ReportWriter User's Guide

Version 10.1

SYNERGY
DE

Printed: June 2013

The information contained in this document is subject to change without notice and should not be construed as a commitment by Synergex. Synergex assumes no responsibility for any errors that may appear in this document.

DCN RW-01-10.1_02

Synergex
2330 Gold Meadow Way
Gold River, CA 95670 USA

http://www.synergex.com
phone 916.635.7300
fax 916.635.6549

Contents

Preface

Part 1: User’s Guide

1 Welcome to ReportWriter

7 Designing Your Report

8 Using Temporary Fields

11 Report Maintenance

12 ReportWriter Utilities

Part 2: Developer's Reference

13 Distributing ReportWriter with Your Application

14 Customizing the ReportWriter Environment

15 Customizing ReportWriter Routines

18 Miscellaneous ReportWriter Information

Appendix A: Maximums

Appendix B: Date and Time Formats

Appendix C: Error Messages

Appendix D: Data Formats

Appendix E: Distributed Shortcuts

Appendix F: Environment Variables

Preface

About this manual

This manual is your complete reference guide to Synergy/DE™ ReportWriter. Part 1, which includes a tutorial, is intended for end-users of ReportWriter. Part 2 is a reference for developers who are including ReportWriter in an application. The information in this manual applies to all operating systems on which ReportWriter is available, unless otherwise noted.

Manual conventions

Throughout this manual, we use the following conventions:

- In code syntax, text that you type is in `Courier` typeface. Variables that either represent or should be replaced with specific data are in *italic* type.
- Optional arguments are enclosed in *[*italic square brackets*]*. If an argument is omitted and the comma is outside the brackets, a comma must be used as a placeholder, unless the omitted argument is the last argument in a subroutine. If the comma is inside the brackets and an argument is omitted, the comma may also be omitted.
- Arguments that can be repeated one or more times are followed by an ellipsis…
- A vertical bar (|) in syntax means to choose between the arguments on each side of the bar.
- Data types are **boldface**. The data type in parentheses at the end of an argument description (for example, (**n**)) documents how the argument will be treated within the routine. An **a** represents alpha, a **d** represents decimal or implied-decimal, an **i** represents integer, and an **n** represents numeric (which means the type can be **d** or **i**).
- This grid indicates on which platforms and in which environments a routine, statement, etc., is supported: in traditional Synergy on Windows (WT), in Synergy .NET on Windows (WN), on UNIX (U), or on OpenVMS (V). By "supported" we mean that the item performs a useful function on that platform or environment. For example, an unsupported routine may cause a compiler error or it may just not do anything.

WT	WN	U	V

WIN

Items or discussions that pertain only to a specific operating system or environment are called out with the name of the operating system.

- To “enter” data means to type it (or highlight it, in the case of a selection window entry) and then press ENTER. (“ENTER” refers to either the ENTER key or the RETURN key, depending on your keyboard.)
- When you are instructed to “select an entry from the menu,” either press the shortcut for that entry, or pull down the menu, highlight the entry, and press ENTER.
- To “exit the window” on Windows, click the OK button or press the Exit shortcut. To exit a window on UNIX or OpenVMS, press the Exit shortcut or press ENTER from the last field, and then ENTER again from the **Make changes or press RETURN to complete input** message.

The screens depicted in this manual may not match your screens exactly. Our screens reflect the way ReportWriter looks when it is distributed. If you or your distributor have customized your version in any way, you may notice some discrepancies.

Other resources

- The ReportWriter release notes, **REL_RPT.TXT**
- *Repository User’s Guide*

Product support information

If you cannot find the information you need in this manual or in the resources listed above, you can reach the Synergy/DE Developer Support department at the following numbers:

800.366.3472 (in the U.S. and Canada)
916.635.7300 (in all other locations)

To learn about your Developer Support options, contact your Synergy/DE account manager at one of the above numbers.

Before you contact us, make sure you have the following information:

- The version of the Synergy/DE product(s) you are running
- The name and version of the operating system you are running
- The hardware platform you are using
- The error mnemonic and any associated error text (if you need help with a Synergy/DE error)
- The statement at which the error occurred
- The exact steps that preceded the problem
- What changed (for example, code, data, hardware) before this problem occurred
- Whether the problem happens every time and whether it is reproducible in a small test program
- Whether your program terminates with a traceback, or whether you are trapping and interpreting the error

Synergex Professional Services Group

If you would like assistance implementing new technology or would like to bring in additional experienced resources to complete a project or customize a solution, Synergex® Professional Services Group (PSG) can help. PSG provides comprehensive technical training and consulting services to help you take advantage of Synergex's current and emerging technologies. For information and pricing, contact your Synergy/DE account manager at 800.366.3472 (in the U.S. and Canada) or 916.635.7300.

Comments and suggestions

We welcome your comments and suggestions for improving this manual. Send your comments, suggestions, and queries, as well as any errors or omissions you've discovered, to **doc@synergex.com**.

Part 1: User's Guide

1

Welcome to ReportWriter

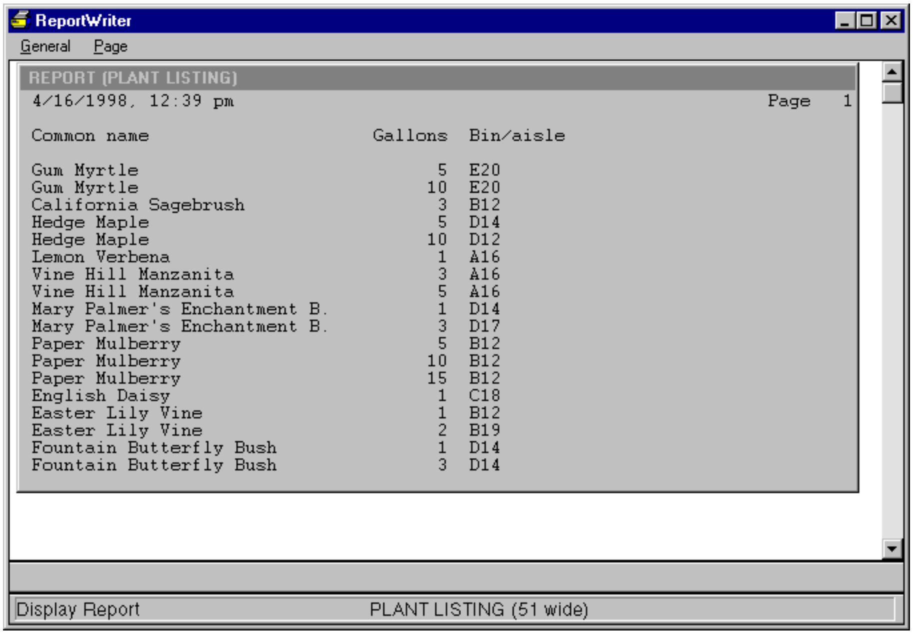
Before You Begin Using ReportWriter

If you're a new user of ReportWriter, we recommend that you read the remaining sections of this chapter before you begin using ReportWriter. These sections define the basic terminology used throughout this manual. They also provide the basic information you need to move around in ReportWriter, including how to run ReportWriter, select entries from the menu, display online help, enter data, edit a field, abandon your changes, select an entry from a list or selection window, and exit a function or ReportWriter.

We also recommend that you work through the tutorial in chapter 2, which takes you through the creation of several simple reports.

What Is ReportWriter?

ReportWriter creates columnar, ad-hoc reports that are organized according to your specifications. You not only specify which columns will appear in your report, but you also determine how the data will be arranged and sorted.



```
REPORT (PLANT LISTING)
 4/16/1998, 12:39 pm                                                  Page    1

 Common name                      Gallons  Bin/aisle

 Gum Myrtle                             5  E20
 Gum Myrtle                            10  E20
 California Sagebrush                   3  B12
 Hedge Maple                            5  D14
 Hedge Maple                           10  D12
 Lemon Verbena                          1  A16
 Vine Hill Manzanita                    3  A16
 Vine Hill Manzanita                    5  A16
 Mary Palmer's Enchantment B.           1  D14
 Mary Palmer's Enchantment B.           3  D17
 Paper Mulberry                         5  B12
 Paper Mulberry                        10  B12
 Paper Mulberry                        15  B12
 English Daisy                          1  C18
 Easter Lily Vine                       1  B12
 Easter Lily Vine                       2  B19
 Fountain Butterfly Bush                1  D14
 Fountain Butterfly Bush                3  D14
```

Figure 1-1. Sample report generated to the screen.

When defining a new report, you must first tell ReportWriter where to get its data. Then you must decide what information will actually be included in your report, by selecting the data fields that contain that information. (A field is a unit of information, such as an address, a social security number, or a first name.)

ReportWriter shows you the basic layout of your report as you select each field. You can easily change the order and position of the fields you have selected, as well as their column headers and the way they are displayed.

Powerful selection operations enable you to display only the information that you want in your report. For example, you may want a report that contains only customers in California who have more than five employees, or patients under 12 years old who have not been examined for a year.

You can sort your report on up to 10 categories. (The report in figure 1-1, for example, is sorted according to gross profit, in descending order.) You might want to create a report of all of your customers sorted by state, sales representative, and dollar volume purchases. For each sort category, you can also specify that a page break and subtotal will occur when the value in that category changes, to break your report into easily readable blocks of data.

In addition to the existing data fields you select, ReportWriter enables you to create your own *temporary fields* for use in your report. You can define five types of temporary fields.

- A *text field* contains a text constant (for example, an asterisk or the words “past due”), which can be printed when necessary.
- A *calculation field* defines a mathematical computation, which is evaluated for each set of data.
- A *question field* contains information supplied by the person running the report in response to a question or prompt (for example, you might prompt for starting and ending dates so that the report can include only transactions that occurred between those dates).
- An *environment field* obtains static data from the system (for example, the current date) when the report is run.
- A *subtotal access field* gives you access to a field’s current total at any break point in the report.

Once created, temporary fields can be printed, sorted on, used in other temporary fields, or used in a selection operation.

ReportWriter enables you to specify the header and footer that will be printed on each page of the report, as well as the header and footer for the entire report. In addition, you can include the date and page number at the top of each page, specify whether your report will merely summarize the data or include detailed information, and change the page size or the number of blank lines between report entries.

At any time, without leaving the function you’re in, you can view the current design of the report you’re defining. A ruler at the top of the screen lets you verify that your text is placed exactly where you want it.

Once your report is complete, you can display it on the screen, print it, or place the data in a file. ReportWriter stores your report definition, so you can run the same report again or modify it to create a similar report.

Essential ReportWriter Concepts

Important terminology

Access key. An actual key in the data file.

Byte. The smallest area of memory addressable by the ReportWriter. One byte contains one character.

Field. A named area of memory used to store a specific type of data.

File. An area of the disk where groups of similar data (records) are stored.

Foreign key. A field in a file that is a key on another file.

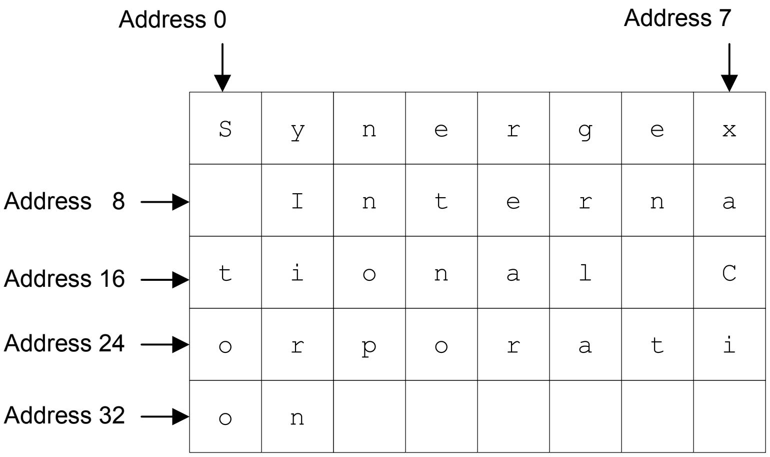
Key. A field on the file that may be used for extremely fast access.

Record. A unit of data consisting of one or more fields.

How your data is stored

Computers store information in single-character units called *bytes*. A computer's main memory consists of many millions of bytes, starting with address zero, up to address *n* million. A byte can contain an alphabetical, punctuation, or numeric character.

You can think of main memory in terms of a pigeon-hole address system:



	S	y	n	e	r	g	e	x
Address 8 →		I	n	t	e	r	n	a
Address 16 →	t	i	o	n	a	l		C
Address 24 →	o	r	p	o	r	a	t	i
Address 32 →	o	n						

Figure 1-2. A graphical representation of main memory.

Keeping track of what's in memory

All programs, including ReportWriter, reserve several thousand bytes to store the information they need. Areas of reserved memory are assigned to contain information items that the program will need. These blocks of data are called *fields*.

A field is referred to by name rather than by address. The program keeps track of the address in the pigeon-hole system at which a field is stored. In the above example, the field SUPPLIER_NAME happens to be stored starting at address 0. It contains the data "Synergex International Corporation."

Each field has two attributes, which are designated by your developer.

- The *data type*, which specifies whether the field is intended to contain textual data (such as letters, numbers, and punctuation characters) or numeric values (which may be used in arithmetic calculations). Textual data is specified as **a**, for alphabetical. Numeric data is specified as **d**, for decimal.
- The *field size*, which is the number of bytes, or characters, occupied by the field

These two attributes are declared together, to designate an overall field type. For example:

d10 The field may store up to 10 numeric digits and be used for arithmetic calculations.

a30 The field may store up to 30 characters of any type. Even if the field happens to contain numbers, it cannot be used in arithmetic calculations.

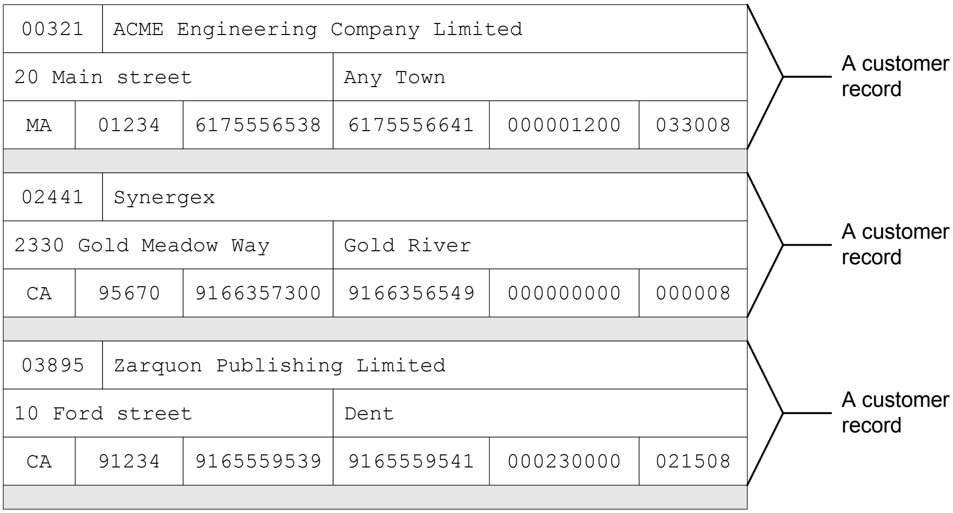
Storing data on disk

The data accessed by ReportWriter was entered into the computer through a program and stored in the memory system described above. Because the computer's main memory is not large enough to contain all of the data stored by your system, data is stored on magnetic disks, which can store billions of bytes of information.

Information stored on disks is arranged into named areas called *files*. Groups of similar data are gathered together and stored in a file. A common example of a file is a customer file, which contains customer data. Because the customer file may contain data about many different customers, it is subdivided into *records*. The file usually contains one record for each customer.

Each record can be viewed in a similar manner to the memory model in figure 1-2. A record is an area of bytes subdivided into fields that are used to store the information about a customer. The fields are named so that you can refer to them easily. All records in a particular file contain exactly the same field layout and are the same size. The size of a record is the number of bytes occupied by all of its fields.

The customer file



```
00321 | ACME Engineering Company Limited
20 Main street          | Any Town
MA | 01234 | 6175556538 | 6175556641 | 000001200 | 033008

02441 | Synergex
2330 Gold Meadow Way    | Gold River
CA | 95670 | 9166357300 | 9166356549 | 000000000 | 000008

03895 | Zarquon Publishing Limited
10 Ford street          | Dent
CA | 91234 | 9165559539 | 9165559541 | 000230000 | 021508
```

Figure 1-3. Storing records and fields.

Each box in figure 1-3 represents a field. You can see the repeating nature of records within a file and how all the records contain fields. Such records and fields are the basic entities on which ReportWriter acts. In our example, the fields are declared as follows:

Name	Field type
CUSTOMER_NUM	a5
CUSTOMER_NAME	a35
STREET_ADDRESS	a20
CITY	a20
STATE	a2
ZIP_CODE	a5
TEL_NUMBER	a10
FAX_NUMBER	a10
OUTSTANDING_BALANCE	d9
DUE_DATE	d6

Accessing data from disk

To speed up access to files, your developer can designate some fields as *keys*. If a field is a key, the system can search for a match on that field quickly, without having to scan the entire file. For example, in the customer file, the CUSTOMER_NUM field would be a key, so that any desired customer record could be accessed quickly.

Keyed access can start at any value, and it continues in ascending or descending order (depending on how the key was defined).

How Does ReportWriter Work?

Retrieving data with ReportWriter

ReportWriter enables you to select which file you want to include data from, and then which fields from that file you want to see on your report. Each field you select will be a column in the final report. ReportWriter scans the file or files you select, reading every record, to extract and print the required data.

ReportWriter's most important feature is its ability to select from a file *only* those records that conform to a set of criteria chosen by you. You can tell ReportWriter to include only data from records in which certain fields are equal to (or less than, or greater than) a specified value.

If the customer file contains 20,000 records and you want to see the customers in California, you don't want a complete customer report. Instead, using the Selections to make menu entry, you can tell ReportWriter only to include records in which the STATE field equals "CA" (the abbreviation for California).

Creating your own fields

The *fields* in the files you select as the data source for your report always exist. As ReportWriter scans the file, reading each record, the *values contained in the fields* change.

If you want to perform a calculation with some of the fields, or if you want ReportWriter to get a value from the person running the report, you can create a temporary field to hold this value. You must give the field a name and decide what sort of data and how many characters the field must contain. ReportWriter then reserves some of the computer's memory to store your field. The Fields to create menu entry enables you to create these temporary fields.

How files relate to one another

Remember how keys can speed up access to files? Some fields are keys in more than one file. For example, if you had a customer file and a file of contacts who worked for that customer, both files would probably use the CUSTOMER_NUM as a key. This type of key is called an *access key.* If two files have the same key, there is a link between the two files, as illustrated in figure 1-4.

But there's another kind of key as well. Some files contain fields that are keys in *other* files, even though they're not keys in the original file. An order file, for example, would contain the CUSTOMER_NUM of the customer who made the purchase. Thus, there is a link from the order file to the customer file, as shown in figure 1-5. If ReportWriter reads an order record, it can very easily read the customer record for the purchaser. The customer number on the order file is called a *foreign key.*

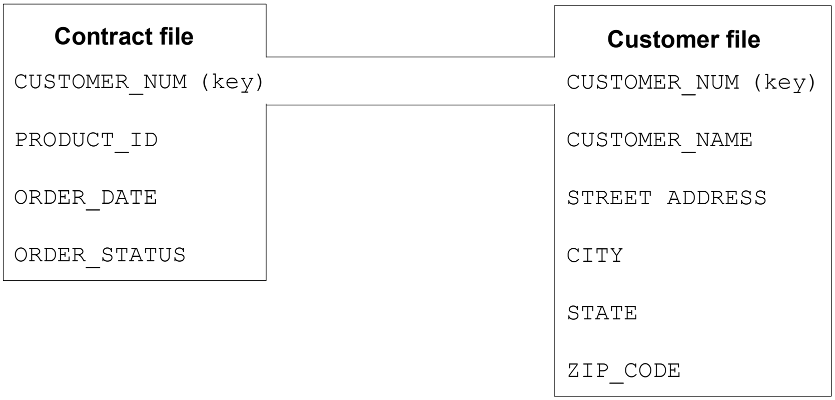


Figure 1-4. Two files linked by an access key.

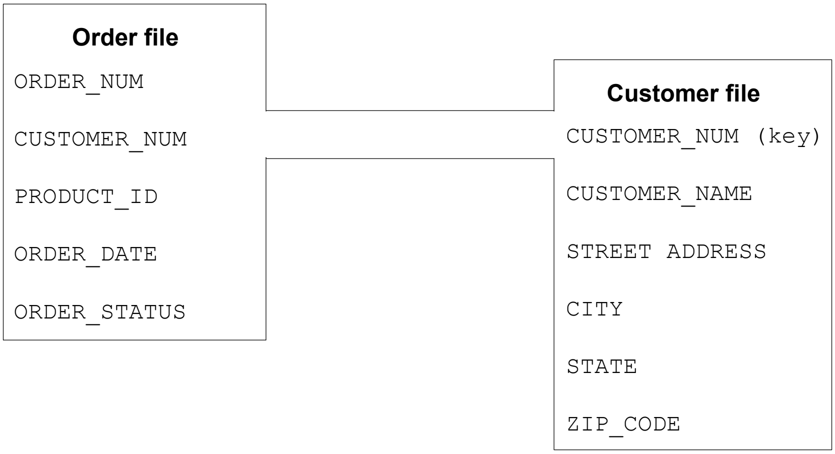


Figure 1-5. Two files linked by a foreign key.

It’s not really important that you remember the distinction between access keys and foreign keys. Simply keep in mind that when you select a file for inclusion in a report, only files related to that file through access keys or foreign keys are also available for inclusion in your report.

Running ReportWriter

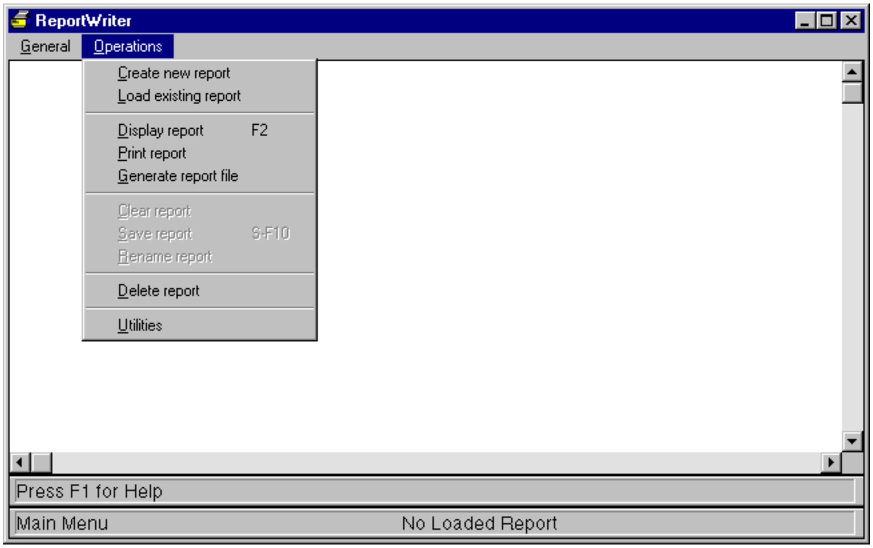
On...	Do this...
Windows	Open a Command Prompt window and enter `dbr RPT:rpt`

UNIX	At the command prompt, set the necessary environment variables with
	`setsde`
	and then enter
	`dbr RPT:rpt`

OpenVMS	At the command prompt enter `run RPT:rpt`

What’s on your screen?

The main menu is displayed when you first start ReportWriter. (See figure 1-6.)



Press F1 for Help

Main Menu No Loaded Report

Figure 1-6. The ReportWriter main menu.

At the very bottom of the screen is the *footer*. It contains the name of the current menu or function, as well as the name of any report that is currently loaded.

Just above the footer is the *information line*. It contains information for the user, such as whether the Find entry function is operating in forward or reverse mode, and brief help information, when ReportWriter is processing input windows.

Getting information about ReportWriter

To display information about your version of ReportWriter, select About from the General menu. The current ReportWriter version number is displayed, along with the date on which ReportWriter was compiled and the versions of Synergy DBL and UI Toolkit under which ReportWriter is running. To exit the information window, press ENTER.

Planning Your Report

Before you actually create your report, you should take a few moments to plan what you're going to do. You need to decide what information you want to include, and what—roughly—you want it to look like. (See "Defining a Simple Report" on page 1-15 for the steps you need to follow to actually create the report.)

What information do you want to include?

Think about what data you want to appear in your report. For example, if the purpose of your report is to itemize and then total the amount of money your customers owe you, you will want to include, at a minimum, some way to identify the customer (such as the customer name or number) and the customer's outstanding balance. Perhaps you also want to include the customer's telephone number and the name of your contact, so you can call and remind them to pay you if payment is overdue.

Where is this information stored?

When you define a new report, you first need to select the files that you want ReportWriter to read. This means you need to know (or figure out) which files contain the information you want to include.

In our example, let's say the customer name, outstanding balance, and telephone number are all stored in the customer file. The customer contact name is stored in the contact file. Therefore, in the Files to read function of ReportWriter, you would need to select both the customer file and the contact file.

Which file should you select first?

The order in which you select files is important because that's the order in which ReportWriter reads them.

In our case, the link between the customer file and the contact file is the customer number. If you select the contact file first, and the contact file contains more than one contact for each customer, your report will contain the same customer more than once. However, if you select the customer file first, ReportWriter will read each customer number and then look for the associated entry in the contact file. In our example, therefore, you would select the customer file first. (If you were defining a report that listed all of your contacts at customer sites, you would want to select the contact file first.)

What if you want to include information that doesn't exist in a file?

OK, so now you've decided which files you want ReportWriter to read:

- Customer file
- Contact file

and what information you want to include:

- Customer name
- Outstanding balance

- Telephone number
- Contact name
- Whether the balance is past due

How do we know if the balance is past due? We could include the due date as a field in our report and then compare it to today's date, but why not let ReportWriter do that work for us? We can create a temporary calculation field that subtracts the due date from today's date and gives us the number of days (if any) that the bill is past due. We could also create a temporary text field to flag with an asterisk customers whose payments are more than 30 days past due.

ReportWriter makes the temporary field information available in a *temporary file*, so once you create a temporary field, you can select it for inclusion in your report just like any other field.

- **Which records do you want to include?**

Do you want to include all of your data records in your report, or just a subsection? For example, in our sample report, should we include all customers, or just customers who have an outstanding balance? Since the purpose of this report is to track the amount of money your customers owe you, there's no point in including customers who don't owe any money.

When you define your report, you can use selection criteria to weed out the records you don't want. To include only customers who don't have a zero balance, you'd set up a selection criterion for an outstanding balance greater than 0.

How do you want your report to look?

You also need to decide what you want your report to look like.

- **In what order do you want the fields to appear?**

What data should go in the first column of your report? What should the second column contain?

Fields are displayed in the order you select them when you define your report. For example, if you want the customer name to appear in the first column, select the CUSTOMER_NAME field first. You can always change the field order later, but it helps to have a mental picture of your intended report layout before you begin.

- **Do you want any totals?**

If you want the numbers in any column of your report to be totaled, you need to tell ReportWriter to total that field. In our example, you'll want to total the outstanding balance field. ReportWriter will then print a total at the bottom of the outstanding balance column.

Defining a Simple Report

This section describes the basic steps you might follow to define a report in ReportWriter. Although you have some latitude as to the order in which you perform these steps, we recommend the order shown here.

1. **Create a new report.**
   - Select Operations > Create new report.
   - Enter a report a name at the prompt.

2. **Select the files to read.**

   Decide which files contain the information you want to include in the report. If you are using more than one file, decide what order you want ReportWriter to read them in, and then select them in that order.

   - Select Design > Files to read.
   - Choose a file from the list.
   - To select additional files to read, select File functions > Add file.
   - See chapter 5 for more information.

3. **Select the fields to print.**

   Decide what information (which fields) you want to print on the report, and whether you want to total the data for any of them.

   - Select Design > Fields to print.
   - Select the fields to include from the list. If the report reads multiple files, the available fields list displays fields from the first file in the report. Press TAB to access the fields for the other files.
   - If you want to total the data for a field, select the field and select Field functions > Total field.
   - See chapter 7 for more information.

4. **Create temporary fields.**

   Decide whether you need to include information, such as the result of a calculation, that doesn’t already exist in a field. If so, you can use a temporary field.

   - Select Design > Fields to create.
   - Choose the type of field you want to create from the list: calculation, question, text, environment, subtotal access.
   - Fill in data for the field definition.
   - Select the field to print, as described above.
   - See chapter 8 for more information.

5. **Define selection criteria.**

   Decide whether you want your report to include all records or just those that meet specific criteria.

   - Select Design > Selections to make.
   - Enter the criterion definition.
   - Repeat this step for additional selection criteria.
   - See chapter 9 for more information.

6. **Specify the fields to sort on and how you want the report to break.**

   Determine how you want your report organized. You can sort on any field, including temporary fields; the sort field does not have to be selected for printing. You can set the report to break to a new page when the value in the sort field changes, or to break without going to a new page.

   - Select Design > Fields to sort.
   - Select one or more fields from the list of available fields.
   - If you want the report to start a new page when the contents of a sort field change, highlight the field and select Sort functions > Change break status. The Flags column displays a ‘B’.
   - If you want the report to break without starting a new page, select Sort functions > Change break status twice. The Flags column displays a ‘b’.
   - See chapter 10 for more information.

7. **Add a header/footer.**

   The default report header is simply the name of your report. You can display other information in the header, as well as create a footer.

   - Select Design > Header/footer layout.
   - Select Report header or Report footer from the Layout functions menu.
   - Select Edit > Insert/overstrike mode so that you can enter text.
   - Enter text as you want it to appear in the header/footer.
   - Press CTRL+F to return to command mode so you can access menu options.
   - See chapter 7 for more information.

8. **Define miscellaneous report settings.**

   Miscellaneous report settings include the number of lines per page, blank lines between records or sections, printing a detailed report vs. a summary, and including descriptions for total fields.

   - Select Design > Miscellaneous and enter data in the fields.
   - See chapter 7 for more information.

9. **View the report on screen.**
   - Select Operations > Display report.
   - Press ENTER to display the next page.
   - To exit the display, select General > Exit.
   - See chapter 4 for more information.

Using the ReportWriter Interface

Making a menu selection (UNIX and OpenVMS)

You can select a menu entry using the arrow keys, quick-select characters, or shortcuts. To use the arrow keys or quick-select characters, you must first activate the menu bar by pressing the process-menu key, CTRL+P. The menu bar is deactivated after you select a menu entry. You can also deactivate it by pressing the process-menu key again.

- **Arrow keys:** Use the LEFT and RIGHT ARROW keys to move across the activated menu bar to select a menu. When you move to a menu, it drops down, displaying the entries. Use the UP and DOWN ARROW keys to move among the menu entries. Press ENTER to select a highlighted entry.
- **Quick-select characters:** A quick-select character is a single character that accesses a menu entry. When a menu is dropped down, type the quick-select character to highlight the menu entry and select it. If the quick-select character is not unique in the menu, it simply highlights the entry, and you will need to press ENTER to select the highlighted entry.
- **Shortcut keys:** A shortcut is a key or key sequence that is associated with a menu entry, and which enables you to execute a function directly without selecting it from a menu. (The exception to this rule is arrow keys: arrow keys can be used as shortcuts only when the menu is *not* enabled.) The shortcut keys are listed to the right of the menu entry and vary depending on the type of terminal. Not all menu entries have shortcuts. See "Appendix E: Distributed Shortcuts" for a list of the most common shortcuts as they are provided with your original ReportWriter distribution. (Note that shortcuts can be reassigned by your system administrator.)

Entering data

In this manual we refer to the ENTER or RETURN key as ENTER. After you have finished typing data for a particular field, press ENTER to enter it.

⚠ Don't forget to press ENTER after you've typed data in the last field of an input window! If you exit the window without pressing ENTER, the data you just typed in that field will be ignored (unless the field is a text field).

ReportWriter converts most of the data you enter into uppercase letters. Descriptions or headings can be entered in either uppercase or lowercase.

Getting field-level help

For each input field, the information line displays a brief help message telling you what data to enter. For more detailed information, select General > Help to display a help window for the field. To close the help window, press ENTER.

Entering default data

In some ReportWriter fields, the program enters a default value for you. You can override the default by typing your own data, or you can accept the default by pressing ENTER. For optional fields, you can override the default and leave the field blank by pressing the spacebar or the BACKSPACE key and then pressing ENTER.

Skipping a field

To skip optional fields, simply press ENTER. To remove a value from an optional field and leave it blank, press the spacebar or the BACKSPACE key and then press ENTER.

Moving between fields

On Windows, press the TAB and SHIFT+TAB keys to move between fields in a window. You can also use the mouse to move directly to a specific field.

On UNIX and OpenVMS, you can use the Previous Field and Next Field entries on the Input menu to move between fields.

Editing a field (UNIX and OpenVMS)

Changing the direction of cursor movement

When you begin entering text in a field, the cursor direction is set to forward. To toggle between moving forward and backward, use the Direction option (on the Edit menu). This setting does not actually change the direction in which the cursor moves as you type; rather, it is honored by some of the options on the Edit menu. (Refer to the tables below for details on which options honor cursor direction.) The current direction is displayed on the right side of the information line.

Switching between insert and overstrike modes

When you begin entering text in a field, the editor is in insert mode. Any data you enter is placed at the current cursor position, and the existing data is moved to the right as you type. To change to overstrike mode, so that the data at the cursor is *replaced* with the data you type, use the Insert/Overstrike option (on the Edit menu). The current mode is displayed on the right side of the information line.

Editing a multi-line text field

When you start to type in a multi-line text field, the Edit menu appears on the menu bar. See the table below for an explanation of the available options on this menu.

To edit existing text in a multi-line field, use the right arrow key to move the cursor before you begin typing; the Edit menu will display on the menu bar, and then you can use the up and down arrow keys to move to the line you need to edit. (When you first access a multi-line text field that already has text in it, the entire field is selected. If you simply start typing, all existing text will be deleted. If this happens, you can abandon changes to recover the text.)

To...	From the Edit menu, select...
Move up one line	Up One Line. The cursor stays in the same column position.
Move down one line	Down One Line. The cursor stays in the same column position.
Move left one character	Left One Character.
Move right one character	Right One Character.
Move to the next word	Move One Word. Cursor direction must be set to **Forward**.
Move to first character of the current word	Move One Word. Cursor direction must be set to **Reverse**. If the cursor is already on the first character of the word, it will move to the first character of the previous word.
Move to the beginning of the line	Beginning of Line. If the cursor is already positioned at the beginning of a line, it will move to the beginning of the previous line.
Move to the end of the line	End of Line. This moves the cursor past the last character in the current line. If the cursor is already positioned after the last character on the line, movement depends on the cursor direction setting. ▶ Forward: the cursor moves to the end of the next line. ▶ Reverse: the cursor moves to the end of the previous line of text.
Re-wrap text (join lines)	Join Lines. This function rewraps the text from the cursor location to the end of the paragraph.

Deleting the data in a field

You can erase the contents of the current field by selecting Input > Clear Field.

To delete...	From the Edit menu select...
The current character	Delete Character.
From the cursor to the beginning of the next word	Delete Word. Cursor direction must be set to **Forward**. Note that this deletes all characters up to the beginning of the next word, including the current character and the space before the next word.
From the cursor to the beginning of the current word	Delete word. Cursor direction must be set to **Reverse**. All characters to the left of the cursor in the current word are deleted.

To delete…	From the Edit menu select…
From the cursor to the end of the line	Delete to End of Line. The end-of-line character is not deleted.
From the cursor to the end of the line (including the end-of-line character)	Delete Line. Cursor direction must be set to **Forward**. If the next line contains text, that text moves up to the current line.
From the cursor to the beginning of the line	Delete Line. Cursor direction must be set to **Reverse**. Note that if the cursor is positioned on the first character of a line, this deletes the previous line of text, including the end-of-line character, and the remaining text moves up one line.

Abandoning changes

To reset a field to its original value, select Input > Reset Field (UNIX and OpenVMS).

To abandon changes to all fields in a window, select General > Abandon. In a tabbed dialog, this abandons all changes to all tabs.

On Windows, you can also click on the Cancel button to abandon changes for the current window or for all tabs in a tabbed dialog.

If you get an error message…

See "Appendix C: Error Messages" for a list of error messages that ReportWriter may generate, along with explanations of the problems that may have caused the errors.

Using lists

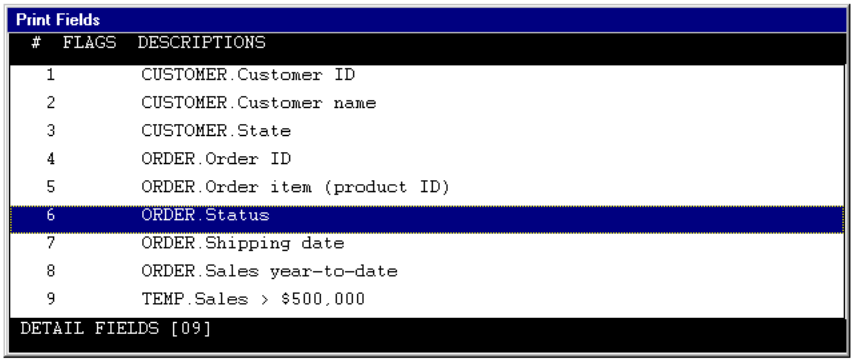
ReportWriter uses two types of lists, modifiable and non-modifiable.

By default, lists display descriptions. To display entry names instead, select List > Toggle view. In a list of formats, Toggle view toggles between format strings and format names.

Modifiable lists

Modifiable lists contain entries that you can edit. You can also add, delete, and possibly move entries in the list. On some modifiable lists, you can scroll horizontally using the options on the List menu. The Print Fields list in figure 1-7 is a modifiable list.

To edit an entry in a modifiable list, highlight the entry and press ENTER. On Windows, you can also double-click an entry to modify it. To exit a modifiable list, press the Exit shortcut.

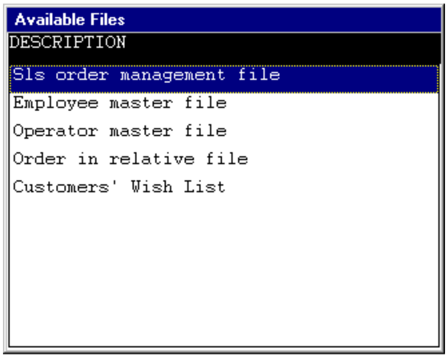


```
Print Fields
 #  FLAGS  DESCRIPTIONS
 1         CUSTOMER.Customer ID
 2         CUSTOMER.Customer name
 3         CUSTOMER.State
 4         ORDER.Order ID
 5         ORDER.Order item (product ID)
 6         ORDER.Status
 7         ORDER.Shipping date
 8         ORDER.Sales year-to-date
 9         TEMP.Sales > $500,000
DETAIL FIELDS [09]
```

Figure 1-7. Modifiable list in ReportWriter.

Non-modifiable lists

Non-modifiable lists contain entries that you can select, but cannot change, add, or delete. The Available Files list in figure 1-8 is a non-modifiable list.



```
Available Files
DESCRIPTION
Sls order management file
Employee master file
Operator master file
Order in relative file
Customers' Wish List
```

Figure 1-8. Non-modifiable list in ReportWriter.

To select an entry in a non-modifiable list, highlight the entry and press ENTER. On Windows, you can also double-click an entry to select it. To exit a non-modifiable list without making a selection, press the Exit shortcut.

Searching for a list entry

In both modifiable and non-modifiable lists, select Find > Find Entry to search for a list entry. At the prompt, enter the name or partial name of the entry you wish to find. If ReportWriter finds a match, it highlights the first matching entry in the list.

The default search direction is forward. To change the direction of the search, select Find > Toggle Direction. The search direction is displayed on the right side of the information line.

Making a selection from a selection window

Like a non-modifiable list, a selection window (sometimes referred to as a drop-down list) contains a list of items from which you can choose.

To select an item on UNIX and OpenVMS, press any key to display the list, use the arrow keys to highlight an item, and then press ENTER. On all systems, you can type the first letter of the item to highlight it, and then press ENTER. On Windows, you can also select an item with the mouse.

To redisplay a selection window on UNIX and OpenVMS once you've made a selection, press any non-shortcut key at the prompt. If the key you press is the first letter of one of the selection window entries, that entry is highlighted when the selection window is displayed.

Exiting the current function

To save your changes and exit the current input window or list, select General > Exit. You are returned to the previous window or menu. Be sure to press ENTER in the last field of the input window before exiting, so that the data in that field is entered.

On Windows, you can click the OK button to save your changes and exit the current input window.

Exiting ReportWriter

There are a couple of ways to exit ReportWriter.

- Press the Exit shortcut to back out of each function until no report is loaded and the Operations menu is displayed. Then, press the Exit shortcut again or select General > Quit.
- Select General > Quit to exit ReportWriter immediately—regardless of what function you're currently in.

If you try to exit ReportWriter without saving your work, you will be prompted to save the report.

2

Using ReportWriter: A Tutorial

Getting Started

You're about to begin using Synergy/DE ReportWriter, a tool that enables you to create customized reports whenever you need them. ReportWriter is easy to learn and use, without sacrificing sophistication for ease-of-use. You can create simple lists of data with ReportWriter, or you can create reports that handle complex calculations and conditions for printing your report data.

Your ReportWriter distribution includes three demonstration data files for a fictional plant nursery:

- **arcust**—Customer file
- **oorder**—Open order file
- **plitem**—Plant item file

Once you've installed ReportWriter, you can use these files to experiment with ReportWriter. ReportWriter enables you to combine data from up to 50 files in one report, but to simplify this tutorial, we're using only three files for our exercises.

This tutorial walks you through writing five reports, and should take about 50 minutes. Our instructions assume you will complete all five exercises without exiting ReportWriter.

In the first exercise, you'll create a simple list-style report. In the second exercise, you'll sort the data and use a text field to format subtotals for the report. You'll learn how to create and use a question field to prompt the user for input in the third exercise. In the fourth exercise, you'll combine data from two files. And you'll create and process a calculation field in the fifth exercise.

Of course, not every feature of ReportWriter is covered in this tutorial. Once you work through the tutorial, we suggest you look through the remainder of this manual and try any of the other features that catch your eye.

Running the ReportWriter tutorial

Once ReportWriter is installed, go to the **rpt** directory, and type the command shown in the table below. This command sets up the environment for the tutorial and starts ReportWriter.

On...	At the command line type...
Windows and UNIX	`tutorial`
OpenVMS	`@tutorial`

If you are running on UNIX or OpenVMS, your screens will differ somewhat from the screens shown in this manual. In addition, if you had the environment variables RPSMFIL, RPSTFIL, and RPTRFIL set prior to running the tutorial, you will need to reset them when you're ready to run ReportWriter with your own data.

Standard menu entries

To access the ReportWriter menu when no menu is pulled down, press the process-menu key, which is ALT on Windows and CTRL+P on all other systems. The process-menu key is a toggle that activates and deactivates the menu bar. You can activate the menu bar at any time to browse through your choices.

When the menu bar is activated (a menu is pulled down), you can use the LEFT ARROW and RIGHT ARROW keys to pull down the desired menu and the UP ARROW and DOWN ARROW keys to highlight the desired menu entry.

When the tutorial tells you to “select” a menu entry, it means to pull down the menu, highlight the entry, and press ENTER. You can also press the shortcut for the entry, which is listed to the right of the entry in the menu. When the tutorial tells you to “enter” data, it means to type the specified information and press ENTER to complete your input.

These are the menu entries you’ll probably use most often:

Help	Obtains online help at any time.
Display report	Displays the current report while you’re designing it.
Exit	Exits the current function.
Abandon	Cancels input to the current window.

Note the shortcuts for these entries on your system.

Exercise One: Creating a Report

Approximate time: 9 minutes

You'll create a new report listing all plants in the database. The report will include each plant's common name, size, and location. You'll learn how to modify the default field headings and change the order in which fields appear in the report.

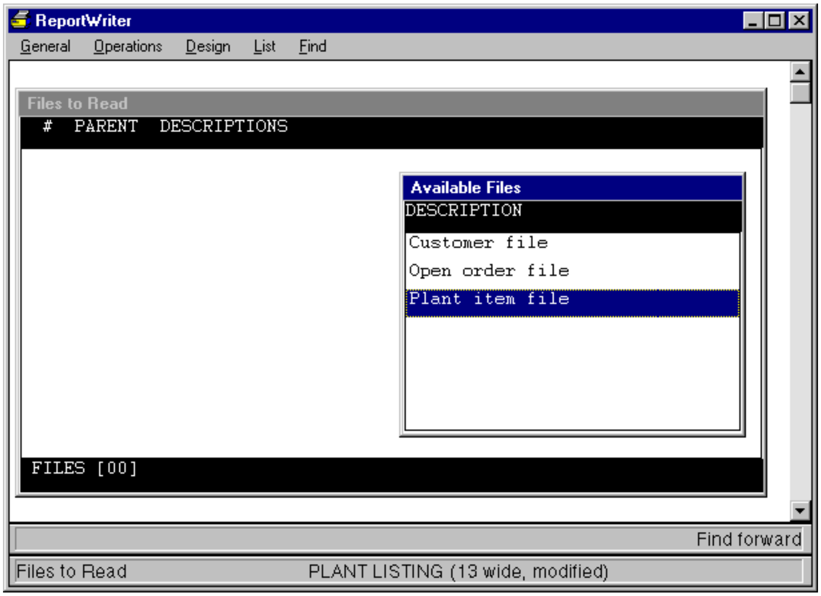
Creating a new report

1. From the Operations menu, select Create new report. You are prompted to enter the report name.
2. Type PLANT LISTING.
3. Press ENTER (or click OK on Windows). After you enter the name of the report, the Design menu is pulled down.

Selecting the files to read

The Files to read function enables you to choose the files from which your report will get its data.

1. From the Design menu, select Files to read. A list of available files is displayed, as shown in figure 2-1. For this report, we'll use the plant item file.



```
 #  PARENT  DESCRIPTIONS
```

```
DESCRIPTION
Customer file
Open order file
Plant item file
```

```
FILES [00]
```

Files to Read PLANT LISTING (13 wide, modified)

Figure 2-1. Selecting files to read from the Available Files list.

2. Select Plant item file from the Available Files list.
3. Press the Exit shortcut.

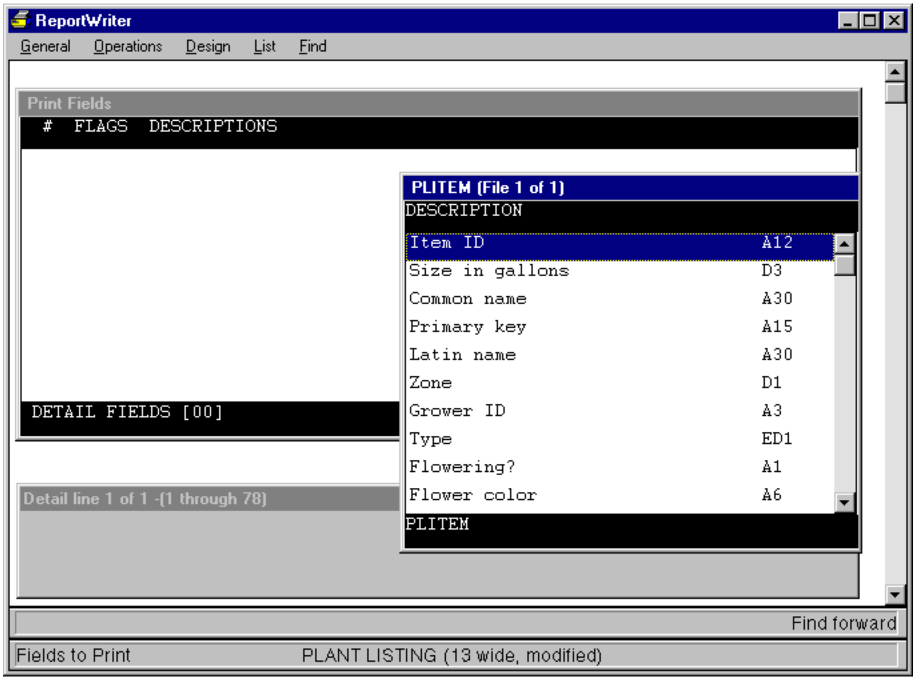
Selecting fields to print

Now we need to select the fields in the plant item file that we want to include in our report.

1. From the Design menu, select Fields to print.

You'll see two lists and a window. The list of available fields shows the fields available for printing in the plant item file. The Print Fields list shows the fields you've selected to include in the report; it's empty until you select a field. The Detail line window at the bottom of the screen shows how your report will look, based on the fields you've selected. (See figure 2-2.)

You can create up to 10 detail lines for each record in your report, and each line can be up to 255 characters long. In this tutorial, we'll work with one line of detail per record (except in exercise 4), and we'll use just 78 characters for each line, so that the reports are easy to see on the screen.



```
 #   FLAGS   DESCRIPTIONS

DETAIL FIELDS [00]
```

PLITEM (File 1 of 1)

```
DESCRIPTION
Item ID                          A12
Size in gallons                  D3
Common name                      A30
Primary key                      A15
Latin name                       A30
Zone                             D1
Grower ID                        A3
Type                             ED1
Flowering?                       A1
Flower color                     A6
PLITEM
```

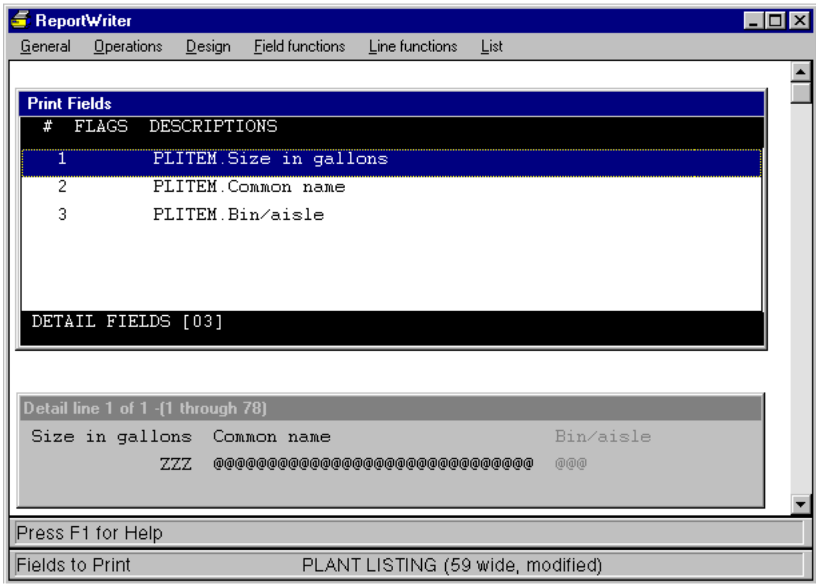
Fields to Print   PLANT LISTING (13 wide, modified)

Figure 2-2. Selecting fields to print from the PLITEM list.

