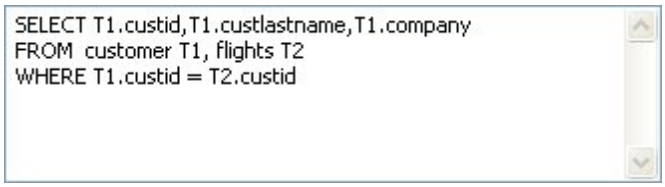
You will be asked to create an alias name for the column. Select "OK", and R:BASE will assign the column alias name for you. R:BASE includes the table alias to explicitly connect each column to its table. "T1.custid" is added to "Current Selection:".

4. Repeat this step to add *custlastname* and *company.*

"Current Selection" lists the three columns 'T1.custid," "T1.custlastname," and "T1.company." The alias name "T1" identifies these columns as belonging to the *customer* table.

5. Click the OK button.

The SELECT syntax should be displayed like the following:



You have now selected all the columns from the *customer* table that will be included in *flightview.* Next, we'll select the columns we want from the *flights* table.

To select the columns from the flights table:

1. In the "Query Builder" window, select the table "flights" from the list of "Tables/Views In Use" and click the right mouse button.
2. From the speed menu, choose Select Columns.

The "Table: [flights]" dialog box is displayed. The columns from the *flights* table are listed in "Column Name."

3. Select the following columns from *flights* to include in the view: *flightdate, slogan, place,* and *totalcharge* and do not add any column aliases.

The columns you select are identified with the table alias "T2," the alias name R:BASE assigned to the *flights* table.

4. Click the OK button.

The SELECT syntax should be displayed like the following to include the columns selected from the *flights* table:

```
SELECT T1.custid,T1.custlastname,T1.company,T2.flightdate,T2.slogan,T2.place,T2.totalcharge
FROM  customer T1, flights T2
WHERE T1.custid = T2.custid
```

You will notice that R:BASE automatically recognized the common column and established the link. The WHERE clause in the SELECT syntax panel reads "T1.custid = T2.custid".

The query for this view is complete. Before we save the query as a view, let's look at the results of the query.

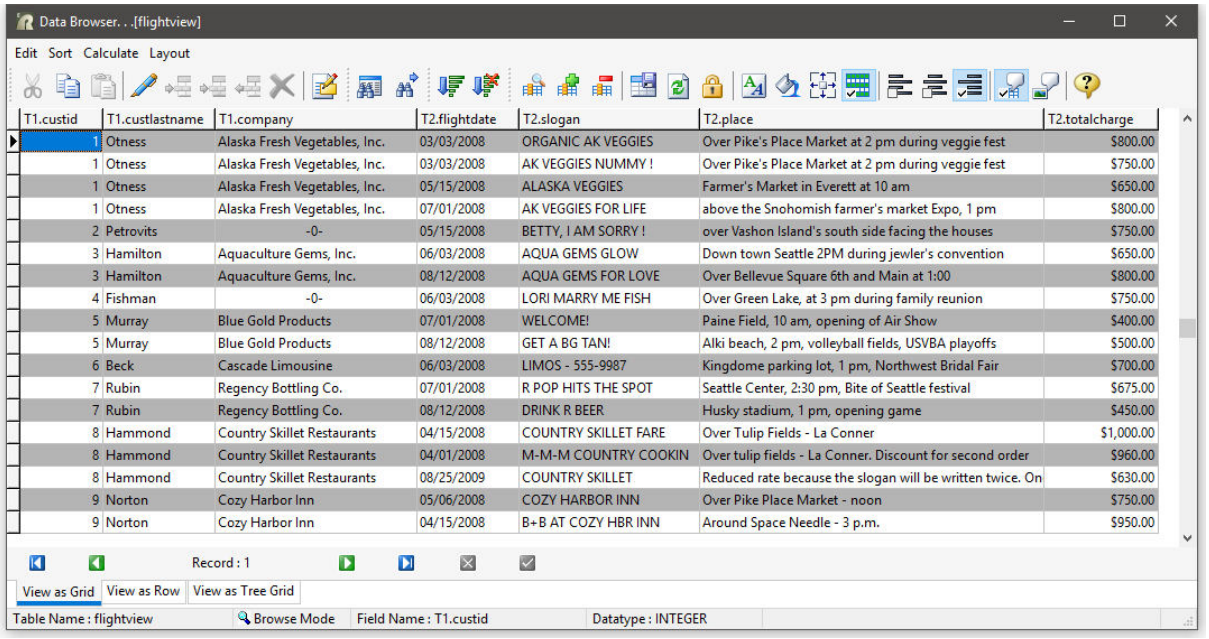
4.26 Looking at the Query Results

As you are building a query, you can switch to the "Data Browser" window to see the data the query is retrieving.

To look at the results of the query you've created to define flightview:

Choose **Query: Browse Query**.

R:BASE displays the data retrieved by the query in the "Data Browser" window.



T1.custid	T1.custlastname	T1.company	T2.flightdate	T2.slogan	T2.place	T2.totalcharge
1	Otness	Alaska Fresh Vegetables, Inc.	03/03/2008	ORGANIC AK VEGGIES	Over Pike's Place Market at 2 pm during veggie fest	$800.00
1	Otness	Alaska Fresh Vegetables, Inc.	03/03/2008	AK VEGGIES NUMMY !	Over Pike's Place Market at 2 pm during veggie fest	$750.00
1	Otness	Alaska Fresh Vegetables, Inc.	05/15/2008	ALASKA VEGGIES	Farmer's Market in Everett at 10 am	$650.00
1	Otness	Alaska Fresh Vegetables, Inc.	07/01/2008	AK VEGGIES FOR LIFE	above the Snohomish farmer's market Expo, 1 pm	$800.00
2	Petrovits	-0-	05/15/2008	BETTY, I AM SORRY !	over Vashon Island's south side facing the houses	$750.00
3	Hamilton	Aquaculture Gems, Inc.	06/03/2008	AQUA GEMS GLOW	Down town Seattle 2PM during jewler's convention	$650.00
3	Hamilton	Aquaculture Gems, Inc.	08/12/2008	AQUA GEMS FOR LOVE	Over Bellevue Square 6th and Main at 1:00	$800.00
4	Fishman	-0-	06/03/2008	LORI MARRY ME FISH	Over Green Lake, at 3 pm during family reunion	$750.00
5	Murray	Blue Gold Products	07/01/2008	WELCOME!	Paine Field, 10 am, opening of Air Show	$400.00
5	Murray	Blue Gold Products	08/12/2008	GET A BG TAN!	Alki beach, 2 pm, volleyball fields, USVBA playoffs	$500.00
6	Beck	Cascade Limousine	06/03/2008	LIMOS - 555-9987	Kingdome parking lot, 1 pm, Northwest Bridal Fair	$700.00
7	Rubin	Regency Bottling Co.	07/01/2008	R POP HITS THE SPOT	Seattle Center, 2:30 pm, Bite of Seattle festival	$675.00
7	Rubin	Regency Bottling Co.	08/12/2008	DRINK R BEER	Husky stadium, 1 pm, opening game	$450.00
8	Hammond	Country Skillet Restaurants	04/15/2008	COUNTRY SKILLET FARE	Over Tulip Fields - La Conner	$1,000.00
8	Hammond	Country Skillet Restaurants	04/01/2008	M-M-M COUNTRY COOKIN	Over tulip fields - La Conner. Discount for second order	$960.00
8	Hammond	Country Skillet Restaurants	08/25/2009	COUNTRY SKILLET	Reduced rate because the slogan will be written twice. On	$630.00
9	Norton	Cozy Harbor Inn	05/06/2008	COZY HARBOR INN	Over Pike Place Market - noon	$750.00
9	Norton	Cozy Harbor Inn	04/15/2008	B+B AT COZY HBR INN	Around Space Needle - 3 p.m.	$950.00

R:BASE has displayed a row of data for each flight. There are multiple rows for customers who purchased multiple flights. You can see, for example, that Otness has purchased four flights for Alaska Fresh Vegetables. Let's save our query as a view.

Close the Data Browser window by selecting the "X" button, or by pressing the [Esc] key.

4.27 Saving the Query

You can save a query as a view.

To save the query as a view:

1. In the Query Builder window, choose **File: Save Query as View** from the Menu Bar.

R:BASE prompts you for the "View Name:" and "View Comment:".

2. Enter "*flightview*" as the new view name.
3. Enter "Customer/Flight" as the comment.
4. Click the OK button.

When you save the query as a view, you are only saving the query. The advantage to this is that each time you use the view; R:BASE includes any new data that you've added to the tables that make up the view.

Now we'll close the "Query Builder" window.

To close the "Query Builder" window:

In the Query Builder window, choose **File: Close** from the Menu Bar.

The view is saved and is listed in the Database Explorer.

To open a view in the Data Browser:

In the Database Explorer window, highlight the view *flightview*, and select the "Open View" option.

R:BASE displays the data retrieved by the query in the Data Browser window.

To close the Data Browser window, select the "X" button, or by press the [Esc] key.

4.28 Summary

In this lesson you have:

- Hidden, moved, and locked columns
- Changed the display size of a column
- Added and deleted data from a table
- Calculated a tally for a column
- Sorted the rows of data
- Built single and multiple condition queries
- Printed the results of a query
- Created a multi-table query by linking two tables
- Saved the query as a view
- Selected the rows R:BASE displays in the view by specifying a condition

To check your knowledge, try answering the following questions:

1. Which window displays the data in a table?
2. Which window lets you specify query conditions?

3. Which menu should you choose from when you want to change the appearance of displayed information?
4. What is a query?
5. Why do you specify conditions?
6. What are the two essential tasks you need to perform when building a multi-table query?
7. How do you save a query?
8. In what way is a view different from a table?

Part V

5 Lesson 4 - Creating and Modifying Forms

A form is a screen display that allows you to enter, edit, and display database information. In this lesson you'll learn how to use the Form Designer to create and modify forms in R:BASE. The Form Designer provides many options for creating attractive custom forms ranging from simple one-table forms to complex multi-table, multi-tab forms.

At the end of this lesson you will be able to:
- Start the Form Designer
- Create a quick form in the Form Wizard
- Create a custom form
- Place and move fields
- Create variables and expressions
- Customize the form and its fields
- Enter data using the forms you created

Charlie and Amelia will need an easy way to add information to their database. The forms you build here will let them keep their employee records and flight information up to date. The form for flights, *flightform,* will allow their salespeople to enter flight information without having to learn anything about database management.

5.1 Designing a Form

Forms are used for three purposes: for entering new data into tables, for editing existing data, or for viewing data. A form should be logical and easy to read. Often, data is entered by an operator who is reading the data from a paper form. The computer form should be designed to match the paper form for efficient data entry. A well-designed form also provides a way of viewing existing data in a format that is familiar to the user. There are three steps to follow when designing your form:

1. Identify the purpose it will serve: Will it be used for entering, editing, or viewing data?
2. Draw the form on paper, placing text and fields.
3. List specific customizations such as tables to be referenced, or colors to be displayed on the form.

5.2 The Form Designer

The Form Designer is where you build a form. In the "Form Designer" window, you lay out your form, place objects to display text and data, and customize the look of your form. Once forms have been defined, the form names are listed in the Database Explorer.

To open the Form Designer:

1. In the Database Explorer, connect to the *sky* database, if it is not already connected, click the Databases option, select *sky,* and then click Connect.
2. Select the "Forms" option.
3. Select the "New Form..." option.

The "New Form" dialog box is displayed. You will now create a quick form using the Form Wizard.

5.3 Creating a Quick Form with the Form Wizard

There are two methods to create forms: a quick form using the wizard and a custom form. Here, we'll build a form with the wizard. For more information about custom forms, see "Creating a Custom Form" later in this chapter.

The Form Wizard feature builds a single-screen entry form, which can serve many immediate data-entry needs. Using the Form Wizard is the quickest and easiest method of providing a data entry screen. However, since the wizard does not take advantage of most of the features available in a form, you will want to modify it later.

This new form being created will allow you to enter and edit employee information in the *employee* table in the *sky* database.

To build a form for the employee table:

1. In the "New Form" dialog box, enter "empform" as the name of the form in "Form Name".
2. Select "employee" from "Table/View Name".
3. Enter "Enter and edit employee data" in "Form Comment".
4. Select the "Use Form Wizard" check box, and select OK.

You will be asked to create a "Single-Table Form" or a "Master/Detail Table Form" as well as well as other options.

5. Keep "Single-Table Form" selected, add a check to the "Use Field Comment as Caption" check box, and click "Next".

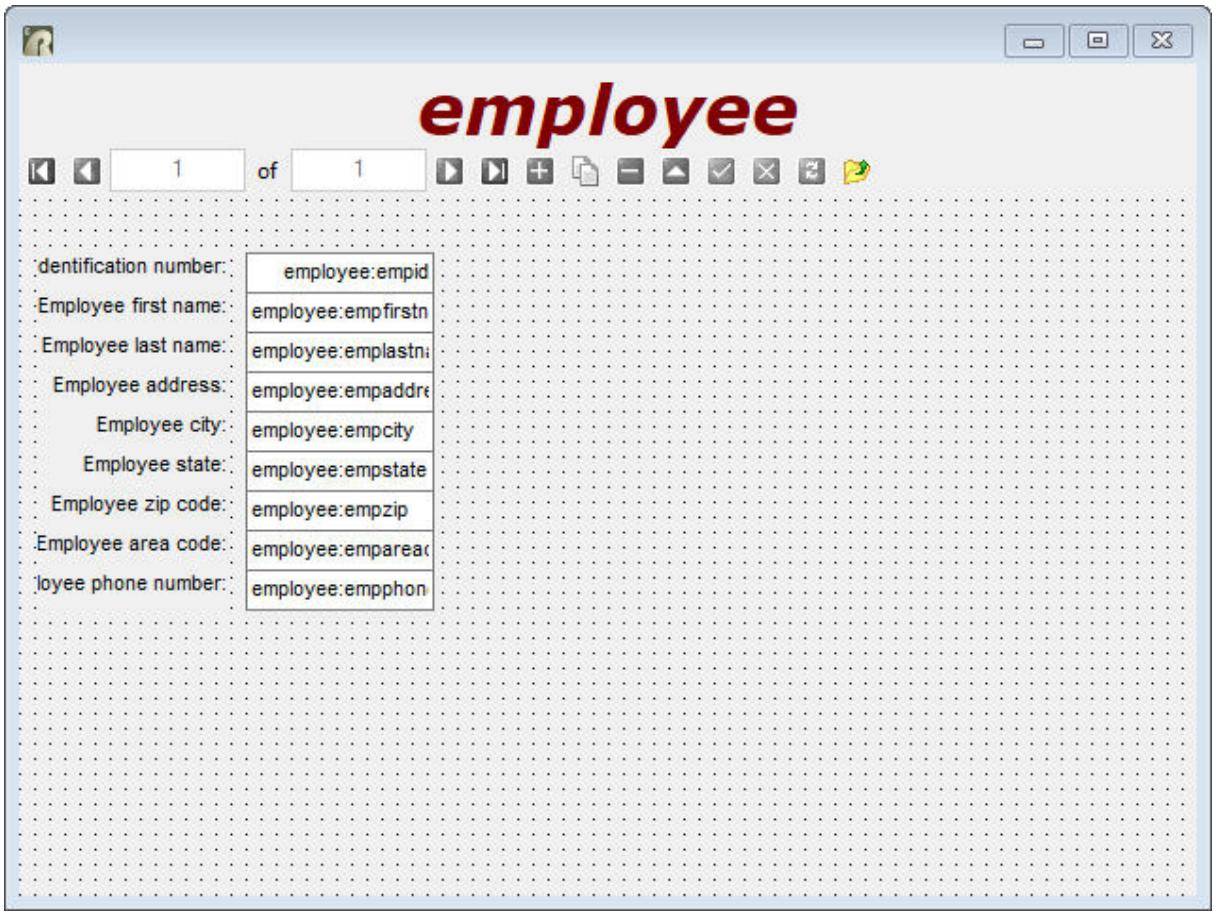
R:BASE displays a list of the columns in the *employee* table. Using the arrows, you can move any columns that you want to appear on the form. You can also use the up and down arrows to place the columns in any particular order.

6. Use the double arrow pointing right to add all the columns to the form, and click "Next".

You are now given four options as to the field arrangement of the columns on the form, Horizontally, Vertically, Grid, and Scrolling Region. You can preview how the form will look by selecting each option.

7. Select "Vertically", and click "Next".
8. Click the "Finish" button to build the form.

R:BASE builds the form, displaying it in the "Form Designer" window.



5.4 Understanding the Form Designer Interface

Now that the form is created, it is time to get familiarized with the Form Designer toolbars.

With the Form Wizard, R:BASE placed the areas where users will enter data, along with text, and a navigator bar, across the top under the employee title, with arrows and other pictures grayed out. These different areas are called field objects. There are many different kinds of field objects that can be placed on forms. The field objects are categorized by control type, within the "Object List" toolbar, which provides options for adding new and reviewing existing form controls. Here is a list of the form toolbar control types:

- Standard Controls
- Database Controls
- Variable Controls
- Additional Controls
- Legacy Controls
- Internet Controls
- File System Controls

Adding New Objects
The Object List toolbar offers increased organization and navigation of possible and existing form controls. Control Groups can be collapsed/expanded and control names shown/hidden. If defined, the "Favorites" will always be located on the top of the list. The "Filter Controls" button allows users to easily find and add available controls, with the help of the component filter, e.g. typing "button" reduces the list to only button-type controls. The arrow button provides the ability to deselect a form control.

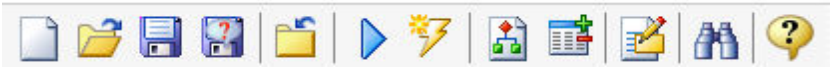
Existing Objects
The "Existing" tab of the toolbar is very helpful in finding form objects, and the properties of the objects, especially in forms where a great number of objects are placed. The controls can be filtered by type and parent control. Columns provide information on the class, description, Component ID, EEPs, and Popup Menus. The EEPs column displays a gold star image if a Custom EEP exists. A blue star is displayed for external EEP files. A green star is displayed within the Pop-up Menu column, if exists. Locked controls are listed with bold text and a gray background color applied.

Toolbars
There are other toolbars to alter the form or the objects placed upon it. Those toolbars are listed here:

- Form
- Edit
- Settings
- Object Inspector
- Format
- Align or Space
- Size
- Nudge

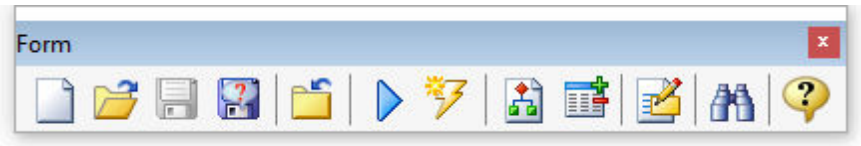
While in the Form Designer, you can position toolbars on any edge of the client window and in any order. Detaching a dockable toolbar from the frame window is called floating the toolbar. Attaching a floating toolbar to the frame window is called docking the toolbar. A toolbar can be docked to any side of its parent window, or can be floated in its own mini-frame window. Whether the toolbars are docked or floating, you will see the hint for any toolbar button when the cursor is hovered over the button. The Form Designer will remember the last docked/floating setting when closing the designer.

When a toolbar is docked, it will look like this:



On the left side a series of characters are displayed to drag the toolbar into a floating position. The "character" can be one or more dots or pipe symbols "||" based on your operating system and current theme. The dots/pipes are used to drag the toolbar into a floating position.

When floating, the toolbar can be moved by clicking and dragging the caption area and can be resized by clicking and dragging the edge. When a toolbar is floating, it will look like this, where a border is wrapped around the toolbar and the name and a close button (x) is displayed in the caption:



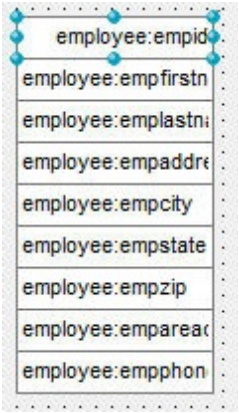
From inside the Report Designer, you can launch the Report Designer help file by pressing the [Shift] +[F1] hot keys, or by pressing the "Help" button on the "Form" toolbar (above).

Tips:

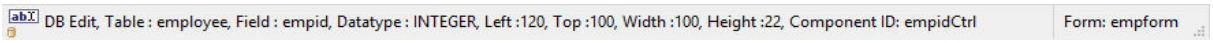
- Double click on the caption area of a floating toolbar to dock it to the last location it was docked
- When floating, the "x" is used to hide a toolbar
- The toolbars can be docked on the sides or bottom of the screen to provide a greater work space
- From the Menu Bar, select the View: Toolbars option to hide or show any toolbar

5.5 Field Object Properties

With the Form Wizard, R:BASE has placed a number of field objects with the table name and the column name with a white background and black text. These objects represent the column data for that table for entering, editing, and browsing data. These objects are very commonly used in the Form Designer and are called DB Edit objects.

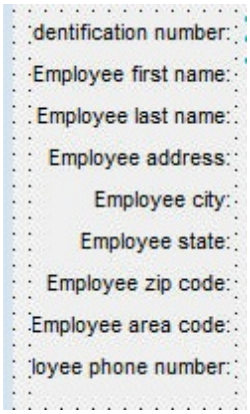


If you select one of the DB Edit objects by clicking on it "Handlebars" appear on the selected object. While it is selected, you will see the status line across the bottom of the screen will display information about that field object. In the lower left corner the numbers indicate the position of the selected object. The "Left" number tells the number of pixels across the window. The "Top" number tells the number of pixels down the window from the upper left comer. For example, Left: 64 Top: 10 means the upper left comer of the selected object is 10 pixels from the left edge and 64 pixels down from the top of the window. The Width and Height numbers indicate the width and height of the selected object, in pixels.

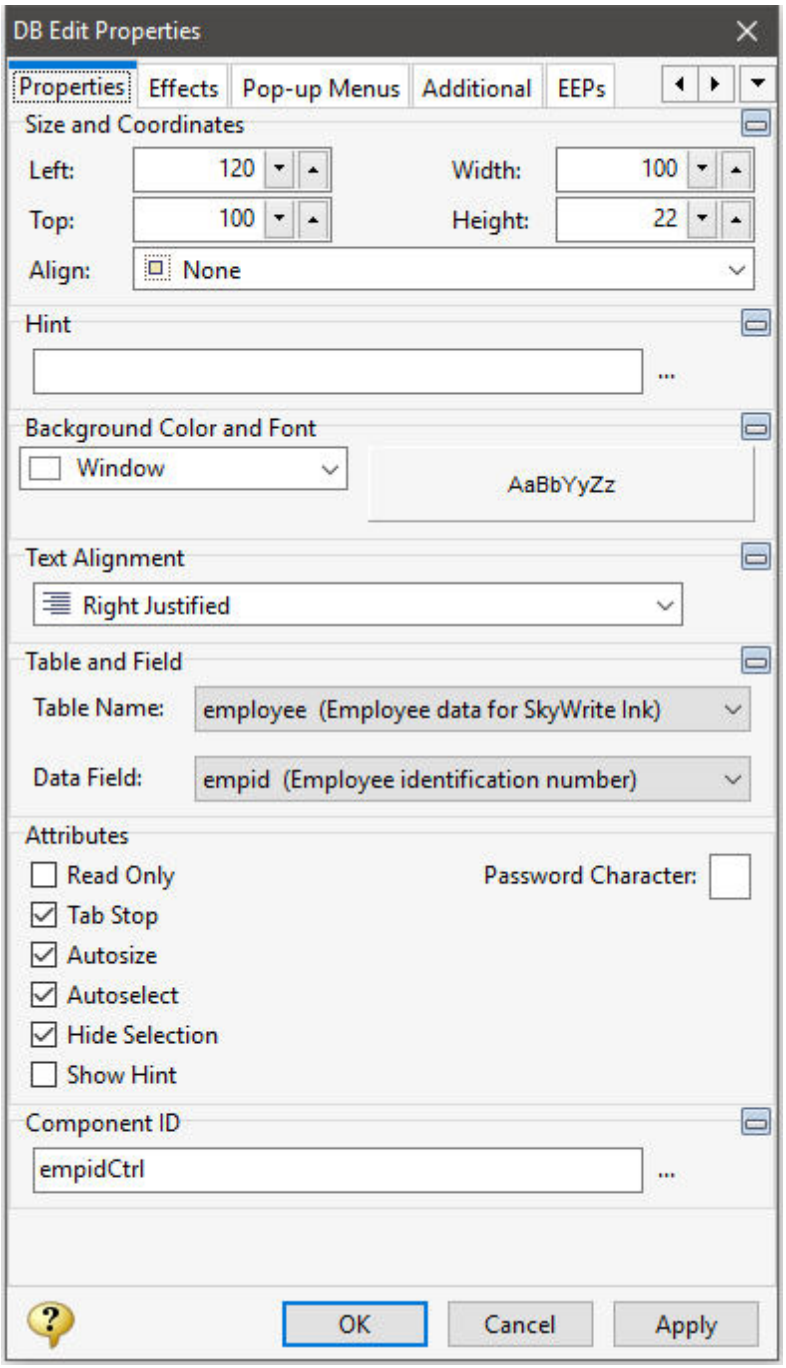


The status line also displays the type of the selected object and detailed information relating to the object. A DB Edit object will display the name of the table the form serves and the column name. The image of the status line above displays that a DB Edit object is selected. The object is used to represent the data in the *empid* column in the *employee* table. A selected Static Text object will display the text caption.

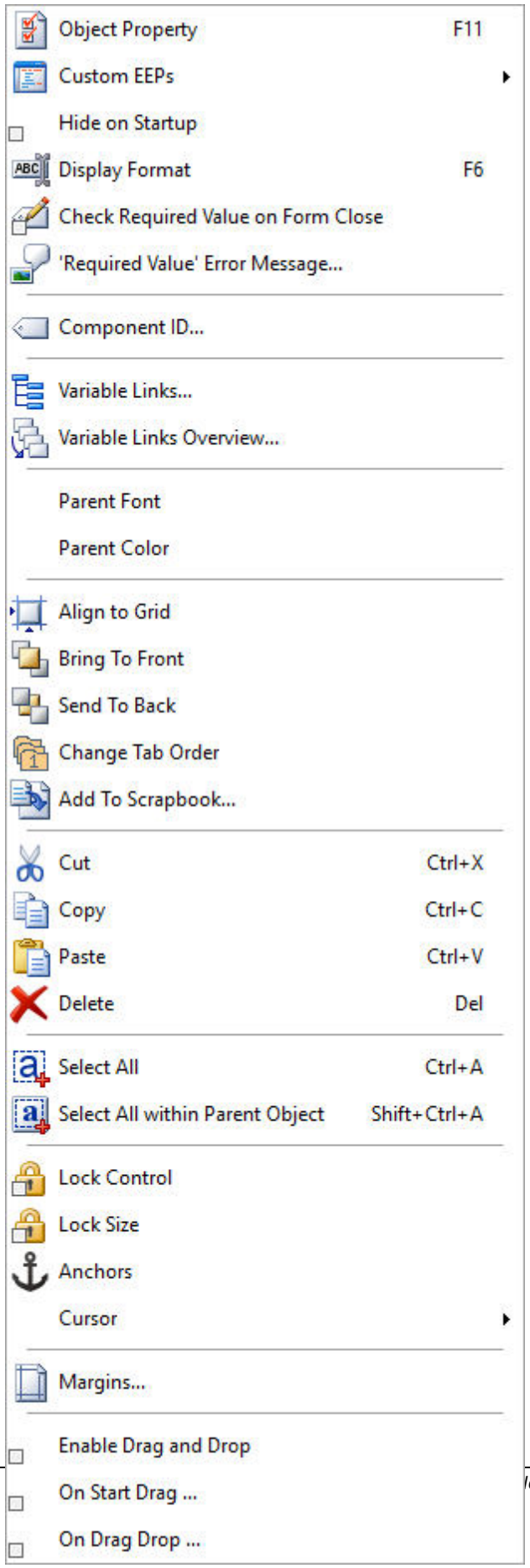
R:BASE has also placed a number of field objects with black text and a gray background next to each DB Edit object. These objects are referred to as Static Text and are also very commonly used in the Form Designer. These objects represent any text, which you would want to appear on the form. The Static Text objects can be used to describe another field or provide instructions for the end users. The R:BASE Form Wizard has placed the column comment as the caption for the Static Text. The title "employee" across the form is also Static Text. The caption for the title Static Text is the table name by default using the Form Wizard.



For all objects that can be placed on forms, there are many various properties that can be changed to enhance the visibility of the object. The properties for all field objects can be edited by right clicking on the object and selecting "Object Property" from the speed menu, or by selecting the object and pressing the [F11] key on the keyboard. For instance, in most field object properties, you can edit the text, color, font, size, justification, hint, effects, read only status, and much more. Each object's properties vary depending on the type of control.



Right clicking on an object will display the Speed Menu options. The Speed Menu will vary on different objects.



5.6 Customizing the Form

You can use *empform* just as it is. However, with very little effort, you can customize the form to make it easier to use.

In the next sections of this lesson, we'll modify the form so that it looks like the one below.

5.7 Editing Text Objects

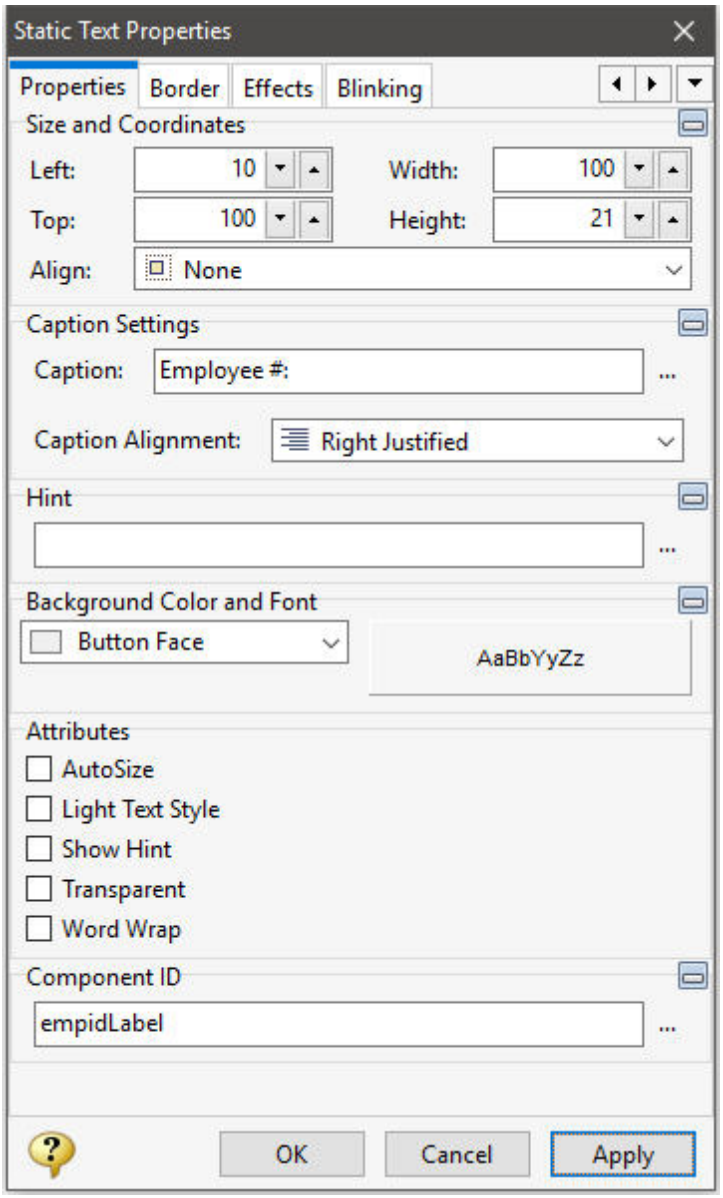
First, we'll edit the Static Text objects that label the column fields. By changing the text, we can make the field labels more meaningful.

To change the text label for empid:

1. In the "Form Designer" window, select the "empid" text object, right-click, and choose "Object Property".

The "Static Text Properties" dialog box is displayed.

2. From the "Properties" tab, in the "Caption" field, replace "Employee identification number:" with "Employee #:".



3. Click the OK button.

With the "Use Field Comment as Caption" check box selected during the Form Wizard, the remaining captions in the other text objects would not need altered, unless you prefer to do so.

5.8 Moving Objects

When you want to move an object to a different place on the form, you select the object with the mouse and drag it to the new location, which is called "drag and drop." You can move objects individually or in groups. When moving objects it is also helpful to use the ruler which is located across the top and along the left side of the form. The measurements on the ruler are screen pixels.

In this section, we'll move the objects to different places on the form to separate the groups of information and make the form easier to read.

To move all the objects on the form:

1. In the "Form Designer" window, lasso the Static Text and DB Edit objects.

All the objects on the form are selected as a group.

2. With the mouse cursor on any one of the selected objects, hold the left mouse button down. Drag the grouped objects to the center of the form. Release the mouse button.
3. Click the mouse cursor outside the objects to deselect them.

Now, we'll move the phone number text and DB Edit objects to the same line and to the right of the area code objects.

To move the phone number objects:

1. In the "Form Designer" window, click on the "Employee phone number:" Static Text object.
2. Press and hold the [Shift] key, then click on the DB Edit object for *emphone.*

The two phone number objects are enclosed with red handlebars. This indicates that they are selected as a group.

3. With the mouse cursor within the fields, click and hold the left mouse button. Drag the objects to the right of the area code objects. When dragging the objects, you will notice guidelines automatically appear, which assist is the horizontal and vertical alignment to other objects. Horizontally align the selected phone number Static Text and DB Edit objects to the area code objects, and release the mouse button.
4. Click the mouse cursor outside the objects to deselect them.

Next, we'll separate some of the objects on the form from the other fields. First, we'll separate the area code and phone number section.

To separate the area code and phone number section:

1. In the "Form Designer" window, click on the *empphone* DB Edit object.
2. Press and hold the [Shift] key, then click on the "Employee phone number:" Static Text object, the *area code* DB Edit object, and the "Area Code" Static Text object.

The four fields are now enclosed with red handlebars, indicating they are selected as a group.

3. With the mouse cursor inside the group, click and hold the left mouse button. Drag the group of fields down about an inch so they are near 400 pixel measurement on the Form Designer ruler. When dragging the objects, you will notice the guidelines appear again. Use the guidelines to vertical alignment the DB Edit objects. Release the mouse button.
4. Click the mouse cursor outside the fields to deselect them.

Using this same procedure, separate the address information from the employee number and name, placing the Address objects between the name and phone number objects.

Now that the object are moved to new locations, we'll adjust the field sizes.

If you have not done so already save the form. From the Menu Bar, choose **Form: Save Form**.

5.9 Changing the Size of Objects

When using the Form Wizard, all of the column objects are the same size, but the columns in the table associated with the form typically allow you to enter different amounts of data. For example, the *empstate* column allows entry of 2 characters, the *empaddress* column accepts up to 30 characters. We'll adjust the sizes of the column objects to indicate how much data can be entered.

To change the width of an object:

1. In the "Form Designer" window, click on the DB Edit object to the right of "Employee #:".

Handlebars appear on the selected object and the status bar shows the column name *empid.*

2. Click and drag the "middle right" handlebar to the left.

The width and height coordinates will appear as a hint once you select the handlebar. The coordinates will change as you drag the mouse to change the size of the field.

3. Release the mouse button when the width value is 50.

The height value remains at 22.

Repeat this procedure for the DB Edit object labeled "First name," only increasing the width to 160.

Now, change the widths for the following DB Edit objects:

Column Name	Width
emplastname	160
empaddress	290
empcity	160
empstate	50
empzip	90
empareacode	50
empphone	120

4. Finally, click on the "Employee phone number:" Static Text object, and click and drag the "middle left" handlebar to the left to increase the object width, so all of the text is visible.

With the widths of all the fields have been adjusted, we'll now make sure all the DB Edit objects are the same height.

To make all the column objects the same height:

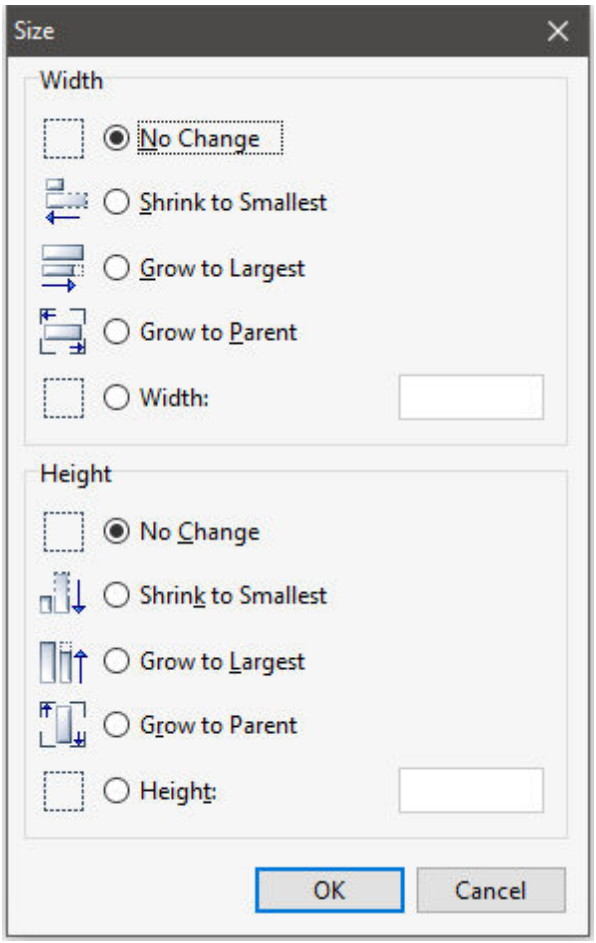
1. In the "Form Designer" window, click on the *empid* DB Edit object.
2. Press and hold the [Shift] key, then click on all the other DB Edit objects on the form. Do not click on any of the text objects. You can also select all the objects by lassoing the objects with your mouse cursor.

The DB Edit objects are now enclosed by red handlebars, as opposed to blue, indicating that they are selected as a group.

3. Right click on any of the objects and choose "Show Size Palette". A "Show Size Palette" toolbar button is also located on the Settings toolbar.

A dialog will appear for adjusting the width and height of the selected objects.

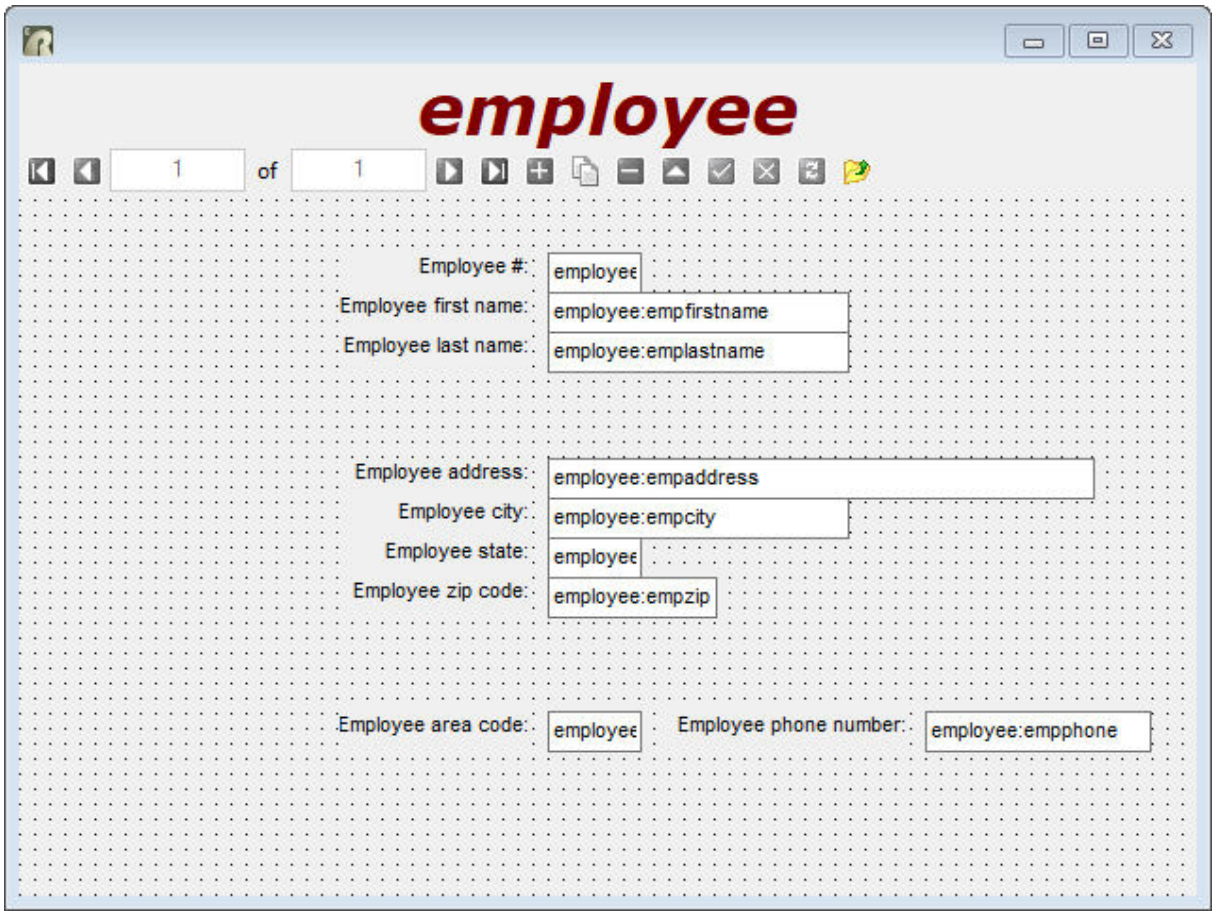
4. Under Height, select "Shrink to Smallest", and select OK.



The height of all the column objects is adjusted to match the smallest height. All the column objects are now the same height.

5. Click outside the red handlebar selected objects to deselect the fields.

The form should now look somewhat like the following:



Next, we'll line up each Static Text object and its corresponding DB Edit object.