Looking at the PLITEM fields list, you see the description of each field, the field data type (A for alpha, D for decimal or implied-decimal), and the field size. (If you want to see the field's name instead of its description, select Toggle view from the List menu or press the Toggle view shortcut.)

2. Select Size in gallons and Common name.
3. Press the Next page shortcut until "Bin/aisle" appears in the list.
4. Select Bin/aisle.
5. Press the Exit shortcut.

As you select each field, you'll see the field display in the Print Fields list and its heading and layout display in the Detail line window.



```
Print Fields
  #  FLAGS  DESCRIPTIONS
  1         PLITEM.Size in gallons
  2         PLITEM.Common name
  3         PLITEM.Bin/aisle

DETAIL FIELDS [03]
```

```
Detail line 1 of 1 -(1 through 78)
Size in gallons  Common name                     Bin/aisle
            ZZZ  @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@  @@@
```

```
Press F1 for Help
Fields to Print                PLANT LISTING (59 wide, modified)
```

Figure 2-3. The Print Fields list and the Detail line window.

Reordering fields

Now, your report layout should look like figure 2-3. The fields are displayed in the order in which you selected them. But it makes more sense to show the name of the plant first, and then its size, right? Let's change the order of the fields so they're in this order: Common name, Size in gallons, Bin/aisle.

1. In the Print Fields list, highlight PLITEM.Size in gallons.
2. Select Reorder fields from the Field functions menu.

You'll notice square brackets around the field. (Press the RIGHT ARROW key to see the right square bracket.) The brackets indicate the field is selected to move.

3. Use the DOWN ARROW key to move the field down one line, so that it's located after PLITEM.Common name.

4. Select Reorder fields again.

   You'll notice that the field's movement is also reflected in the Detail line window.

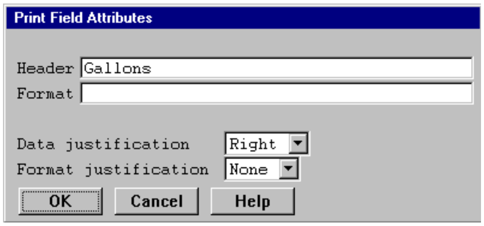
Changing a field heading

Now that the fields are in the correct order, let's change the "Size in gallons" heading to read "Gallons" instead.

1. PLITEM.Size in gallons should still be highlighted; if it's not, highlight it now.

2. Press ENTER to display the Print Field Attributes window. From here, you can change the header for this field in the report, the format of the field (used primarily with numeric data), the justification of the data (left, center, or right), and the justification of the format (none, left, or right).

3. Type "Gallons" in the Header field, as shown in figure 2-4. When you start typing, the field is automatically cleared and ready to accept new text.


Print Field Attributes

Header Gallons
Format

Data justification Right
Format justification None

Figure 2-4. Changing the field header in the Print Field Attributes window.

4. Press ENTER. Note that the heading has changed in the detail window at the bottom of your screen.

5. Press the Exit shortcut (or click OK on Windows).

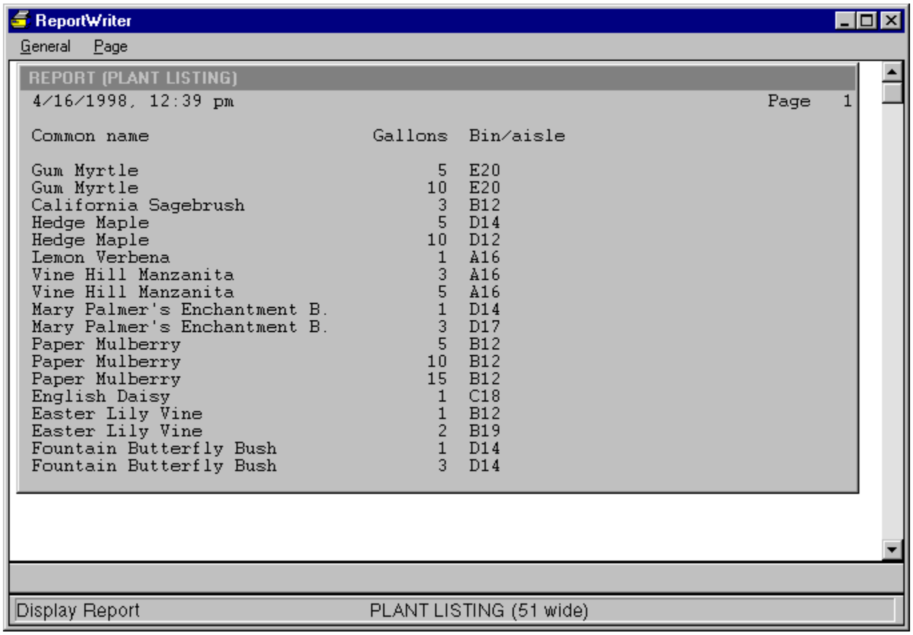
Displaying the report

Now let's display your first report.

1. Press the Display report shortcut, or select Display report from the Operations menu. The report header is displayed.

2. Press the Next page shortcut to view the report detail, as shown in figure 2-5. You can browse through the report using the Next page and Previous page shortcuts. Note that when you browse previously viewed pages, the Last page shortcut takes you to the last page that has been viewed.

3. When you're ready to return to the menu, press the Exit shortcut.



```
REPORT (PLANT LISTING)
 4/16/1998, 12:39 pm                                                    Page    1

 Common name                        Gallons  Bin/aisle

 Gum Myrtle                               5  E20
 Gum Myrtle                              10  E20
 California Sagebrush                     3  B12
 Hedge Maple                              5  D14
 Hedge Maple                             10  D12
 Lemon Verbena                            1  A16
 Vine Hill Manzanita                      3  A16
 Vine Hill Manzanita                      5  A16
 Mary Palmer's Enchantment B.             1  D14
 Mary Palmer's Enchantment B.             3  D17
 Paper Mulberry                           5  B12
 Paper Mulberry                          10  B12
 Paper Mulberry                          15  B12
 English Daisy                            1  C18
 Easter Lily Vine                         1  B12
 Easter Lily Vine                         2  B19
 Fountain Butterfly Bush                  1  D14
 Fountain Butterfly Bush                  3  D14

Display Report                     PLANT LISTING (51 wide)
```

Figure 2-5. Displaying the PLANT LISTING report.

Summary

What you learned...	For more information see...
How to create a new report.	Chapter 3, "Creating or Loading a Report"
How to select a file for the report to read.	"Selecting a File to Read" on page 5-2
How to select fields to be printed on the report.	"Selecting a Field to Print" on page 7-2
How to rearrange fields.	"Moving a field" on page 7-9
How to change a field heading.	"Modifying a field header" on page 7-6
How to view your report on screen.	"Displaying a Report on the Screen" on page 4-2

Exercise Two: Using Text Fields and Sorting

Approximate time: 6 minutes

In this exercise, we're going to work with sorting and break processing. In our first exercise, we didn't specify the order in which we wanted to print the records. When no sort field is specified, records are sorted based on the value of the primary key in the primary file. So in our report, the data was sorted by plant ID and size, the primary key in the **plitem** file. However, you can sort your report output on any field you choose.

Break processing is a special way of representing sorted data. Let's say, for example, that we're sorting a sales report by sales rep. If we also "break" on the sales rep field, we can do special processing when the sales rep data changes, such as printing subtotals by rep or printing additional information before or after the sales rep changes.

Now let's modify the existing report to sort the plant listing based on the zone in which the plant grows. This report will break on the zone and print subtotals showing the number of plants in each zone. We'll use a pre-break line to print the zone for each group.

Selecting a field to sort on

1. Select Fields to sort from the Design menu.

   The list of field descriptions is displayed.

2. Select Zone. This will sort the report by the zone field data in each record.

3. Press the Exit shortcut.

   Now that you're in the Sort Fields list, you can tailor the sort to your specifications.

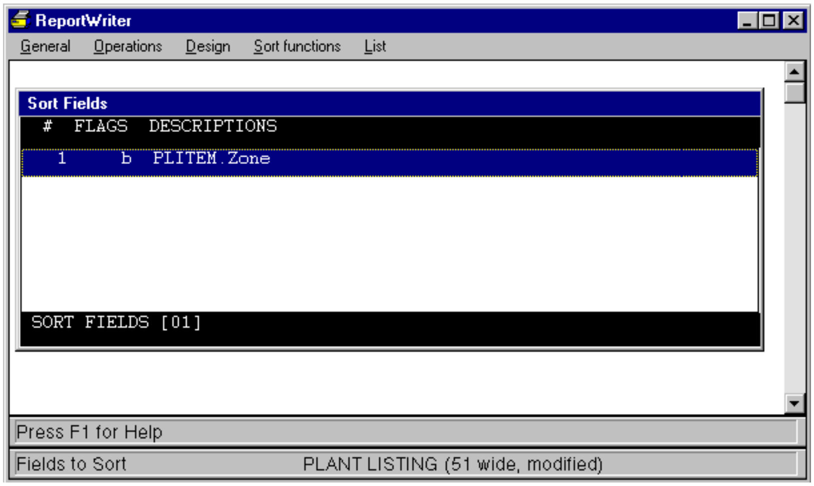
4. Select Change break status from the Sort functions menu.

   Notice that an uppercase **B** appears to the left of the field name. This tells ReportWriter to break on the sort field and generate a new page after each break. We don't want a page break after each zone, so let's change the break status again.

5. Select Change break status again.

   Now the **b** is lowercase. The lowercase **b** tells ReportWriter to break on the field without generating a page break. (See figure 2-6.)

6. Press the Exit shortcut.



Sort Fields

```
 #  FLAGS  DESCRIPTIONS
  1      b  PLITEM.Zone

SORT FIELDS [01]
```

Press F1 for Help

Fields to Sort PLANT LISTING (51 wide, modified)

Figure 2-6. Sorting on the Zone field and setting the break status.

Creating a text field

Now we need to consider what we want to see before each zone break. If we displayed our report right now, the fields we've selected would be shown, but the user would have no idea how the report was sorted. Rather than making the user guess how the data is sorted, let's print a heading like this:

Zone 1

before each break (or grouping of sorted data). To do this, we need to create a text field that prints "Zone," and then select that field and the actual zone data field to print in a pre-break line. First, let's create the text field.

1. From the Design menu, select Fields to create.

   Two lists are displayed: the Temporary Fields list and the Available Field Types list.

2. Select Text Fields from the list of Available Field Types.

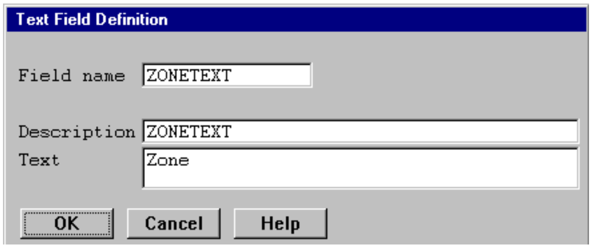
   The Text Field Definition window is displayed. You'll be prompted for three things when defining a text field: the field name, a field description (which defaults to the field name), and the text associated with the field.

3. Enter "ZONETEXT" in the Field name field.

4. Press ENTER in the Description field.

5. Enter "Zone" in the Text field, as shown in figure 2-7. You've now created a text field. Any user-created fields are stored in their own temporary file for use in your report.

6. Press the Exit shortcut (or click OK on Windows).


Text Field Definition

Field name: ZONETEXT

Description: ZONETEXT

Text: Zone

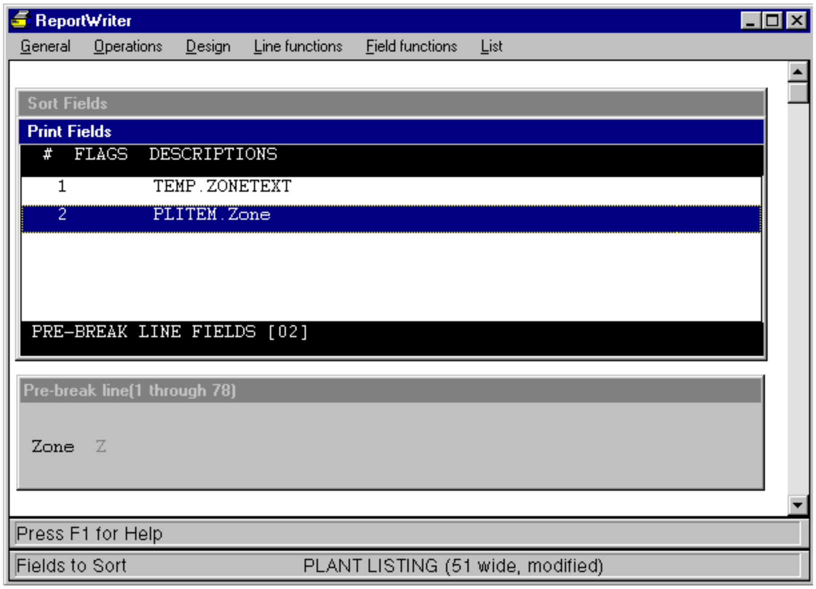
Figure 2-7. Defining the ZONETEXT text field.

Adding a pre-break line

We'll use our newly-created text field to create a pre-break line with this text:

Zone #

1. From the Design menu, select Fields to sort.
2. From the Sort functions menu, select Pre-break line.
3. Press TAB to move to the fields from TEMP selection list. (Note: Pressing TAB cycles through the available report files.)
4. Select ZONETEXT.
5. Press TAB to move back to the list of fields in the plant item file.
6. Select Zone.
7. Press the Exit shortcut. (See figure 2-8.)
8. Press the Display report shortcut to generate the report, as shown in figure 2-9.
9. Press the Exit shortcut to exit when you're finished viewing the report.



```
 #  FLAGS  DESCRIPTIONS
 1          TEMP.ZONETEXT
 2          PLITEM.Zone

PRE-BREAK LINE FIELDS [02]
```

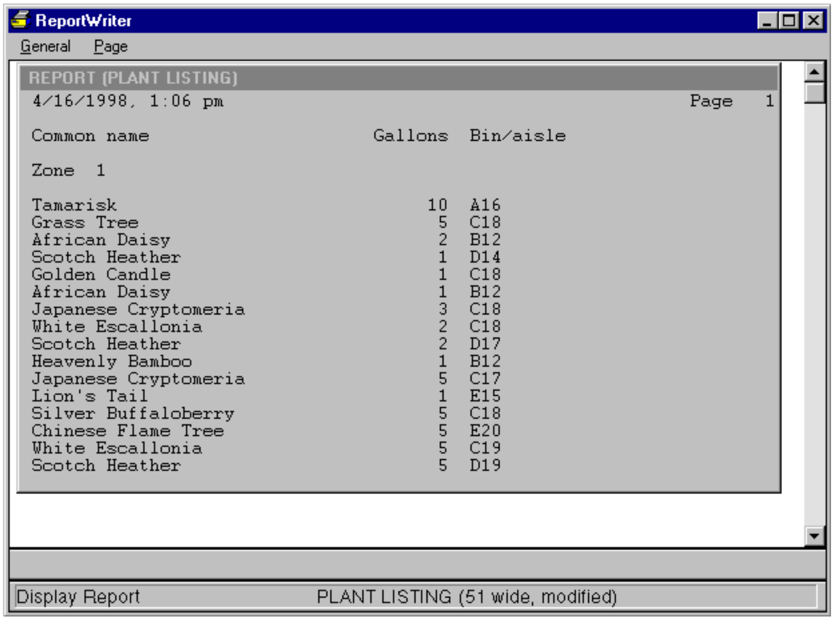
Pre-break line(1 through 78)

```
Zone  Z
```

Press F1 for Help

Fields to Sort — PLANT LISTING (51 wide, modified)

Figure 2-8. Creating a pre-break line.



REPORT (PLANT LISTING)

```
4/16/1998, 1:06 pm                                Page   1

Common name                      Gallons  Bin/aisle

Zone  1

Tamarisk                              10  A16
Grass Tree                             5  C18
African Daisy                          2  B12
Scotch Heather                         1  D14
Golden Candle                          1  C18
African Daisy                          1  B12
Japanese Cryptomeria                   3  C18
White Escallonia                       2  C18
Scotch Heather                         2  D17
Heavenly Bamboo                        1  B12
Japanese Cryptomeria                   5  C17
Lion's Tail                            1  E15
Silver Buffaloberry                    5  C18
Chinese Flame Tree                     5  E20
White Escallonia                       5  C19
Scotch Heather                         5  D19
```

Display Report — PLANT LISTING (51 wide, modified)

Figure 2-9. Displaying the PLANT LISTING report sorted by zone.

Summary

What you learned...	For more information see...
How to select a field to sort on and specify a break status.	Chapter 10, "Sorting Your Report"
How to create a text field.	Chapter 8, "Using Temporary Fields"
How to place the text field and the sort field into a pre-break line.	"Creating a Pre- or Post-Break Line" on page 10-8

Exercise Three: Using Question Fields

Approximate time: 12 minutes

In exercise three, we'll create a new report and use a question field to print only plants with a user-specified sun requirement. Before we create the new report, though, let's save the work we've already done.

Saving your report

1. From the Operations menu, select Save report.
2. Press ENTER to continue.

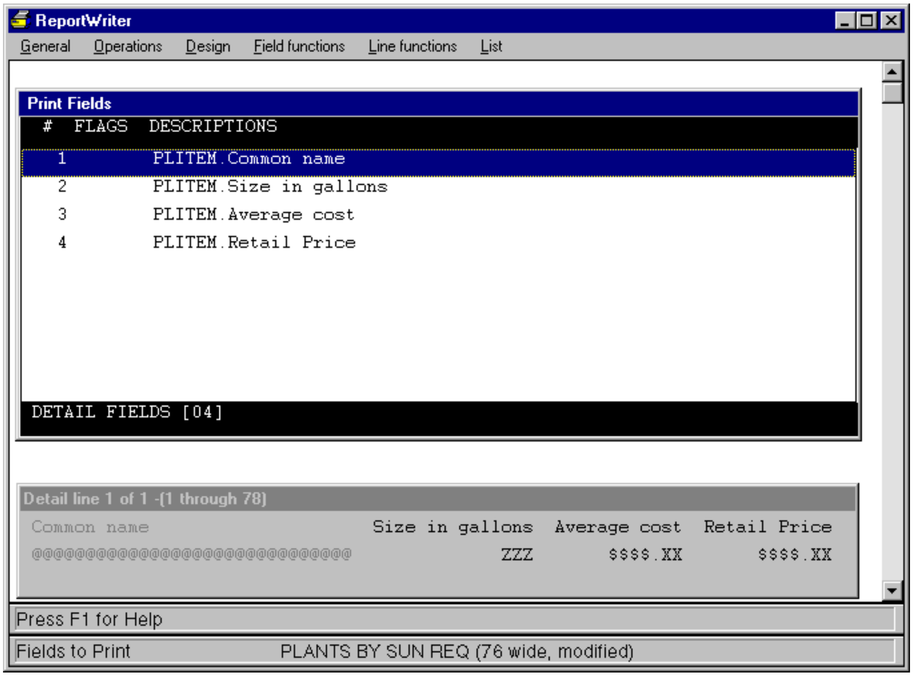
Creating a new report

1. From the Operations menu, select Create new report.
2. Enter "PLANTS BY SUN REQ" in the Report name field.
3. Select Files to read.
4. Select Plant item file.
5. Press the Exit shortcut.

Selecting fields to print

1. Select Fields to print.
2. Select Size in gallons and Common name.
3. Press the Next page shortcut until "Average cost" appears in the list.
4. Select Average cost and Retail Price.
5. Press the Exit shortcut.

   If you'd like, you can move the fields around so that the Common name field is the first field in the report, as shown in figure 2-10. If you need help, see "Reordering fields" on page 2-6. You can also change the field headings; see "Changing a field heading" on page 2-7.
6. Press the Exit shortcut again.



```
Print Fields
 #  FLAGS  DESCRIPTIONS
   1         PLITEM.Common name
   2         PLITEM.Size in gallons
   3         PLITEM.Average cost
   4         PLITEM.Retail Price

DETAIL FIELDS [04]
```

```
Detail line 1 of 1 -(1 through 78)
Common name                              Size in gallons  Average cost  Retail Price
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@              ZZZ       $$$$.XX       $$$$.XX
```

```
Press F1 for Help
Fields to Print                PLANTS BY SUN REQ (76 wide, modified)
```

Figure 2-10. Selecting fields to print for the PLANTS BY SUN REQ report.

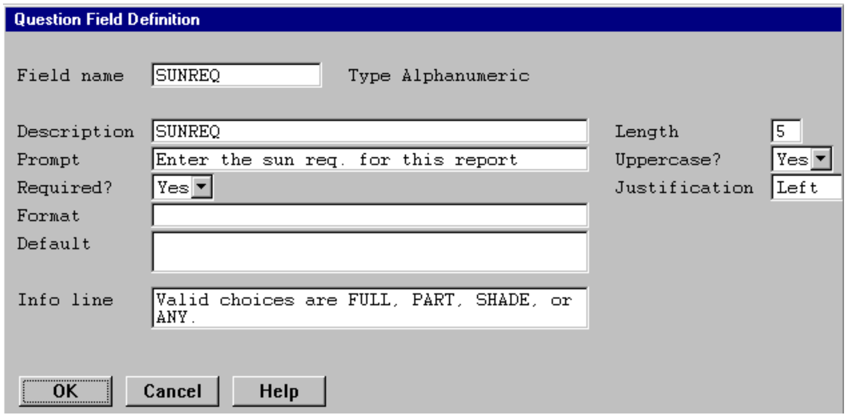
Creating a question field

Now we'll specify which records we want included in the report. In the first and second exercises, we included all records. For this report, though, we're going to let the user specify which records will be printed. To do this, we'll create a question field.

Question fields are a powerful mechanism for customizing reports. For example, if you want to print a customer listing by state but don't want to store 50 reports (one for each state), you can just create a question field that prompts the user for the state to include on the report.

For this report, we're going to prompt the user for the sun requirements of the plants to include in the report. Because the sun requirements are stored in the file as "FULL," "PART," "SHADE," or "ANY," we'll need to create an alphanumeric field.

1. From the Design menu, select Fields to create.
2. Select Question Fields from the Available Field Types list.
3. Select Alphanumeric from the Question Field Type field to display the Question Field Definition window. (See figure 2-11.)
4. Enter "SUNREQ" in the Field name field.



Question Field Definition

Field name SUNREQ Type Alphanumeric

Description SUNREQ
Prompt Enter the sun req. for this report
Required? Yes
Format
Default
Info line Valid choices are FULL, PART, SHADE, or ANY.

Length 5
Uppercase? Yes
Justification Left

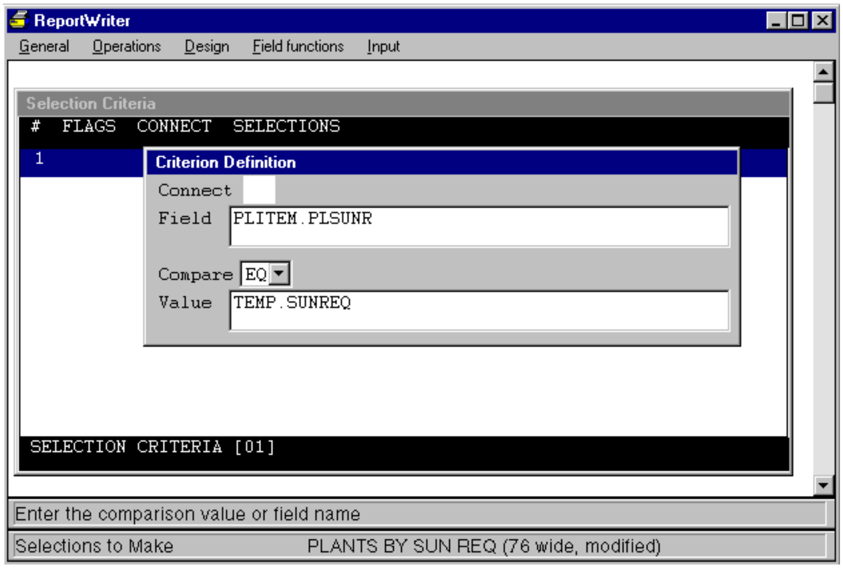
Figure 2-11. Defining the SUNREQ question field.

5. Press ENTER in the Description field to accept the default.
6. Enter “Enter the sun req. for this report” in the Prompt field. This is the prompt that the user sees on the screen when generating the report.
7. Select Yes and then press ENTER in the Required field.
8. Press ENTER in the Format field. We don’t need a format in this instance. (You might enter a format if the user were entering numeric data that you wanted displayed with the appropriate commas and periods.)
9. Press ENTER in the Default field. If a particular choice will be used more often than others, you would enter that choice in this field.
10. Enter “Valid choices are FULL, PART, SHADE, or ANY.” in the Info line field. This text displays in a reverse video bar on the bottom part of the screen when the user generates the report. You’ll most commonly use this area for short help messages and other user assistance. (Note that the text in this field wraps automatically to the next line; you don’t have to press ENTER.)
11. Enter “5” in the Length field.
12. Select Y for Uppercase and press ENTER.
13. Press ENTER in the Justification field to accept the default justification (left).
14. Press the Exit shortcut (or click OK on Windows).

Now that we’ve created our question field, as shown in figure 2-11, let’s use it to make selections for our report.

Defining a selection criterion

We need to relate the data the user enters (FULL, PART, SHADE, or ANY) to the records in the data file, as shown in figure 2-12.



Connect
Field PLITEM.PLSUNR
Compare EQ
Value TEMP.SUNREQ

SELECTION CRITERIA [01]

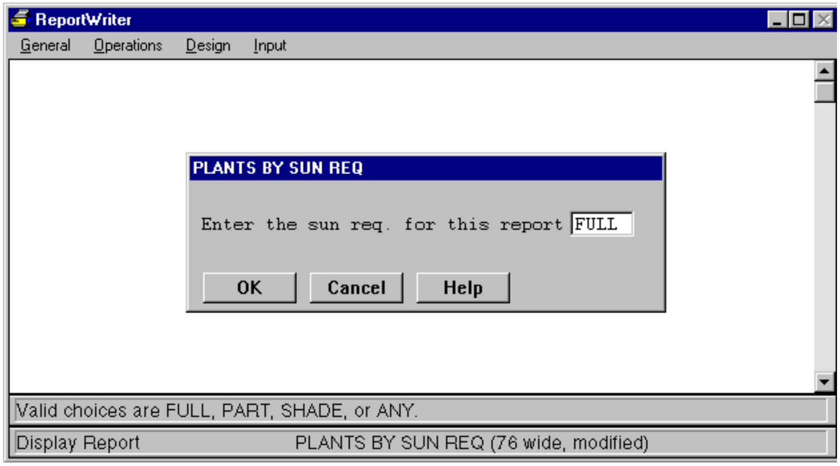
Enter the comparison value or field name

Selections to Make PLANTS BY SUN REQ (76 wide, modified)

Figure 2-12. Selecting on the SUNREQ field.

1. From the Design menu, select Selections to make.
2. Press the List selections shortcut or select List selections from the Field functions menu. This shortcut gives you a list of the fields that are available for your report.
3. Press the Previous page shortcut until “Sun requirements” appears in the list.
4. Select Sun requirements.
5. Press ENTER to enter the name of the Sun requirements field.
6. Select EQ in the Compare field.
7. Press the List selections shortcut in the Value field.
8. Press TAB to display the list of temporary fields.
9. Select SUNREQ.
10. Press ENTER to enter the data in the Criterion Definition window.
11. Now press the Display report shortcut to generate your report.

Note that the question is displayed just as we entered it and that our information is displayed on the information line, as shown in figure 2-13.



Enter the sun req. for this report FULL

Valid choices are FULL, PART, SHADE, or ANY.

Display Report PLANTS BY SUN REQ (76 wide, modified)

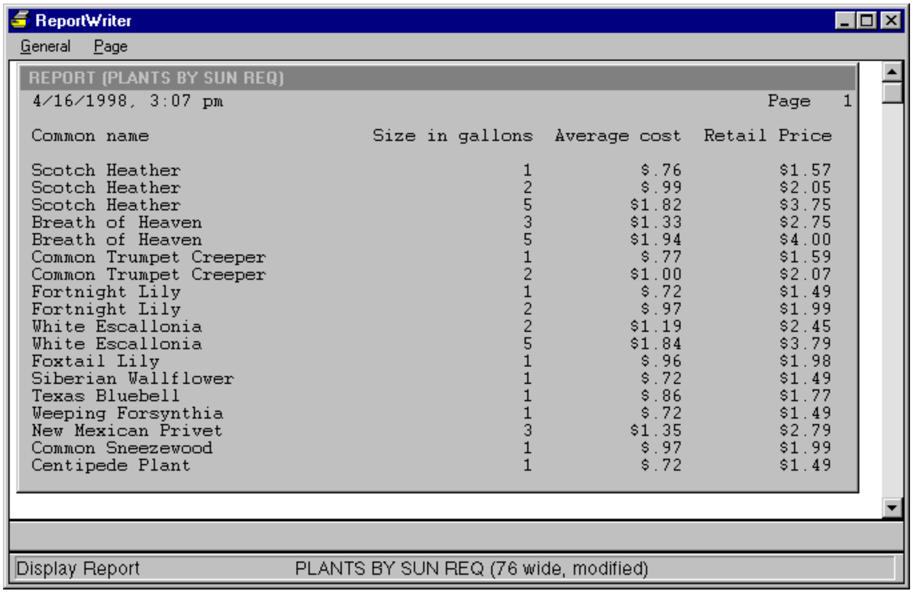
Figure 2-13. Questioning the user for the sun requirements for this report.

12. Type “FULL” and then press ENTER (or click OK on Windows) to see the report. (See figure 2-14.)

Redisplay the report and change your answer to the prompt; notice how the data changes. You can use this same technique for starting and ending dates, customer categories, dollar amounts, or anything else the user can enter. When you’re done viewing the report, press the Exit shortcut, and then save the work you’ve done.

13. From the Operations menu, select Save report.

14. Press ENTER to continue.



```
4/16/1998,  3:07  pm                                                Page    1

Common name                         Size in gallons  Average cost  Retail Price

Scotch Heather                                    1          $.76         $1.57
Scotch Heather                                    2          $.99         $2.05
Scotch Heather                                    5         $1.82         $3.75
Breath of Heaven                                  3         $1.33         $2.75
Breath of Heaven                                  5         $1.94         $4.00
Common Trumpet Creeper                            1          $.77         $1.59
Common Trumpet Creeper                            2         $1.00         $2.07
Fortnight Lily                                    1          $.72         $1.49
Fortnight Lily                                    2          $.97         $1.99
White Escallonia                                  2         $1.19         $2.45
White Escallonia                                  5         $1.84         $3.79
Foxtail Lily                                      1          $.96         $1.98
Siberian Wallflower                               1          $.72         $1.49
Texas Bluebell                                    1          $.86         $1.77
Weeping Forsynthia                                1          $.72         $1.49
New Mexican Privet                                3         $1.35         $2.79
Common Sneezewood                                 1          $.97         $1.99
Centipede Plant                                   1          $.72         $1.49
```

Display Report PLANTS BY SUN REQ (76 wide, modified)

Figure 2-14. Displaying the PLANTS BY SUN REQ report.

Summary

What you learned...	For more information see...
How to save a report.	"Saving a Report" on page 11-2
How to create a question field that prompts the user of the report for information.	"Creating a Question Field" on page 8-15
How to define selection criterion for the question field.	"Defining a Selection Criterion" on page 9-2

Exercise Four: Reading Multiple Files

Approximate time: 9 minutes

This exercise shows how to create a report that uses data from more than one file. Our report will use data from two files: the open order file and the customer file.

Here's the scenario: One of our suppliers tells us that he can't deliver some one-gallon plants we have on order. We need to inform all customers who have open orders for this item so they can decide whether they want to order something else or wait for the supplier's delivery.

Since part of the information we need (the open orders for the item in question and the customer number) is in the open order file, and part of the information (the customer name, contact name, and phone number) is in the customer file, we'll pull information from both files into one report.

Creating a new report

1. From the Operations menu, select Create new report.
2. Name the report "PROBLEM OPEN ORDERS".
3. Select Files to read.
4. Select Open order file.

   Once you've selected the Open order file, you can select another file that's associated with it.

5. From the File functions menu, select Add file. The file list displays only files that are associated with the Open order file.
6. Select Customer file.
7. Press the Exit shortcut.

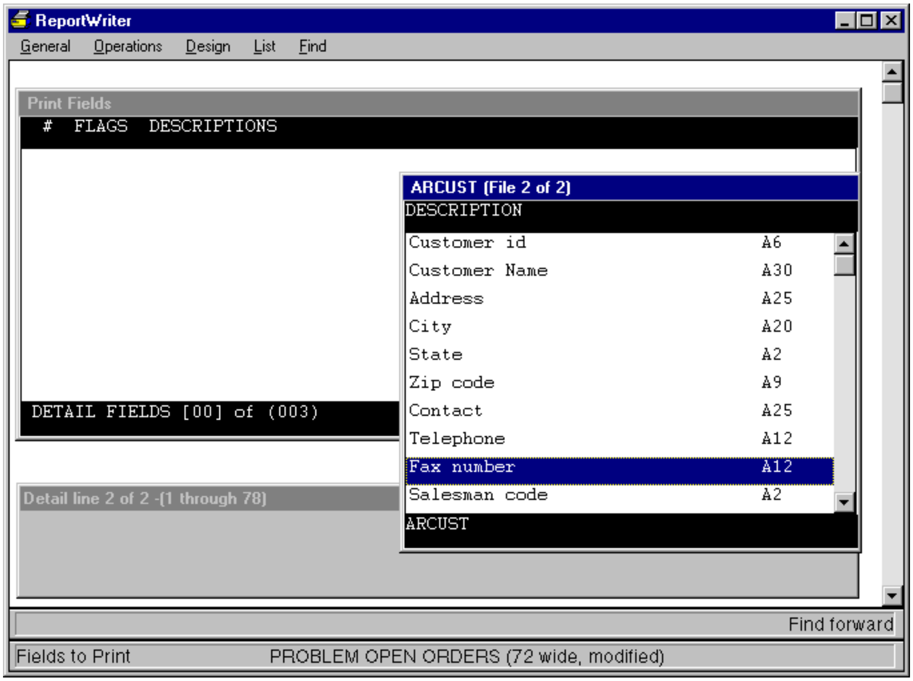
Selecting fields to print

1. Select Fields to print.
2. Press TAB to display fields from the Customer file.
3. Select Customer Name, Contact, and Telephone.
4. Press the Exit shortcut.

   Before we select fields from the Open order file, we'll create a new line on which to display those fields. We could create a report that's wider than 80 columns, but creating a new line makes it easier to see the information on the screen.

5. From the Line functions menu, select Create new line.

   Note that the window at the bottom of the screen is now titled "Detail line 2 of 2," and no fields are displayed, as shown in figure 2-15.



```
Print Fields
  #  FLAGS  DESCRIPTIONS

DETAIL FIELDS [00] of (003)

Detail line 2 of 2 -(1 through 78)
```

ARCUST (File 2 of 2)

DESCRIPTION	
Customer id	A6
Customer Name	A30
Address	A25
City	A20
State	A2
Zip code	A9
Contact	A25
Telephone	A12
Fax number	A12
Salesman code	A2

ARCUST

```
                                                   Find forward
Fields to Print          PROBLEM OPEN ORDERS (72 wide, modified)
```

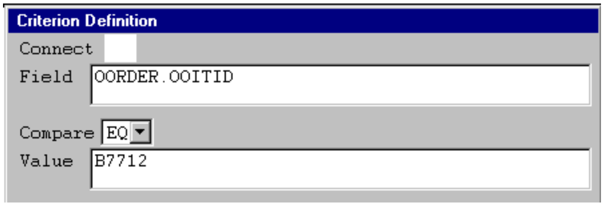
Figure 2-15. Selecting fields for a second detail line.

6. Press TAB to switch back to the fields from the Open orders file.
7. Press the Next page shortcut until “Description” appears in the list.
8. Select Description and Qty ordered.
9. Press the Exit shortcut two times.

Defining the selection criterion

Now that the format of the report is defined, we need to indicate which item is on back order, so that we select only those customers who have placed an order for that item. The item that's not available is item ID B7712.

1. From the Design menu, select Selections to make.
2. Press the List selections shortcut at the Field prompt.
3. Select Plitem id and press ENTER to enter the name of the Plitem id field.
4. Select EQ in the Compare field.
5. Enter B7712 in the Value field, as shown in figure 2-16.


Criterion Definition

Connect

Field OORDER.OOITID

Compare EQ

Value B7712

Figure 2-16. Selecting records whose item ID equals B7712.

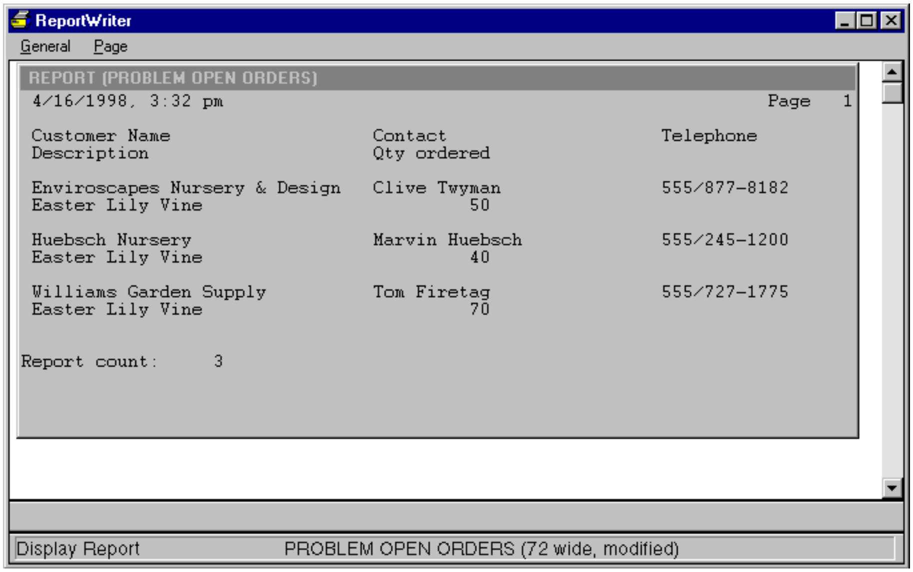
6. Press the Exit shortcut.
7. Press the Display report shortcut to view your report.

   You'll notice that the information is a little too crowded to read easily. So let's add a blank line or two between each selection.

8. Press the Exit shortcut to exit the report display.

Adding a blank line between records

1. From the Design menu, select Miscellaneous.
2. Press ENTER in the Lines per page field.
3. Enter 1 in the Blank lines between records field.
4. Press the Exit shortcut (or click OK on Windows).
5. Press the Display report shortcut to see how it looks with the additional spacing. (See figure 2-17.)
6. Press the Exit shortcut to exit the report display.
7. From the Operations menu, select Save report.
8. Press ENTER to continue.



```
REPORT (PROBLEM OPEN ORDERS)
 4/16/1998, 3:32 pm                                              Page    1

 Customer Name                    Contact                    Telephone
 Description                      Qty ordered

 Enviroscapes Nursery & Design    Clive Twyman               555/877-8182
 Easter Lily Vine                          50

 Huebsch Nursery                  Marvin Huebsch             555/245-1200
 Easter Lily Vine                          40

 Williams Garden Supply           Tom Firetag                555/727-1775
 Easter Lily Vine                          70

Report count:     3
```

Display Report PROBLEM OPEN ORDERS (72 wide, modified)

Figure 2-17. Displaying the PROBLEM OPEN ORDERS report with a blank line between records.

Summary

What you learned...	For more information see...
How to select more than one file for use in a report.	"Selecting a File to Read" on page 5-2
How to create a new detail line.	"Adding a new detail line" on page 7-5
How to add a blank line between records to improve readability of the report.	"Specifying Miscellaneous Design Options" on page 7-25

Exercise Five: Using Calculation Fields

Approximate time: 11 minutes

In this exercise, we'll create a report that gives the year-to-date gross profit percentage for each inventory item. We'll sort this report by plant size and give an average gross profit figure for each plant size.

Creating a new report

1. From the Operations menu, select Create new report.
2. Name the report "YTD GROSS PROFIT PER ITEM".
3. Select Files to read.
4. Select Plant item file.
5. Press the Exit shortcut.

Although all the data we need to calculate the gross profit is stored in the plant item file, there is no gross profit field. To determine the year-to-date gross profit for each item, we'll need to create a calculation field where the calculation is defined and its result is stored.

Creating a calculation field

Before we can define a calculation field, we need to know how to get the calculated result. In this case, we're going to use the following equation as our calculation:

```
((ytd sales $ - ytd cost $)/ytd qty)*100
```

1. Select Fields to create.
2. Select Calculation Fields.
3. Enter "GROSPROF" in the Field name field.
4. Select Numeric for Type.
5. Enter 4 in the Length field and 2 in the Precision field. The field length will be four places, with two to the right of the decimal point.

Now you're in the GROSPROF field, which is where you define the calculation. Ultimately, our calculation will look like this:

```
((PLITEM.PLDLYR-PLITEM.PLCSYR)/PLITEM.PLQTYR)*100
```

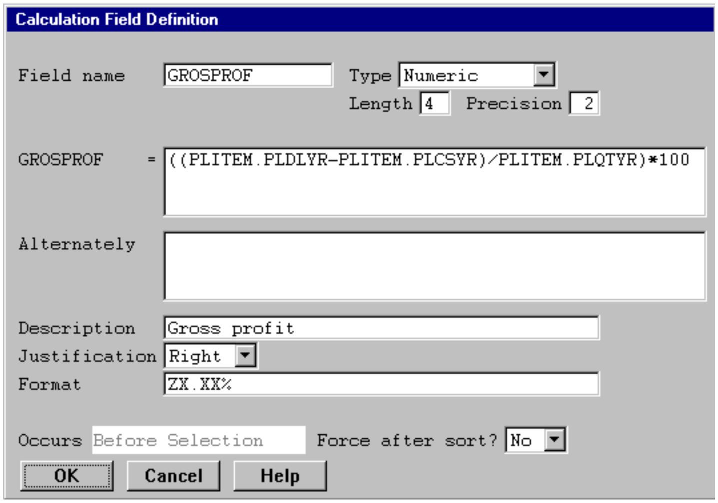
There are two ways to enter the calculation: either you can just type in the above string exactly as you see it, or you can build the calculation as follows:

6. Type ((.
7. Press the List selections shortcut.
8. Press the Next page shortcut until Sales $ ytd displays, and then select it.
9. Type –.
10. Press the List selections shortcut and select Cost ytd.
11. Type)/.
12. Press the List selections shortcut and select Qty sold ytd.
13. Type)*100.
14. Press ENTER. Your completed calculation should now look like the one above.
15. Press ENTER in the Alternately field.
16. Enter "Gross profit" in the Description field. This text is used as the default field header in the report.
17. In the Justification field, select Right. This is a numeric value, so we'll want to right-justify the results.
18. Enter ZX.XX% in the Format field (uppercase letters only). The characters Z and X are placeholders for significant digits. The Z suppresses the digit (prints a blank) if it's equal to zero, and the X prints a zero if a digit is equal to zero. The percent sign (%) is not a formatting character, so it will be transferred directly to the right of the calculation result.
19. Select No in the Force after sort field. Your screen should look like figure 2-18.
20. Press the Exit shortcut (or click OK on Windows).

Now that the calculation field is defined, let's define the layout for the report.

Defining the report layout

1. Select Fields to print.
2. Press the First entry shortcut to move to the top of the Fields from PLITEM list.
3. Select Common name and Size in gallons.
4. Press TAB to display the TEMP fields.
5. Select Gross profit.
6. Press the Exit shortcut two times.



Field name GROSPROF
Type Numeric
Length 4 Precision 2
GROSPROF = ((PLITEM.PLDLYR-PLITEM.PLCSYR)/PLITEM.PLQTYR)*100
Alternately
Description Gross profit
Justification Right
Format ZX.XX%
Occurs Before Selection
Force after sort? No

Figure 2-18. Defining the GROSPROF calculation field.

Now that you've defined the field layout, let's sort the report so the most profitable items are shown at the top.

Sorting the report

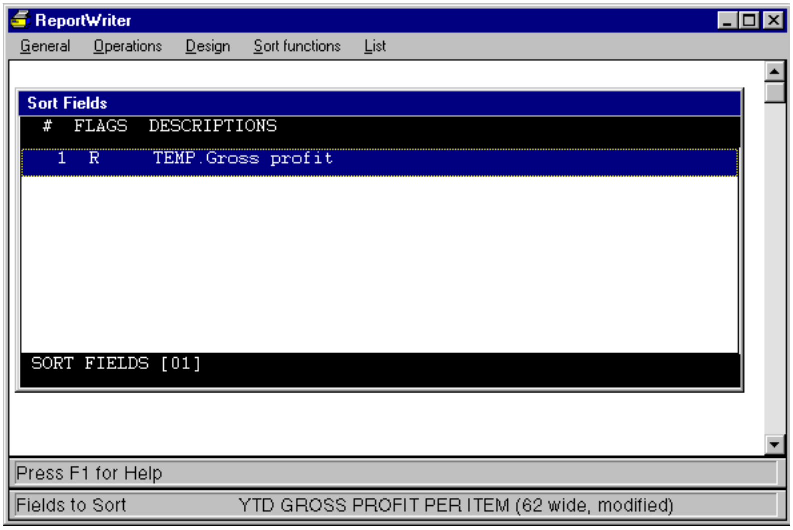
1. Select Fields to sort.
2. Select Gross profit.

   By default, all fields are sorted in ascending order. But it makes more sense to show the most profitable items first, so we'll change the sort order.

3. Press the Exit shortcut.
4. From the Sort functions menu, select Change sorting order.

   Notice that an uppercase "R" appears to the left of the field name. This tells ReportWriter to reverse sort on that field. (See figure 2-19.)

   That's it—go ahead and generate the report, as shown in figure 2-20.



Sort Fields

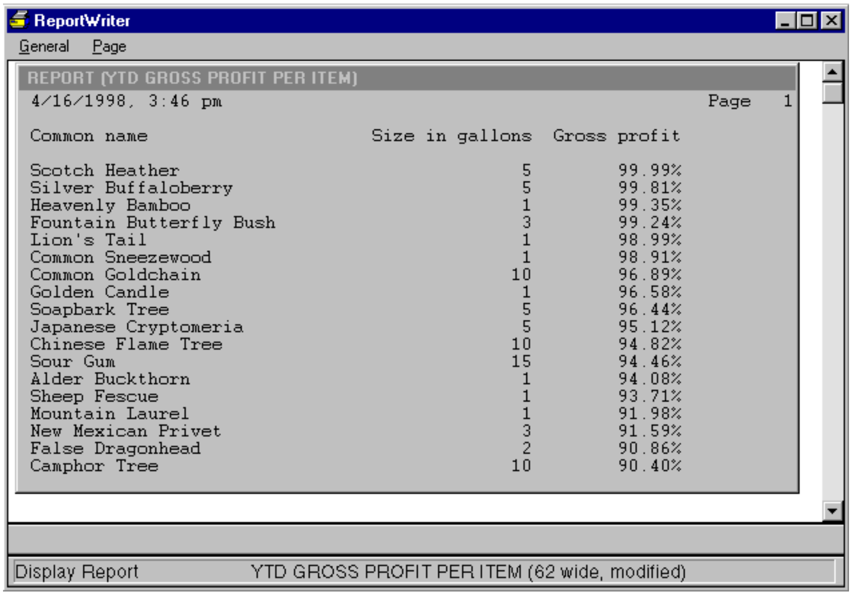
```
 #  FLAGS  DESCRIPTIONS
   1  R      TEMP.Gross profit

SORT FIELDS [01]
```

Press F1 for Help

Fields to Sort YTD GROSS PROFIT PER ITEM (62 wide, modified)

Figure 2-19. Changing the sorting order to descending.

5. Press the Display report shortcut to look at the report. When you're done, press the Exit shortcut.



REPORT (YTD GROSS PROFIT PER ITEM)

```
4/16/1998, 3:46 pm                                          Page   1

Common name                      Size in gallons  Gross profit

Scotch Heather                                 5        99.99%
Silver Buffaloberry                            5        99.81%
Heavenly Bamboo                                1        99.35%
Fountain Butterfly Bush                        3        99.24%
Lion's Tail                                    1        98.99%
Common Sneezewood                              1        98.91%
Common Goldchain                              10        96.89%
Golden Candle                                  1        96.58%
Soapbark Tree                                  5        96.44%
Japanese Cryptomeria                           5        95.12%
Chinese Flame Tree                            10        94.82%
Sour Gum                                      15        94.46%
Alder Buckthorn                                1        94.08%
Sheep Fescue                                   1        93.71%
Mountain Laurel                                1        91.98%
New Mexican Privet                             3        91.59%
False Dragonhead                               2        90.86%
Camphor Tree                                  10        90.40%
```

Display Report YTD GROSS PROFIT PER ITEM (62 wide, modified)

Figure 2-20. Displaying the YTD GROSS PROFIT PER ITEM report.

6. Select Save report from the Operations menu, and the press ENTER to continue.

7. Select Quit from the General menu to exit ReportWriter.

Summary

What you learned...	For more information see...
How to create a calculation field.	"Creating a Calculation Field" on page 8-5
How to sort a report in reverse order.	"Setting sort order" on page 10-6

3

Creating or Loading a Report

Creating a New Report

1. From the Operations menu, select Create new report.
2. In the Report name field, enter a name of the report you want to create.
3. Exit the window.

The name you type is the default header for your report. (See "Creating the Header and Footer Layout" on page 7-17 for details on how to change a report header.)

If you type the name of a report that already exists or select a report from the directory listing, you are prompted

A report already exists with that name. Use it anyway?

Select Yes to overwrite the old report with the new one; the existing report name is used, and the existing report definition is overwritten when you save the new report. Select No to cancel the request and bring up the new report name prompt again.