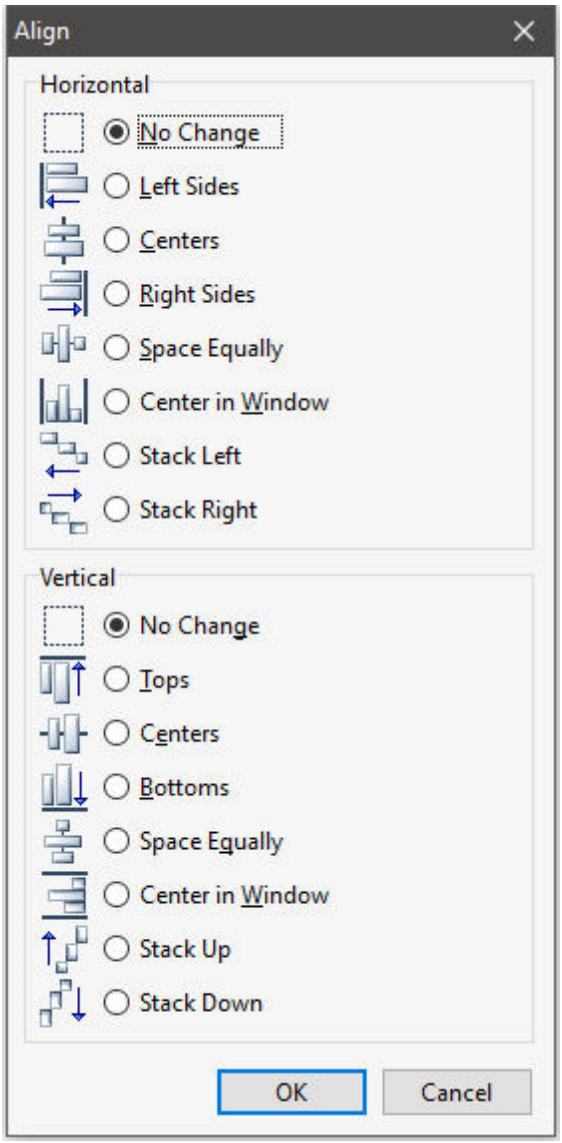
5.10 Aligning Objects

When using the Form Wizard, the top edge of each text object and its corresponding column object were lined up; they had the same "Left" value respectively. When moving objects with the mouse it can be difficult to position them at an exact pixel location, especially if the guidelines are turned off.

To align objects horizontally:

1. In the "Form Designer" window, click on the "Employee #" Static Text object.
2. Press and hold the [Shift] key, then click on the *empid* DB Edit object.
3. Right click on any of the objects and choose "Align".

A dialog will appear for adjusting the horizontal and vertical alignment of the selected objects. A toolbar button for the "Align" dialog box is also located on the Settings toolbar.



4. From the dialog box, in the "Vertical" list, choose "Bottoms" and select OK.

The objects are aligned along their bottom edges. The "Align" dialog box and "Bottoms" align option are used to line up a horizontal group of objects. The "Horizontal" and "Vertical" alignment of the objects refers to the direction the objects are moving, not the end result.

5. Click outside the red handlebars to deselect the objects.

Repeat this procedure for each line of objects on the form.

The objects might also need to be aligned vertically.

To align objects vertically:

1. In the "Form Designer" window, click the *empid* DB Edit object.
2. Press and hold the [Shift] key, then click on the DB Edit objects *empfname, emplname, empaddress, empcity, empstate,* and *empzip.* Do not select the *areacode* or *empphone* DB Edit objects.
3. Right click on any of the objects and choose "Align".

The same dialog box will appear for adjusting the horizontal and vertical alignment of the selected objects.

4. From the dialog box, in the "Horizontal" list, choose "Left Sides".

The selected objects are lined up along the left edges.

5. Click outside the objects to deselect them.

Repeat this procedure to align the text objects.

5.11 Entering and Editing Text Objects

To display text on a form, you place a Static Text object. The Static Text object contains the text you want to display. You can format the Static Text object to set the color, font, style, size, alignment, border, and more of the displayed text. The Static Text button is located on the "Standard Controls" toolbar as a capital "A".

To enter the company name for the form:

1. In the "Form Designer" window, choose the "Static Text" button.

The button will remain selected.

2. Position the mouse cursor at the bottom of the workspace below the other objects, and click the left mouse button.

The "Static Text Properties" dialog box is displayed with the default text, "Static Text" in the "Caption" field.

3. Overwrite the default text with the company name "skyWrite Ink." in the field.

We want edit the company name to be displayed similar to the title.

4. Click the "Font" button, which has the text "AaBbYyZz" on it.

The "Font" dialog box is displayed.

5. Select "18" in "Size" and "Maroon" under "Color".
6. Click the OK button.

The actual font has been changed on the Font button, which indicates the current font, style, color, etc. which is in use.

7. In the "Caption Settings" panel, choose "Centered" from the "Caption Alignment" drop down box.
8. Click the OK button.

9. Now, you can use the handlebars to increase the width/height of the Static Text object. Also drag the text in the middle of the form. When centering the Static Text in the form, guidelines will appear on the right and left sides of the object.

To edit the title for the form:

1. In the "Form Designer" window, select the title "employee"
2. Right-click on the object and select "Object Property".

The Static Text dialog box is displayed.

3. Edit the title of the form to "Employee Information" in the "Caption" field.
4. In the "Attributes" panel, remove the check in the "Auto Size" check box. This setting will adjust the width and height of the object based on the text in the caption field.
5. Click the OK button.

There are additional options for the Static Text object, which allow you to enhance the text even further.

Locate the object with arrows and other shapes on the form. This is an Enhanced DB Navigator object. It allows the end user to easily move in the form when looking at multiple rows of data. We will now change the font color for the Enhanced DB Navigator.

To change the font color for the Enhanced DB Navigator control:

1. Select the Enhanced DB Navigator object.
2. Right-click on the object and select "Object Property".

The "Enhanced DB Navigator Properties" dialog box is displayed.

3. Click the "Font" button, which has the text "AaBbYyZz" on it.

The "Font" dialog box is displayed.

4. Select "Maroon" under "Color".
5. Click the OK button.

Notice the font of the text changed color.

6. Click the OK button to save and close the properties dialog.

5.12 Creating a Bevel

By placing a Bevel, you can add dimension to your form. A Bevel is useful for placing borders around your form, grouping data, and dividing your form into sections.

We'll place a Bevel around the objects on *empform.* The Bevel button is located on the Additional Controls toolbar.

To place a Bevel around objects on the form:

1. In the "Form Designer" window, choose the Bevel button.
2. Position the cursor up and to the left of the text objects and below the title and click the left mouse button.

The Bevel Properties dialog will appear.

3. From the "Style and Shape" panel, select the "Style:" drop down box and select "Raised".
4. Click the OK button.

You will see the Bevel box on the form where your cursor was positioned.

5. Click and drag the lower right comer handlebar of the Bevel so that the box surrounds all of the text and column objects. Leave the form title and DB Navigator objects above the Bevel and the company name static text below the Bevel.

6. Right click on the Bevel and select "Send To Back" from the pop-up menu. This places the Bevel on a lower layer than that of the DB Edit and Static Text fields. If the objects do not fit in the Bevel neatly, the fields may now be selected and moved accordingly.

The text and column objects are now within the Bevel border. The form title object is above the Bevel.

If the bevel does not fit in the form uniformly, you may increase the width of the form by clicking and dragging the right border.

5.13 Additional Objects

In addition to the bevels, you can place clocks, lines, boxes, buttons, images, and more. All of these features help you make your forms easy to use. Please refer to the Forms Additional Controls section of the Help for more information of the additional controls and the properties for each.

5.14 Form Properties

You can customize the form size, caption, caption buttons, color, closing behavior, and much more. The Form Properties can be accessed directly from any clear background space on a form by right clicking on the background and selecting "Object Property". In some forms, you may have less room after placing a great number of objects. Another method to open the Form Properties is by using the "Edit" toolbar and selecting the "Form Properties" button. The button is on the far right of the toolbar.

To change the form caption

1. In the "Form Designer", select the "Form Properties" button.

The "Form Properties" dialog will appear.

2. Under the panel "Caption and Hint", change the text in "Form Caption" to "Employee Information".
3. Select the OK button.

The form should now look somewhat like the following:

5.15 Saving, Testing and Closing Forms

You can save changes at any time as you are working on the form. When you close the "Form Designer" window, R:BASE will prompt you to save changes if necessary.

To save the changes you've made to the form:

From the Menu Bar, choose **Form: Save Form**.

If you have already saved the form, this option will be grayed out.

To test the form:

You can test how your form will look by running it right from the designer.

From the Menu Bar, choose **Form: Run Form**.

Press the [Esc] key to return to the designer.

To close a form:

From the Menu Bar, choose **Form: Close**.

Closing the form will return you to the Database Explorer window.

5.16 Running the Form

Let's test the form even further by using it to enter some information. We'll start it from the Database Explorer.

To run the form and add a row of data:

1. In the Database Explorer, click the Forms option.
2. Select "empform".
3. Click the "Run Form" option.

The "Run Form empform" dialog box is displayed so that you can choose to start the form and either "Enter", "Edit", or "Browse" table data.

4. Select the "Enter" option.
5. Click the OK button.

R:BASE displays the form ready for you to enter information. The field for *Employee #* is already filled in. Because the column is an auto numbered column, R:BASE enters the next available number for the employee number automatically. If you hover the mouse cursor over the Enhanced DB Navigator control, a hint will appear for each button. The "-1" value is displayed in the record box as the row has not yet been added to the table.

Let's enter a row of data into the database. Use your own name, address, and phone.

You can move from field to field in the form. Press [Shift]+[Tab] to move to the previous field, or press [Tab] to move to the next field.

6. After entering the phone number, choose the select the button with a check mark on it. The hint is "Save"

R:BASE stores the row of information in the database.

7. Select the "Close Window" button on the DB Navigator to close the form.

5.17 Creating a Custom Form

You have now used a few of the features R:BASE offers when you are editing a form. In the remainder of the lesson, we will create another form. This time we'll build the form without using the "Form Wizard" option. You will learn how to further customize a form and how to include variables and expressions.

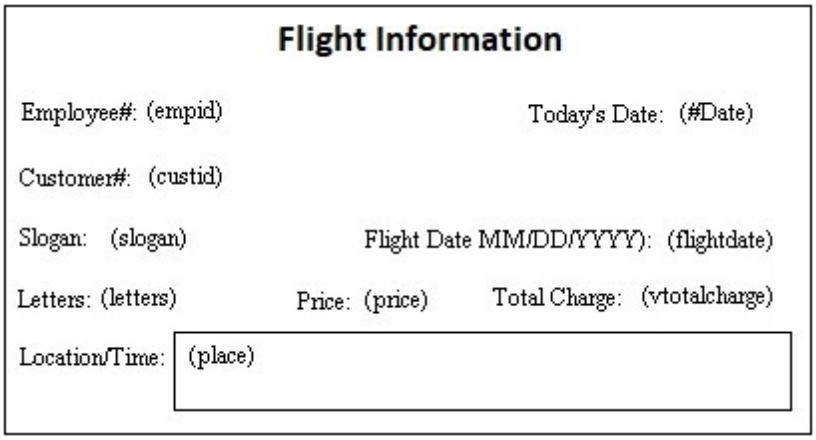
We will follow the three design steps for our form:

1. Define the form's purpose.

We will use the form to enter and edit information about the flights made for customer advertising. The information includes:

Information	Column or Variable Name
Current date	*#date*
Employee number	*empid*
Customer number	*custid*
Flight date	*flightdate*
Slogan	*slogan*
Number of letters	*letters*
Price per letter	*price*
Total charge for slogan	*vtotalcharge*
Location and time notes	*place*

2. Draw the form.



The sketch of the form shows its title, the labels for fields, and the location of fields. All of the fields are column objects except the fields for *vtotalcharge* and #date. These fields are variable objects. The form will have a Bevel.

3. List specific customizations such as tables to be referenced, or colors to be displayed on the form.

The form will have three customized fields. One field, *flightdate,* will have custom colors. The column *custid* will be placed as a combo box and the column *empid* will have a pop-up menu defined. Now that you have defined the steps, you can begin creating your form.

When you build a custom form you're provided with an empty workspace, as R:BASE doesn't place any objects for you.

To create the custom form, flightform:

1. In the Database Explorer, click the Forms option.
2. Click the "New Form" option.

The "New Form" dialog box is displayed.

3. Enter "flightform" within "Form Name".
4. Select "flights" from "Table/View Name".
5. Enter "Enter & edit flight information" in "Form Comment".
6. Leave the "Form Wizard" option unchecked.
7. Click the OK button.

R:BASE opens the "Form Designer" window. Maximize the window so that you have more workspace.

5.18 Using Expressions and Variables

We first used expressions in Lesson Two when we created the computed column *totalcharge.* Remember, an expression is a calculation used to determine a value. In Lesson Two, the column *totalcharge* was assigned the expression: *(letters * price).*

An expression can also be assigned to a variable. A variable is a name you create that will contain the answer to an expression. For example, in the expression *vquantity* = (12 * 10)*,* the variable *vquantity* contains the value 120.

You can assign a variable to an expression and then place the variable on the form. When the form is used, it will display the value of the variable.

In *flightform* we include a field labeled "Total Charge" that shows the total amount of money the customer is charged for the flight. To display this amount, we assign a variable to an expression that makes the same calculation that R:BASE performs for the computed column *totalcharge.*

The reason we are using an expression in the form to do the same calculation that R:BASE performs in the table is a matter of timing: R:BASE only performs the calculation for the computed column *totalcharge* when the data is stored in the table. Since we want to see the result in the form before R:BASE stores the data, we will perform the same calculation in the form.

To define an expression in the form:

1. In the "Form Designer" window, choose **Variables: Add/Edit Variables...**

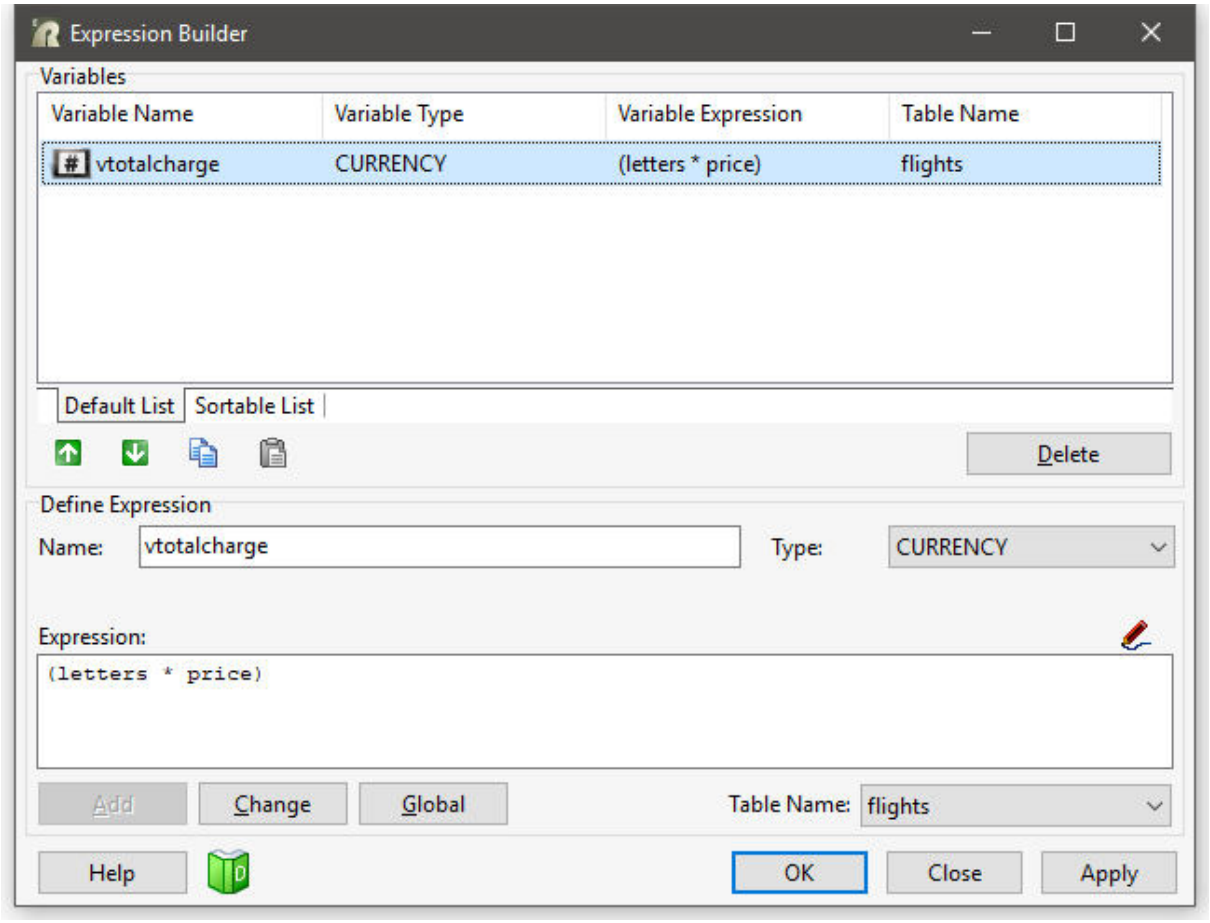
The "Expression Builder" dialog box is displayed.

2. Enter "vtotalcharge" in "Name:"
3. Choose "CURRENCY" from the "Type:" drop down box.
4. Enter "(letters * price)" in the "Expression:" panel.

Spaces on either side of the operator are optional.

5. Click the Add button.

The variable is added and the expression is displayed in "Variables".



Note: If you get an error message when you click the Add button, correct the expression by editing it. After editing the expression, click the Add button.

6. Click the "Apply" button, then"Close".

To look at the expression you just defined:

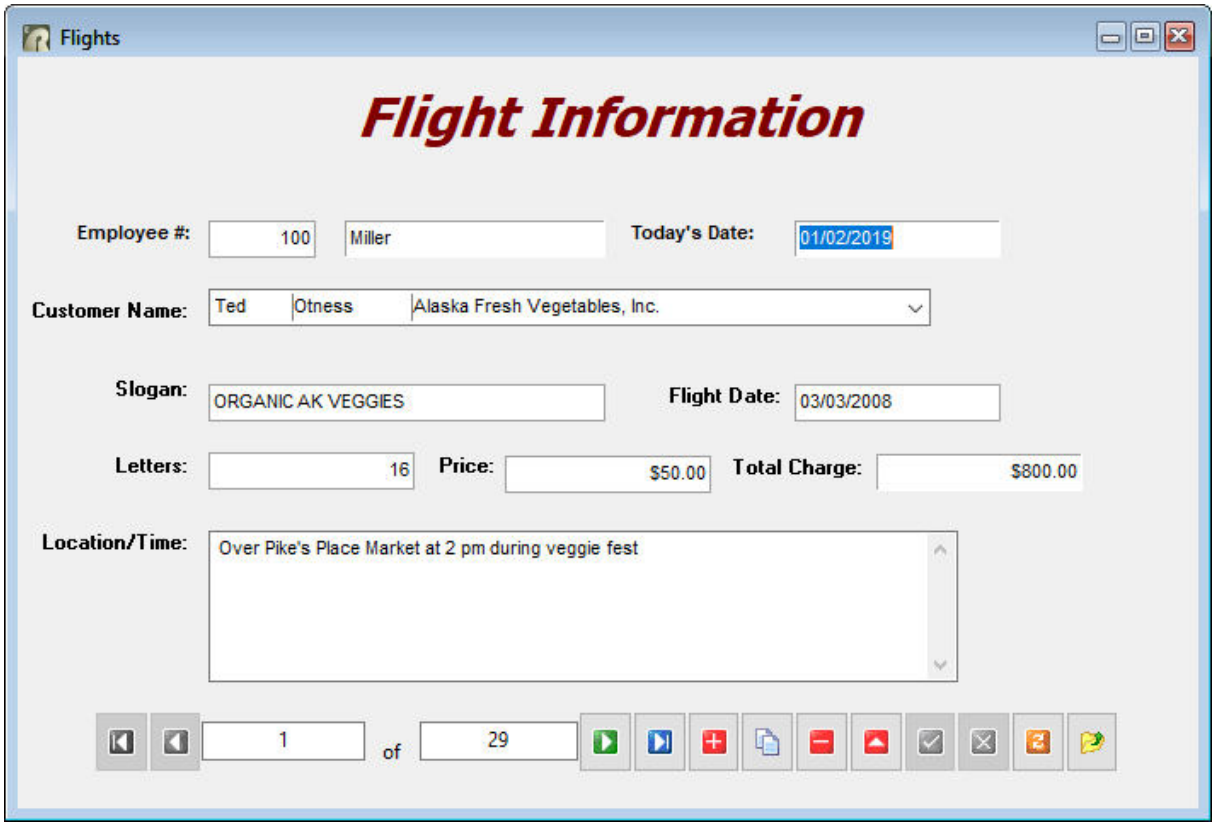
1. In the "Form Designer" window, choose **Variables: List of Expressions**.

R:BASE displays the expressions associated with the form. The expressions are listed table by table. R:BASE has assigned a number, 1, and data type, CURRENCY, to the variable you defined.

2. Select the "X" to return to the "Form Designer" window.

5.19 Placing Objects

The screen below shows the layout of *flightform.* The form contains a title and a label for each field as well as the fields themselves.



We will practice placing the different object types a few times, and then you can place the remaining objects on the form. Refer to the sketch of the form when placing objects on the form.

5.20 Placing Text Objects

The first object we'll place displays the text label for the employee identification column in *flights* table.

To place text objects on flightform:

1. In the "Form Designer" window, choose the "Static Text" button from the "Standard Controls" toolbar. From earlier, you may remember that the button has a capital "A" on it and nothing else.
2. Position the cursor in the upper left quadrant of the workspace and click the left mouse button.

The "Static Text Properties" dialog box is displayed.

3. Enter "Employee #:" in "Caption:".
4. Click the OK button, and increase the object width as needed.
5. Choose the "Static Text" button, again.
6. Position the cursor in the upper right quadrant of the form workspace, opposite the "Employee #:" text object, and click the left mouse button.

The "Static Text Properties" dialog box is displayed.

7. Enter "Today's Date:" in "Caption:".
8. Click the OK button, and increase the object width as needed.

5.21 Placing DB Edit (column) Objects

Now we'll place the DB Edit object for the employee number.

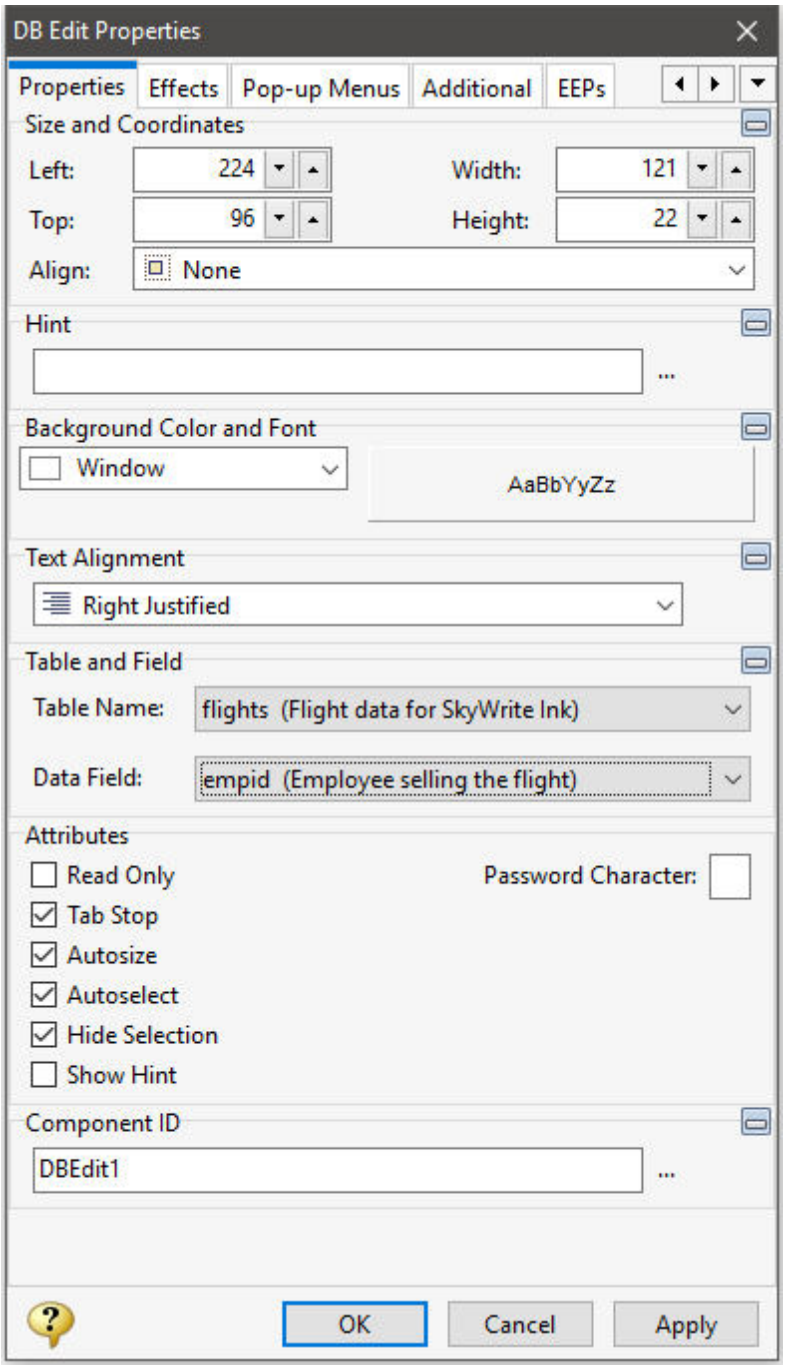
To place DB Edit objects on flightform:

1. In the "Form Designer" window, choose the "DB Edit" button from the "Database Controls" toolbar.

The "Database Controls" toolbar specifically places objects that reference data that is stored in the table. You will notice that each button has a picture of a table in the background. The "DB Edit" button is the second button from the top left.

2. Position the cursor immediately to the right of the "Employee #:" text object and click the left mouse button.

The "DB Edit Properties" dialog box is displayed. You will notice a panel with "Table and Field". You will use the "Data Field:" drop down box to select from a list of available columns.



3. Select *empid*.
4. Click the OK button.

5.22 Placing Variable Objects

In the design sketch for *flightform* we included a field labeled today's date. The form will display the current date automatically in this field by using a system variable.

A system variable is a variable that R:BASE creates and are always available for use. The R:BASE system variables include:

- *#DATE*
- *#NOW*
- *#PI*
- *#TIME*
- *SQLCODE*
- *SQLSTATE*

You can place *#DATE,* which holds the date stored in your computer, on the form to display the current date. The procedure for placing variable objects is very much like placing a column object. Here we place the system variable *#DATE.*

To place a variable object on flightform:

1. In the "Form Designer" window, choose the "Variable Edit" button from the "Variable Controls" toolbar.

The "Variable Controls" toolbar specifically places objects that reference variables that are defined. You will notice that each button is similar to the "Database Controls" toolbar. The "Variable Edit" button is the second button from the top left.

2. Position the cursor immediately to the right of the "Today's Date" text object and click the left mouse button.

The "Variable Edit Properties" dialog box is displayed. The "Variable" panel displays the R:BASE system variables as well as all defined form variables.

3. Select the system variable "#DATE".
4. Click the OK button.

5.23 Placing DB Memo (scrolling) Objects

When you are placing column or variable objects with a NOTE data type, you can make the object location smaller than the actual length of the text you will be entering and yet enter as much text in the field as the column length allows. The objects can be placed as a single line object or a multiple-line scrolling object.

We will place the *place* column as a multiple-line scrolling object called a DB Memo object. The column has a NOTE data type and can hold as many as 4,050 characters. The salesperson can enter as much information as necessary in the field.

To place a DB Memo object on flightform:

1. In the "Form Designer" window, choose "DB Memo" button from the "Database Controls" toolbar.

The "DB Memo" button has a piece of paper with multiple lines on it with a table in the background and is the third button from the top left.

2. Position the cursor toward the bottom of the Form Designer workspace and click the left mouse button. Refer to the sketch of the form as needed for object placement.

The "DB Memo Properties" dialog box is displayed.

3. From the "Table and Field" panel, choose *place* from the "Data Field" drop down box.
4. Make sure that the "Word Wrap" option in the "Attributes" panel is checked.

Across the top of the dialog, there are more tabs for additional properties.

5. Choose the "Effects" tab.
6. From the "Scroll Bars" panel across the bottom, select the "Vertical" radio button to add a vertical scroll bar.
7. Click the OK button.

The column object is placed with the default width.

8. Click on the handlebar on the lower right corner of the *place* object and drag it to the right to increase the size of the box for several lines of text to wrap.

With the "Word Wrap" attribute, the text will wrap at the end of each line. When the text reaches the end of the last line of the box, the top line will scroll out of sight leaving an empty line at the bottom of the field.

When running the form, you can display all the data by double clicking in the DB Memo field to launch the R:BASE BLOB Editor. The R:BASE BLOB Editor can be used to view and edit the text of NOTE and VARCHAR data type columns. It can also be used to view and edit images which are stored in the database.

If you have not done so already save the form. From the Menu Bar, choose **Form: Save Form**.

5.24 Placing a DB Lookup Combo Box

A DB Lookup Combo Box is a drop down list from which the user can select a value. Both a combo box and a pop-up menu can help ensure that information entered in the database is accurate. Because users choose an item from a menu or a list, typing mistakes are eliminated and users can't choose information that doesn't belong in the field.

There are two types of DB Lookup Combo Boxes:

Single column - lists values taken from a single column in a table
Multi column - lists values taken from multiple columns in a table

In *flightform,* we'll use a combo box to select the customer information. The menu will display only customers that exist in the database. Users must then select one of these customers, which help to enforce the relationship constraints. Later, we'll define a pop-up menu for employee information.

To place a DB Lookup Combo Box on flightform:

1. In the "Form Designer" window, choose "DB Lookup Combo Box" from the "Database Controls" toolbar. It is the tenth button from the right.
2. Position the cursor below the *empid* column object and click the left mouse button.

The "DB Lookup Combo Box Properties" dialog box is displayed. In the "Table and Field" panel, *flights* should be listed as the table.

3. From the "Data Field" drop down box, select *custid*.

Notice the multiple tabs across the "DB Lookup Combo Box Properties" dialog box.

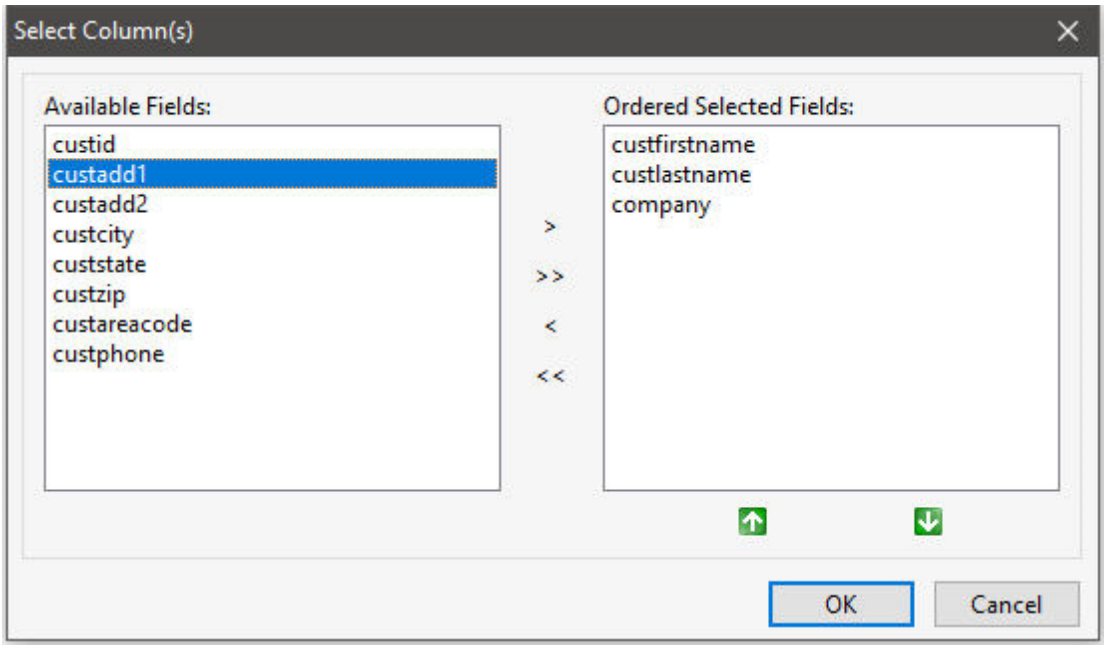
4. Select the "Effects" Tab

There you will see the "Row Count:" field in the "Drop-Down". This setting indicates how far the list will drop down when the user selects it. For instance, if you only want to display a few values at one time, decrease the number count. Or, if you want to display more values, increase the number count. To display a longer list of companies we'll increase the combo box drop-down count.

5. Increase the "Row Count:" from 16 to 20.

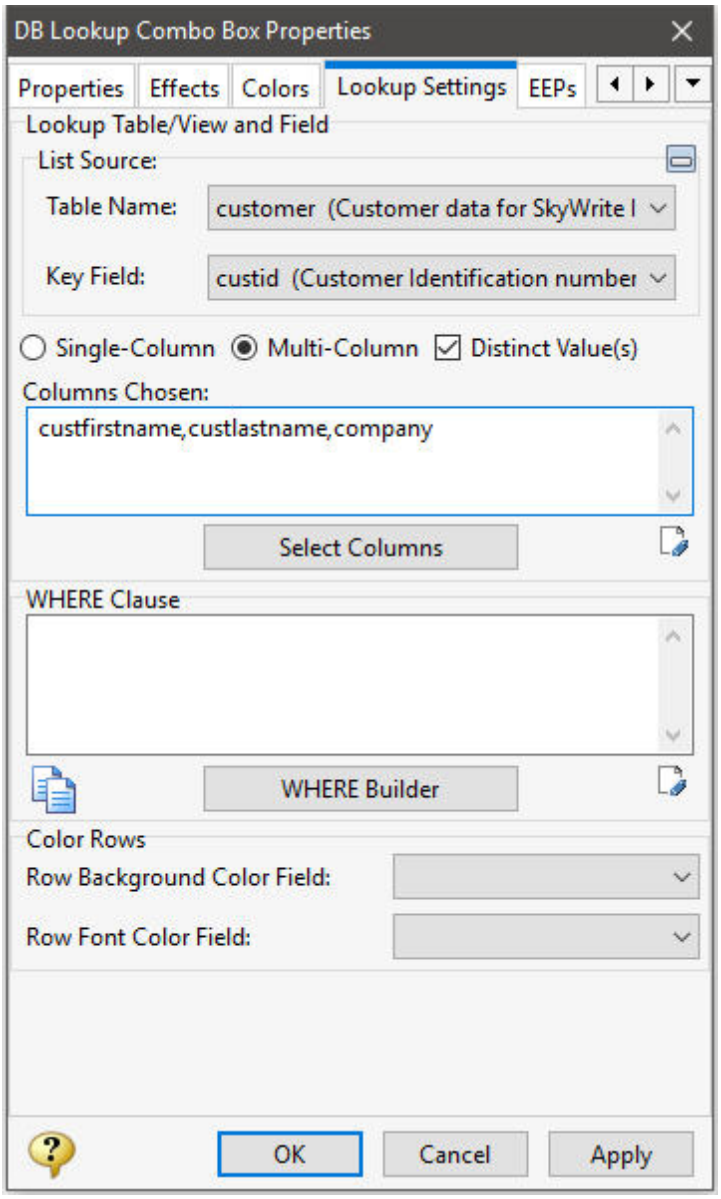
Now you must define the settings for the table and column, which you want to "look up".

6. Select the "Lookup Settings" tab from the "DB Lookup Combo Box Properties" dialog box.
7. From the "Lookup Table/View and Field" panel, select "customer" from "List Source" > "Table Name:". The combo box will display all of the available tables and views.
8. In the same panel, select "custid" from the "Key Field" drop down box. This displays data from the *customer* table and will return the value from the selected column.
9. Select the "Multi-Column" radio button.
10. Click the "Select Columns" button to pull up the "Select Column(s)" dialog.
11. Choose the *custfirstname*, *custlastname*, and *company* columns and add each to the right panel using the arrow button.



12. Select the OK button.

We added the customer's name and company because not all customers have a company name.



The data values from the *custfirstname*, *custlastname*, and *company* columns in the *customer* table will be displayed in the combo box when the form is run.

The *custid* value corresponding to the selected *custfirstname*, *custlastname*, and *company* value will be the value that the form uses. The relationship between the tables' *customer* and *flights* is directly related to the primary and foreign keys that were defined on the *custid* columns in each table. Remember, *custid* is a primary key in the *customer* table and a foreign key in the *flights* table.

We will make one more enhancement to the combo box so the data does not run together.

13. Select the "Effects" tab again from the "DB Lookup Combo Box Properties" dialog.
14. From the "Show Lines" panel, place a check in the "Vertical" check box.
15. Click the OK button to return to the "Form Designer".

There will more data displayed in the combo box, so increase the width of object.

16. Use the handlebars on the DB Lookup Combo Box to drag the control and increase its width.

5.25 DB User Defined Combo Box

Another option to display a combo box is a User Defined Combo Box. The User Defined Combo Box is very similar to a DB Lookup Combo Box with the exception that it will display a combo box of preset values or an expression that you define and inserts your choice into the table. This would be used if you want the user to select from a custom list within the drop down menu.

5.26 Defining a Pop-Up Menu

A pop-up menu is similar to a combo box in that users select the data from a list. There are four types of pop-up menus:

Single column - a pop-menu displaying a list of values taken from a single column in a table

Multiple column - a pop-menu displaying a list of values from multiple columns in a table. You can display many columns, but only one column's value is returned to the field. This column does not have to be displayed.

Expression - a pop-up menu displaying a list of values from an expression. You can use this type of pop-up, for example, when entering a person's full name; use an expression that combines data from a first name column and last name column to list full names. The value displayed on the menu is returned to the field.

Expression with a column - a pop-up menu displaying a list of values from an expression, but returning a value from a column to the field

Because Charlie and Amelia only have seven employees, it often easier to just type in the employee number. The pop-up menu is optional. We will also define a look-up variable to display the employee's last name on the form.

To define a pop-up menu for empid:

1. In the "Form Designer" window, click on the existing *empid* DB Edit object to select it.
2. Right click on it and select "Object Property".
3. Across the top of the "DB Edit Properties" dialog, choose the "Pop-up Menus" tab.
4. Check "User can Access a Pop-up Menu".

Another option in the dialog box "Use a Custom Pop-up" is now activated.

5. Check "Use a Custom Pop-up".

The other options in the dialog are now activated.

6. Enable the options for "Automatic Pop-up" and "Skip to Next After".

By enabling these options, the pop-up will display automatically and then the cursor will jump to the next field once the selection is made.

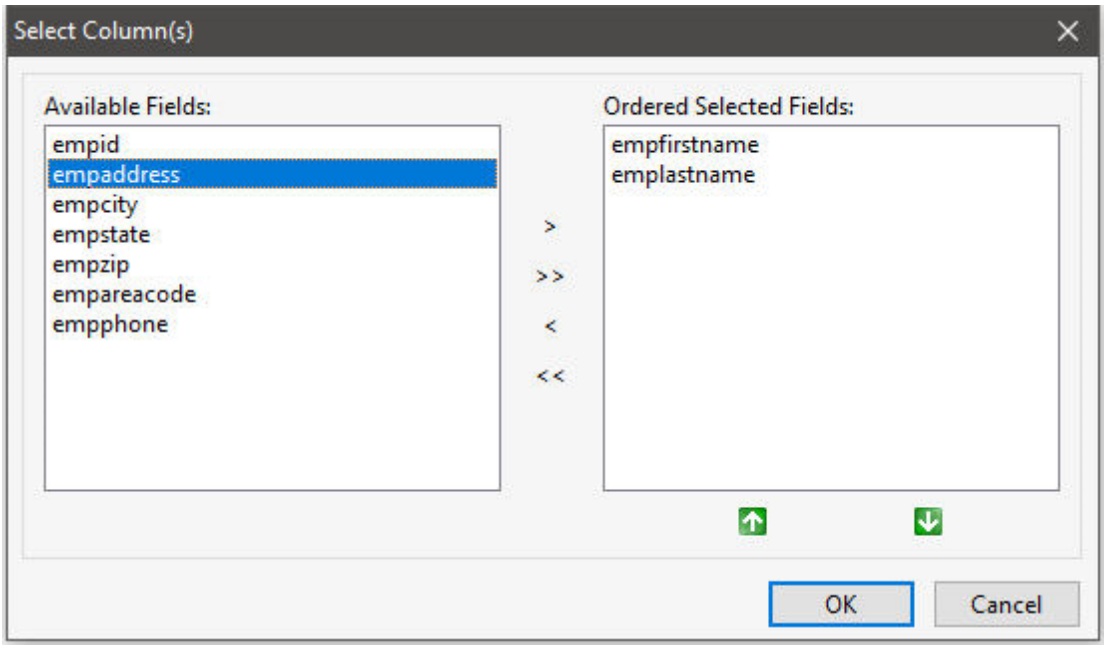
7. Select "employee" in "Table/View Name" drop down box.

You will notice that all of the tables and views are available to define the Pop-up menu.

8. Select "Multi Column" in "Pop-up Type".

The "Select Column(s)" menu is displayed listing the columns in the *employee* table.

9. Using the arrows to move your selection from one panel to the other select "empfirstname" move it the right side and select "emplastname" and move it to the right side.



10. Click the OK button.

The column names are displayed in "Menu Values".

11. Select "empid" within "Column Returned".

The value that is selected from the *empid* column in the *employee* table will be returned to the *empid* column in the *flights* table. The relationship between the tables *employee* and *flights* is directly related to the primary and foreign keys that were defined on the *empid* columns in each table. Remember, *empid* is a primary key in the *employee* table and a foreign key in the *flights* table.

12. Enter "Employee List" in "Menu Caption".
13. Increase the "Number of Lines" field to 7 to display more employees.

To define a hint for the empid object:

1. Select the "Properties" tab.
2. Enter "Press [Shift]+[F3] for Menu" in the "Hint" field, and verify the "Show Hint" attribute is checked.
3. Press the OK button.

Next, we'll define a look-up variable to display the employee's last name on the form. Then, we'll place the variable. A look-up expression is used to retrieve data from another table and display the data on a form.

To define a look-up expression:

1. In the "Form Designer" window on the Menu Bar, choose **Variables: Add/Edit Variables**.

R:BASE displays the "Expression Builder" dialog box in which you can enter the expression.

2. In "Define Expression," enter "vemplname" in the "Name:" field.
3. In "Type," select the "TEXT" data type.
4. In "Define Expression," enter "emplastname IN employee WHERE empid = empid" in the "Expression:" panel.
5. Click the Add button.

The variable is added and the expression is displayed in "Variables"

6. Click the OK button.

When the form is run, the look-up expression will find the employee's last name from the *employee* table that matches the *empid* value we enter. We can enter the *empid* value by selecting it from the pop-up menu or by entering it directly into the field.