When you enter a report name, that name is displayed in the lower right corner of the screen, and the Design menu is pulled down from the menu bar. The entries in the Design menu comprise the steps you'll need to perform to create your report. At a minimum, you'll need to select at least one file to read and one field to print before you can generate your report. The following chapters explain these steps in detail:

Chapter 5, "Choosing the Files to Read"

Chapter 7, "Designing Your Report"

Chapter 8, "Using Temporary Fields"

Chapter 9, "Specifying Selection Criteria"

Chapter 10, "Sorting Your Report"

Once you've specified a report name in the **Report name** field, that report is referred to as the *loaded* report. To have a report loaded means that you have exclusive control of modifying that report.

Creating a new report by copying an existing one

You can also create a new report by loading an existing report using the Load existing report function and saving it to a new name. This method is beneficial when the report you want to create is similar to an existing report, because you can just make minor changes to the report instead of defining it from scratch.

1. From the Operations menu, select Load existing report.
2. In the Report name field, enter the name of the report you want to copy. (To select a report from a list of existing reports, use the List selections function.)
3. Exit the window.
4. Save the report by selecting Save report from the Operations menu.
5. In the Report name field, enter a new name for the report.
6. Exit the window. You'll see a message that the report was saved.
7. Press ENTER to continue.

The original report is still loaded. To modify your new report, select Load existing report from the Operations menu and enter the name of the new report in the Report name field.

Loading an Existing Report

1. From the Operations menu, select Load existing report.
2. In the Report name field, enter the name of an existing report or use the List selections function on the Name functions menu to select from a list of available reports.

   You can display the creation and last modification dates for these reports by selecting Toggle view from the List menu. To redisplay the report names without dates, select Toggle view again.
3. Exit the window.

   When the report is loaded, its name and width are displayed in the lower right corner of the screen. To have a report loaded means that you have exclusive control of modifying that report. You can edit any of the files to read, fields to create, fields to print, selection criteria, sort fields, headers/footers, and so forth that were designated when the report was created. If you change any of these items, you'll be asked if you want to save your changes when you exit the report.

   You can also create a new report that has all the characteristics of the loaded report by saving the current one to a new report name. See "Saving a Report" on page 11-2 for more information.

Loading a report in read-only mode

Loading a report in read-only mode enables you to edit any of the files to read, fields to create, fields to print, selection criteria, sort fields, and so forth and then display or print the report, without actually affecting the report definition.

If you try to load a report that's already in use, you are prompted

Report in use. Do you want to load the report anyway?

Press ENTER if you don't want to load the report. To load the report for read-only purposes, select Yes. The letter "<R>" (in brackets) is appended to the report name and width displayed in the lower right corner of the screen. For example, if your report is called SALES, it may look like this:

```
SALES (50 wide)<R>
```

If you've modified the report and you want to save your changes, select Save report from the Operations menu. You are prompted with the name of the current (read-only) report, which you must modify. You cannot save your changes under the current name.

4

Generating Your Report

Displaying a Report on the Screen

To display a report on your screen, select Display report from the Operations menu.

- If a complete report is currently loaded, that report is displayed. The report remains loaded when the display is complete.
- If no report is loaded, you are prompted for the report name. Enter the name of the report you want to view. To select a report from a list of available reports, select List selections from the Name functions menu. When the display is complete, no report is loaded.

You can cancel the display of a report by pressing the interrupt key for your system.

You can select Display report from anywhere in ReportWriter, but if you haven't finished defining at least one field to print (on a detail line, break line, or header/footer line), you'll get an error message.

Viewing hints

- Once the report is displayed, ReportWriter pauses at the end of each screen page.
- If the right side of the report is out of view, you can select Scroll right from the Page menu to display it. Select Scroll left to redisplay the left side of the report.
- When you're ready to view the next page, select Next page from the Page menu.
- To redisplay previously viewed pages of the report, select Previous page to move up one page.
- To go to the top of the report, select First page.
- To return to the most recently viewed page, select Last page. (Note that Last page does not go to the last page in the report.)
- To exit the report display, press the Exit shortcut.

Printing a Report

To send a report to the printer, select Print report from the Operations menu.

- If a report is currently loaded, that report is printed. The report remains loaded when printing is complete.
- If no report is loaded, you are prompted for the report name. Enter the name of the report you want to print. To select a report from a list of available reports, select List selections from the Name functions menu. When printing is complete, no report is loaded.

To cancel printing, press the interrupt character for your system. An error message is displayed.

Printing Labels

ReportWriter is an excellent tool for printing labels. Use these guidelines to help create a report that prints labels.

1. Select your report files.
2. In the Miscellaneous Report Attributes window,
   - set **Lines per page** to the distance between the top of one label to the top of the next.
   - set **Detail record count** to No.
   - set **Form feed** to No.
3. Define your report record. (The number of lines in the report record must be less than, but greater than half of, the **Lines per page** value set above.)
4. Turn off the display of the page header, page footer, report header, report footer, and date and page number line.

Generating a Report File

Follow these instructions to create an ASCII file that contains your report output.

1. From the Operations menu, select Generate report file.

If no report is loaded, you are prompted for the report name. Enter the name of the report for which you want to generate a file. To select from a list of available reports, select List selections from the Name functions menu.

The Generate Report File window is displayed.

2. Enter data in the fields:

Output filename. Enter a name for your report file. The default file extension is **.ddf**.

Output format. Select an output format:

Report form Generate the entire report to a file, including headers, footers, subtotals, and so forth.

Data only Include only the actual detail record data in the file, without headers, footers, the date and page number line, break lines, subtotal and total lines, break counts, strip characters, and blank lines.

Worksheet Generate an export file for applications such as spreadsheets. If you select this option, a Worksheet Options window is displayed. Enter data in the fields:

Include first page header?: Select Yes to include the header for the first page of the report at the top of the output file. Otherwise, select No.

Field delimiter: Select a delimiter character to separate the report data fields from one another. Valid options are Comma (default), Tab, Space, Semicolon.

The current report data (detail and break line) is blank stripped and separated by the specified delimiter character. Alphanumeric, date, and time fields are enclosed in quotation marks. Formatted numeric fields honor the “Z”, “X”, “.”, and “-” formatting characters; all others are ignored.

3. Exit the window.

To cancel report file generation, press the interrupt character for your system.

5

Choosing the Files to Read

Selecting a File to Read

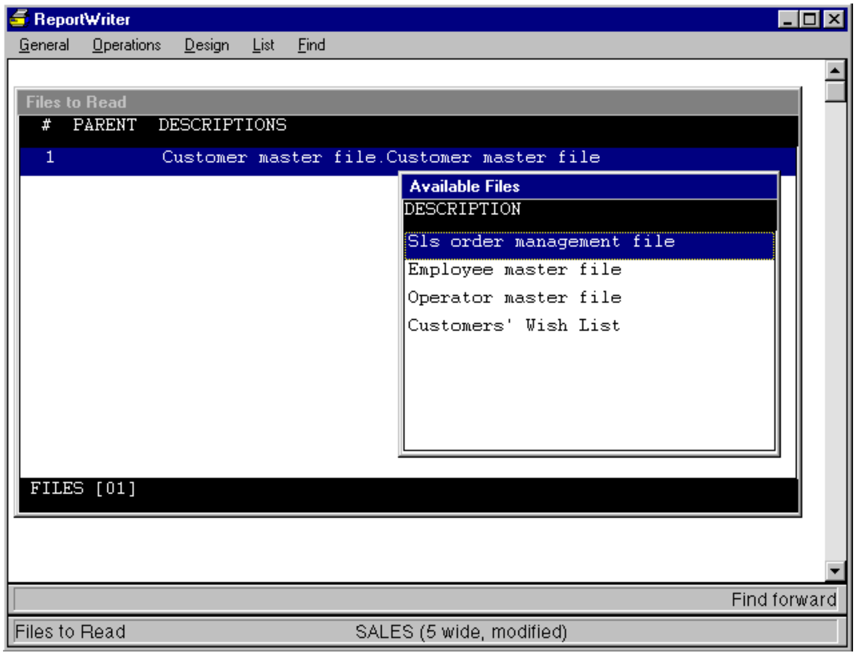
The Files to read function enables you to choose the files from which your report will get its data.

1. From the Design menu, select Files to read.

If you're creating a new report, two lists are displayed.

- The Files to Read list is the large list in the center of the screen. For a new report, this list is empty.
- The Available Files selection list is displayed on the right side of the screen. As you select files from the Available Files list, they are placed in the Files to Read list. (See figure 5-1.)

By default, the Available Files list displays the file description. To display the file definition name instead, select Toggle view from the List menu. Select Toggle view again to display the description.



Files to Read

```
 #  PARENT  DESCRIPTIONS
 1          Customer master file.Customer master file
```

Available Files

```
DESCRIPTION
Sls order management file
Employee master file
Operator master file
Customers' Wish List
```

```
FILES [01]
```

Figure 5-1. Selecting files to read.

When you first enter the Files to read function for a new report, the cursor is in the Available Files list.

2. Select the first file you want ReportWriter to read. You must select files in the order you want ReportWriter to read them. The first file you select is the primary file. The primary file determines which additional files can be read, as well as which fields are available for inclusion in your report.

   If more than one structure is assigned to the file you select, an Available Structures list is displayed. Select a structure from the list. The selected structure determines how ReportWriter views the data in the file. If a structure tag was defined in the repository, ReportWriter filters out the records that meet the tag criteria.

   The selected file is displayed in the Files to Read list in this format:

   ```
   file description.structure description
   ```

3. To add additional files to the Files to Read list, select Add file from the File functions menu.

   The Available Files list contains a list of files that are related to the primary file. ReportWriter processes other files based on what it finds in the primary file.

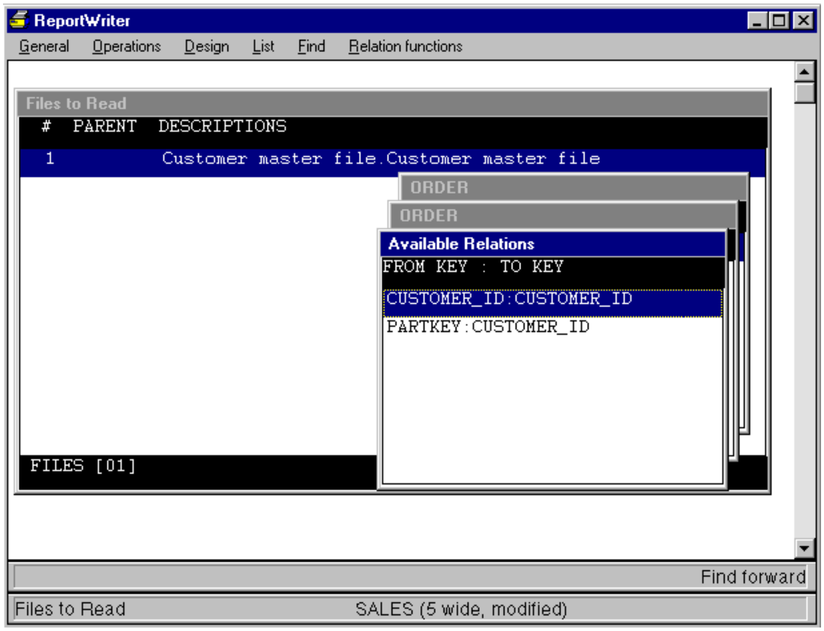
   For example, if the first file we select is a sales order management file and the second file is a customer master file, ReportWriter will read each sales order entry and then look for the associated entry in the customer master file. If the sales order file has more than one entry for each customer, our report will contain the same customer twice. On the other hand, if we select the customer master file first, ReportWriter will read each name and then look for the associated entries in the sales order management file.

4. Select the next file you want ReportWriter to read from the Available Files list.

   When you select a secondary file, it too can have multiple structures assigned to it, and you may need to select the one you want. In addition, if the structure selected (explicitly or implicitly) is used in more than one relation between it and the structure of the related file (currently highlighted in the Files to Read list), the Available Relations list is displayed. (See figure 5-2.)

   This list provides the “from” and “to” keys involved in the available relations. (You can view additional information about these keys to help you make your selection. See “Examining a relationship between files” on page 5-5.) Select the desired relation. It is displayed in the Files to Read list in this format:

   ```
   file description.structure description.to key
   ```



```
Files to Read
 #  PARENT  DESCRIPTIONS
 1          Customer master file.Customer master file
FILES [01]
```

```
ORDER
ORDER
Available Relations
FROM KEY : TO KEY
CUSTOMER_ID:CUSTOMER_ID
PARTKEY:CUSTOMER_ID
```

```
Find forward
Files to Read                SALES (5 wide, modified)
```

Figure 5-2. Selecting files with multiple structures and relations.

5. Continue to add any files that you want in your report.

   The file that is highlighted in the Files to Read list when you select Add file determines which available files are displayed in the Available Files list.

   The number of the file to which a file is related is shown in the PARENT column. For example, if file #1 was highlighted when you selected Add file, a number 1 appears in the PARENT column for the added file. To examine this relationship in more detail, see "Examining a relationship between files" on page 5-5.

   Let's assume that our primary file is the Customer master file. The files available from the Customer master file might be Employee master file, Operator master file, Customer wish list, and Sls order management. Let's say we also selected the Sls order management file to be read. If we highlighted the Sls order management file and selected Add file, the Available Files list would now contain the files that are available from the Sls order management file.

   You can select a maximum of 50 files for your report. However, this limit may be further restricted by the number of open files permitted by your operating system or your available system resources.

6. When you're finished selecting files, press the Exit shortcut from the Files to Read list.

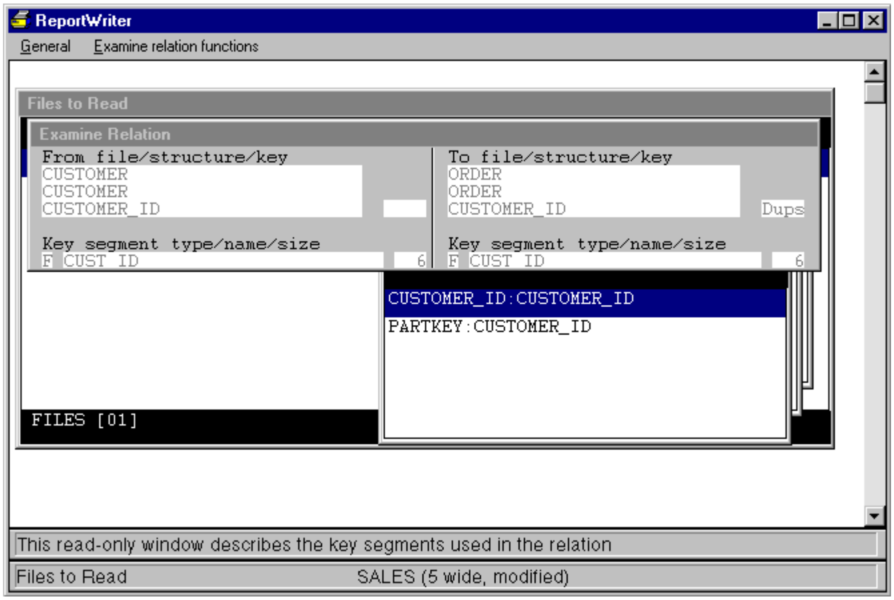
Examining a relationship between files

You can display additional information about the relationship that exists between two files from either the Files to Read list or the Available Relations list using the Examine relation function. This function is available only when two or more files have been selected as Files to Read.

1. Highlight either the relation or a file in the relation that you'd like to examine.

2. From the Relation functions or File functions menu, select Examine relation.

   The Examine Relation window is displayed. This read-only window describes the key segments that are used in the current relation. It contains the names of the "from" file, structure, and key and the names of the "to" file, structure, and key.



From file/structure/key
CUSTOMER
CUSTOMER
CUSTOMER_ID

Key segment type/name/size
F CUST_ID 6

To file/structure/key
ORDER
ORDER
CUSTOMER_ID Dups

Key segment type/name/size
F CUST_ID 6

CUSTOMER_ID:CUSTOMER_ID
PARTKEY:CUSTOMER_ID

FILES [01]

This read-only window describes the key segments used in the relation

Files to Read SALES (5 wide, modified)

Figure 5-3. Examining a relationship between files.

If a key allows duplicates, the word "Dups" is displayed to the right of the key name. The segment information includes the type and size of each key segment:

F	Field
L	Literal
E	External
R	Record number

If the segment is a field or external segment, the field name is specified. You can display the field description instead of the field name by selecting Toggle view from the Examine relation menu. If the segment is a literal, the literal string is displayed.

You can view this information for any file (except the first in the Files to Read list), but it is most useful when a given file/structure combination is related to another file/structure combination more than once (using different keys). When displayed in the Available Files list, these file/structure combinations appear to be the same. The Examine relation function enables you to view the specific relationship that each one represents.

3. To exit the Examine Relation window, press the Exit shortcut.

Examining *temporary* relations is allowed only from the Files to Read list.

If you're examining a temporary relation based on a name link, the name of the "from" key will be "NAME_LINK_KEY." This key is a temporary one that ReportWriter has created for use in this report.

Selecting related files based on name links

Another way to add files to your report is by defining temporary relations when you design the report. These temporary relations are based on name links between fields and access keys.

1. Select the first file you want ReportWriter to read.
2. To add additional files to the Files to Read list, select Add file via name link from the File functions menu.

To access this function, either you must have a repository cross-reference file called *RPSDAT:rpsxref.ism*, or the environment variable RPSXFIL must be set to the name of the cross-reference file that corresponds to the repository you're using.

The Link Fields from *filename* list contains a list of fields in the current file that have name links to other files.

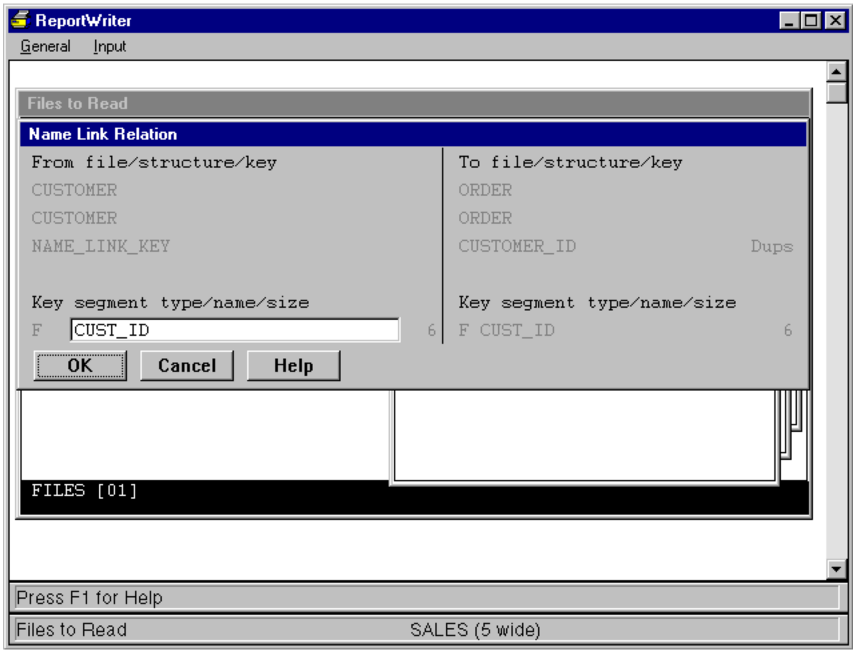
3. Select the field you want ReportWriter to use in accessing a secondary file.

The Available Files list contains a list of files that can be accessed using the selected link field.

4. Select the next file you want ReportWriter to read from the Available Files list.

Because a secondary file can have multiple structures assigned to it, you may need to select the one you want. In addition, if the selected link field matches the link field in more than one access key for the selected structure, the Available Relations list is displayed. This list provides the name of the link field (the "from" key) and the name of the access or "to" key involved in the temporary relation. (If you would like to view more information about these keys to help you make your selection, see page 5-5.) Select the desired relation.

Once you select the secondary file, the Name Link Relation input window is displayed (see figure 5-4), showing you the temporary key and relation that ReportWriter will be creating to access the secondary file. The name of the "from" key will be "NAME_LINK_KEY."



```
From file/structure/key               To file/structure/key
CUSTOMER                              ORDER
CUSTOMER                              ORDER
NAME_LINK_KEY                         CUSTOMER_ID                 Dups

Key segment type/name/size            Key segment type/name/size
F  CUST_ID                        6   F CUST_ID                      6
```

```
FILES [01]
```

```
Press F1 for Help
Files to Read                         SALES (5 wide)
```

Figure 5-4. Defining a temporary name link relation.

Depending on the segments in the “to” key, ReportWriter may be trying to match the full key size, in which case you can’t modify the “from” key, or it may be doing a partial key match, and you can optionally add other name link fields or literal segments to the “from” key to make it a full key match.

If the “to” key consists of more than one segment, ReportWriter will match as many segments as possible with name linked fields. You can change to a literal or clear any of the NAME_LINK_KEY segments other than the first. You may not skip any segments; segment definitions must be contiguous. If you clear a Field type segment, when you set the type back to Field, ReportWriter redisplays the appropriate name linked field.

Press ENTER to accept the current temporary relation.

When you’ve selected your secondary file, it is displayed in the Files to Read list in the following format:

```
file description.structure description.to key
```

An “X” appears in the PARENT column to identify this relation as temporary based on a name link.

5. Continue to add any files that you want in your report.

6. When you’re finished selecting files, press the Exit shortcut from the Files to Read list.

For information about how ReportWriter determines the list of secondary files based on name links, see “Selecting Files to Read” on page 18-2.

Modifying name link relations

To modify a name link relation, highlight the “to” file in the Files to Read list and select Modify name link relation from the File functions menu.

The Name Link Relation input window is displayed. You can change to a literal or clear any of the NAME_LINK_KEY segments other than the first. You may not skip any segments; segment definitions must be contiguous.

Deleting a File from Your Report

1. Highlight the file in the Files to Read window.
2. From the File functions menu, select Delete file.

   You can delete a file provided that all of the following conditions are true:

   - No field from this file is being printed in a detail line, pre- or post-break line, or header or footer line.
   - No field from this file is a sort field.
   - No field from this file is used in a selection statement or other conditional.
   - No field from this file is used as a subscript.
   - No field from this file is used as a range offset or length.
   - No field from this file is used in a calculation field expression.
   - No field from this file is used in a subtotal access field definition.
   - The structure associated with this file is not used to access another file to read. (In other words, it is not linked to another file through a Repository relation.)
   - This file is not used in a multiple projection.

⚠ If you delete all of the files to read in your report, the temporary file and any temporary fields you've created are deleted as well.

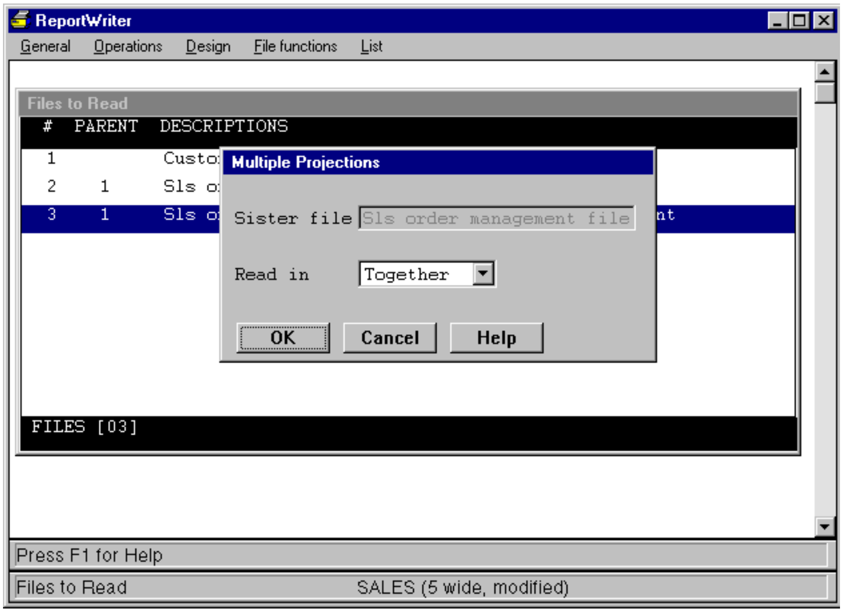
Defining a Multiple Projection

When one of your files to read is directly related to two or more files (for example, File A relates to File B and File A relates to File C), you can specify the method in which those secondary files should be processed. The default method is to create detail records that combine data from both secondary files. The multiple projection option enables you to create detail records first from one file, then from the second file, for a given key value in the primary file. Optionally, using a common sort key, you can then merge the records.

To define multiple projections, both files must be on the same level, which means that both must be related to the file above them if a tree of relationships were to be drawn. Also, both relationships must define one-to-many relationships, which means that one record in File A can relate to multiple records in File B.

Before you can define a multiple projection, both of the secondary files must be selected as files to read.

1. Highlight the second file.
2. From the File functions menu, select Multiple projections. In the Multiple Projections window, the Sister file field displays the description of the other file you selected on the same level. See figure 5-5.



```
 #  PARENT  DESCRIPTIONS
 1          Custo
 2    1     Sls o
 3    1     Sls o                            nt
```

Sister file Sls order management file

Read in Together

FILES [03]

Press F1 for Help

Files to Read SALES (5 wide, modified)

Figure 5-5. Defining a multiple projection.

3. Enter the requested data:

   Read in. The value in this field determines how the current file's records should be processed in relation to its sister file. Valid values are:

   Together — Detail records are created by combining data from this file with data from its sister file. (default)

   Separately — Detail records are created from data in this file only after detail records are created for data in the sister file (for a particular key value). If you select this option, an M displays next to the file in the Files to Read window.

4. Exit the window.

   See "Understanding Multiple Projections" on page 18-6 for more information.

6

Choosing Fields

The List of Available Fields

Entering the list

You can enter the list of available fields in one of two ways:

- Select Add field or Add fields from the Fields to print, Fields to sort, Pre-break line, Post-break line, or Header/footer layout functions.
- From a prompt in an input window that asks you to specify a field name, select List selections from the menu or press its shortcut.

Contents of the list

The first time this list is displayed for a given function, it contains the fields that are available from the primary file you selected in the Files to read function. The title of the list is "*FILENAME* (File 1 of 1)," where *FILENAME* is the name of the primary file. The list footer contains the name of the structure associated with the file.

The data type and the number of characters reserved for each field are listed to the right of the field. If the field is an array, the type and size are preceded by an asterisk (*).

The possible data types are as follows:

A	Alphanumeric
D	Decimal or implied-decimal
DT	Date
TM	Time
U	User-defined
ED	Enumerated (decimal)

For more detailed information about these data types, see "Additional Field Data Type Information" on page 6-4.

Displaying field names versus field descriptions

The fields in the list of available fields can either be displayed as field descriptions or field names. The default is field descriptions. To display field names instead, select Toggle view from the List menu. To switch back to a display of field descriptions, select Toggle view again.

Moving through the list

You can move among the entries in the list of available fields using the UP ARROW and DOWN ARROW keys. If the list of fields from the current file has more than one page, you can move to another page by selecting the appropriate entry from the List menu.

Moving to the next report file

To display the fields that are available for selection from the next report file (if you've selected more than one file to read), select Next report file from the List menu. As you rotate through the files by pressing the Next report file (or Previous report file) shortcuts, the title and footer of the list of available fields changes accordingly. (The footer may include the name of the "to key" to help identify duplicate file/structure combinations.) You can select fields from any of these lists.

Additional Field Data Type Information

Alphanumeric fields

An alphanumeric data type, A, indicates that data in the field can consist of any ASCII characters.

Decimal and implied-decimal fields

A data type of **D** indicates a decimal or implied-decimal data type. Decimal data represents a signed, whole number consisting of ASCII numeric characters (0 through 9). Implied-decimal data is a signed number with a whole number part and a fractional precision, in which either the whole number *or* the fractional precision may be zero. For both decimal and implied-decimal data, a negative sign (–) doesn't take a data storage space. Instead, the sign is stored in the rightmost digit with a 'p' through 'y', corresponding to 0 through 9. For implied-decimal, the decimal point does not take a data storage space either.

Date and time fields

A data type of **DT** or **TM** indicates that the data in that field is stored in a ReportWriter supported date/time storage format. See "Appendix B: Date and Time Formats" for a list of the supported formats.

User-defined fields

The user-defined data type, **U**, indicates that the data in the field may need to be modified by the user before being displayed by ReportWriter. To support such a field, ReportWriter calls a user-replaceable subroutine (RW_USAGE_METHOD) before displaying the field, and it passes the field and its redisplay format to that subroutine each time the field is displayed. When RW_USAGE_METHOD is called, the possible data types are displayed as UA, UD, UDT, and UTM. See page 15-45 for more information about RW_USAGE_METHOD.

Enumerated fields

The enumerated data type, **ED**, indicates that a decimal value stored in the field is associated with an alphanumeric string. For enumerated fields, the length of the format string corresponds to the length of the associated alphanumeric string. You can use a maximum of 99 enumerated fields in one report.

Enumerated fields are defined in the Repository and are specified in conjunction with a Toolkit allow list, selection list, or selection window. See the description of the Enumerated field in "Validation information" in the "Working with Fields" chapter of the *Repository User's Guide* for more information.

If you use an enumerated field that is *not* associated with an allow list or selection list in a report, ReportWriter just displays blanks for your enumerated string value. (ReportWriter can't access the window entry text for windows that are predefined or built on-the-fly.)

The alpha string is used when the enumerated field is selected as any of the following:

- field to print
- field to sort
- selection field
- break line field
- header/footer field
- conditional field

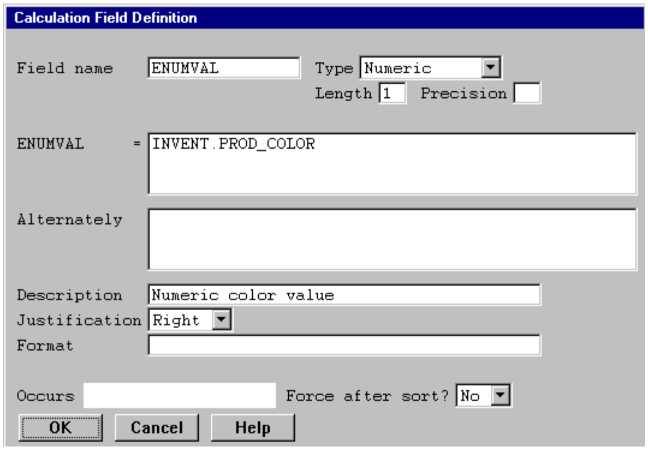
The numeric value is used when the enumerated field is used in any of the following:

- calculation field expression
- subscript to an arrayed field
- range offset or value

To use the numeric value from the file for printing, selections, or sorting, create a calculation field that assigns the numeric value into a temporary field.

When an enumerated field is used in selection statements or conditionals, the comparison is always case insensitive; therefore, you should not enclose the comparison string in quotation marks.

For example, when you use the calculation field defined in figure 6-1, you get the numbers 1, 2, 3, and so on, instead of blue, red, yellow, and so on. This temporary field can be used for printing, selecting, sorting, or in conditionals.



Calculation Field Definition

Field name: ENUMVAL
Type: Numeric
Length: 1
Precision:

ENUMVAL = INVENT.PROD_COLOR

Alternately:

Description: Numeric color value
Justification: Right
Format:

Occurs:
Force after sort? No

Figure 6-1. Using an enumerated field.

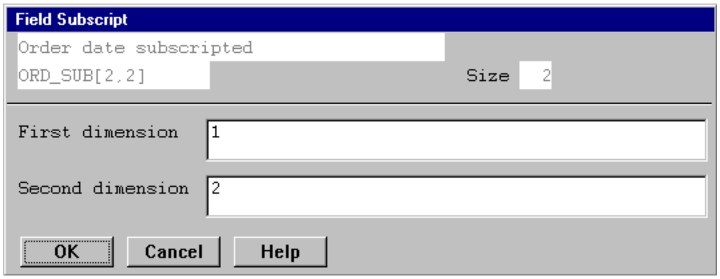
Selecting a Field

To select a field from the list of available fields, highlight the desired field and press ENTER.

If the field you selected is larger than 99 characters, the Field Range window is displayed so you can specify a range of characters to be printed for that field. See "Ranging a selected field" on page 6-8 for details on entering ranging information.

Selecting an arrayed field

If the field you select is an arrayed field (its type and size are preceded by an asterisk in the list of available fields), the Field Subscript window is displayed. This appearance of this window differs depending on the number of dimensions in the array. The maximum number of dimensions is four.



Field Subscript

Order date subscripted

ORD_SUB[2,2] Size 2

First dimension 1

Second dimension 2

Figure 6-2. Defining subscript information.

In each dimension field (First dimension, Second dimension, and so on), do one of the following:

- Enter a literal for each dimension to specify which element this report field is to reference.
- Use a decimal field as a subscript. Enter the field name, or select List selections from the Field functions menu and then select the field from the displayed list.

When a field is used as a subscript, you must specify it by its name, not its description. If another field in another file has the same name, the filename, followed by a period (.), must precede the field name (for example, **PRODUCT.TAX**, where **PRODUCT** is the filename and **TAX** is the name of the field).

If a field's file has more than one structure assigned to it, the structure name may need to be specified between the filename and the field name to avoid ambiguity. The filename, the structure name, and the field name must be separated by periods.

To prevent a runtime "Invalid Subscript specified" error, make sure that numeric fields used as subscript values will contain data that is within the proper bounds. Use selection criteria to impose the correct range, if necessary.

For the arrayed field shown in figure 6-2, two dimension fields (First dimension and Second dimension) appear in the Field Subscript input window. In a two-dimensional array, the total number of elements in four. (The [2,2] after the field name indicates that the array has two rows and two columns.) Entering **1** *in the First dimension field and* **2** *in the Second dimension field specifies that this report field references the second element of the first dimension, as illustrated in figure 6-3.*

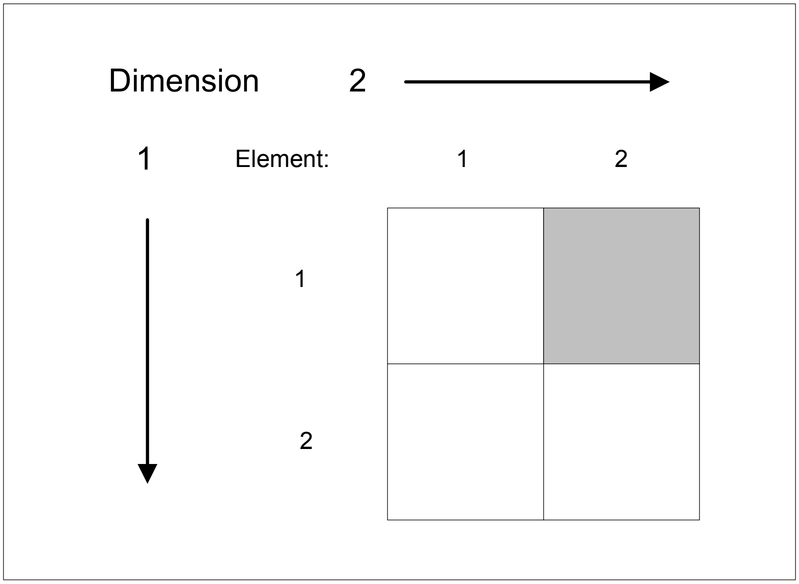


Figure 6-3. An arrayed field with two dimensions.

Ranging a selected field

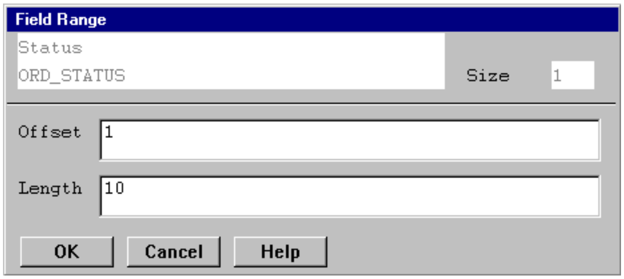
If the size of the field you select is larger than 99 characters, the Field Range window is automatically displayed. (See figure 6-4.) This window contains the description, name, and size of the current field. Because the maximum field size supported in ReportWriter is 99, you must specify a subset, or range, of characters to use from this field.

Enter data in each field as instructed below.

Offset. Enter the starting character position for this range within the field's data size, or press the List selections shortcut to select a field to use as a variable offset (which is base one from field position one).

Length. Enter the length of the range within the field's data size, or press the List selections shortcut to select a field to use as a variable length. The range length cannot exceed 99 characters or the result of the following operation:

```
field data size – offset + 1
```



Field Range
Status
ORD_STATUS
Size 1
Offset 1
Length 10

Figure 6-4. Defining ranging information.

When a field is used as an offset or length, you must specify it by its name, not its description. If another field in another file has the same name, the filename, followed by a period (.), must precede the field name (for example, **PRODUCT.TAX**, where **PRODUCT** is the filename and **TAX** is the name of the field).

If a fields file has more than one structure assigned to it, the structure name may need to be specified between the filename and the field name to avoid ambiguity. The filename, the structure name, and the field name must be separated by periods.

7

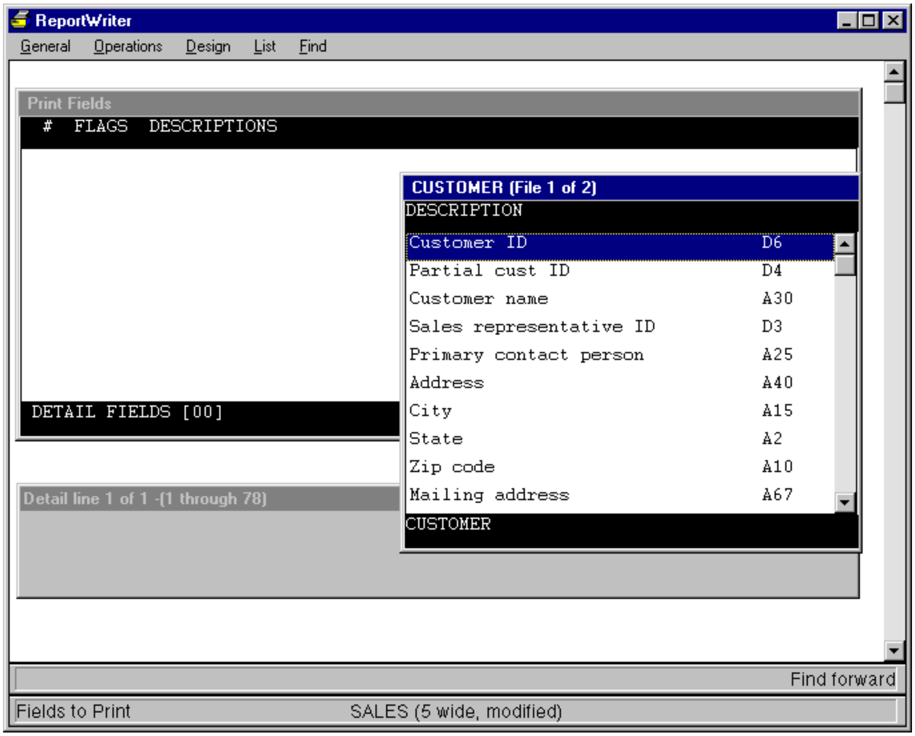
Designing Your Report

Selecting a Field to Print

When you exit the Files to read function, the Design menu is again pulled down, but this time Fields to print is highlighted. This entry enables you to select the data fields that will be printed in your report and to determine where in the report they will be printed. It also enables you to determine which fields will be totaled, to modify field headers and formats, to strip trailing blanks from specific fields, and to specify a conditional to determine whether or not a given field will be printed in a given report record.

Note that you can select the Fields to print function (or press its shortcut) from any of the Design function windows. Thus, you can easily add print fields while specifying other report information.

1. Select Fields to print from the menu. If you're creating a new report, two lists and a window are displayed on the screen.



```
 #  FLAGS  DESCRIPTIONS

DETAIL FIELDS [00]
```

```
CUSTOMER (File 1 of 2)
DESCRIPTION
Customer ID                     D6
Partial cust ID                 D4
Customer name                   A30
Sales representative ID         D3
Primary contact person          A25
Address                         A40
City                            A15
State                           A2
Zip code                        A10
Mailing address                 A67
CUSTOMER
```

```
Fields to Print                 SALES (5 wide, modified)
```

Figure 7-1. Selecting fields to print.

- The Print Fields list occupies the majority of the screen. If you're creating a new report, it is empty. As you select field names from the list of available fields, they're placed in the Print Fields list.

- On the right side of the screen is the list of available fields, which is described in detail in chapter 6, "Choosing Fields." This list contains the fields that are available from the primary file you selected using the Files to read function.
- Near the bottom of the screen is a window that shows you the current layout of a detail record line in your report. Each field is represented by its format, if one exists. Otherwise, each field is represented by its default format:

@s	Alpha, user, and enumerated fields
Zs	Numeric fields
MM/DD/YY	Date fields
HH:MM	Time fields

  If you're creating a new report, this window is empty.

  The title of the layout window tells you which detail record line you're currently defining. (For example, the title would be "Detail line 1 of 2" if you were on the first line of a two-line record.) Your report can contain up to 10 lines per record.

  Notice that the layout window shows you the layout of the current line and tells you the range of characters being displayed. (For example, if the 79th through 158th characters were displayed on the first and only layout line, the window would be titled "Detail line 1 of 1 – (79 through 158).") The current field highlighted in the Print Fields list is also highlighted in the layout window.

When you first enter the Fields to print function for a new report, the cursor is in the list of available fields. This list is normally invoked by the Add field or Add fields function. However, when no fields to print have been selected, this list is displayed for you automatically.

2. From the list of available fields, select the first field you want to include in your report. If you need assistance, see "Selecting a Field" on page 6-6.

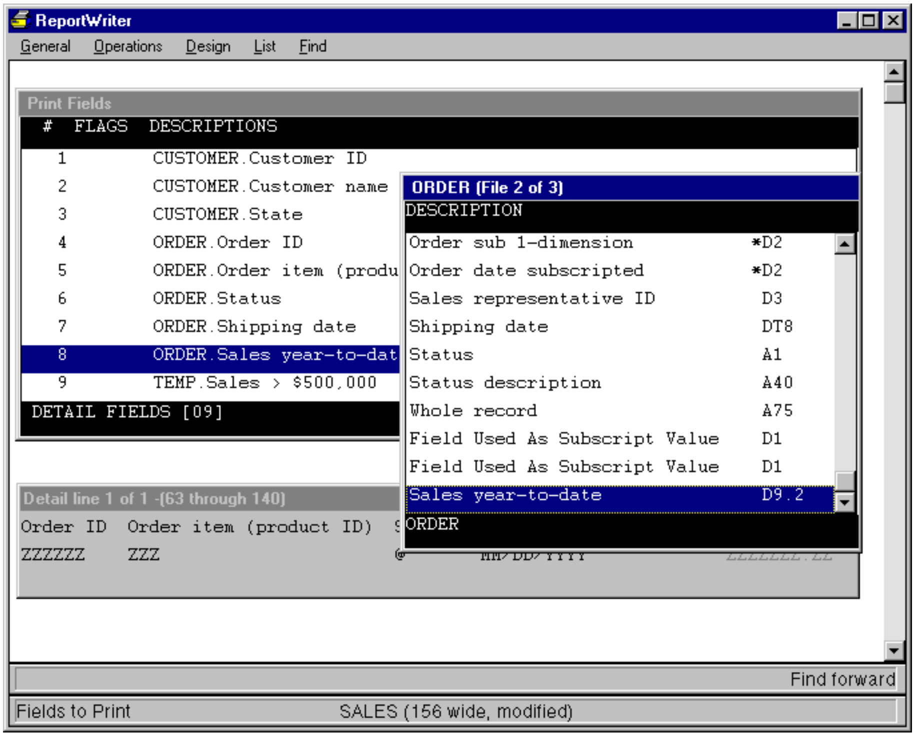
When you select a field, that field appears in the Print Fields list, and it is also displayed in report form in the layout window at the bottom of your screen. If you want to select a field from a different file, select Next report file from the List menu to display the next list of available fields.

3. Continue selecting fields.

As you select fields, the layout window automatically scrolls so that the currently selected field is always visible. You can include up to 99 fields per line, or a total of 990 fields in your report.

The total number of characters in the fields you choose (plus spacing) on the current line is computed and displayed in the lower right corner of the screen. This number is the width of the current line. (When you're in a function other than Fields to print, this number represents the maximum line size for the report.) Your report can have a maximum width of 255 characters.

We'll continue the example from the preceding section. If Customer master is our primary file, available fields might be customer ID, customer name, address, and so forth. Let's say we decide to print the fields customer ID, customer name, sales representative, state, order ID number, order item, order status, shipping date, and sales year-to-date in our report. We also want to print a flag *to mark sales figures that are $500,000 or more. (The example in “Creating a Text Field” on page 8-21 explains how to define such a flag.) See figure 7-2.*



```
Print Fields
 #  FLAGS  DESCRIPTIONS
 1         CUSTOMER.Customer ID
 2         CUSTOMER.Customer name
 3         CUSTOMER.State
 4         ORDER.Order ID
 5         ORDER.Order item (produ
 6         ORDER.Status
 7         ORDER.Shipping date
 8         ORDER.Sales year-to-dat
 9         TEMP.Sales > $500,000
DETAIL FIELDS [09]

ORDER (File 2 of 3)
DESCRIPTION
Order sub 1-dimension          *D2
Order date subscripted         *D2
Sales representative ID         D3
Shipping date                   DT8
Status                          A1
Status description              A40
Whole record                    A75
Field Used As Subscript Value   D1
Field Used As Subscript Value   D1
Sales year-to-date              D9.2
ORDER

Detail line 1 of 1 -(63 through 140)
Order ID  Order item (product ID)  S
ZZZZZZ    ZZZ                      @      MM/DD/YYYY          ZZZZZZZ.ZZ

                                                    Find forward
Fields to Print              SALES (156 wide, modified)
```

Figure 7-2. Selecting fields to print.

4. Press the Exit shortcut when you’re finished.

Once you exit the list of available fields, if you want to select additional fields to print, you have to use the Add field or Add fields function.

Adding a single field

To add *one* additional field to the Print Fields list (if the list of available fields is not already displayed),

1. From the Field functions menu, select Add field. The list of available fields is displayed.

2. Select the field you want to include in your report.

That field is added immediately after the highlighted field in the Print Fields list. The list of available fields is removed from the screen.

Adding multiple fields

To add more than one field to the list of fields to print (if the list of available fields is not already displayed),

1. From the Field functions menu, select Add fields. The list of available fields is displayed.
2. Select a field you want to include in your report.

   The field is added immediately after the highlighted field in the Print Fields list. The list of available fields remains on the screen and the next field in the list is highlighted.
3. Select any additional fields to print.
4. When you're done, press the Exit shortcut to remove the list of available fields.

Using multiple detail lines

Adding a new detail line

If you're creating a new report, the first detail record line is created automatically. You can add additional detail lines to each report record. A report record can have up to 10 detail lines.

1. From the Line functions menu, select Create new line.

   A new detail record line is inserted after the current line, a new Print Fields list is displayed for the new line, and the list of available fields is displayed so you can choose the fields to print. The layout window at the bottom of the screen now says "Detail line 2 of 2" (or whatever line number you've just created).
2. Select any fields you'd like to print on the new detail line.

   If you exit without selecting any fields on the new line, that line will be deleted. (To create blank lines between records, see "Specifying Miscellaneous Design Options" on page 7-25.)

Moving between detail lines

The Print Fields list contains the fields to print on the current line only. The layout window at the bottom of the screen also shows the layout of the current line only. All Fields to print operations apply only to the current line.

To move between lines, select Next line or Previous line from the Line functions menu. The line to which you've just moved becomes the current line.

Deleting a detail line

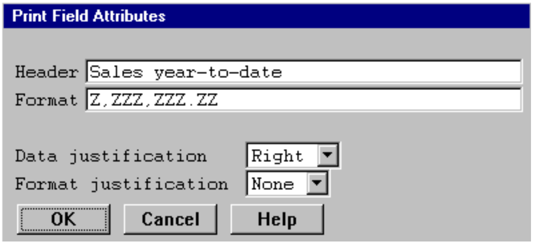
To delete the detail line that is currently displayed in the layout window, select Delete current line from the Line functions menu. If the deleted line was not the last line, the line that followed it is now the current line. If the deleted line was the last line, the line that preceded it is now the current line.

Modifying a Field to Print

Editing a print field’s attributes

You can modify a field’s header, format, data justification, or format justification.

1. Highlight the field in the Print Fields list and press ENTER.



Header Sales year-to-date
Format Z,ZZZ,ZZZ.ZZ
Data justification Right
Format justification None

Figure 7-3. Editing a print field’s attributes.

2. Modify the fields in the Print Field Attributes window as necessary. See the following sections for instructions.

   Header. See “Modifying a field header” on page 7-6.

   Format. See “Modifying a field format” on page 7-7.

   Data justification. See “Modifying data justification” on page 7-8.

   Format justification. See “Modifying format justification” on page 7-8.

3. When you’re finished making changes, exit the window. The new header and format are displayed in the layout window.

Modifying a field header

Field headers are displayed at the top of each page in your report. The default field header is either the header that was specified when the field was defined in the repository or, for temporary fields, the description that you specified. (See chapter 8 for more information about temporary fields.)

A field header can have a maximum of three lines. To begin a new header line, place a caret (^) in the header at the point where you want to split the line. If you want your header to contain an actual caret character, precede the caret with a backslash (\^). Two backslashes in a row (\\) cause a backslash character to be included in the header.

If no field header exists in ReportWriter or the repository, the header defaults to the field description. If the field does not have a description, the field name is used. Field headers for the current line are displayed in the layout window at the bottom of your screen.

Once you've modified the page header, ReportWriter no longer updates the header automatically and you may not necessarily see the original header line for the current detail line. (See the note on page 7-20.)

If you want to replace the existing header completely, begin typing a new header at the Header prompt. To insert text in an existing header, use the arrow keys to position the cursor to the appropriate location, and then begin typing. (To insert text at the beginning of the header, first press the RIGHT ARROW key and then press the LEFT ARROW key before you begin typing; if you don't, the entire header will be erased.)

Once you've selected a field to print, any modifications you make to the field's header affect only that selected field. If you select the same field again, its header still defaults to the header specified in the repository or to the field description, if the field is a temporary field.

Modifying a field format

Field formats define how data is redisplayed or reprinted in a report. For example, a telephone number is stored internally as 10 characters, but it can be redisplayed in your report as 12 characters (*xxx*/*xxx*-*xxxx*). ReportWriter uses the format that was specified when the field was defined in the repository, unless you override that format here.

If a field has a format, that format is displayed in the layout window at the bottom of the screen. If no format exists, ReportWriter assigns a default format to the field. Refer to the table below.

Format character	Meaning
@	Alpha, user, and enumerated fields
Z	Numeric fields
PP/YY	4-byte date fields
MM/DD/YY	5- or 6-byte date fields
PP/YYYY	6-byte period date fields
MM/DD/YYYY	7- or 8-byte date fields
24:MM	4-byte time fields
24:MM:SS	6-byte time fields

To replace the existing format, type a new format in the Format field or select List selections from the Format functions menu to display a list of available formats from the repository. Highlight the format you want to select and press ENTER. For date and time fields, the format list displays only those formats that are appropriate to the data type. You should always select from the format list for date and time fields.

See "Appendix D: Data Formats" for a more detailed explanation of data formatting characters.

For date and time fields, the formats in the format list are predefined by Repository and actually reorder the data being displayed. (For example, a date stored as YYMMDD can be displayed as MM/DD/YY.) If you enter your own format, ReportWriter treats the field as a normal decimal field and does not reorder the data. (See "Appendix B: Date and Time Formats" for a list of predefined redisplay formats.)

To insert text in an existing format, use the arrow keys to position the cursor in the appropriate location and begin typing. (To insert text at the beginning of the format, first press the RIGHT ARROW key and then press the LEFT ARROW key before you begin typing; if you don't, the entire format will be erased.)

As an example, we defined a dollar format for the Sales year-to-date field. Since that field is a D9.2 field and it will contain a monetary value, we used the format **Z,ZZZ,ZZZ.ZZ***, as shown in figure 7-3.*

Once you've selected a field to print, any modifications you make to the field's format will only affect that selected field. If you select the same field again, its format still defaults to the format specified in the repository or to the ReportWriter default format.

Modifying data justification

Alpha data can be left-justified, right-justified, or centered within a field's display range, while numeric data must be right-justified. The data is justified within the field's display range. The width of the display range is determined by either the field header or the field length, depending on which is longer. (The field length is either the length of the format, if one exists, or the length of the field.)

To change the default data justification at the Data justification prompt, press any key to display a selection list, and select the justification you want. Valid options are Left, Right, and Center.

Modifying format justification

You can also specify whether the field's format is left-justified or right-justified within the field before it's applied to the data. (In other words, if the format is too long, specifying **Left** at the Format justification prompt causes it to be truncated on the right, while specifying **Right** causes it to be truncated on the left.)

To change the default format justification at the Format justification prompt, press any key to display a selection list and select the format justification you want. Valid options are None, Left, and Right.

Moving a field

Reordering your fields

Once you've selected more than one field, you can move a field to a different position.

1. Highlight the field you want to move.
2. From the Field functions menu, select Reorder fields.

   The field is now enclosed in square brackets ([]). (To see the close bracket, press the Scroll right shortcut.)
3. Use the UP ARROW or DOWN ARROW key to move the field up or down in the list.
4. When the field is where you want it, select Reorder fields again.

   The field's location also changes in the report layout at the bottom of the screen.

   As you move through the Print Fields list, the contents of the layout window automatically scroll to the right or to the left, so that the highlighted field is always visible.

Modifying the field position

When you select a field to print, ReportWriter places it two spaces away from the previous field by default. You can modify this default positioning and even overlap fields, if you wish.

1. Highlight the field you want to reposition in the Print Fields list.
2. From the Field functions menu, select Arrange field position.

   A ruler displays at the top of the layout window and the header line for the detail record line is shown. If you have modified the page header, this header line remains fixed as you reposition the field.
3. Press the LEFT ARROW key to move the field to the left (delete spaces in front of the field). Press the RIGHT ARROW key to move the field to the right (insert spaces in front of the field). When you move a field, all fields that follow that field also move.

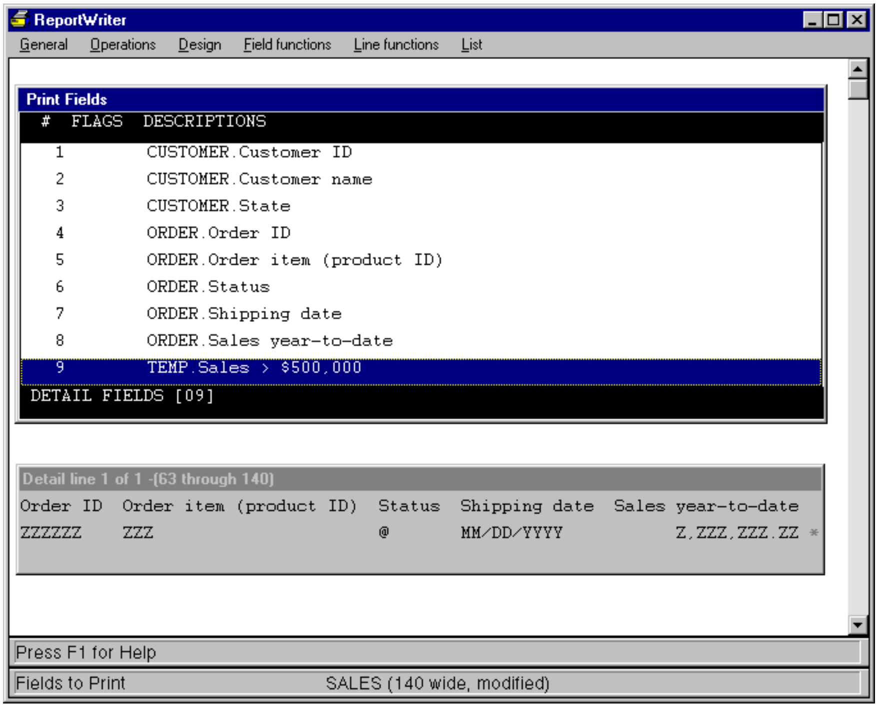
   In figure 7-4, we modified the position of the Sales > $500,000 text field (which is an asterisk flag) to be one character away from the Sales year-to-date field. Note that we also removed the header for this field.
4. Press the Exit shortcut to exit field positioning mode.

Stripping trailing blanks

ReportWriter can automatically strip trailing blanks from a field, bringing the next field closer.

1. In the Print Fields list highlight the field from which you wish to strip trailing blanks.
2. From the Field functions menu, select Strip trailing blanks.

   An input window is displayed.



```
Print Fields
  #  FLAGS  DESCRIPTIONS
  1         CUSTOMER.Customer ID
  2         CUSTOMER.Customer name
  3         CUSTOMER.State
  4         ORDER.Order ID
  5         ORDER.Order item (product ID)
  6         ORDER.Status
  7         ORDER.Shipping date
  8         ORDER.Sales year-to-date
  9         TEMP.Sales > $500,000
 DETAIL FIELDS [09]

Detail line 1 of 1 -(63 through 140)
Order ID  Order item (product ID)  Status  Shipping date  Sales year-to-date
ZZZZZZ    ZZZ                      @       MM/DD/YYYY           Z,ZZZ,ZZZ.ZZ *

Press F1 for Help
Fields to Print                              SALES (140 wide, modified)
```

Figure 7-4. Modifying the position of a field.

3. At the Strip character prompt, enter the character that you want to follow the blank-stripped field.

For example, you might want to strip blank spaces after a city name so the state code will follow immediately. In this case, you'd probably want your strip character to be a comma (**,**) since a comma usually appears between city and state. The default strip character is one blank space.

4. Exit the window.

The strip character that you enter appears in the layout window after the stripped field, and a letter **S** displays in the FLAGS column for that field in the Print Fields list. If your strip character is non-blank, all trailing blanks are replaced with one occurrence of the specified strip character and an additional space in your report.

The Strip trailing blanks menu entry is a toggle. If you select it a second time, blank stripping is turned off.

Blank stripping only affects the field immediately following the stripped field; all subsequent fields are printed in their original locations. Blank stripping does not affect field headers.

Modifying subscripting of an arrayed field

1. Highlight the field in the Print Fields list.
2. From the Field functions menu, select Change subscripting to display the Field Subscript window.
3. See "Selecting an arrayed field" on page 6-6 for details on entering subscript information for an arrayed field.

Modifying ranging

Sometimes you will want to print only a portion of a field. You can do this by specifying the range of characters in a field that you want to print.

1. Highlight the field in the Print Fields list.
2. From the Field functions menu, select Change ranging.

   The Field Range window is displayed. (If your field is larger than 99 characters, this window is displayed automatically.)
3. Enter the offset and length as instructed below.

   Offset. Enter the starting character position for the range within the field's data size, or press the List selections shortcut to select a field to use as a variable offset (which is base one from field position one).

   Length. Enter the length of the range within the field's data size, or press the List selections shortcut to select a field to use as a variable length. The range length cannot exceed the maximum field size (99) or the result of the following operation:

   ```
   field data size – offset + 1
   ```
4. Exit the window.

Once you've specified ranging for a field, you cannot clear it without deleting and reselecting the field in your report.

You cannot range a subscripted field. Instead, you must subscript within a calculation field, as shown below, and then range the calculation field.

```
CALC_FIELD = ARRAYED_FIELD[2]
```

Also, if you select an arrayed field as the offset or length, ReportWriter always uses the first element of the array (subscript = 1). To use an element other than the first, you must subscript within a calculation field and then select the calculation field at the Offset or Length prompt.

Deleting a field from a report

From the Field functions menu, select Delete field. The fields in the Print Fields list are renumbered accordingly.

Designating a field to be totaled

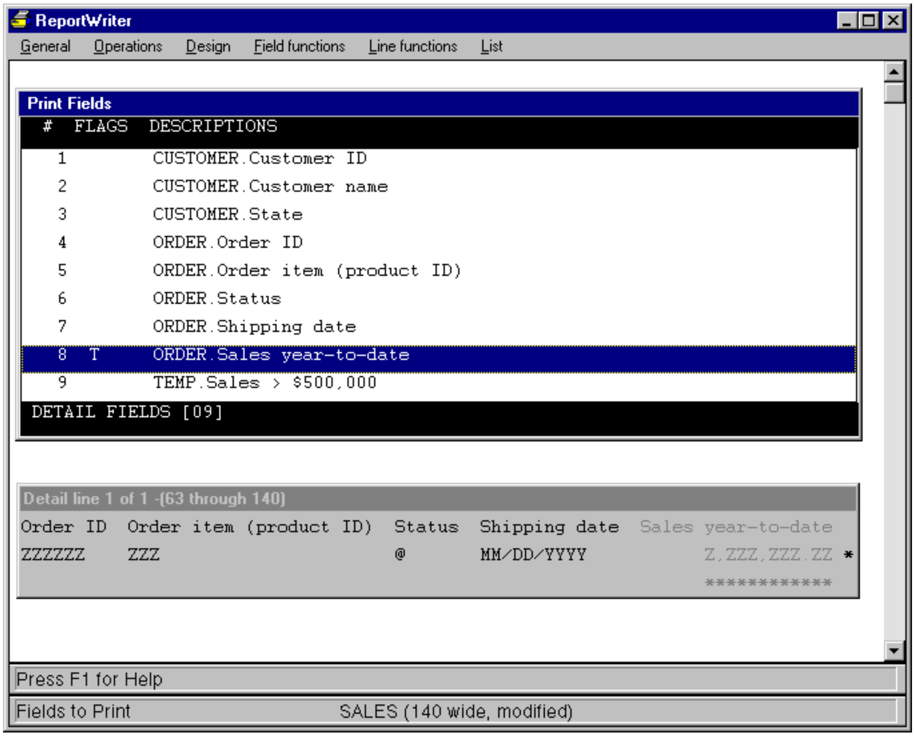
ReportWriter can automatically total any numeric field (decimal, implied-decimal, or integer).

1. Highlight the field in the Print Fields list.
2. From the Field functions menu, select Total field.

This entry is a toggle: select Total field again to turn the totaling function off.

When a field is going to be totaled, it is underscored by a line of asterisks in the report layout window, and a letter T is displayed in the FLAGS column for that field in the Print Fields list.

For our example, we'll designate sales year-to-date as a field to be totaled. (See figure 7-5.)



```
Print Fields
 #  FLAGS  DESCRIPTIONS
  1         CUSTOMER.Customer ID
  2         CUSTOMER.Customer name
  3         CUSTOMER.State
  4         ORDER.Order ID
  5         ORDER.Order item (product ID)
  6         ORDER.Status
  7         ORDER.Shipping date
  8  T      ORDER.Sales year-to-date
  9         TEMP.Sales > $500,000
DETAIL FIELDS [09]

Detail line 1 of 1 -(63 through 140)
Order ID  Order item (product ID)  Status  Shipping date  Sales year-to-date
ZZZZZZ    ZZZ                      @       MM/DD/YYYY           Z,ZZZ,ZZZ.ZZ *
                                                                ************

Press F1 for Help
Fields to Print                SALES (140 wide, modified)
```

Figure 7-5. Designating a field to be totaled.

For information on generating subtotals, see "Setting a report break" on page 10-5.

Every report that you generate contains a report summary line. When the report does not contain fields being totaled, the report summary line contains the following:

- The report description
- The report count (the number of records included in the report)

When one or more fields is being totaled, the report summary line contains the following:

- The report description
- The word “total”
- The report count in parentheses
- The total amounts

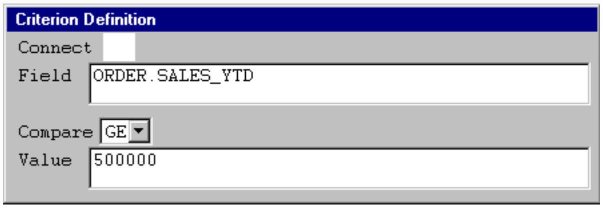
This line is sometimes referred to as the report total or grand total line.

Assigning a conditional to a field

ReportWriter can assign a conditional to a field that will be evaluated before each record. If the condition is true, the field is printed. If the condition is false, the field is not printed. You can connect up to five conditionals together (with AND and OR) and assign them to a single field.

When a conditional is assigned to a field, a letter C displays in the FLAGS column for that field in the Print Fields list.

For example, let’s assume we want to print an asterisk after the Sales year-to-date field, but only if Sales year-to-date contains a value of $500,000 or more. First, we’d define a text field that contains an asterisk. (See “Creating a Text Field” on page 8-21 for information about defining text fields.) Then we’d select that field as a field to print and specify a conditional for it, as illustrated in figure 7-6.



Connect
Field ORDER.SALES_YTD
Compare GE
Value 500000

Figure 7-6. Specifying a conditional to suppress printing.

1. Highlight the field.
2. From the Field functions menu, select Specify conditions to display the Condition Criteria list. As you define conditions, they are added to the end of this list.
    - If there are no conditions defined for the field, the list is empty and the Criterion Definition window is also displayed.
    - If a condition has already been defined for this field, select Add condition from the Condition functions menu to display the Criterion Definition window.

3. Enter data in each field as instructed below.

Connect. If this is the first condition defined for this field, skip this field.

If a condition has already been defined for this field, enter the connection operator (AND or OR) to indicate how this condition is connected to the first condition. AND designates that the two conditions must *both* be true for a record to match. OR designates that a record will be considered to match if one *or* the other condition is true.

Comparisons occur from the top of the list to the bottom. For example,

a AND b OR c

is *not* the same as

b OR c AND a

To define parenthetical type conditionals like (a AND b) OR (c AND d), see "Evaluation order for parenthetical selection statements" on page 9-6.

Field. Enter the name of the field you want to use as a conditional. Note that the fields you specify as conditionals don't have to be fields to print.

To select from a list of available fields that can be used in the conditional, select List selections from the Field functions menu. See "Selecting a Field" on page 6-6 for more information.

If you select an arrayed field for use in a conditional, the Field Subscript window is displayed. See "Selecting an arrayed field" on page 6-6 for details on entering subscript information.

To use only part of a field in a conditional, you can specify a range. See "Modifying ranging" on page 9-9.

Compare. Select one of the comparison operators:

EQ	Equal to
NE	Not equal to
LE	Less than or equal to
LT	Less than
GE	Greater than or equal to
GT	Greater than

Value. Enter either a specific value or the name of a field to which data in your records will be compared. For detailed information about the rules that apply to comparisons, see "Completing the Value field" on page 9-4.

In our asterisk example, we specified **SALES_YTD** *as the field you want to compare,* **GE** *(greater than or equal to) as the comparison operator, and* **500000** *as the comparison value. This means that if the equation (SALES_YTD).GE.{500000} is true (in other words, the value in the Sales year-to-date field is greater than or equal to 500,000), the text field that contains the asterisk will be printed.*

4. Exit the window; the new conditional displays on the Condition Criteria list.

5. To define additional conditions, select Add condition from the Condition functions menu.
6. To return to the list of fields to print, press the Exit shortcut while in the Condition Criteria list.

Defining a conditional by copying an existing one

1. Highlight the conditional you want to copy.
2. From the Condition functions menu, select Copy condition.

   The Criterion Definition window is displayed with a copy of the current conditional.
3. Edit any value, including the connect or comparison operators.
4. Exit the window when you have finished making changes.

   The new conditional is added to the Condition Criteria list.

Editing a conditional

1. Highlight that conditional in the Condition Criteria list.
2. Press ENTER.

   The Criterion Definition window is displayed with the current conditional.
3. Edit any value.

   To display the list of connection or comparison operators, type any letter at the Connect or Compare prompt, respectively.
4. Exit the window when all information is correct.

Reordering conditionals

1. Highlight the conditional you want to move.
2. From the Condition functions menu, select Reorder conditions.

   The conditional is enclosed in square brackets ([ ]). (To see the close bracket, press the Scroll right shortcut.)
3. Use the UP ARROW and DOWN ARROW keys to move the conditional to the desired location.
4. When the conditional is where you want it, select Reorder conditions again.

   If you move the conditional that you created first, or if it is no longer the primary conditional, ReportWriter assigns it a default AND connector.

Deleting a conditional

1. Highlight the conditional in the Condition Criteria list.
2. From the Condition functions menu, select Delete condition.

Assigning a conditional to a detail line

You can assign a conditional to any detail record line. This is useful when you want to suppress one or more detail lines within a multi-line detail record. To suppress single-line detail records or all lines of a multi-line detail record, use the Selections to make function (see "Defining a Selection Criterion" on page 9-2). The detail record line is printed on the report only if the conditions specified in the conditional are true. You can connect up to five conditionals together (with AND and OR) and assign them to a detail line.

1. From the Line functions menu, select Specify line conditions to display the Condition Criteria list.
    - If there are no conditions defined for the detail line, the list is empty and the Criterion Definition window is also displayed.
    - If a condition has already been defined for this detail line, select Add condition from the Condition functions menu to display the Criterion Definition window.
2. See "Assigning a conditional to a field" on page 7-13 for information on completing the fields.

    If you select an arrayed field for use in a conditional, the Field Subscript window is displayed. See "Selecting an arrayed field" on page 6-6 for details on entering subscript information.
3. To save your conditional and return to the Condition Criteria list, exit the window. The word "Conditioned" is displayed at the bottom of the layout window.

Keep the following design limitation in mind: If all lines in a multi-line detail record have conditionals, and all evaluate to false, the record is still counted in the total record count for the report. Use a subtotal access field to access the record count instead.

Modifying a detail line conditional

To copy, edit, move, or delete detail line conditionals, refer to the following sections:

"Defining a conditional by copying an existing one" on page 7-15
"Editing a conditional" on page 7-15
"Reordering conditionals" on page 7-15
"Deleting a conditional" on page 7-15

Creating the Header and Footer Layout

You can specify headers and footers for pages of a report or the entire report. The headers/footers can include text and data fields. For example, you might want to include two question fields that specify a starting and ending date for your report (see "Using a field in your header or footer" on page 7-20 for more information). You can also specify printing control information (such as suppressing the date and page number) for each header or footer with the Header/Footer Miscellaneous Attributes function.

1. From the Design menu, select Header/footer layout.

   The Layout functions menu is pulled down. It contains options for report headers and footers, page headers and footers, and miscellaneous header/footer attributes.

2. Select the entry you want to define. A text editing window is displayed.

Using the header/footer editing window

When you select a header/footer menu entry, a header/footer text editing window is displayed. There's a ruler at the top of the window, so you can see the exact placement of each header or footer character. The window scrolls between 1 and 255 positions.

This window supports three input modes—command, insert, and overstrike:

- When you're in *command* mode, you can use any of the menu entries.
- When you're in *insert* mode, you can insert text in the text editing window. You cannot make selections from the menus.
- When you're in *overstrike* mode, you can type over text in the text editing window. You cannot make selections from the menus.

If your terminal allows it, the cursor blinks when you're in insert or overstrike mode, but not when you're in command mode.

When the header/footer text editing window first displays, you are in command mode.

- To switch to insert mode, select Insert/overstrike mode from the Edit menu (or press CTRL+K). On UNIX and OpenVMS, the word INSERT appears in the lower right corner of the editing window.
- To switch to overstrike mode, press CTRL+K a second time. (Note that you cannot select Insert/overstrike mode again from the menu because you cannot make selections from the menu once you're in insert mode.) On UNIX and OpenVMS, the word OVERST appears in the lower right corner of the editing window.
- To return to command mode, press the Command mode shortcut, CTRL+F.

Creating a report header

The report header appears at the beginning of your report. The default header is the name of the report, in uppercase letters. The report header can be up to 10 lines long; each line automatically scrolls to 255 columns.

> If you've copied this report from another report, the default header is the name of the *original* report, not the current report.

1. From the Layout functions menu, select Report header.

   The cursor is in the text editing window, and you're in command mode.

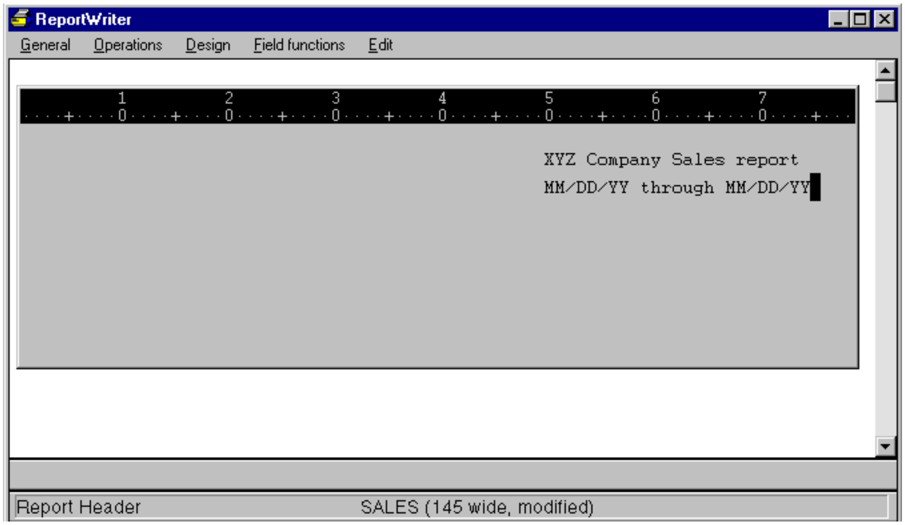
2. From the Edit menu, select Insert/overstrike mode (or press CTRL+K) to go to insert mode. (If you'd rather be in overstrike mode, press CTRL+K while you're in insert mode.)

3. Enter a new header for the report.

   For example, we decided to print the header **XYZ Company Sales Report** *on the first page of our report, with the dates covered in the report on the second line of the header. (See figure 7-7.)*

4. When you've finished editing the header, press CTRL+F to return to command mode.

5. Press the Exit shortcut to close the header window and return to the Layout functions menu.



```
XYZ Company Sales report
MM/DD/YY through MM/DD/YY
```

Report Header SALES (145 wide, modified)

Figure 7-7. Creating a report header.

Creating a report footer

The report footer appears at the end of your report. It can be up to 10 lines long; each line automatically scrolls to 255 columns.

1. From the Layout functions menu, select Report footer.

   The cursor is in the text editing window, and you're in command mode.

2. From the Edit menu, select Insert/overstrike mode to go to insert mode. (If you'd rather be in overstrike mode, press CTRL+K while you're in insert mode.)

3. Enter a footer for the report.

4. When you've finished, press CTRL+F to return to command mode.

5. Press the Exit shortcut to close the footer window and return to the Layout functions menu.

Creating a page header

The page header appears at the top of each page of your report. The default page header is the field header line(s) for your report entries. The page header can be up to 10 lines long; each line automatically scrolls to 255 columns.

1. From the Layout functions menu, select Page header.

   The cursor is in the text editing window, and you're in command mode. A data layout window is displayed at the bottom of the screen. If you're creating a new report, the field header lines are displayed in the text editing window.

2. From the Edit menu, select Insert/overstrike mode to go to insert mode. (If you'd rather be in overstrike mode, press CTRL+K while you're in insert mode.)

3. Enter a new header to be used at the top of each page.

   To assist you in lining up your page header with your detail records, the last six lines of the screen display the layout of your report. The lines scroll upward or downward automatically when you move your cursor to a header line that was not previously visible in the layout window.

   You can view the layout of the pre- and post-break lines by selecting Toggle data line/break lines from the Page header functions menu. The first time you select it, the pre-break line layouts are displayed. The next time you select it, the post-break line layouts are displayed. To return to the detail record layout, select Toggle data line/break lines one more time. (For more information on break lines, see "Creating a Pre- or Post-Break Line" on page 10-8.)

4. When you've finished, press CTRL+F to return to command mode.

5. Press the Exit shortcut to close the page header window and return to the Layout functions menu.

Once you've modified the page header in this window, ReportWriter no longer updates the header automatically when the field header lines change (for example, when fields are moved or deleted, field headers are modified, or lines are inserted and deleted). To reset the page header to the current field header lines and remove any page header modifications, select Reset page header from the Page header functions menu.

Also, once you have modified the page header in this window, you will not necessarily see the original header line for a given detail line in the Fields to Print function. After you modify the page header, a given detail line always displays the correspondingly numbered header line. For example, if detail line 1 has a two-line heading (because of a split heading), after you modify the page heading, detail line 1 will display heading line 1, and detail line 2 will display heading line 2, which is actually the second line of the heading for detail line 1.

Creating a page footer

The page footer appears at the bottom of each page of your report. It can be up to 10 lines long; each line automatically scrolls to 255 columns.

1. From the Layout functions menu, select Page footer.

   The cursor is in the text editing window, and you're in command mode.

2. From the Edit menu, select Insert/overstrike mode to go to insert mode. (If you'd rather be in overstrike mode, press CTRL+K while you're in insert mode.)
3. Enter a new footer for the page.
4. When you've finished, press CTRL+F to return to command mode.
5. Press the Exit shortcut to close the page footer window and return to the Layout functions menu.

Using a field in your header or footer

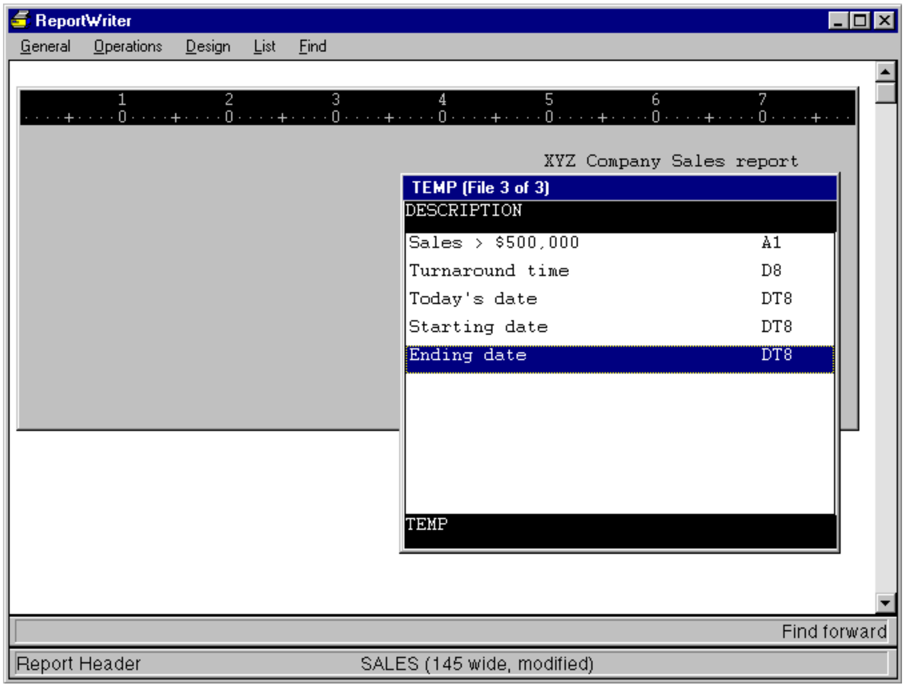
You can include a data field in the report or page header or footer you're currently defining.

1. In the header/footer editing window, move your cursor to the location where you want to insert the field. You cannot insert a field if your cursor is on a field.

   The field will be inserted beginning at the current cursor position and will *overwrite* any text currently in that space. Be careful to leave enough room for the field if you don't want to overwrite existing text.

2. Press CTRL+F to return to command mode (if you're not already in it).
3. From the Field functions menu, select Add field. The list of available fields is displayed.
4. Select the field you want to include in your header or footer. The field is inserted (represented by its data format).

   In the example in figure 7-8, we've inserted the **Starting date** *field on the second line of the report header; gone into insert mode to type a space, followed by the word "through," followed by another*



```
XYZ Company Sales report
TEMP (File 3 of 3)
DESCRIPTION
Sales > $500,000          A1
Turnaround time           D8
Today's date              DT8
Starting date             DT8
Ending date               DT8
TEMP
```

Figure 7-8. Adding a field to a report header.

space; and are about to select **Ending date** *from the list of available fields. (Starting date and Ending date are question fields that we define as an example on page 8-15.)*

Once you add, delete, or edit a field or assign a field conditional to your header or footer, you can no longer abandon your changes.

Viewing a header or footer field

You can display a Print Fields list that contains the fields to be printed in the current header or footer.

1. From the Field functions menu, select Show field names. The Print Fields list is displayed at the bottom of your screen.

   Note: From this list, you can also change field attributes, subscripting, or ranging; specify a conditional; or add or delete a field.

2. Press the Exit shortcut to close the Print Fields list.

Changing a field's attributes

You can change a field's format or the way that format is justified within the field.

1. Place the cursor anywhere within that field in the text editing window.
2. From the Field functions menu, select Change field attributes.

   Note: You can also use the Show field names function to display the Print Fields list, highlight the field in the list, and press ENTER.

   The Print Field Attributes window is displayed.
3. Enter data in each field as instructed below.

   Format. Enter a new display format for this field. You can either type a new format or select a format from a list of available formats using the List selections function.

   Data justification. This value determines how data is justified within the field's display range (the area between the field's starting position and the adjusted field size). Select the justification you want for this field. Valid values are None, Left, Right.

   Format justification. This value determines how the field's format is justified within the field before it's applied to the data. Select the format justification you want for this field. Valid values are None, Left, Right.
4. Exit the window.

Modifying subscripting of an arrayed header/footer field

1. Place the cursor anywhere within the field in the text editing window.

   Note: You can also use the Show field names function to display the Print Fields list and highlight the field in the list.
2. From the Field functions menu, select Change subscripting to display the Field Subscript window.
3. See "Selecting an arrayed field" on page 6-6 for details on entering subscript information for an arrayed field.

Modifying ranging

You can specify that a range of characters in a header/footer field be printed, instead of the whole field.

1. Place the cursor anywhere within the field in the text editing window.

   Note: You can also use the Show field names function to display the Print Fields list and highlight the field in the list.
2. From the Field functions menu, select Change ranging to display the Field Range window.
3. See "Modifying ranging" on page 7-11 for details on entering ranging information for a field.

Assigning a conditional to a header or footer field

You can connect up to five conditionals together (with AND and OR) and assign them to a field to print in a header or footer.

1. Place the cursor anywhere within the field in the text editing window.

   Note: You can use the Show field names function to display the Print Fields list and highlight the field in the list.

2. From the Field functions menu, select Specify conditions.

   - If there are no conditions defined for the field, the list is empty and the Criterion Definition window is also displayed.
   - If a condition has already been defined for this field, select Add condition from the Condition functions menu to display the Criterion Definition window.

3. See “Assigning a conditional to a field” on page 7-13 for information on completing the fields.

Deleting a header or footer field

1. From the Field functions menu, select Show field names.
2. Highlight the field in the Print Fields window.
3. From the Field functions menu, select Delete field.

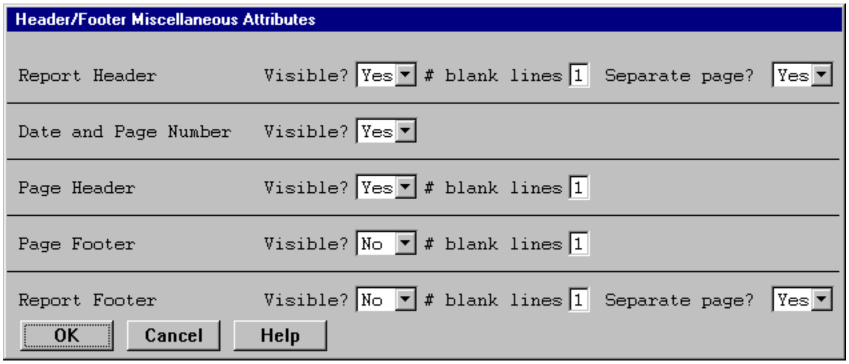
Specifying miscellaneous header and footer attributes

You can specify the following header/footer attributes:

- Whether headers and footers are printed on the report
- The number of blanks lines between the headers and footers and the first or last report entry
- Whether the report header or footer is printed on a page by itself
- Whether the page number and the current date and time are included at the top of each report page (above the page header)

1. From the Layout functions menu, select Miscellaneous.

   Note: You cannot access this function while a header/footer editing window is displayed.



Header/Footer Miscellaneous Attributes

Report Header Visible? Yes # blank lines 1 Separate page? Yes

Date and Page Number Visible? Yes

Page Header Visible? Yes # blank lines 1

Page Footer Visible? No # blank lines 1

Report Footer Visible? No # blank lines 1 Separate page? Yes

Figure 7-9. Specifying miscellaneous header and footer attributes.

2. In the Header/Footer Miscellaneous Attributes window, enter data in the fields as instructed below.

   Visible. Enter Yes if you want the specified header or footer and the page number and current date to be printed on each page. Enter No if you don’t want the header, footer, or page number and date printed. The default response is Yes for report headers, page headers, and the page number and date, and No for report and page footers.

   We entered **Yes** *for the report header, since we want it to be printed.*

   # of blank lines. Enter the number of blank lines that should separate the report header or footer from the rest of the report, the page header from the first report entry on each page, or the page footer from the last report entry on each page. The default is one line; the maximum is nine lines. (For report headers and footers, this number is meaningful only if you plan to answer No to the Separate page prompt.)

   We ignored this field for the report header, because we’ll have it printed on a separate page.

   Separate page. Enter Yes if you want to print the report header on a separate page at the start of the report or the report footer on a separate page at the end of the report. Enter No if you want the report header to appear on the first page of report entries or the report footer to appear on the last page of report entries. The default response is Yes. If you specify a separate page, the report header or footer is centered vertically on the page.

   We entered **Yes** *for the report header.*

3. When you’ve finished designing headers and footers, exit the window to return to the Layout functions menu.

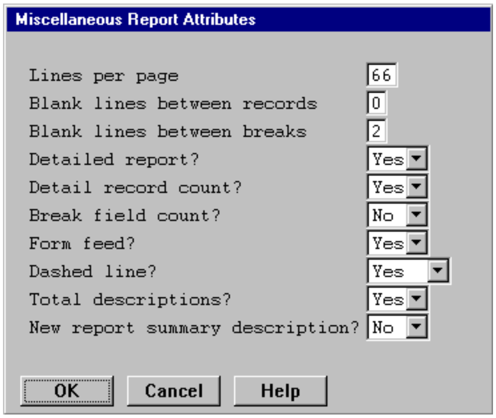
Specifying Miscellaneous Design Options

The Miscellaneous Report Attributes window enables you to modify the following:

- Page size
- The number of blank lines between records and breaks
- Printing detail records and detail record and break counts
- Printing form feeds, dashed lines, and total descriptions
- The report summary description

1. From the Design menu, select Miscellaneous.

The Miscellaneous Report Attributes window displays the default values. (See figure 7-10.)



Miscellaneous Report Attributes

```
Lines per page                    66
Blank lines between records       0
Blank lines between breaks        2
Detailed report?                  Yes
Detail record count?              Yes
Break field count?                No
Form feed?                        Yes
Dashed line?                      Yes
Total descriptions?               Yes
New report summary description?   No
```

Figure 7-10. Defining miscellaneous design features.

2. Enter data in each field as instructed below.

Lines per page. Enter the actual, *physical* size of the printer page in lines (*not* the number of lines printed on the page). The default is 66 lines, which is the number of lines on a standard page (11 inches long, with 6 lines per inch).

Blank lines between records. Enter the number of blank lines to be included between report records. The maximum number of blank lines is 9. The default is 0, which means no blank lines will appear between records.

Blank lines between breaks. Enter the number of blank lines to be included between break sets if the break doesn’t generate a new page. The default is 2.

Detailed report. Select Yes if you want each detail record to be printed on your report. Select No to generate a summary report that prints only the break and report summary lines. (If the report contains fields being totaled, these lines are the subtotal and total lines.) The default is Yes.

Detail record count. Select Yes if you want your report to include the number of detail lines that were printed. Otherwise, select No. The default is Yes.

The detail record count appears at the end of each break section and at the end of the report. If one or more numeric fields are being totaled, the detail record count appears after the total and subtotal descriptions and is enclosed in parentheses.

Break field count. Select Yes if you want your report to include the number of breaks that occurred for each break (sort) level. Otherwise, select No. The default is No.

If your report has only one break level, the break field count appears at the end of the report. If your report has more than one break level, a break field count appears at the end of each break level (except for the lowest level). For example, look at the report on page 8-24, which breaks on state and then on city. The report contains data for three states—California, Florida, and Nevada—so the state break field count would be 3. The report includes data for two cities in California, one city in Florida, and two cities in Nevada, so the city break field counts would be 2, 1, and 2, respectively.

Form feed. Select No if you want ReportWriter to issue blank lines for the remaining lines on a report page rather than issuing a form feed. You can use this feature when printing labels (see "Printing Labels" on page 4-4 for more information). It does not affect the display of the report on the screen. The default is Yes.

Dashed line. Select Blank to replace the dashed lines with blanks. Select No to suppress the printing of the dashed lines above subtotal and total amounts. The default is Yes, which means the dashed lines will be printed.

Total descriptions. Select No if you want to suppress the printing of the break and report summary total descriptions. This includes the word "total" and the parentheses around the record count. The default is Yes, which means the break and report summary total descriptions will be printed.

New report summary description. Select Yes if you want to specify a report summary description other than the default, which is the word "Report." If you select Yes, a second input window is displayed where you can enter the text of the new report summary description. The maximum length of this description is 64 characters. The default is No.

3. Exit the window to save your settings and exit from the Miscellaneous function.

Viewing the Design of Your Report

Any time a complete report is loaded, regardless of what function you're performing, you can view your report's current format. Note that this shows you the format only, not the actual data. To view your report on screen, see "Displaying a Report on the Screen" on page 4-2.

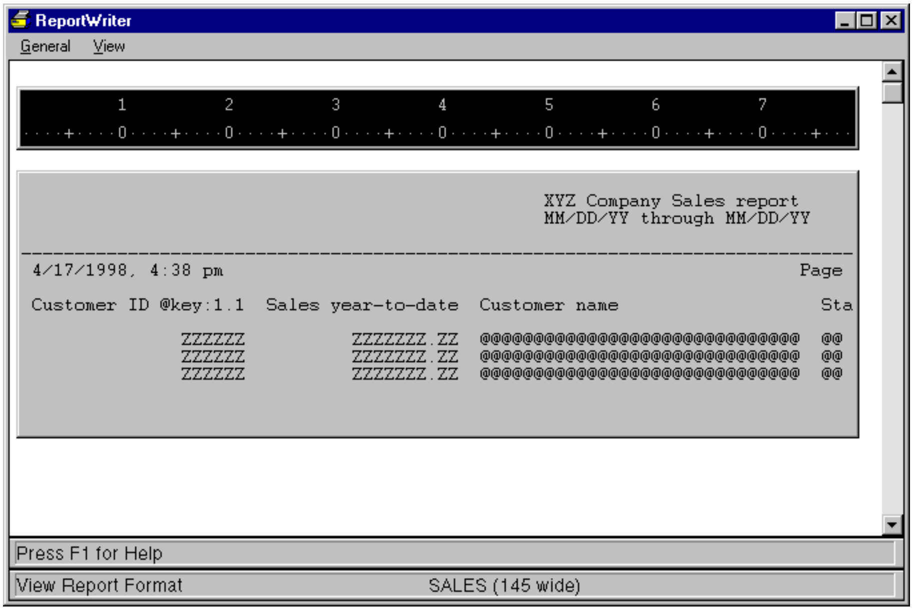
1. From the General menu, select View.

   The report header, date and page number line, page header, data record lines (without actual data), pre- and post-break lines, page footer, and report footer are all displayed. A ruler is displayed at the top of the screen. To remove the ruler, select Toggle ruler from the View menu.

   If the report header and/or footer are to be printed on separate pages, the view window will be divided into logical pages. A dashed line is displayed at the last line of a logical page to indicate a page break. The maximum size of a logical page is 16 lines. If the total size of your page header, date and page number line, data lines, pre- and post-break lines, page footer, and report header and footer (if they are not on separate pages) exceeds 16 lines, the logical page of your report format is truncated. An error message will inform you if this situation occurs.

   You can scroll through the view window using the arrow keys. Additional scrolling functions are available from the View menu column.

2. To exit this function, press the Exit shortcut.



```
                                                XYZ Company Sales report
                                                MM/DD/YY through MM/DD/YY

------------------------------------------------------------------------------
4/17/1998, 4:38 pm                                                      Page

Customer ID @key:1.1  Sales year-to-date  Customer name                   Sta

              ZZZZZZ          ZZZZZZZ.ZZ  @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@  @@
              ZZZZZZ          ZZZZZZZ.ZZ  @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@  @@
              ZZZZZZ          ZZZZZZZ.ZZ  @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@  @@
```

Press F1 for Help

View Report Format SALES (145 wide)

Figure 7-11. Viewing the design of your report.

8

Using Temporary Fields

Introduction to Temporary Fields

After you’ve selected one or more files to read for your report (using the Files to read function), you can create temporary fields. There are five types of temporary fields: text, calculation, question, environment, and subtotal access.

Temporary fields that are created in a report can be selected for printing and sorting and used in a selection criterion or conditional. Numeric temporary fields can also be used as an arrayed field subscript, as a range value, or in a calculation expression.

All temporary fields are placed in a temporary file. This temporary file is available when you select fields to print, fields to sort, or selection criteria, or when you specify a field range or subscript or a value in a calculation or subtotal access field. It appears on the screen as another file in the list of available fields. As you press TAB to cycle through the available report files, the temporary file is the last available file.

When you select a temporary field, we recommend that you always precede the field name with its filename (separated by a period) to avoid possible ambiguity. For example:

TEMP.*fieldname*

where *fieldname* is the name of the temporary field.

Creating a temporary field

1. From the Design menu, select Fields to create.

   You can select the Fields to create function from any of the Design function windows. Thus, you can easily create temporary fields while specifying other report information.

   When you select Fields to create, the Temporary Fields list is displayed. If you are creating a new report or adding temporary fields for the first time, the Available Field Types list is also displayed.

2. Select the type of temporary field you’d like to create from the Available Field Types list. (If this list is not displayed, select Add field from the Field functions menu to display it.)

3. Follow the instructions in this chapter for the type of temporary field you’d like to create:

   Calculation field, page 8-5
   Question field, page 8-15
   Text field, page 8-21
   Environment field, page 8-22
   Subtotal access field, page 8-24

Temporary field names must begin with a letter. The remaining characters can be letters, digits, underscores (_), or dollar signs ($).

Editing a temporary field

1. Highlight the field in the Temporary Fields list.
2. Press ENTER.

   An input window pops up, displaying the current field attributes.
3. Make your changes.

   If you change the expression for a calculation field or the text for a text field, the value in the field is immediately affected.

You cannot change the size of a temporary field (including changing the size of the text string in a text field) if it has been selected to print. To change the size, you must delete all instances of it being printed, edit the temporary field, and then reselect it to print.

Keep in mind that once a temporary field has been selected as a field to print, any changes made to its format (from the Fields to create function) *won't have any effect* on that selected field. However, the new format *will* be assigned to the field if you select it as a field to print *after* you make the changes. Therefore, if you need to change the format, you must first delete the temporary field from the Print Fields list (see page 7-12), make your changes, and then reselect the field.

Another way to modify the format is from the Fields to print function. We don't recommend this method, however, because the temporary field definition still displays the old format, and someone looking at the definition might be confused as to why the field appears differently in the report. We suggest that if you need to modify a format, you always change the format of your original temporary field definition.

Reordering temporary fields

Because temporary fields are evaluated in the order in which they are defined, they must be ordered correctly.

1. Highlight the field you want to move.
2. From the Field functions menu, select Reorder fields.

   The field is now enclosed in square brackets ([ ]).
3. Use the UP ARROW or DOWN ARROW key to move the field up or down in the list.
4. When the field is where you want it, select Reorder fields again.

Deleting a temporary field

Deleting a temporary field *actually deletes the temporary field definition* (unlike deleting a field to print, a field to sort, and so forth, which only removes the field from a list).

A temporary field can be deleted only when all of the following conditions are true:

- It is not selected as a field to print.
- It is not used in a selection criterion or conditional.
- It is not used as a sort field.
- It is not used in a calculation.
- It is not used as a subscript for an arrayed field.
- It is not used as a range value.

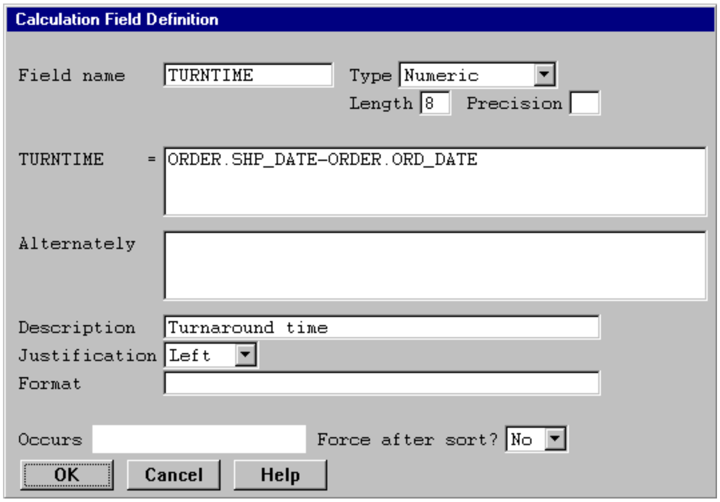
1. Highlight the field in the Temporary Fields list.
2. From the Field functions menu, select Delete field.
3. At the prompt, select Yes to delete the field or No to cancel the deletion.

If you delete all of the files to read in your report, the temporary file and all of its temporary fields are deleted as well.

Creating a Calculation Field

A calculation field defines a mathematical expression, which can be a final calculation or an intermediate value to be used in another calculation.

1. From the Available Field Types list, select Calculation Fields.



Calculation Field Definition

Field name TURNTIME
Type Numeric
Length 8
Precision
TURNTIME = ORDER.SHP_DATE-ORDER.ORD_DATE
Alternately
Description Turnaround time
Justification Left
Format
Occurs
Force after sort? No

Figure 8-1. Defining a calculation field.

2. In the Calculation Field Definition window, enter data in each field as instructed below.

As an example, we've defined a calculation field that contains the turnaround time between the day an order is taken (ORD_DATE) and the day the product is shipped (SHP_DATE).

Field name. Enter a name to identify your temporary calculation field in the temporary file. This name must be unique among all temporary fields.

We called our field **TURNTIME**.

Type. From the displayed selection window, select the data type of the calculation's result:

Alphanumeric
Numeric
Date
Time

You can use alphanumeric calculation fields to assign a string into a field (the calculation field) based on some condition. This calculation field can then be printed and will contain one of two strings based on some set of criteria.

If the result of the calculation is to be a date or time value, keep in mind that only limited calculations can be performed on dates and times. See the table below for the types of calculations that can be performed and the result type. See "Performing calculations on date and time fields" on page 8-9 for more information about date and time calculations.

Calculation	Result type
date field +/– date field = # of days	Numeric
time field +/– time field = # of minutes	Numeric
(date field + time field) – (date field + time field) = # of days – HHMM	Time
date field +/– numeric field(s) or literal(s) = YYMMDD	Date
time field +/– numeric field(s) or literal(s) = # of days – HHMM	Time

Since TURNTIME will contain the difference between two dates, the data type must be **Numeric***.*

Length. Enter the desired length for the calculation's result.

When using date or time fields in your calculation, the length should match the storage length of the base value in the expression. For example, in the fourth rule above, the length of the calculation field should be the same as the length of the date field.

In our example, the length of the field is 8.

Precision. If the result is to be an implied-decimal value, enter the number of digits to the right of the decimal point. The maximum precision is 10. If none of the operands in the calculation expression are implied-decimal, you must multiply by 1.0 to get an implied-decimal result.

Our example field is not implied-decimal, so we left this field blank.

***FIELD NAME* =.** *FIELD NAME* is the name of the field you entered in the Field name field above. (For example, in figure 8-1, the prompt is **TURNTIME =** .) Enter the calculation(s) you want ReportWriter to perform when it encounters this field while generating a report.

Your expression can be composed of

- operands (literals and field names).
- arithmetic operators (**+** for addition, **–** for subtraction, ***** for multiplication, and **/** for division).
- intrinsic functions (%BRKCNT, %DATE, %TIME, %SUM, %RECNUM, and %GOTREC).
- parentheses ["(" and ")"], to specify precedence or to specify a field range of the form (*offset:length*).

You can either type these elements directly or select them from a list. See "Entering an expression" on page 8-10 for information about entering field names, literals, and operands. See "Using an intrinsic function in an expression" on page 8-11 for information about entering intrinsic functions.