To place the vemplname variable on the form:

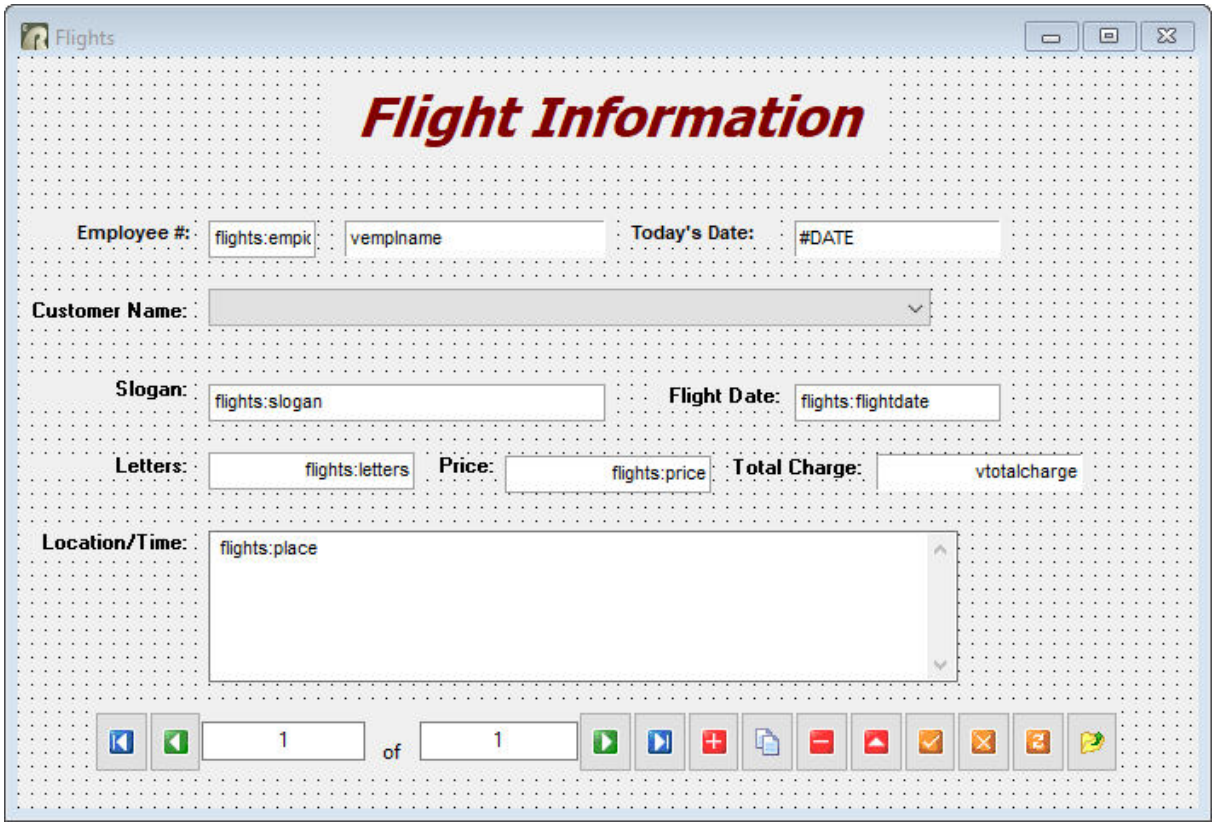
1. In the "Form Designer" window, choose the "Variable Edit" button from the "Variable Controls" toolbar.
2. Position the cursor immediately to the right of the *empid* column object and click the left mouse button.

R:BASE displays the "Variable Edit Properties" dialog box. The R:BASE system variables and form variables are listed in the "Variable" drop down box.

3. Select the variable "vemplname".
4. Click the OK button.

5.27 Completing the Form

Now that you've learned how to place the different type of objects, complete the form by placing the remaining text, column, and variable objects. The table below shows the objects that need to be added. The screen shows the positions of the objects. Add the Enhanced DB Navigator and edit the form caption as shown.



Remaining Features for flightform

Text objects	Column or Variable Objects
Flight Information	slogan
Location/Time:	flightdate
Customer Name:	letters
Slogan:	price
Flight Date:	vtotalcharge
Letters:	
Price:	
Total Charge:	

Once the remaining objects are placed, you can move them, size them, and align them until the layout of the form matches the sketch. At any time in the form design process, you can save and preview the form layout.

5.28 Form Hints

Just as R:BASE displays a hint over the toolbar buttons in the Form Designer, you can add hints the data entry objects on forms.

To define a hint message for the *place* DB Memo object:

1. In the "Form Designer" window, click on the *place* DB Memo object.
2. Right click and select Object Property.

The "DB Memo Properties" dialog box is displayed.

3. In the "Hint:" field, enter "Enter all relative flight information." You may need to expand the "Hint" panel if no field is displayed directly beneath the word Hint, and verify the "Show Hint" attribute is checked.

When you hover your mouse cursor over the field, the hint is displayed. The hint will appear when you run the form and in the Form Designer.

4. Click the OK button.

5.29 Changing Object Properties

There are three objects placed on *flightform* where the value is calculated: *vtotalcharge, vemplastname,* and *#date.* The user does not enter or modify these values. We can specify settings for these objects to prevent users from changing data in these fields. We will also exclude the object from being enabled in the cursor tab order. The Tab Order is the order in which you place objects on a form is the order R:BASE prompts you to enter data. Next in this chapter you will review how to edit the Tab Order.

To change an object's properties:

1. In the "Form Designer" window, click on the *vtotalcharge* variable object, right click, and choose Object Property.

The "Variable Edit Properties" dialog box is displayed.

2. Check the "Read Only" option, and uncheck the "Tab Stop" and "Autoselect" fields options.

The "Autoselect" option will toggle whether all of the text in the field is selected or if just a blinking cursor will appear. The "Tab Stop" check box toggles whether or not the field is part of Tab Order, which is discussed next.

3. Click the OK button.

Repeat this procedure for the *vemplastname* and *#date* variable objects.

5.30 Changing the Tab Order

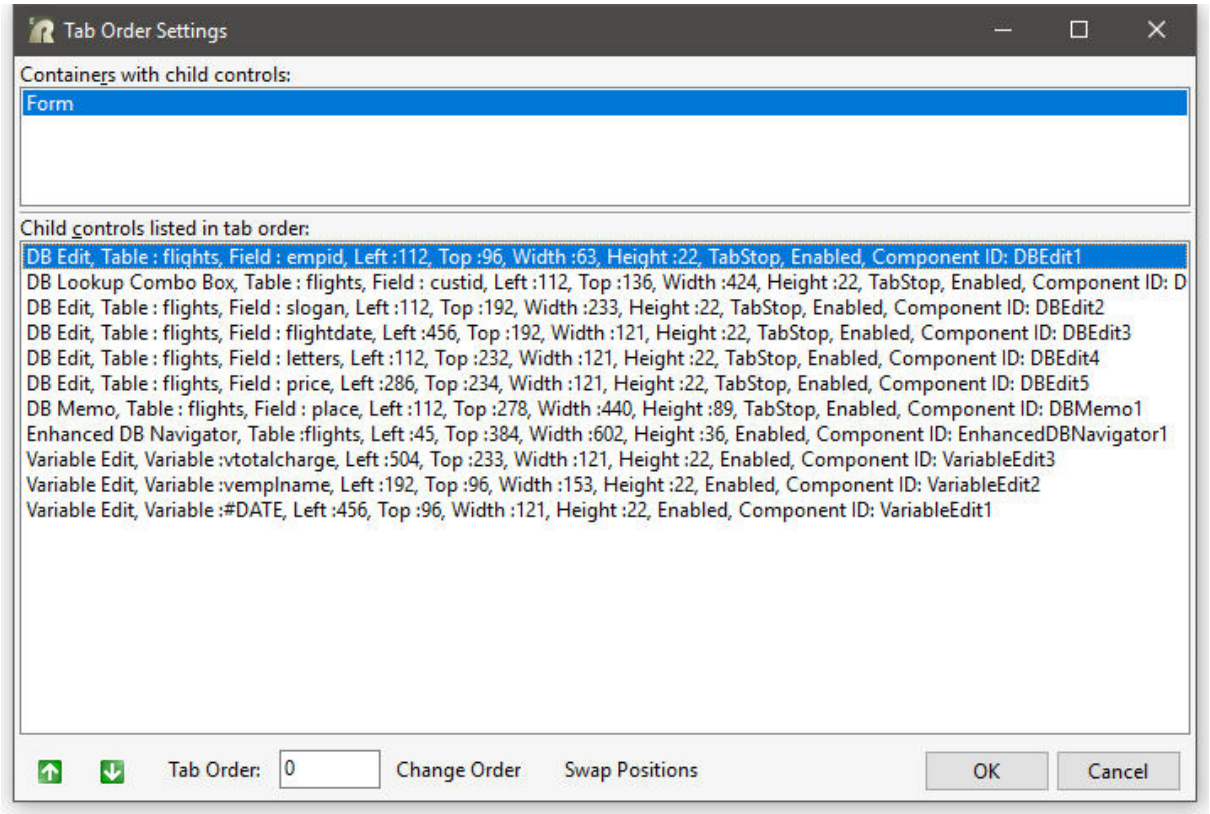
The order in which you place objects on a form is the order R:BASE prompts you to enter data. When you have finished building a form, you will probably want to check the object order and then make some adjustments. If you want to change the order, you must change the order of the column and variable objects.

When building *flightform* we did not place the fields in the best order for data entry. We placed the note field *place* third, and so it appears third in the field order. The form would be easier to fill out if it appeared last in the order.

To change the field order for flightform:

1. In the "Form Designer" window on the Menu Bar, choose **Layout: Change Tab Order**.

The "Tab Order Settings" dialog will appear with the objects on the form listed by "Control Type" and the field properties. If no items appeared in the "Tab Order Settings" dialog, then close the window, deselect the object which is selected on the form, and repeat Step 1.



2. Select "DB Lookup Combo Box, Table : flights, Field : custid"
3. Click and hold the green up arrow button until "DB Lookup ComboBox, Table : flights, Field : custid" is second in the list, right below "DBEdit, Table : flights, Field : empid".

You want users to select the employee booking the flight as the first data item when using this form, and the company booking the flight as the second data item.

4. Repeat Steps 1 through 3 to place the fields in the following order:

DBEdit, Table : flights, Field : empid
DB Lookup ComboBox, Table : flights, Field : custid

DBEdit, Table : flights, Field : slogan
DBEdit, Table : flights, Field : flightdate
DBEdit, Table : flights, Field : letters
DBEdit, Table : flights, Field : price
DBMemo, Table : flights, Field : place

The Variable Edit objects and DB Navigator will appear at the bottom of the list. These items are not enabled with the "Tab Stop" option in their Object Property settings and can be left here.

5. Click the OK button.

5.31 Save the Custom Form

You can save your changes at any time as you are working on the form. When you close the "Form Designer" window, R:BASE will prompt you to save changes if necessary.

To save the changes you've made to the form:

From the Menu Bar, choose **Form: Save Form**.

If you have already saved the form, this option will be grayed out.

Close the Form Designer.

5.32 Running the Custom Form

Once you have made changes to a form, run and test it to make sure it is working as you wanted.

To run the form:

1. In the Database Explorer, click the Forms tab.
2. Select "flightform" and click the "Run" option.

The "Run Form" dialog box is displayed with "flightform" selected.

The dialog displays options to start the form and either "Enter", "Edit", or "Browse" table data.

3. Select "Enter"
4. Click the OK button.

The form is displayed in the "Flight Information" window and the Pop-up for "Employee List" automatically appears. The current date is displayed in the *Today's Date* field.

You can enter data and move in the form. When you test a form, keep in mind that any data you save will appear in the database. If you are testing with false data, be sure to remove it from the database.

5. In the pop-up window, select "Jessica Jennings", and press the [Enter] key.

R:BASE displays the name "Jennings" and the corresponding *empid* value, *102.* Only the *empid* value is loaded into the *flights* table, the last name is displayed only and the cursor jumps to the next field automatically.

6. Enter the following information into the remaining fields.

Field Label	**Data**
Customer Name:	Country Skillet Restaurants
Slogan:	COUNTRY SKILLET
Flight Date:	08/25/2009
Letters:	14
Price:	45

R:BASE automatically calculates the total charge based on the expression you defined for *vtotalcharge.* R:BASE then moves to the last field in the form, *Location/Time.*

7. In the "Location/Time" field, enter "Reduced rate because the slogan will be written twice. Once on the Eastside over Hwy 520 and 405 intersection, at 10 am and once directly over the Space Needle in Seattle, 12 noon."

As you enter the text, you can see the text wrap, allowing you to enter more information.

8. When you have finished entering the text, choose the "Insert" button on the DB Navigator (plus character "+").

R:BASE stores the data in the database and clears the form so you can enter more information. You can practice entering more data.

9. When you are done testing the form, select the "Close Window" button on the DB Navigator.

5.33 Create Customer Form

To use the rest of the Tutorial, create a third form based upon the customer table.

5.34 Summary

In this lesson you have:

- Learned how to start and use the forms menus
- Created and modified a form with the Form Wizard
- Created and modified a Custom form
- Placed objects on a form
- Created variables and expressions
- Placed a Bevel
- Customized form and field properties
- Changed the field order
- Entered data and tested a form

To check your knowledge, try answering the following questions:

1. What is a form used for?
2. What are the three steps to follow in designing your form?
3. What is a system variable?
4. What are some object properties that can be changed for form field objects?
5. What are handlebars?

5.35 Tips

Here are a few tips to help you get the most out of the R:BASE Form Designer:

- You can cut, copy and paste one or more form controls at a time, either in the same form or between different forms.
    - It is important to note that when using copy and paste for form objects, the control being copied will retain the Top, Left, Height and Width coordinates. This logic is in place by design for the purpose of moving a control from one form to another, where the control will appear in the same location with the same size. This means when you copy an object that is is defined with 800 pixels as the Left value, and paste it into a small Panel with a 400 pixel width, the newly pasted object is another 400 pixels beyond the panel's border, and not visible. Pressing the [Ctrl+Left Arrow] keys will place the object where it is visible.

- You can paste external content (text, rich text, images) directly into the Form Designer from the Windows clipboard.
- To hide Form Ruler, right click on the ruler and select 'Hide' from the speed menu option.
- While working with the Form Designer in its maximized window state, you can display the "Object List" toolbar by selecting **View: Toolbars: Object List** from the Menu Bar.
- You should pre-define all global variables used in a form. You may access R> Prompt window to define global variables or use [Global] button while defining your Form Expressions.
- When adding several of the same type of object, select the "Sticky Control" on the "Settings" toolbar. This will allow you to add the same type of control until the "Sticky Control" is unselected, preventing several mouse click steps.
- When initially designing the form, you should create and place the objects within a form so that is suitable for users with various screen resolutions, preferable starting at 1024x768, as you never know the end users' preferences.
- After a section of the form is finished, you can lock the objects to prevent accidentally adjusting them with the "Lock" feature. With any number of controls selected, choose **Layout: Lock Control**. Then, when the controls are selected, the handlebars will be grayed out.
- To prevent unauthorized editing of a form, you can assign a "Design-Time Password". To do so, choose **Layout: Passwords: Set Design-Time Password**
- Use the "Format" toolbar options (Font Name, Font Size, Font Style, and Justification) to set the font, colors and justification of controls. Font Name selects the font name for text controls. Use True Type fonts (indicated by "T" icon) when possible. These render well on the screen.
- The "Format" toolbar also contains the "Bring to Front and "Send to Back" buttons which are used when layering the form objects on top of one another. The controls in the front display last, and the controls in the back display first.
- Use the "Align or Space" toolbar options to position controls relative to one another and relative to the parent control in which they appear.
    - Align Left aligns a group of controls with the "leftmost" control's position.
    - Align Middle centers a group of controls based on the "horizontal center" of the control group.
    - Align Right aligns a group of controls with the "rightmost" control's position.
    - Align Top aligns a group of controls with the "topmost" control's position.
    - Align Center aligns a group of controls based on the "vertical center" of the control group.
    - Align Bottom aligns a group of controls based with the "bottommost" control's position.
    - Space Horizontally spaces a set of controls based on the "leftmost" control's position and the "rightmost" control's position.
    - Space Vertically spaces a set of controls based on the "topmost" control's position and the "bottommost" control's position.
    - Center Horizontally in Window centers a control "horizontally" within a parent control.
    - Center Vertically in Window centers a control "vertically" within a parent control.
    - Stack Left arranges a set of controls in a pile to the "leftmost" control, or to the control that was selected first.
    - Stack Right arranges a set of controls in a pile to the "rightmost" control, or to the control that was selected first.
    - Stack Up arranges a set of controls in a pile to the "topmost" control, or to the control that was selected first.
    - Stack Down arranges a set of controls in a pile to the "bottommost" control, or to the control that was selected first.
- Use the "Size" toolbar options to adjust width and height of controls.
    - Shrink Width to Smallest determines the "minimum width" of all the selected controls, and then sets the "width" of the controls to that value.
    - Grow Width to Largest determines the "maximum width" of all selected controls, and then sets the "width" of the controls to that value.

  - Shrink Height to Smallest determines the "minimum height" of all the selected controls, and then sets the "height" of the controls to that value.
  - Grow Height to Largest determines the "maximum height" of all selected controls, and then sets the "height" of the controls to that value.
  - Grow to Parent Width determines the width of the parent control, and then sets the "width" of the control(s) to that value.
  - Grow to Parent Height determines the height of the parent control, and then sets the "height" of the control(s) to that value.

- You may use the [Shift+Right Arrow], [Shift+Left Arrow],[Shift+Up Arrow] or [Shift+Down Arrow] key combinations to increase width, decrease width, increase height or decrease height of any individual control or group of selected controls.

- Use the "Nudge" toolbar options to move the controls pixel by pixel.

  - Nudge Up moves all selected components one pixel "up".
  - Nudge Down moves all selected components one pixel "down".
  - Nudge Left moves all selected components one pixel "left".
  - Nudge Right moves all selected components one pixel "right".

- Save your work frequently to avoid losing changes should your computer unexpectedly lose power or crash.

- Save the form and close the designer window prior to closing or exiting R:BASE.

Part VI

6 Lesson 5 - Creating and Modifying Reports

So far we have discussed acquiring data, and how to organize, edit, and query the data. Now we are ready to create a report. A report is data from your database that is organized to your specifications. Reports include invoices, checks, statements, and address lists, as well as standard reports. You can print reports on a printer, you can display them on a screen, and you can save reports to a file.

R:BASE offers many tools to create attractive, customized documents. Using some of these tools, you will create two reports, one for a list of employee phone numbers, and one for a list of scheduled flights.

At the end of this lesson you will be able to:

- Start the Report Designer
- Create a quick report and a custom report
- Place text and fields on a report
- Create expressions and place the variables in a report
- Use header, footer, and Detail sections
- Preview a report to the screen
- Print a report
- Print a HTML report

When Amelia and Charlie talked about skyWrite Ink's business information needs, they asked for various lists and combinations of information: employee lists, address labels for mailings to customers, and flight schedules. The report-building features of R:BASE will give skyWrite Ink the information it needs.

6.1 Defining a Report

When you create a report, you must associate it with a table or view; this table or view provides the majority of the information you want to include in the report. If you want information from other tables, you can use expressions to look up this information.

One way to include information from more than one table in a report is to build a query to specify the data you need, and then save the conditions as a view. You can then use the view as the driving view for the report.

You can create the following types of reports:

Quick Report (Report Wizard) - Use this option to create a report from a table or view you select using the Report Wizard. The Report Wizard automatically places column objects that correspond to columns you choose. A quick report doesn't contain any other features; however, you can add features to this report and change the layout.

When you create a quick report, you also select how the field objects will be displayed; in rows across the page, or in columns down the page.

Custom Report - Use this option to create a blank report, and then you can add the features you like.

We will be designing and building our reports in the Report Designer.

6.2 Creating an Employee Phone List

Designing is the most important step in building a report. Knowing what you want to include and how you want it to look will help you create a useful report.

To design a report:

- Decide the purpose of the report
- Sketch how you want the report to look

You already know that the purpose of this report is to generate a list of employee phone numbers, but what should it look like? Here's a sketch for the report:

Employee Phone List

First name	Last name	Area code	Phone
Larry	Miller	206	881-0276
Charles	Short	206	564-9123
Jessica	Jennings	206	882-0274
Samuel	Booth	206	637-8241
David	Wallace	206	881-0382
Diane	Vance	206	227-8132
Virginia	Hill	206	565-9205

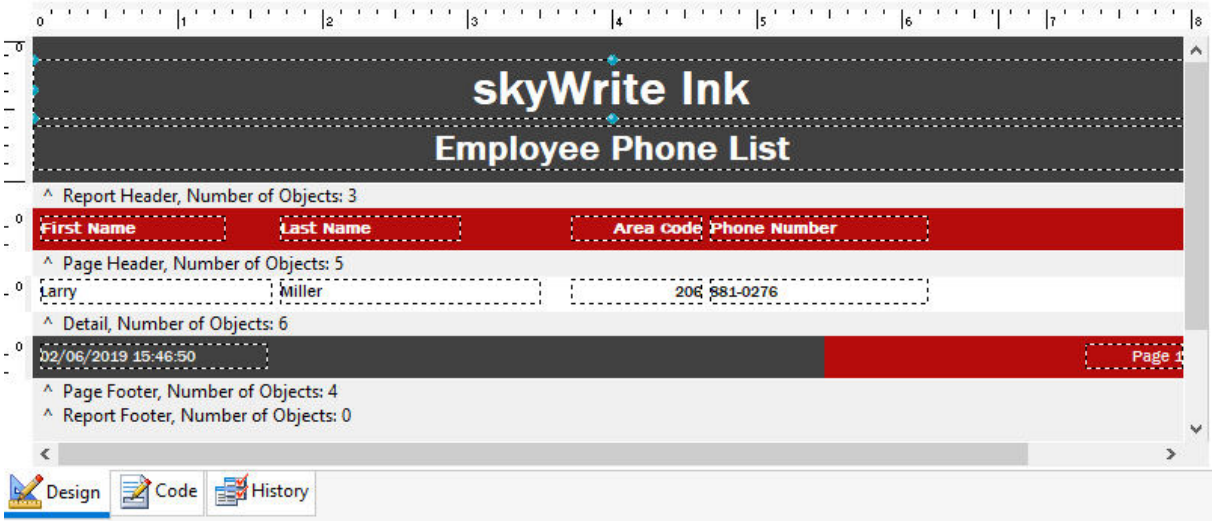
Use the sketch to identify the following:

- The driving table or view-this is the table or view on which the report is based. The driving table or view contains the rows and columns from which all or most of the data is gathered.
- Which columns to include
- The positioning of the columns on the report
- The report sections you'll need-There are five possible report sections. Each one has a specific purpose. You use report sections to determine how and where descriptive text will be printed and how the rows of data will be organized.

In this report, you want the title, *Employee Phone List,* and the column headings, *First name, Last name, Area code,* and *Phone,* to be printed at the top of each page. You'll place this information as descriptive text in a Page Header.

To print the names and phone number of employees, you'll place the fields for this information in a single line in the Detail section. The Detail section is repeated for each row of data drawn from the database.

Take a look at the finished report to see what you will be placing in the report sections.



6.3 Building the Employee Phone List with the Report Wizard

As you're building the Employee Phone List report, you can refer to the sketch as you need to. Make sure that you are connected to the *sky* database.

To start building the employee phone list report:

1. If the *sky* database is not open, click the "Databases" option in the Database Explorer and select "sky" and click the "Connect Database".
2. In the Database Explorer Group Bar, click the "Reports" option.
3. Click "New Report".

R:BASE displays the "New Report" dialog box.

4. Enter "emplist" in "Report Name".

Next, you must select the driving table or view on which the report will be based. Remember, this is the table or view that contains the rows and columns from which most data is gathered.

5. Select "employee" from "Table/View Name".
6. Enter "Employee Phone List" in "Report Comment:".

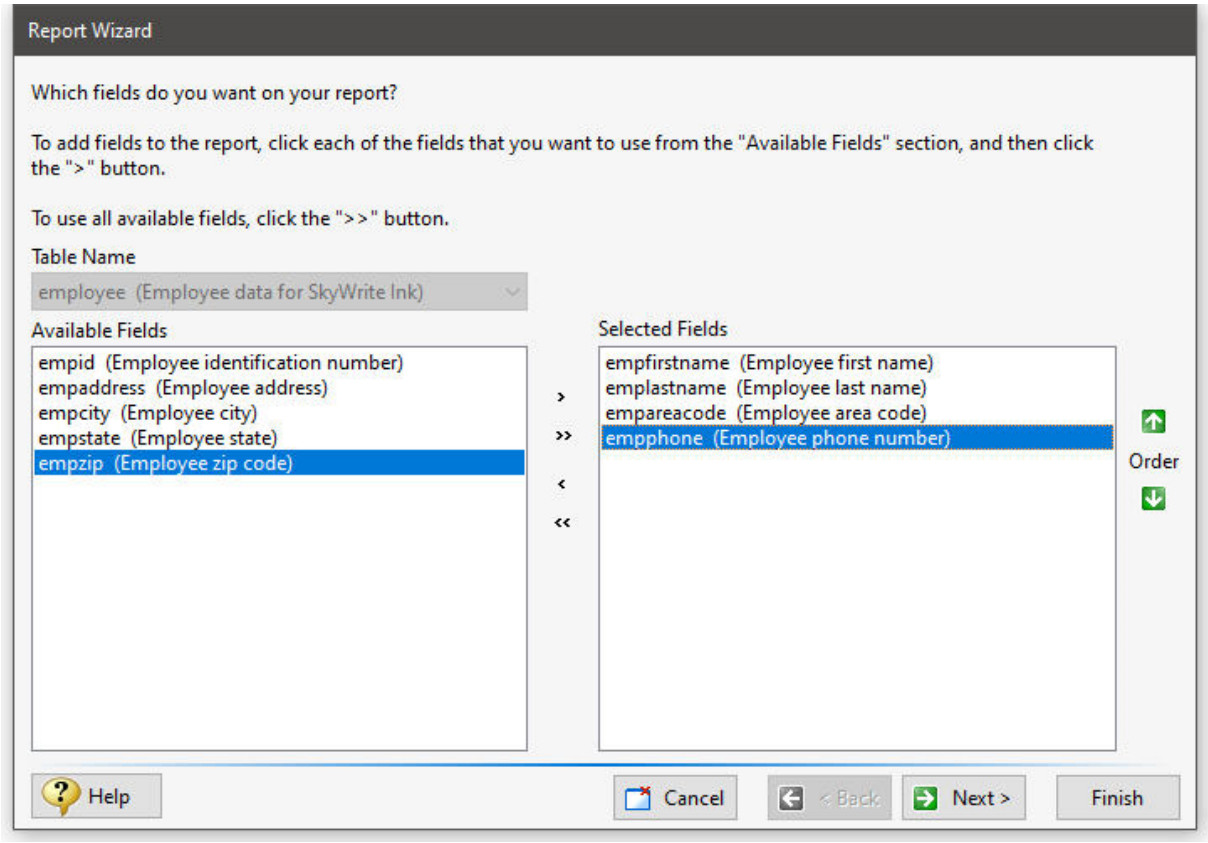
For the Employee Phone List report, we'll use the "Report Wizard" option to build a basic report, and then modify it.

7. Place a check in the "Report Wizard" check box.

R:BASE displays a list of the columns in the *employee* table. Because this report will be used as a telephone list, you don't want to include all of the columns-just those containing name and telephone numbers.

8. Using the arrows between "Available Fields" and "Selected Fields", one at a time select the following columns from the list: *empfirstname, emplastname, empareacode,* and *empphone* and move the column to the "Selected Fields" panel.

The green arrows for "Order" allow you to sort the columns in any order. The order should be *empfirstname, emplastname, empareacode,* and *empphone.*



9. Select the "Next" button.

The Report Wizard is prompting you to place a Break in the report. This item will not be used on this report and will be discussed later in this Lesson.

10. Leave the "Breaks" panel empty and select the "Next" button.

The Report Wizard now provides "Layout" and "Orientation" options for your report. You can toggle between the "Vertical" and "Tabular" Layout options to see how the report will look in the preview panel on the left.

In a "Tabular" layout report, the data from your columns is printed like a table on the report.

First name	Last name	Area code	Phone
Larry	Miller	206	881-0276
Charles	Short	206	564-9123

In a "Vertical" layout report, the data from columns is printed in rows.

First name: Larry
Last name: Miller
Area code: 206
Phone: 881-0276

First name: Charles
Last name: Miller
Area code: 206
Phone: 564-9123

11. Select the "Tabular" option for Layout.

You can also toggle between the "Portrait" and "Landscape" Orientation options.

12. Select the "Portrait" option for Orientation.

An additional option is available to use the field descriptions as captions, where the column descriptions are applied to label objects associated to the corresponding DB Label objects.

13. Check the option for "Use field descriptions as captions"
14. Click the "Next" button to accept the information you've entered.

With the Report Wizard, you are now given the option to add a style to your report. You can toggle through all the styles and see the available colors and fonts. For skyWrite, Ink, we will follow the same color pattern.

15. Select "Corporate" and select the "Next" button.

The final option prompts you to preview the report or modify the report's design.

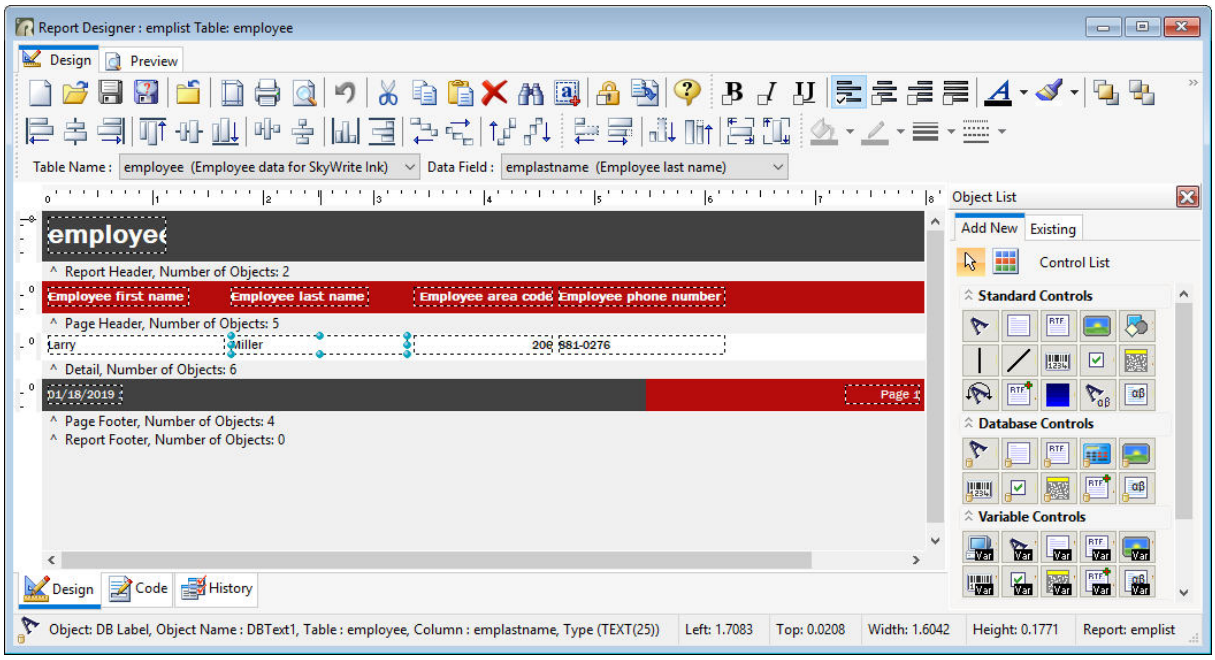
16. We will go straight to the designer. So leave the "Modify the report's design" option selected, and select "Finish".

R:BASE uses all the information you've entered to build a basic report and displays the report in the "Report Designer" window.

6.4 The Report Designer

The Report Designer includes all of the features and options you need to build reports for your database. Here, you establish your report layout, place objects to display text and data, and customize the look of your report. Once reports have been defined, the report names are listed in the Database Explorer.

Once you enter the information that R:BASE needs to build the report-the report name and description, the driving table, the columns you want to include, and the orientation-R:BASE displays the report in the "Report Designer" window. The following example shows the "Report Designer" window with the Employee Phone List report displayed in it.



In the "Report Designer" window, the Menu Bar now includes menu options specific to the Report Designer: File, Edit, View, Report, and Variables.

Two tabs are located on the top left of the Report Designer screen, "Design" and "Preview". Using the tabs, you can preview the report layout before printing.

With the Report Wizard, R:BASE placed the areas where data will appear on the report. These areas are called field objects. There are different kinds of field objects that can be placed on reports. The field objects are categorized by control type, within the "Object List" toolbar, which provides options for adding new and reviewing existing form controls. These controls are used for placing different objects such as images, lines, shapes, bar codes, check boxes, text, column objects, and variable objects. Here is a list of the report toolbar controls:

- Standard Controls
- Variable Controls
- Database Controls
- Advanced Controls

Additional toolbars are available for editing the value, fonts, alignment, size, colors, format, etc. of the objects added to the report. Here is a list of these toolbars:

- Standard
- Format
- Edit
- Draw
- Align or Space
- Size
- Nudge
- Object Inspector

For additional information on each of the toolbar buttons, please refer to the Report Controls chapter in the Report Designer Help.

The Report Designer also includes the "Object Tree" and "Data Tree" toolbars, for the objects on the report. These are very helpful in finding objects and the properties of the objects in reports where there are a great number of items added.

Toolbars

While in the Report Designer, you can position toolbars on any edge of the client window and in any order. Detaching a dockable toolbar from the frame window is called floating the toolbar. Attaching a floating toolbar to the frame window is called docking the toolbar. A toolbar can be docked to any side of its parent window, or can be floated in its own mini-frame window. Whether the toolbars are docked or floating, you will see the hint for any toolbar button when the cursor is hovered over the button. The Report Designer will remember the last docked/floating setting when closing the designer.

When a toolbar is docked, it will look like this:

On the left side a series of characters are displayed to drag the toolbar into a floating position. The "character" can be one or more dots or pipe symbols "||" based on your operating system and current theme. The dots/pipes are used

When floating, the toolbar can be moved by clicking and dragging the caption area and can be resized by clicking and dragging the edge. When a toolbar is floating, it will look like this, where a border is wrapped around the toolbar and the name and a close button (x) is displayed in the caption:

Tips:

- Double click on the caption area of a floating toolbar to dock it to the last location it was docked
- When floating, the "x" is used to hide a toolbar
- The toolbars can be docked on the sides or bottom of the screen to provide a greater work space
- From the Menu Bar, select the View: Toolbars option to hide or show any toolbar

There are horizontal and vertical rulers that can be set to various measurements to help you control the layout of objects on your report. Right click on the ruler to edit the ruler setting. When moving your cursor in the workspace, you will see a line move within the horizontal and vertical rulers. This allows you to place objects at a specific measurement.

The Status Bar, across the bottom on the Report Designer window, gives you information about the report object such as the object name, type, and coordinates of a selected object. Select some of the objects on the report to see the details on each.

By default, a report has the left margin at 0 and the right margin at 8.0 inches. The margins are indicated in the report workspace and can be edited using the Page Setup, another tool to help you alter the display of your report. The Page Setup can be located from the Menu Bar at **File: Page Setup...**

While you are designing the report, you can see the column data by choosing **View: Show Data**. This setting may or may not be enabled already in the Report Designer. Toggle the setting on and off to see the how the column data can be viewed. Leave the setting off before continuing.

From inside the Report Designer, you can launch the Report Designer help file by pressing the [Shift] +[F1] hot keys, or by pressing the "Help" button on the "Standard" toolbar (above).

6.5 Modifying the Employee Phone List Report

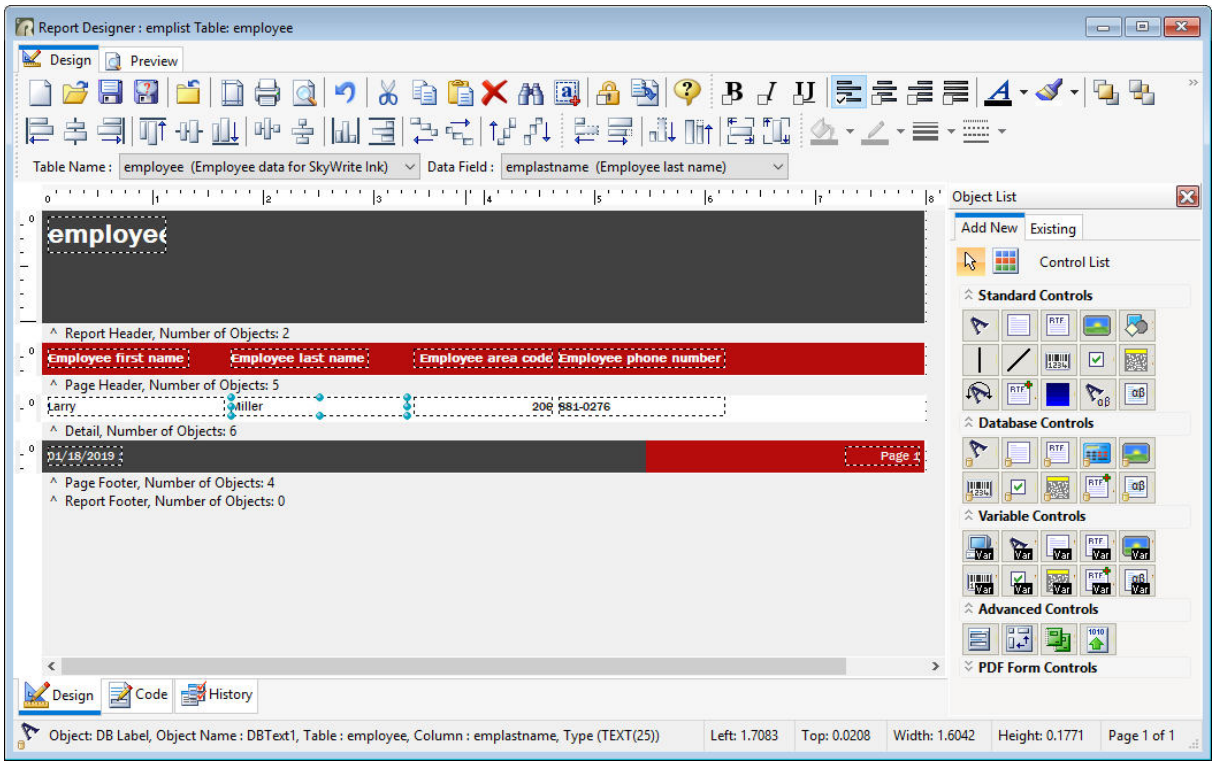
Your report already contains all the basic information that you need for the telephone list. Now, so that the report matches the sketch, edit the title, add the company name, change the column headings, and resize the column field objects.

6.6 Editing the Page Title

First, we want to edit the text object for the title in the Report Header section (the gray band across the screen marked Report Header) and increase the size of the section. A report header is printed at the top of the first page of the report.

To make the Report Header section larger:

1. In the "Report Designer" window, click on the gray band marking the bottom of the Report Header section and drag it down the screen to the 1 inch line on the ruler, making the section larger. You will notice the cursor will change to an arrow, pointing up and down.



The vertical ruler adjusts to reflect the new size of the Report Header section.

Now that we have made the Report Header section larger, we'll edit and add text for the report title.

To edit the report title text object:

1. In the "Report Designer" window, choose the object "employee" in the Report Header.

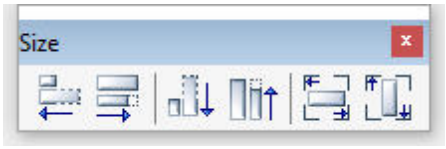
You will notice the blue handlebars on the selected item and that the "Edit" toolbar will become enabled with the text "employee" in it. If you do not see the Edit toolbar, deselect the object, and select it again.

The Edit toolbar will appear differently based on the selected item. If you select or add text, the text will appear in the toolbar for editing. If you select or add a column object, the Edit toolbar displays two drop down boxes showing the table in one, and the available columns in the second. If you select or add a variable, a drop down box appears with a list of defined variables.

2. Replace the text "employee" within the Edit toolbar with "Employee Phone List". To easily overwrite the existing text, double click in the Edit toolbar to select all the text, and then begin typing.
3. Hover your mouse cursor over the right handlebar and drag the right side of the object so you can see all of the letters.

Next, we will center the text so it appears directly in the center of the page using the toolbars.

4. With the text object still selected, click the "Grow to Parent Width" button on the Size toolbar. It is the second button from the right. Use the hints to find the correct button. Doing so will increase the text field to the same width as the report band.



5. Then, click the "Center Horizontally in Band" button on the Align or Space toolbar. It is the sixth button from the right. Use the hints to find the correct button. Doing so will center the object within the report band.

Similar to word processors and other Windows-based programs, you will notice the toolbar with a list of font styles and sizes with buttons to bold, italicize, and underline text. Next to those buttons are the buttons to edit the justification of selected text as well as color. This is the "Format" toolbar, which primarily formats text.

6. Select the "Center" button for text justification.

Now we will move the text down to make room for the company name to be added above the title. Moving objects in the Report Designer is the same as moving objects in the Form Designer.

To move a report object:

1. In the "Report Designer" window, choose the text "Employee Phone List".
2. Press and hold down the [Ctrl] key on the keyboard.
3. Use the down arrow key to move the text down, right above the Report Header band.

Now we will add the company name to the report.

To add a text object to the report title:

1. In the "Report Designer" window, first make sure the text "Employee Phone List" is selected, then choose the "Label" button.

The Label button is located on the Standard Controls toolbar with "A" on it. Like the Form Designer, when a button is chosen, it will remain selected.

2. Position the cursor above the existing report title and click the left mouse button.

The new text will take on the style, font, and size of the selected object on the report designer. The text should be Franklin Gothic Medium, 18, Bold. This saves times when adding multiple objects. The text in the label should read "Label1".

3. In the Edit toolbar, change the text from "Label1" to "skyWrite Ink".
4. Change the font size to 24.

Now we will center and resize the company name with the title using the "Align or Space" and "Size" toolbars again.

5. With the the "skyWrite Ink" text still selected, click the "Grow to Parent Width" button on the Size toolbar.
6. Then, click the "Center Horizontally in Band" button on the Align or Space toolbar.

The company text will now be the same width and size of the band.

7. Select the "Center" option for "Justification".
8. Click the mouse button elsewhere in the workspace to deselect the items.

If you want to move the company text up or down, select the item and use the [Ctrl] + [Up Arrow]/ [Down Arrow] keys.