If a division operation involves two decimal or integer values, any fractional part of the result will be truncated without rounding. For example, the expression 5/2 yields a result of 2, not 2.5.

Integer values cannot be on the left side of an equation in a calculation field. For example, int*10 gives an incorrect value, but 10*int is correct.

When you multiply or divide implied-decimal operands, the intermediate result of the operation is calculated to a fractional precision of 11 digits. It is then rounded to 10 digits. For example, the expression 2.0/3.0 yields a result of 0.6666666667.

The calculations in this field are performed by default. If the calculation field has an associated conditional, the calculations are performed only if the result of the specified condition is true. See "Assigning a conditional to a calculation field" on page 8-14 for information about assigning a conditional to your calculation field.

The expression in our TURNTIME example is composed of two field names and an operator. To get the time difference between the two dates, we needed to subtract the order date field from the shipping date field, like this:

```
ORDER.SHP_DATE-ORDER.ORD_DATE
```

(First we selected **SHP_DATE** *from the list of available fields using the List selections function, then we typed the minus sign, then we selected* **ORD_DATE** *using List selections.)*

Alternately. If the calculation field is associated with a conditional, you can enter an alternate expression to be evaluated if the specified condition is false.

If you don't specify an alternate expression and the condition is false, no calculation is performed.

If a conditional is associated with this calculation field, the word "Conditioned" is displayed below the Alternately field.

If the alternate expression is similar to the main expression, you can copy the main expression to this field. Select Copy expression from the Field functions menu. You can then select Edit field from the Input menu and use the functions in the Edit menu to complete the alternate expression. The Copy expression function can also be used to copy the contents of the alternate expression to the main expression. To do this, highlight the main expression field and select Copy expression.

Description. Enter a description for the calculation field. The default is the field name. This description is used as the default field header when this field is printed. The description is also used when this field is listed in the Print Fields list, Sort Fields list, or list of available fields.

A good description for our example might be **Turnaround time***.*

Justification. Select Right, Left, or Center to indicate how you want the result of this field justified. Center is allowed only if the type is alphanumeric.

Our calculation field will be left-justified.

Format. Enter the format in which this calculation field should be printed.

To display a list of predefined formats, select List selections from the Field functions menu. If the field is not a date or time field, a list of the global formats from the repository is displayed. If the field is a date or time field, a list of appropriate display formats is displayed. These date and time formats are predefined by ReportWriter and actually reorder the data being displayed. (For example, a date stored as YYMMDD can be displayed as MM/DD/YY.) You should always use a format from the format list for date and time fields. If you enter your own format, ReportWriter treats the field as a normal decimal field and does not reorder the data. (See "Appendix B: Date and Time Formats" for a list of predefined redisplay formats.) Select the format you want to use.

We accepted the default format. (Note that although the two fields involved in the calculation are dates, the result of the calculation is not a date.)

Make sure you read "Editing a temporary field" on page 8-3 for important information about modifying the format of a temporary field after it has been selected as a field to print.

Occurs. This read-only field tells you when the calculation field is evaluated. It either says **Before Selection** (if you selected No in the Force after sort field or if you are creating a new calculation field) or **After Sort** (if you selected Yes in the Force after sort field).

Force after sort. By default, calculation fields are evaluated before the report record is tested against the selection criteria and before the report records are sorted. If you want to force the calculation to occur after the sort, select Yes. The default Force after sort? value is No, unless the calculation field is dependent on one or more subtotal access fields (see page 8-24) or the calculation field references itself. In this case, the Force after sort value defaults to Yes, and you cannot modify this value.

You can cancel creation of your calculation field by pressing the Abandon shortcut.

3. To save your new calculation field definition, exit the window.

If the calculation of a temporary calculation field (FIELD_A) is dependent on the result of another calculation field (FIELD_B), FIELD_B must preceed FIELD_A in the list of temporary fields. See "Reordering temporary fields" on page 8-3 for more information.

Performing calculations on date and time fields

For the calculations below, the resulting field is stored as a decimal amount. You can use a format to modify the display length.

date field +/– date field = # of days

6/25/91 – 6/04/91 = 21

time field +/– time field = # of minutes

12:23 – 10:15 = 128

For the calculation below, the resulting field will be stored as a four-digit number of days and HHMM format for the time portion. The default display format is "Days – HH:MM". If you want to display only the time portion, you can select one of the available time display formats. If the number of days is greater than 9999, it will be truncated.

(*date field + time field*) – (*date field + time field*) = *# of days – HHMM*

(8/01/91 + 10:50) – (7/10/91 + 3:18) = 22 – 7:32

(12/16/90 + 3:00) – (12/13/90 + 7:10) = 2 – 19:50

For the calculation below, ReportWriter assumes that your numeric field or literal represents an amount in days. The resulting field will be stored as YYMMDD, and you can select the appropriate date format.

date field +/– numeric field(s) or literal(s) = YYMMDD

3/26/91 + BILLING:LTPER = 4/05/91

3/26/91 + 10 = 4/05/91

For the calculation below, the resulting field will be stored as a four-digit number of days with HHMM format for the time portion. The default display format is "Days – HH:MM". If you want to display only the time portion, you can select one of the available time display formats. If the number of days is greater than 9999, it will be truncated.

time field +/– numeric field(s) or literal(s) = # of days – HHMM

20:10 – TRANS:60MIN = 19:10

20:10 – 60 = 19:10

Neither YYPP nor YYYYPP are supported as operands within an expression. That is, you can't add or subtract them from other dates or from numeric fields or literals.

If you take the difference between a **d6** time field and a **d4** time field (HHMM) or between two **d6** time fields, ReportWriter assumes that the seconds are 0 for both fields.

Entering an expression

The information in this section applies to both primary and alternate expressions.

Evaluation order

Multiplication and division operations are normally evaluated before addition and subtraction operations; otherwise, the expression is evaluated from left to right. You can use parentheses to override these precedence rules and indicate which portion(s) of the expression you want evaluated first.

For example, consider the expression

6 – 2 * 3

The 2 * 3 (which equals 6) is evaluated first, because it is a multiplication operation. This leaves us an operation of 6 – 6 which equals 0. However, if we specify our expression as

(6 – 2) * 3

the 6 – 2 (which equals 4) is evaluated first, because it's enclosed in parentheses. This leaves us an operation of 4 * 3, which equals 12. As you can see, the result can be quite different when you change the precedence order.

Using a field in an expression

You can either type in a field name for inclusion in your calculation field definition or select one from a list by selecting List selections from the Field functions menu. You can select fields from any of the available lists, including the list from the temporary file. See chapter 6 for more information about selecting fields.

When you select a field, its name is inserted at the current cursor position in the *FIELD NAME* = field and the cursor is repositioned after the field. The filename precedes the field name to avoid ambiguity. (If the field you select is an arrayed field, see page 6-6.)

For ranging a field within the expression, use a colon (:) as the delimiter. For example:

FIRM(3:10)

Using a literal in an expression

You can also include a literal in your expression by typing it directly. For example, you might want to calculate a percentage of some field by multiplying the literal percentage value by that field. If you enter an implied-decimal value in an expression, you must place at least one digit before the decimal point. Alpha literals must be enclosed in quotation marks.

Using an operator in an expression

To include an operator in your calculation field definition, either type it in or select List operators from the Field functions menu. Highlight the desired operator and press ENTER. The selected operator is inserted at the current cursor position in the *FIELD NAME* = field, and the cursor is repositioned after the operator.

Using an intrinsic function in an expression

ReportWriter supports six intrinsic functions in a calculation field's expression:

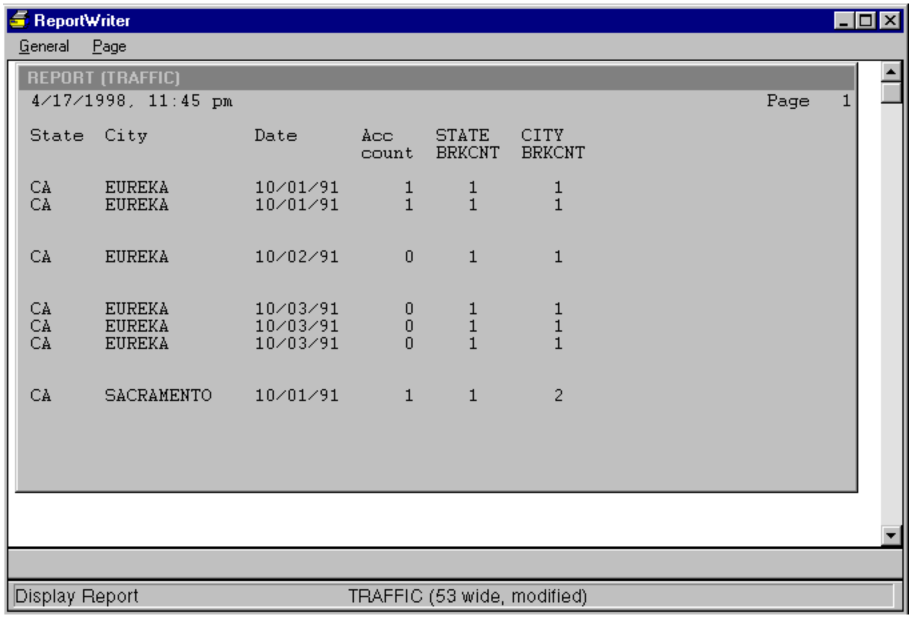
- %BRKCNT
- %DATE
- %TIME
- %RECNUM
- %SUM
- %GOTREC

Using intrinsics creates temporary fields that can then be printed, used in conditionals and selection criteria, or used in other calculations.

%BRKCNT(*break_field*)

Returns the current number of different values of a given break field. For example, if your report is sorted and breaks on customer name, the %BRKCNT(CUSTOMER_NAME) field always contains the number of *different* customer names that have occurred so far (the number of breaks on customer name).

%BRKCNT can only be used with break fields and is always reset when a higher level break occurs. For example, if your report is sorting first on state and then on city, %BRKCNT(CITY) contains the number of different cities that have occurred since the last STATE change. (See figure 8-2.) If you want to calculate cumulative break counts, you can use subtotal access fields.



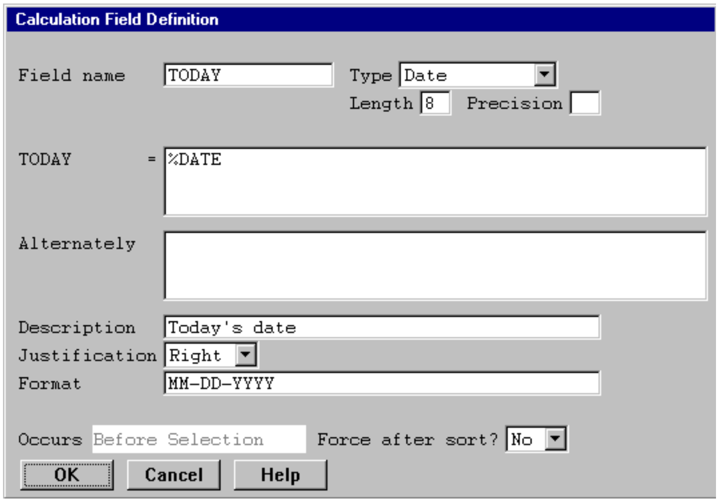
```
REPORT (TRAFFIC)
 4/17/1998, 11:45 pm                                                  Page   1

 State  City           Date       Acc    STATE   CITY
                                  count  BRKCNT  BRKCNT

 CA     EUREKA         10/01/91      1      1       1
 CA     EUREKA         10/01/91      1      1       1


 CA     EUREKA         10/02/91      0      1       1


 CA     EUREKA         10/03/91      0      1       1
 CA     EUREKA         10/03/91      0      1       1
 CA     EUREKA         10/03/91      0      1       1


 CA     SACRAMENTO     10/01/91      1      1       2
```

Display Report TRAFFIC (53 wide, modified)

Figure 8-2. Using the %BRKCNT intrinsic function.

%DATE

Returns the current date as an eight-digit value of the storage format YYYYMMDD. To use this function in a calculation field, select a type of **Date** and a length of **8**, as shown in figure 8-3.



Field name: TODAY
Type: Date
Length: 8
Precision:

TODAY = %DATE

Alternately:

Description: Today's date
Justification: Right
Format: MM-DD-YYYY

Occurs: Before Selection
Force after sort? No

Figure 8-3. Using the %DATE intrinsic function.

%DATE is not explicitly supported in calculations on date fields. For example, you can't subtract %DATE from another date. You must first assign %DATE into a calculation field whose type is Date. You can then use that calculation field in another calculation field that subtracts one field from another.

%TIME

Returns the current time as a four-digit value of the storage format HHMM. To use this function in a calculation field, select a type of **Time** and a length of **4**.

%RECNUM

Returns the current record number (sequence number) being read in the primary file.

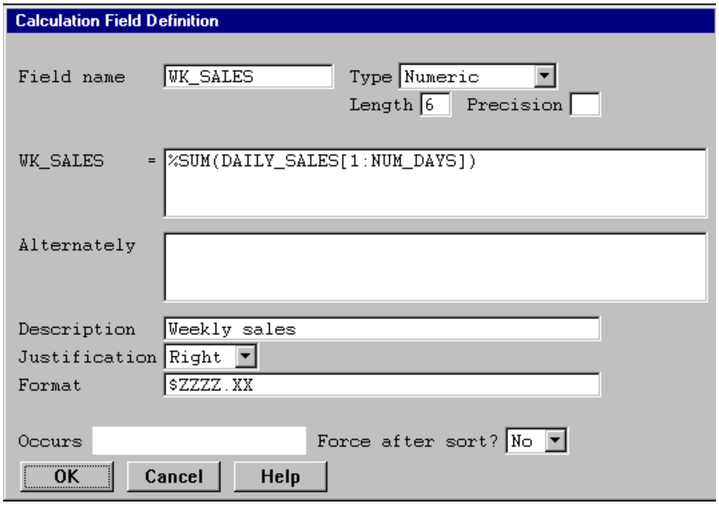
You can use the %RECNUM intrinsic in a calculation field and then use that calculation field in a selection criterion to optimize relative (primary) files. See "Understanding ReportWriter Optimization" on page 18-13 for more information about optimizing files.

%SUM(*array_field[beg:end]*)

Returns the sum of a one-dimensional, numeric array (decimal or implied-decimal), where *array_field* is the name of the array field (with or without the file definition and/or structure name), *beg* is a decimal field or literal that specifies the first element to include in the sum, and *end* is a decimal field or literal that specifies the last element to include in the sum.

If *beg* is greater than *end*, the sum will be 0. If *beg* or *end* is less than 1 or greater than the number of elements in *array_field*, a nonfatal subscript error will be generated.

As an example, we've created a calculation field named **WK_SALES** *that has the characteristics shown in figure 8-4.*



Calculation Field Definition

Field name: WK_SALES
Type: Numeric
Length: 6
Precision:

WK_SALES = %SUM(DAILY_SALES[1:NUM_DAYS])

Alternately:

Description: Weekly sales
Justification: Right
Format: $ZZZZ.XX

Occurs:
Force after sort? No

Figure 8-4. Using the %SUM intrinsic function.

%GOTREC(*file*)

You can use this function when you're doing multiple projections (see page 5-10). %GOTREC returns a value of 1 if a record was read from the specified file or a value of 0 if no record was read.

You can specify *file* as the file definition name, the structure name, or a file definition/structure name combination, as long as *file* is unique to the set of file/structure combinations used in the report.

Once this value has been stored in the calculation field, you can use it as a conditional for selections, printing, or additional calculations.

Assigning a conditional to a calculation field

You can define and assign a conditional to your temporary calculation field from the Calculation Field Definition window.

1. From the Field functions menu, select Specify conditions.

   The Condition Criteria list is displayed. If you haven't defined any conditions for the calculation field, this window is empty and the Criterion Definition window overlays it. As you define conditions, they are added to the Condition Criteria list.

2. See "Assigning a conditional to a field" on page 7-13 for detailed instructions on specifying a conditional.

   Once you've defined the first condition, you can add additional conditions to this list by selecting Add condition from the Condition functions menu. You can connect five conditions together (with AND and OR) and assign them to a calculation field.

   To copy, edit, move, or delete a conditional, refer to the following sections:

   "Defining a conditional by copying an existing one" on page 7-15
   "Editing a conditional" on page 7-15
   "Reordering conditionals" on page 7-15
   "Deleting a conditional" on page 7-15

3. To return to the Calculation Field Definition window, press the Exit shortcut from the Condition Criteria window.

   When a conditional is assigned to a calculation field, the word "Conditioned" is displayed below the Alternately field.

Creating a Question Field

Question fields store answers that the user enters when the report is run. The contents of a question field remain static throughout the generation of the report but are discarded when the report is complete.

Question fields allow a single report definition to be used to generate similar yet differing reports, depending on the user's specific needs or preferences. Some examples of data that might be obtained through question fields are

- desired beginning and ending value for selection purposes.
- other selection criteria, such as minimum balance required.
- balance forward cut-off date or aging date.
- other values to be used in calculations, such as percentage to apply.
- other values that remains constant throughout the report but must be obtained from the user.

1. Select Question Fields from the Available Field Types list to display the Question Field Type window.
2. Select the type of the data to be entered in your question field:

   Alphanumeric
   Numeric
   Date
   Time
   User
   Inherit

   The Question Field Definition window is displayed. See figure 8-5. (Note: If you select **Inherit**, a list of fields is displayed; see "Inherited question fields" on page 8-19 for more information.)

3. Enter data in each field as instructed below.

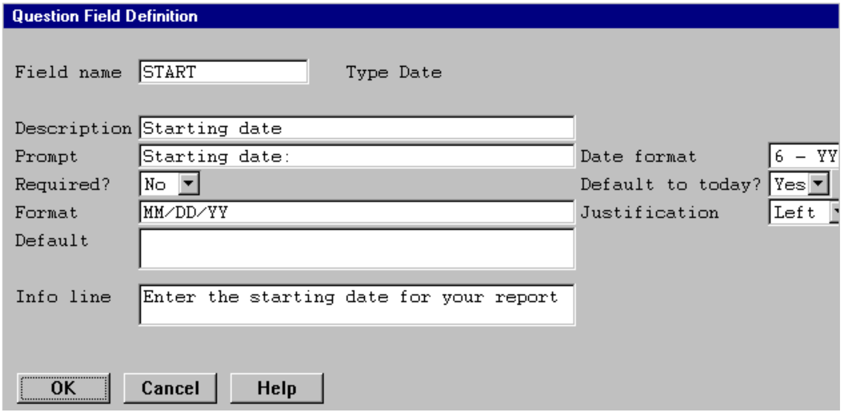
   As an example, let's assume we want the user to define a range of order dates that will determine which records will be included in the report. First, we need to define two question fields to prompt the user for the starting and ending order dates that should be included. (See figure 8-5.) Then we must define selection criteria based on the user's responses to our prompts. (See "Defining a Selection Criterion" on page 9-2.) The prompts that will be displayed to the user at report-generation time are shown in figure 8-6.

   Field name. Enter a name to identify your temporary question field in the temporary file. This name must be unique among all temporary fields.

   Our first question field is called **START** *and our second called* **END**.

   Type. This read-only field displays the data type of the data to be entered in your question field.

   The type is **Date** *for both question fields.*



Question Field Definition

Field name START Type Date

Description Starting date

Prompt Starting date:

Required? No

Format MM/DD/YY

Default

Info line Enter the starting date for your report

Date format 6 - YY

Default to today? Yes

Justification Left

Figure 8-5. Defining a question field.

Description. Enter a description for your question field. The default description is the field name. This description is used as the default field header when this field is printed. The description is also used when this field is listed in the Print Fields list, Sort Fields list, or the list of available fields.

We described our fields as **Starting date** *and* **Ending date**.

Prompt. Enter the question that should prompt the user at report generation time. The maximum size is 40 characters. Don't forget to enter a colon or other punctuation that you want to display.

Our prompt is **Starting date:** *for the first field and* **Ending date:** *for the second field.*

Required. Select Yes if you want the user to be required to enter a value in this field. The default is No.

We want the fields to default to today's date, so no user response is necessary. We've entered **No** *in both instances.*

Format. Enter the display format that should be used if this question field is selected as a field to print. To display a list of existing formats, select List selections from the Field functions menu.

If the field is not a date or time field, a list of the global formats from the repository is displayed. If the field is a date or time format, a list of appropriate predefined display formats is displayed. You should always use a format from the format list for date and time fields. If you enter your own format, ReportWriter treats the field as a normal decimal field. (See "Appendix B: Date and Time Formats" for a list of predefined redisplay formats.)

Select the format you want to use.

We selected **MM/DD/YY** *from the selection list.*

Make sure you read "Editing a temporary field" on page 8-3 for important information about modifying a temporary field's format after it has been selected as a field to print.

Default. Enter the default value, if any, that will be displayed after the prompt when the question is presented to the user at report generation time. Maximum size is 80 characters.

If the question field type is date, you can specify the default value in the same ways the user can enter input in the field when the report is run. (This default value string is not converted to the expanded date until the report is run.) See "If the value is a date" on page 9-5 for a list of valid date formats.

We don't want to display a default until the user presses ENTER*, so we'll just leave this field blank.*

Info line. Enter a message that will be displayed on the information line at the bottom of the screen when the user enters data at report generation time. Maximum size is 80 characters. If you don't enter a message, the information line will be blank.

We'll tell the user what information we're looking for with the messages **Enter the starting date for your report** *and* **Enter the ending date for your report**.

4. To complete the additional fields specific to the data type of your field, refer to the sections below, and then continue with step 5 on page 8-19.

▸ Alphanumeric question fields

If the selected data type is alphanumeric, you must specify the following information:

Length. Enter the length of the question field. If you want to use this value in a selection and you want the comparison to be case sensitive set the length to two characters longer than the desired length, so the user can enclose the value in quotation marks (" ").

Uppercase. Just press ENTER to enter No if you don't want the data the user enters to be forced into uppercase letters. To force all data to uppercase, enter Yes.

Justification. From the displayed selection window, select Left if you want the display of this field to be left-justified or Center if you want the display to be centered.

▸ Numeric question fields

If the selected data type is numeric, you must specify the following information:

Length. Enter the length of the question field.

Precision. Enter the number of characters to the right of the decimal point. The maximum number of characters after the decimal point is 10. The default number is 0.

Decimal required? If a decimal point is required, select Yes from the selection window. The default is No.

Negative allowed? If a negative number is allowed, select Yes from the selection window. The default is No.

Blank if zero? If the question field is to be displayed as blank when it contains a value of 0, select Yes from the selection window. The default is No.

Justification. Select how you want the display to be justified: Left or Right.

Date question fields

If the selected data type is date, you must specify the following information:

Date format. A selection window containing the date storage formats for this question field is displayed. Select how you want the specified data to be stored by ReportWriter during report generation.

We selected date format **6***, which is YYYYMMDD, since the ORD_DATE field to which we're going to compare these fields for selection purposes has a size of 8.*

By default, the user must provide input in the format MMDDYY. See "If the value is a date" on page 9-5 for a list of valid date formats. To change the default date order used when entering dates, set the environment variable RPTDATE. If RPTDATE is set to 1, the input order is DDMMYY. If RPTDATE is set to 2, the input order is YYMMDD.

Default to today? To make this date field default to today's date when the user presses ENTER without entering any data, select Yes. If you don't want the date to default to today, enter No (the default). Note that when the question field appears, today's date will not be displayed in the field until the user presses ENTER. If a default value is specified, the user can still get today's date by clearing the question field and pressing ENTER.

We want our default to be today's date, so we entered **Yes***.*

Justification. select how you want the user's response to be justified: Left or Right.

We decided to left-justify the responses and selected **Left***.*

Time question fields

There are no additional fields. Note that time question fields are limited to a length of 4.

User-defined question fields

If the selected data type is user-defined, you must specify the following information:

Length. Enter the length of the question field. If you want to use this value in a selection and you want the comparison to be case sensitive, set the length to two characters longer than the desired length, so the user can enclose the value in quotation marks (" ").

Uppercase? Just press ENTER to enter No if you don't want the data that the user enters to be forced into uppercase letters. To force all data to uppercase, enter Yes.

Justification. select how you want the display to be justified: Left or Center.

Data. Enter a string of up to 30 characters to identify your user-defined data type.

Text. Enter a user text string of up to 80 characters to be associated with the question field.

When a report contains a user-defined question field, ReportWriter calls the RPS_DATA_METHOD subroutine to format the data that the user enters in that field. ReportWriter also stores the data in that format, in case the data is later used in a selection comparison. (For more information about RPS_DATA_METHOD see page 15-39. If you don't write your own RPS_DATA_METHOD subroutine, the data entered in the user-defined question field will be displayed and stored as entered.)

- **Inherited question fields**

  An inherited question field adopts the characteristics of some other field of your choice. You might want to use this feature when you will be comparing this question field against one of the fields from your data file and you don't want to have to know what the characteristics of that other field are. When you select the data type **Inherit** in the Type field, a list of available fields from which to "inherit" characteristics is displayed. Select a field from the list. That field's characteristics are copied into the question field input window.

  For example, if you select

  - a date field, the data type for the question field you are creating is set to **Date**. The question field also inherits the length and date format of the field you selected, as well as the "Default to today?" flag.
  - an alpha field, the new question field inherits the data type, length, and uppercase value of the field you selected.
  - a decimal field, the question field inherits the data type, length, precision, and decimal required, negative allowed, and blank if zero values of the field you selected.
  - an enumerated field, the inherited question field will have an alpha data type and the same length as the enumerated field.
  - a user-defined field, your question field inherits the data type, length, and data and text strings of the user-defined field.
  - all inherited fields inherit the description, prompt, required flag, format, default value, information line, and justification of the field you selected.

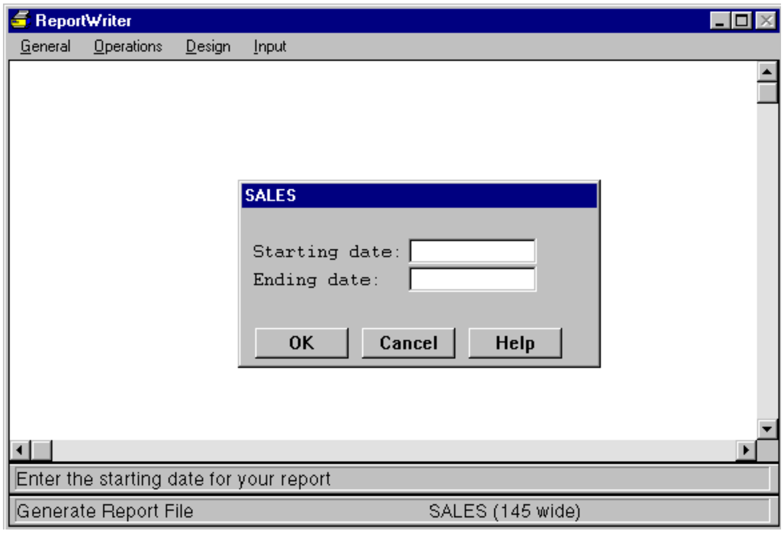
  An inherited time field always has a length of 4, regardless of the length of the field from which it is inherited.

  For our example, we could have created the **Start** *field and then created the* **End** *field with the* **Inherit** *data type.*

5. Exit the window to save your new question field definition. Until you exit the Question Field Definition window, you can cancel creation of the question field by pressing the Abandon shortcut.

   When you generate your report, ReportWriter creates an input window that contains referenced question fields (selected as a field to print or sort or in a calculation expression, selection criterion, subscript, range, or conditional). The fields are displayed in the window according to your definitions, in the order in which they were created. Each prompt appears on a line by itself, with a field for the user's response to the right, as shown in figure 8-6. If all of the questions won't fit into one input window, ReportWriter creates additional windows.

   When report generation begins, these windows are placed on the screen, and the user is prompted to enter data in each field. If the user selects Abandon or Quit from the General menu during input, the input and subsequent report generation are terminated.



Starting date:

Ending date:

Enter the starting date for your report

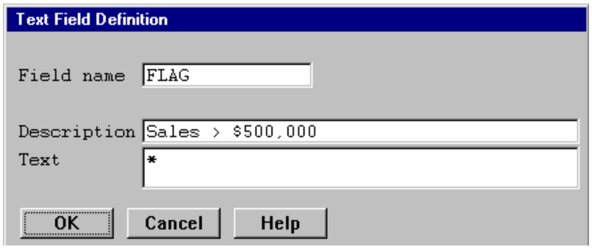
Generate Report File SALES (145 wide)

Figure 8-6. Questioning the user at report-generation time.

Creating a Text Field

A text field defines a text string to print.

1. From the Available Field Types list, select Text Fields.



Text Field Definition

Field name FLAG

Description Sales > $500,000

Text *

Figure 8-7. Defining a text field.

2. In the Text Field Definition window, enter data in each field as instructed below.

Field name. Enter a name to identify your temporary text field within the temporary file. This name must be unique among all temporary fields.

On page 7-13, we defined a conditional for a text field that should be printed only if the Sales year-to-date field contains a value of 500,000 or more. We called our field **FLAG**.

Description. Enter an optional description for this field. This description is the default field header when you select this field to be printed. The description is also used when the field is listed in the Print Fields list or the list of available fields. The default description is the field name.

Our description is **Sales > 500,000**.

Text. Enter the text string that should be printed when this text field is selected as a field to print. The maximum size of the text string is 80 characters.

The text in this field is an asterisk (*), as shown in figure 8-7.

You can cancel creation of your text field by pressing the Abandon shortcut.

3. To save your new text field definition, exit the window.

Creating an Environment Field

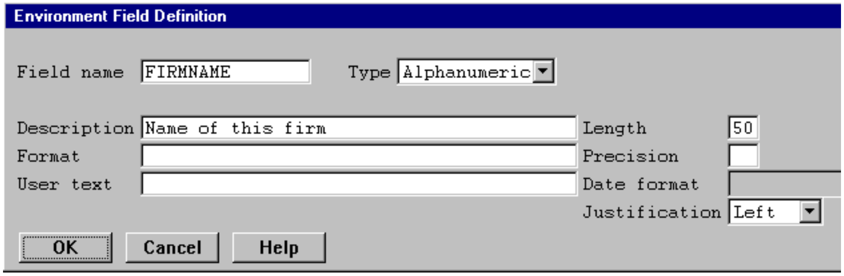
Environment fields are used to obtain data from the system environment. Like question fields, the contents of environment fields remain static throughout report generation and are discarded when the report is completed. Also like question fields, environment fields can enable a single report definition to generate similar yet differing reports. While question fields require the user to supply the distinguishing parameters, environment fields draw these values from the environment.

Here are some examples of values that might be drawn from the environment:

- Static data contained in a global control record, such as a firm name or firm-level options
- Shell or operating system values, such as the working directory
- Any other data that remains constant for the duration of the report and that can be supplied without user input

ReportWriter loads the contents of environment fields in one of two ways. One method is to load it with the contents of a shell or operating system value. The other method is to load it with the contents of a Synergy DBL SEND/RECV message. (See "Defining the Contents of Your Environment Fields" on page 15-30 for information on customizing ReportWriter's environment field processing routine for your particular environment.)

1. Select Environment Fields from the Available Field Types list.



Environment Field Definition

Field name FIRMNAME

Type Alphanumeric

Description Name of this firm

Length 50

Format

Precision

User text

Date format

Justification Left

Figure 8-8. Defining an environment field.

2. In the Environment Field Definition window, enter data in each field as instructed below.

As an example, we've defined an environment field that gets the name of the firm at which you're running the report. (We've assumed that this information is stored in a firm master file record on the system.)

Field name. Enter a name to identify your temporary environment field in the temporary file and in the user-defined environment subroutine. This name must be unique among all temporary fields.

We called our environment field **FIRMNAME**.

Type. Specify the type of your environment field:

Alphanumeric
Numeric
Date
Time

Our example field is **Alphanumeric**.

Description. Enter a description for the environment field. The default is the field name. This description is used as the default field header when this field is printed, and when it is listed in the Print Fields list, Sort Fields list, or the list of available fields.

We thought a good description was **Name of this firm**.

Format. Enter the display format that should be used for this environment field.

To display a list of existing formats, select List selections from the Field functions menu. If the field is not a date or time field, a list of the global formats from the repository is displayed. If the field is a date or time format, a list of predefined display formats is displayed. Select a format from the list. Do not enter your own format; ReportWriter will treat the field as a normal decimal field. (See "Appendix B: Date and Time Formats" for a list of predefined redisplay formats.)

Make sure you read "Editing a temporary field" on page 8-3 for important information about modifying a temporary field's format after it has been selected as a field to print.

Since the **FIRMNAME** *field is the name of a company, we didn't need to enter a format.*

User text. Enter a 40-character user-defined text string associated with this field. (For information on how this text string can be used, see "Defining the Contents of Your Environment Fields" on page 15-30.)

Our example didn't require a user text string.

Length. Enter the length of the environment field.

When we looked at our firm master file record, we saw that firm names are stored in an **a50** *field. Therefore, we entered an environment field length of* **50**.

Precision. If this is a **Numeric** field, enter the number of characters to the right of the decimal point. The maximum number is 10. The default is 0.

Date format. If this field is a **Date** field, a selection window displays the date storage formats for this environment field. Select how you want the specified data to be stored by ReportWriter during report generation. If this field is not a date field, a 0 is entered by default.

Justification. Select how you want the display of this data to be justified: Left, Right, or Center.

Since this is a name, we selected **Left** *to left-justify the data.*

You can cancel creation of your environment field by pressing the Abandon shortcut.

3. To save your new environment field definition, exit the window.

Creating a Subtotal Access Field

A report can be sorted on up to 10 fields. Each of those 10 fields can be “break” fields. Break fields specify the levels at which subtotals will be generated for all fields that are totaled in the report.

Subtotal access fields give you access to a field’s total at any of the break levels. For example, suppose you have a report that breaks on the state and then on the city and calculates a total for the number of traffic accidents occurring within each city on a given day. The report might look something like this:

```
CA     EUREKA       10/01/91     10
CA     EUREKA       10/02/91      3
CA     EUREKA       10/03/91      2
                                 --
EUREKA total (count  3):         15

CA     SACRAMENTO     10/01/91   18
CA     SACRAMENTO     10/02/91    8
                                 --
SACRAMENTO total (count  2):     26

CA total (count  5):             41

FL     SARASOTA      10/01/91     3
FL     SARASOTA      10/02/91     9
FL     SARASOTA      10/03/91     1
FL     SARASOTA      10/04/91     2
                                 --
SARASOTA total (count  4):       15

FL total (count  4):             15

NV     RENO         10/01/91      2
NV     RENO         10/02/91      5
                                 --
RENO total (count  2):            7

NV     LAKE TAHOE   10/01/91      3
NV     LAKE TAHOE   10/02/91      6
NV     LAKE TAHOE   10/03/91      1
                                 --
LAKE TAHOE total (count  3):     10

NV total (count  5):             17

Report total (count  14):        73
```

With subtotal access fields, you can access a variety of information. The above report has two break levels: state and city. The subtotal for the number of accidents is reset at the lowest break level, city. You can define a subtotal access field that contains the current accident count at any break level or at the report level. For example, you could create a subtotal access field that contains the current running total of accidents for all cities, in all states, for the entire report. If you then printed this total on the report, the report would look something like this:

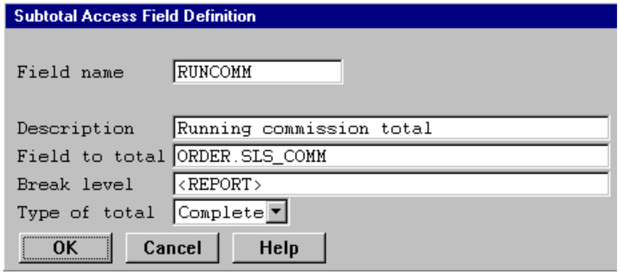
```
State  City         Date         Acc ct      Accum ct
CA     EUREKA       10/01/91     10          10
CA     EUREKA       10/02/91      3          13
CA     EUREKA       10/03/91      2          15
                                 --
EUREKA total (count  3):         15

CA     SACRAMENTO   10/01/91     18          33
CA     SACRAMENTO   10/02/91      8          41
                                 --
SACRAMENTO total(count  2):      26

CA total (count  5):             41

FL     SARASOTA      10/01/91     3          44
FL     SARASOTA      10/02/91     9          53
FL     SARASOTA      10/03/91     1          54
FL     SARASOTA      10/04/91     2          56
                                 --
SARASOTA total (count  4):       15

FL total (count  4):             15

NV     RENO         10/01/91      2          58
NV     RENO         10/02/91      5          63
                                 --
RENO total (count  2):            7

NV     LAKE TAHOE   10/01/91      3          66
NV     LAKE TAHOE   10/02/91      6          72
NV     LAKE TAHOE   10/03/91      1          73
                                 --
LAKE TAHOE total (count  3):     10

NV total (count  5):             17

Report total (count  14):        73
```

Or you could use a subtotal access field to store the total of a given field, and then use that value in a calculation. Suppose you want to know what percentage of the total number of accidents for a given city the given day's accident count was, and what percentage it was for the state. Your report might look like this:

```
State  City          Date      Acc ct   % of city  % of state
                                         subtotal  subtotal
CA     EUREKA        10/01/91      10    .67        .24
CA     EUREKA        10/02/91       3    .20        .07
CA     EUREKA        10/03/91       2    .13        .05
                                   --
EUREKA total (count  3):           15

CA     SACRAMENTO    10/01/91      18   .69        .44
CA     SACRAMENTO    10/02/91       8   .31        .20
                                   --
SACRAMENTO total (count  2):       26

CA total (count  5):               41

FL     SARASOTA      10/01/91       3   .20        .20
FL     SARASOTA      10/02/91       9   .60        .60
FL     SARASOTA      10/03/91       1   .07        .07
FL     SARASOTA      10/04/91       2   .13        .13
                                   --
SARASOTA total (count  4):         15

FL total (count  4):               15

NV     RENO          10/01/91        2   .29        .12
NV     RENO          10/02/91        5   .71        .29
                                   --
RENO total (count  2):             7

NV     LAKE TAHOE    10/01/91      3    .30        .18
NV     LAKE TAHOE    10/02/91      6    .60        .35
NV     LAKE TAHOE    10/03/91      1    .10        .06
                                   --
LAKE TAHOE total (count  3):       10

NV total (count  5):               17

Report total (count  14):          73
```

Another use for subtotal access fields is to subtotal one field at the lowest break level and to subtotal another field only at a higher break level. Normally, when you designate a field to be totaled in the Print Fields list, ReportWriter prints a subtotal at each break. To control the break level at which the subtotal should be printed, you would create a subtotal access field, *not* designate it as a field to be totaled, and then select it to be printed on the desired post-break print line. (See "Creating a Pre- or Post-Break Line" on page 10-8 for more information about creating post-break print lines.)

1. From the Available Field Types list, select Subtotal Access Fields.



Subtotal Access Field Definition

Field name: RUNCOMM
Description: Running commission total
Field to total: ORDER.SLS_COMM
Break level: <REPORT>
Type of total: Complete

Figure 8-9. Defining a subtotal access field.

2. In the Subtotal Access Field Definition window, enter data in each field as instructed below.

   As an example, we've defined a subtotal access field that contains the running sales commission total. We can then print it in our report to see how it accumulates for each record. See figure 8-9.

   Field name. Enter a name to identify your subtotal access field within the temporary file. This name must be unique among all temporary fields.

   We called our subtotal access field **RUNCOMM**.

   Description. Enter an optional description for the subtotal access field. The default is the field name. This description is used as the default field header when this field is printed. The description is also used when this field is listed in the Print Fields list, Sort Fields list, or the list of available fields.

   We used the description **Running commission total**.

   Field to total. Enter the name of the field for which you want a total. To display the list of available fields, select List selections from the Field functions menu. You can select any numeric field from any of the available lists. If you need assistance selecting a field, see "Selecting a Field" on page 6-6.

   If you select an arrayed field, ReportWriter always uses the first element of the array. To subscript the array, specifying an element other than the first, use a calculation field to assign the array element to a temporary field.

   To create a subtotal access field that contains the record count for a given break level, enter **<COUNT>** or select Record count from the Field functions menu.

   Because our subtotal access field will contain the current total for the sales commission field, we selected **ORDER.SLS_COMM** *from the list of fields.*

Break level. Enter the name of the break field that corresponds to the break level for which you want a total. To display a list of available break fields, select List selections from the Field functions menu. If you need assistance selecting a field, see “Selecting a Field” on page 6-6.

If you want the total for the entire report, enter **<REPORT>** or select Report total from the Field functions menu.

Because we want to know the running total for the entire report, we accepted the default, **<REPORT>**.

Type of total. Select Complete if you want the complete total for the specified break level. (ReportWriter preprocesses the records to find out the total for a given level.) Select Running to access the current cumulative, or running, total for the given field at the specified level.

We selected **Running** *to print the current cumulative total for each record.*

You can cancel creation of your subtotal access field by pressing the Abandon shortcut.

3. To save your new subtotal access field definition, exit the window.

9

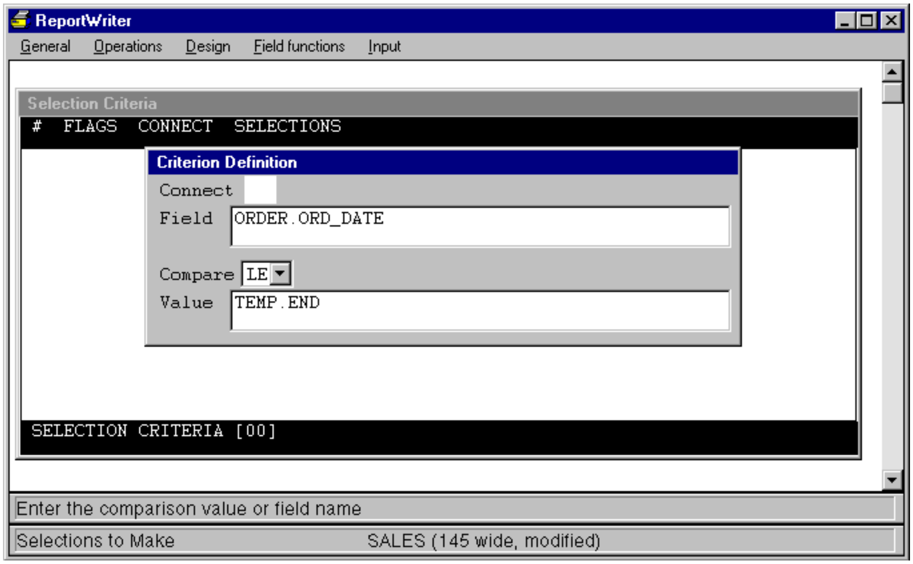
Specifying Selection Criteria

Defining a Selection Criterion

The “Selections to make” function enables you to choose which fields ReportWriter uses and compares in selecting the records for your report. You can define up to 25 selection criteria per report.

1. From the Design menu, select Selections to make to display the Selection Criteria list. As you define selections, they are added to this list at the current cursor position.

   - If you have not yet defined any selections, this list is empty and the Criterion Definition window is also displayed.
   - If a selection criterion has already been defined, select Add selection from the Selection functions menu to display the Criterion Definition window.



```
#  FLAGS  CONNECT  SELECTIONS
```

Connect
Field ORDER.ORD_DATE
Compare LE
Value TEMP.END

```
SELECTION CRITERIA [00]
```

Enter the comparison value or field name

Selections to Make SALES (145 wide, modified)

Figure 9-1. Specifying selection field criteria.

2. Enter data in each field as instructed below.

   We’ll continue the example from page 8-15, in which we defined two question fields (START and END) to be used as selection criteria for our report.

   We need to compare the Order initiated date field to the values the user enters in the START (Start date) and END (End date) fields to determine which records will be included in our report. We want to include all records for which the order date is greater than or equal to the user-defined start date and less than or equal to the user-defined end date.

Connect. If this is the first selection criterion you define, skip this field.

If this is the second (or subsequent) selection criterion, enter the connection operator (AND or OR) to indicate how this criterion is connected to the first criterion. AND designates that the two criteria must *both* be true for a record to match. OR designates that a record will be considered to match if one *or* the other criterion is true.

Comparisons occur from the top of the list to the bottom. For example,

a AND b OR c

is *not* the same as

b OR c AND a

To define parenthetical type selection criteria like (a AND b) OR (c AND d), see "Evaluation order for parenthetical selection statements" on page 9-6.

This field is blank for our first selection criterion. Once we've defined the first criterion (that is, that the order date is greater than or equal to the value the user enters in the START field), we'll select **AND** *as the connection operator, because we want both criteria to be true (not one or the other).*

Field. Enter the name of the field you want to use as a selection field. Note that the fields you specify as selection criteria don't have to be fields to print.

To display the list of available fields that can be used in the conditional, select List selections from the Field functions menu. You can select fields from the list of available fields. See chapter 6 for more information about selecting fields.

If you select an arrayed field, the Field Subscript window is displayed. See "Selecting an arrayed field" on page 6-6 for details on entering subscript information for an arrayed field.

To use only part of a field in a selection, you can specify a range. See "Modifying ranging" on page 9-9.

For our example, we selected **ORD_DATE** *from the list of available fields. (See figure 9-1.)*

Compare. Select one of the comparison operators:

EQ	Equal to
NE	Not equal to
LE	Less than or equal to
LT	Less than
GE	Greater than or equal to
GT	Greater than

For the start date we selected **GE***.*

Value. Enter either a specific value or the name of a field to which data in your records will be compared. For a detailed explanation of the rules that apply to comparisons and the type of data you can enter in this field, see "Completing the Value field" on page 9-4.

We selected **START** *from the list of available fields for this criterion.*

To cancel creation of this selection criterion, press the Abandon shortcut.

3. Exit the window to enter your selection criterion in the Selection Criteria list.
4. To define any additional selection criteria, select Add selection from the Selection functions menu.

Completing the Value field

The following rules apply to comparisons.

- **If the field is numeric**

  The value can be one of the following:

  - A number
  - The name of another numeric field. If the selection field's data type is decimal, the comparison field must also be decimal. If the selection field's data type is implied-decimal, the comparison field must also be implied-decimal.

  To display the list of available fields, select List selections from the Field functions menu. See chapter 6 for more information about selecting fields.

  Decimal values are displayed in selection criteria in parentheses—for example, (1) or (FLD:F1).

- **If the field is alpha, user, or enumerated**

  The value can be one of the following:

  - A string of letters and/or numbers. The comparison is case insensitive. (For example, if you type **No**, ReportWriter will select **no**, **NO**, **No**, and **nO**.)
  - A string of letters and/or numbers in quotation marks. The comparison is case sensitive.
  - The name of an alphanumeric field. The comparison is case insensitive. (An exception to this rule occurs if the alphanumeric field is a question field and the user has entered a value in quotation marks.) To display the list of available fields, select List selections from the Field functions menu. See chapter 6 for more information about selecting fields.
  - A string of letters and/or numbers that contains one or more wildcard characters (*). An asterisk represents any number of characters and can be used anywhere in the string. For example, ReportWriter interprets these strings as follows:

abc*	Any character string that starts with **abc**
***abc**	Any character string that ends with **abc**
abc	Any character string that contains **abc**
a*b*	Any character string that starts with **a** and contains **b**

    When used in a string that's enclosed in quotation marks, an asterisk is treated as a regular character, not a wildcard character. (For example, if you enter "*abc", ReportWriter searches for that exact string: *abc.)

One form of wildcard matching is supported for case sensitive strings. For example, **“abc”*** represents any character string that starts with **abc**, and the comparison is case sensitive.

- A string of letters and/or numbers that contains one or more question marks (?). A question mark represents any single character and can be used anywhere in the string. For example, **ab?** represents any three-character string whose first two characters are **a** and **b**.

  Asterisks and question marks can be used together. For example, **?ab*** represents any string whose second and third characters are **a** and **b**.

  When used in a string that’s enclosed in quotation marks, a question mark is treated as a regular character. (For example, if you enter “?abc”, ReportWriter searches for that exact string: ?abc.)

Alpha values are displayed in selection criteria in curly braces (for example, {ABC} or {FLD:F2}).

Additionally, if the field is user-defined and you enter a string value (rather than a field), this value is passed to the RPS_DATA_METHOD subroutine. RPS_DATA_METHOD is a user-replaceable subroutine in which you can reformat the input data. ReportWriter also stores the data in that format for future selection comparisons. (For more information about RPS_DATA_METHOD see page 15-39.)

If the value is a date

You can enter it in a variety of ways. For example, if today is October 5, 1993, the following date formats are all valid:

10-05-93
10-5-93
10/05/93
10/5/93
100593
Oct 5 1993
Oct 5 93

The default date is today’s date. If you omit any portion of the date, ReportWriter fills it in for you. For example, if you omit the year, ReportWriter fills in the current year; if you omit the month, ReportWriter fills in the current month; and so on. Therefore, formats like the following are also valid for the date October 5, 1993:

RETURN (no entry)
5
//1993
10/5
/5/93
Oct
Oct 5

The RPTDATE environment variable can be set to change the default order used when entering dates. If the environment variable RPTDATE is not set or is set to 0, the order is month, day, year. If RPTDATE is set to 1, the order is day, month, year. If RPTDATE is set to 2, the order is year, month, day. See RPTDATE on page F-8 for more information.

Asterisks (*) and question marks (?) are not allowed in date fields.

Date values are displayed in selection criteria in parentheses, for example, (1/1/93) or (FLD:F3).

▸ If the value is a time…

You must enter the actual storage format. Time values are displayed in selection criteria in parentheses, for example, (123000).

▸ If you press ENTER without typing any data…

The following default values are entered in the specified types of fields:

Field type	Value
Alpha Enumerated User	“ ” (This tells ReportWriter to select only records in which the field is empty.)
Decimal Implied-decimal Time	0
Date	Today’s date

Evaluation order for parenthetical selection statements

When specifying selection criteria, you can define parenthetical type selection criteria such as (a AND b) OR (c AND d). As with other selection statements, ReportWriter performs comparisons from the top of the list to the bottom.

For example:

```
(CMCLNT.CCOPEN).GT.(6/01/93)      (Account opened after June 1)
AND (CMCLNT.CCHOUR).EQ.(1)        (1-hour time difference)
OR (CMCLNT.CCHOUR).EQ.(2)         (2-hour time difference)
OR (CMCLNT.CCHOUR).EQ.(3)         (3-hour time difference)
```

is not the same as

```
(CMCLNT.CCHOUR).EQ.(1)
OR (CMCLNT.CCHOUR).EQ.(2)
OR (CMCLNT.CCHOUR).EQ.(3)
AND (CMCLNT.CCOPEN).GT.(6/01/93)
```

If you use the first example as a selection statement, a record is selected if *any* of the following are true:

- The account was opened after June 1, 1993, *and* the time difference is one hour.
- The time difference is two hours.
- The time difference is three hours.