Next, we'll edit the column headings.

6.7 Changing the Column Headings

R:BASE used the column comments in the *employee* table for the column headings on the report. The column heading text can be modified to improve the display of the report, by removing the repetitive "Employee" text, and adding capital letters for the all of the words.

To edit the column heading text:

1. In the "Report Designer" window, click on the "Employee first name" text object in the Page Header section.

Handlebars appear on the selected object and the "Edit" toolbar displays *Employee first name.* The Status Bar will display that you selected a "Label" object.

2. Change the text to "First Name" in the Edit toolbar.

3. Repeat this procedure for the rest of the column headings, using the following text:

Column Names	**Text**
Employee last name	Last Name
Employee area code	Area Code
Employee phone number	Phone Number

4. Choose **File: Save** to save changes to the report.

6.8 Resizing Objects

In the Report Wizard, the text and column field objects are sized to best fit the report. However, since the data contained in the columns vary in length, for example area codes with three digits, we can shorten the area code fields.

To shorten the area code object:

1. In the "Report Designer" window, click on the DB Label column object for area code in the Detail section.

Handlebars appear on the selected object and the Edit toolbar displays two drop down boxes, one containing the table name and another for the column name, *empareacode.* The Status Bar displays the name of the selected object, DB Label.

2. Press the [Shift] key and select the text "Area Code".

Red handlebars appear on the selected objects.

3. Press the [Shift] + [Left Arrow] keys to shorten the length of the items together.
4. Press the [Ctrl] + [Right Arrow] key to move the items to the right.
5. Click elsewhere in the workspace to deselect the objects.

The objects were made smaller and left some room to increase the last name object. Now, we'll adjust the size of the last name object.

To lengthen the last name object:

1. In the "Report Designer" window, click on the DB Label column object for *emplastname.*

Blue handlebars appear on the selected object.

2. Press the [Shift] + [Right Arrow] keys to increase the length of the item.
3. Choose **File: Save** to save changes to the report.

6.9 Report System Variables

By default, the Report Wizard added two system variables to display the date and page number in the Page Footer of the report.

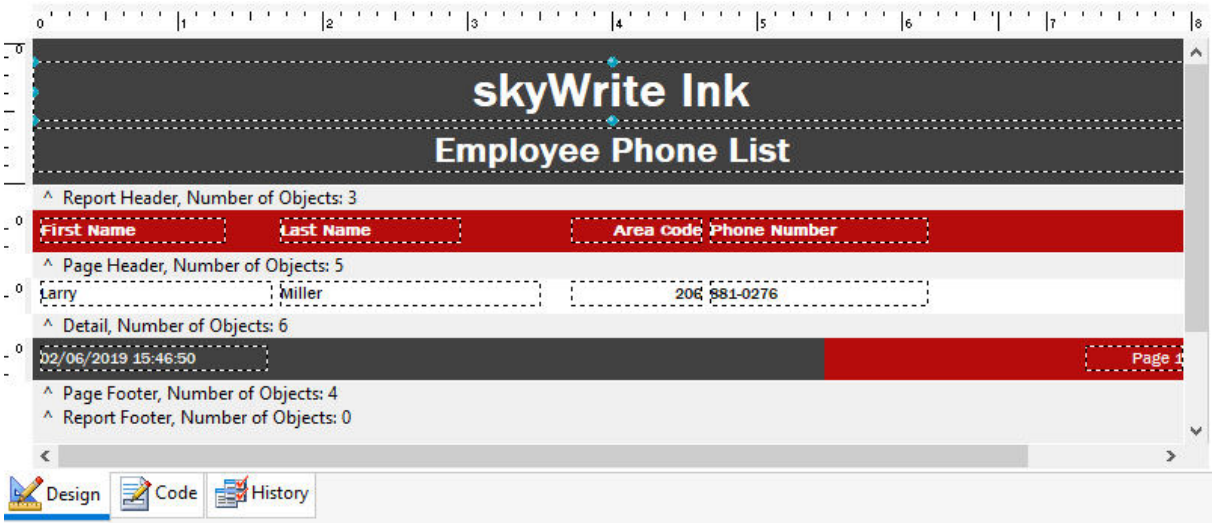
To view the available system variables:

1. In the Report Designer, select the System Variable on the left which has today's date.
2. Select the drop down box on the Edit toolbar.

Notice the variable is set to generate the "Print Date/Time" when the report is printed.

6.10 Final Result

This report prints the names and phone numbers for all the employees in the database. Your report is complete and should look like the following report:



Now you can preview the report.

6.11 Previewing the Employee Phone List

You can preview a report at any time during the design process "Report Preview" using the tab.

To preview the Employee Phone List report:

1. In the "Report Designer" window, click the "Preview" tab.

R:BASE displays the layout of the report in the preview window with data in the table.

2. Click the "Design" tab to return to the designer.

If the report didn't look the way you wanted it to, you could make some more changes. This report looks the way you designed it, so you can close the "Report Designer" window.

To close the "Report Designer" window:

From the Menu Bar, choose **File: Close**.

If you click the Reports tab in the Database Explorer now, you'll see that it includes *emplist.*

6.12 Printing the Employee Phone List

The report can be printed to the screen, a printer, or to a file.

To print the employee phone list to the printer:

1. In the Database Explorer, click the Reports option.
2. Select "emplist".
3. Click the "Print Report..." option.

The "Print Report emplist" dialog box is displayed. The *emplist* report description and driving table is displayed. The "Print Target:" is on the "Screen" option by default.

4. Select "Printer" from the "Print Target:" options.
5. Click the OK button and the report is printed to the current printer.

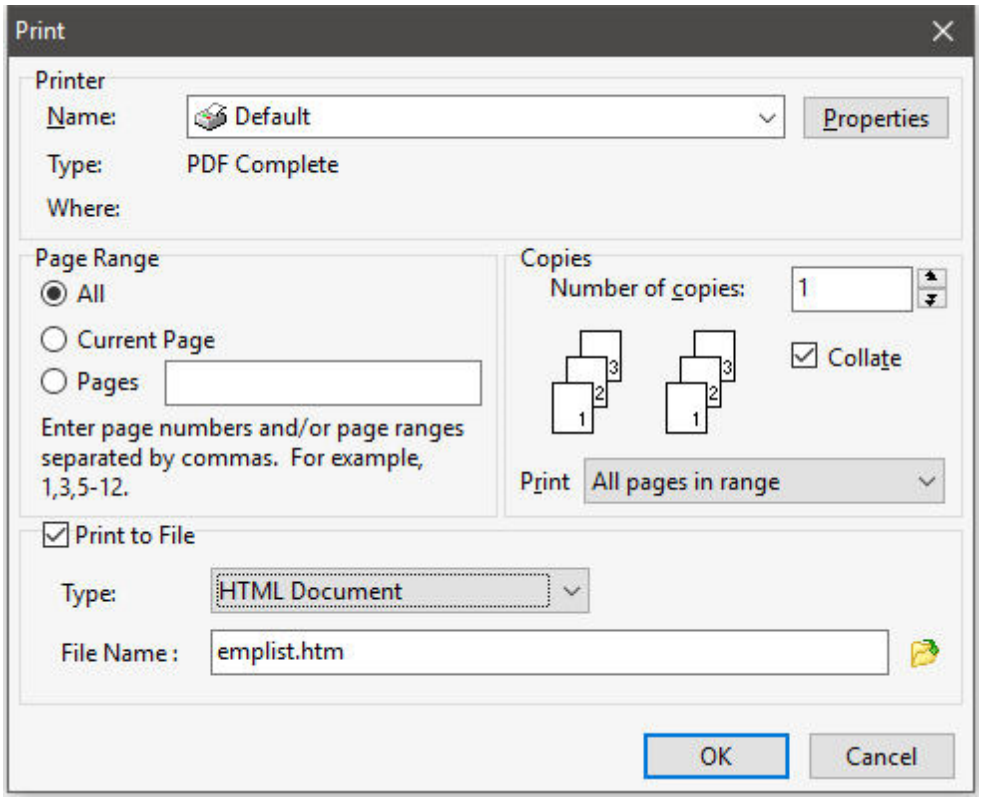
To print the employee phone list to a HTML file:

1. In the Database Explorer, click the Reports option.
2. Select "emplist."
3. Click the "Print Report..." option.

The "Print Report emplist" dialog box is displayed.

4. Select "File" from the "Print Target:" options.
5. Click the OK button.

The "Print" dialog box is displayed with a check in the "Print to File" option.



Select the "Type:' drop down box to display the different file types to print the current report.

6. Select "HTML Document" from the "Type:" drop down box.
7. In the "File Name:" field, specify the name of the HTML file. For this report, call it "EmpList.htm".
8. Click the OK button.

The next screen will provide many different options for setting up the HTML file format. For now, we will skip the additional options and move on.

9. Click the OK button.

The report is printed to the current directory where the database files are located and you are now at the Database Explorer screen.

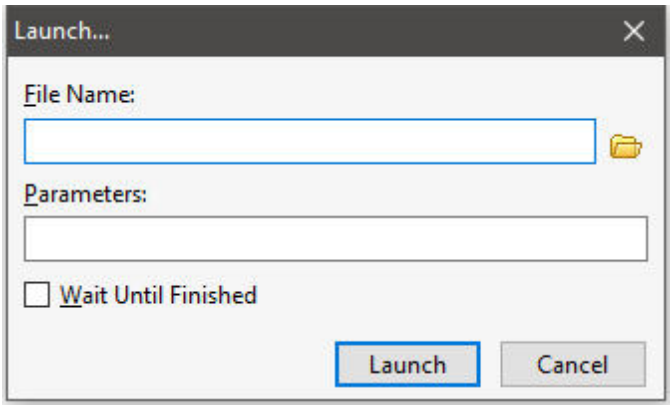
6.13 Launching Files

To view the HTML report you just created, you can launch the file directly from the Database Explorer.

To launch a file from the Database Explorer:

1. From the Menu Bar, choose **Tools: Launch...**

The "Launch..." dialog will appear.



2. In the "File Name" field, click the folder icon on the right side of the field.

The "Open" dialog will appear with a list of all the files in the directory.

3. Choose the HTML file that was just created, "EmpList.htm".
4. Select the "Open" button.

The path and file name will fill the field.

5. Click the "Launch" button.

The Employee Phone List HTML report will launch using the default Internet browser on your computer.

6. Close the browser windows to return to R:BASE at the Database Explorer.

Files may also be launched at the R> Prompt window using the LAUNCH command. To replicate the above example, you would type "LAUNCH EmpList.htm" at the R>.

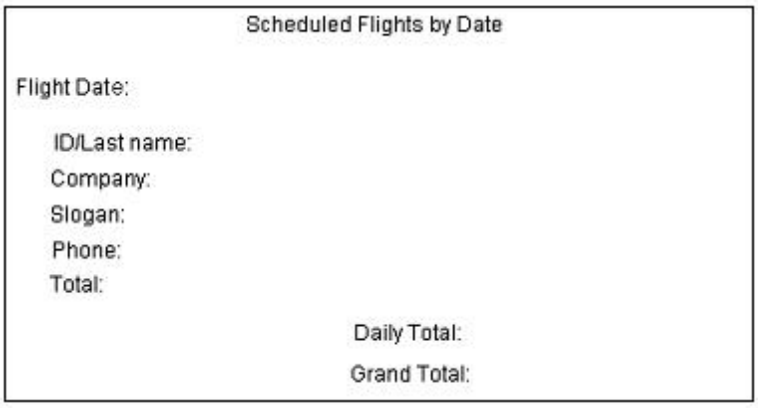
6.14 Creating a Scheduled Flights List

The next report you'll build is a list of scheduled flights. This report will be more complex than the Employee Phone List report. Before you build the report, take a look at the purpose and sketch the report.

6.15 Designing the Scheduled Flights Report

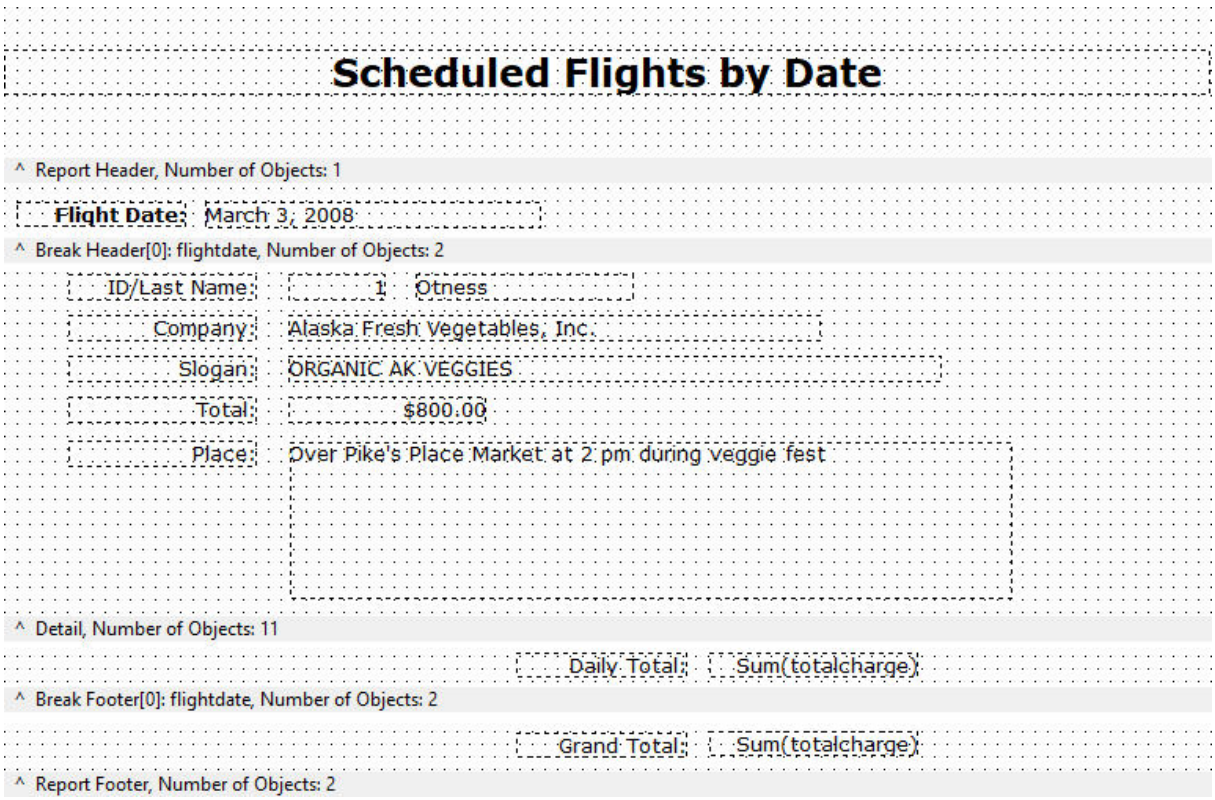
The purpose of the Scheduled Flights report is to list the flights scheduled for each day, to calculate the total revenue for each day, and compute the total revenue for all the flights shown on the report.

Here's a sketch of the report:



This report requires more report sections than the first report you built. As you can see, you can include any combination of report sections that you need.

Here's how the report will be set up in the Report Designer workspace:



6.16 Building the Scheduled Flights Report

The scheduled flights report begins as a Custom report. R:BASE opens the Report Designer with the five basic report sections, and then we place the text and column objects in the Report Designer workspace. As you are working on the Scheduled Flights report, use the sketch as a guide.

To start building the Scheduled Flights report:

1. If the *sky* database is not open, click the Databases tab in the Database Explorer and select "sky".

2. Click the Connect option.
3. In the Database Explorer, click the Reports option.
4. Click the "New Report" option.

R:BASE displays the "New Report" dialog box.

5. Enter "flightlist" in "Report Name".

Next, you must select the driving table or view on which the report will be based. Remember, this is the table or view that contains the rows and columns from which most data is gathered.

6. Select "flightview".
7. Enter "List of scheduled flights by date" in "Report Comment".
8. Leave the Report Wizard check box empty.
9. Click the OK button.

R:BASE opens the "Report Designer" window and displays a blank report.

6.17 Report Sections

A report is divided into five sections so that you can organize information. The section lines provide you a visual representation of the report organization. When you create a Custom report, the Report Designer marks each of the following sections:

Report Header (RH) - This section displays information once at the beginning of the report. For example, you can place the date and report title in this section.

Page Header (PH) - This section displays information at the top of each page. For example, you can place page numbers or section headings in this section.

Detail (D) -This section displays a block of data from the driving table or view. This section is repeated for each row of data.

Page Footer (PF) - This section displays information at the bottom of each page. For example, you can display amount totals for the data on each page.

Report Footer (RF) - This section displays information once at the end of the report. For example, you can place the report's grand totals in this section.

You can expand, contract, add, and delete these sections. For example, if you don't want to use a Page Footer section in the report, you can delete that section. Or, if you delete a section, you can add it back.

6.18 Removing Report Sections

For this report, we aren't going to placing any fields or text in the Page Header or Page Footer sections, so we'll remove those sections from the report.

To remove report sections:

1. In the "Report Designer" window, Choose **Report** from the Menu Bar.

In the drop-down menu, the sections are listed with a image to the left. A toggle is displayed to the left of the menu option, if the report section is set to appear on the report. Selecting the menu option will toggle the setting and add or remove the section from the report.

Click the menu option for **Report: Page Header**
Click the menu option for **Report: Page Footer**

R:BASE removes the Page Header and Page Footer sections and the screen adjusts accordingly.

6.19 Adding a Report Title

The report title is placed in the Report Header section. A Report Header section is printed only at the top of the first page of the report.

To add the report title:

First, you must drag the Report Header section down so we can add the report title.

1. In the "Report Designer" window, place the cursor on the Report Header gray band.

The cursor is replaced with a double-headed arrow.

2. Click and drag the band down to the "1 inch" mark on the ruler to create enough work space to add text.
3. Select the Label button from the Standard Controls (first button from the left).

4. Position the cursor in the Report Header section at the far left side, then click the left mouse button.

The label will appear in the Report Header section with the text "Label1" as well as the Edit toolbar will become enabled with the text "Label1" inside it.

5. In the Edit toolbar replace the text with "Scheduled Flights by Date".

Now, we will edit the font for the text. From the Format toolbar, you will see the available fonts and sizes.

6. With the object still selected, choose "Verdana" if it is not already selected.
7. Select "18" from the size drop down box.
8. Select the "Center" option for justification.

Next, we will center the text so it appears directly in the center of the page.

9. Just as done with the Employee Phone List report, click the "Grow to Parent Width" button on the Size toolbar.
10. Then, click the "Center Horizontally in Band" button on the Align or Space toolbar.
11. Click the left mouse button to deselect the object.

6.20 Changing the Size of the Detail Section

R:BASE makes all the report sections the same size in a Custom report. By looking at the sketch of the report, we see that we will be placing most of the information in the Detail section. We need to make the Detail section larger so all the information will fit.

To change the size of the Detail section:

In the "Report Designer" window, click on the gray band of the Detail section and drag it down. The Detail section needs to be at 2.5 inches in size. Use the vertical ruler on the left side of the Report Designer to gauge the size of the section.

Now, we're ready to place objects in the Detail section.

6.21 Placing Label (text) Objects

First, we'll place the Label objects that identify the data in columns.

To place Label objects:

1. In the "Report Designer" window, and with the title still selected, choose the Label button from the Standard Controls toolbar.
2. Position the cursor near the top of the Detail section by the left margin and click the left mouse button. Refer to the sketch of the report to help with positioning objects.

The Label will appear in the Report Header section with the text "Label2" as well as the Edit toolbar will become enabled with the text "Label2" inside it. The number next to the Label will increase as more Label objects are added to the report. The font size of the object will be set to 18.

3. Change the font size to 10.
4. In the Edit toolbar replace the text with "ID/Last Name:".
5. Select the "Right Justify" option for justification.
6. To see the complete text click and drag the right side of the object until all of the letters are visible.

Continue to place Label objects on the report. Remember, the objects added will use the same font, size, and justification as the object selected. If a report object with different font settings is selected, the "Format" toolbar would change to the properties of that object. Any objects placed after selecting that object will follow its properties. Repeat the steps to place the remaining text objects:

Company:
Slogan:
Total:
Place:

6.22 Aligning the Label Objects

After placing all of the Label objects, we'll align them vertically so the text is printed evenly. Use the alignment features available in the Report Designer instead of trying to place each object at a specific pixel location. Objects can be aligned both vertically and horizontally.

To align Label objects vertically:

1. In the "Report Designer" window, click on the "ID/Last name:" Label object. Press and hold the [Shift] key, then click on the "Company:", "Slogan:", "Total:", and "Place:" Label objects.

The Label objects are enclosed in red handlebars. If you prefer to use just the mouse, you can lasso the objects. To lasso the objects, place the mouse cursor to the top left corner of the objects, click and drag the cursor to the bottom right corner of the objects.

2. From the "Align or Space" toolbar, choose the "Align Right" button (third button from the left).

3. From the "Size" toolbar, choose the "Grow Width to Largest" button (second button from the left).
4. If the labels are not lined up along the left sides, choose the "Align Left" button from the "Align or Space" toolbar.
5. Click outside the objects to deselect them.

The selected objects are lined up along the right edges and all have the same width.

6.23 Spacing the Label Objects

After aligning the Label objects, they may not be perfectly spaced apart. You can use the spacing features available in the Report Designer instead of trying to space apart the objects yourself. Objects can be spaced both vertically and horizontally.

To space the Label objects vertically:

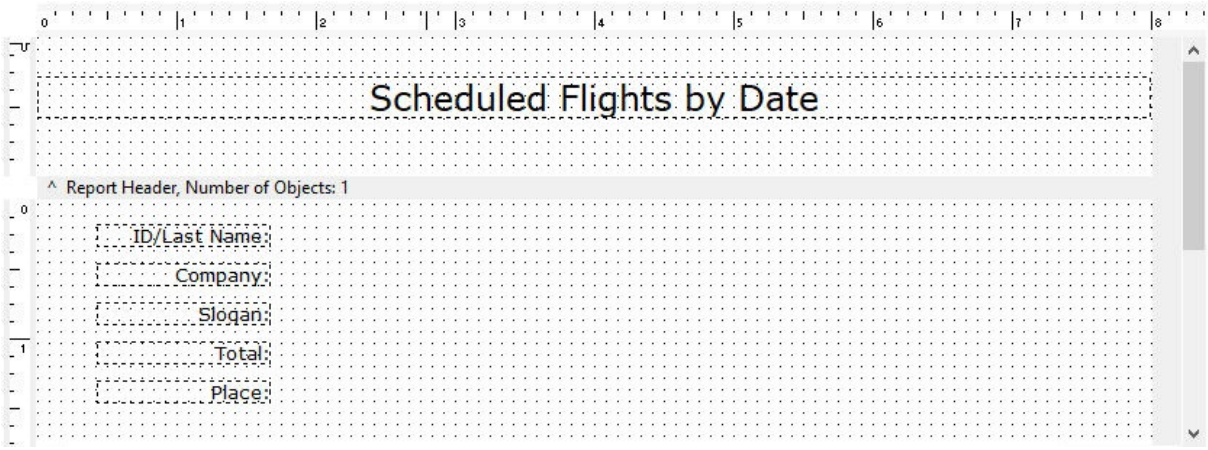
1. In the "Report Designer" window, click on the "ID/Last name:" Label object. Press and hold the [Shift] key, then click on the "Company:", "Slogan:", "Total:", and "Place:" Label objects.

The Label objects are enclosed in red handlebars. If you prefer to use just the mouse, you can lasso the objects. To lasso the objects, place the mouse cursor to the top left corner of the objects, click and drag the cursor to the bottom right corner of the objects.

2. Select the "Space Vertically" button on the the "Align or Space" toolbar (third button from the right).

The objects are spaced vertically.

3. Click outside the objects to deselect them. If you want to expand the spacing between the objects, move the "Place:" Label object down a few pixels and repeat the above steps.



6.24 Placing DB Label (column) Objects

Next, we place the DB Label objects opposite the corresponding text object. The *custid* and c*ustlastname* objects are placed on the same line. First, we place and resize the *custid* object, and then we place the rest of the DB Label objects.

To place the custid DB Label object:

1. In the "Report Designer" window, and with one of the Label objects still selected, choose the "DB Label" button from the Database Controls toolbar (the first button on the left, with the "A" and a picture of a database cylinder in the bottom left corner).

2. Position the cursor immediately to the right of the "ID/Last Name" text object and click the left mouse button.

The DB Label object will appear in the section with the text "DBText1". Take note that the Edit toolbar will become enabled with the "Table Name:" and "Data Field:" drop down boxes. The number next to the DB Label object will increase as more DB Label objects are added to the report. Also, with one of the existing Label objects selected while adding the DB Label, the Verdana font and size is automatically applied.

3. From the edit toolbar, select "flightview" from the "Table Name:" drop down box, and select select "custid" from the "Data Field:" drop down box.

We will make the *custid* DB Label object smaller. DB Label objects are automatically placed as a default size. We want to resize the object to accommodate the size of the actual data.

To resize the custid DB Label object:

The *custid* DB Label object is still the selected object. Drag the middle right handlebar of the *custid* object to the left until the width is approximately 1/2 of the original size.

Now we're ready to place the rest of the DB Label objects in the Detail section.

To place DB Label objects:

1. In the Report Designer with the *custid* DB Label object still selected, choose the "DB Label" button from the Database Controls toolbar.
2. Position the cursor immediately to the right of the *custid* column object and click the left mouse button.

The DB Label object will appear in the section with the text "DBText2" as well as the Edit toolbar will become enabled with the "Table Name:" and "Data Field:" drop down boxes.

3. Select "custlastname" from the drop down box.
4. Select the "Left Justify" option for justification.

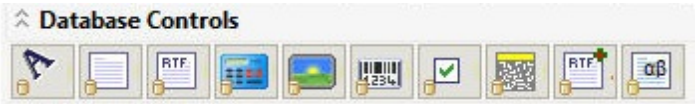
Place the rest of the DB Label objects to be printed in the Detail section of the report. Each of the objects is placed immediately to the right of its corresponding text object. The DB Label objects to be placed are: *company, slogan,* and *totalcharge.* Select the custlastname field before adding the DB Label objects to assign the same font, size, and justification.

6.25 Placing a DB Memo (column) Object

Next, we'll place a DB Memo object for the *place* column, which uses the NOTE data type. The DB Memo object contains different properties for the NOTE data type, one of which is the ability to stretch when large amounts of data is displayed and the object's size is not enough. As the *place* column is a NOTE data type, each entry in it can be very large. The DB Memo object can be used for any data type, and would be helpful if the report space is limited and data is needed to display down instead of across.

To place DB Memo (column) object:

1. In the "Report Designer" window and with the *slogan* DB Label object selected, choose the "DB Memo" button from the "Database Controls" toolbar (the second button from the left).



2. Position the cursor immediately to the right of the "Place" text object and click the left mouse button.

The DB Memo object will appear in the section with the text "DBMemo1" as well as the "Edit" toolbar will become enabled with the "Table Name:" and "Data Field:" drop down boxes.

3. From the "Edit" toolbar, select "flightview" from the "Table Name:" drop down box, and select select "place" from the "Data Field:" drop down box

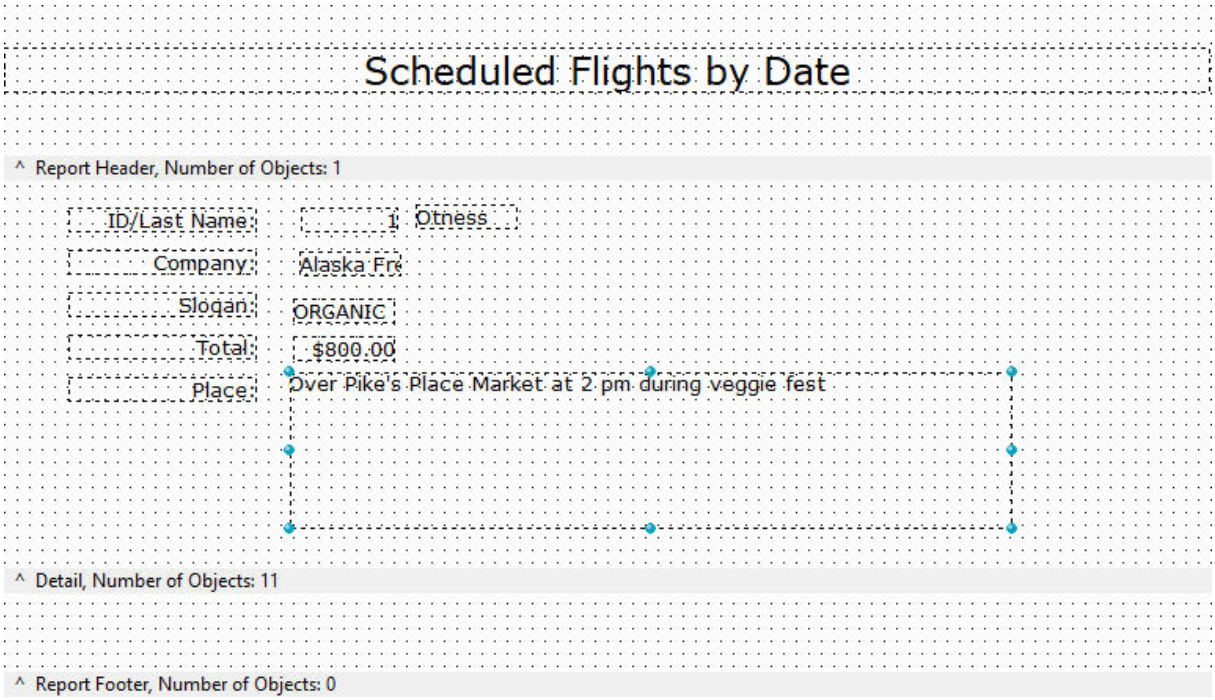
We will adjust the *place* DB Memo object to have a longer width, and a smaller height.

4. Place your cursor on the bottom right handlebar on the DB Memo object and drag the object to appear like the design sketch.

We will set the object to wrap the text when the data is longer that the width.

5. Right click on the DB Memo object and select the "Char Wrap" option.

Your report should look like the following report:



6.26 Resizing and Aligning Column Objects

The sizing for DB Label and DB Memo objects behave differently. The size of DB Label object will limit the data that appears on the printed report if the object's size is not large enough. To print more data in the object, you have to make the object wider. We will increase the length of some DB Label objects on the report.

To change the height and width of the DB Label column objects:

1. In the "Report Designer" window, click on the DB Label object for *company.*

Handlebars appear on the selected object.

2. Click on the handlebar in the middle of the right edge of the selected object, then drag it to the right until the size of the field is 3 inches in size.

Using the [Shift] + [Left]/[Right] arrow keys, you can also modify the width of the selected object. When comparing the exact location, height, and width measurements for multiple objects use the coordinates, Left, Top, Width, and Height, for the selected objects, which are located in the Status Bar across the bottom of the Report Designer.

3. Using the [Shift] + [Down] arrow key, to increase the height of the DB Label *company* object so the bottom of the letters like "g" and "y" can be seen. Repeat this step for the other DB Label objects, so all of the height for the DB Labels objects are set to 0.1771. Repeat this step as well for the Label objects, so all of the height for the Labels objects are set to 0.1771.

4. Change the width of the *custlastname* DB Label object to about 2 inches, the *slogan* DB Label object to about 3.5 inches, and the *totalcharge* DB Label object to about 1.5 inches. The *company* DB Label object should already be about 3 inches.

We will want the *place* column object to automatically wrap to another line if necessary as multiple lines may be in the data field.

5. Right click on the *place* DB Memo object, to access the speed menu, and select the "Stretch" option. This will increase the height of the object when more text is present than the size of the object.

After placing all of the column objects in the Detail section, align the column objects vertically. Then, align each column object with it's corresponding text object, ensuring the report is printed evenly.

To align objects vertically:

1. In the "Report Designer" window, click on the column object for *totalcharge.* Press and hold the [Shift] key, then click on the column objects *place, slogan, company,* and *custid.* Do not select the column object *custlastname.*

The column objects are enclosed in red handlebars.

2. From the "Align or Space" toolbar, choose "Align Left".

The selected objects are lined up along the left edges.

3. Click outside the objects to deselect them.

To align text and column objects:

1. In the "Report Designer" window, Click on the text object for *ID/Last name.* Press and hold the [Shift] key, then click on the column objects *custid* and *custlastname.*

The objects are enclosed in red handlebars.

2. From the "Align or Space" toolbar, choose "Align Top".

The selected objects are lined up along the top edges.

3. Click outside the objects to deselect them.

Use the previous procedures to align each the rows of text and column objects.

Before you make any more changes, preview the layout of the report.

6.27 Saving and Previewing the Scheduled Flights Report

You can save a report and preview it as often as you need to while you're building it to check the report layout.

To save and preview the Scheduled Flights report:

1. In the "Report Designer" window, choose **File: Save** from the Menu Bar.
2. Click the "Preview" tab on the Report Designer toolbar.

R:BASE displays the report in the "Preview" window. You need to make some more changes to make it match the sketch.

3. Click the "Design" tab.

Now you can improve the report by making a few more changes.

6.28 Modifying the Scheduled Flights Report

The following modifications are necessary to match the report to the sketch shown in "Designing the Scheduled Flights Report":

- Add a format that prints null values as blank.
- Group the rows by flight date by adding a breakpoint and placing the *flightdate* field.
- Add a field that computes the total revenue for each day.
- Add a field that computes the grand total for the entire report.

6.29 Adding a Format for Null Values

Some of your message requests might come from customers who are not associated with a company. Rather than having R:BASE leave the field blank, you can have it print "N/A" when there is no data for the *company* column.

To specify a null format:

1. In the "Report Designer" window, right click the mouse button on the *company* column object.

A speed menu of options is displayed.

2. From the pop-up menu, choose "Null Format...".
3. Enter "N/A" for "Null Format:"
4. Click the OK button.

Now you're ready to make the next change-adding a breakpoint.

6.30 Adding a Breakpoint

The next improvement you want to make is to organize the report so that each date appears only once, and all the flights for a date are grouped under it. You create a breakpoint, which groups the rows by the values in a specified column; in this case, the column *flightdate.* In addition, when you use a column as a breakpoint, R:BASE can sorts the data in the column in ascending or descending order. For example, the report will contain all the flights scheduled for 03/02/2009 followed by the flights for 04/01/2009, and so on.

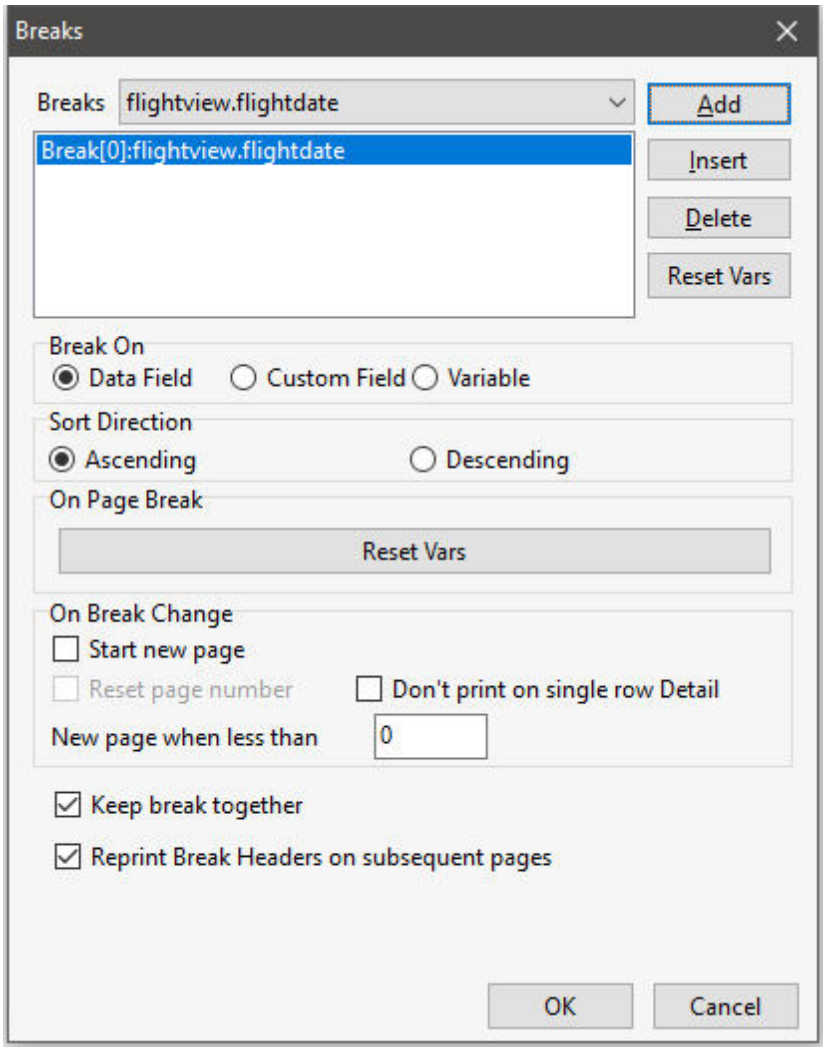
To add a breakpoint:

1. In the "Report Designer" window, choose **Report: Breaks...**

R:BASE displays the "Breaks" dialog box. The drop down box contains a list of columns for the report to break on. We are using *flightdate*.

2. Choose flightview:flightdate from the "Breaks" drop down box of available columns.
3. Click the Add button.

The first available breakpoint, Break[0], is added to the list of current breakpoints.



4. All of the default setting will be kept. Click the OK button.

R:BASE returns to the "Report Designer" window and automatically adds the Break Header and Break Footer sections to the report. The report bands are labeled with the name of the report section and the column that the sections break on, *flightdate.*

6.31 Placing Objects in the Break Header

Now you're ready to place the *flightdate* objects into the Break Header section. You've already placed both Label and DB Label objects on this report. You use the same procedure to place objects in the Break Header section as you did to place objects within the Detail section. The first thing we must do is stretch the Break Header section down so we can place the objects inside it.

To place text objects:

1. In the "Report Designer" window, hover your cursor over the "Break Header[0]: flightdate" section, click and drag the band a 1/2 inch down.
2. From the "Standard Controls" toolbar, choose "Label". Remember to select an existing object first to capture the same font.
3. Click the left mouse button near the left margin in the Break Header section.

The "Label7" text will appear as well as the "Edit" toolbar becomes enabled.

4. Enter "Flight Date:" in the Edit toolbar.
5. Select the "Right Justify" button on the "Format" toolbar.
6. Stretch the width of the Label object so all of the letters are visible.

Now, place the *flightdate* DB Label object.

To place the DB Label objects:

1. In the "Report Designer", choose the "DB Label" button from the "Database Controls" toolbar.
2. Click the left mouse button immediately to the right of the "Flight Date:" Label.

The "DBText6" text will appear as well as the "Edit" toolbar becomes enabled.

3. Select "flightdate" from the list of columns within the "Edit" toolbar.
4. Select the "Left Justify" button on the "Format" toolbar.
5. Stretch the width of the DB Label object to about 3 inches by raging the handlebars.
6. Click the mouse button in an empty area of the workspace to deselect the object.

The Label and DB Label objects for the breakpoint, *flightdate,* are placed in the Break Header section. Next, we align them to make sure they are printed on the same line.

To align the flightdate Label and DB Label objects:

1. In the "Report Designer" window, click on the "Flight Date:" Label object.
2. Press and hold the [Shift] key, then click on the DB Label object for *flightdate.*

The objects are enclosed in red handlebars.

3. From the Align or Space" toolbar, choose "Align Top".

The selected objects are lined up along the top edges.

4. Increase the height of the objects using the [Shift] + [Down] arrow keys until you can see the bottom part of the letter "g" in "Flight".
5. Click outside the objects to deselect them.

On this report, we want dates printed with a full month name, date, and a four-digit year. To do so, we'll add a Display Format to the *flightdate* DB Label object.

6.32 Adding a Display Format

The Display Format offers different display options for an object, based on the data type of the object. A CURRENCY data type object will provide options to alter how a dollar amount is displayed. A DATE data type object will provide options to display the month, day, and year.

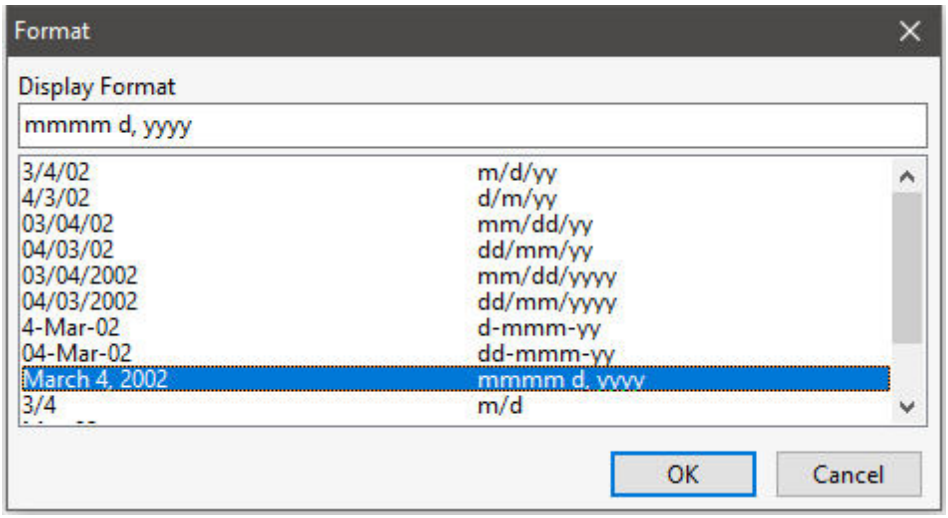
To add a Display Format:

1. In the "Report Designer" window, right click on the *flightdate* DB Label object.

A speed menu of options is displayed.

2. Choose "Display Format...".

The "Format" dialog box appears.



3. Choose "mmmm d, yyyy" from the right column.

The text appears in the Display Format field.

4. Click the OK button.

Next you'll add a field to compute the total daily revenue.

6.33 Choosing DB Calc over using a computed variable in reports

The DB Calc object in the R:BASE Report Designer eliminates the use of COUNT, MINIMUM, MAXIMUM, AVERAGE, and SUM column expressions used in earlier versions of R:BASE. The DB Calc object can be used in place of locations where a variable would be used to for a computation of COUNT, MINIMUM, MAXIMUM, AVERAGE and SUM for a column.

However, there are instances where you must use a variable instead of a DB Calc object. If you plan to use the result of the DB Calc object in the expression of another variable, then you must use a variable for the actual COUNT, MINIMUM, MAXIMUM, AVERAGE and SUM for the column.

6.34 Adding a Daily Total object using DB Calc

To compute the total daily revenue, you'll place a DB Calc object. The DB Calc object is a data-aware report object used to perform common report calculations and print the resulting value such as COUNT, SUM, MINIMUM, MAXIMUM and AVERAGE. The calculation is performed based on the value of the column specified in "Edit" toolbar. The first thing we must do is stretch the Break Footer section down so we can place the objects inside it.

To add the Daily Total DB Calc object to the Break Footer:

1. In the "Report Designer" window, hover your cursor over the "Break Footer[0]: flightdate" section, click, and drag the band a 1/4 inch.
2. From the "Database Controls" toolbar and with one of the other objects selected, choose the "DB Calc" button (looks like a blue calculator).
3. Click the left mouse button in the Break Footer at the 4.5-inch mark on the horizontal ruler.

The DB Calc object will appear with the aggregate function "Sum(?)" as the default and the "Edit" toolbar will become enabled.

4. From the "Edit" toolbar, choose the column object *totalcharge*.

The "SUM" DB Calc object adds the values in *totalcharge* for the rows in each date group because you placed it in the Break Footer.

5. Using the handlebars, stretch the width and height of the DB Calc object.

To place a Label describing the Daily Total DB Calc object:

1. In the "Report Designer" window, choose the "Label" button from the "Standard Controls" toolbar.
2. Click the left mouse button in the Break Footer section to the left of the DB Calc object.

The "Label8" text object will appear as well as the "Edit" toolbar will become enabled.

3. Enter "Daily Total:" in the "Edit" toolbar.
4. Select the "Right Justify" button on the "Format" toolbar.
5. Using the handlebars, stretch the width and height of the Label object.

Now you can define the fields for the grand total.

6.35 Placing Objects in the Report Footer Section

Now you'll place the grand total DB Calc and Label objects in the Report Footer section. The Report Footer section is printed at the very end of the report after all the other data has been printed. The Report Footer section is automatically created wide enough so that the grand total will be separated from the body of the report.

6.36 Adding a Grand Total DB Calc object

We will add a DB Calc object for a grand total field for the Report Footer section.

To define the Grand Total DB Calc object:

1. In the "Report Designer" window, choose the "DB Calc" button from the "Database Controls" toolbar.
2. Click the left mouse button in the Report Footer under the Daily Total DB Calc object.

The DB Calc object will appear with the aggregate function "Sum(?)" as the default and the "Edit" toolbar will become enabled.

3. From the "Edit" toolbar, choose the column object *totalcharge*.

This "SUM" DB Calc object will add all the values in the *totalcharge* column for all the rows in the report because you placed it in the Report Footer section.

4. Using the handlebars, stretch the width and height of the DB Calc object.

To place a Label describing the Grand Total DB Calc object:

1. In the "Report Designer" window, choose the "Label" button from the "Standard Controls" toolbar.
2. Click the left mouse button in the Break Footer section to the left of the DB Calc object.

The "Label9" text object will appear as well as the "Edit" toolbar will become enabled.

3. Enter "Grand Total:" in the "Edit" toolbar.
4. Select the "Right Justify" button on the "Format" toolbar.
5. Using the handlebars, stretch the width and height of the Label object.

To align objects across sections:

1. In the "Report Designer" window, click on the DB Calc object for *totalcharge* in the Break Footer section.
2. Press and hold the [Shift] key, then click on the *totalcharge* DB Calc object in the Report Footer section.

The two objects are selected as a group and enclosed in red handlebars.

3. From the "Align or Space" toolbar, choose the "Align Left" button.

The objects are aligned along the left edges.

4. Click outside the objects to deselect them.
5. Use the same procedure to align the text objects "Daily Total:" and "Grand Total:" along their right sides.

6.37 Changing Printing Styles

You can change the way text appears on a report. You can change the font for both text objects and column or variable objects. You can also change the color for both text and data. When designing reports to be printed on the screen, color can be used to emphasize areas of the report.

On this report, you'll make the report title and the text and data for flight date appear in bold print.

To make the report title bold:

1. In the "Report Designer" window, select the text object in the Report Header section.
2. On the "Format" toolbar, choose the "Bold" button (it has a capital B on it).

The report title text is now bold.

3. Repeat this procedure for the "Flight Date:" text field.

6.38 Finishing the Report

All the objects are placed in the appropriate sections and have been customized. The final step is creating the *flightlist* report is to finish aligning the objects and close up gaps in the report layout. Closing up gaps in the report layout can be done by moving the objects closer to the band above the objects and dragging the bottom band up. Most of the objects were aligned as they were placed, but some touch up might be needed.

To check the object alignment and spacing of the report, preview the report by selecting the "Report Preview" tab. If any finishing touches are needed, modify the report in the Report Designer to make the alignment and spacing changes.

If you want to add a line object to separate the flights, you would place it at the bottom of the Details section. If you want to add a line object to separate the Daily Total from the flights, you would place it in the Break Footer section.

You may have noticed that the Daily Total and the Grand Total amounts did not display a dollar sign or the cents in the preview screen. We will need to alter the display format for the total DB Calc objects like we did for the *flightdate* DB Label object in the Break Header section.

To change the Display Format for the DB Calc objects:

1. In the "Report Designer" window, right click on the *totalcharge* DB Calc object in the Break Footer section.

A speed menu of options is displayed.

2. Choose "Display Format...".

The "Format" dialog box appears.

3. Choose the fifth option, "$#,0.00;-$#,0.00" from the right column.

The text appears in the Display Format field to display the data with a dollar sign, $, and the cents values.

4. Click the OK button.
5. Repeat this step for the *totalcharge* DB Calc object in the Report Footer section.

6.39 Saving and Closing the Report

There are different ways to save a report; you can save it using the Menu Bar under **File: Save**, you can press the [Ctrl] +[S] keys on the keyboard, you can use "Save" button on the "Report" the toolbar, or you can let R:BASE prompt you to save the report when you close the "Report Designer" window.

To close the "Report Designer" window:

1. In the "Report Designer" window, choose **File: Close** from the Menu Bar.
2. If you have made changes since the last time you saved the report, R:BASE asks if you want to save changes. Click the Save button.

R:BASE closes the "Report Designer" window.

If you click the Reports tab in the Database Explorer now, you'll see that it lists both *emplist* and *flightlist.*

6.40 Printing the Scheduled Flights Report

You preview a report in the Report Designer to check the layout of a report. You can also preview the report by printing it to the screen or printer.

To print flightlist to the printer or the screen:

1. In the Database Explorer, click the Reports option.

The names of the reports in the *sky* database are listed.

2. Select "flightlist" and click the "Print Report..." option.

R:BASE displays the "Print Report flightlist" dialog box.

You can print the report to a printer, the screen, or several file formats. To limit the rows that are printed, you can specify a WHERE clause by clicking the Where Builder button.

3. If you have a printer attached to your computer, select the "Printer" option, otherwise leave the Print Target set to the Screen.
4. Click the OK button to print the report.

If you send the report to the printer, R:BASE prints the entire report while displaying the page printing status, then returns you to the Database Explorer window.

If you send the report to the screen, you'll see the first part of page in the Print Preview window. Maximize the window to see the entire report. Use the arrows on the toolbar to display the next page of the report or the last page of the report. Press the "Close" button to return to the Database Explorer window.

6.41 Summary

In this lesson you have:

- Learned how to start the "Report Designer".
- Created a quick report and a custom report.
- Modified the "Employee Phone List" report by adding a Page Header section and changing the field labels.
- Modified the "Scheduled Flights by Date" report by adding a Report Header section, making field larger, defining a break header, moving fields, defining Display Formats, and changing the font of some of the report objects.

- Printed the "Scheduled Flights by Date" report.
- Created DB Calc objects

To check your knowledge, try answering the following questions:

1. What are the two steps to remember when designing a report?
2. What is a driving table or view?
3. Describe the following report sections:
   - Report header
   - Page header
   - Break header
   - Detail
   - Break footer
   - Report footer
4. What are the two ways to orient the data in a Quick report?
5. What are two ways to launch a file in R:BASE?

6.42 Tips

Below are valuable tips to help get the most productivity out of the R:BASE Report Designer:

- When the Report Designer is launched the main menu bar includes options specific to the designer: File, Edit, View, Report, and Variables.
- Two tabs are located on the top left of the Report Designer; "Design" and "Preview", where users can toggle between the report's preview and design, before printing.
- The properties for all report objects may be adjusted by using the right click context menu.
- To resize a report band, use the mouse by positioning the cursor over the gray rectangular area (band) below the white space, press and hold the left mouse button, and drag.
- You can cut, copy and paste one or more report controls at a time, either in the same report or between different reports.
- Maximize the Report Designer window and use the right click context menus as much as possible.
- You can paste external content (text, rich text, images) directly into the Report Designer from the Windows clipboard.
- Use the "Format" toolbar options (Font Name, Font Size, Font Style, and Justification) to set the font, size, style, colors, and justification of objects.
- When added new text-based label objects, select a control to mimic before selecting the desired control. When placed, the new object will take on the style, font, and size of the selected object, which saves time when adding multiple objects.
- Use the [Shift+Right Arrow], [Shift+Left Arrow],[Shift+Up Arrow] or [Shift+Down Arrow] key combinations to accurately increase width, decrease width, increase height or decrease height of any individual control or group of selected controls.
- Use the [Ctrl+Right Arrow], [Ctrl+Left Arrow],[Ctrl+Up Arrow] or [Ctrl+Down Arrow] key combinations to accurately move any individual control or group of selected controls.
- Set up the Page Size and Margins before placing any controls on any bands. Select **File: Page Setup...** from the menu bar.
  - Leave the Printer as 'Default' unless required for special circumstances where a hard-coded printer is assigned to only print on that particular printer
  - Verify the Paper Size, Width, Height, and Orientation
  - Leave the Paper Tray as 'Default' unless required for special circumstances where a hard-coded paper tray is assigned to only use assigned paper tray
  - Leave Layout settings as default, unless designing columnar reports

  - Adjust the Top, Bottom, Left and Right margin as required

- The width of the page (white space) displayed in the Report Designer is the actual width, after taking the page margins into consideration.

- To set the ruler units, use the **Report: Units** menu bar option or right click on the ruler and select the appropriate unit from the available context menu options. To hide the a ruler, right click on the ruler and select 'Hide' from the speed menu options. The ruler may be displayed again by selecting **View: Rulers** from the menu bar.

- The status bar, across the bottom on the designer window, provides information about the select object such as the name, type, coordinates, and/or table, column and variable details.

- You can position toolbars on any edge of the window and in any order. Whether the toolbars are docked or floating, you will see the hint for any toolbar button when the cursor is hovered over the button. The designer will remember the last docked/floating setting when closing the designer. From the menu bar, select the **View: Toolbars** option to hide or show any toolbar.

- Use the Magnifying Glass utility to display a zoomed area of the R:BASE environment and monitor, which is useful when editing objects very close to each other. The Magnifying Glass can be launched by selecting the **Utilities: Magnifying Glass** menu bar option or using the [Ctrl+Alt+M] key combination.

- The "Preview" tab displays only the first 100 records in the table/view. The record count can be decreased or increased in the Default Report/Label Settings. To view the Default Report/Label Settings, choose **Settings: Report/Label Designer: Default Settings** from the menu bar. A custom WHERE clause may be entered and applied to the preview results with the **Report: Preview WHERE Clause** menu option. The specified WHERE clause replaces the default (100 records) display limit for the preview results. The Preview WHERE Clause can be launched with the [Ctrl+Shift+P] key combination.

- To assign object display formats, right click on a field to use the context menu to view the "Display Format" dialog. R:BASE will determine the data type assigned to the object and display several formats appropriate for that data type. For example, a field object based upon the DATE data type will provide a list of commonly used date formats. After one of the pre-defined formats is selected, that format may also be customized.

- You should pre-define all global variables used in a report. You may access the R> Prompt window to define global variables or use the [Global] button while defining your Report Expressions. Variable may also be pre-defined within a report's "On Before Design" action. Select **Report: Actions: On Before Design** from the menu bar to defined variables as such.

- All report bands include right click context menu options to run a Custom EEP before or after the data within the band is generated.

- The Code tab allows users to easily review and modify the command syntax in Report Actions and for Custom EEPs in report bands. The EEP contents of the Code tab is intentionally read only, and supports the ability to double click the mouse or press the [Enter] key to launch the R:BASE Editor for code modifications.

- The History tab displays created report backups, every time changes are made to the report and then saved. A revision number and date/time are used to track changes over time. An option is available within the **Settings: Report/Label Designer: Default Settings** to disable the History feature, if desired.

- Use the EEP Map utility to displays all defined EEPs for a report in a single layout, arranging EEPs by location. Any Custom EEP may be launched in the R:BASE Editor by double clicking on the listed item, or by selecting the "Edit..." button. The EEP Map can be launched by selecting **Report: Document Custom EEPs: EEP Map** from the menu bar or using the [Shift+Ctrl+M] key combination.

- Use the DB Calc object when you need to calculate a SUM, COUNT, MIN, MAX or AVG aggregate function value for a table field, which can be placed in any report band for dynamic results.

- The COUNT option of DB Calc can be used in the Detail Band to display the line number for each record in the report.

- To prevent unauthorized viewing of a report, assign a "Print Password". To do so, choose **Report: Passwords: Set Print Password** from the menu bar. A design password may also be specified.

- Set a report band height property to "Dynamic" when you want the band to use page space on an "as needed" basis, which shrinks or stretches the band to accommodate the table data displayed. When the report band height is set to "Static", the band uses the exact amount of space specified (unless the band is not set as Visible, in which case it uses zero page space).

- Set the "Stretch" property of a Memo object when you want the height of the memo field to automatically stretch, to allow the entire contents of the memo to be printed. If you are framing the memo within a shape, set the shape's "Stretch With Parent" property to "True" (checked) and the height of the shape object will stretch to accommodate the height of the memo field. And, use the "Shift With Parent" property of the other report controls to determine whether the position of the report controls should move as the memo stretches.

- You can use the "Visible" property of each object, or the entire band, to control what is displayed on a report. To hide all the objects in a band, set the Visible property of the band to False (unchecked). To hide individual objects, set the Visible property of individual object to False (unchecked).

- Use True Type fonts (indicated by "T" icon) when possible. These render well to both the screen as well as printer output. If you are using a dot-matrix printer, the printer driver may supply printer fonts (indicated by a printer icon) that you can use to speed up the printing of the report.

- The "Format" toolbar contains the "Bring to Front" and "Send to Back" buttons which are used when layering report objects on top of one another. The controls on the top layer print last, and the controls on the bottom layer print first. Use the "Existing" tab of the Object List toolbar to see the layer order of the controls within each band.

- Use the Scrapbook utility as a repository for commonly used objects. The "Add to Scrapbook..." menu option is available when any object is right clicked. With the Scrapbook filled with content, those objects may be added to a report by launching the Scrapbook and double clicking on the desired item. The Scrapbook can be launched by selecting **View: Open Scrapbook** from the menu bar or using the [Ctrl+Alt+B] key combination.

- The Favorites Toolbar allows users to customize their favorite objects onto a single toolbar which stores their commonly used items. Select **View: Favorites** from the menu bar to open the "Favorite Controls" dialog, where desired objects may be gathered from Standard, Database, Variables, Advanced, and PDF Report Controls. The controls can be arranged in a desired order as well. After populating a list of favorites, the customized toolbar is displayed at the top of the Object List toolbar.

- Use the "Align or Space" toolbar options to position controls relative to one another and relative to the band in which they appear.

    - Align Left aligns a group of controls with the "left-most" control's position.
    - Align Middle centers a group of controls based on the "horizontal center" of the control group.
    - Align Right aligns a group of controls with the "right-most" control's position.
    - Align Top aligns a group of controls with the "top-most" control's position.
    - Align Center aligns a group of controls based on the "vertical center" of the control group.
    - Align Bottom aligns a group of controls based with the "bottom-most" control's position.
    - Space Horizontally evenly spaces a set of controls based on the "left-most" control's position and the "right-most" control's position.
    - Space Vertically evenly spaces a set of controls based on the "top-most" control's position and the "bottom-most" control's position.
    - Center Horizontally in Band centers a control "horizontally" within a parent control.
    - Center Vertically in Band centers a control "vertically" within a parent control.
    - Stack Left arranges a set of controls in a pile to the "left-most" control, or to the control that was selected first.
    - Stack Right arranges a set of controls in a pile to the "right-most" control, or to the control that was selected first.
    - Stack Up arranges a set of controls in a pile to the "top-most" control, or to the control that was selected first.
    - Stack Down arranges a set of controls in a pile to the "bottom-most" control, or to the control that was selected first.

- Use the "Size" toolbar options to adjust the width and height of controls.
    - Shrink Width to Smallest determines the "minimum width" of all the selected controls, and then sets the "width" of the controls to that value.
    - Grow Width to Largest determines the "maximum width" of all selected controls, and then sets the "width" of the controls to that value.
    - Shrink Height to Smallest determines the "minimum height" of all the selected controls, and then sets the "height" of the controls to that value.
    - Grow Height to Largest determines the "maximum height" of all selected controls, and then sets the "height" of the controls to that value.
    - Grow to Parent Width determines the width of the parent control, and then sets the "width" of the control(s) to that value.
    - Grow to Parent Height determines the height of the parent control, and then sets the "height" of the control(s) to that value.
- Use the "Nudge" toolbar options to move the controls pixel by pixel.
    - Nudge Up moves all selected components one pixel "up".
    - Nudge Down moves all selected components one pixel "down".
    - Nudge Left moves all selected components one pixel "left".
    - Nudge Right moves all selected components one pixel "right".
- Save your work frequently to avoid losing changes, should your computer unexpectedly lose power, or fail to respond.
- Save the report and close the designer window prior to closing or exiting R:BASE, or disconnecting from the database.

Part VII

7 Lesson 6 - Creating and Modifying Labels

In this lesson you will learn how to use the Label Designer to create mailing labels for the skyWrite customer list. The Label Designer provides you with many of the same tools you used to define reports. For example, you can define variables, set fonts and colors, and preview the label.

At the end of this lesson you will be able to:

- Start the Label Designer
- Select a template for the label
- Define variables
- Place objects on a label
- Preview the label on the screen

skyWrite Ink offers special skywriting rates several times a year. Amelia and Charlie have found that their most successful promotions are the ones mailed directly to their customers. To facilitate these mailings, they want a label that they can print on standard mailing stock.

7.1 Building a Label

The Label Designer is where you build a label. In the "Label Designer" window, you layout your label, specify a label size, and place objects to display text and data. To start the Label Designer, in the Database Explorer, click the Labels option. To make a new label, click the "New Label..." option. To modify an existing label, select the label name, and click the "Design" option.

The label you'll build in this lesson is a mailing label for the skyWrite customer list. These are the steps you'll follow in building this label:

- Design the label
- Choose a template
- Define the variables
- Place column and variable objects
- Save the label
- Preview the label

7.2 Designing a Mailing Label

Designing is an important step in building a label. Before you begin to build the label, you should know what you want to include and how you want it to look.

Customer's full name
Company name
Address2
Address3
City, State Zip

To give your final output a professional appearance, you'll need to define two variables to complete this label. Creating variables allows you to combine the information in two or more columns.

In the *sky* database, customer names are stored in two columns-*custfirstname* and *custlastname.* On the label, you want only one space between first and last name, so you'll define a variable that combines the values in these two columns. That way, even if a customer has a very short first name, there won't be any gap between it and the last name.

It's a good idea to define a variable for the line that contains the city, state, and zip code. By defining a variable for this information, you can add a comma after *custcity* and eliminate any gaps between the fields.

7.3 Building a Label for a Customer Mailing List

Building a label starts in the same manner as building a form or report. First, you name the label and specify the table or view the label will draw data from, and then layout the label in the "Label Designer" window.

To build a customer mailing list label:

1. Begin at the Database Explorer. If the database is not open, click the Databases tab, select *sky,* and click the Connect option.
2. Click the Labels option.
3. Click the "New Label..." option.

The "New Label" dialog box opens.

4. Enter "custlabel" in "Label Name:".
5. Select "customer" in "Table/View Name:".
6. Enter "Customer mailing list" in "Label Comment:".
7. Click the OK button.

The "Label Templates" dialog opens.

R:BASE gives you a choice of templates that represent many of the standard label stocks used today, which range from single-feed laser printer labels to continuous-feed 3 x 5 index cards. You can choose from a Dot Matrix printer to Laser and Ink Jet. The "Products" panels displays a large number of product numbers for labels sizes.

8. Select the Avery Standard "5161 - Address" from the "Products" panel. Press the "5" key on the keyboard to jump to that number.

The template preview displays 20 labels with 2 columns, 10 labels in each column.

9. Click the OK button.

R:BASE opens the "Label Designer" window. Before you begin, take a look at the components of this window. You will notice that the Label Designer and the Report Designer are very similar.

The Label Designer displays three sections. The horizontal gray bands across the screen are:

- Column Header
- Detail
- Column Footer

The Column Header and Footer sections would be used if any data needs to be displayed in the first or last label of a label paper column. If you placed a field object in the Column Header on a label, and printed the label on a sheet with 30 labels (3 columns of 10), the field object would appear across the top three labels. The Page Header and Page Footer bands are also available.

R:BASE has placed the template size displayed in the Label Designer workspace to reflect the template size we selected.

To change the template for custlabel:

1. In the "Label Designer" window, choose **File: Label Templates**

R:BASE displays the "Label Templates" dialog box.

2. Select the Avery Standard "5160 - Address" from the "Products" panel.

The template preview displays 30 labels with 3 columns, 10 labels in each column.

3. Click the OK button.

R:BASE will place the template size displayed in the Label Designer workspace to reflect the new template size we selected. The template will also be listed across the bottom of the designer in the status bar.

The next step in building this label is to define the variables that were identified in the sketch.

7.4 Defining Variables for the Label

When designing the label, you identified two variables you'll need to complete it. You may find that it's easier to define all the variables used in the label before you begin to place fields, rather than defining them as you're placing fields.

The two variables you'll define for this label are:

- v*fullname* for customer first and last name
- v*citystate* for city, state, and zip code

Defining vfullname and vcitystate

The variable *vfullname* concatenates the customer's first and last names together to remove extra spaces. This makes the label look better.

To define vfullname:

1. In the "Label Designer" window, choose **Variables: Add/Edit Variables...**

The "Expression Builder" dialog box opens, ready for you to enter an expression for the variable.

2. Enter "vfullname" under "Name:".
3. Select "TEXT" in "Type" drop down box.
4. Enter "(custfirstname & custlastname)" in the "Expression" panel.

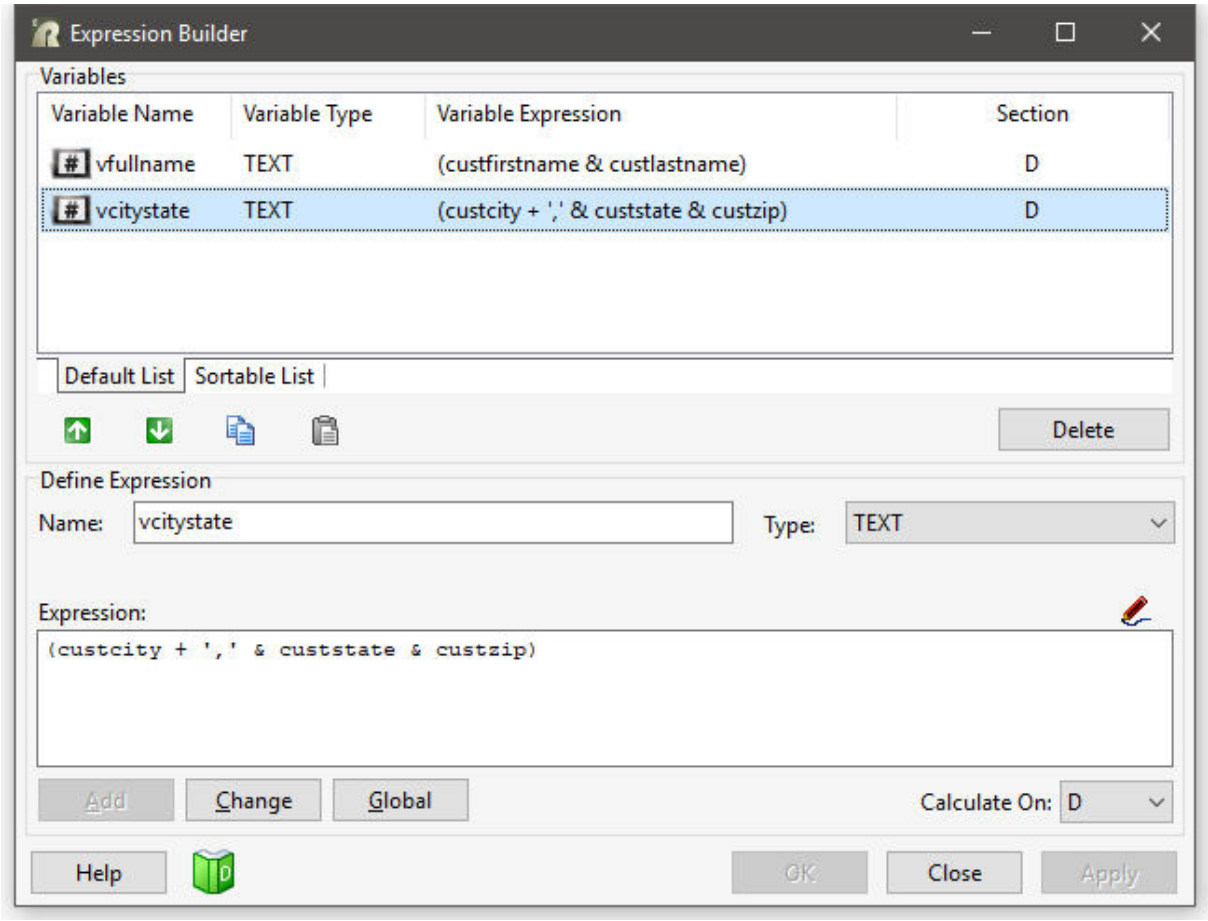
The ampersand (&) tells R:BASE to combine or concatenate the values in these two columns, leaving one space after the customer's first name.

5. Click the "Add" button.

R:BASE adds the variable *vfullname* to "Variables".

6. Enter "vcitystate" under "Name:".
7. Select "TEXT" in "Type" drop down box.
8. Enter "(custcity + ',' & custstate & custzip)" in the "Expression" panel.

The plus sign (+) tells R:BASE to concatenate the value in *custcity* and the text enclosed in the single quotation marks, leaving no space between them. The single quotation marks identify the comma (,) as a literal text string. This expression creates the following output: Olympia, WA 98076.



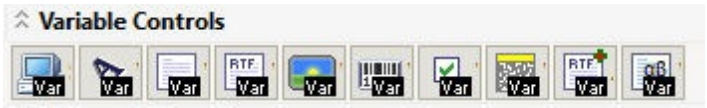
9. Click the "Add" button.
10. Click the "Apply", and then the "Close" buttons.

7.5 Placing Objects on the Label

You are now ready to place objects on the label. You can place the objects using the toolbars just in the Report Designer.

To place the vfullname object:

1. In the "Label Designer" window, choose the "Variable Label" button from the "Variable Controls" toolbar. The button has an "A" with the text "Var" across the bottom.



1. Position the cursor in the upper left comer of the label template, and click the left mouse button.

An object appears in the Label Designer workspace and the "Edit" toolbar becomes enabled. The Edit toolbar will contain all variables that are defined.

2. Select "vfullname" in from the Edit toolbar.

While the *vfullname* object is the selected object, we'll make the object wider. For a label, you can make all the objects as wide as the label template.

3. Click on the handlebar in the middle of the right side of the *vfullname* object and drag it to the right until the right edge of the object is close to the right edge of the label template. Release the left mouse button.

7.6 Placing the Remaining Objects

Using the toolbars, place the "DB Label" objects *company, custadd1,* and *custadd2.* Place the objects right below each other on successive lines. You'll align and size the objects after all the objects have been placed. Place the "Variable Label" object *vcitystate* below the *custadd2* column object.

To align and size the objects on custlabel:

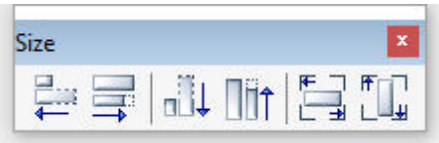
1. In the "Label Designer" window, choose **Edit: Select All**.

All the objects on the label are enclosed in red handlebars to indicate they are selected as a group.

2. From the "Align or Space" toolbar, choose 'Align Left" (first button on the left).

The objects are aligned along the left edges.

3. From the "Size" toolbar, choose "Grow Width to Largest" (second button from the left).



The objects are all resized to the size of the *vfullname* object.

4. On the "format" toolbar, increase the default font to "9" from "8".

The font for all of the objects will increase.

5. Select the "Space Vertically" button on the "Align or Space" toolbar.

The objects are evenly spaced vertically.

6. Click outside the objects to deselect them.

7.7 Previewing the Label

It's a good idea to preview a label on the screen before you print it so that you can verify that it's displaying information the way you intended. Just like the Report Designer, the Label Designer has the "Preview" tab.

To preview custlabel on the screen:

Select the "Preview" tab.

The "Preview" window displays the label to the screen with the customer data.

Select the "Designer" tab to return to the designer.

7.8 Saving and Closing the Label

You will want to save the changes made to the customer label.

To save a label:

In the "Label Designer" window, choose **File: Save**.

R:BASE saves the label.

To close the "Label Designer" window:

In the "Label Designer" window and choose **File: Close**.

If you click the Labels option in the Database Explorer now, you'll see that it lists the label just defined, *custlabel*.

7.9 Line Compression

You will notice spaces or null values "-0-" between the customer name and the address on the labels as not all customers have company names and not all customers have two address lines. We want R:BASE to compress the lines of the label if there is no data in a column. We will create a new label to utilize the line compression.

To print a label to compress the lines:

1. In the Database Explorer, choose the "New Label..." option.
2. In the "Label Name:" field, enter "custlabelc".
3. Choose "customer" from the list of tables/views.
4. Enter "Customer Label-compressed"
5. Click the OK button.

The "Label Templates" dialog is displayed. We will select the same template as the previous label.

6. From the list of "Products", choose the Avery "5160 - Address" label.
7. Click the OK button.

The Label Designer window is displayed.

The first step is to define a variable that contains all of the available columns used for the address.

8. In the "Label Designer" window, choose **Variables: Add/Edit Variables...**

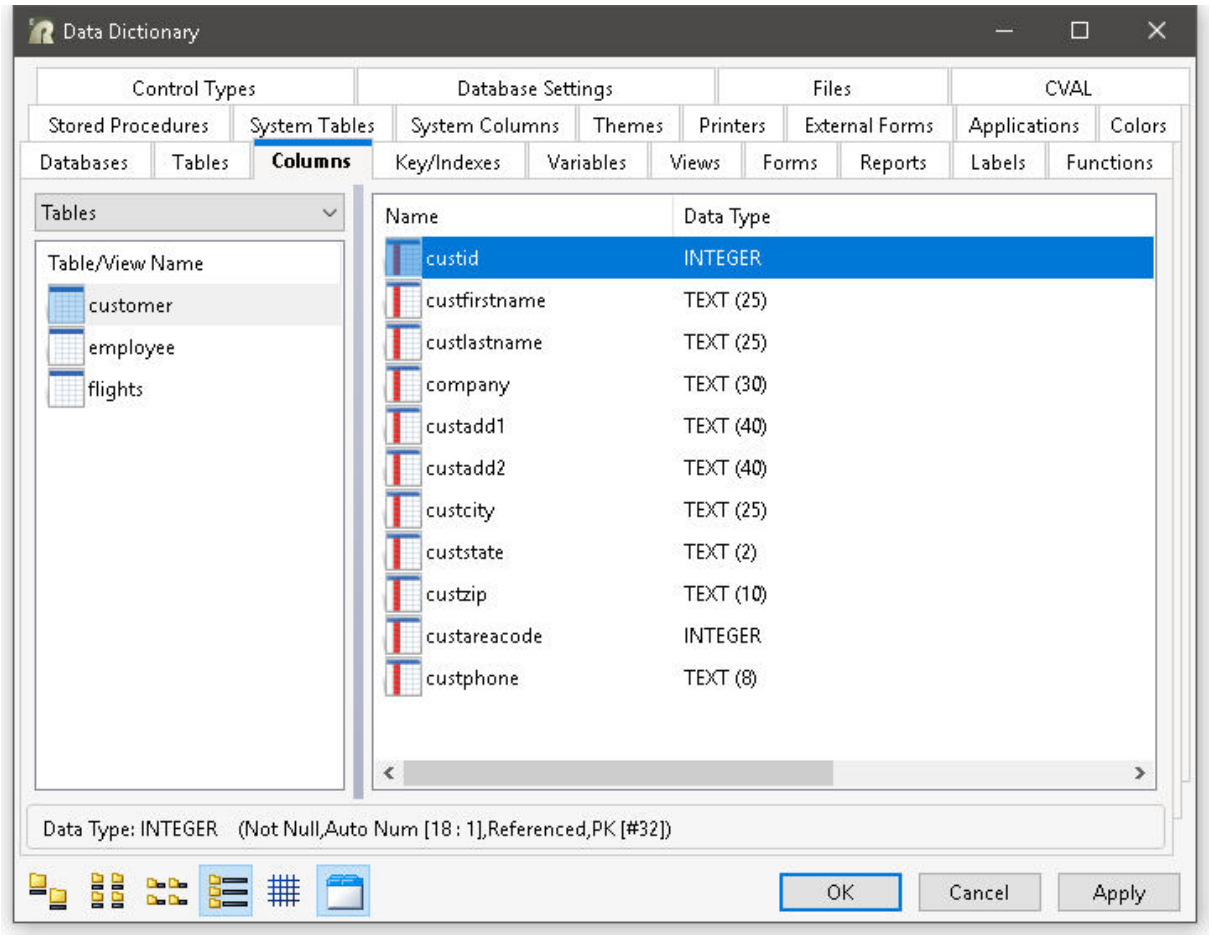
The "Expression Builder" dialog box opens, ready for you to enter an expression for the variable.

9. Enter "vAddress" under "Name:".
10. Select "TEXT" in the "Type" drop down box.
11. Place the cursor in the "Expression" panel and press the [F3] key to launch the Data Dictionary.

    The Data Dictionary is a utility that displays, and allows users to capture, information about the current R:BASE environment and/or connected database. For this label, the column names will be captured to insert into the variable expression.

12. Select the "Columns" tab, and then select the "customer" table.

    The Data Dictionary will display the columns defined for the *customer* table.



13. Within the column list, select "*custfirstname, custlastname, company, custadd1, custadd2*, *custcity, custstate,* and *custzip*", in this order. To select all of the columns, hold down the [Ctrl] key while selecting each column or use the mouse to lasso the columns from top to bottom.
14. After they are all selected, press the "OK' button.

    All of the column names will be inserted into the "Expression" panel. Now, the expression string must be built.

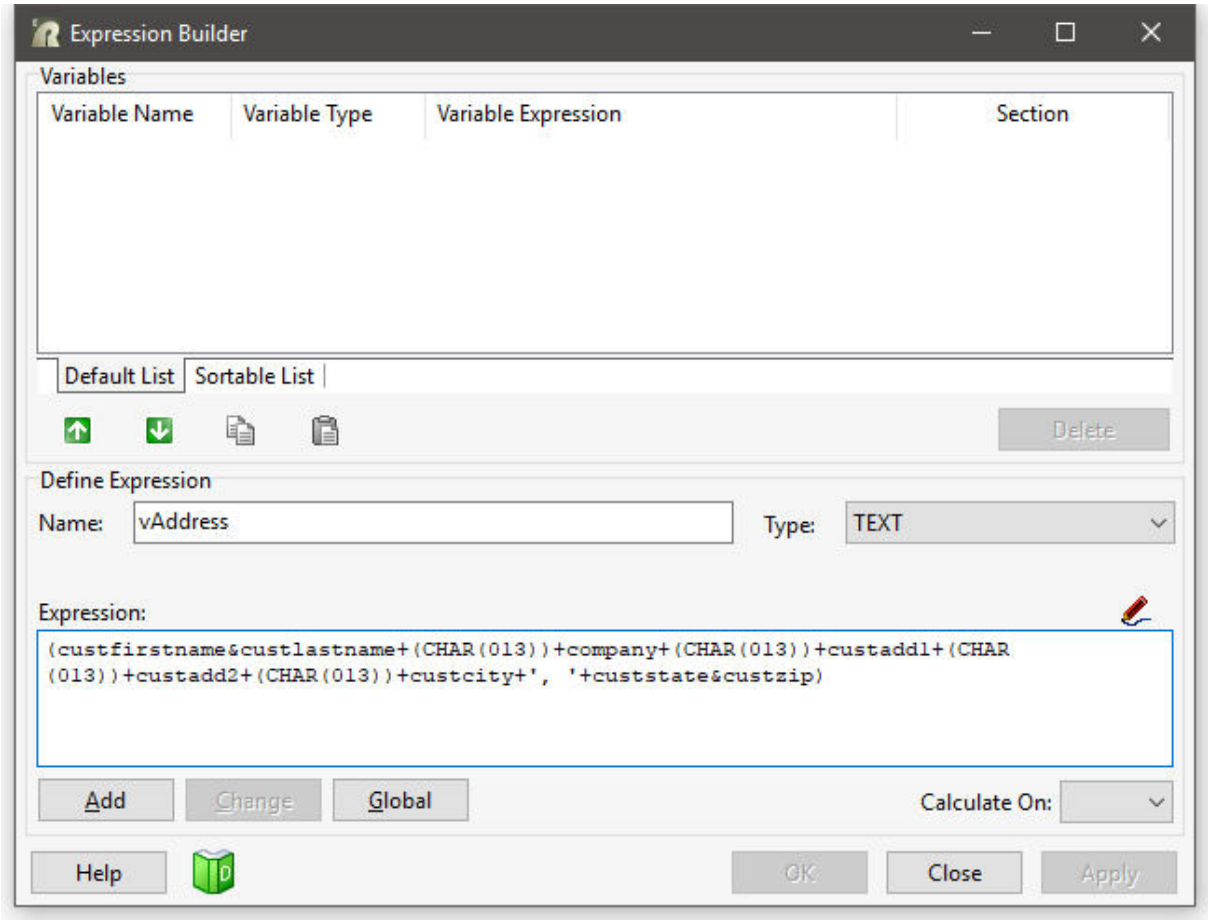
15. Place an open parenthesis at the beginning of the column list, and a close parenthesis at the end.
16. Between the *custfirstname* and *custlastname* columns remove the comma and insert the ampersand character (&).
17. Between the *custstate* and *custzip* columns remove the comma and insert the ampersand character.
18. Between the rest of the columns, remove the commas and place the plus character (+).
19. Next, between each each individual label address line insert the (CHAR(013)) function with a plus character. For example, the name will be on a separate address line than the company name, so insert the function there.

    The CHAR portion of the variable expression is another R:BASE function. It is categorized as a "conversion" function that converts an ASCII integer value to its corresponding character. The (CHAR(013)) function represents a carriage return, so when the address is printed, this portion will be sent to the line below.

    (CHAR(integer))

20. After the city column, add a comma and space enclosed in single quote characters followed by a plus character. This will inserts a comma and a space to correctly display the city state and zip line accurately.

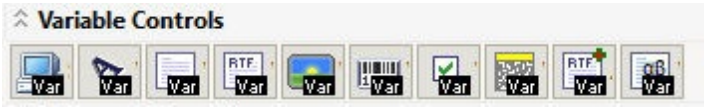
The image below is an example of what the variable expression should look like. Note that the name is separated from the company with the (CHAR(013)) function, along with the two addresses, and the city state and zip code columns, just like the standard address format. Also note that between each (CHAR(013)) function and a column there is a plus character. The plus characters are used to combine the text string.



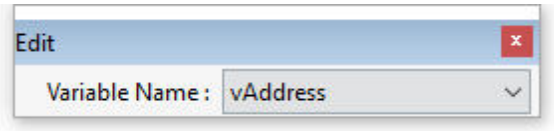
21. Select the "Add" button to save the variable. If you receive any errors, fix the typo(s), or simply copy and paste the following into the "Expression" panel:

    (custfirstname&custlastname+(CHAR(013))+company+(CHAR(013))+custadd1+(CHAR(013))
    +custadd2+(CHAR(013))+custcity+', '+coststate&custzip)

22. Press the "Apply" button to save the variable, and then the "Close" button to close the Expression Builder window.
23. Next, add a Variable Memo control to the top left corner of the label. On the Variable Controls Toolbar, it is the third button from the left.



24. Stretch the control so it fills the entire available label space.
25. With the control selected, assign the variable *vAddress* from the "Edit Toolbar.



26. From the "Format" Toolbar, change the font to "Tahoma", the size to "9", and the bold style by selecting the "B" button.
27. Now, preview the labels by selecting the "Preview" tab.

    The labels will display, only empty lines will appear where the data does not exist in the table. For example, the fourth label down for "Steven Fishman" does not have a company or a second address, so there are two empty lines in the address label.

28. Select the "Design" tab to return to the label Designer.
29. Then, right click on the Variable Memo control and select "**Suppress Empty Lines**" from the speed menu.

    This will remove the empty lines and create a fully dynamic address that will adjust for any missing lines.

27. Again, preview the labels by selecting the "Preview" tab.

    The empty lines should be removed from the labels.

28. Select the "Design" tab.

    Now, save the report.

29. In the "Label Designer" window, choose **File: Save**.
30. Close the Label Designer to return to the Database Explorer.

7.10 Summary

In this lesson you have:

- Learned how to start the "Label Designer".
- Learned how to concatenate text values.
- Created a mailing label for the sky Write customer list.
- Created line compression on a label using the Variable Memo object.

To check your knowledge, try answering the following questions:

1. Reports can have up to five sections; labels have three. What are they?
2. What would be the result of this expression, which concatenates two text values: 'Charlie' & Earbergh'?
3. What would be the result of this expression: 'Amelia' + 'Lindhart'?
4. How do you select a template?

Part VIII

8 Lesson 7 - Using R:BASE Commands

In this lesson, you learn to use R:BASE commands at the R> Prompt, and how to use variables and expressions.

At the end of this lesson you will be able to do the following:

- Open the R:BASE R> Prompt
- Use the Help at the R> Prompt
- Use R:BASE Commands at the R> Prompt
- Define Variables and Expressions

8.1 Opening the R:BASE R> Prompt

The R:BASE R> Prompt is the area of R:BASE in which you can enter SQL and R:BASE-specific commands for using a database. The R> Prompt is identified by the characters "R>" displayed on the top left side of the screen. The easiest way to learn how to use the R:BASE R> Prompt is to use it.