If you use the second example, a record is selected only when the time difference is one, two, or three hours *and* the account was opened after June 1, 1993.

The same precedence rules apply to conditional statements that are assigned to print fields, calculation fields, and break lines.

Defining a selection criterion by copying an existing one

You can use the copy function to add a selection that's similar to an existing one.

1. Highlight the selection you want to copy.
2. From the Selection functions menu, select Copy selection. The Criterion Definition window is displayed with a copy of the current selection criterion.
3. Edit any value, including the connect or comparison operators.
4. Exit the window when you have finished making changes. The new selection criterion is added to the Selection Criteria list.

Modifying a Selection Criterion

Editing a selection criterion

1. Highlight the selection you want to edit in the Selection Criteria list.
2. Press ENTER.

   The Criterion Definition window displays the current selection criterion.
3. Edit any value.

   To display the list of connection or comparison operators, type any letter at the Connect or Compare prompt, respectively.
4. Exit the window when all information is correct.

Reordering selection criteria

1. Highlight the selection you want to move.
2. From the Selection functions menu, select Reorder selections.

   The selection is enclosed in square brackets ([]). (To see the close bracket, press the Scroll right shortcut.)
3. Use the UP ARROW and DOWN ARROW keys to move the selection to the desired location.
4. When the selection is where you want it, select Reorder selections again.

   If you move the selection criterion that you created first, or if it is no longer the primary criterion, ReportWriter assigns it a default AND connector.

Modifying subscripting

To change the subscript of an arrayed field in a selection criterion,

1. Place the cursor in the Field or Value field of the Criterion Definition window.
2. From the Field functions menu, select Change subscripting.

   The Field Subscript window is displayed.
3. See "Selecting an arrayed field" on page 6-6 for details on entering subscript information for an arrayed field.

Modifying ranging

Sometimes you will want to use only part of a field for comparison. You can do this by specifying the range of characters to be compared in a selection criterion.

1. Place the cursor in the Field or Value field of the Criterion Definition window.
2. From the Field functions menu, select Change ranging.

   The Field Range window is displayed. (If your field is larger than 99 characters, this window was displayed automatically when you selected the field.)
3. See "Modifying ranging" on page 7-11 for details on entering ranging information for a field.

Setting a selection to be "true if blank"

ReportWriter uses selection criteria to determine which records to print in a report. If a record meets all of the selection criteria, it is printed. On occasion, you may want to select a record if a given field matches a certain value, or if the field is blank or contains a value of zero. Instead of setting up two selection criteria, you can set up the first criterion and then set the "true if blank" flag. This flag makes a selection criterion evaluate to true if either the selection statement is met *or* the first operand is blank or zero. (It will also work on the second operand if the second operand is a question field.) This feature is very useful if your report puts up an input window with multiple question fields and you want the user to be able to enter data in any or all of the fields.

For example, let's suppose we have a question field that is used to prompt the user to enter a customer name at report generation time. If the user enters a name, we want to compare it to a field in the file and only include records for that customer. If the user enters nothing, we want to include records for all customers. We can create a question field and then use it in a selection statement, as follows:

```
(TEMP.CUS_NAM.EQ.FILE01.CUS_NAM)
```

If we then set the "true if blank" flag, ReportWriter will select all records that match the specified name, or, if no value is entered, ReportWriter will include all records in the file.

1. Highlight the appropriate selection criterion in the Selection Criteria list.
2. From the Selection functions menu, select Set "true if blank".

   This entry is a toggle, so if you change your mind, you can use it to turn the flag off again.

   When a selection field has been flagged as "true if blank," a letter S is displayed in the FLAGS column for that field in the Selection Criteria list.

Deleting a selection criterion

1. Highlight the criterion in the Selection Criteria list.
2. From the Selection functions menu, select Delete selection.

10

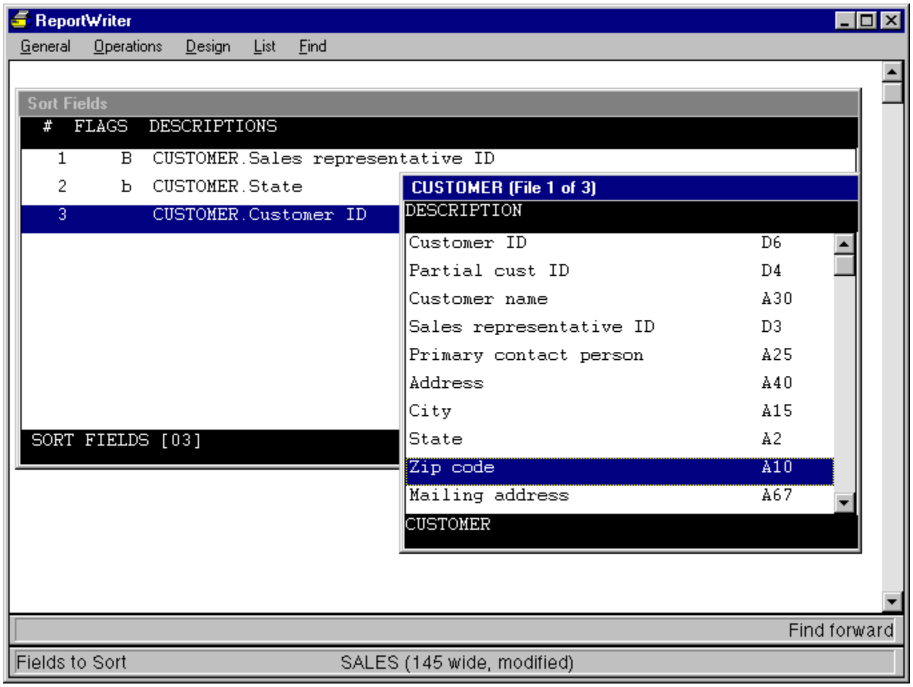
Sorting Your Report

Selecting a Sort Field

The Fields to sort menu entry enables you to determine the organization of your report by specifying on which fields the report will be sorted. The sort fields do not have to be the same as the fields to print or the selection fields. They can be from any file, including the temporary file.

1. From the Design menu, select Fields to sort.

   The Sort Fields list is displayed. If you're creating a new report, or if you haven't defined any sort fields yet, this window is empty and the list of available fields is also displayed. As you select sort fields from the list of available fields, they are added to the Sort Fields list.



```
 #  FLAGS  DESCRIPTIONS
 1    B    CUSTOMER.Sales representative ID
 2    b    CUSTOMER.State
 3         CUSTOMER.Customer ID
SORT FIELDS [03]
```

```
CUSTOMER (File 1 of 3)
DESCRIPTION
Customer ID                     D6
Partial cust ID                 D4
Customer name                   A30
Sales representative ID         D3
Primary contact person          A25
Address                         A40
City                            A15
State                           A2
Zip code                        A10
Mailing address                 A67
CUSTOMER
```

```
                                              Find forward
Fields to Sort          SALES (145 wide, modified)
```

Figure 10-1. Selecting sort fields.

When you sort, ReportWriter creates temporary files with the names **RG***job#***.TMP** and **RB***job#***.TMP**, where *job#* is the job number of the ReportWriter process. ReportWriter deletes these files automatically unless the program terminates abnormally, in which case you may see these files in your directory.

2. Highlight the field on which you want to sort and press ENTER. See chapter 6 for more information about selecting fields.

3. Select any other fields on which you want your report to be sorted.

You can select up to 10 fields on which to sort. If the field you select is an arrayed field, only literals are allowed as subscripts. If the field you select is larger than the maximum field size (99), you are prompted to specify a range of characters.

4. When you're finished selecting sort fields from the list of available fields, press the Exit shortcut.

   Once you exit the list of available fields, you have to use the Add sort field or Add sort fields function to select any other fields to sort.

Adding a single field

To add *one* additional field to the Sort Fields list (if the list of available fields is not already displayed),

1. From the Sort functions menu, select Add sort field.

   The list of available fields is displayed.

2. Select the field on which you want to sort.

   That field is added immediately after the highlighted field in the Sort Fields list. The list of available fields is removed from the screen.

Adding multiple fields

To add more than one field to the list of fields to sort (if the list of available fields is not already displayed),

1. From the Sort functions menu, select Add sort fields.

   The list of available fields is displayed.

2. Select a field on which you want to sort.

   That field is added immediately after the highlighted field in the Sort Fields list. The list of available fields remains on the screen.

3. Select any additional fields to sort.

4. When you've selected all the fields on which you want to sort, press the Exit shortcut to remove the list of available fields.

Modifying a Sort Field

Reordering sort fields

1. Highlight the field you want to move in the Sort Fields list.
2. From the Sort functions menu, select Reorder sort fields.

   The sort field is enclosed in square brackets ([ ]). (To see the close bracket, press the Scroll right shortcut.)
3. Use the UP ARROW and DOWN ARROW keys to move the field to the desired location.
4. When the sort field is where you want it, select Reorder sort fields again.

Modifying subscripting of an arrayed sort field

1. Highlight the field in the Sort Fields list.
2. From the Sort functions menu, select Change subscripting.

   The Field Subscript window is displayed.
3. See "Selecting an arrayed field" on page 6-6 for details on entering subscript information for an arrayed field.

When you subscript in the Fields to sort function, you can only use a literal as your subscript value.

Modifying ranging

Sometimes you will want to use only part of a field for sorting. You can do this by specifying a range of characters to be used in a sort field.

1. Highlight the field in the Sort Fields list.
2. From the Sort functions menu, select Change ranging.

   The Field Range window is displayed. (If your field is larger than 99 characters, this window was displayed automatically when you selected the field.)
3. See "Modifying ranging" on page 7-11 for details on entering ranging information for a field.

When you're in the Fields to sort function, you can only use literals as your range values.

Setting a report break

Any or all of the 10 sort fields can specify that a change in the sort field value is to generate a break in the report. Follow these steps to establish an initial break status for a sort field.

1. Highlight the field in the Sort Fields list.
2. From the Sort functions menu, select Change break status.

   An uppercase letter **B** is displayed in the FLAGS column of the Sort Fields list to indicate that the report is to break and generate a new page when that field changes. If you select Change break status again, a lowercase letter **b** is displayed to indicate that the report is to break *without* a page break when that field changes. Each time you select this function, the break status toggles between **B** and **b**. To clear the break status for a field, select Clear break status from the Sort functions menu.

   At each report break, ReportWriter computes a subtotal for any field that is totaled at the end of the report. (See "Designating a field to be totaled" on page 7-12.)

   For example, we decided to sort our report by sales representative. By setting a **B** *break on the Sales representative field, a new page of the report will start when the name of the salesperson changes. If Sales year-to-date is to be totaled at the end of the report, ReportWriter will compute a subtotal of sales year-to-date for each salesperson.*

   At each report break, ReportWriter generates a break summary line. When the report does not contain fields being totaled, the break summary line contains the following:

   - The break description
   - The break count (the number of records printed in the break set)

   When one or more fields are being totaled, the break summary line (sometimes referred to as the subtotal line) contains the following:

   - The break description
   - The word "total"
   - The break count in parentheses
   - The subtotal amounts

   For break summary lines, the break description is the value of the field on which the break is occurring. For example, if you are breaking on a field that contains the day of the week, the break description for the first break set would be "Monday," the next would be "Tuesday," and so forth. For information about suppressing record counts and/or break total descriptions, see "Specifying Miscellaneous Design Options" on page 7-25.

   Breaks are hierarchical.

   *For instance, we've set a break on both the Sales representative field and the State field, and ReportWriter will generate subtotals on each sales rep, as well as subtotals for each state for each sales rep (in addition to a grand total for the entire report). See figure 10-1. The Sales representative break (***B***) will also generate a new page, while the State break (***b***) will not.*

Clearing a sort field's break status

1. Highlight the field in the Sort Fields list.
2. From the Sort functions menu, select Clear break status.

Clearing a report break also deletes any associated pre- and post-break lines and their conditionals. For more information, see "Creating a Pre- or Post-Break Line" on page 10-8.

Setting sort order

By default, a field is sorted in forward order, which means alphanumeric data is sorted in alphabetical order and decimal data is sorted from lowest to highest. You can reverse this order so that alphanumeric data is sorted in reverse alphabetical order, and decimal data is sorted from highest to lowest.

1. Highlight the field in the Sort Fields list.
2. From the Sort functions menu, select Change sorting order.

   When reverse order is set, the letter **R** appears in the FLAGS column for that sort field.

   The Change sorting order function is a toggle; select it again to change the sorting order back to forward.

Merging sort fields

Merging sort fields enables you to treat multiple fields as one for the purposes of sorting. This feature is used in conjunction with multiple projections. (See pages 5-10 and 18-6 for more information about multiple projections.)

The field that you select for merging will be merged with the field that precedes it in the Sort Fields list. The field that you are *merging with* can be a break field, but the field *being merged* cannot. Sort fields that are merged together must be the same data type and length and must be sorted in the same order.

1. Highlight the sort field that you want to merge.
2. From the Sort functions menu, select Merge sort fields.

   The Merge sort fields function is a toggle; select it again to un-merge the fields.

   When a sort field has been merged, a letter M appears in the FLAGS column for that field in the Sort Fields list. You cannot move a merged sort field (or the field with which it is merged), nor can you split merged sort fields by moving another field between them. ReportWriter treats merged fields as a pair and skips the position between the merged fields when you move other fields in the list.

   You must delete merged sort fields in "last in, first out" order (in other words, from the bottom up).

Deleting a sort field

A sort field can be deleted provided both of the following conditions are true:

- It is not merged with the sort field that follows it.
- It is not used in a subtotal access field definition.

1. Highlight that field in the Sort Fields list.
2. From the Sort functions menu, select Delete sort field.

Creating a Pre- or Post-Break Line

A *pre-break line* is a report line that is printed at the beginning of a report break. A *post-break line* is a report line that is printed at the end of a report break, following the break summary line.

Break lines can contain text fields, data fields, environment fields, question fields, calculation fields, and subtotal access fields. Any data field printed on a pre-break line uses the values from the first record of the break set. Any data field printed on a post-break line uses the values from the last record of the break set.

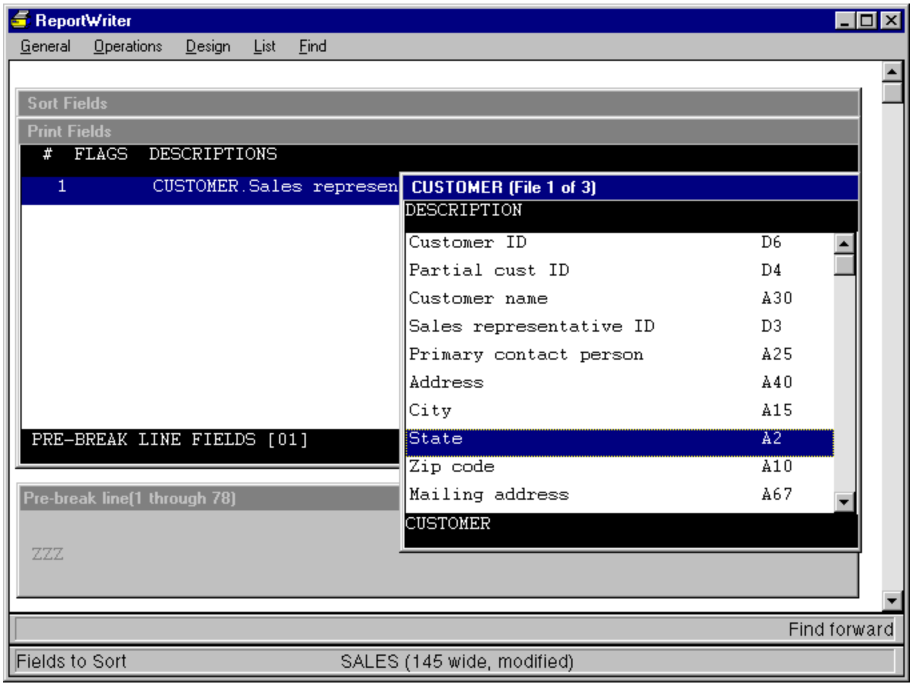
A pre- or post-break line can contain a maximum of 99 fields and have a maximum width of 255 characters. One pre-break and one post-break line can be associated with each break field.

The following instructions refer to the “pre-break line,” but they apply to post-break lines as well.

1. Highlight the break field (indicated by a B or b in the FLAGS column) to which the line will be assigned in the Sort Fields list.
2. From the Sort functions menu, select Pre-break line. Two lists and a window are displayed:
   - In the center of the screen is the Print Fields list. If you’re creating a new break line, this list is empty. As fields are selected from the list of available fields, they’re placed in the Print Fields list. The Print Fields list displays field descriptions by default.
   - On the right side of the screen is the list of available fields. This list contains the fields that are available from the files chosen as files to read.
   - Near the bottom of the screen is a window that shows you the current layout of the pre-break print line. Each break line field that you select is represented by its format, if one exists. Otherwise, each field is represented by its default format:

@	Alpha, user, and enumerated fields
Z	Numeric fields
MM/DD/YY	Date fields
HH:MM	Time fields

If you’re creating a new break line, this window is empty.



```
Sort Fields
Print Fields
 #  FLAGS  DESCRIPTIONS
  1          CUSTOMER.Sales represen

PRE-BREAK LINE FIELDS [01]

Pre-break line(1 through 78)
ZZZ
```

```
CUSTOMER (File 1 of 3)
DESCRIPTION
Customer ID                  D6
Partial cust ID              D4
Customer name                A30
Sales representative ID      D3
Primary contact person       A25
Address                      A40
City                         A15
State                        A2
Zip code                     A10
Mailing address              A67
CUSTOMER
```

```
                                                  Find forward
Fields to Sort              SALES (145 wide, modified)
```

Figure 10-2. Creating a pre-break line.

Selecting a field to print on a break line

1. From the list of available fields, highlight the field you want to include on the break line.
2. Press ENTER. (See chapter 6 for more information about selecting fields.)

   When you select a field, that field appears in the Print Fields list, and it is also displayed in report form in the layout window at the bottom of the screen. A new field is added immediately after the highlighted field in the Print Fields list. The layout window automatically scrolls so that the currently selected field is always visible.

   As an example, we've included the Sales representative ID field in the pre-break line, as shown in figure 10-1. We also removed the Sales representative ID field from the detail lines, so that it prints only once for each sales rep in the report.

   You can include up to 99 fields on the break line. The line can have a maximum width of 255 characters.
3. Press the Exit shortcut when you're finished. Once you exit the list of available fields, if you want to select additional fields to print, you must use the Add field or Add fields function.

Modifying a break line field

Reordering break line fields

1. Highlight the field you want to move in the Print Fields list.
2. From the Field functions menu, select Reorder fields.

   The break line field is enclosed in square brackets ([ ]). (To see the close bracket, press the Scroll right shortcut.)
3. Use the UP ARROW and DOWN ARROW keys to move the field to the desired location.
4. When the field is where you want it, select Reorder fields again.

Modifying subscripting of an arrayed break line field

1. Highlight the field in the Print Fields list.
2. From the Field functions menu, select Change subscripting to display the Field Subscript window.
3. See "Selecting an arrayed field" on page 6-6 for details on entering subscript information for an arrayed field.

Modifying ranging

Sometimes you will want to use only part of a field in your break line. You can do this by specifying the range of characters to be included in the break line field.

1. Highlight the field in the Print Fields list.
2. From the Field functions menu, select Change ranging to display the Field Range window. (If your field is larger than 99 characters, this window displayed automatically when you selected the field.)
3. See "Ranging a selected field" on page 6-8 for details on entering ranging information for a field.

Stripping trailing blanks

1. Highlight the field in the Print Fields list.
2. From the Field functions menu, select Strip trailing blanks. An input window is displayed.
3. At the Strip character prompt, enter the character that you want to follow the blank-stripped field. The default strip character is one blank space.
4. Exit the window.

   The strip character that you enter appears in the layout window after the stripped field, and a letter **S** appears in the FLAGS column for that field in the Print Fields list. If your strip character is nonblank, all trailing blanks in your break line are replaced with one occurrence of the specified strip character and an additional space.

   The Strip trailing blanks entry is a toggle: select it again to turn blank stripping off.

Modifying the field position on a break line

When you select a field to print on a break line, ReportWriter places it two spaces after the previous field by default. You can modify this default positioning and even overlap fields, if you wish.

1. Highlight the field in the Print Fields list.
2. From the Field functions menu, select Arrange field position.

   A ruler is displayed at the top of the layout window.

   If any of your fields to print are being subtotaled at this break level, a row of asterisks (*) is displayed to represent the location of that field. This feature should assist you in designing your break line.
3. Press the LEFT ARROW key to move the field to the left (delete spaces in front of the field). Press the RIGHT ARROW key to move the field to the right (insert a space in front of the field).

   A space that precedes a field is attached to the field, so when you move the field, the space moves as well. Note that when a field is moved, all fields that follow it also move.

   As you move the field, the layout window automatically scrolls to the left or right, if necessary, so that the highlighted field is always visible.
4. To exit break line field positioning mode, press the Exit shortcut.

Assigning a conditional to a break line field

You can define and assign a conditional to a break line field. You can connect up to five conditionals together (with AND and OR).

1. Highlight that field in the Print Fields list.
2. From the Field functions menu, select Specify conditions to display the Condition Criteria list. As you define conditions, they are added to this list.
   - If you haven’t defined any conditions for the break line field, this list is empty and the Criterion Definition window is also displayed.
   - If a condition has already been defined for the break line field, select Add condition from the Condition functions menu to display the Criterion Definition window.
3. See “Assigning a conditional to a field” on page 7-13 for detailed instructions on specifying a conditional.

   To copy, edit, move, or delete a conditional, refer to the following sections:

   “Defining a conditional by copying an existing one” on page 7-15
   “Editing a conditional” on page 7-15
   “Reordering conditionals” on page 7-15
   “Deleting a conditional” on page 7-15
4. To return to the Print Fields list, press the Exit shortcut from the Condition Criteria window.

Deleting a field from the break line

1. Highlight the field you want to delete.
2. From the Field functions menu, select Delete field.

Suppressing a blank line

Pre- and post-break lines are separated from the records in that break set by a blank line.

To suppress the printing of the blank line associated with a pre- or post-break print line, from the Line functions menu, select Suppress blank line.

The text at the bottom of the layout window tells you if the blank line is currently being suppressed.

The Suppress blank line entry is a toggle; select it again to print the blank line.

Assigning a conditional to a pre/post-break line

You can define and assign a conditional to each pre- and post-break line. The break line is only printed on the report if the conditions specified in the conditional are true. You can connect up to five conditionals together (with AND and OR) and assign them to a break line.

1. From the Line functions menu, select Specify line conditions to display the Condition Criteria list. As you define conditions, they are added to the end of this list.
    - If you haven't defined any conditions for the break line, this list is empty and the Criterion Definition window is also displayed.
    - If a condition has already been defined for this break line, select Add condition from the Condition functions menu.
2. See "Assigning a conditional to a field" on page 7-13 for more information on specifying a conditional.

    If you select an arrayed field for use in a conditional, ReportWriter always uses the first element of the array. To subscript the array, specifying an element other than the first one, use a calculation field to assign the array element to a temporary field.
3. To save your conditional and return to the Condition Criteria list, exit the window.

    If you've defined a conditional for the break line, the word "Conditioned" is displayed at the bottom of the layout window.

A sort field's associated conditionals are deleted when you clear the field's break status.

Modifying a pre- or post-break line conditional

To copy, edit, move, or delete conditionals, refer to the following sections:

“Defining a conditional by copying an existing one” on page 7-15
“Editing a conditional” on page 7-15
“Reordering conditionals” on page 7-15
“Deleting a conditional” on page 7-15

Exiting the break line function

When you’ve finished selecting and arranging the fields that you want printed on your break line, press the Exit shortcut. If one or more fields are being printed on the break line, the letter P (for pre-break) or T (for post-break) precedes the letter B (or b), which indicates a break, in the Sort Fields list.

11

Report Maintenance

Saving a Report

Any time a report is loaded, you can save your changes with the Save report function.

Saving a new report

From the Operations menu, select Save report. The report is saved with the name you gave it when you created it.

Saving an existing report

You can save an existing report with the same name or a different name.

1. From the Operations menu, select Save report.
2. You are prompted for the report name, and the current report name is displayed.
3. To save the report under this name, exit the window. To save the report under a different name, enter the new name and exit the window.

   If you enter the name of a report that already exists, an error message is displayed.

   To cancel saving of the report, press the Abandon shortcut.
4. A “Report saved” message is displayed. Press ENTER to continue. Your original report is still loaded.

Clearing a Report

Clearing a report deletes all files to read, fields to print, temporary fields, selection criteria, and sort fields from the report. When you clear a report, it is as if you just created a brand new report with the same name as the old one.

1. From the Operations menu, select Clear loaded report.
2. At the confirmation prompt, select Yes to clear the report or press ENTER to cancel clearing.

⚠ Be careful when using this function! It deletes the entire contents of your report!

Renaming a Report

1. From the Operations menu, select Rename loaded report.
2. At the Report name prompt, enter a new name for the report.
3. Exit the window.

If you type the name of a report that already exists, you are prompted

A report already exists with that name. Use it anyway?

Select Yes to overwrite the old report with the new one or No to cancel.

When you rename a report, the name of the report changes in the lower right corner of the screen. Because ReportWriter doesn't know what modifications you may have made to the report header, it doesn't attempt to change the header to contain the new report name. You must modify the header yourself.

You must still save your changes when you exit, or the report name will not be changed.

Deleting a Report

1. From the Operations menu, select Delete report.
2. At the Report name prompt, enter the name of the report you want to delete.

   To select form a of available reports, select List selections from the Name functions menu. (To exit this list without making a selection, press the Exit shortcut.)
3. Exit the window.
4. At the confirmation prompt, select Yes to delete the report or No to cancel the deletion.

   If the report you're trying to delete is currently in use, you'll get an error message.

12

ReportWriter Utilities

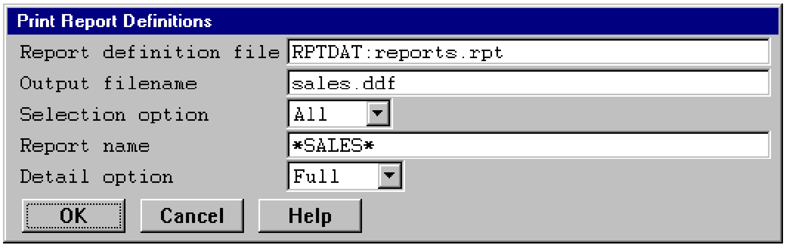
The ReportWriter utility functions enable you to print, load, and unload report definitions, generate and load report schemas, generate file cross-references, and create a new report definition file.

To use the ReportWriter utilities, select Operations > Utilities to display the ReportWriter Utilities menu.

Printing Report Definitions

The Print Report Definitions utility generates a list of report definitions to a file or the screen.

1. From the ReportWriter utilities menu, select Print report definitions.



Print Report Definitions

Report definition file	RPTDAT:reports.rpt
Output filename	sales.ddf
Selection option	All
Report name	*SALES*
Detail option	Full

Figure 12-1. Printing report definitions.

2. In the Print Report Definitions window, enter data in the fields as instructed below.

Report definition file. Enter the name of the report definition file from which you want to print definitions. This field defaults to the name of the report definition file ReportWriter is currently using.

Output filename. Enter the name of the file into which the report definition should be printed. If you don't specify a filename extension, it defaults to **.ddf**. To print output to the screen, leave this field blank.

Selection option. Indicate what you want included in the output file:

All	Print all report definitions in the specified report definition file. You can also use a wildcard character (* or ?) to specify a set of reports (see below).
Single	Print a particular report.

Report name. If you selected **All** in the Selection option field, you can enter a partial report name combined with wildcard characters to specify a set of reports. For example, if you type **ORDER***, all report names that start with "ORDER" will be printed. If you type ***ORDER***, all report names that contain the character sequence "ORDER" will be printed.

If you selected **Single** in the Selection option field, enter the name of the report definition you want to print. To select from a list of available reports, select List selections from the Utility functions menu.

Detail option. Select the detail level for the report:

Summary	Print only the major attributes of the report.
Detailed	Print both summary and detailed sections that describe attributes of files to read, fields to print, selection criteria, and sort keys.

Full	Print file relationships, indexing information, conditional information, temporary field descriptions, and so forth, in addition to all of the attribute information.

3. To generate the report definitions, exit the window.

If the filename you specified in the Output filename field does not exist, the file is created. If the file already exists, you are prompted

File already exists. Do you want to delete it?

- To overwrite the existing file with the new definitions, select Yes.
- To append the new definitions to the end of the existing file, select No.
- To return to the Output filename field and enter another filename, select Cancel.

When processing is complete, you are returned to the ReportWriter utilities menu.

If you didn't specify an output filename, output goes to the screen. Press the Next page shortcut to view the next page, or press the Exit shortcut to exit the viewing function.

Sample output from the Print Report Definitions utility

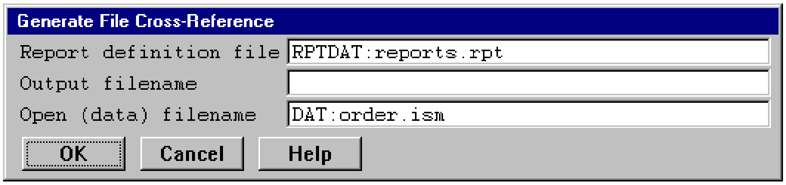
```
Report Name     : SALESRPT

Generation Date : 04-NOV-96, 13:40:40
Definition File : RPSRPT:reports.rpt
Report Version  : 6.1
Report Size     : 65

***************************** Summary Section*******************
   Files to Read: 4
   Total Lines: 3
     Data Lines: 1
   Fields to Print: 3
   Selections to Make: 1
   Fields to Sort: 2
   Fields to Create: 1
     Question Fields: 1
   Header/Footer Layout:   Visible  Separate  BlankLines  Modified
     Report Header         Y        Y          1
     Report Footer         N        Y          1
     Page Header           Y        N          1          N
     Page Footer           N        N          1
     Date and Page #       Y
   Miscellaneous:
     Lines Per Page: 66
     Blank Lines Between Records: 0
     Blank Lines Between Breaks: 0
     Detailed Report? Y
     Detail Record Count? Y
     Break Field Count? N
     Form Feeds? Y
     Dashed Lines? Y
     Total Descriptions? Y
     New Report Summary Description? N

**************************** Detailed Section****************************

Filename Description                Structure Level ParentFile Tandem
External
-------------------------------------------------------------------------
Employee master file.Employee maste EMPLOYEE     0
Customer master file.Customer maste CUSTOMER     1  EMPLOYEE
Sls order management file.Sales ord ORDER        1  EMPLOYEE
Product management file.Product man PRODUCT      2  ORDER

Field Name          Type Size Cond Totl Strp Frmt Hder Indx Line Pos  Just
-------------------------------------------------------------------------
ORDER.SALES_REP     D      3                      H          D 1    2  L
ORDER.ORD_ITEM      D      3                      H          D 1   27  L
PRODUCT.PRDT_PRICE  D      8       T              H          D 1   52  L

Connect  Selection Field  Indexed  Operator  Comparison       Indexed
---------------------------------------------------------------------
         TEMP.REP?                 EQ        ORDER.SALES_REP

Sort Field Name    Indexed   Merged   Reversed   Break
---------------------------------------------------------------------
ORDER.SALES_REP                                  b
ORDER.ORD_ITEM                                   b

************************ Full Description Section ***********************

<Files to Read>

#    FromStructure   FromKeyName            ToStructure   ToKeyName
-----------------------------------------------------------------------
( 1) EMPLOYEE       : SKEY          =======> CUSTOMER     : SALES REP
( 2) EMPLOYEE       : EMPLOYEE ID   =======> ORDER        : SALES REP
( 3) ORDER          : PRODUCT ID    =======> PRODUCT      : PRODUCT ID

FileName      FileType       OpenFilename
-----------------------------------------------------------------------
EMPLOYEE       DBL ISAM       DAT:employee.ism
CUSTOMER       DBL ISAM       DAT:customer.ism
ORDER          DBL ISAM       DAT:order.ism
PRODUCT        DBL ISAM       DAT:product.ism
```

```
<Create Fields>

Question Fields
    REP?           :- Type:D  Size:3  Just:R  Prec:0
               Upcase:N  BlankZero:N  Negative:N  Required:N
        Default    :
        Info Line  :
        Prompt     : Enter rep name?
        Description: REP?
        Format     :

<Fields to Print>

Field Name     Heading                   Line #         Position
-----------------------------------------------------------------
SALES_REP    : Sales representative ID   (D - 1 of 1)     2
ORD_ITEM     : Order item (product ID)   (D - 1 of 1)     27
PRDT_PRICE   : Product price             (D - 1 of 1)     52

 <Fields to Sort>

1) Sort Key Field   - ORDER:SALES_REP
2) Sort Key Field   - ORDER:ORD_ITEM

************************ End of Report (SALESRPT) ***********************
```

Generating a File Cross-Reference

The Generate File Cross-Reference utility generates a list of all reports that reference a given data file. Such a list could be useful when you plan modifications to a file and need to know how they will affect your existing reports.

1. From the ReportWriter utilities menu, select Generate file cross-reference.



Report definition file RPTDAT:reports.rpt
Output filename
Open (data) filename DAT:order.ism

Figure 12-2. Generating a file cross-reference.

2. In the Generate File Cross-Reference window, enter data in the fields as instructed below.

Report definition file. Enter the name of the report definition file to check for references to the specified open (data) filename. This field defaults to the name of the report definition file ReportWriter is currently using.

Output filename. Enter the name of the file into which the cross-reference information for the given data filename should be printed. If you don't specify a filename extension, it defaults to **.ddf**. To print output to the screen, leave this field blank.

Open (data) filename. Enter the name of the data file for which cross-reference information is to be printed. This file specification can include a full path or logical name. You can also specify a wildcard filename such as **DAT:*** or ***.ap**. All reports that reference the specified open filename will be printed, as will the file definition names and the file's sequence number (as a "file to read"). For example, if the CUSTOMER file **DAT:customer.ism** is selected as the third file to read, "(03)" will be printed next to the file definition name. If you don't specify a filename, cross-reference information will be printed for all reports and the files they reference.

In our example in figure 12-2, cross-references to the **DAT:order.ism** *file in* **RPTDAT:reports.rpt** *are printed to the screen.*

3. To generate the cross-reference, exit the window.

If the filename that you specified in the Output filename field does not exist, the file is created. If the file already exists, you are prompted

File already exists. Do you want to delete it?

- To overwrite the existing file with the new cross-reference, select Yes.
- To append the new cross-reference to the end of the existing file, select No.
- To return to the Output filename field and enter another filename, select Cancel.

During processing, the report names being examined are displayed in the lower right corner of the screen. When processing is complete, you are returned to the ReportWriter utilities menu.

If you didn’t specify an output filename, output goes to the screen. Press the Next page shortcut to view the next page, or press the Exit shortcut to exit the viewing function.

Sample output from the Generate File Cross-Reference utility

```
Generation Date : 22-NOV-96, 13:35:13
Definition File : RPTDAT:reports.rpt

Report Name: File Definition - Open Filename
-----------------------------------------------------------
CLIENT_HIST92: CUST92 - DAT:customer.ism (01)
CLIENT_HIST93: CUST93 - DAT:customer.ism (02)
CLIENT_MSTR: CMCLNT - DAT:customer.ism (01)
CUST92: CUSTOMER - DAT:customer.ism
(02)<$startrange>cross-reference:generating;Generate File Cross-Reference
utility;file:cross-reference:generating;generating:file
cross-reference;utilities:Generate File Cross-Reference
```

Unloading Report Definitions

The Unload Report Definitions utility unloads one or more report definitions from the specified report definition file to a sequential file.

1. From the ReportWriter utilities menu, select Unload report definitions.
2. In the Unload Report Definitions window, enter data in the fields as instructed below.

   Report definition file. Enter the name of the report definition file from which you want to unload report definitions. This field defaults to the report definition file ReportWriter is currently using.

   Output filename. Enter the name of the sequential file into which the report definitions will be unloaded. If you don't specify a filename extension, it defaults to **.ddf**.

   Selection option. Select which reports you want to be unloaded to the specified output file:

   All Specify all report definitions or a set of reports using a wildcard character (* or ?).

   Single Specify a particular report.

   Report name. If you chose **All** in the Selection option field, you can enter a partial report name with a wildcard character to specify a set of reports. For example, if you type **ORDER***, all report names that start with "ORDER" will be selected. If you type ***ORDER***, all report names that contain the character sequence "ORDER" will be selected.

   If you chose **Single** in the Selection option field, enter the name of the report definition you want to select. To select from a list of reports in the report definition file, select List selections from the Utility functions menu.
3. To unload the report definitions, exit the window.

   If the filename that you specified in the Output filename field doesn't exist, the file is created. If the file already exists, you are prompted

   File already exists. Do you want to delete it?

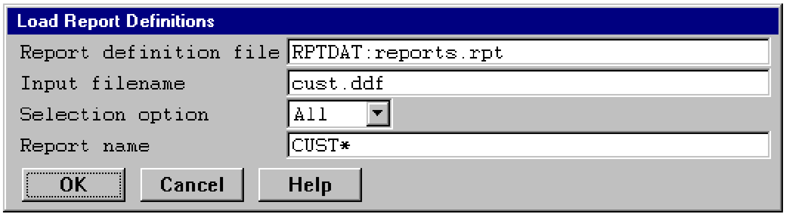
   - To overwrite the existing file with the new definitions, select Yes.
   - To append the new definitions to the end of the existing file, select No.
   - To return to the Output filename field and enter another filename, select Cancel.

   When processing is complete, you are returned to the ReportWriter utilities menu.

Loading Report Definitions

The Load Report Definitions utility loads report definitions from a sequential file to the specified report definition file. You must use this utility (rather than the Synergy DBL **isload** utility) when loading a sequential report definition file back into a Synergy™ ISAM file.

1. From the ReportWriter utilities menu, select Load report definitions.



Load Report Definitions

Report definition file: RPTDAT:reports.rpt
Input filename: cust.ddf
Selection option: All
Report name: CUST*

Figure 12-3. Loading report definitions.

2. In the Load Report Definitions window, enter data in the fields as instructed below.

   Report definition file. Enter the name of the report definition file into which you want to load report definitions. This field defaults to the name of the report definition file ReportWriter is currently using.

   Input filename. Enter the name of the unloaded file from which to load report definitions. If you don't specify a filename extension, it defaults to **.ddf**.

   In the example in figure 12-3, the file from which to load report definitions is called **cust.ddf**.

   Selection option. Indicate which reports you want loaded:

All	Specify all report definitions or a set of reports using a wildcard character (* or ?).
Single	Specify a particular report.

   Report name. If you chose **All** in the Selection option field, you can enter a partial report name with a wildcard character to specify a set of reports. For example, if you type **ORDER***, all report names that start with "ORDER" will be selected. If you type ***ORDER***, all report names that contain the character sequence "ORDER" will be selected.

   If you chose **Single** in the Selection option field, enter the name of the report definition you want to select. To select from a list of reports that are already in the report definition file, select List selections from the Utility functions menu.

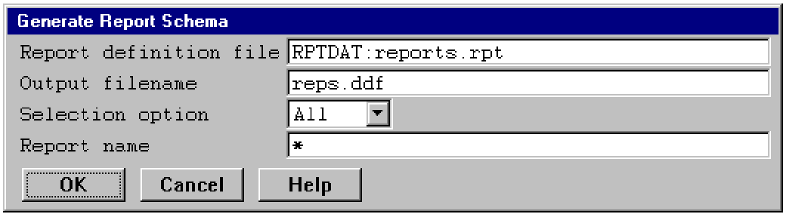
   In our example, all report names that start with "CUST" will be selected.

3. To load the report definitions, exit the window. When processing is complete, you are returned to the ReportWriter utilities menu.

Generating a Report Schema

The Generate Report Schema utility generates a Report Definition Language description of your report definition file to a file. See chapter 17 for more information on Report Definition Language.

1. From the ReportWriter utilities menu, select Generate report schema.



Generate Report Schema

Report definition file RPTDAT:reports.rpt

Output filename reps.ddf

Selection option All

Report name *

Figure 12-4. Generating a report schema.

2. In the Generate Report Schema window, enter data in the fields as instructed below.

   Report definition file. Enter the name of the report definition file whose schema (Report Definition Language description) you want to generate. This field defaults to the name of the report definition file ReportWriter is currently using.

   Output filename. Enter the name of the file into which the report schema should be generated. The default filename extension is **.rdl**.

   Selection option. Indicate which reports you want to include in the schema:

All	Specify all report definitions or a set of reports using a wildcard character (* or ?).
Single	Specify a particular report.

   Report name. If you chose **All** in the Selection option field, you can enter a partial report name with a wildcard character to specify a set of reports. For example, if you type **ORDER***, all report names that start with "ORDER" will be selected. If you type ***ORDER***, all report names that contain the character sequence "ORDER" will be selected.

   If you chose **Single** in the Selection option field, enter the name of the report definition you want to select. To select from a list of reports in the report definition file, select List selections from the Utility functions menu.

   In our example in figure 12-4, a schema for **RPTDAT:reports.rpt** *will be generated to the file* **reps.ddf**. *All existing reports will be included.*

3. To generate the report schema, exit the window.

   If the filename you specified in the Output filename field does not exist, the file is created. If the file already exists, you are prompted

   File already exists. Do you want to delete it?

   - To overwrite the existing file with the new schema, select Yes.
   - To append the new schema to the end of the existing file, select No.
   - To return to the Output filename field and enter another filename, select Cancel.

   When processing is complete, you are returned to the ReportWriter utilities menu.

Sample output from the Generate Report Schema utility

```
;  REPORT DEFINITION LANGUAGE OUTPUT
;
;  REPORTS DEFINITION FILE : RPTDAT:reports.rpt
;
;  CREATION DATE           : 10-NOV-96, 09:40:11
;  REPORTWRITER VERSION    : 6.1

REPORT  "STATISTICS"

FILE PLAYERS.PLAYERS  ID 1
FILE DRAFT.DRAFT  ID 2  PARENT 1  RELATION 1

TEMPTEXT FLAG TTEXT "*" DESCRIPTION "Cost flag"

SUBTOTAL RT_HRUNS TOTALFIELD B_HR BREAKFIELD TEAMNAME TOTALTYPE COMPLETE
    DESCRIPTION "Total Home Runs"

TEMPTEXT HDR_TXT TTEXT "-- Total number of home runs"
    DESCRIPTION "Pre-break text"

QUESTION AVG? TYPE NUMERIC SIZE 3 NODECIMAL BLANKIFZERO
    PROMPT "Batting average:"

CALCULATION AVERAGE TYPE NUMERIC SIZE 3
    EXPRESSION "PLAYERS.B_HITS*1000/PLAYERS.B_ATBATS"
    DESCRIPTION "Avg" FORMAT ".XXX"

SELECT AVERAGE GE AVG? AND B_ATBATS NE "0"

SORT BY TEAMNAME PAGEBREAK BY AVERAGE REVERSE
LINE REPORT_HEADER NOTSEPARATE
TEXT "BASEBALL STATS BY TEAM AND BATTING AVERAGE" COLUMN 1
```

```
LINE PRE-BREAK BREAKFIELD TEAMNAME
FIELD TEAMNAME COLUMN 2 JUST LEFT STRIP
FIELD HDR_TXT COLUMN 24 JUST LEFT
FIELD RT_HRUNS COLUMN 54 JUST RIGHT

LINE DETAIL
FIELD FLAG COLUMN 1 HEADER "" JUST LEFT IF D_COST GE "100"
FIELD LASTNAME COLUMN 4 HEADER "Player" JUST LEFT STRIPCHR ","
FIELD FIRSTNAME COLUMN 21 HEADER "" JUST LEFT
FIELD B_ATBATS COLUMN 35 HEADER "At bats" JUST RIGHT
FIELD B_HITS COLUMN 44 HEADER "Hits" JUST RIGHT
FIELD AVERAGE COLUMN 50 HEADER "AVG" JUST RIGHT
FIELD B_HR COLUMN 56 HEADER "Home runs" JUST RIGHT
FIELD D_COST COLUMN 67 HEADER "Cost" JUST RIGHT FORMAT "ZZZXK" TOTAL
```

Loading a Report Schema

The Load Report Schema utility converts the contents of a Report Definition Language file into one or more report definitions, adding to or updating an existing report definition file. See chapter 17 for more information on Report Definition Language.

1. From the ReportWriter utilities menu, select Load report schema.
2. In the Load Report Schema window, enter data in the fields as instructed below.

   Report definition file. Enter the name of the report definition file into which you want the schema (the Report Definition Language description) loaded. This file must already exist. This field defaults to the name of the report definition file ReportWriter is currently using.

   Input filename. Enter the name of the file that contains the report schema to be loaded. If you don't specify a filename extension, it defaults to **.rdl**.

   Error log filename. Enter the name of the log file to write error messages to. The default log file is **RDLERR.LOG**. The default extension is **.log**. If the file doesn't exist, it is created. If it does exist, it is overwritten.

   Selection option. Indicate which reports you want to load from the schema to the report definition file:

All	Specify all report definitions or a set of reports using a wildcard character (* or ?).
Single	Specify a particular report.

   Report name. If you chose **All** in the Selection option field, you can enter a partial report name with a wildcard character to specify a set of reports. For example, if you type **ORDER***, all report names that start with "ORDER" will be selected. If you type ***ORDER***, all report names that contain the character sequence "ORDER" will be selected.

   If you chose **Single** in the Selection option field, enter the name of the report definition you want to select.

3. To load the report schema, exit the window.

   If a report being loaded already exists in the report definition file, you are prompted

   Report *name* exists. Do you want to replace it?

   - If you select Yes, the existing report will be replaced by the one loaded from the schema.
   - If you press ENTER without entering a value, you are prompted

     Do you want to create a new report?

     If you select No, the current report in the schema file will be skipped. If you press ENTER without entering a value, you are prompted for a new report name. The report will be loaded into the report definition file with the new name.

If you chose **All** in the Selection option field, you will be prompted as indicated above for each report that already exists in the report definition file.

As the report schema is being loaded, status messages are displayed in the lower left corner of the window. The Load Report Schema utility validates all schema (Report Definition Language) statements. When an error occurs within a statement, the utility writes a message to the error log file and ignores the entire statement. The line number of the beginning of the statement precedes the error message.

When processing is complete, ReportWriter displays a message that lists the number of errors that were logged and the number of reports that were loaded.

4. To return to the ReportWriter utilities menu, press ENTER.
5. If any errors were logged, check the log file for specific statement numbers and error messages.
6. Correct the schema file and reload it.

Creating a Report Definition File

The Create Report Definition File utility creates a new report definition file. See "Creating Report Definition Files" on page 14-2 for more information.

1. From the ReportWriter utilities menu, select Create report definition file.
2. At the prompt, enter the name of the report definition file to be created. The default extension is **.ism**.
3. Exit the window.

If the specified file already exists, the Create Report Definition File utility displays an error message and no file is created. You can specify a new filename or delete the old file and run the utility again.

Part 2: Developer’s Reference

13

Distributing ReportWriter with Your Application

Distributing ReportWriter with Your Application

This chapter is intended to guide you in distributing ReportWriter with your application.

Below are three possible scenarios for how you might distribute ReportWriter. Your situation may be different, depending on the requirements of your application.

- You predefine reports, which can subsequently be generated from your application. The user simply selects the report (for example, from a menu or a list of reports) and it is generated. In this scenario, the user never sees ReportWriter, and may not even be aware that it was used to create the reports.
- You provide users with the ReportWriter application and a repository, and allow them to create their own reports. You may want to launch ReportWriter from your application (for example, from a menu option in your application), or you may want to have the user run it as a separate application.
- A combination of the above two scenarios. You provide users with the tools necessary to create their own reports, but you also provide them with some pre-defined reports that they can either use "as is" or as a model for their own reports.

The steps below should serve as a general guide for what you need to do to distribute ReportWriter. Your actual procedures may vary, depending on how much customization you choose to do, how you set up your development system, whether your development system consists of the same operating system and hardware as your target system, and so forth.

1. If desired, customize ReportWriter by doing one or more of the following:

   - Modify ReportWriter's handling of filename mapping, printer interfaces, environment fields, user-defined file types or data types, selection list item information, and report initialization by overloading any or all of ReportWriter's default routines. See chapter 15, "Customizing ReportWriter Routines."

     If the target system is a different endian type than your development system, you must rebuild the ELB that contains these routines. See "Big-endian and little-endian" in the "Synergy/DE on UNIX: The Basics" chapter of the *Professional Series Portability Guide*.

   - Modify help messages, input prompts, or menu column entries and shortcuts in the **rptctl.ism** file. See "Working with the ReportWriter Window Library File" on page 14-3.
   - Modify ReportWriter screen messages, including the title message, by changing the relevant text in the Synergy/DE message file, **syntxt.ism**. See "The Synergy UI Toolkit Control Panel" in the "General Utilities" chapter of *Synergy Tools*.

2. If your customers are currently using Repository version 6.1.x or 6.3.x, and you want to use those definitions with version 10, run the repository conversion program, **rpscnvt.dbr**. This program is distributed with Repository. See the Repository release notes (**REL_RPS.TXT**) for more information.

3. If your customers will be writing their own reports, create repository definitions for this distribution of ReportWriter. For information, see "Creating a New Repository" in the "Utility Functions" chapter of the *Repository User's Guide*.

4. If you are distributing predefined reports that depend on name link associations between the files, or if you want to give this capability to your customers, generate and distribute a repository cross-reference file. For information, see "Generating a Cross-Reference File" in the "Utility Functions" chapter of the *Repository User's Guide*.

5. Include the required application files.

   For UNIX and OpenVMS, refer to the table below, which shows the application files you need to include when you distribute ReportWriter.

   For Windows, you need to install Core Components and ReportWriter before installing your own application. (Note: If you will need to convert a repository, you must copy **rpscnvt.dbr** from the RPS directory and include it with your distribution.)

   For more information about spawning, chaining to, and calling ReportWriter, see chapter 16.