1. Start R:BASE, if you have not already done so.
2. On the main R:BASE toolbar, select the "R> Prompt" button (third from the left).

Note: Another way to open the R> Prompt is the [Ctrl]+[R] key combination.

You should now have a prompt, R>, displayed on the top left side of the screen. The R> Prompt window is divided into three parts; the Input Console, which is next to the "R>", the Output Console, and the Command History panel. When you type a command at the Input Console and press the [Enter] key, the "Output Console" displays the output results. The typed command at the input console is then cleared. The Command History panel stores all commands that were entered at the R> for an R:BASE session and allows users to recycle commands entered.

The R> Prompt indicates that you can enter commands to use R:BASE. Using R:BASE commands requires only that you know what command you want and how the command needs to be phrased (syntax) for R:BASE to understand it. The R:BASE online help provides the complete command syntax for all R:BASE commands within the Command Index.

The next section leads you through the different ways you can use the R:BASE help.

8.2 Using the Help at the R> Prompt

There are help screens that you display by entering the word HELP at the R> Prompt or by pressing the [F1] function key.

1. At the R> Prompt, enter "HELP" and press the [Enter] key, or press [F1].

R:BASE displays the main help window, with the right pane showing the first help screen.

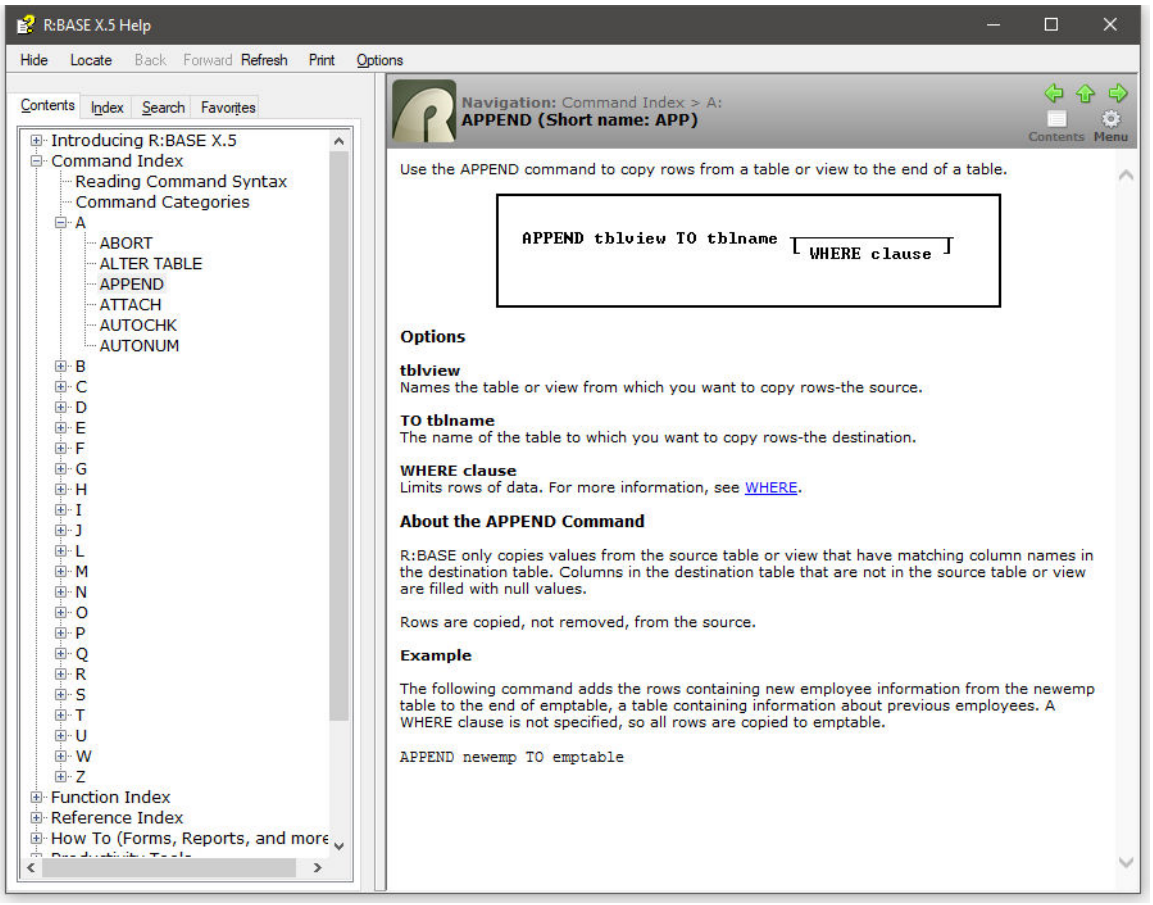
2. Close the Help

8.3 R:BASE Command Syntax

You can also enter the name of a R:BASE command after the HELP keyword, such as SELECT, or APPEND to open the help file for that specific command.

1. At the R> Prompt, enter "HELP APPEND" and press the [Enter] key.

R:BASE displays the help window, with the right pane showing the APPEND command. Notice that the screen not only contains an explanation of the command, but also contains a syntax diagram for the command. The screen looks like this:



Throughout the help documentation, commands and specific keywords used by each command are printed in upper case. Arguments, which are the conditions and requirements you include for a command, are printed in lower case. The general format of all commands is:

COMMAND KEYWORD argument **KEYWORD** argument

R:BASE syntax (the structure of a command) is used in the documentation and on R:BASE help screens. Read the syntax from left to right with no breaks in the flow of the main line of the syntax. Optional portions of a command are offset below the command line and, if used, must be in the same relative position in the command.

In the syntax diagram, the word APPEND, on the main line, is required. The following words *tblview* **TO** *tblname* are also required. The clause **WHERE clause** (a list of conditions) is optional and, if omitted, does not break the "left to right" flow of the syntax main line.

The keyword-argument combinations vary with each application of this command. In the syntax shown above, the command is APPEND, the keywords are **TO** and **WHERE**, and the arguments are *tblview*, *tblname*, and **clause**. The table names, column names, and conditions that you provide must be of the type shown in the syntax. For example, *tblname* indicates entry of a table name. The R:BASE commands and keywords do not change.

2. Close the Help

8.4 Using R:BASE Commands at the R> Prompt

First, you need to open a database in order to use the R:BASE commands in this section. If you are not connected to the database, do so now.

1. At the R> Prompt, enter "**CONNECT sky**"

R:BASE displays "Database Exists" and then returns the R> Prompt.

R:BASE is now connected to the database.

To find the correct syntax for a command, you can look it up in the Command Index, or use the R> to launch the help as described earlier, by typing "HELP" followed by the command name.

If an incorrect spelling for a command is entered, the error message "Unrecognized command - retype it" is displayed. If any invalid parameters or invalid sequence is entered, the error message "Syntax is incorrect for the command" is displayed.

2. At the R> Prompt, enter **SELECT** and press [ENTER].

R:BASE displays an error message.

Commands that operate with a single word, such as **CLS** or **EXIT**, obviously will not generate the error message using this method, but all commands requiring more than one word will.

If you know the correct syntax, you simply type the command and its parameters at the R> Prompt.

3. Enter **SELECT ALL FROM Customer**

R:BASE displays as many columns from the table Customer as will fit on the screen, and will prompt you to continue if more output follows.

R:BASE remembers the last command line you entered so that you can easily repeat the last command if you want. You can also add to or edit the command once it is redisplayed.

4. Press the [Page Up] key.

R:BASE displays **SELECT ALL FROM Customers** and the cursor is positioned at the end of the line. To edit the command, you use the left-arrow key to move the cursor to where you want to edit. You use the [Insert] key to add characters or use the [Delete] key to delete characters, or you can overwrite existing characters. The cursor will appear differently based on whether the Insert or Overwrite is specified for the keyboard. When Insert is in use, the cursor will blink as a block. When Overwrite is in use, the cursor will blink as an underscore.

5. Press the left-arrow key until the cursor is positioned on the "A" of the word **ALL**.
6. Press the [Delete] key three times to delete the word **ALL**.
7. Make sure the cursor is a block character by pressing the [Insert] key.
8. Enter **Company**
9. Press the [End] key to go to the end of the command line and then press [ENTER].

R:BASE displays only the column Company from the table Customers.

Commands that display data from a database can use the **WHERE** and **ORDER BY** clauses. The **WHERE** clause sets conditions by which to display the information, and **ORDER BY** indicates the order to display it.

10. Press the [Page Up] key to retrieve the command line.

Enter a space, then **WHERE Company LIKE 'C%' ORDER BY Company DESC** followed by the [Enter] key.

The **WHERE** clause restricts the rows displayed to those companies whose names begin with C. The % following the C is a wildcard character which means any next characters. This character is the same as

the * wildcard used in DOS operating system commands. The **ORDER BY** clause sorts the data descending by the company name.

After entering each of the commands, note that the Command History panel stores all of the commands entered. To reenter any command listed in the Command History panel, double click on it. To display the command at the Input Console, select it.

8.5 Using Variables and Expressions

Variables have many uses at the R> Prompt as well as all other modules of R:BASE. You can define variables or use expressions to calculate subtotals and totals, manipulate text values, draw data from tables, perform complex mathematical operations, return financial data such as interest rates, or perform many other functions.

A variable is an item that holds data. It is similar to a column, except that a variable is not connected to any particular table in a database. You can define a variable at the R> Prompt and then use the contents of the variable with more than one database. Variables, then, are global within R:BASE.

An expression is the computation that gives a variable its value. Expressions can also provide the value of a column as in computed columns, or they can be used to calculate values on the fly with the **SELECT** command.

1. Enter **SET VARIABLE v1 = 100**

   This defines a variable named v1 to a value of 100. Since you did not specify a data type, R:BASE automatically assigns an INTEGER data type because 100 is a whole number without a decimal point.

2. Enter **SHOW VARIABLE**

   R:BASE displays the list of variables, both the system variables (#DATE, #TIME, #PI, SQLCODE, SQLSTATE, and #NOW) and the variables you define. The display looks similar to this:

```
Variable           = Value                              Type
------------------   -------------------------------    -------
#DATE              = 02/19/2019                         DATE
#TIME              = 11:03:24                           TIME
#PI                = 3.14159265358979                   DOUBLE
SQLCODE            = 0                                  INTEGER
SQLSTATE           = 00000                              TEXT
#NOW               = 02/19/2019 11:03:24                DATETIME
v1                 = 100                                INTEGER
RBTI_REPORT_NAME   = custlabelc                         TEXT
```

The RBTI_LABEL_NAME variable is a system variable that was created when R:BASE opened and used the *custlabelc* label, which the value of the variable. If you have restarted R:BASE since starting this chapter, you will not see this variable.

You can assign the data type of a variable before or after any value is placed in it as long as the value and the data type are compatible.

3. Enter **SET VARIABLE v1 REAL**

   v1 already has the INTEGER value 100, but the data type can be changed to REAL.

4. Enter **SHOW VARIABLE v1**

On the screen, R:BASE displays "100." When you include the variable name on the line with SHOW VARIABLE, only the specified variable value is displayed. R:BASE does not display the heading or data types that are shown with the SHOW VARIABLE command when it is used without a variable name.

You can abbreviate most commands in R:BASE to the first three characters. If the command has more than one keyword, then each keyword can be abbreviated. Some secondary keywords can be abbreviated to a single character. For example, SHOW VARIABLE can be abbreviated to SHO VAR or SHO V.

You can combine text values together in a variable.

5. Enter **SET VAR text1 = 'Now is the time'**
6. Enter **SET VAR text2 = 'for all good men'**
7. Enter **SHOW VAR**

   As you can see, text1 and text2 are assigned TEXT data types.

9. Enter **SET VAR text3 = (.text1 & .text2)**

This combines the two text variables text1 and text2 together into one variable, text3. When you want to use the value of a variable, you precede the variable name with a dot, or period. This tells R:BASE to use the value of the variable, not just the name, as a text string. The ampersand "&" combines text strings and places a space between them. A plus "+" combines text strings without a space.

10. Enter **SHOW VAR text3**

R:BASE displays "**Now is the time for all good men**" on the screen. Because the default display width of a TEXT variable is 30 characters, the last word, men, wraps to a second line.

You can specify the width to display variables.

11. Enter **SHOW VAR text3 = 35**

Notice that the last word does not wrap because you allowed enough space for the full line by adding = 35 following the variable name.

You can do arithmetic for numeric variables as well.

12. Enter **SET VAR v1 = (.v1 + 25)**
13. Enter **SHOW VAR v1**

R:BASE added 25 to the value of the variable *v1* to get 125.

R:BASE has a large number Functions included. Functions perform a series of set calculations or convert values into other forms.

14. Enter **SET VAR v2 = (SQRT(.v1))**

Be sure that you always enclose in parentheses any expression containing a function.

15. Enter **SHOW VAR v2**

The **SQRT** function has calculated the square root of the value of *v1* and the **SET VARIABLE** command has placed that value into variable *v2*. The result is a DOUBLE (double-precision real) number.

You can combine many items and functions in a single expression.

16. Enter **SET VAR v3 = (LOG10(.v1) * SQRT(.v1) / EXP(.v2) * 15)**

Use parentheses around expressions to make sure that R:BASE recognizes an expression as an expression rather than a text string. R:BASE calculates expressions more quickly if it does not have to first decide whether or not an expression is to be calculated or simply be treated as a text string.

17. If you want, look at variable v3.

    R:BASE displays the DOUBLE data type number 0.004904177044216

You might wonder how you might use these variables once they are defined. Variables have many uses including:

- Calculating the value for a column from other column values, as with computed columns
- Providing a value for comparison to other values to determine if another operation should be performed, as in programs
- Collecting data from a table to use in a calculation
- Passing data between tables or databases
- Temporarily storing totals for the current session

You do not need to keep the variables you created in this lesson, so remove them now. Global variables may be deleted individually or all together. To delete variables, you use the CLEAR command. CLEAR completely removes the variables, not just the value of the variable.

18. Enter **CLEAR ALL VARIABLES** (or the abbreviated form CLE ALL VAR).

If you enter SHOW VARIABLE again, you will see that R:BASE has removed all the global variables you entered. Only the system variables remain.

If you want, you can exit from R:BASE at this time.

19. At the R> Prompt, enter **EXIT**

You are then returned to the operating system.

8.6 Summary

In this lesson you have:

- been introduced to the R>
- opened and accessed the help
- been introduced to the command syntax
- used commands at the R>
- created and calculated variables at the R>

To check your knowledge, try answering the following questions:

1. What hot keys are used to open the R>?
2. What hot key launches the help?
3. Give one example of a single word command.
4. What command is used to add conditions to restrict the data returned?
5. Give one example of a system variable.
6. Give one reason why you would use a variable.

Part IX

9 Lesson 8 - Writing Command Files

In this lesson, you learn how to use the R:BASE Editor (RBEDIT) to enter command lines into a command file and how to create and test a command file. There are no prerequisite lessons for this lesson; however, familiarity with using R:BASE commands at the R> Prompt (Lesson 7) is recommended.

While you can use R:BASE interactively, by typing commands at the R> Prompt, for many of the tasks you want to perform, it is useful to store the series of commands that perform a task in a permanent file that you can then execute whenever needed.

A stored series of commands is called a command file. R:BASE provides all of the tools you need to produce command files that, in turn, can be combined with other command files to form applications.

This lesson describes:

- How to use the R:BASE Editor, or RBEDIT, to enter and edit command files
- How to design and run a command file to perform a specific task

9.1 Using The R:BASE Editor (RBEDIT)

If you have not already done so, start R:BASE.

1. On the main R:BASE toolbar, select the "R> Prompt" button.

You are now at the R> Prompt. You need to open the SKY sample database, created in this tutorial, for use in this lesson.

2. At the R> Prompt, enter **`CONNECT sky`**

R:BASE displays "Database Exists" and then returns the R> Prompt.

3. At the R> Prompt, enter **`RBEDIT`**

The R:BASE Editor window will be displayed with a blank screen and a blinking cursor waiting for your commands.

The lines are numbered, starting at one incrementing down the page. New commands are added by typing in each line. Press [ENTER] at the end of each line to move down to the next line. The current position of the cursor is displayed in the lower left corner of the screen. For example, on line 2, column 1, the position is displayed as [2, 1]. Editing is simple with **RBEDIT** since it uses many of the same keys you use editing your form and report definitions.

There are many features in R:BASE Editor that allow you to enhance your command file writing experience. To review the Help for the R:BASE Editor, press the [Shift]+[F1] hot keys. After your review, you may close the Help.

Now, you are ready to enter your first command file.

9.2 Designing And Writing A Command File

Command files generally perform the same tasks you do by typing in commands at the R> Prompt. The major difference is that a command file saves the sequence of commands so that they can be repeated by you or someone else, such as a data-entry operator. For example, suppose you want to provide an option for the operator to check the total transaction amount for the current date.

In the following steps, you will learn how to enter the command file using the R:BASE Editor, and how to execute the command file outside of an application structure.

1. On line 1, within the RBEDIT screen, enter the **SELECT** command like this:

```
SELECT SUM(totalcharge) FROM flights WHERE flightdate = 08/12/2008
```

You need not enter upper and lower case as shown in the examples.

This command calculates the sum of the total charges column (totalcharge) for flights performed whose date (flightdate) is 08/12/2008. The command compares the value of flightdate to the date value 08/12/2008 and, if they match, includes the value in the sum for the column totalcharge.

2. Now save the file. To do so, you have several options. You can choose **File: Save** from the main Menu Bar, click on the blue diskette icon on the R:BASE Editor toolbar, or use the hot keys [Ctrl] +[S].

RBEDIT asks for a name for the new file within a "Save As" dialog window.

3. Within the "File name:" field, type **MyProg.rmd**
4. Within the "Save as type:" field, select the "RMD Files (*.rmd)" option.
5. Click on the "Save" button.

Next, you are going to switch from the "R:BASE Editor" window to the "R> Prompt" window. By now you may have seen the button for each opened module across the bottom of the R:BASE program. Right now, you may see buttons for the "Database Explorer, the "R:BASE Editor", and the "R> Prompt" window.

6. Choose the "R> Prompt" button to move the focus to the "R> Prompt" window.

Now you can try out your program using the RUN command.

7. At the R> Prompt, enter: **RUN MyProg.rmd**

The command file displays the total of the transactions entered on 08/12/2008 - $2,650.00.

What can be done to enhance this program to make it more useful and meaningful for anyone using the application? First, consider whether you want to execute this program from a menu. If so, you may want to display a message to the operator along with the total. In this case, you can enhance the **SELECT** command to place the total in a variable.

8. To edit the program again, choose the "R:BASE Editor" button to move the focus to the "R:BASE Editor" window
9. Position the cursor on the F in the word FROM. Enter the words "**INTO vSum**" and add a space.

The command file now looks like this:

```
SELECT SUM(totalcharge) INTO vSum FROM flights WHERE flightdate = 08/12/2008
```

This creates a variable vSum to hold the sum computed by the command.

10. Press the [End] key to place the cursor at the end of the line, then [Enter] to move the cursor to the next line.

11. For the second and third lines, enter:

```
SET VAR vMessage TEXT = ('The total for 08/12/2008 is: ' + (CTXT(.vSum)))
PAUSE 2 USING .vMessage CAPTION 'Sum' ICON INFO
```

When R:BASE executes the command file, these lines display a message box with the text "The total for 08/12/2008 is: $2,650.00", with the word "Sum" in the dialog caption, and an "Info" icon. The default text for the button will be"OK", but even that can be changed.

The **SET VAR** command creates a TEXT variable vMessage that combines two portions of the displayed message. The first portion defines the actual text "The total for 08/12/2008 is: ". The second portion contains the vSum variable within the **CTXT** Function. From the previous lesson, Functions were described to be used within R:BASE to perform a series of set calculations or convert values into other

forms. The **CTXT** is an example of a Function that converts a value into another form. The **CTXT** Function takes the .vSum variable, which is defined as a CURRENCY data type from the totalcharge column, and converts it to a TEXT data type within the **SET VAR** command. The vSum variable needed to be converted to TEXT as you cannot combine values with TEXT and CURRENCY data types. The data types must match, or must at least be compatible (e.g. DOUBLE and INTEGER).

Since this command file is to be executed as a menu option, The **PAUSE 2** command will give the operator some time to look at the total before returning to the menu display, until a key is pressed.

12. Finally, you want to return to the menu. Enter this line as line 4 in the command file:

    RETURN

The entire command file looks like this:

```
SELECT SUM(totalcharge) INTO vSum FROM flights WHERE flightdate = 08/12/2008
SET VAR vMessage TEXT = ('The total for 08/12/2008 is ' + (CTXT(.vSum)))
PAUSE 2 USING .vMessage CAPTION 'Sum' ICON INFO
RETURN
```

13. Now save the file. To do so, you have several options. You can choose **File: Save** from the main Menu Bar, click on the blue diskette icon on the R:BASE Editor toolbar, or use the hot keys [Ctrl] +[S].

14. Close the R:BASE Editor window by choosing **File: Close File** from the main Menu Bar, clicking on the "Close File" icon on the toolbar, or by pressing the lower "x" on the top right corner. If you use hot keys, you can press [Ctrl]+[F4].

9.3 Testing The Command File

You now want to test the command file.

1. Choose the "R> Prompt" button to move the focus to the "R> Prompt" window.

2. At the R> Prompt, enter **RUN MyProg.rmd**

If you entered everything accurately, the message should display that the total for 08/12/2018 is $2,650.00.

If the message does display the same results or if you receive an R:BASE message, return to RBEDIT and check to be sure that MyProg.rmd is correct.

3. Press any key to return to the R> Prompt.

9.4 Enhancing The Command File

Suppose you want to allow the operator to enter the date for the total, or you want to provide for a weekly or monthly total by allowing the operator to enter a beginning and ending date. You can change the MyProg.rmd file to allow this.

1. At the R> Prompt, enter **RBEDIT MyProg.rmd**

2. Edit the first part of the command file to match the syntax below.

   The text in the **blue** font are new commands to add to the file
   The text in the **green** font are requires changes to the existing command
   The text in the **maroon** font we will keep unchanged

You'll need to move the cursor to the first line and press [Enter] 17 times to open sufficient space at the top of the command file for the new lines. Use the space bar, not [Tab] to indent lines. You can even copy and paste the below command syntax into your file.

```
SET VARIABLE vDay1 TEXT = NULL
SET VARIABLE vDay2 TEXT = NULL

DIALOG 'Enter Beginning Date: ' vDay1 vKey1 1 +
  CAPTION 'Question' ICON QUESTION
IF vDay1 IS NULL THEN
  SET VARIABLE vDay1 DATE = .#DATE
ENDIF

DIALOG 'Enter Ending Date: ' vDay2 vKey2 1 +
  CAPTION 'Question' ICON QUESTION
IF vDay2 IS NULL THEN
  SET VARIABLE vDay2 DATE = .#DATE
ENDIF

SET VARIABLE vDay1 DATE
SET VARIABLE vDay2 DATE

SELECT SUM(totalcharge) INTO vSum FROM flights +
WHERE flightdate >= .vDay1 AND flightdate <= .vDay2

SET VAR vMessage TEXT = +
('The total from' & (CTXT(.vDay1)) & 'to' & +
(CTXT(.vDay2)) & 'is:' & (CTXT(.vSum)))

PAUSE 2 USING .vMessage CAPTION 'Sum' ICON INFO
RETURN
```

Some of the commands shown above are indented by a few spaces. This is not necessary but makes the command file easier to read. Blank lines may be used for the same purpose. Notice that the indented commands allow you to easily see where programming structures such as the commands inside an **IF**...**ENDIF** structure begin and end. Also, R:BASE allows you to use upper or lower case or a mixture as shown in the listing. Upper case is used in the manuals to show keywords such as commands or command clauses, and lower case is used to show variable information such as column or variable names.

The first two **SET VARIABLE** commands set the data types of two variables to the TEXT data type so they are accepted by the **DIALOG** command, which only uses TEXT variables.

The **DIALOG** commands on lines 4 and 10 accept the beginning and ending dates and place them in variables vDay1 and vDay2. The command is too long to enter on a single line, and so it is continued by entering a space and then a plus sign as the last character of the first line. As is the case when you enter a command at the R> Prompt, when R:BASE sees a plus sign at the end of a line, whether it is a command or data, it expects the next line to continue the previous one.

As a date value is entered, the **IF** structure checks to see if a value was actually entered. If no date is entered, the variable has a null value, so the command file sets the variable to a DATE data type and to the current date (.#DATE). It does this by checking the date variables for **IS NULL**. I**S NULL** is the keyword used to check for null entries.

After the **DIALOG** commands, the two **SET VARIABLE** commands set the data types of the two variables to the DATE data type in the event that the operator entered valid date values (i.e 08/12/2008). Since the resulting value of the **DIALOG** variables are always of the TEXT data type, the resulting variables must be converted to the correct data type if they are to be compared against a data type other than

TEXT. The variables vDay1 and vDay2 must be forced to DATE data types so the variable values can be correctly compared against the flightdate column values.

The **SELECT** command now includes a WHERE clause that limits the computation to the rows with a flight date greater than or equal to the beginning date and less than or equal to the ending date.

The **SET VAR** command which built the vMessage variable has now been enhanced to include the vDay1 and vDay2 variables with the value of vSum, which is, of course, the computed total for the transactions.

The **PAUSE 2** command displays the value of the dates and the computed total for the transactions.

When the command file is executed, the vDay1 and vDay2 date values change according to what the operator enters. The sum changes depending on the dates entered.

3. Now save the file.

4. Choose the "R> Prompt" button to move the focus to the "R> Prompt" window.

5. At the R> Prompt, enter **RUN MyProg.rmd**

   The command file should execute prompting you for dates.

6. Enter the beginning date "04/15/2008", and click the "OK" button.

7. Enter the ending date "07/01/2008", and click the "OK" button.

   The message should display that the total from 04/15/2018 to 07/01/2008 is $12,450.00.

This process may seem a little complicated. All that is happening is that you are using the concatenation operator **&** to combine the values into a single variable and then display that single variable with a **PAUSE** command.

The data types of vDay1, vDay2, and vSum variables change to TEXT so they can be concatenated with other text. The SET VARIABLE command combines all the text values. The **&** operator is used to insert a space between each text value.

Notice that when you want R:BASE to use the value of a variable, you put a dot in front of the variable name. For example, in the SET VARIABLE command that performs the concatenation, the variables vDay1 and vDay2 have dots because you want to use the value of these variables, not just the variable names. The basic rule is that if the variable name is to the right of a comparison operator such as an equal sign, then the variable needs to be dotted. If the variable is to the left of a comparison operator, it should not be dotted.

Your next enhancement is to add comments to the file. A command file should always be documented in case another user needs to read what it does.

8. Enter the following command at Line 1, and move the SET VAR commands down

   ```
   -- Calculates flight totals for a given flight date range
   ```

Notice that R:BASE recognizes the hyphens "--" at the beginning of the line and adds a pink italicized highlighting. This means that the text will not be read by R:BASE when you run the file. You could also place your name and date in the file.

Your final enhancement is make your command file clean up after itself. You do not want to collect unneeded variables, because each variable uses space in your computer's memory that might better be used by R:BASE for its own processing needs.

8. Enter the following command after the **PAUSE** command and before the **RETURN** command:

   ```
   CLEAR VAR vDay1, vDay2, vSum, vMessage, vKey1, vKey2
   ```

   This command deletes the variables you have used in this command file. You do not want to delete all the defined variables by using the **CLEAR ALL VARIABLES** form of this command, because there

may be some variables needed in another procedure file, that defines some variables it needs for its own operation.

The entire command file now looks like this:

```
-- Calculates flight totals for a given flight date range
-- Written by James Howlett 02/19/2019

SET VARIABLE vDay1 TEXT = NULL
SET VARIABLE vDay2 TEXT = NULL

DIALOG 'Enter Beginning Date: ' vDay1 vKey1 1 +
  CAPTION 'Question' ICON QUESTION
IF vDay1 IS NULL THEN
  SET VARIABLE vDay1 DATE = .#DATE
ENDIF

DIALOG 'Enter Ending Date: ' vDay2 vKey2 1 +
  CAPTION 'Question' ICON QUESTION
IF vDay2 IS NULL THEN
  SET VARIABLE vDay2 DATE = .#DATE
ENDIF

SET VARIABLE vDay1 DATE
SET VARIABLE vDay2 DATE

SELECT SUM(totalcharge) INTO vSum FROM flights +
WHERE flightdate >= .vDay1 AND flightdate <= .vDay2

SET VAR vMessage TEXT = +
('The total from' & (CTXT(.vDay1)) & 'to' & +
(CTXT(.vDay2)) & 'is:' & (CTXT(.vSum)))

PAUSE 2 USING .vMessage CAPTION 'Sum' ICON INFO

CLEAR VAR vDay1, vDay2, vSum, vMessage, vKey1, vKey2

RETURN
```

If your own command file does not look like this, use RBEDIT to edit it. When you finish, leave RBEDIT so that you can test the command file.

9. Save your file and leave the R:BASE Editor.

10. To execute the program, at the R> Prompt, enter **RUN MyProg.rmd**

The dates contained in the flights table range from 03/03/2008 to 09/07/2008, so enter any combination of these dates for beginning and end dates as long as the first date is earlier than the second date.

In the next lesson, you will learn how to create and modify an application by using the Application Designer.

If you want, you can exit from R:BASE at this time.

11. At the R> Prompt, enter **EXIT**

9.5 Summary

In this lesson you have:

- been introduced to the R:BASE Editor (RBEDIT)
- opened and accessed command files
- been introduced to new commands
- ran a command file at the R>
- concatenated text variables
- cleared variables

To check your knowledge, try answering the following questions:

1. What R> command launches the R:BASE Editor?
2. What hot keys launch the R:BASE Editor help?
3. Provide one method to save a command file.
4. What does the **IS NULL** keyword check for?
5. Why should you clear variables at the end of a command file?

Part X

10 Lesson 9 - Creating and Using an Application

An application handles data and procedures involved in performing tasks. An application is useful for procedures that are repetitive or are performed by many people. Applications are ideal solutions where efficiency, consistency, and ease of training are concerns.

In this lesson you'll design an application for the *sky* database and build it with the R:BASE Application Wizard. When you're finished, the application, named *skyapp,* will consist of menus with options for entering, editing, deleting, and printing data from all the tables in the database.

At the end of this lesson you'll be able to:

- Start the Application Designer
- Develop an application with three menu levels
- Create WHERE clauses for forms and reports
- View the application and startup files from R:BASE
- Run an application
- Modify existing rows in a table using a form

The application you'll create in this lesson will allow the employees at skyWrite, Ink to enter and access their employee, customer, and flights information. They'll use *skyapp* to keep current, to print reports, and to answer questions using data from the database. This application will be especially useful for repetitive tasks, such as adding new customer information or printing mailing labels.

10.1 Designing an Application

There are three steps in designing a database application:

- Identify the tasks the application will perform
- Organize the tasks into menus
- Decide which menu types to use

10.2 Identifying the Tasks Skyapp Will Perform

In the *sky* database there are three tables in which data needs to be maintained: *employee, customer,* and *flights.* Write the name of each table, and list the tasks needed to maintain them. Also, include any reports you want to print.

Tasks related to the employee table:
- Add new information about employees
- Modify or delete employee information by last name
- Print a list of employees

Tasks related to the customer table:
- Add new information about customers
- Modify or delete customers by last name
- Print mailing labels for the customer list; sometimes mailing labels need to be sorted by company, other times by zip code

Tasks related to the flights table:
- Add new flight information
- Modify or delete flight information by flight date
- Print all flights by flight date

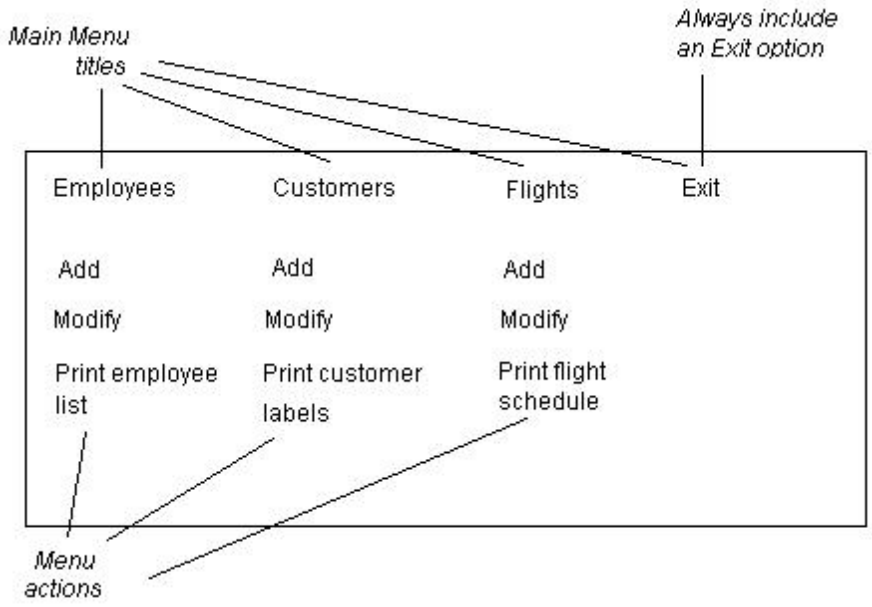
10.3 Organizing the Tasks Into Menus

Next consider how you can best organize these tasks into menus. You can organize your menus by any one, or a combination, of the ways identified here.

- Do the tasks have to be performed in a certain order? If so, organize them by function.
- Are some tasks performed more often than others? If so, organize them by their frequency of use.
- Are the tasks related to the structure of the database? If so, organize them by table.

In *skyapp,* you can group the tasks by table.

Now, using the table names as menu titles and the tasks as the options on each menu, turn your list of tasks into a menu tree. A menu tree shows the order and hierarchy of menus in your application.



10.4 Building an Application

After designing *skyapp* on paper, you're ready to build it. The finished *skyapp* menu will look like the following:

As you click each Main Menu name, the drop-down menu is displayed and the user selects a menu action to perform.

To begin the R:BASE Application Designer:

1. Begin at the Database Explorer. If the database is not connected, select "sky" and select the "Connect database" option.
2. Click the Applications option (it is near the bottom of the Group Bar), then click "New Application...".

R:BASE displays the dialog "Do you want to use the Application Wizard?"

3. Choose "Yes".

The first screen in the Application Designer prompts you for the caption for the application window and if you would like to add a wallpaper or a custom icon for the window. For now, we will only add the caption.

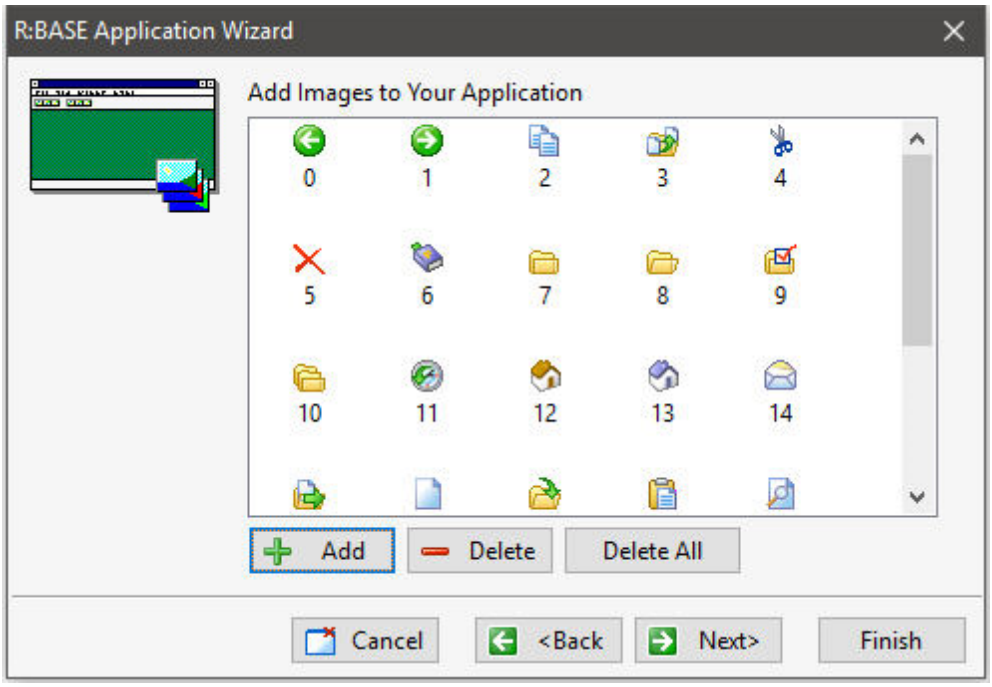
4. Enter "skyWrite Ink Application" in the "Application Caption:" field.
5. Select the "Next" button.

The Application Designer now prompts you to add images to your application to be included with your menu titles and menu options. These images must be .BMP files with the size 16 X 16 pixels. These images must be added one at a time. A collection of sample images are included with the R:BASE X.5 installation.

For R:BASE X.5 Enterprise, you will find the images here: C:\RBTI\RBGX5E\Samples\BMPS\16x16

For R:BASE X.5, you will find the images here: C:\RBTI\RBGX5\Samples\BMPS\16x16

6. To add an image, choose the "Add" button, navigate to the sample image directory, choose a bmp file, and click "Open". The image will be added.
7. Repeat Step 6 to as many additional images as you like. You can also use the operating system to copy a group of bitmaps from the above image folder to your current folder in order to add the images faster by not having to navigate to the image folder for every bitmap.



8. Once you are done adding images, select the "Next" button.

10.5 Creating Menu Actions to Enter Data

Now, you will actually begin creating the functional part of the application. The first step in this process is to define the actions that you want to associate with the menu options. An action tells R:BASE what to do when the menu option is chosen. The action is a command that performs a specific function. You can assign any number of actions to a menu option, but normally only one action is assigned to each menu option.

1. Select the "Add" button to add an action.

The "New Item" dialog will appear for you to enter information.

The "Caption:" field defaults to the action number. You can edit the text to display in the Caption field. This text is what you will see on the actual button or drop-down menu item.

2. Overwrite "Action1" with "Add Employee".

The "Hint:" field can be specified as well. If you hover over the object with your mouse cursor when the application is run, the hint will appear for a short period to give you a quick explanation of the action. **Note:** Not all objects support the Hint property in the runtime mode. Objects that support the Hint property are the Drop-Down Menu Buttons and the Toolbar Buttons.

3. Leave the "Hint:" field empty.

The "Action Name:" is a sequential number that is given to the action as a default when you create it. You can choose to leave this alone, or customize the Action Name to your specifications. One important key to remember is that the Action Name MUST be unique, and the system will verify this for integrity.

4. Do not alter the "Action Name:" field.

The "Image Index:" allows you to assign one of the 16X16 images, from the Image Index library, to the action you are defining. If you have added any images, you can assign them now.

The "List of Actions:" panel arranges your actions. Right now, no actions are defined, so the panel is empty.

Use the "New Action" drop-down box to select from a list of commonly used pre-defined actions. The following table lists the available actions.

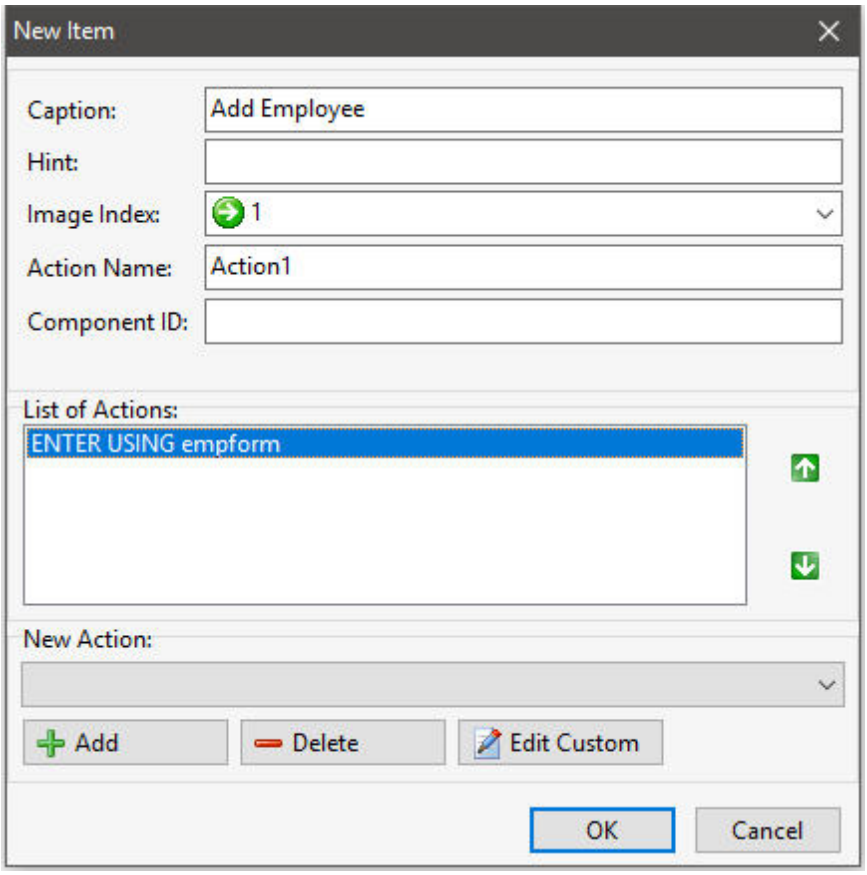
Action	**Purpose**
Enter data using a form	Adds data to a table using a predefined form
Edit data using a form	Edits data in a table using a predefined form
Delete rows from a table	Removes selected rows from a table
Browse and edit rows in a table	Uses the Data Browser to view and modify data
Update rows in a table	Changes selected rows of data without using a form
Print reports	Prints selected data using a report
Print labels	Prints selected data in a label format
Set user identifier	Enters or prompts for a user identifier
Exit	Exits from the application
Run A File	Runs a command file
Custom Action	Adds custom command lines to an application file

5. Choose the option "Enter data using a form".
6. Click the "Add" button.

A dialog appears for you to select the form.

7. Since we are creating the action for "Add Employee", choose the *empform* form.
8. Click the OK button.

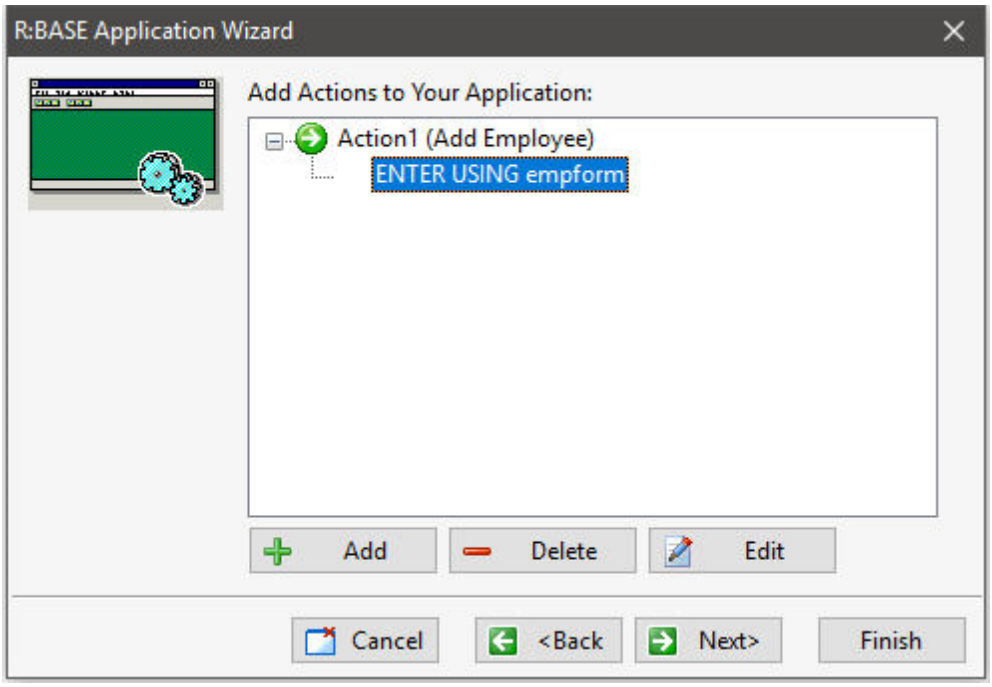
After selecting the form, you will return to the previous screen where you see the defined action and add additional actions, if necessary.



9. Click the OK button.

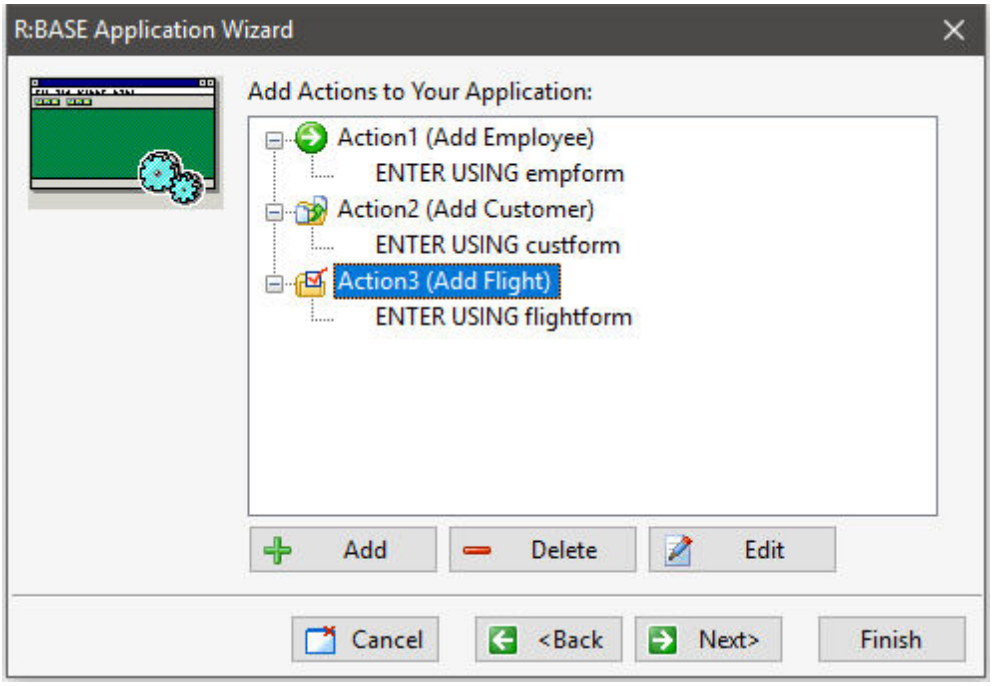
Action1 with the caption has been added to the Application Wizard screen.

By clicking on the plus (+), you can open the action to see what it does.



10. Now, add two more actions through in the Application Wizard for "Add Customer" and "Add Flight". These actions will use the forms for the *customer* and *flights* tables to enter data.

Your actions should be listed as displayed below. If you have not added images to your actions, you can do so at a later time.



10.6 Creating Menu Actions to Edit Data

Next, we will add the actions for modifying information in the tables. The Modify pull-down submenu options allow users to edit or delete employee, customer, or flight information. You want users to use a form to edit the data. In addition, you want users to be able to identify the row to edit, so you'll define a WHERE clause that allows them to select the row by entering the data specific to that row. For the action to modify an employee's record, we will use the employee's last name to pull the correct record to edit.

To define the action for editing data:

1. With the "Add Actions to Your Application:" screen, in the Application Wizard, still displayed, select the "Add" button.

R:BASE displays the "New Item" dialog.

2. Enter "Modify Employee" in the "Caption:" field.
3. Choose "Edit Data Using a Form" from the "New Action:" drop down box.
4. Click the "Add" button.

R:BASE displays the "Available Forms:" dialog.

5. Choose *empform* from the drop down box.

We want to build a condition based on the employee's last name. When the application option is chosen, we want to be prompted to enter a last name, and then have that row selected.

6. Click the "WHERE Builder" button.

The "Where Builder" dialog box is displayed. R:BASE knows that the form *empform* is based on the *employee* table. *Employee* is selected in "Table Name" and the columns in the *employee* table are listed in "Select Columns".

7. Select "emplastname" column in the "Select Columns" panel.
8. Click the "Add Column" button.

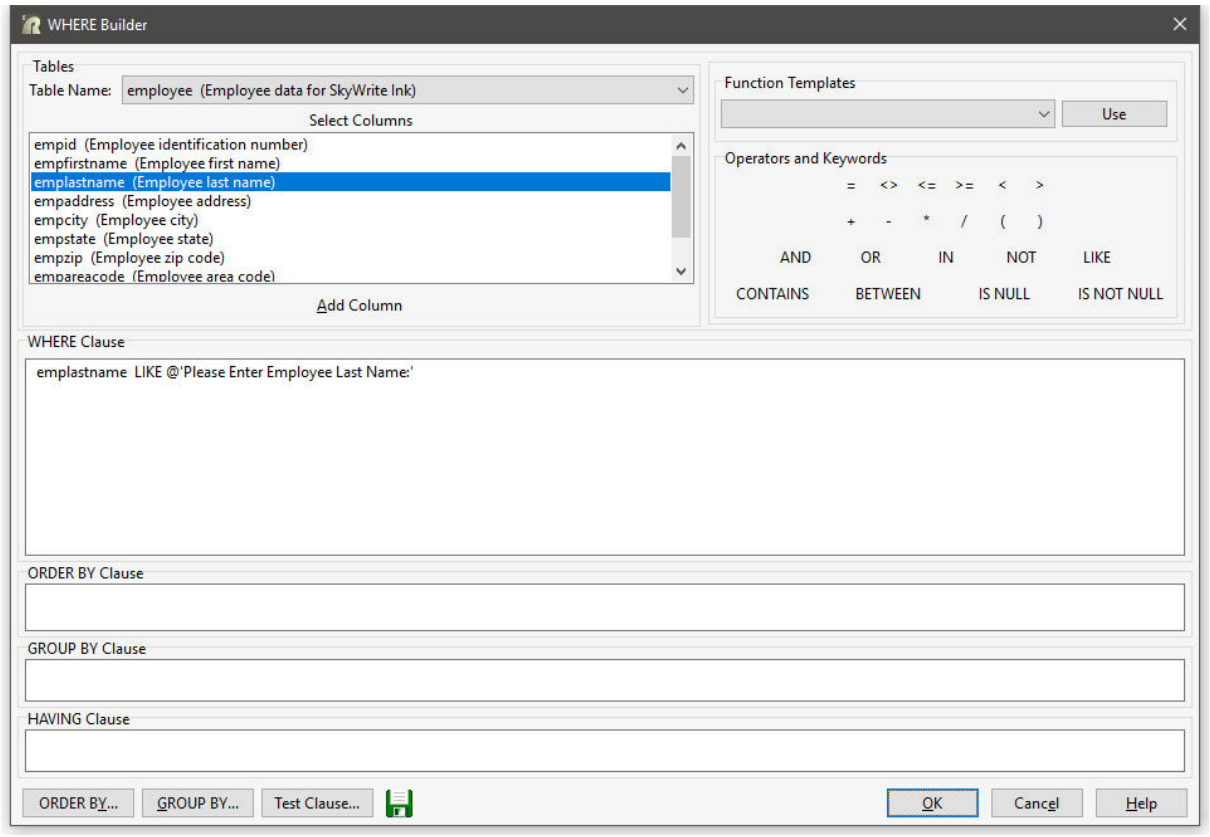
The column "emplastname" is displayed in "WHERE Clause" panel.

9. Click the "LIKE" button in the "Operators and Keywords" panel.

The LIKE operator is added to the "Where Clause" panel. By selecting the LIKE operator, we can retrieve data by entering a complete employee last name, or using a wildcard character, we can enter just part of a last name. The wildcards are useful for searching for more than one row when the first few characters are known.

Wildcard characters are used to represent one or more characters in a search. The default R:BASE wildcard characters are the underscore _ (one character) and the percent % (one or more characters). For example, the underscore can be added to the end of the last name search of "Mil" and would return valid values ranging from "Mila" to 'Milz". Using the percent the end of the last name search of "Mil" would return valid values that begin with "Mil" with no limit on the additional characters in the last name, like "Miller" or "Milbourne". It is more likely that the many wildcard character will be used.

10. At the end of "LIKE" enter "@'Please Enter Employee Last Name:'" in "WHERE Clause" panel.



The "@" indicates to R:BASE that a dialog message follows. The dialog message is displayed when the application option is run. Do not put a space between the "@" and the message.

11. Click the OK button.

The "Available Forms:" dialog is displayed with the WHERE clause you built in the WHERE Builder.

12. Click the OK button.

The "New Item" dialog box is displayed with the complete command to perform the action for the "Modify Employee" menu option. If you have not added an image to the action, you can do so.

13. Click the OK button.

A fourth action is now displayed in the list of actions for the Application Wizard.

Now, add two more actions in the Application Wizard for "Modify Customer" and "Modify Flight". Both actions will use the predefined action for editing data using a form, but the WHERE Clause for each search will be different. The search on the *customer* table will be based on the customer last name, while the WHERE Clause for the *flights* table will be based on the flight date.

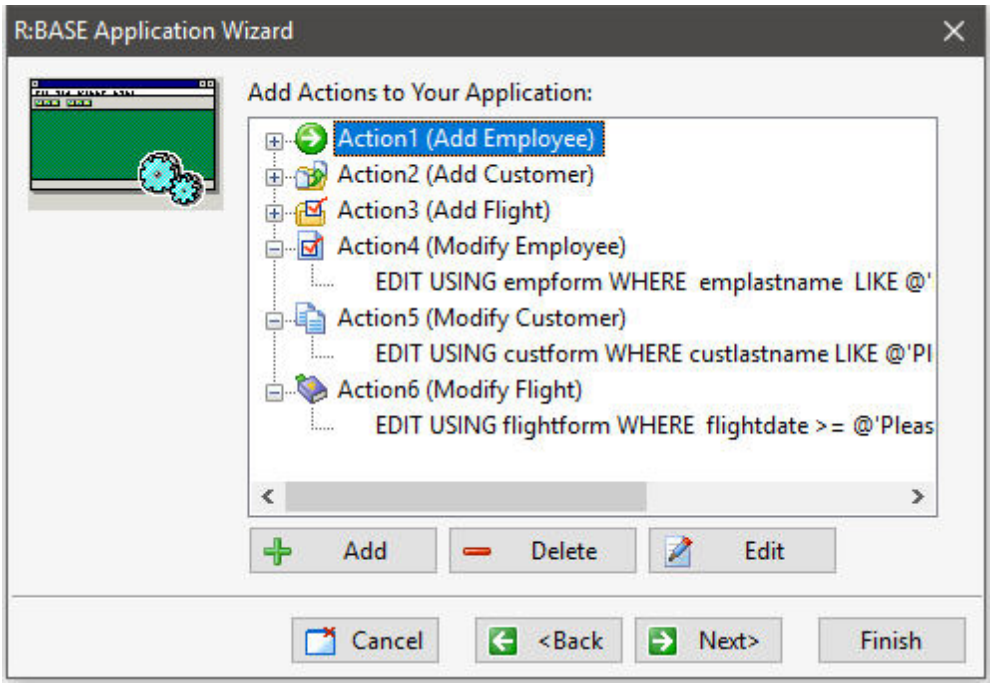
Using the following:

Modify Customer
- Enter "Modify Customer" for action "Caption:"
- Select the "Edit Data Using a Form" predefined action and the *customer* form.
- In the WHERE Builder, build the WHERE Clause "custlastname LIKE @'Please Enter Customer Last Name:'

Modify Flight
- Enter "Modify Flight" for action "Caption:".

- Select the "Edit Data Using a Form" predefined action and the *flights* form.
- In the WHERE Builder, build the WHERE Clause "flightdate >= @'Please Enter Flight Date:'.
- While in the WHERE Builder, use the "ORDER BY.." button to sort the column *flightdate* in ascending order.

There should now be six actions defined in the Application Wizard screen.



10.7 Creating Menu Actions to Print Data

You have now finished defining actions for adding and editing data for the Employees, Customers, and Flights menu. Next, we'll define the actions for printing the data in the tables. We will create the menu action option for printing the employee list report, which you created in Lesson 5, a menu action option for printing customer labels, which you created in Lesson 6, and a menu action option to print the flight schedule, also from Lesson 5.

To define the action for printing the employee list:

1. With the "Add Actions to Your Application:" screen, in the Application Wizard, still displayed, select the "Add" button.

R:BASE displays the "New Item" dialog.

2. Enter "Print Employee List" in the "Caption:" field.
3. Choose "Print Reports" from the drop down box.
4. Click the "Add" button.

R:BASE displays the "Available Reports:" dialog.

5. Choose *emplist* from the drop down box.

We will not define a WHERE clause for printing this report because we want to list all employees.

6. Click the OK button.

The "New Item" dialog box is displayed with the action "PRINT empllist" menu option.

7. Click the OK button.

The new action is now displayed in the list of actions in the Application Wizard.

We have now finished defining menu actions for all the menu options on the Employees main menu.

Next, we'll define the actions for the "Print Customer Labels" submenu. These actions will provide the choices to print the customer labels by company name and by zip code. First, we will define the actions for each submenu option.

To define the action for printing the customer labels:

1. With the "Add Actions to Your Application:" screen, in the Application Wizard, still displayed, select the "Add" button.

R:BASE displays the "New Item" dialog.

2. Enter "By Company" in the "Caption:" field.
3. Choose "Print Labels" from the drop down box.
4. Click the "Add" button.

R:BASE displays the "Available Labels:" dialog.

5. Choose *custlabelc* from the drop down box.

We want this action to print the customer labels by the company name.

6. Click the "WHERE Builder" button.

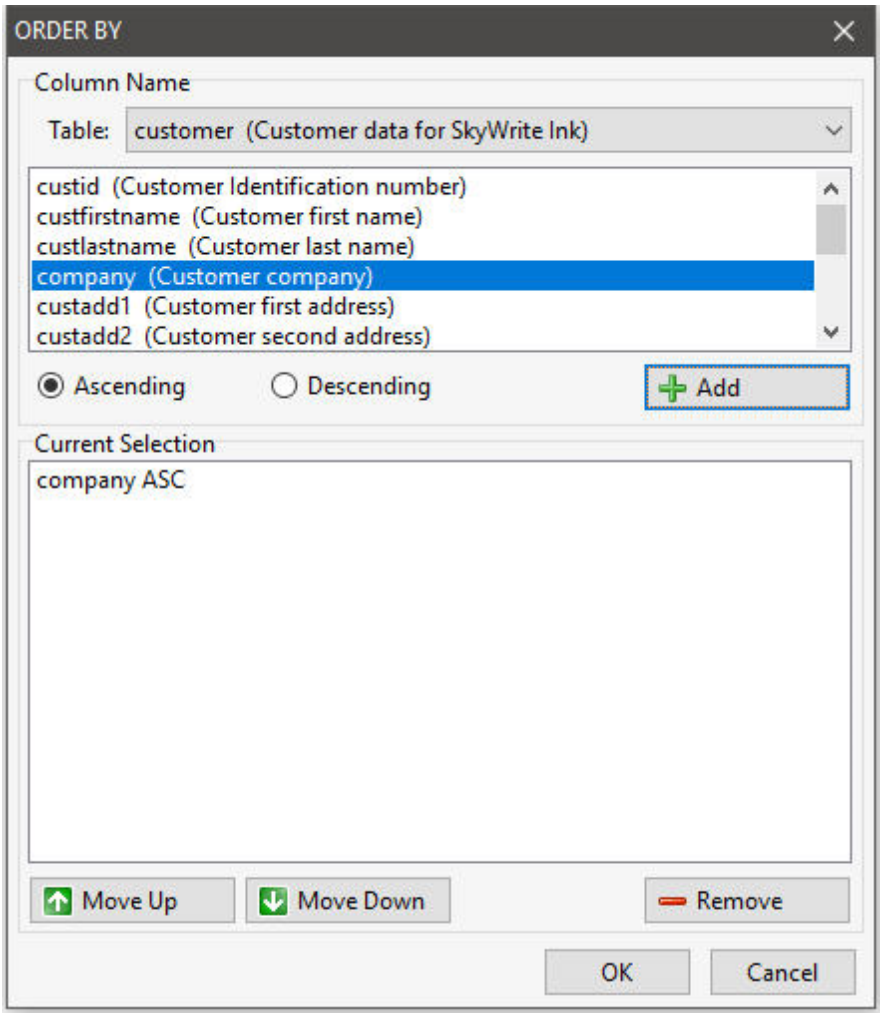
The "Where Builder" dialog box is displayed. R:BASE knows that the label *custlabelc* is based on the *customer* table. *Customer* is selected in "Table Name" and the columns in the *Customer* table are listed in "Select Columns".

7. Select the "ORDER BY..." button on the bottom left corner.

The "ORDER BY" dialog is displayed. This dialog allows you to distinguish which data appears first when it is displayed.

8. Choose the company column.

We want the company name to appear in ascending order. Make sure the radio button for "Ascending" is selected.



9. Click the Add button

The "Current Selection:" panel fills with the data "company ASC".

10. Click the OK button to close the "ORDER BY" dialog.
11. Click the OK button to close the "WHERE Builder" dialog.
12. Click the OK button to return to the "New Item" dialog.

The "New Item" dialog box is displayed with the complete command to perform the action for printing the customer label by company name in ascending order.

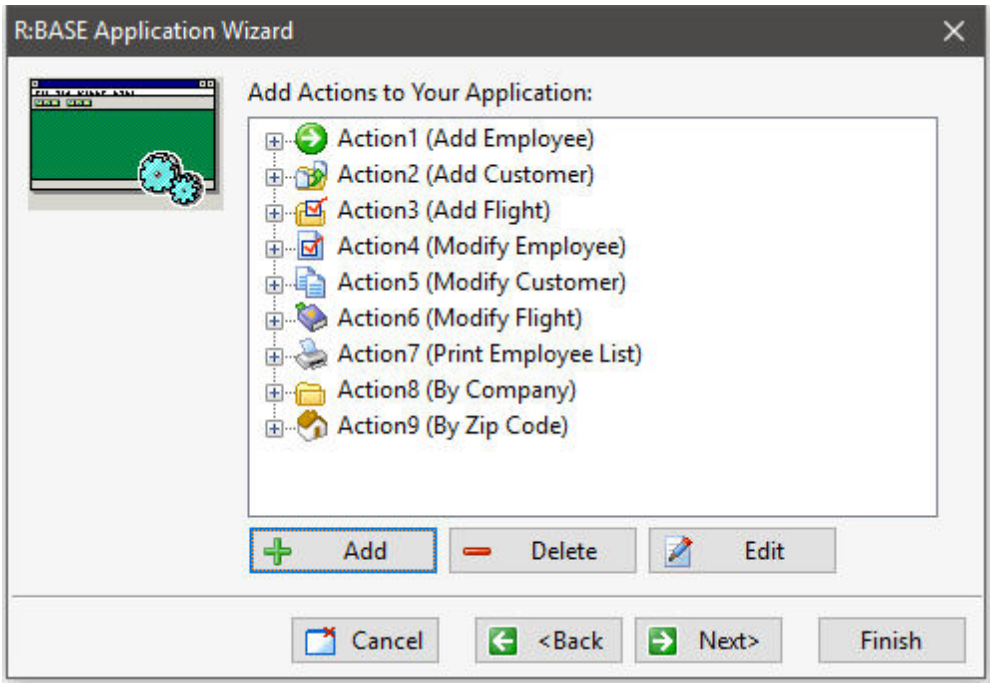
13. Click the OK button.

The new action is now displayed in the list of actions in the Application Wizard.

14. Repeat the above steps to create the next action to print customer labels by zip code using the following:

- Enter "By Zip Code" for action "Caption:"
- Select the "Print Labels" predefined action and the *custlabelc* label.
- In the ORDER BY dialog, select the *custzip* column to print in ascending order.

The new action should now be displayed in the list of actions in the Application Wizard. If you have not added images to your actions, you can do so at a later time.



10.8 Assigning Actions to the Flights and Exit Menus

We are almost complete defining the menu actions. There are three actions left to define.

- Print Flight Schedule
- Calculate Total for Date Range
- Exit Application

Now you're going to complete the Print Flight Schedule action on your own, the information you need to define the actions is provided. If you have any difficulty in completing the menus, review the previous procedures.

1. To define the action for the "Print Flight Schedule" menu action, select "Print Reports" as the predefined action and "flightlist" as the report.

2. Next, assign the action for "Calculate Total for Date Range" menu action.
3. Select "Run a File" as the predefined action, and select the "Add" button.

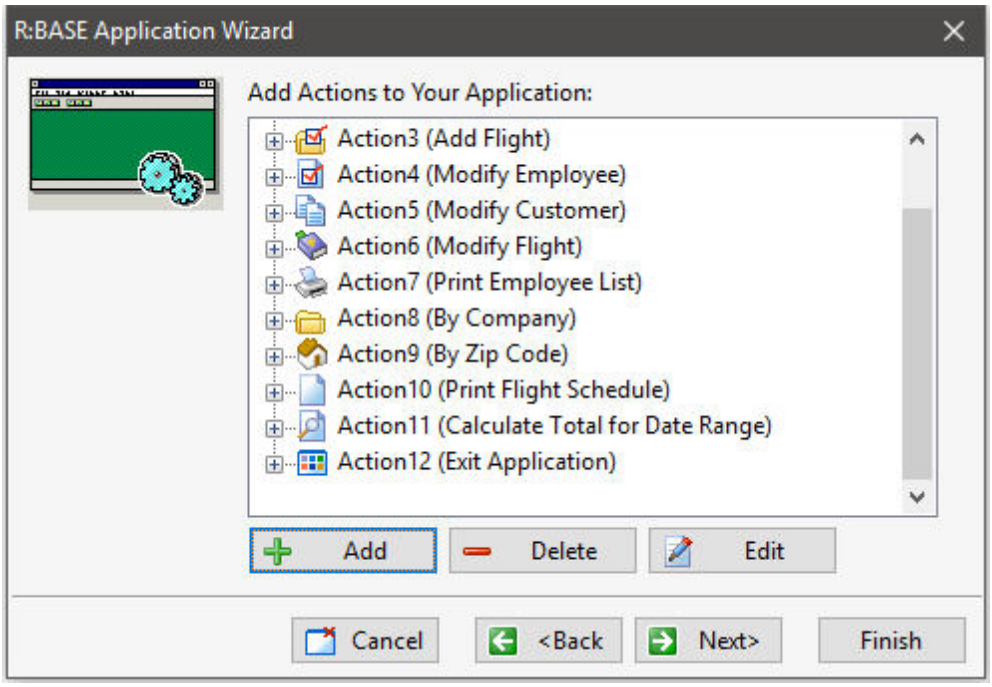
You will be prompted to select a file stored on your computer.

4. Select the "MyProg.rmd" file that you created in the previous lesson and select the "Open" button.
5. Click OK to save this action.

You have assigned actions to all the options on the Flights submenu.

6. Now, you'll define the last action, which is for the Exit main menu. Select "Exit" as the predefined action for the "Exit Application" menu action.

You should have 12 actions defined. You're now ready to name and define the main menu titles.



7. Click the "Next" button in the R:BASE Application Wizard screen.

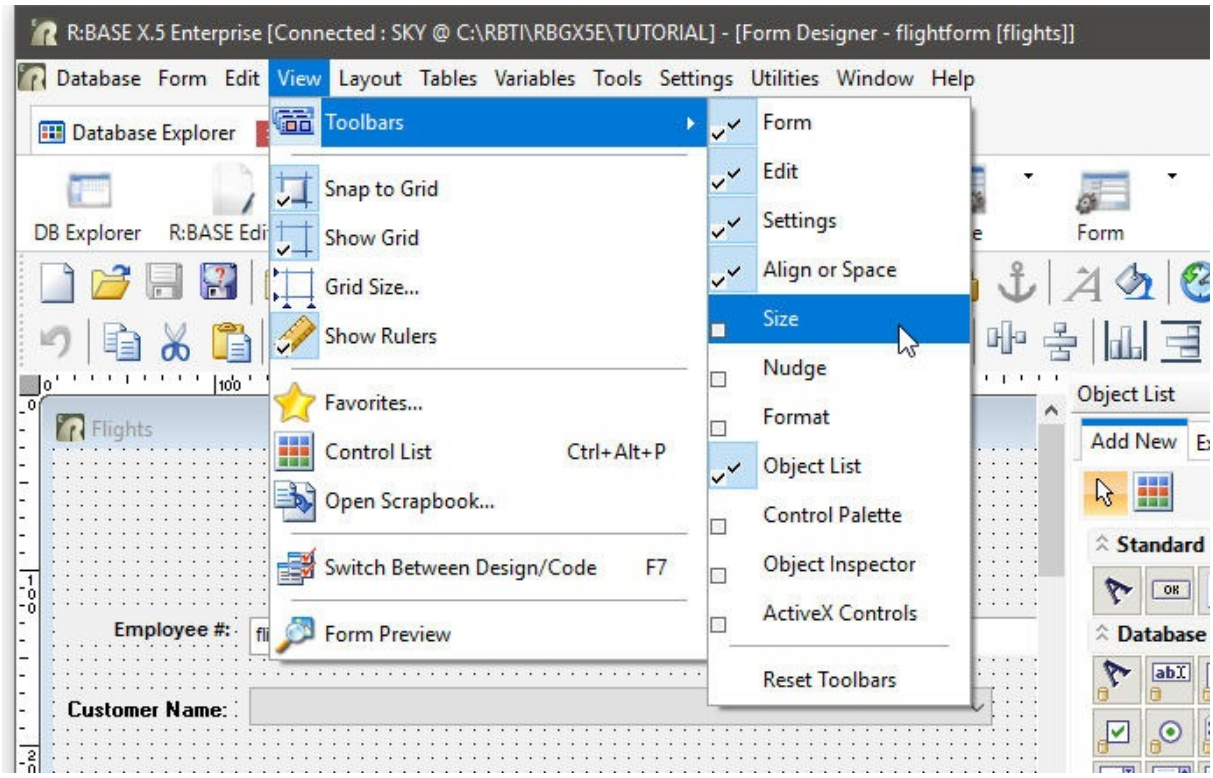
10.9 Defining the Main Menu

The R:BASE Application Wizard displays the "Creating Your Application Main Menu" window. Here, we will define the Main Menu titles and add the actions to each.

The R:BASE Application environment will create a window with a menu bar and pull-down menus similar to a menu bar and pull-down menus that you would see in any Windows based program.

A pull-down menu has three parts: a menu bar, pull-down menus, and submenus, which branch off of the drop-down menu. The name of the menu appears in the menu bar. Each menu has one or more options on the pull-down menu and submenu. Users choose an option to perform the programmed action.

The image below of the R:BASE Form Designer window is an examples of the pull-down menu parts. The Menu Bar option "View" is just like the Menu Bar option "Customers" on the skyWrite Ink application. The drop-down menu option "Toolbars" is just like the drop-down menu option "Print Customer Labels". The submenu option, for all the different toolbars, is the submenu options, "By Company", "By Zip Code".



For a menu, you first name it, and then add the actions for the pull-down menu and submenu. We will begin with the "Employees" Main Menu.

10.10 Defining the Employees Main Menus

The Employees menu will have actions for adding, modifying, and printing employee information.

To define the Employees Main Menu:

1. In the R:BASE Application Wizard "Creating Your Application Main Menu" window, select the "Add Item to Root" button.

The "Menu Item Editor" dialog appears.

2. Enter "&Employees" in the "Menu Item Caption:" field.

The ampersand (&) is placed in front of the hot key letter. When the application is run, the hot key for "Employees" is the "E". Based on your computer's operating system, the hot key letter appears underlined. When you choose the [Alt] key plus the underlined letter, the drop-down menu opens down. For example, [Alt]+[E] will open the "Employees" drop-down menu.

3. Add an image to the menu, then click the OK button. If no images were added earlier, skip this step.

"&Employees" now appears in the window.

4. Highlight "&Employees" and select the "Add Item to Selected" button.

The "Menu Item Editor" dialog appears. This time, we will use the "Action List" panel.

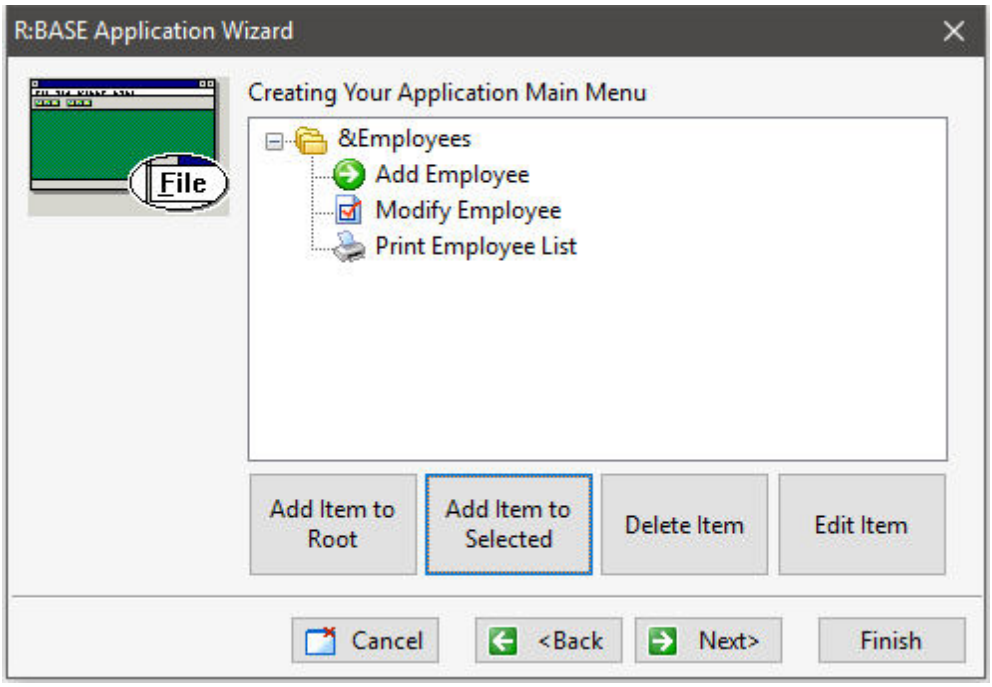
5. Select "Add Employee" from the list of actions, which is the first menu action to appear in the "Employee" drop-down menu.
6. Click the OK button.

7. Highlight "&Employees" and select the "Add Item to Selected" button.
8. Select "Modify Employee" from the list of actions, which is the second menu action to appear in the "Employee" drop-down menu
9. Click the OK button.

You can see now how a tree structure will be displayed for your application menus.

10. Highlight "&Employees" and select the "Add Item to Selected" button.
11. Select "Print Employee List", the third drop-down menu action.
12. Click the OK button.

The drop-down menu for the menu "Employees" is complete.



Defining the Customers Menu

The "Customers" menu will have options for adding, changing, and printing customer information.

To define the Customers Menu:

1. In the R:BASE Application Wizard "Creating Your Application Main Menu" window, select the "Add Item to Root" button.

The "Menu Item Editor" dialog appears.

2. Enter "&Customers" in the "Menu Item Caption:" field.
3. Add an image to the menu, then click the OK button.

"&Customers" now appears in the window within its own branch.

4. Highlight "&Customers" and select the "Add Item to Selected" button.

The "Menu Item Editor" dialog appears. This time, we will use the list of actions on the left panel.

5. Select "Add Customer" from the list of actions, which is the first menu action to appear in the "Customer" drop-down menu.
6. Click the OK button.
7. Highlight "&Customer" and select the "Add Item to Selected" button.

8. Select "Modify Customer" from the list of actions, which is the second menu action to appear in the "Customer" drop-down menu
9. Click the OK button.

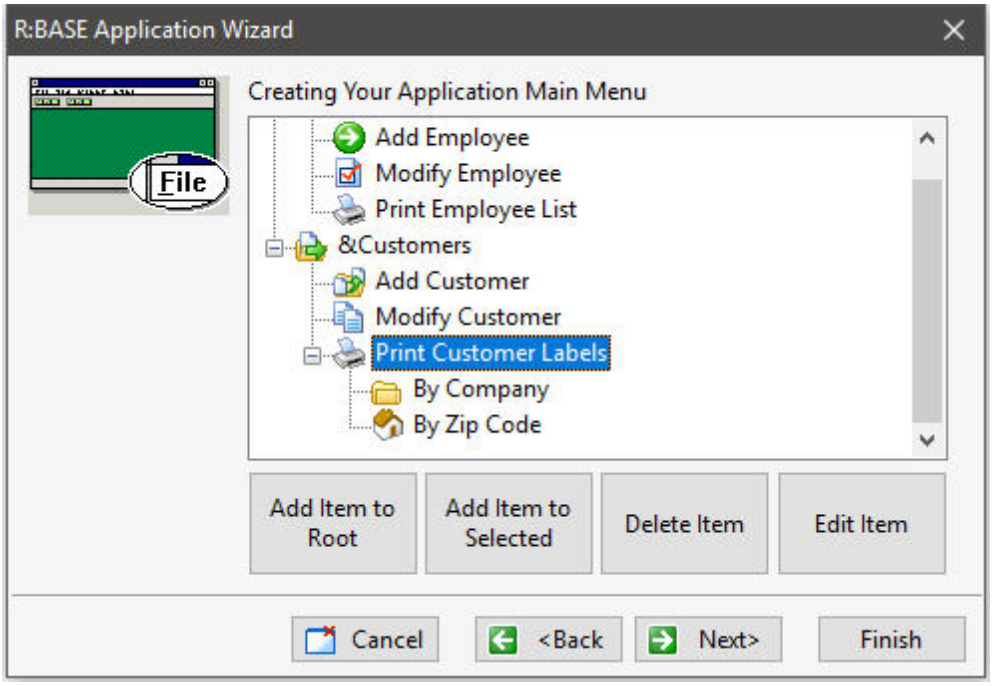
Now, will display a sub-menu to provide two ways to print customer labels.

10. Highlight "&Customers" and select the "Add Item to Selected" button.
11. Enter "Print Customer Labels" in the "Menu Item Caption:" field.
12. Add an image to the menu, then click the OK button.
13. Now, highlight "Print Customer Labels" and select the "Add Item to Selected" button.
14. Select "By Company" from the list of actions.
15. Click the OK button.

The menu "Print Customer Labels" now contains its own branch.

16. Highlight "Print Customer Labels" and select the "Add Item to Selected" button.
17. Select "By Zip Code" from the list of actions.
18. Click the OK button.

You may need to click on the plus sign (+) for the submenu to branch out for display.



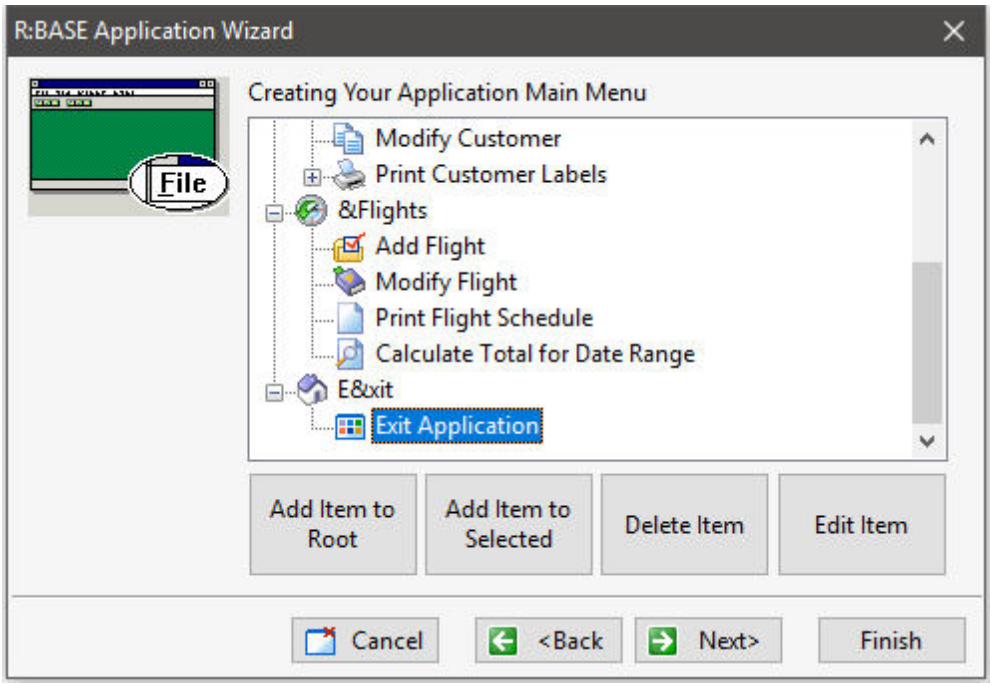
The drop-down menu for the menu "Customers" is complete.

10.11 Defining the Flights and Exit Menus

Now that you've defined the Employees and Customers menus, you're ready to name the other menus and define the pull-down submenu options. Use the information below and follow the previous procedure to complete this portion of your application.

Menu Name	Options
&Flights	Add Flight
	Modify Flight
	Print Flight Schedule
	Calculate Total for Date Range
E&xit	Exit Application

You have now defined all the menus for your application. The tree should look like the following.



Select the "Next" button to move on.

10.12 Final Steps

The next screen in the Application Wizard states, "Add Toolbar Drop-Down Menus to your Application". The screen will be skipped as we do not need to add any new menus.

1. Select the "Next" button.

The next screen in the Application Wizard states, "Add a Toolbar to your Application". The screen will be skipped as we do not need to add the Toolbar. To review how to add a toolbar and the toolbar menus, please review the "How To: Applications" section in the Help.

2. Select the "Finish" button.

You need to name the application. An application is used with only one database. To make it easy to keep track of which application is associated with a database, incorporate the database name in the application name.

3. Enter "skyApp.rba" in "File Name:" field.
4. Click the "Save" button.

Your application has now been saved and the Application Designer window is displayed with your application file, skyApp.rba, opened.

10.13 The Application Designer Environment

The application environment display tabs across the window, which may look familiar as they match the Application Wizard steps. Choosing a tab, and altering the settings can change any portions of the application file.

A toolbar for the Application Environment is displayed above the tabs.

From left to right, the following describes the functions of each button:

- New Application
- Open Application
- Save Application
- Close Application
- Test Application
- Set Application Design Password
- Set Database and Path
- Set User ID and Password
- Find in Custom EEPs
- Application Designer Help

For additional information on the Application Designer, select the Application Designer Help button.

Now that you have completed the design of your application, you can test the appearance and functionality.

10.14 Testing an Application

Just as you wouldn't fly a new plane without first testing the engine, landing gear, and other parts on the ground, you shouldn't trust your data to a new application until you've tested each part. In particular, you should ensure that the data entry, data editing, and reporting features work correctly.

Testing all potential exceptions on just a few rows of data confirms whether your design works well or whether there are things that need fixing or improving. Most applications are tested and refined a number of times before being used.

Next, we will test the menu options. You'll learn how to use the *skyApp* application. You'll modify data in the *employee, customer,* and *flights* tables, and print reports and labels.

Simply by clicking the "Test Application" button, you can actually run the application, and return to the Application Designer when you close it.

Select the "Test Application" button.

You can easily see the main areas or topics: Employees, Customers, Flights, and Exit.

None of the names on the menu bar is selected. When you choose a menu name, the drop-down menu is displayed. Now verify the menu options available in your application. Choose Employees, Customers, Flights, and Exit. You can choose a menu option by choosing it with the mouse or by using the defined hot key.

10.15 Testing the Employees Menu

When testing an application you need to run every menu option to be sure they all work as expected. To test *skyApp* for employee processing you're going to display and modify data, and print an employee list.

Remember that during the testing you are working with the actual database and any changes made will be saved.

Entering New Data in the Employee Table

When skyWrite Ink hires a new employee, you use the **Add Employee** option on the **Employees** menu to enter information about the new employee.

To add a new employee:

1. In the S*kyApp* application, choose **Employees**.

R:BASE displays the pull-down menu.

2. Choose **Add Employee**.

Empform, the form you created in Lesson Four, is displayed.

R:BASE automatically assigned an employee number.

In Lesson Four you tested the form by adding new data, so there's no need to test it again. Now that we know the Add option works, we can exit the form.

3. Close the form using the "X" in the top right corner or the "Close" button on the DB Navigator.
4. When you are prompted to save changes, click the No button.

Printing Data from the Employee Table

To look at the employee records, you can print an employee list on the screen.

To print the employee list report:

1. In the S*kyApp* application, choose **Employees: Print Employee List**.

The report is displayed on the screen in a window.

2. Maximize the screen

After viewing the rows, you realize that Charles Short's phone number is incorrect-it has been entered as 564-9123 and should be 564-9124.

3. Press the "Close" button to clear the report from the screen.

Modifying Existing Data in the Employee Table

To correct Mr. Short's row, use the Edit option on the Employees menu.

To change data for an employee:

1. In the S*kyApp* application, choose **Employees: Modify Employee**.

You are prompted for the last name of the person to be edited. R:BASE accepts any combination of lower and upper case letters. Earlier, wildcard were discussed when being prompted for data. Use the many wildcard character, percent (%), to only enter a few characters of Mr. Short's last name.

2. Enter "Sho%"
3. Click the OK button.

R:BASE displays *empform* with the information for Charles Short.