If you are...	Include this file	From this directory
Spawning or chaining to ReportWriter	**rpt.dbr**	RPT
	rptctl.is?	RPTDAT
	synrpt.elb	RPTLIB
	syntxt.is?	SYNTXT
	tklib.elb	WND
Calling ReportWriter with the external subroutine interface	**rptctl.is?**	RPTDAT
	synrpt.elb	RPTLIB
	syntxt.is?	SYNTXT
	tklib.elb	WND
Converting an existing repository	**rpscnvt.dbr**	RPS

6. Include the required data files:

   - The repository main and text files (for example, **rpsmain.is?** and **rpstext.is?**)
   - The cross-reference file, if you are using one (for example, **rpsxref.is?**)
   - The report definition file, if you are distributing predefined reports (for example, **reports.rp?**)

Keep in mind that your repository files and—if you're using one—the cross-reference file must be in sync with one another, must agree with the end-user's data file layout, and must all be distributed together.

For information on moving the repository main and text files to other systems see "Moving repository files" in the "Welcome to Repository" chapter of the *Repository User's Guide*. For information on moving cross-reference files and report definition files, see chapter 14 in this manual.

7. Set the necessary environment variables, such as RPSMFIL and RPSTFIL, to point to the location of the data files.

14

Customizing the ReportWriter Environment

Creating Report Definition Files

All reports created by ReportWriter are stored in a report definition file. The default report definition file is **reports.rpt**, and its location is defined with the RPTDAT environment variable. An empty **reports.rpt** file is included with your distribution.

You can create as many additional report definition files as you like. For example, you may want to create a separate definition file for a customer with customized reports. If you do not use the default report definition file, use the environment variable RPTRFIL to define the report definition file location.

ReportWriter searches for the report definition file as follows:

- If the RPTRFIL environment variable is defined, the report definition filename is the value of RPTRFIL.
- If RPTRFIL is not defined, ReportWriter attempts to open the file as **RPTDAT:reports.rpt**.
- If ReportWriter can't open **RPTDAT:reports.rpt**, it attempts to open **reports.rpt** in the current directory.

Note that the reports definition file must be writable (that is, users must have "write" or "update" permissions), even when users are only viewing reports.

Creating and using a new report definition file

1. Create a new file with the Create Report Definition File utility. See "Creating a Report Definition File" on page 12-15 for instructions.
2. Exit ReportWriter and set the environment variable RPTRFIL to the new file.
3. Restart ReportWriter. It is now using the new report definition file.

Moving report definition files

Once you've defined reports in a report definition file, you can copy that file onto other systems. The advantage of this feature is that it enables you to do all of your development on one system and then move just your finished reports to other systems. To move your report definition files to another system, do one of the following:

- Unload and reload the report definition file. First, unload the report definition with the Unload Report Definition utility (page 12-8). Then, use FTP to transfer the file in binary mode. Finally, use the Load Report Definition utility (page 12-9) to reload the file.
- Create a schema file, transfer it, and then reload it. First, use the Generate Report Schema utility (page 12-10) to generate a Report Definition Language description of your reports. Then, copy the file to another system, and use the Load Report Schema utility (page 12-13) to convert the contents of the Report Definition Language file to a new report definition file. This method works between any systems, because the file generated by the Generate Report Schema utility is a text file.

Working with the ReportWriter Window Library File

Prompts, help messages, shortcuts, menu column headings, and menu entry text reside in the window library file **rptctl.ism**. This window library file is created from the ReportWriter window script file, **rptctl.wsc**.

The **RPTDAT** environment variable defines the location of the ReportWriter window library file.

ReportWriter searches for the window library file as follows:

- If the RPTDAT environment variable is defined, ReportWriter attempts to open the file as **RPTDAT:rptctl.ism**.
- If RPTDAT is not defined, ReportWriter attempts to open **rptctl.ism** in the current directory.

Modifying the contents of a ReportWriter window library file

You may want to modify the contents of a ReportWriter window library file to change shortcuts or translate prompts and help messages to another language.

1. Move to the RPTDAT directory.
2. Modify the **rptctl.wsc** window script file as desired.
3. Run the UI Toolkit script compiler (Script) to store compiled window script information in the window library file, **rptctl.ism**. See "Script: The Script Compiler" in the "Script" chapter of the *UI Toolkit Reference Manual* for a more detailed description of script files and the Script compiler.

Do *not* modify the following items:

- .INPUT, .WINDOW, or .SELECT commands
- Entry names in .ENTRY and .ITEM commands
- Field names or the **select** qualifier entry list in .FIELD commands

We also recommend that you not modify TAB and arrow key shortcuts, as these keys are explicitly referenced throughout the Synergy/DE documentation.

Moving the ReportWriter window library file

If you need to move (or distribute) a ReportWriter window library file to another system, do one of the following:

- Copy the window library files following the general guidelines for moving data files to other systems in "Moving Database Files to Other Systems" in the "Synergy DBMS" chapter of *Synergy Tools*. Note that the ReportWriter window library file contains integer data.
- Copy the customized window script file, **rptctl.wsc**, to the new system and then run the Toolkit script compiler to create the window library file **rptctl.ism**. (This means that UI Toolkit must also be installed on the destination system.)

Creating a Cross-Reference File

A cross-reference file contains records that associate fields with access keys, based on name link matches. ReportWriter uses a cross-reference file when you want to add files to a report based on relationships that aren't defined in the repository.

Because a cross-reference file is optional, one is *not* included in your distribution. To create a cross-reference file, use the Repository Generate Cross-Reference utility. For instructions, see "Generating a Cross-Reference File" in the "Utility Functions" chapter of the *Repository User's Guide*.

Once created, this file is never read or modified by Repository—only ReportWriter reads this file. To ensure that this file contains valid data, you should regenerate it each time you modify your repository.

By default, the cross-reference file is named **rpsxref.ism**, and its location is defined with the RPSDAT environment variable. If you want to name your cross-reference file differently, set the RPSXFIL environment variable to the desired file.

ReportWriter searches for the cross-reference file as follows:

- If the RPSXFIL environment variable is defined, the cross-reference filename is the value of RPSXFIL.
- If RPSXFIL is not defined, ReportWriter attempts to open the file as **RPSDAT:rpsxref.ism**.
- If **RPSDAT:rpsxref.ism** can't be opened, ReportWriter attempts to open **rpsxref.ism** in the current directory.

Moving cross-reference file

Move the cross-reference file following the general guidelines for moving data files to other systems in "Moving Database Files to Other Systems" in the "Synergy DBMS" chapter of *Synergy Tools*.

15

Customizing ReportWriter Routines

Getting Information about a Selection Window Item 15-51

Modifying ReportWriter Initialization 15-52

Using User-Overloadable Routines

ReportWriter provides a number of subroutines and functions that you can overload, or override, with your own routines. You can name these routines anything you want, but you must register the name with ReportWriter using the RW_METHOD subroutine.

Some of these routines perform functions that you can adapt to your own requirements. For example, the RW_PRTCLOSE_METHOD routine supplied with ReportWriter closes the channel and submits the temporary report file to the default printer queue with the Synergy DBL LPQUE statement. You can overload this routine with one that prompts the user for a printer to which output should be sent instead of automatically printing to the default printer.

Other routines do nothing and are simply called to enable you to do any processing you desire at that point in the program. For example, if you want to map the open filename from **ARO*xxx*.ism** to the actual data filename, which is based on the customer number, you can replace the empty RPS_FILNAM_METHOD routine with one that makes the necessary filename substitution.

The user-overloadable routines enable you to provide or modify the processing of user-defined file types, file definition open filenames, printer output channels, environment fields, user-defined data type fields, selection list item information, and report initialization.

You must put your overloaded routines in an ELB (or shared image on OpenVMS). You can either set the logical RWUSRLIB to point to this ELB, or, if you are using the external subroutine interface, you can pass the ELB name as a parameter to %RW_INIT. (See "Accessing ReportWriter by External Subroutine Interface" on page 16-3.) On OpenVMS, ReportWriter will call OPENELB on the shared image that is specified with RWUSRLIB or passed to %RW_INIT before attempting to call the routines.

ReportWriter user-overloadable routine names begin with "RW_"or "RPS_" and end with "_METHOD". You will never call these routines directly; they are called only by ReportWriter. Pages 15-7 through 15-52 give the syntax and arguments for each routine and provide either the contents of the routine as it is distributed with ReportWriter or a sample user-defined version. The following routines can be overloaded at runtime:

- RPS_CLOSE_METHOD
- RPS_DATA_METHOD
- RPS_FILNAM_METHOD
- RPS_OPEN_METHOD
- RPS_READ_METHOD
- RPS_READS_METHOD
- %RW_CENTURY_METHOD
- RW_ENV_METHOD
- RW_FOOTER_METHOD
- RW_HEADER_METHOD
- RW_GETVAL_METHOD
- RW_INIT_METHOD
- RW_INITBLD_METHOD
- RW_LSTINFO_METHOD
- RW_PRTCLOSE_METHOD
- RW_PRTOPEN_METHOD
- RW_USAGE_METHOD

Overloading a user-overloadable routine in ReportWriter

1. Write your customized routines, including RW_INIT_METHOD. (See page 15-7 for information about RW_INIT_METHOD. A sample routine is included with your Synergy/DE distribution in the file **overload.dbl**.)

2. If you're *not* using the external subroutine interface to call ReportWriter, name the RW_INIT_METHOD routine USR_RW_INIT.

   If you *are* using the external subroutine interface, you can optionally pass the *user_init* argument to %RW_INIT to specify a different name for RW_INIT_METHOD. See page 16-13 for more information about %RW_INIT.

3. Compile the Synergy DBL source file(s) containing your customized routines.

4. If your customized routines are in more than one source file, use the librarian to create an OLB:

On...	Use this command...
Windows and UNIX	`dblibr -c` *library_file*`.olb` *source_1 source_2 ...*
OpenVMS	`libr /create` *library_file source_1,source_2,...*

5. Create an ELB or shared image file that contains your routines:

On...	Do this...
Windows and UNIX	Link your routines as follows: `dblink -l` *library_file library_file*`.olb`
OpenVMS	Follow the guidelines in "Building Shared Images" in the "Building and Running Traditional Synergy Applications" chapter of *Synergy Tools*. Edit the file **rwusrbld.com**. If you have changed the routine names from those specified in the manual, find the "procedure" and "universal" entry for each routine and change it appropriately. Delete or comment out any unused entries.

6. Distribute the ELB or shared image, as well as ReportWriter, with your application. Set the RWUSRLIB environment variable to the full path and filename of the ELB or shared image you created in step 5. (You may want to add this step to your setup scripts.) Or, if you're using the external subroutine interface to call ReportWriter, you can pass the *user_routine_elb* argument to %RW_INIT to specify a logical that points to the ELB or shared image.

   See "Error messages" on page 16-23 for a list of error messages that might be generated.

Limitations of overloading

ReportWriter always overloads any EHELP_METHOD subroutine with TKP_HELP during processing.

If your application overloads the UI Toolkit close method subroutine (ECLOSE_METHOD), ReportWriter saves your close method's address and logs its own close method. If the close method is called, ReportWriter explicitly calls your close method first. If your close method returns *a_cancel* as a nonzero value, the close method will be aborted. If *a_cancel* is returned as 0, ReportWriter will perform all tasks necessary for shut-down. If the user is prompted to save the report, Cancel is not an option in the message box. If your application doesn't overload the close method, the user gets a Cancel option if prompted to save the report.

Overloading a UI Toolkit routine in ReportWriter (OpenVMS only)

On OpenVMS, if you are overloading ReportWriter routines with Toolkit routines that access Toolkit methods, and those methods don't support an ELB (shared image) argument, you have to link the Toolkit methods so that ReportWriter can find them at runtime. You do this by creating a new ReportWriter main routine that wraps the ReportWriter executable and includes “dummy” calls to your overloaded Toolkit methods. When you link the new main routine with the shared image that contains your methods, you enable OpenVMS to find your methods within the image's link map.

For example, say you want to use RW_PRTOPEN_METHOD and display a list of available printers. To build the Toolkit list, you would write a Toolkit list load method and register it with L_METHOD. Your version of RW_PRTOPEN_METHOD might look something like this:

```
subroutine MY_RW_PRTOPEN_METHOD
.
.
.
; Register your list load method
xcall L_METHOD (list_id, D_LLOAD, "MY_PRINTER_LIST")
; Begin list processing
```

MY_PRINTER_LIST is the Toolkit list load method you wrote. The Toolkit routine L_METHOD does not provide a way to register the shared image that MY_PRINTER_LIST resides in. Therefore, a call to MY_PRINTER_LIST must be included in your new ReportWriter main routine:

```
main MY_RPT
proc
    xcall rpt_main  ; Call ReportWriter program. This is equivalent to
                    ; typing $RUN RPT:RPT.EXE at the command line
    stop            ; Processing stops here, so following not called
    xcall MY_PRINTER_LIST    ; Dummy call to your Toolkit list load method
endmain
```

After you write the main routine, compile it and link it with the SYNRPT shared image and the shared image you created that contains your methods. Then, use this new executable when you run ReportWriter instead of the distributed ReportWriter executable.

⚠ The RPT_MAIN subroutine is supported only for the use described above. Do not use it in any other way.

Initializing User-Overloadable Routines

%RW_INIT calls RW_INIT_METHOD before opening any files, thereby enabling you to register any of your user-overloadable routines for use by %RW_REPORTS.

RW_INIT_METHOD – Initialize user-overloadable routines

```
subroutine RW_INIT_METHOD
    a_status   ,n    ;RETURNED - You must set this to be 0 if there
                     ; are no errors and to a nonzero value if there
                     ; is an error.
```

Discussion

RW_INIT_METHOD should call %RW_METHOD or %RPS_METHOD to register your customized routines. If any error occurs in a given %RPS_METHOD or %RW_METHOD, no further tries are made. %RW_INIT will return the error information from the routine that failed.

RW_INIT_METHOD can also include calls to UI Toolkit's E_METHOD subroutine to overload desired Toolkit routines such as EHELP_METHOD, EFKEY_METHOD, and so on. After the return from this call, ReportWriter will register its own overloaded Toolkit routines.

RW_INIT_METHOD is also an appropriate place from which to call the Synergy DBL LOCALIZE subroutine, to apply European formatting conventions to numeric values.

Examples

```
subroutine rw_init_method
    a_sts     ,n

.include "RPTLIB:reports.def"

record
       err    ,i4

proc
; Comment out any routines that you do not overload
    err = %rps_method(M_RPS_CLOSE, "RPS_CLOSE_METHOD")
    err = (err .or. %rps_method(M_RPS_OPEN, "RPS_OPEN_METHOD"))
    err = (err .or. %rps_method(M_RPS_READ, "RPS_READ_METHOD"))
    err = (err .or. %rps_method(M_RPS_READS, "RPS_READS_METHOD"))
    err = (err .or. %rps_method(M_RPS_FILNAM, "RPS_FILNAM_METHOD"))
    err = (err .or. %rw_method(M_RW_PRTOPEN, "RW_PRTOPEN_METHOD"))
    err = (err .or. %rw_method(M_RW_PRTCLOSE, "RW_PRTCLOSE_METHOD"))
    err = (err .or. %rw_method(M_RW_ENV, "RW_ENV_METHOD"))
    err = (err .or. %rw_method(M_RW_HEADER, "RW_HEADER_METHOD"))
    err = (err .or. %rw_method(M_RW_FOOTER, "RW_FOOTER_METHOD"))
    err = (err .or. %rps_method(M_RPS_DATA, "RPS_DATA_METHOD"))
```

```
        err = (err .or. %rw_method(M_RW_GETVAL, "RW_GETVAL_METHOD"))
        err = (err .or. %rw_method(M_RW_USAGE, "RW_USAGE_METHOD"))
        err = (err .or. %rw_method(M_RW_LSTINFO, "RW_LSTINFO_METHOD"))
        err = (err .or. %rw_method(M_RW_INITBLD, "RW_INITBLD_METHOD"))
        a_sts = err

    ; You can overload UI Toolkit user-defined routines for ReportWriter.
    ; Here, the check field method is overloaded for checking the input value
    ; of a user-defined question field.

        xcall e_method(D_METH_CHKFLD, "ECHKFLD_METHOD")
        xreturn
    endsubroutine
```

Supporting User-Defined File Types

ReportWriter (and Repository) explicitly support three file types: ASCII, DBL ISAM, and relative. They also support the user-defined file type, which enables you to provide your own support for any additional file types using I/O subroutines that you write yourself.

You can overload these I/O subroutines in ReportWriter. Whenever ReportWriter performs an I/O function where the file type is user-defined, it calls one of the four routines listed below.

The versions of these routines linked with your original ReportWriter distribution are “dummy” routines; they simply return. You can overload these routines with your own versions that provide support for file types not supported by ReportWriter or Repository.

RPS_CLOSE_METHOD – Close a channel

```
subroutine RPS_CLOSE_METHOD
    a_channel       ,n     ;I/O channel (d3)
    a_user_area     ,a     ;User-defined data area (a60)
    a_record        ,n     ;Record number, if file is relative (d5)
    a_error         ,n     ;Returned with an error code (d2)
```

RPS_OPEN_METHOD – Open a channel

```
subroutine RPS_OPEN_METHOD
    a_channel       ,n     ;I/O channel returned (d3)
    a_mode          ,a     ;I/O mode in which to open the channel
                           ; (for example, U:I) (a3)
    a_filename      ,a     ;Name of file to open (Not a file
                           ; definition name.) (a64)
    a_user_area     ,a     ;User-defined data area (a60)
    a_record        ,n     ;Record number, if file is relative (d5)
    a_error         ,n     ;Returned with an error code (d2)
```

RPS_READ_METHOD – Read a record

```
subroutine RPS_READ_METHOD
    a_channel       ,d3    ;I/O channel to use (d3)
    a_record        ,a     ;Record returned
    a_key_val       ,a     ;Key value that identifies the record
    a_key_ref       ,n     ;Explicit key of reference; if no keys are
                           ; defined for the structure, this value is
                           ; -1 (d1)
    a_user_area     ,a     ;User-defined data area (a60)
    a_record        ,n     ;Record number, if file is relative (d5)
    a_error         ,n     ;Returned with an error code (d2)
```

RPS_READS_METHOD – Read a record sequentially

```
subroutine RPS_READS_METHOD
    a_channel       ,n     ;I/O channel to use (d3)
    a_record        ,a     ;Record returned
    a_user_area     ,a     ;User-defined data area (a60)
    a_record        ,n     ;Record number, if file is relative (d5)
    a_error         ,n     ;Returned with an error code (d2)
```

Error codes to return from I/O subroutines

These are the valid error codes to return from the four I/O subroutines listed above:

```
DD_IO_ERROR         ,-1    ;Operation error
DD_IO_NORMAL        ,0     ;Normal operation
DD_IO_NOFIND        ,1     ;"Record not found" error
DD_IO_UNKNOWN       ,2     ;Unknown error
DD_IO_INVFIL        ,3     ;Invalid file type
DD_IO_EOF           ,4     ;End of file
DD_IO_CANCEL        ,8     ;Interrupt signal entered
```

Sample user-defined file type I/O subroutines

This example illustrates the use of the user-defined file type for supporting an MCBA-like file structure. Also shown is the definition file this subroutine uses. The files containing these routines and their record definitions are included in your distribution.

```
;-------------------------------------------------------------------
;
; Source:           USRDCTIO.DEF
;
; Description:      User-defined file type I/O control error
;                   codes and example user control area records
;
;------------------------------------------------------------------
;
; Define error codes
;
.define DD_IO_ERROR       , -1         ;Operation error
.define DD_IO_NORMAL      , 0          ;Normal operation
.define DD_IO_NOFIND      , 1          ;Record not found
.define DD_IO_UNKNOW      , 2          ;Unknown error
.define DD_IO_INVFIL      , 3          ;Invalid file type
.define DD_IO_EOF         , 4          ;End of file
.define DD_IO_CANCEL      , 8          ;Interrupt signal entered
```

```
; Example user file type definition area

record usr_type                         ;User area 1-15
    usrtyp          ,a15                ;User file type
     usr_ftyp        ,a1  @usrtyp+1     ;Specific file type

; Example user file structure for MCBA master type

record usr_mcba                         ;User area 16-41
    idx_chn         ,d2                 ;Index file channel
    keyref          ,d1                 ;Key reference used
    orgcnt          ,d5                 ;Organized record count
    reccnt          ,d5                 ;Record count
    maxrec          ,d5                 ;Maximum record count
    delcnt          ,d3                 ;Delete record count
    recnum          ,d5                 ;Record number (pointer)

.ifndef SHOW_DEF_LIST
.START NOPAGE, LIST
.endc

;-------------------------------------------------------------------
;
; Source:           USRMCBA.DBL
;
; Description:      Sample user-defined file type I/O access
;                   routines for an MCBA-like file
;
; Routines:         RPS_OPEN_METHOD, RPS_READ_METHOD,
;                   RPS_READS_METHOD, RPS_CLOSE_METHOD
;
;-------------------------------------------------------------------
subroutine rps_open_method
;
; Description: This is a sample user-defined file type open routine.
;
; Arguments:
;
     a_chn          ,n            ;Returned open channel
     a_mod          ,a            ;Returned file open mode
                                  ; in this example, not used
     a_filnam       ,a            ;Open filename
     a_userarea     ,a            ;Returned user control area
     a_ddarea       ,n            ;Returned record number
     a_sts          ,n            ;Returned status
```

```
; Special Notes:
;   This example doesn't use "RPS_FILNAM_METHOD" to process the open
;   filename. Instead, this routine processes the filename tag character
;   to determine the necessary operation.
;
;   User file types are specified at the end of the open filename.
;
;   User file types supported:
;     "@M" type file: MCBA-like standard master file with index (example
;     file specification DAT:CUSMAS, IDX:CUSIDX@M)
;
; User data area structure:
;
; usrtyp            ,a15          ;User file type
; idx_ch            ,d2           ;Index file channel
; keyref            ,d1           ;Key reference used
; orgcnt            ,d5           ;Organized record count
; reccnt            ,d5           ;Record count
; maxrec            ,d5           ;Maximum record count
; delcnt            ,d3           ;Delete record count
; recnum            ,d5           ;Record number (pointer)

.define SHOW_DEF_LIST
.include "usrdctio.def"

record
    len             ,d2
    mstlen          ,d2
record  rec_buf
    buffer          ,a200         ;Temporary buffer to read the MCBA
                                  ; control record
proc
    clear usr_mcba
    len = %trim(a_filnam)         ;Get actual size of the filename
    if (a_filnam(len-1:2).eq."@M") then
      begin
        a_mod = "i"
        usrtyp = "@M"             ;Load user type
        len = len - 2
        call mcba_master
        xreturn                   ;We are done here!
      end
    else                          ;Invalid file type
      begin
        a_sts = DD_IO_INVFIL
        xreturn
      end
    xreturn
```

```
mcba_master,
;
; Do MCBA master file type open
;
    mstlen = %instr(1, a_filnam(1,len), ',')   ;Find delimiter
    if (.not.mstlen)             ;Invalid file specification
      begin                      ;Index file required for this type
        a_sts = DD_IO_INVFIL
        xreturn
      end
                                 ;Open the index file and store channel
                                 ; in the user data area
    xcall u_open(idx_chn, "i", a_filnam(mstlen+1,len),,, a_sts)
    if (a_sts)
      xreturn
    len = mstlen - 1             ;Get the master filename length
    xcall u_open(a_chn, a_mod, a_filnam(1,len),,, a_sts)
    if (a_sts)
      xreturn
                                 ;We don't care about actual record size
    reads(a_chn, rec_buf)  [eof=nofind, err=errexit]
    reccnt = rec_buf(1,5)        ;Record count
    recnum = 2                   ;And set next record to read
    a_userarea(1,15) = usrtyp    ;Save user file type
    a_userarea(16,41) = usr_mcba ;Save control in user area
    return
errexit,
    a_sts = DD_IO_ERROR
    return

nofind,
    a_sts = DD_IO_NOFIND
    return
end

subroutine rps_read_method
;
; Description:      Sample user-defined file type random read routine.
;
; Arguments:
;
     a_chn          ,n           ;File open channel
     a_recbuf       ,a           ;Returned record buffer
     a_keyval       ,a           ;Search key value
     a_keyref       ,n           ;Search key reference ID
     a_userarea     ,a           ;User control data area
     a_ddarea       ,n           ;Returned record number
     a_sts          ,n           ;Returned status
```

```
.include "usrdctio.def"

record
    len             ,d5
    rtn             ,d5

proc
    usrtyp = a_userarea(1,15)           ;Get user type
    len = %len(a_recbuf)                ;Get size of record buffer
    if (usrtyp.eq."@M") then            ;Do random read on user case
      begin
        usr_mcba = a_userarea(16,41)    ;Get local user area
        xcall mcba_search(idx_chn, reccnt, a_keyval, a_keyref,
  &                       recnum, a_sts)
        if (a_sts)
          goto nofind
        get(a_chn, a_recbuf(1,len), recnum) [err=errexit, eof=nofind]
        keyref = a_keyref               ;Save for sequential read by key
        a_userarea(16,41) = usr_mcba    ;Save update
      end
    else                         ;DBL ISAM
      goto errexit
     xreturn

errexit,
    a_sts = DD_IO_ERROR
    xreturn

nofind,
    a_sts = DD_IO_NOFIND
    xreturn
end

subroutine rps_reads_method
;
; Description: Sample user-defined file type sequential read routine.
;
; Arguments:
;
     a_chn          ,n           ;File open channel
     a_recbuf       ,a           ;Returned record buffer
     a_userarea     ,a           ;User control data area
     a_ddarea       ,n           ;Returned record number
     a_sts          ,n           ;Returned status
```

```
.include "usrdctio.def"

record
    len             ,d5
    rtn             ,d3
    arecsiz         ,a4

proc
    usrtyp = a_userarea(1,15)     ;Get the user type
    len = %len(a_recbuf)          ;Get size of record buffer
    if (usrtyp.eq."@M") then      ;Do sequential read on user case
      begin
        usr_mcba = a_userarea(16,41);Get local user area
        xcall mcba_search(idx_chn, reccnt,, keyref, recnum, a_sts)
        if (a_sts)
          xreturn
        get(a_chn,a_recbuf(1,len), recnum)
  &         [eof=nofind,key=nofind,err=errexit]
        a_userarea(16,41) = usr_mcba    ;Save update
      end
    else
      goto errexit
    xreturn

errexit,
    a_sts = DD_IO_ERROR
    xreturn

nofind,
    a_sts = DD_IO_NOFIND
    xreturn
end

subroutine rps_close_method
;
; Description: Sample user-defined file type close routine.
;
; Arguments:
;
     a_chn                ,n           ;File open channel
     a_userarea           ,a           ;User control data area
     a_ddarea             ,n           ;Record number
     a_sts                ,n           ;Returned status

.include "usrdctio.def"
```

```
proc
    usrtyp = a_userarea(1,15)
    xcall u_close(a_chn)
    if (usrtyp.eq."@M")
      begin
        usr_mcba = a_userarea(16,41)
        xcall u_close(idx_chn)
      end
    clear a_userarea
    xreturn
end

subroutine mcba_search
;
; Description:Search MCBA master index and return record number
;
; Arguments:
;
     a_chn          ,n            ;Index file open channel
     a_reccnt       ,n            ;Organized record count
     a_keyval       ,a            ;Index key value
     a_keyref       ,n            ;Index key reference ID - 0 based
     a_recnum       ,n            ;Returned record number
     a_sts          ,n            ;Returned status
;
; Special Notes:
;   This routine assumes the index file contains a single keyed index with
;   a record pointer (number), and the index file is sorted by the index.
;

.include "usrdctio.def"

record  idx_rec1                  ;Sample index record structure
    index_key       ,a50          ; Index part
    rec_num         ,d5           ; Pointer part
                    ,a1           ; Record Terminator
record  idx_rec, X                ;Sample index record structure
    idxrec          ,a55          ; Data only

record                            ;For binary search
    first           ,d5
    last            ,d5
    saved           ,d5
```

```
proc
; Do necessary search operation on the a_keyval and a_keyref
    if (.not.%passed(a_keyval)) then  ;Sequential read on keyref
      call  do_seq
    else if (a_keyval.eq.' ') then    ;Initial read on keyref
      call do_seq
    else
      call do_random
    a_recnum = rec_num + 1            ;Increment by one for control record
    xreturn

do_seq,
    do
      reads(a_chn, idx_rec)  [eof=nofind, err=errexit]
    until (idx_rec.ne.']' .and. index_key.ne.'     ')
    return

do_random,
; Do sequential search or binary search
  if (.not.%passed(a_reccnt)) then        ;Sequential search
    do forever
      begin
        reads(a_chn, idx_rec)[eof=nofind, err=errexit]
        if (index_key.eq.a_keyval)
          return
      end
  else                                    ;Do binary search
    begin
      recnum = a_reccnt / 2               ;Initialize the indexes
      first = 1
      last = a_reccnt
      do forever
        begin
          saved = recnum                  ;Save last middle index
          get(a_chn, idx_rec1, recnum) [eof=nofind, err=errexit]
          if (index_key.eq.a_keyval) then ;Found
            return
          else if (recnum.eq.first .or. recnum.eq.last)  ;Not found
            goto nofind
          else if (index_key.gt.a_keyval) ;Try left half
            begin
              if ((last-recnum).eq.1) then ;No middle item set to first
                decr recnum
              else                         ;Set the next middle index
                recnum = recnum - (last-recnum) / 2
              last = saved
            end
          else                            ;Try right half
            begin
              if ((last-recnum).eq.1) then ;No middle item set to last
                incr recnum
              else                        ;Set the next middle index
                recnum = recnum + (last-recnum) / 2
              first = saved
            end
        end
    end
return

nofind,
    a_sts = DD_IO_NOFIND
    xreturn

errexit,
    a_sts = DD_IO_ERROR
    xreturn
end
```

Modifying Filenames at Runtime

When you need multiple versions of the same file definition (for example, if you have an "ARO*xxx*" filename, where *xxx* is a client-specific code), Repository enables you to create one file definition and then map the filename when necessary. Every time ReportWriter opens a file, the RPS_FILNAM_METHOD subroutine is called to do any filename mapping. You can supply your own version of this subroutine and link it with ReportWriter.

RPS_FILNAM_METHOD – Map a filename

```
subroutine RPS_FILNAM_METHOD
    a_inp_file      ,a     ;Original filename (a64)
    a_out_file      ,a     ;Returned with mapped filename (a64)
```

Discussion

ReportWriter passes the data filename that you entered at the Open filename prompt in your file definition to this subroutine as **a_inp_file**. For example, if you've defined the file that contains your customer data, you could map the data filename to the appropriate name for your customer and return that name as **a_out_file**.

The version of RPS_FILNAM_METHOD that is supplied with your original ReportWriter distribution returns the original filename, unmodified, in **a_out_file**.

If you're using *xf*ODBC, this routine must be named USR_DD_FILNAM. For additional information, see "Using USR_DD_FILNAM to change replaceable characters" in the "Preliminary Steps" chapter of the *xfODBC User's Guide*.

Examples

This is an example of a user-supplied filename mapping subroutine which replaces various tokens contained in the filenames.

```
;
; Source:           UTS:usrdd.def
;
; Description:      Global data section for sample rps_filnam_method
;                   routine
;

.ifdef USRDD_INIT
global data section usrdd, init
.iff
global data section usrdd
.endc
```

```
record usrdd_ctls
    usrdd_needinfo         ,d1,  1       ;Cleared when info supplied
    usrdd_cltdesc          ,a30, 'Client Number'
    usrdd_pathdesc         ,a30, 'Data Location'
    usrdd_cltno            ,d5           ;Client Number
    usrdd_ttnum            ,d3           ;Terminal Number
    usrdd_path             ,a40          ;Data Location
    usrdd_pathlen          ,d2           ;%trim of usrdd_path

endglobal

subroutine rps_filnam_method
;
; Description:  Perform filename mapping
;
; The location of #### is replaced by the 4-digit client number.
; The location of ##### is replaced by the 5-digit client number
; The location of %%% is replaced by the 3-digit terminal number.
; The location of $$$ is replaced by the variable-length data path.
;
    a_filnam        ,a            ;Repository filename
    a_retnam        ,a            ;Returned filtered filename

.define USRDD_INIT
.include "UTS:usrdd.def"          ;Global region

record
    retlen          ,d3           ;Return value length
    loc             ,d3           ;Instr pointer

proc
    a_retnam = a_filnam           ;Map filename initially
    retlen = %len(a_retnam)

    while (%instr(1, a_retnam, '#####', loc))
      begin
        if (usrdd_needinfo)
          xcall rps_getinfo_method
        a_retnam(loc:5) = usrdd_cltno, 'XXXXX'
      end

    while (%instr(1, a_retnam, '####', loc))
      begin
        if (usrdd_needinfo)
          xcall rps_getinfo_method
        a_retnam(loc:4) = usrdd_cltno, 'XXXX'
      end
    while (%instr(1, a_retnam, '%%%', loc))
      begin
        if (usrdd_needinfo)
          xcall rps_getinfo_method
        a_retnam(loc:3) = usrdd_ttnum
      end

    while (%instr(1, a_retnam, '$$$', loc))
      begin
        if (usrdd_needinfo)
          xcall rps_getinfo_method
        a_retnam(loc+usrdd_pathlen, retlen) = a_retnam(loc+4, retlen)
        a_retnam(loc:usrdd_pathlen) = usrdd_path
      end
xreturn
endsubroutine

subroutine rps_getinfo_method
;
; Description: Get information from the user for rps_filnam_method
;
; Arguments: None

.include "UTS:usrdd.def"
.include "WND:tools.def"

record
    chn             ,d2           ;Library channel
    id              ,d2           ;Window ID
    srch            ,d2           ;Search variable

proc
    xcall tnmbr(usrdd_ttnum)
    xcall e_enter
    xcall u_open(chn, 'I:I', 'SYR:utwlib', srch)
    xcall i_ldinp(id, chn, 'utdd_info', srch)
    xcall i_dspfld(id, 'cltdesc', usrdd_cltdesc)
    xcall i_dspfld(id, 'pathdesc', usrdd_pathdesc)
    if (usrdd_path) then
      nop
    else
      usrdd_path = 'DAO:'         ;Default
    repeat
      begin
        xcall i_input(id,, usrdd_ctls,,,,D_NOTERM)
        if (g_select) then
          begin
            case g_entnam of
```

```
              begincase
                'O_EXIT':
                  exitloop
              endcase
            else
              call beep
          end
        else
          if (.not. g_setsts)
            exitloop
      end

    xcall e_exit
    clear usrdd_needinfo
    usrdd_pathlen = %trim(usrdd_path)
    xreturn

beep,
    display(g_terminal, 7)
    return
endsubroutine
```

Opening and Closing ReportWriter Printer Channels

You can overload ReportWriter's default printer interface with user-supplied routines to open and close the output channel for the Print report and Generate report file functions.

RW_PRTOPEN_METHOD – Open a file/printer for printing

```
subroutine RW_PRTOPEN_METHOD
a_channel     ,n     ;Returned as the channel for output (d2)
a_file        ,a     ;Either name of report file to generate, if
                     ; print_flag is zero, or unique filename
                     ; that may be used for temporary file storage,
                     ; if print_flag is nonzero (a30)
a_prnt_flg    ,n     ;Indicates whether report is being printed (d1)
                     ; 0   Generate report file; do not print it.
                     ; 1   Print report.
a_quit        ,n     ;Returned with true if no further report
                     ; generation should occur (d1)
a_mode        ,n     ;Optional, file open mode. If passed and
                     ; nonzero, output file will be opened in
                     ; append mode. Otherwise, it will be opened
                     ; in creation mode (d1)
```

Discussion

The version of this routine linked with the original ReportWriter distribution opens the specified file and returns the channel. If an error occurs on open, **a_quit** is returned with the error code. Any user-overloaded version of this routine should at minimum perform these functions. ReportWriter outputs the report to the channel returned in **a_channel**, using the Synergy DBL WRITES and FORMS statements.

Examples

This is the version of RW_PRTOPEN_METHOD that is linked with your original ReportWriter distribution.

```
subroutine rw_prtopen_method
;
; Description:  Printer open routine for ReportWriter
;
; Arguments:
;
    a_chan          ,n     ;Output channel (incoming unassigned)
    a_file          ,a     ;A unique filename which may be used
    a_qflg          ,n     ;Queue flag - 1 = Print, 0 = Create file
    a_quit          ,n     ;Returned true if no further report generation
                           ; should occur
    a_mode          ,n     ;Optional file open mode
```

```
.include "WND:tools.def"
.include "RPT:rptmsg.def"
.include "DBLDIR:dbl.def"

record
    line            ,a80
    linlen          ,d2

proc
.ifdef D_VMS
    ;If this is printing on an OpenVMS system, use the O:S format
    if (a_qflg) then
      line = "/SI=96/DEQ=96/BUFNUM=2/BUFSIZ=16/IO=O:S"
    else
.endc
    ;Part of the above 'else'
    clear line
    if (%passed(a_mode)) then
      begin
        if (a_mode) then
           xcall u_open(a_chan, 'A', a_file, line,, a_quit)
        else
           xcall u_open(a_chan, 'O', a_file, line,, a_quit)
      end
    else
      xcall u_open(a_chan, 'O', a_file, line,, a_quit)
    if (a_quit)
      begin
        xcall ru_sbld(line, linlen, MSG_OOPNERR, a_file)
        xcall u_message(line(1:linlen), D_ERROR)
      end
    xreturn
endsubroutine
```

RW_PRTCLOSE_METHOD – Close the output file/printer

```
subroutine RW_PRTCLOSE_METHOD
a_chan       ,n     ;Channel used for output (d2)
a_file       ,a     ;Filename passed to RW_PRTOPEN_METHOD (a30)
a_pflg       ,n     ;Indicates whether report is being printed (d1)
                    ; 0   Generate report file; do not print it.
                    ; 1   Print report.
```

Discussion

The version of this routine linked with the original ReportWriter distribution closes the channel specified by **a_chan**, and if **a_pflg** is true, the file specified by **a_file** is submitted to the queue via the Synergy DBL LPQUE statement. At a minimum, any user-supplied version of this routine must close **a_chan**.

Examples

This is the version of RW_PRTCLOSE_METHOD that is linked with your original ReportWriter distribution.

```
subroutine rw_prtclose_method
;
; Description:  ReportWriter printer close routine
;
; Arguments:
;
    a_chan          ,n     ;Channel on which output performed
    a_filnam        ,a     ;Filename passed to RW_PRTOPEN_METHOD
    a_qflg          ,n     ;0 = Generating report file; don't print
                           ;1 = Output to printer

.include "WND:tools.def"
.include "RPT:rptmsg.def"

proc
    xcall u_close(a_chan)
    if (a_qflg)
      begin
        xcall e_state(D_ON, D_INTR)      ;Allow interrupt
        onerror noprt
        lpque(a_filnam, DELETE)
        offerror
        xcall e_state(D_OFF, D_INTR)     ;Disallow interrupt
      end
    xreturn
```

```
noprt,
    offerror
    xcall ru_message(MSG_NOPNTR, D_ERROR)
    xcall e_state(D_OFF, D_INTR)  ;Disallow interrupt
    xreturn
endsubroutine
```

Here's another example:

```
subroutine rw_prtclose_method
;
;Description: Processes printer close for ReportWriter
;
;Arguments:
;
    a_channel       ,n     ;Channel used for output. Must be closed
                           ; by routine
    a_filename      ,a     ;Filename passed to RW_PRTOPEN_METHOD
    a_genprt        ,n     ; 0 - Generate report file - do not print
                           ; 1 - Printer output
;
;Notes:
;      This subroutine is called when ReportWriter is finished
;      generating a printed report.
;
;      This routine requires an external file that has the following
;      format:
;
;      "Name you want user to see", "LPQUE queue name"
;
;      In other words, the format is a single quoted string with the name
;      you want your users to see, followed by a comma, followed by
;      another quoted string with the queue name LPQUE uses for printing
;      on the particular device.
;
;      This file is named PRT:print.lst, although you can change the
;      filename by changing the OPEN statement in the fifth line after
;      the PROC statement.
;
;      In addition, the following needs to be placed in the window script
;      file opened as win_libchn:
;
; --------------------------------------------------------
;
; print.wsc - window script file for rw_prtclose_method
;
;---------------------------------------------------------
;;
```

```
;; Menu column definition
;;
;.column s_select, "Selection functions"
;.entry s_down, "Next selection", key(down)
;.entry s_up, "Prior selection", key(up)
;.end
; --------------------------------------------------------
;
.include "WND:tools.def"
.include "WND:windows.def"

.define ITEMS, 10

.align
static record
    win_libchn      ,i4    ;Window library channel
    colid           ,i4    ;Column ID for input window
    wndid           ,i4    ;Input window ID
    list_ix         ,i4    ;List index

    name            ,ITEMS a30   ;List of printer names
    device          ,ITEMS a15   ;List of printer system device names
.align
record
    file_chn        ,i4    ;File channel
    error           ,i4    ;Error number
    start_loc       ,i4    ;Start of parse string
    end_loc         ,i4    ;End of parse string
    sel_printer     ,i4    ;Selected printer
    status          ,i4    ;Status of routine
    prt_name        ,a40   ;Printer name
    prt_device      ,a15   ;Printer's device name for operating system
    wnd_name        ,a25   ;Window name
    buffer          ,a80   ;Buffer for buffered output
    item_pos        ,ITEMS d2    ;Item position list for s_parse
    item_length     ,ITEMS d2    ;Item length list for s_parse
    item_type       ,ITEMS d2    ;Item type list for s_parse
    num_items       ,d2          ;Number of items for s_parse

proc
    xcall u_close(a_channel)
    if (.not.list_ix)            ;Has list been loaded?
      begin
        xcall u_open(file_chn, 'I', "PRT:print.lst",,, error)
        if (error)
          goto no_print
```

```
            repeat
              begin
                reads(file_chn, buffer, err)
                if (buffer(1:1).eq.';')     ;Comment line
                  nextloop
                call get_prtinfo
                incr list_ix         ;Load lists
                name[list_ix] = prt_name
                device[list_ix] = prt_device
              end

    err,
            xcall u_close(file_chn)
          end
        if (.not.win_libchn)         ;Has window library been loaded?
          begin
            xcall u_open(win_libchn, 'I:I', "PRT:print",,, error)
            if (error)
              goto no_winlib
          end
        xcall e_enter
        xcall m_column(D_REMOVE, D_LOCAL)  ;Remove unprocessed columns
        xcall m_ldcol(colid, win_libchn, "s_select", D_NOPLC)
        if (.not.wndid)
          xcall s_selbld(wndid, 'PRINTERS', list_ix, 15, name)
        xcall w_brdr(wndid, WB_TITLE, "Select a printer", WB_TPOS, WBT_TOP,
      &             WB_TON)
        xcall u_window(D_PLACE, wndid, 5, 5)
        while (.not.sel_printer) do
          begin
            xcall s_select(wndid, 1,, colid, sel_printer)
            case (g_entnam) of
              begincase
                "O_EXIT": exitloop
                "O_CANCEL": exitloop
              endcase
            else
              begin
                xcall u_beep
                nextloop
              end
          end
        if (.not.sel_printer)
          begin
           xcall u_message("No printer was selected, using default printer.")
            sel_printer = device[1]
          end
        if (FALSE)
          begin
    no_winlib,
           xcall u_message("PRT:print was not found, using default printer.")
            sel_printer = device[1]
          end
        if (a_genprt)
          begin
            xcall e_state(D_ON, D_INTR)
            onerror ($ERR_IOFAIL) no_print
            lpque(a_filename, LPNUM:device[sel_printer], DELETE)
            offerror
            xcall e_state(D_OFF, D_INTR)    ;Disallow interrupt
          end

    cleanup,
        xcall e_exit
        xreturn

    get_prtinfo,
        xcall s_parse(buffer, start_pos, 10, item_pos, item_length,
      &               item_type, num_items, end_pos)
        return

    no_print,
        xcall u_message("File not printed.  Filename was: " + a_filename)
        goto cleanup
    endsubroutine
```

Defining the Contents of Your Environment Fields

When report generation begins, ReportWriter calls the routine RW_ENV_METHOD for each environment field to obtain the field's contents. You can register your own version of this routine to provide whatever mechanism you think is necessary to define the contents of your environment fields. See "Creating an Environment Field" on page 8-22 for additional information about environment fields.

RW_ENV_METHOD – Obtain contents of an environment field

```
subroutine RW_ENV_METHOD
a_fld_name    ,a     ;Name of environment field to load (a15)
a_type        ,n     ;Field type (d1)
                     ; 1     Alpha.
                     ; 2     Decimal.
                     ; 3     Implied-decimal.
a_dec_pl      ,n     ;Number of decimal places (if implied-decimal)
                     ; (d2)
a_class       ,n     ;One of the following values (d1)
                     ; 0     Neither date nor time field.
                     ; 1 - 6 Date field.
                     ; 8 - 9 Time field.
                     ; These values correspond to date and time
                     ; storage formats in Appendix B.
a_user_area   ,a     ;Associated user-text string (a40)
a_fld_data    ,a     ;Field area to load (a99)
```

Discussion

The version of RW_ENV_METHOD that is supplied with your original ReportWriter distribution operates as follows:

- By default, RW_ENV_METHOD searches for an environment variable (or logical symbol) at the shell or operating-system level that has the same name as the defined environment field, and the field area is loaded with the contents of that variable, interpreted as alpha. If the variable is not defined, or if it contains data that is invalid for the field type, the field area is cleared.
- If the user text string for the field is set to "MSG," the routine looks instead for a Synergy DBL SEND/RECV message with a message ID that matches the first six characters of the field name. The contents of the message are then loaded into the field area in the same way as an environment variable (as described above). If no such message is found, or if the data is invalid for the field type, the field area is cleared.

Examples

In the sample RW_ENV_METHOD subroutine below, two field names are recognized as coming from a firm master file. If the master file record has not yet been read, the file is opened and the records retrieved. The firm master record, along with the flag to indicate whether or not it has been previously read, are stored in a global data section so that they will be preserved between calls to RW_ENV_METHOD. Thus, the routine doesn’t have to reread the firm master file for each environment field.

The FIRMOFFC environment field will be loaded with a Y or N if it is declared alpha and a 1 or 0 if it is numeric.

If this routine does not recognize the environment field as coming from the firm master file, it looks for a logical symbol with the same name as the field and loads the data into the field according to its type. If the logical symbol is not defined, or if it contains invalid data for the specified type, the field is cleared.

```
subroutine rw_env_method
;
;  Description: Supply values for environment fields.
;
    a_fldnam        ,a     ;Name of field to load
    a_type          ,n     ;Field type (1 = alpha, 2 = decimal,
                           ; 3 = implied-decimal)
    a_dec           ,n     ;# of decimal places, if implied-decimal
    a_cls           ,n     ;Class (0 = normal, 1 = date, 9 = time)
    a_use           ,a     ;Associated user string
    a_field         ,a     ;Returned loaded field

record
    chn             ,d2    ;Firm master file channel
    srch            ,d1    ;Search parameter

global data section read_test, init

record
    master_read     ,d1    ;Has master file record been read yet?

record master              ;Firm management master file record
    firm_name       ,a50   ;Firm name
    firm_office     ,d1    ;Does firm use multiple offices?

endglobal
```

```
proc
    case a_fldnam of
      begincase
        'FIRMNAME':
          begin
            if (.not. master_read)
              call read_master
            a_field = firm_name
          end
        'FIRMOFFC':
          begin
            if (.not. master_read)
              call read_master
            case a_type of
              begincase
                ;1 - Alpha, assume Y/N
                  begin
                    if (firm_office) then
                      a_field = 'Y'
                    else
                      a_field = 'N'
                  end
                ;2 - Decimal, load as 1/0
                  %d(a_field) = firm_office
                ;3 - Implied-decimal, load as 1.0/0.0
                  %f(a_field, a_dec) = firm_office
              endcase
          end
      endcase
    else
      begin                      ;Unrecognized, look for logical
        clear buf
        xcall getlog (a_fldnam, buf)
        onerror (20) clear_it    ;If bad data, clear the field
        case a_type of
          begincase
            ;1 - Alpha
              a_field = buf
            ;2 - Decimal
              %d(a_field) = buf
            ;3 - Implied-decimal
              %f(a_field, a_dec) = buf
          endcase
        offerror
      end
return
```

```
clear_it,                        ;Clear field, based on data type
    case a_type of
      begincase
        ;1 - Alpha
          clear a_field
        ;2 - Decimal
          clear %d(a_field)
        ;3 - Implied-decimal
          clear %f(a_field, a_dec)
      endcase
return

read_master,                     ;Read firm master file
    xcall u_open (chn, 'I:I', 'FIRMMSTR', srch)
    read (chn, master, 1)
    if (.not. srch)
      xcall u_close (chn)
    incr master_read             ;Flag master file as read
    return
endsubroutine
```

Specifying a Century for Two-Digit Years

By default, ReportWriter assumes a year of 19YY for all dates stored in YYMMDD, YYJJJ, and YYPP formats. Using the %RW_CENTURY_METHOD function, you can specify in what century a date stored in any of these formats will fall.

%RW_CENTURY_METHOD – Define default century for two-digit years

```
function RW_CENTURY_METHOD, ^val
    a_date                  ,d      ;Date to provide default century for
    a_class                 ,n      ;ReportWriter class for the date
                                    ; (see the Discussion below)
    a_structure_name        ,a      ;Name of date field's structure
    a_field_name            ,a      ;Name of date field
    a_user_text             ,a      ;Contents of user text field in
                                    ; repository
```

Discussion

The %RW_CENTURY_METHOD function returns the century to be used for a given date. ReportWriter does not validate the return value for the routine and expects it always to be 19 or 20.

A_date is provided in storage format.

Valid field classes for *a_class* are as follows (as defined in **reports.def**):

```
D_RW_YYMMDD   ,1     ; YYMMDD
D_RW_YYJJJ    ,3     ; YYJJJ
D_RW_YYPP     ,5     ; YYPP
```

You can overload the routine using RW_METHOD(M_RW_CENTURY, “RW_CENTURY_METHOD”).

If %RW_CENTURY_METHOD is not defined, the ReportWriter routine RW_INIT checks the value of the SYNCENTURY environment variable, which specifies the year used to “split” two-digit years between centuries. (Years prior to the specified year use one century, while years the same as or later than the specified year use a different century.) See SYNCENTURY on page F-15 for more information.

If the method is not overloaded and SYNCENTURY is not defined or is invalid, ReportWriter uses 19 as the century for all dates stored in YYMMDD, YYJJJ, and YYPP formats.