4. Press the [Tab] key down to the phone number field.
5. Change the last digit from "3" to "4."
6. Hover your cursor over the check mark button. "Save" will appear.
7. Click the "Save" button.

R:BASE replaces the row in the table with the changes you've made.

8. Close the form.

10.16 Testing the Customers Menu

Now let's test the application on the *customer* table. You created *custform* at the end of lesson 4, but you haven't tested this form yet.

Entering New Data in the Customer Table

To test *custform,* you'll add the information for a new customer.

To add a new customer:

1. In the S*kyApp* application, choose **Customers: Add Customer**.

Custform, the form you created earlier, is displayed.

2. Add the record shown below to the *customer* table.

Field	Data
custfirstname:	Laurie
custlastname:	Keller
company:	City Flowers
custaddl:	4301 Union Ave.
custadd2:	
custcity:	Seattle
coststate:	WA
custzip:	98101
custareacode:	206
custphone:	633-2632

3. When you have finished entering data in the last field, choose the check mark button to save the data and close the form. R:BASE adds the row to the *customer* table and returns you to the application's menus.

Modifying Data in the Customer Table

Assume that your contact at City Flowers has changed from Laurie Keller to Annette Ely. Following the procedure for the previous section for employees, edit the existing record for Ms. Keller.

When you finish editing the row be sure to save it.

Printing Customer labels

Printing customer labels is similar to printing the employee list, except for the sorting options. Next, you'll print the customer labels to the screen using both sort options.

To print customer labels:

1. In the S*kyApp* application, choose **Customers: Print Customer Labels**.

The submenu you defined is displayed. The two sorting options available for printing the labels are **By Company** and **By Zip Code**.

2. Choose **By Company**.

The label is displayed on the screen.

3. Maximize the screen.
4. After viewing the labels, press the "Close" to clear the labels from the screen.

Now print the labels by zip code. When you are done printing the labels, press the "Close" button to return to the application.

10.17 Testing the Flights Menu

To test *skyApp* for the *flights* table, you're going to modify data using *flightform,* and then print a flight schedule on the screen.

Editing Flight Information

The **Modify Flight** option on the **Flights** menu works differently than the modify options on the **Employees** and **Customer** menus. The action is defined to edit data based on *flightdate.* Let's look at and modify a row in the *flights* table.

To change data about a scheduled flight:

1. In the *skyApp* application, choose **Flights: Modify Flight**.

The process now differs from that for the **Employees** and **Customers** menus. R:BASE prompts you for the flight date. Enter the beginning date for the records you want to view.

2. For the flight date, enter "06/03/2008".
3. Click the OK button.

The flight form*, flightform,* is displayed.

R:BASE retrieves all rows from the *flights* table where the date is greater than or equal to June 3, 2008. The first row of data that equals or follows the specified date appears on the screen. This row is for the customer Nicki Jambour.

Now, let's scroll through the rows.

4. Choose the green right arrow button, which displays the hint "Next".

Another row for Performance Specialists is displayed.

5. Choose the green right arrow button again.

The third row of data that equals or follows the specified date is displayed. This row is for Aquaculture Gems, Inc. This customer had changed the skywriting time from noon to 2 pm.

6. Click in "Location/Time" and change the time.
7. Choose the check mark button to save the changes when you have finished.

We've just modified and saved this row.

Exit the form, by clicking the "Close" button, after you make the change.

Printing the Flight Schedule

Now you can print the Scheduled Flights report.

To print the list of scheduled flights:

1. In the *skyApp* application, choose **Flights: Print Flight Schedule**.
2. When you are done viewing the data, press the "Close" button to clear the report from the screen.

10.18 Exiting the Application

The application main menu includes an Exit option to leave the *skyApp* application. This option will close the application. When you are testing an application, you can always exit the application and return to the designer by pressing the "X", but in testing, the Exit menu option will not close the application. We'll test the Exit option by first closing the Application Designer and then running the application..

To close the Application Designer:

1. Choose **File: Close**.

You will be returned to the "Applications" menu in the Database Explorer.

To test exiting the application:

1. In the Database Explorer, select the *skyApp* from the list and click the "Run Application" option.

The application will launch.

2. Select **Exit: Exit Application**

The application window and R:BASE should close.

We still have more work to do, so launch R:BASE. Make sure you are in the directory where the *sky* sample database is located. Use the "Connection history" to easily connect to your sample work.

1. From the Menu Bar, choose **Databases: Connection history: (Your sample database)**

You will be connected to your sample database.

10.19 Modifying an Application

Either when you're testing a new application or using an existing one, you might discover features that you want to change. You can use the Application Designer to change your application in the same way that you created it.

To modify an application:

1. In the Database Explorer, click the "Applications" option. You'll see *skyApp* listed.
2. Highlight the application.
3. Click the "Design Application" option.

Now, make the following changes.

To add a image background:

In the Application Designer, you should be located at the "Application Main Window" tab

1. Select the "Load Image..." button from the "Image" panel.

You will prompted to navigate for an image.

For R:BASE X.5 Enterprise, navigate here: C:\RBTI\RBGX5E\Tutorial

For R:BASE X.5, navigate here: C:\RBTI\RBGX5\Tutorial

2. Select the "SKYWRITE.jpg" file, and choose the "Open" button.

3. Place a check in the "Show Image" check box.

The image will be displayed.

4. Make sure the "Image Style" specified is "Center".

Next, add an icon for the application caption bar.

To add an icon:

1. In the Application Designer, select the "Properties" tab
2. Choose the "Load Icon" button.

You will prompted to navigate for an icon image. If you have an icon image file on your computer, navigate to that location and press the "Open" button. If not, press "Cancel".

Without an icon file loaded, the R:BASE X.5 icon will be displayed.

Now, increase the application window size.

To increase the application window:

In the Application Designer, you should be located at the "Properties" tab

1. Change the "Width:" value to 800.
2. Change the "Height:" value to 600.

Now save all of the changes.

3. Choose **File: Save** from the Menu bar.

You may want to run the application to test any new changes.

4. Close the Application Designer by choosing **File: Close** from the Menu bar.

10.20 Running the Final Application

To run the skyApp application:

1. In the Database Explorer, click the Applications option.

Applications in the current directory are listed. The *skyApp* application you just built is listed.

2. Select "skyApp", and click the "Run Application" option.

The SKYAPP.RBA application starts and automatically opens the *sky* database. Your application menu bar replaces the R:BASE X.5 window.

The application can also be set up to launch directly from a desktop shortcut.

To set up a desktop shortcut for the skyApp application:

1. From the operating system desktop, right click on the background and select "New" > "Shortcut"

2. Use the "Browse" button to select "R:BASE X.5"

3. Click the "OK" button, then the "Next" button.

4. Change the name to "skyWrite Ink", and select the "Finish" button.

Another desktop shortcut will appear with the R:BASE X.5 icon with the text "skyWrite Ink" underneath.

5. Right click on the new shortcut, and select Properties.

The "Target:" and "Start in:" fields will control what R:BASE will launch, and from what location.

6. Edit the "Target:" by adding a space after the "EXE" name, followed by the SKYAPP.RBA application file.

It would read: C:\RBTI\RBGX5\RBGX5.EXE SKYAPP.RBA or C:\RBTI\RBGX5E\RBGX5E.EXE SKYAPP.RBA

7. Edit the "Start in:" field to insure it is the folder location where the RBA application file reside.

It may read: C:\RBTI\RBGX5\Samples or C:\RBTI\RBGX5E\Samples

8. Click the "OK" button.

9. Now, double click on the "skyWrite Ink" desktop shortcut. The application should launch.

10.21 Summary

In this lesson you have:

- Defined *skyApp,* an application with four menus.
- Added a condition to edit rows in the *employee* and *customer* tables by a specific last name.
- Ran the application and tested the options on the Employees, Customers, and Flights menus.

To check your knowledge, try answering the following questions:

1. How many databases is an application associated with?
2. How many menus can you have in an application?
3. How many menu levels can an application have?
4. How many menu levels does *skyApp* have?
5. In this lesson you built a condition to use when editing rows in the *employee* and *customer* tables. What is the purpose of building a condition?
6. When editing rows of data using a form, how do you display the next row?
7. How do you modify an application?
8. How do you use the hot key to open menus?

10.22 Tips

- You can use the RBAPP command at the R> Prompt to launch your application into the Application Designer.

  Examples:

  ```
  RBAPP 'Test Application.RBA'
  RBAPP skyApp.RBA
  ```

- You can use LAUNCH command at the R> Prompt or in a command file to run your application file.

  ```
  LAUNCH skyApp.RBA
  ```

- While building Actions, after selecting a form, report or label and building your WHERE clause and/or ORDER BY clause, if you would like the option of adding a user input dialog, use the '@' symbol.

  Examples:

  ```
  WHERE empid = @'Enter Employee ID'

  WHERE EmpLName LIKE @'Enter the first few characters of Employee Last Name followed by % symbol'

  WHERE EmpLName CONTAINS @'Enter Part of Employee Last Name'
  ```

- If you use the hyphen character "-" for the caption of an Action, this will create a separator bar that you can place between Menu Items to create an organized appearance.

- You can add an ampersand (&) in front of a letter in the caption field to create a "[Alt]" hot key for a Main Menu item. This places an underline beneath the appropriate letter when the application is run and the [Alt] key is pressed.

- During application development, R:BASE can create backups of your RBA files. To enable this feature, select Settings" > "Application Designer" > "Create Backup Copy" from the main Menu Bar. The backup copy will reside in the same folder as the original file, only the file extension will contain ".rb~".

Part XI

11 Lesson 10 - Maintaining a Database

This lesson covers two important database maintenance procedures: making backup copies of your database and application, and routinely freeing disk space no longer needed by the database, thereby increasing speed and efficiency.

At the end of this lesson you will be able to do the following:

- Copy your database files
- Back up your database
- Check your database for damage
- Restore your database files
- Compress your database
- Compress and copy your database files to another file

Note: Once you put a database into operation, it becomes another asset of the company to insure and maintain, just as skywrite Ink insures and maintains its airplanes. The insurance is the backup copy of the database. The maintenance plan includes procedures that help decrease the database's response time and ensure the integrity of its data.

11.1 Backups

The data contained in a database is valuable. For skyWrite Ink, data represents time and money. No one can predict when or how a database will be damaged. If human error or a technical malfunction destroyed or damaged the data, all employee data would be lost as well as the valuable customer and flight information. To prevent this, you should make a copy of the database on a regular basis. The copy is referred to as a backup. You can store backups on a tape drive, on a hard disk, or on floppy disk(s).

While it seems like extra work to make regular backups, it is good insurance and not difficult. With a backup, you can restore the data. Without a backup, your database is gone.

If you do not make good, consistent, daily backups then how important is your data? Yes, there are software utilities and techniques available to recover data from damaged databases, but no method is 100% reliable. None is guaranteed to recover 100% of your data, and using such a data recovery method may involve hours to days of down time.

Restoring a current backup and re-entering any missing data is by far the fastest and most accurate method of data recovery. A current backup usually ranges between 12-24 hours since the data was last modified. With the number of commercial software programs available to do backups there is no excuse for not doing them. If you have a large database, invest in a streaming tape backup. The cost of regular backups saves money. You will recover the cost of the backup and more the first time you use it.

Store backups away from the computer. Use off site backups for the most important, irrecoverable data. Store disks in a locked, fireproof box.

Test your backup procedure to make sure the procedure works. The time to find out that there is a bad disk or the data did not copy is not when you need to restore the backup. Who has not heard of users faithfully executing backup commands, but, until they needed to restore, not realizing that the procedure is not completing successfully? In fact, two different backup methods are best for vital data.

How important is your data? How much time and effort would be involved if you had to recreate the data from scratch, assuming that you can recreate all the data? What state would your business be in if someone breaks into your office and steals your computer? What about a hard disk failure? Do not assume "that will never happen to me". It happens and it only needs to happen once. Backups do not take time; they save you time, and money.

11.2 What to Back Up

In the previous lessons, you learned about various types of R:BASE files: the application files (.RBA), the command files, (.RMD) and the database files (.RB1, .RB2, .RB3, and .RB4) or (.RX1, .RX2, .RX3, and .RX4). You need to back up the command files and application files only when they change. However,

because the database files contain any changes or additions you make to the database, you should back up these files regularly. Throughout the rest of this lesson, we will be referring to these files.

To determine how often to back up the database, you should consider the following:

- How much data is entered into the database and how often?
- How hard would it be to reconstruct the data if lost?

skyWrite Ink is a fairly small company. Employees enter a few rows of information about new customers and flight information each day. Because there are only a few new rows added daily and the information is easy to reconstruct, weekly backups are sufficient.

11.3 Checking the Database

Because your information is important, you want to check the database for errors before making a backup. You don't want to back up a damaged database. In addition, you should check a database after power failures, electrical storms, and any time the database is not exited normally.

To check the sky database for errors:

1. In the Database Explorer, click the Databases option.
2. Select "sky" and click the Connect option.
3. Choose **Tools: R> Prompt**, select the R> button on the toolbar (third button from the left).

The "R:BASE R> Prompt" window opens.

4. Type "AUTOCHK FULL" and press [Enter].

R:BASE checks the database. The message "No errors found" is returned. If the database was damaged, an error message indicating the type of damage would be displayed.

11.4 Copying Database Files

If your database is small enough to fit on your desired media; flash drive, CD-ROM, external hard drive, floppy disk, etc., you can simply copy the four database files. You can use the COPY command in the "R:BASE R> Prompt" window or the operating system COPY functionality. Copying doesn't change the database files. Always copy the four database files together. The files can be copied to another media or to another location on your hard drive. You can copy the four database files as a set when you use an operating system wildcard (?, %, or *).

You should store the backup copy of your database on a storage medium that is different than the one on which your original files are stored. For example, if you are using a hard disk to store the data, you can back up your database to a tape drive or external hard drive. In this exercise, you will back up the data from the hard drive to a network location.

The UNLOAD and RUN commands will be used in the following exercises.

11.5 Understanding R:BASE UNLOAD and RUN

The R:BASE UNLOAD and RUN commands are useful tools for backing up databases and for backing up the database structure elements. The R:BASE UNLOAD command makes an ASCII file of R:BASE commands; it does not compress the data in any way. The UNLOAD command copies your database into a file that has a different format than your database files. This file cannot be used by R:BASE in place of the database files. Before you can use the contents of the backup file, you must return it to the four database files.

There are three parts to the R:BASE UNLOAD command: STRUCTURE, DATA, and ALL. The STRUCTURE contains table and column definitions including indexes, view definitions, granted access rights, rules, constraints, auto-number definitions, and comments. The STRUCTURE does not contain the form, report, or label definitions. Those are actually stored as data. The DATA contains the data from the user-defined tables as well as data from some system tables (forms, reports, labels, and layouts). The structure and data of a database are backed up together by using ALL.

Between using ALL or using STRUCTURE and DATA separately as a means to back up the database, the preference is entirely up the you. As it is always recommended that the database be checked before it is backed up, either method will work when restoring a database.

The only negative aspect to using ALL is that if you ever encounter errors when restoring a database, it is very difficult to edit the file to correct problems as both the structure and data are in one large ASCII file. With ninety-nine percent of the problems with restoring a database involving a problem with re-creating the database structure, it is recommended that a backup of the STRUCTURE and DATA be stored separately. Ideally, the best method would be to create frequent backups using ALL, and less frequent backups of the STRUCTURE and DATA separately. This way, if there is any need for restoring a database, you will have no issues.

In addition to backing up structure to one file and data to another file for ease of correcting problems, the structure of a database does not usually change once it is set. Data, obviously, changes on a daily basis. By backing up database elements separately, the backup process is customized to ensure accurate and easily restored backups.

The backup process uses three commands:

```
OUTPUT filename
UNLOAD <parameters>
OUTPUT SCREEN
```

The OUTPUT command creates the file to contain the backup data; the UNLOAD command puts the requested data in the file; the second OUTPUT command closes the file. The created file is an ASCII file containing a series of R:BASE commands-the commands that would recreate the database if they were issued at the R> Prompt.

The first OUTPUT command can be enhanced with the added ENCRYPT parameter. When issuing the ENCRYPT parameter, you will be immediately prompted for a password. Then, when the file is RUN, or opened with the R:BASE Editor, you will be prompted for the password. The password is protected with a 512-bit encryption algorithm. To use encryption, the same commands above would be used as follows:

```
OUTPUT filename ENCRYPT
UNLOAD <parameters>
OUTPUT SCREEN
```

The file created always begins with a series of SET commands to set the database environment exactly as it was when the UNLOAD command was issued. This ensures that the QUOTE, DELIMIT and NULL settings match those in the commands and data in the backup file. Object names, such as table names, will be enclosed within the IDQUOTES character, the current setting for DELIMIT is used to separate values, text strings are enclosed in the current setting of QUOTES, and the current value of WIDTH affects the width of data lines in the backup file. When backing up DATE data, R:BASE always sets the DATE FORMAT and SEQUENCE to a four-digit year to ensure that the data is restored accurately. The end of the UNLOAD file resets these to the database setting.

The following are three tips that will ensure your database is restored from a back up properly:

- Set the null symbol to -0- (the R:BASE default) before backing up.
- Do not set a special character to the same setting as another special character.
- For best results backing up, keep all the default settings.

To view the current database character settings and operating conditions:

Choose **Setting: Configurations Settings** from the Menu Bar.

Since a backup file is an ASCII file, when using the R:BASE UNLOAD command to backup up your database or parts of your database, view the file with any text editor to become familiar with the format.

UNLOAD does not back up computed-column values; the values will be computed when the database is restored. UNLOAD does not change the data or the structure in the original database. Use the RUN command to restore your data. If a UNLOAD command is included in a transaction when transaction

processing is on, the backup cannot be rolled back. The UNLOAD command creates a file with a .LOB extension for binary large objects, and a file for the data and/or structure.

11.6 Backing Up Database Structure

When you backup database structure, R:BASE puts commands in the file to rebuild the structure from scratch. There is always a CREATE SCHEMA command to create or open the database. The CREATE SCHEMA command automatically creates all system tables in R:BASE. The table and column definitions are backed up to CREATE TABLE commands. Views become CREATE VIEW commands. Indexes are CREATE INDEX commands and so on.

If you are transferring information to a different database, you need to edit or remove the CREATE SCHEMA command. It is better to edit the command line to change the database name instead of removing the line. A DISCONNECT command is included in the backup file so that database setting such as STATICDB and ROWLOCKS can be set. If the CREATE SCHEMA line is deleted, the following lines must also be deleted

```
DISCONNECT
SET STATICDB OFF
SET ROWLOCKS ON
SET FASTLOCK OFF
```

Many converted databases contain table and column names that are reserved words but not illegal. Many times, even illegal names continue to work, depending on where they are used. The CREATE TABLE command, however, errors out on illegal names and the database does not restore. Often, all that is necessary is to use the setting SET ANSI OFF. This limits the reserved word list. It does not allow illegal names or names like OWNER. If you have backed up your database structure to a separate file it is a relatively simple process to edit the file and change the reserved or illegal name. The name is changed not only in the CREATE TABLE command, but in CREATE VIEW commands, in CREATE INDEX and GRANT commands, in RULES definitions, in AUTONUMBER definitions, in ALTER TABLE commands that define constraints, and in comments.

The following sequence of commands backs up database structure, including forms, reports and labels. Forms, reports, and labels would be backed up again each time they are modified or new ones added.

```
OUTPUT dbname.str
UNLOAD STRUCTURE
OUTPUT SCREEN

OUTPUT dbname.frm
UNLOAD DATA FROM SYS_FORM3
OUTPUT SCREEN

OUTPUT dbname.rep
UNLOAD DATA FROM SYS_REPORTS3
OUTPUT SCREEN

OUTPUT dbname.lab
UNLOAD DATA FROM SYS_LABELS3
OUTPUT SCREEN
```

This backs up all the database structure elements to separate files making it easy to restore individual elements. Structure can be backed up for individual tables or views. Backing up structure for an individual table includes other structure elements defined for that table such as rules, auto-number and comments. Defined constraints, indexes, access rights autonumbered definitions, comments, rules and triggers can also be backed up separately.

Note that the created file is an ASCII file and contains the database owner password on the CREATE SCHEMA command. If you do not utilize the OUTPUT <filename> ENCRYPT method, store your R:BASE backup files in a secure location to protect that password.

11.7 Backing Up Database Data

Your data is, of course, the most important part of the database. The structure can be recreated, but it may take months to reenter data. Some of the data may go back years and not be recoverable. The UNLOAD DATA command creates an ASCII file with SET commands at the beginning to set the environment, then a LOAD *tblname* command followed by the data in ASCII delimited format.

If you backup form, report and label data with the structure, do not use the UNLOAD DATA command to backup your data tables. That command includes the data for forms, reports and labels. Instead, use the UNLOAD DATA FOR *tblname* command and do each table individually. Some tables, such as lookup tables that do not change frequently, are backed up together. Create command or application files to do the backup process. Setting up a backup routine is a one-time process, but it pays for itself the first time you need to restore.

For example:

```
OUTPUT looktab.dat
UNLOAD DATA FOR looktab1
UNLOAD DATA FOR looktab2
UNLOAD DATA FOR looktab3
OUTPUT SCREEN

OUTPUT tblname1.dat
UNLOAD DATA FOR tblname1
OUTPUT SCREEN

OUTPUT tblname2.dat
UNLOAD DATA FOR tblname2
OUTPUT SCREEN
```

If you do not wish to backup form, report and label data separately within their own files, you can easily include all of the data in one file.

For example:

```
OUTPUT dbname.dat
UNLOAD DATA
OUTPUT SCREEN
```

When backing up data, it is important to set the NULL symbol in the database to -0- or some other character that does not appear in your data. When R:BASE loads the data during the restore, it checks the first four characters of each data field to see if it matches the NULL symbol. If it matches, the field is loaded as a NULL. This is why backing up and restoring forms and reports with the NULL symbol set to a blank causes problems - part of the form and report data is blank lines, these blank lines become NULL if restored with the NULL symbol set to a blank and the forms and reports do not work correctly.

In R:BASE, you will have large object (LOB) data in your database. All forms, reports, and labels are stored as LOB data rather than ASCII data. When the output from the UNLOAD command is sent to a backup file, (e.g. *db_backup.all*) the LOB data will be placed into a separate LOB file, only it will have a .LOB file extension (e.g. *db_backup.lob*). The backup output file contains references to the LOB file where the table data is loaded. Both files are required in order to restore the database, and they must reside in the same folder location.

11.8 Backing Up a Database

To back up the sky database from your hard drive to a network drive:

1. For this example, assume you have a "Backup" folder on a network drive "F:".
2. In the Database Explorer, click the Databases option.
3. Select "sky" and click the Connect option.

4. Choose **Tools: R> Prompt**, or select the R> Prompt button on the toolbar (third button from the left).

The "R:BASE R> Prompt" window opens.

5. Type "OUTPUT F:\Backup\SKY.ALL" and press [Enter].
6. Type "UNLOAD ALL" and press [Enter].

R:BASE starts the backup process to the network drive.

7. Type "OUTPUT SCREEN" and press [Enter].

The database backup is now located in drive F:\Backup. Two files will be located in the folder with the names SKY.ALL and SKY.LOB.

11.9 Confirming the Backup File was Created

You can confirm that the backup was completed.

To confirm the backup file was created:

1. At the R> Prompt, type "F:" and press [Enter].

The R:BASE working directory changes to the root F: drive.

2. At the R> Prompt, type "CHDIR Backup" and press [Enter].

The R:BASE working directory changes to the F:\Backup folder.

2. At the R> Prompt, type "DIR" and press [Enter].

R:BASE displays a line of information for each file in the folder. Each line lists the date and the time the file was created, its size in bytes, and file name. The number of bytes for the backup file is relatively large because the backup file contains the information stored in each of the four database files.

11.10 Restoring a Database

If the original database files become damaged, you can restore the files using a backup file. R:BASE uses the backup file to restore the four database files to the same drive and directory of the backup file location. Make sure the current location of the backup file contains enough free space before attempting to restore the database. The backup file is executed using the R:BASE RUN command. The commands in the file are read and executed, and the database is rebuilt. The restore process will not overwrite existing database structure or data. It uses the LOAD command for data and appends that data to the end of the table. Remember, the backup file is just an ASCII file of R:BASE commands.

Before restoring an entire database, delete or rename the existing database files that have the same name. Otherwise, you will receive an error stating the database already exists. Before restoring forms, reports or labels, you must delete all rows from the appropriate system table. Otherwise, the result will be multiple forms, reports or labels with the same name. Before restoring a user table, delete all rows from the table, or copy the table, and delete rows from the original table. Otherwise, again you will have duplicate entries.

Before you restore the database, change the name of the damaged database files. The *sky* database really isn't damaged; we are only testing the backup.

To change the name of the damaged database:

First, the working directory needs to be changed from the network drive back to where the *sky* database is located.

1. From the R:BASE Menu Bar, select **Database: Connection history: (Your sample *sky* database)**.

The R:BASE working directory changes to the *sky* database folder.

2. In the Database Explorer, select "Databases" from the Group Bar.
3. Select "sky".
4. Choose "Disconnect Database" to disconnect from the database.

Now, rename the database.

5. Click the "Rename" option.

You are prompted to enter the new name for the database files.

6. Enter "SKYBAK" as the new name.
7. Click the OK button.

R:BASE renames the SKY database files to SKYBAK.

The next step is to restore the database on the network drive location.

You can restore an unloaded database backup with the RUN command. Remember, the RUN command restores the four database files to the same drive and directory of the backup file location. You can either copy the backup files to another location, but right now they are still in the "F:\Backup" folder.

To restore a database:

1. At the R> Prompt, type "F:" and press [Enter].

The R:BASE working directory changes to the root F: drive.

2. At the R> Prompt, type "CHDIR Backup" and press [Enter].

The R:BASE working directory changes to the F:\Backup folder.

3. At the R> Prompt, check your location by typing "DIR" and press [Enter].

Right now, R:BASE should be at the "F:\Backup" folder. The R> Prompt output console will show the current location and the two backup files.

4. At the R> Prompt, type "RUN SKY.ALL" and press [Enter].

The output console will show the creation and loading of tables.

5. When the process is complete, focus will return to the R> Prompt output console with the cursor blinking again.

Now, check folder for the database files.

6. At the R>, type "DIR" and press [Enter].

The R> Prompt output console will show the current location, the four database files, and the two backup files.

After you are sure that the backup files were successfully restored, you can delete the damaged database. It is always recommended that you verify the integrity of your backups by restoring the backups periodically and checking for errors.

7. Reconnect to the original database by selecting **Database: Connection history: (Your sample database)** from the Menu Bar.

11.11 More on the UNLOAD Command

The UNLOAD command also has the "AS ASCII" or "AS CSV" options for creating a file of data only, with no R:BASE commands, which will even unload the data from computed columns or expressions.

The AS ASCII/CSV options create a delimited ASCII file of data with no R:BASE commands. There are no SET commands or LOAD *tblname* command. The file is delimited with commas (or the current DELIMIT setting) separating the fields, and text data enclosed in single quotes (or the current QUOTE setting). The options are used to create a file to transfer data to another program, or to a different table. A file created using UNLOAD DATA...AS ASCII/CSV is put back into a database using the LOAD command. Only one table can be unloading to a file using the AS ASCII/CSV options. There are no R:BASE commands separating the data from different tables.

The UNLOAD command assumes that the data is returning to an R:BASE database. UNLOAD includes R:BASE commands that set the database environment and identify the table to load the data into. Since the data is loaded back into an R:BASE table, data for computed columns is not unloaded. It is re-computed as the data is loaded. The AS ASCII/CSV options include no R:BASE commands and is just data. As such, UNLOAD DATA...AS ASCII/CSV includes the data from computed columns in the output file. In addition, the AS ASCII/CSV options allow the use of expressions in the USING *collist* parameter.

For example, the code below creates the file that follows it:

```
UNLOAD DATA FOR employee USING +
empid, (empfname & emplname) , empaddress, + (empcity + ',' & empstate & empzip) AS
ASCII

102, 'June Wilson', '3278 Summit Drive', 'Seattle, WA 98115'
129, 'Ernest Hernandez', '12390 Windermere Dr.', 'Seattle, WA 98115'
131, 'John Smith', '3050 N.E. 41st', 'Seattle, WA 98105'
133, 'Peter Coffin', '4105 29th Ave N.E.', 'Duvall, WA 98004'
160, 'Mary Simpson', '101 West Mercer', 'Redmond, WA 98052'
165, 'Darnell Williams', '8806 88th Street', 'Seattle, WA 98103'
166, 'John Chou', '5001 Main Street', 'Woodinville, WA 98072'
167, 'Sandi Watson', '1002 S. Front Ave.', 'Redmond, WA 98052'
```

The full name and the city, state and zip code are considered single fields. This option makes it easy to format the data for transfer to other software programs. A WHERE clause and an ORDER BY clause can also be used when a single table is unloaded.

The UNLOAD DATA FROM <tablename> AS ASCII/CSV command specification will not work with columns where LOB data exists. If you have the requirement to unload table data in an ASCII or CSV format from a table that uses LOB data, you must add the USING *collist* parameter to the command in order to capture the text data.

11.12 Compressing the Database

When you delete rows or remove columns or tables from a database, the information is gone, but the disk space it occupied remains unusable because R:BASE still regards it as assigned space. Also, as you add or modify data in tables, R:BASE stores the data wherever space is available on the disk. Over time, the rows of a particular table can become scattered over the disk, causing R:BASE to spend more time looking for the rows when you use the table.

11.13 Pack In Place Versus Reload

Once you have ensured the integrity of your database files, you can compress your database. R:BASE has two options for compression, Pack In Place and Reload, which are compared in the table below.

Pack In Place	**Reload**
Eliminates unusable disk space	Eliminates unusable disk space
Does not reorganize rows	Reorganizes rows
Does not change response time	Might improve response time

Operates row by row within the open database	Copies the open database to new files
Does not require extra disk space	Requires that a copy of your database fit on your hard disk

The menu options for Pack In Place and Reload are represented as the PACK and RELOAD command.

11.14 Using the Pack In Place Option

If you have not added, changed, or deleted many rows in your database, use the Pack In Place option. Pack In Place eliminates the space occupied by deleted rows, columns, or tables within the existing database. This option cannot reorganize rows on the disk. Pack In Place also has options that allow you to restructure indexes or compress the database structure.

Before you compress your database, make sure you have a current backup of it. If your database is large, the compression can take a considerable amount of time, even several hours. If the operation is interrupted for any reason, your database will be damaged or destroyed.

Pack In Place has three options:

Schema - eliminates unused space from the database structure only
Keys - packed the entire index file (File 3) and rebuilds indexed columns
All - eliminates unused space from both database structure and database data and rebuilds indexed columns
Passwords - eliminates unused space from the SYS_PASSWORDS System Table
Index - packs all indices for the currently connected database, and is supported in multi-user environments.

In the following exercise, you'll compress the database using the "Pack in Place" menu option.

To compress the *sky* database files using Pack in Place:

First the database must be connected in single-user mode, which must be set when you are disconnected from the database files.

1. In the Database Explorer, select "Databases" from the Group Bar.
2. Select "sky".
3. Choose "Disconnect Database" to disconnect from the database.
4. From the R:BASE Menu Bar, select **Settings: Configuration Settings**
5. In the dialog window, choose the "Multi-User" tab, and remove the check from the "MULTI" check box.
6. Click the OK button, and confirm the changes were saved.
7. From the R:BASE Menu Bar, select **Database: Connection history: (Your sample *sky* database)**
8. From the R:BASE Menu Bar, choose **Utilities: Pack in Place: All**

R:BASE displays the following warning message: "You should back up this database before you PACK it." and a confirmation dialog window prompting, "Continue with an in-place PACK of your database?"

9. Click the Yes button to continue with the process because you already have a backup copy of the database.

R:BASE displays all the tables, columns, and indexes as it packs them. Pack In Place rebuilds the indexes for all indexed columns. If messages are set off in this database, you will not see these messages.

Once you are complete, disconnect from the database and set R:BASE back to "multi-user" mode.

11.15 Using the Reload Option

If you have added, changed, or deleted many rows in your database, use the Reload option. Reload copies the open database table by table to a new database name that you specify, preserving the original database. As it duplicates the database, Reload places the rows for each table in a single area of your database data file (File 2), thereby eliminating unusable disk space. Reload does not have options that allow you to compress indexes or the database structure only.

To use the Reload option, you must have room for a copy of your database on your hard disk. If you do not have enough space on your hard disk, use the Pack In Place option to compress the database instead.

To compress the *sky* database files using Reload:

1. If the *sky* database is not open, click the Databases tab in the Database Explorer and select "sky".
2. Click the Connect option.

3. From the R:BASE Menu Bar, select **Utilities: Reload**

You will be prompted to enter a new database name.

4. Enter SKY2 and select the "Save" button.

After the process is complete, you will see the additional database with the name *sky2* listed in the Database Explorer. Once you are sure that your new database is working properly, you can use the Database Explorer to delete the old database files and rename the new files to the original database name.

11.16 Summary

A database like the *sky* database becomes a valuable asset to a company and needs to be carefully maintained. You must make backup copies on a regular basis.

In this lesson you have:

- Learned the difference between copying database files and backing up database files.
- Learned the importance of back ups.
- Backed up the *sky* database to a floppy disk using the UNLOAD command.
- Learned the difference between packing or compressing a database in place, and packing to another file.

To check your knowledge, try answering the following questions:

1. Why is it important to make regular backup copies of your database?
2. What are the database files?
3. What is the command used to restore a database?
4. What is the purpose of the Reload and Pack In Place options, and how do they differ?
5. What should you always do before packing (compressing) your database?

Part XII

12 Useful Resources

. R:BASE Home Page:	https://www.rbase.com
. Up-to-Date R:BASE Updates:	https://www.rbaseupdates.com
. Current Product Details and Documentation:	https://www.rbase.com/rbgx5
. Support Home Page:	https://www.rbase.com/support
. Product Registration:	https://www.rbase.com/register
. Official R:BASE Facebook Page:	https://www.facebook.com/rbase
. Sample Applications:	https://www.razzak.com/sampleapplications
. Technical Documents (From the Edge):	https://www.razzak.com/fte
. Education and Training:	https://www.rbase.com/training
. Product News:	https://www.rbase.com/news
. Upcoming Events:	https://www.rbase.com/events
. R:BASE Online Help Manual:	https://www.rbase.com/support/rsyntax
. Form Properties Documentation:	https://www.rbase.com/support/FormProperties.pdf
. R:BASE Beginners Tutorial:	https://www.rbase.com/support/rtutorial
. R:BASE Solutions (Vertical Market Applications):	https://www.rbase.com/products/rbasesolutions

Part XIII

13 Feedback

Suggestions and Enhancement Requests:

From time to time, everyone comes up with an idea for something they'd like a software product to do differently.

If you come across an idea that you think might make a nice enhancement, your input is always welcome.

Please submit your suggestion and/or enhancement request to the R:BASE Developers' Corner Crew (R:DCC) and describe what you think might make an ideal enhancement. In R:BASE, the R:DCC Client is fully integrated to communicate with the R:BASE development team. From the main menu bar, choose "Help" > "R:DCC Client". If you do not have a login profile, select "New User" to create one.

If you have a sample you wish to provide, have the files prepared within a zip archive prior to initiating the request. You will be prompted to upload any attachments during the submission process.

Unless additional information is needed, you will not receive a direct response. You can periodically check the status of your submitted enhancement request.

If you are experiencing any difficulties with the R:DCC Client, please send an e-mail to rdcc@rbase.com.

Reporting Bugs:

If you experience something you think might be a bug, please report it to the R:BASE Developers' Corner Crew. In R:BASE, the R:DCC Client is fully integrated to communicate with the R:BASE development team. From the main menu bar, choose "Help" > "R:DCC Client". If you do not have a login profile, select "New User" to create one.

You will need to describe:

- What you did, what happened, and what you expected to happen
- The product version and build
- Any error message displayed
- The operating system in use
- Anything else you think might be relevant

If you have a sample you wish to provide, have the files prepared within a zip archive prior to initiating the bug report. You will be prompted to upload any attachments during the submission process.

Unless additional information is needed, you will not receive a direct response. You can periodically check the status of your submitted bug.

If you are experiencing any difficulties with the R:DCC Client, please send an e-mail to rdcc@rbase.com.

Index

- . -

.RB 42

- A -

action 173, 177, 179
add 54
add object 127
Add Text Object 88
add title 127
Additional Objects 90
Aggregate 56
alias 68
align 86, 120, 128
align object 86, 128
align objects 131
align text 128
aligning 120
All 198, 201, 205
application 170
application designer 187
application menu 183
application wizard 171
AS ASCII 204
AS CSV 204
ASCII 198, 200, 201, 202, 204
AUTOCHK 198

- B -

back up 197
BACKUP 194, 197, 200, 201
bands 126
Bevel 89
Bevel Object 89
BLOB 57
break 133, 134
breakpoint 133, 134
browser 70
build label 146
builder 60, 67
building menu 173

- C -

calculate 56
Change Object Prperties 105
Change Properties 105
change section 127
Change Size 83
change text 120
CHDIR 202
check 198
CLEAR 54, 164
Close 91
Close Form 91
close label 150
close report 139
Column 51, 53, 54, 55, 61, 68, 69, 96
column fields 129, 130
Column Object 96
column objects 129, 130
Combo Box 99, 102
ComboBox 99
Complete 104
Complete Form 104
Completing 104
compress 204
compression 150
Compute 56
computed 44
computed columns 44
concatenate 164
condition 62, 64
CONNECT 162
copy 198
create 31, 66
create database 31
Crosstab 56
CSV 204
CTXT 162, 164
CURRENCY 162
custom 81, 92, 112, 125
Custom Form 92
custom report 112, 125
customer label 146
Customize 81
Customizing Form 81

- D -

daily total 136
data 65, 198, 201
data rules 44
data types 162, 164
database 198
database design 21, 28
database files 42
DATE 164
DB Calc 136, 137
DB Edit 96
DB Edit Object 96
DB Lookup 99
DB Lookup Combo Box 99
DB Memo 98, 130
DB Text 129
DB User Defined 102
DB User Defined Combo Box 102
DBCalc 136, 137
DBMemo 130
DBText 129
define table 31
defining variables 147
delete 54
design 20, 21, 28, 67, 74, 170
designer 74, 115, 187
desktop 193
DIALOG 164
DIR 202
display format 135
display row 50

- E -

edit 54, 120
edit action 177
edit application 192
edit rule 44
edit section 127
Edit Size 83
Edit Static 81, 88
edit table 44
edit text 120
Edit Text Object 81, 88
editor 57
ENCRYPT 198
encryption 198
Enter Text Object 88
exit application 191
Expression 93

- F -

feedback 210
files 42, 197
footer 137
foreign 36
foreign key 36
Form 74, 91
Form Design 74
Form Designer 74, 76
Form Interface 76
Form Properties 90
Form Summary 108
Form Wizard 74
format 135
Format tool bar 127
Function 150, 162, 164

- G -

grand total 137
gray bands 126
Group 56

- H -

hidden 51, 52
hide 51
Hint 105
Hints 105
horizontal spacing 129
hot key 177, 184
hotkey 184

- I -

IDQUOTES 198
IF...ENDIF 164
image 57
import 43
import data 43

index 40, 205
indexes 40
Interface 76
IS NULL 164

- K -

key 36
keys 36, 205

- L -

label 127, 145, 146
Label button 127
label components 148
label field 148
label objects 148
label preview 149
label template 146
label variable 147
large object 201
LAUNCH 123
launching 123
launching files 123
line compression 150
link 68
LIST 42
LOAD 202
LOB 201, 204
lock 53, 54
Lookup 99
Lookup Combo Box 99

- M -

mail label 145
mailing label 145
many 64
Memo 98
menu 102, 173, 183
mode 54
modify table 44
Move 53, 82
Move Object 82
multi 55

- N -

new table 31
NOTE 57
NULL 133, 198, 201
Null format 133

- O -

Object 96, 97
Object Properties 77
Object Property 77
Object Prperties 105
Object Size 83
object spacing 129
objects 90, 131
Order 106
OUTPUT 198, 200, 201

- P -

pack 204, 205
Page Header 118
Page Title 118
Passwords 205
PAUSE 2 162, 164
phone list 112
Place Text Object 88
placing 129, 130
placing column fields 129, 130
placing column objects 129, 130
placing componenets 128
Placing DB Edit 96
placing fields 128
placing label objects 148
placing objects 95, 128
placing text 128
Placing Text Object 95
Placing Variable 97
Pop 102
Popup 102
Pop-up 102
Pop-up Menu 102
preview 122, 132
preview label 149
preview report 122, 132

primary 36
primary key 36
print 65, 122, 138, 139
print data action 179
print report 122, 139
printing style 138
Properties 77, 90, 105
Property 77

- Q -

QBE 68
query 57, 60, 61, 62, 64, 65, 68, 70, 71
quick 74, 112
Quick Form 74
quick report 112
QUOTE 204
QUOTES 198

- R -

RBEDIT 162
redisplay 52
reload 204, 206
remove report section 126
remove section 126
report 112, 140
report designer 115
report footer 137
report section 127
report sections 126
report tips 140
report tool bar 115
report wizard 113
reports 112
resize 120
resize objects 131
resizing 120
RETURN 162, 164
rich text 150
row 54
rule 44
rules 44
RUN 162, 164, 198, 202
run application 193
Run Form 91, 107
Running a Form 91, 107

- S -

save 71, 91, 132
Save Form 91
save label 150
save report 132, 139
schema 205
Scroll 98
Scrolling 98
Scrolling Object 98
section 126, 127, 133, 134
SELECT 69, 162, 164
SET 198
SET VAR 162, 164
settings 68, 194
shortcut 193
single 55
size 53, 83, 204
sort 55
spacing 129
spacing objects 129
SQL 69
Standard Controls 127
Start in 193
Static 81, 88, 95
Static Text 81, 88, 95
structure 198, 200
style 138
Summary 108
system table 42
system variables 121

- T -

Tab 106
Tab Order 106
table 31, 68
Tally 56
Target 193
techniques 194
template 146
test 188
test application 188
Test Form 91
Testing a Form 91
TEXT 81, 88, 95, 162, 164

text object 81, 88, 95, 128
tip 140
tips 140, 194
Toobar 76
tool bar 76, 115
Toolbars 76
Total 136, 137

- U -

Understand Form Interface 76
UNLOAD 198, 201, 204
UNLOAD ALL 201
unlock 54
User Defined 102
USING collist 204
Using Expression 93
Using Variable 93

- V -

Var 97
Variable 93, 97
Variable Object 97
variables 162, 164
vertical spacing 129
view 66, 67, 68, 69, 71
view row 50
viewer 57

- W -

WHERE 164
wildcard 177
wizard 57, 74, 113, 171

Back Cover