D6, d5, and **d4** dates will not be optimized if %RW_CENTURY_METHOD is specified or SYNCENTURY is set. YYYYPP will still be optimized.

Examples

The following example is a sample century method that returns the two-digit century for each date passed.

```
function rw_century_method, ^val
    a_date          ,d            ;Date to provide default century for
    a_class         ,n            ;Storage format of date
    a_structure_name,a            ;Structure name for the field
    a_field_name    ,a            ;Field name
    a_user_text     ,a            ;Contents of user text field in
                                  ; repository


stack record
    century         ,i4           ;Century for current field
    year            ,i4           ;Year of date passed in

proc
    century = 19                  ;Default to 19XX
    year = a_date(1:2)

; Change the century for certain dates
    using (a_structure_name) select
    ("CUSMAS "),                  ;Customer master
      using (a_field_name) select
      ("INVCDT "),                ;Invoice date
        if (year .lt. 75)
          century = 20            ;Use 20XX
      (),                         ;Other fields
        if (year .lt. 50)
          century = 20            ;Use 20XX
      endusing
    (),                           ;All other structures
      using (a_class) select
      (D_RW_YYJJJ),
        if (year .lt. 50)
          century = 20            ;Use 20XX
      (),
        if (year .lt. 70)
          century = 20            ;Use 20XX
      endusing
    endusing
    freturn century
endfunction
```

Modifying the Contents of Report Writer Headers and Footers

You can provide support for modification of a report's header or footer each time the report is run. RW_HEADER_METHOD will be called for each line of the report and page header. RW_FOOTER_METHOD will be called for each line of the report and page footer.

We recommend that you use ReportWriter's ability to place data fields in headers/footers instead of using these routines. These routines are maintained only for compatibility with previous versions.

RW_HEADER_METHOD – Modify the report or page header

```
subroutine RW_HEADER_METHOD
    a_source        ,a     ;Source header line (a255)
    a_dest          ,a     ;Returned with destination header line (a255)
    a_width         ,d     ;Report width (d3)
    a_line          ,d     ;Line number (1 - 10) of header to be modified
                           ; (d2)
    a_type          ,d     ;Indicates whether report header or page header
                           ; is being modified (d1)
                           ; 0      Report header.
                           ; Non-0 Page header.
   a_control        ,a     ;ReportWriter control structure
```

Discussion

ReportWriter will call this routine for each line of the report header and page header, up to the last nonblank line. (For example, if the header contains text on lines 1 and 2, this routine will be called twice. If the header contains text on lines 1 and 4, this routine will be called four times.)

Because this routine is called when the report is being generated, the report width has already been determined. If you modify the header so that it is wider than the current report width, it may be truncated.

The version of this routine linked with ReportWriter in your original distribution is a “dummy” routine; it simply copies **source** to **a_dest**.

Examples

Here's a sample RW_HEADER_METHOD subroutine that looks at an operating system environment variable and builds the first line of the page header based on that variable.

```
subroutine rw_header_method
;
; Description:  Modify the page header
;
; Arguments:
;
    a_srchdr        ,a           ;Source header line
    a_dsthdr        ,a           ;Destination header line
    a_rptwid        ,d           ;Current report width
    a_line          ,d           ;Line number (1 thru 10)
    a_type          ,d           ;0=report header;non-0=page header
    a_ctl           ,a           ;Report control structure

record
    cmpnam    ,a25               ;Company name to precede
    len      ,d3

proc
    if (a_type) .and.
      (a_line.eq.1) then         ;The first header line
        begin
          xcall getlog("RPTCMPNAM", cmpnam, len);Get co. name first
            if (len) then
              xcall s_bld(a_dsthdr(1,a_rptwid),,"[%a] %a",cmpnam,
  &                       a_srchdr)
            else                 ;If not set, get department name
              begin
                xcall getlog("RPTDEPNAM", cmpnam, len)
                if (len) then
                  xcall s_bld(a_dsthdr(1,a_rptwid),,"-%a- %a",cmpnam,
  &                           a_srchdr)
                else
                  a_dsthdr = a_srchdr
              end
        end
    else
      a_dsthdr = a_srchdr
    xreturn
endsubroutine
```

RW_FOOTER_METHOD – Modify the report or page footer

```
subroutine RW_FOOTER_METHOD
    a_source  ,a     ;Source footer line (a255)
    a_dest    ,a     ;Returned with destination footer line (a255)
    a_width   ,d     ;Report width (d3)
    a_line    ,d     ;Line number (1 - 10) of footer to be modified
                     ; (d2)
    a_type    ,d     ;Indicates whether report footer or page footer
                     ; is being modified (d1)
                     ; 0      Report footer.
                     ; Non-0 Page footer.
```

Discussion

ReportWriter will call this routine for each line of the report footer and page footer, up to the last nonblank line. (For example, if the footer contains text on lines 1 and 2, this routine will be called twice. If the footer contains text on lines 1 and 4, this routine will be called four times.)

Because this routine is called when the report is being generated, the report width has already been determined. If you modify the footer so that it is wider than the current report width, it may be truncated.

The version of this routine linked with ReportWriter in your original distribution is a “dummy” routine; it simply copies **a_source** to **a_dest**.

Supporting User-Defined Data Type Fields

You can define fields whose data type is user-defined. These fields are treated as alpha fields in most cases, except when ReportWriter displays them. Every time ReportWriter needs to access a user-defined field, it calls one of the functions RW_USAGE_METHOD, RW_GETVAL_METHOD, and RPS_DATA_METHOD, as follows:

- RW_USAGE_METHOD is called when a structure with user-defined fields is accessed by ReportWriter, either when defining new structures/files from which to read or when opening a report with one of these structures. RW_USAGE_METHOD validates usage and determines the internal data type.
- RW_GETVAL_METHOD is called when the field is being read from a data file and determines the actual value ReportWriter will use when sorting, selecting, performing calculations on the field, and so forth.
- RPS_DATA_METHOD is called when ReportWriter needs to display the field. This subroutine formats the data as necessary.

You can register your own versions of these routines to provide the desired mechanisms for displaying and using user-defined data type fields. These routines are ideal for supporting data types that are not supported directly or for supporting alternate date storage formats. (An example of supporting an alternate date storage format appears on page 15-46.)

If you use these routines to access data that is larger than the stored representation, ReportWriter will access the user-defined field only as a field to print.

RPS_DATA_METHOD – Format a user-defined data type field

```
subroutine RPS_DATA_METHOD
a_source      ,a     ;Source field data (a99)
a_dest        ,a     ;Returned with modified field data (a99)
a_use         ,n     ;Indicates ReportWriter's use for the data
                     ; (d1)
                     ; 0   Output (screen, printer, or file).
                     ; 1   Input window.
a_user_text   ,a     ;User text string associated with the field
                     ; (a80)
a_user_data   ,a     ;User data string associated with the field
                     ; (a30)
a_format      ,a     ;(optional) Format string associated with the
                     ; field (a40)
```

Discussion

ReportWriter calls RPS_DATA_METHOD every time a user-defined data type field is displayed. (However, this routine is *not* called when the user-defined data type field is used as a sort field.) RPS_DATA_METHOD converts the storage format created by RW_GETVAL_METHOD into the format you want ReportWriter to display. Note that a format must be provided when you select a user-defined data type to print, and it must have at least the number of characters you wish to display within the format.

If *a_use* contains a value of 0, **a_source** is a field being printed on a detail line or break line. If **a_use** is nonzero, **a_source** is the input in a question field or a selection criteria comparison value. The user text string, user data string, and format string can contain whatever data is required to process the field.

Date and time fields do not automatically use the ReportWriter (or Repository) formats to display dates and times; you must perform this conversion yourself. This display conversion can also be performed using UI Toolkit U_FMTDAT and U_FMTTIM subroutines.

Remember to access implied-decimal fields as such before formatting, using the ^D function:

```
^d(a_source, precision)
```

The version of this subroutine linked with the ReportWriter in your original distribution returns the original data, unmodified, in **a_dest**.

Examples

Here’s a sample RPS_DATA_METHOD subroutine that formats the specified user-defined data type field based on the specified format. Also see the example that begins on page 15-46.

```
subroutine rps_data_method
;
; Description:      This subroutine formats the given user-defined
;                   data type field based on a format string stored
;                   in the field's user text area. Otherwise, it uses
;                   the format string passed.
;
; Arguments:
;
    a_srcdat        ,a     ;Input source data string
    a_dstdat        ,a     ;Output result data string
    a_use           ,a     ;Use for user-defined field
    a_usrtxt        ,a     ;User text string associated with source field
    a_dattyp        ,a     ;Data type string associated with source field
    a_fmtstr        ,a     ;Optional format string

; Note: This example assumes the size of a_dstdat is big enough to load
; the extra format characters. The destination size can be controlled by a
; dummy format string size.
```

```
record
    ix              ,d3
    t_dec           ,d18
    t_date1         ,d8 @t_dec
    t_date2         ,d6 @t_dec

proc
    if (a_use)                   ;Only format for output
      xreturn
    upcase a_usrtxt
    case a_usrtxt of
      begincase
        "USER DATE #1":
          begin
            t_date1 = %d(a_srcdat);Destination source length must be
            case a_usrtxt(16:3) of; enough
              begincase
                "--":
                  xcall s_bld(a_dstdat,,"%4d%a%2a%a%2a",
  &                           t_date1(1:4), a_usrtxt(16:3),
  &                           t_date1(5:2), a_usrtxt(16:3),
  &                           t_date1(7:2))
                "***":
                  xcall s_bld(a_dstdat,, "%a%4d%a%2a%a%2a%a",
  &                           a_usrtxt(16:3), t_date1(1:4),
  &                           a_usrtxt(16:3), t_date1(5:2),
  &                           a_usrtxt(16:3), t_date1(7:2),
  &                           a_usrtxt(16:3))
              endcase
          end
        "USER DATE #2":
          begin
            t_date2 = %d(a_srcdat) ;Destination source length must be
            case a_usrtxt(16:3) of ; enough
              begincase
                "--":
                  xcall s_bld(a_dstdat,,"%4d%a%2a%a%2a",
  &                           t_date2(1:4), a_usrtxt(16:3),
  &                           t_date2(5:2), a_usrtxt(16:3),
  &                           t_date2(7:2))
                "***":
                  xcall s_bld(a_dstdat,, "%a%4d%a%2a%a%2a%a",
  &                           a_usrtxt(16:3), t_date2(1:4),
  &                           a_usrtxt(16:3), t_date2(5:2),
  &                           a_usrtxt(16:3), t_date2(7:2),
  &                           a_usrtxt(16:3))
              endcase
          end
      endcase
    else
      if (%passed(a_fmtstr)) then              ;Can use format string to
        if (%instr(1, a_fmtstr, "&") then      ; control format operation
           begin
             clear ix
             do                  ;For all blanks, replace with "*"
               begin
                 incr ix
                 if (a_srcdat(ix:1).eq." ") then
                   a_dstdat(ix:1) = "*"
                 else
                   a_dstdat(ix:1) = a_srcdat(ix:1)
               end
             until (ix.eq.%len(a_srcdat))
           end
        else
          a_dstdat = a_srcdat
      else
        a_dstdat = a_srcdat
    xreturn
endsubroutine
```

RW_GETVAL_METHOD – Return a user-defined data type value

```
subroutine RW_GETVAL_METHOD
a_source      ,a     ;Source field data
a_dest        ,a     ;Returned with modified field data
a_user_text   ,a     ;User text string associated with field (a80)
a_user_data   ,a     ;User data string associated with field (a30)
a_type        ,n     ;Internal data type for field (d1)
a_size        ,n     ;Internal size of field (d4)
a_precision   ,n     ;Internal precision of field if internal type
                     ; is implied-decimal (d2)
```

Discussion

During report generation, ReportWriter calls this routine once for each record for each user-defined data type field selected in the report. The **a_user_data** and **a_user_text** strings and **a_type**, **a_size**, and **a_precision** are values that were returned from the RW_USAGE_METHOD routine when the user-defined field was selected.

You can use this routine to convert the actual file data for user-defined fields before ReportWriter uses them in a calculation, selection, or conditional or for sorting or totaling. The data value is retrieved as an alpha field with the contents in user storage format. RW_USAGE_METHOD converts this value to an alpha field that contains the Synergy DBL storage format for the type to which you are converting.

Avoid explicit casting (using the ^D or ^A functions or *alpha* = *decimal* format) in creating the field's new type, as you may lose the sign of a decimal field by performing these conversions. Instead, use a decimal field overlaid with an alpha field, as follows, to convert the field to alpha format implicitly.

```
record
    decimal          ,d28
     dec_alpha       ,a28 @decimal
```

ReportWriter does not support changing data size using these routines. In this situation, you should convert to a decimal or alpha field of the same size.

The version of this subroutine linked with ReportWriter in your original distribution returns the original data, unmodified, in **a_dest**.

Note the following restrictions:

- If a user-defined field is a key segment in the primary file, optimizations in selection criteria and sorting use the storage format defined in the file, instead of the value returned from RW_GETVAL_METHOD. If you want to perform selection criteria on a user-defined field that is a key segment in the primary file and use the values as returned from RW_GETVAL_METHOD, you must create a temporary field that has that field as its expression. If a user-defined field is a segment of the primary key of the primary file, the default report is sorted on the storage format in the file of the user-defined field. The same is true if you depend on ReportWriter optimizations for sorting. For this reason, we do *not* recommend using user-defined fields in key segments.

- If two user-defined fields are overlaid, RW_GETVAL_METHOD is called for both fields. If the first field (as defined in the Repository) is reordered by RW_GETVAL_METHOD, the second field will be affected.
- If a user-defined field is used as a segment in a foreign key, any relations on that field use the value as returned from RW_GETVAL_METHOD.

Examples

Refer to the example that begins on page 15-46.

RW_USAGE_METHOD – Determine user-defined data type usage

```
subroutine RW_USAGE_METHOD
a_user_text   ,a     ;User text string associated with field (a80)
a_user_data   ,a     ;User data string associated with the field
                     ; (a30)
a_type        ,n     ;Returned with the field's data type for
                     ; internal use (d1)
                     ; 0   No data type change.
                     ; 1   Alphanumeric.
                     ; 2   Decimal.
                     ; 3   Implied-decimal.
                     ; 4   Integer.
a_size        ,n     ;Returned with field length (d4)
a_precision   ,n     ;Returned with field's precision if internal
                     ; type is implied-decimal (d2)
a_class       ,n     ;Returned with field's class and is one of
                     ; the following values (d1)
                     ; 0   Neither date nor time field.
                     ; 1-6 Date field.
                     ; 8-9 Time field.
                     ;These values correspond to date and time
                     ; storage formats listed in your Repository
                     ; User’s Guide.
```

Discussion

During file processing, ReportWriter calls this routine once for all user type data fields. The **a_user_data** and **a_user_text** strings, **a_size**, **a_precision**, and **a_class** are those that have been defined in the Repository for the selected field.

By default, user-defined data type fields are treated as alpha fields. If you want ReportWriter to use a different data type internally when processing the field, you can modify the distributed version of this routine.

A_type, **a_size**, **a_precision**, and **a_class** tell ReportWriter how to treat the field internally when it is used in a calculation, selection, or conditional or when it is used for sorting or totaling. During report execution, ReportWriter calls RW_GETVAL_METHOD to convert the actual data in the user-defined field each time it is referenced.

If this routine modifies the data type, when the field is displayed in ReportWriter, its data type will be shown as the internal data type, preceded by a “U”. For example, if the field was defined in the repository as a “U4” and this routine specifies its internal type to be decimal with a size of 4, it will be shown as “UD4”.

This routine is always passed a type value of 0. The version of this subroutine linked with the ReportWriter in your original distribution returns the **a_type** value unchanged, meaning that the user-defined field will be treated as alpha. The default version of this subroutine also returns the **a_size**, **a_precision**, and **a_class** unchanged.

Examples

The following example supports a user-defined data type field for dates stored as MMDDYY(YY).

```
; Filename:          usr.dbl
;
; Function:          Defines user-overloadable subroutines for
;                    ReportWriter
;
subroutine rps_data_method
;
; Description: Formats user-defined data type fields
;
; Arguments:
;
    a_source         ,a     ;Source field data (a99)
    a_dest           ,a     ;RETURNED - Modified field data (a99)
    a_use            ,n     ;ReportWriter's use of data (see Notes) (d1)
    a_usrtxt         ,a     ;User text string associated with data (a80)
    a_usrdata        ,a     ;User data string associated with data (a30)
    a_format         ,a     ;Format string associated with field (a40)
;
; Notes:
;      ReportWriter calls this routine every time a user-defined
;      field is displayed. See the Repository User’s Guide for more
;      information on user-defined fields.
;
;      a_use:        When a_use is 0, the item is being output (printer,
;                    screen, or file). When a_use is not 0, the item is
;                    being used to format an input window.
;
.include "WND:tools.def"

record
    year2            ,a2    ;2-digit year
    year4            ,a4    ;4-digit year
    year             ,a4    ;Temporary storage for date (year)
    month            ,a2    ;Temporary storage for date (month)
    day              ,a2    ;Temporary storage for date (day)
    date             ,a10   ;Temporary storage for date (entire)
    r_date           ,a10   ;Date returned from u_fmtdat

proc
    if (a_use)              ;Format only for output
      xreturn
    if ^passed(a_format) then
      begin
        case (a_format) of
```

```
          begincase
            "MM-DD-YY ":
              call format1
            "MM/DD/YYYY ":
              call format2
          endcase
        else
          a_dest = a_source;Default behavior of rps_data_method
      end
    else
      a_dest = a_source
    xreturn

format1,
    date = ^d(a_source(1:6)), "XXXXXX"
    xcall u_fmtdat(3, r_date, ^d(date),,, D_LEFT, 0, '-')
    if (r_date(3:1).ne.'-') then
      a_dest = "0" + r_date
    else
      a_dest = r_date
    return

format2,
    date = ^d(a_source(1:8)), "XXXXXXXX"
    xcall u_fmtdat(6, r_date, ^d(date),,, D_LEFT, 0, '/')
    if (r_date(3:1).ne.'/') then
      a_dest = "0" + r_date
    else
      a_dest = r_date
    return
endsubroutine

subroutine rw_usage_method
;
; Description: Determines user-defined data type usage
;
; Arguments:
;
    a_usrtxt        ,a     ;User text string associated with data (a80)
    a_usrdata       ,a     ;User data string associated with data (a30)
    a_type          ,n     ;RETURNED - field's data type for internal
                           ; usage (d1) (see Notes)
    a_size          ,n     ;RETURNED - field's length (d4)
    a_precision     ,n     ;RETURNED - Precision if internal type is
                           ; implied-decimal (d2)
    a_class         ,n     ;RETURNED  Field class (see Notes) (d1)
;
```

```
;Notes:
;      ReportWriter calls this routine every time a user-defined
;      field is selected in a report. See the Repository User's Guide
;      for more information on user-defined fields.
;
proc
    upcase a_usrdata
    case (a_usrdata) of
      begincase
        "MMDDYY ":
          call usage1
        "MMDDYYYY":
          call usage2
      endcase
    else
      nop                   ;Default behavior of rw_getval_method
    xreturn

usage1,
    a_type = 2              ;Decimal
    a_size = 6
    a_precision = 0
    a_class = 1             ;YYMMDD
    return

usage2,
    a_type = 2              ;Decimal
    a_size = 8
    a_precision = 0
    a_class = 2             ;YYYYMMDD
    return
endsubroutine

subroutine rw_getval_method
;
;Description: Returns user-defined data type value
;
;Arguments:
;
    a_source        ,a      ;Source field data (a99)
    a_dest          ,a      ;RETURNED - Modified field data (a99)
    a_usrtxt        ,a      ;User text string associated with data (a80)
    a_usrdata       ,a      ;User data string associated with data (a30)
    a_type          ,n      ;Internal data type for the field (d1)
    a_size          ,n      ;Internal size of the field (d4)
    a_precision     ,n      ;Internal precision of the field if
                            ; implied-decimal (d2)
;
```

```
;Notes:
;      ReportWriter calls this routine during report generation
;      for each user-defined data type field in each data record.
;
;      This routine converts actual file data for user-defined fields
;      before ReportWriter uses them for calculating, selecting,
;      sorting, totaling, or defining a conditional.
;
;      The values for each parameter are those returned from the
;      RW_USAGE_METHOD routine when the user-defined field was selected.
;
record
    syn_date1       ,d6    ;YYMMDD - Synergy/DE date format #1
     s_year1        ,d2 @syn_date1
     s_month1       ,d2 @syn_date1 + 2
     s_day1         ,d2 @syn_date1 + 4
record
    syn_date2       ,d8    ;YYYYMMDD - Synergy/DE date format #2
     s_year2        ,d4 @syn_date2
     s_month2       ,d2 @syn_date2 + 4
     s_day2         ,d2 @syn_date2 + 6
record
    my_date1        ,d6    ;MMDDYY - User-defined date format #1
     m_month1        ,d2 @my_date1
     m_day1          ,d2 @my_date1 + 2
     m_year1         ,d2 @my_date1 + 4
record
    my_date2        ,d8    ;MMDDYYYY - User-defined date format #2
     m_month2        ,d2 @my_date2
     m_day2          ,d2 @my_date2 + 2
     m_year2         ,d4 @my_date2 + 4
record
    date_out        ,d8    ;Output for final date returned from dyadd
    date_in         ,a10   ;Temporary storage for date transfer

proc
    upcase a_usrdata
    case (a_usrdata) of
      begincase
        "MMDDYY ":
          call getval1
        "MMDDYYYY":
          call getval2
      endcase
    else
      a_dest = a_source    ;Default behavior of RW_GETVAL_METHOD
    xreturn
```

```
getval1,
    date_in = ^d(a_source(1,6)), "XXXXXX"
    my_date1 = ^d(date_in)
    s_year1 = m_year1;Transfer date information
    s_month1 = m_month1
    s_day1 = m_day1
    a_dest = syn_date1, "XXXXXX"
    return

getval2,
    date_in = ^d(a_source(1,8)), "XXXXXXXX"
    my_date2 = ^d(date_in)
    s_year2 = m_year2      ;Transfer date information
    s_month2 = m_month2
    s_day2 = m_day2
    a_dest = syn_date2, "XXXXXXXX"
    return
endsubroutine
```

Getting Information about a Selection Window Item

The RW_LSTINFO_METHOD routine can provide information for the Information menu entry in ReportWriter. This menu entry can be made accessible when the user is viewing a selection list; it is intended to give more information about the highlighted list item.

RW_LSTINFO_METHOD – Get information about highlighted list item

```
subroutine RW_LSTINFO_METHOD
a_content     ,n     ;Determines content of following arguments (d1)
a_file        ,a     ;File definition name (a30)
a_structure   ,a     ;Structure definition name (a30)
a_field       ,a     ;Field or relation definition name, depending on
                     ; content (a30)
```

Discussion

This routine works with the Information menu entry, which can be made available in ReportWriter when the user views a selection list of files, structures, relations, or fields. You can overload the distributed “dummy” routine with one that retrieves more information about the highlighted list item. For example, you might use the Repository Subroutine Library routines (DD_FIELD, DD_FILE, and so forth) to look up long description or template information in your repository.

The Information entry (whose internal name is **usr_fld**) is disabled in the **rptctl.wsc** file that came with your Synergy/DE distribution. If you write your own RW_LSTINFO_METHOD routine, you must edit the script file and remove the **disable** attribute from all **usr_fld** menu entries.

Example

This is the version of RW_LSTINFO_METHOD that’s linked with your ReportWriter distribution.

```
subroutine rw_lstinfo_method
;
;Description: May get called to show more information on a list entry.
;
a_content     ,n     ;Content of following arguments
a_filnam      ,a     ;File definition name
a_strnam      ,a     ;Structure definition name
a_fldnam      ,a     ;Field or relation definition name depending
                     ; on a_content
.include "RPT:rptmsg.def"

proc
    xcall ru_message(MSG_NOHELP)
    xreturn
endsubroutine
```

Modifying ReportWriter Initialization

The RW_INITBLD_METHOD subroutine was designed to be used in conjunction with one or more of the other user-overloadable subroutines. You can use it to initialize data structures that will subsequently be used by other routines, such as RPS_FILNAM_METHOD or RW_ENV_METHOD.

RW_INITBLD_METHOD – Initialize the report building phase

```
subroutine RW_INITBLD_METHOD
a_rep_name    ,a     ;Current report name (a40)
a_files       ,n     ;Number of files used in report (d2)
a_filenames   ,a     ;Array of file definition names (a30)
a_structs     ,a     ;Array of corresponding structure names (a30)
a_quit        ,n     ;Returned with true if report building phase
                     ; and subsequent report generation should
                     ; be aborted (d1)
```

Discussion

The report name, number of files, filenames, and structure names are passed to RW_INITBLD_METHOD so you can do any initialization required for specific reports or for specific files or structures used in a report. For example, if structure A contains aged amounts, RW_INITBLD_METHOD can check to see if that structure is used in the current report, and if so, you can update aging, if necessary. Another example is if an intermediate file is required; use of that file definition in a report can trigger RW_INITBLD_METHOD to build the file.

If the processing time required for your version of this routine is significant, you may want to check for user keyboard input and return the **a_quit** argument set to true if the user wants to interrupt processing. The version of this routine linked with your original ReportWriter distribution is a “dummy” routine; it simply returns.

Example

Here’s a sample RW_INITBLD_METHOD subroutine. Without this routine, RPS_FILNAM_METHOD would only ask for the information on the first call that required it, for as long as you stayed in ReportWriter. This subroutine makes it ask for the information whenever a report is run.

```
subroutine rw_initbld_method
;
;Description: Initialize user-supplied data structures before report
;             generation.
;Arguments:
;
    a_report        ,a           ;Report name
    a_nmfils        ,n           ;Number of files used
    a_filnam        ,a          ;Array of filenames
    a_strnam        ,a          ;Array of structure names
    a_quit          ,n          ;Returned true to abort report gen.

.include "UTS:ddusr.def"        ;(See the sample RPS_FILNAM_METHOD
                                ; routine.)

proc
    usrdd_needinfo = 1          ;This will cause the first call to
                                ; RPS_FILNAM_METHOD, which requires data
                                ; location and client number, to ask
                                ; for that information, while subsequent
                                ; calls will inherit the information.
    xreturn
endsubroutine
```

16

Accessing ReportWriter from Your Application

Accessing ReportWriter from Your Application

You can include ReportWriter as part of your application, either pre-defining reports for your customers to run or enabling them to design their own reports. There are three ways to access ReportWriter from a Synergy application.

- Call one report or the entire ReportWriter application using the subroutines and functions in the ReportWriter external subroutine interface (sometimes referred to as the XCALL interface). This method is the most seamless way to integrate ReportWriter with your application. See “Accessing ReportWriter by External Subroutine Interface” on page 16-3.
- Spawn ReportWriter. See “Spawning and Chaining to ReportWriter” on page 16-24.
- Chain to ReportWriter. See “Spawning and Chaining to ReportWriter” on page 16-24.

Accessing ReportWriter by External Subroutine Interface

The following subroutines and functions, contained in **RPTLIB:synrpt.elb** (**synrpt.exe** on OpenVMS), enable you to call and control ReportWriter directly from your application:

- %RPS_METHOD
- %RW_CLOSE
- RW_ERRINFO
- RW_ERRMSG
- %RW_GENRPT
- RW_GETRPT
- %RW_INIT
- %RW_METHOD
- RW_PARSEMSG
- %RW_REPORTS
- RW_RPTCOPY
- RW_RPTLIST

To call ReportWriter using these routines, follow the guidelines below.

- Include the file **RPTLIB:reports.def** in your calling program using the Synergy DBL .INCLUDE statement.
- Call U_START with one header line defined. If you're using %RW_REPORTS, one footer line must be defined as well; the footer is used to display information during schema loading and generating.
- Call %RW_INIT to initialize ReportWriter; see page 16-13 for details.
- Note for OpenVMS only: If you make calls from the ReportWriter external subroutine interface routines to user-defined routines in a shared image, you must open the shared image for the user-defined routines with OPENELB before calling %RW_INIT. See also OPENELB in the "System-Supplied Subroutines and Functions" chapter of the *Synergy DBL Language Reference Manual*.
- Use any of the other external subroutine interface routines in your program as desired.
- If you're overloading any routines in ReportWriter, see "Overloading a user-overloadable routine in ReportWriter" on page 15-4.
- Link **RPTLIB:synrpt.elb** (and **RPSLIB:ddlib.elb** on UNIX and Windows) with your program or open it using the OPENELB subroutine.

The callable ReportWriter uses the default message library. You can override the default by passing the *text_lib* argument when you call U_START.

If the application calling ReportWriter uses a proportional font as the global font, this proportional font will remain in effect. We recommend that you change the global font to a fixed font with %U_WNDFONT before calling ReportWriter.

Limitations of the external subroutine interface

- Your screen size must be 80 x 24.
- You must be running Synergy DBL and UI Toolkit 6.1 or higher.
- You can call the RW_ routines only from within a UI Toolkit program.
- If the RPTDATE environment variable is set, it will affect the UI Toolkit global variable **g_date_order**, thereby affecting your Toolkit application.

Examples

The following program calls up a session in ReportWriter and enables the user to edit reports, generate reports, and so forth.

```
main
record
.include "WND:tools.def"          ;Include the Toolkit definitions
.include "RPTLIB:reports.def"     ;Include the ReportWriter definitions

proc
    xcall u_start(,1,1)           ;Start Toolkit with 1 header & 1 footer
; Initialize the reports file and the Repository files
    if (.not.%rw_init(,, "CUST1:reports.rpt", "CUST1:rpsmain.ism",
  &     "CUST1:rpstext.ism")) then
      begin
        if (%rw_reports)          ;Run ReportWriter interactively
          xcall rw_errmsg         ;Generate error message
        xcall rw_close            ;Close the ReportWriter files
      end
    else
      xcall rw_errmsg             ;Generate error message
    xcall u_finish
endmain
```

The following program generates one report to a file and another to the printer.

```
main
record
.include "WND:tools.def"          ;Include the Toolkit definitions
.include "RPTLIB:reports.def"     ;Include the ReportWriter definitions

proc
    xcall u_start(,1)             ;Start Toolkit with one header
; Initialize the reports file and the Repository files
    if (%rw_init(,, "CUST1:reports.rpt", "CUST1:rpsmain.ism",
  &     "CUST1:rpstext.ism"))
```

```
      begin
        xcall rw_errmsg
        xcall u_finish
        stop
      end
; Generate the report "FIRST" to the file "HOME:first.ddf"
    if (%rw_genrpt("FIRST", RO_FILE, "HOME:first.ddf"))
      xcall rw_errmsg
; Generate the report "SECOND" to the printer
    if (%rw_genrpt("SECOND", RO_PRINT))
      xcall rw_errmsg
    xcall rw_close         ;Close the ReportWriter files
    xcall u_finish
endmain
```

%RPS_METHOD – Overload Repository methods

WT		U	V

`%RPS_METHOD(`*[event_method_pair, ...]*`)`

Arguments

event_method_pair

(optional) One of the following:

M_LIBRARY*[, mylibrary]*	Set the library from which to load subsequent method routines.
M_RPS_CLOSE, *method*	Close a channel.
M_RPS_OPEN, *method*	Open a channel.
M_RPS_READ, *method*	Read a record.
M_RPS_READS, *method*	Read a record sequentially.
M_RPS_FILNAM, *method*	Map a filename.
M_RPS_DATA, *method*	Format a user-defined data type field.

Return value

%RPS_METHOD returns either 0 (no error occurred) or a handle to error data as defined in **reports.def**.

Discussion

%RPS_METHOD “registers” your overloaded Repository subroutine. You must call %RPS_METHOD from within an RW_INIT_METHOD routine.

%RW_CLOSE – Close ReportWriter and its files

WT		U	V

```
%RW_CLOSE
```

or

```
xcall RW_CLOSE
```

Return value

%RW_CLOSE always returns 0.

Discussion

%RW_CLOSE closes all files opened by ReportWriter.

RW_ERRINFO – Retrieve ReportWriter error information

WT		U	V

```
xcall RW_ERRINFO(rw_error)
```

Arguments

rw_error

The error information as defined in **reports.def**. (**a**)

Discussion

The RW_ERRINFO subroutine enables a developer to examine detailed error information for the most recent ReportWriter error.

RW_ERRMSG – Generate a ReportWriter error message

WT		U	V

```
xcall RW_ERRMSG([err_type])
```

Arguments

err_type

(optional) The type of error to generate: (**n**)

D_ALERT	Use alert processing. (default)
D_ERROR	Use error processing.

Discussion

The RW_ERRMSG subroutine generates the error message for the most recent ReportWriter error using the UI Toolkit U_MESSAGE subroutine. See “Error messages” on page 16-23 for explanations of the messages.

%RW_GENRPT – Generate a report

WT		**U**	**V**

`%RW_GENRPT(`*[report], [out_location], [out_file], [out_format], [page_header][, separator])*

Arguments

report

(optional) The name of the report to generate. (**a**)

out_location

(optional) The output location: (**n**)

RO_SCREEN	Screen (default)
RO_PRINT	Printer
RO_FILE	File

out_file

(optional) The output file, if *out_location* is **RO_FILE**. (**a**)

out_format

(optional) The output format, if *out_location* is **RO_FILE**: (**n**)

RF_REPORT	Report (default)
RF_DATA	Data only
RF_WORKSHEET	Worksheet

page_header

(optional) The page header output flag, if *out_location* is **RO_FILE**: (**n**)

R_NOHEADER	No page header output (default)
R_HEADER	Page header output

separator

(optional) The separator character to use for worksheet output: (**n**)

RS_COMMA	Comma (default)
RS_TAB	Tab
RS_SPACE	Space
RS_SEMICOLON	Semicolon

Return value

%RW_GENRPT returns either 0 (no error occurred) or a handle to error data as defined in the **reports.def** file.

Discussion

%RW_GENRPT enables users to generate a ReportWriter report without chaining or spawning to ReportWriter.

Make sure you call %RW_INIT before calling %RW_GENRPT.

%RW_GENRPT requires a WRUN license. If there is no WRUN license available, it will attempt to use an RPTW license. If neither license type is available, %RW_GENRPT returns an error. %RW_GENRPT releases the license when report generation is complete.

%RW_GENRPT does not perform any message parsing from either a SEND message or an RPTUSR environment variable.

Any EUTILS_METHOD that you have overloaded with E_METHOD will be overridden by the EUTILS_METHOD that ReportWriter overloads in %RW_GENRPT.

RW_GETRPT – Get a sequence of reports

WT		**U**	**V**

`xcall RW_GETRPT(`*channel, key, report_name, status[, report_file]*`)`

Arguments

channel

A channel for the reports file. (**n**)

key

The name of the first report to find. (**a**)

report_name

Returned with the name of the next report found. (**a**)

status

The error status: (**n**)

0	Report name found
1	No more matching reports
n	Synergy DBL error number

report_file

(optional) The name of the reports file to open. (**a**)

Discussion

The RW_GETRPT routine gets a sequence of reports or all reports in a given reports file. This routine is intended to be called iteratively to retrieve report names that match a specified key portion (*key*). If *key* is “ ”, all report names will be returned in sequence (one per call). If *key* is “CUST”, for example, all report names beginning with “CUST” will be selected, and so on.

On the first call, *channel* should not be open or allocated, and *report_name* should be blank. The routine will open the report definition file and return the first matching report name.

Subsequent calls to the routine should pass *report_name* with the last report name found. This will cause the routine to seek the next matching report name in the reports file.

If *report_file* is not specified, the routine will attempt to open the file pointed to by the RPTRFIL environment variable, and if RPTRFIL is not set, the routine will attempt to open the **reports.rpt** file in the directory that RPTDAT points to.

%RW_INIT – Initialize ReportWriter

WT		U	V

```
%RW_INIT([user_routine_elb], [user_init], [def_file], [main_file], [text_file], [name_link_file],
&        [lib_file], [rend_file][, header])
```

Arguments

user_routine_elb

(optional) A logical pointing to an ELB (or shared image on OpenVMS) for user-overloaded routines. (**a**)

user_init

(optional) The name of the routine used to initialize user-overloaded routine names. (**a** or **n**)

def_file

(optional) The name of a report definition file. (**a**)

main_file

(optional) The name of a repository main file. (**a**)

text_file

(optional) The name of a repository text file. (**a**)

name_link_file

(optional) The name of a repository cross-reference file. (**a**)

lib_file

(optional) The name of a ReportWriter window library. (**a**)

rend_file

(optional) The name of a UI Toolkit renditions file. (**a**)

header

(optional) The header for ReportWriter to display instead of “ReportWriter.” (**a**)

Return value

%RW_INIT returns either 0 (no error occurred) or a handle to error data as defined in **reports.def**.

Discussion

%RW_INIT initializes ReportWriter for a series of calls to the RW_ routines. %RW_INIT must be called before any of the other RW_ routines are called. It opens the specified files following the file-open logic specific to that file; see table below.

File	For file-open logic see...
Report definition file (*def_file*)	Page 14-2
Repository main and text files (*main_file* and *text_file*)	"Determining the repository files used" in the "Welcome to Repository" chapter of *Repository User's Guide*
Repository cross-reference file (*name_link_file*)	"Generating a Cross-Reference File" in the "Utility Functions" chapter of *Repository User's Guide*
ReportWriter window library (*lib_file*)	Page 14-3
UI Toolkit renditions file (*rend_file*)	"Renditions" in the "Customizing UI Toolkit" chapter of *UI Toolkit Reference Manual*

If *user_routine_elb* is passed, the default library for all user-overloaded routines is assumed to be *user_routine_elb*. Otherwise, if the logical RWUSRLIB is set, the default library is the file pointed to by RWUSRLIB. In either case, if the library cannot be opened, an error status is returned. If *user_routine_elb* is not passed and the logical RWUSRLIB is not set, no overloading of ReportWriter routines is done.

If *user_init* is passed and alphanumeric, the routine with the name *user_init* will be called to initialize ReportWriter. If *user_init* is passed and numeric, it is assumed to be an address returned from a call to the XADDR subroutine, and the routine at that address will be called. Otherwise, the RW_INIT_METHOD routine will be called. If ReportWriter cannot access *user_init* or the RW_INIT_METHOD routine as defined above, an error status will be returned.

Any routines that are not specifically overloaded will directly call default routines that have the default behaviors as specified in this chapter.

If another %RW_INIT has been done, ReportWriter will call RW_CLOSE before opening files.

%RW_METHOD – Overload ReportWriter methods

WT		U	V

`%RW_METHOD(`*[event_method_pair, ...]*`)`

Arguments

event_method_pair

(optional) One of the following values:

M_LIBRARY*[, mylibrary]*	Set the library from which to load subsequent method routines.
M_RW_PRTOPEN, *method*	Open a file/printer for printing.
M_RW_PRTCLOSE, *method*	Close the output file/printer.
M_RW_ENV, *method*	Obtain contents of environment field.
M_RW_GETVAL, *method*	Return user-defined data type value.
M_RW_USAGE, *method*	Determine user-defined data type usage.
M_RW_LSTINFO, *method*	Get information about highlighted list item.
M_RW_INITBLD, *method*	Initialize the report building phase.

Return value

%RW_METHOD returns either 0 (no error occurred) or a handle to error data as defined in **reports.def**.

Discussion

%RW_METHOD “registers” your overloaded subroutine so that ReportWriter will recognize it. You must call RW_METHOD from within an RW_INIT_METHOD routine.

RW_PARSEMSG – Parse message to send to the XCALL interface

WT		**U**	**V**

```
xcall RW_PARSEMSG(message, [def_file], [chain_file], [main_file], [text_file], [msg_file],
&                 [lib_file], [rend_file], [report], [output_location], [output_file],
&                 [output_format], [page_header], [separator][, link_file])
```

Arguments

message

Returned with the message you want to map to the new XCALL argument structure. (**a**)

def_file

(optional) Returned with the name of the report definition file. (**a**)

chain_file

(optional) Returned with the name of the program that ReportWriter should chain to upon termination. (**a**)

main_file

(optional) Returned with the name of the repository main file to use. (**a**)

text_file

(optional) Returned with the name of the repository text file to use. (**a**)

msg_file

(optional) Returned with the name of the text message file to use. (**a**)

lib_file

(optional) Returned with the name of the ReportWriter window library to use. (**a**)

rend_file

(optional) Returned with the name of the renditions file to use. (**a**)

report

(optional) Returned with the name of an existing report to be output. (**a**)

output_location

(optional) Returned with the output location as follows: (**n**)

0 Output goes to the screen. (default)

1 Output goes to the printer.

2 Output goes to a file.

output_file

(optional) Returned with the name of the output file to generate if you specified an *output_location* value of 2. (**a**)

output_format

(optional) If you specified an *output_location* value of 2, returned with the output format as follows: (**n**)

0 Output is generated in report form (including headers, footers, and the like). (default)

1 Output consists of detail record data only. (See "Generating a Report File" on page 4-5 for more information on the available output formats.)

2 ReportWriter generates a report in spreadsheet format.

page_header

(optional) Returned with a flag to indicate whether a spreadsheet has a page header generated as the first record of the report. Use only if *output_format* is 2. (**n**)

0 No page header will be generated. (default)

1 Page header will be generated.

separator

(optional) Returned with a flag that indicates the separator used for spreadsheet output. Use only if *output_format* is 2. Valid values are as follows: (**n**)

1 Comma (default)

2 Tab

3 Space

4 Semicolon

link_file

(optional) Returned with the name of the name link file to use. (**a**)

Discussion

RW_PARSEMSG provides a migration path from spawning or chaining to ReportWriter's external subroutine interface (XCALL interface). It takes the message you've passed to ReportWriter or the value of the RPTUSR environment variable and translates it into a set of parameters to send to the interface.

The external subroutine interface does not support the *chain_file* and *msg_file* arguments. To chain to another file when ReportWriter terminates, you must STOP to the program name that you pass to RW_PARSEMSG. To specify a text message file for ReportWriter to use, you must pass the filename to U_START as the seventh argument.

You are not required to call U_START or %RW_INIT before calling RW_PARSEMSG.

%RW_REPORTS – Create an interactive ReportWriter session

WT		U	V

```
%RW_REPORTS
```

Return value

%RW_REPORTS returns either 0 (no error occurred) or a handle to error data as defined in the **reports.def** file.

Discussion

%RW_REPORTS starts an interactive ReportWriter session.

Make sure you call %RW_INIT before calling %RW_REPORTS.

Call U_START with one header line and one footer line defined.

%RW_REPORTS requires an RPTW license. If no license is available, it returns an error. %RW_REPORTS releases the license when the user closes the ReportWriter session.

%RW_REPORTS does not perform any message parsing from either a SEND message or the RPTUSR environment variable.

Any EUTILS_METHOD that you have overloaded with E_METHOD will be overridden by the EUTILS_METHOD that ReportWriter overloads in %RW_REPORTS.

RW_RPTCOPY – Copy a report definition

WT		U	V

`xcall RW_RPTCOPY(`*source_file, source_name, [dest_file], [dest_name][, sts]*`)`

Arguments

source_file

The name of the source report definition file. (**a64**)

source_name

The name of the report from the source file to copy. (**a40**)

dest_file

(optional) The name of the destination report definition file. (**a64**)

dest_name

(optional) The name of the report to store in *dest_file*. (**a40**)

sts

(optional) The error status: (**d1**)

0	Success
1	Failure

Discussion

This subroutine copies the specified report from the source report definition file to the destination report definition file. (They can be the same file, as long as the report names are unique.) If you don’t specify the destination definition filename, **RPTDAT:reports.rpt**, is used. If you don’t specify the destination report name, the source report name is used. If a report already exists with that name, an error is returned.

This subroutine uses approximately 4K of global memory.

Examples

The following call copies the report A/R_MASTER from the file **reports.rpt** and stores it in the file **myrpts.rpt** with the name MY_A/R.

```
xcall rw_rptcopy("reports.rpt", "A/R_MASTER", "myrpts.rpt", "MY_A/R")
```

This call copies the report PAYROLL from the file **reports.rpt** and stores it in the file **myrpts.rpt**, keeping the same name.

```
xcall rw_rptcopy("reports.rpt", "PAYROLL", "myrpts.rpt")
```

RW_RPTLIST – Retrieve a list of reports

WT		U	V

`xcall RW_RPTLIST(`*channel, array, [start], [end], names_req, [#names], [filename][, sts]*`)`

Arguments

channel

Returned with the channel number of the open report definition file. (**d2**)

array

Returned with the array of report names. (**a40**)

start

(optional) Contains the name at which to start. (**a40**)

end

(optional) Contains the name at which to end. (**a40**)

names_req

The number of report names requested. (**d4**)

#names

(optional) Returned with the number of report names. (**d4**)

filename

(optional) The name of the report definition file to open. (**a64**)

sts

(optional) The error status: (**d1**)

0	Success
1	Failure

Discussion

RW_RPTLIST returns a list of report names from the specified report definition file. If no report definition file is specified, **RPTDAT:reports.rpt**, is used. RW_RPTLIST returns as many report names as are found or as are requested, whichever occurs first. The actual number of names in the array can be returned in *#names*.

Channel must be 0 on the first call to this routine, so the report definition file is opened. The report definition file is closed when either the end of file or an error condition occurs, or when *names_req* is 0.

You can specify a wildcard character (*) in your start value to return only the reports that start with a specific string (for example, only the reports that start with “ACCT”).

Array must be an arrayed variable.

Examples

Let’s assume the following data division.

```
record
    names     ,20a40         ;Array for report names
    ct        ,d2            ;Return count
    sts       ,d1            ;Error status returned
```

This call would get the first 10 report names.

```
xcall rw_rptlist(chn, names,,, 10, ct,, sts)
```

This call would get the next 10 report names.

```
xcall rw_rptlist(chn, names, names[10],, 11, ct,, sts)
```

This call would get all reports (maximum of five) that start with “ACCT” from the report definition file **myrpt.rpt**.

```
xcall rw_rptlist(chn, names, "ACCT*",, 5, ct, "myrpt.rpt", sts)
```

Error messages

The table below lists error messages that might be generated if you're calling ReportWriter using the external subroutine interface or if you've overloaded user-overloadable subroutines.

Message	Cause	What to do about it
File "%a" not found.	ReportWriter could not find the specified file.	Check the filename and environment variables.
Invalid parameter in %a.	You specified an invalid argument in the specified routine.	Check your call to the routine.
Invalid parameter to RW_INIT: "%a".	The value stated is invalid for the argument to RW_INIT.	Change your call to RW_INIT.
Protection violation in access of "%a".	The specified file cannot be accessed by ReportWriter.	Check the file privileges and directory.
ReportWriter methods cannot be overloaded outside of RW_INIT.	You tried to call %RW_METHOD or %RPS_METHOD outside of RW_INIT_METHOD.	Move the method calls to RW_INIT_METHOD.
ReportWriter not initialized due to error in RW_INIT.	You passed back a false status flag in your RW_INIT_METHOD subroutine.	Check your RW_INIT_METHOD to see why it is returning false.
ReportWriter utilities cannot be called without calling %RW_INIT.	You tried to call RW_REPORTS or RW_GENRPT before RW_INIT was called.	Modify your code to call RW_INIT before making these other calls.
Routine "%a" not found.	ReportWriter could not find the named routine in RWUSRLIB.	Check the ELB or shared image that RWUSRLIB is set to.

Spawning and Chaining to ReportWriter

When you use the SPAWN subroutine, when ReportWriter terminates, you are returned to the point in your application at which you called ReportWriter.

You can use an argument string to specify which report definition and repository files to use, as well as to cause ReportWriter to chain to a program of your choice. If you just want to output an existing report (without going into the ReportWriter application), specify the report name in the argument string. You can also specify whether that report is to be output to the screen, a printer, or a file.

When you spawn ReportWriter, the argument string is assigned to the environment variable RPTUSR. ReportWriter looks at this environment variable to determine which options have been specified.

When you chain to ReportWriter, the argument string can either be assigned to the environment variable RPTUSR or passed in the Synergy DBL SEND statement. The ID in the SEND statement is "RPTUSR." ReportWriter first looks for a message from the SEND statement. If it doesn't find one, it checks the RPTUSR environment variable.

The argument string can have up to 200 characters.

Syntax

"*[def_file]*|*[chain_file]*|*[main_file]*|*[text_file]*|*[msg_file]*|*[lib_file]*|*[rend_file]*|*[report]*
& |*[output_location]*|*[output_file]*|*[output_format]*|*[page_header]*|*[separator]*|*[link_file]*"

Arguments

def_file

(optional) The name of the report definition file. (**a**)

chain_file

(optional) The name of the program to which ReportWriter is to chain upon termination. (**a**)

main_file

(optional) The name of the repository main file to use. (**a**)

text_file

(optional) The name of the repository text file to use. (**a**)

msg_file

(optional) The name of the text message file to use. (**a**)

lib_file

(optional) The name of the ReportWriter window library to use. (**a**) *rend_file*

(optional) The name of the renditions file to use. (**a**)

report

(optional) The name of an existing report to be output. (**a**)

output_location

(optional) Determines the output location as follows: (**n**)

0 Output goes to the screen. (default)

1 Output goes to the printer.

2 Output goes to a file.

output_file

(optional) The name of the output file to generate. Specify this argument only if *output_location* is 2. (**a**)

output_format

(optional) If you specified an output location value of 2, determines the output format as follows: (**n**)

0 Output is generated in report form (including headers, footers, and the like). (default)

1 Output consists of detail record data only. (See "Generating a Report File" on page 4-5 for more information on the two available output formats.)

2 Output consists of detail record data only and is output in a spreadsheet format. ReportWriter looks for the *page_header* and *separator* arguments.

page_header

(optional) Determines if a spreadsheet has a page header generated as the first record of the report. Specify this argument only if *output_format* is 2. (**n**)

0 No page header will be generated. (default)

1 Page header will be generated.

separator

(optional) Determines the separator used for spreadsheet output. Specify this argument only if *output_format* is 2. The separator character defaults to a comma if *separator* is not specified. Valid values are as follows: (**n**)

1 Comma

2 Tab

3 Space

4 Semicolon

link_file

(optional) The name of the name link file to use. (**a**)

Discussion

Keep the following points in mind:

- All arguments must be separated by a vertical bar (|) or a backslash (\). The vertical bar or backslash must also be used as a placeholder for omitted arguments. (See Examples on page 16-27.) If you have a vertical bar anywhere in your command string, it will be used as the argument delimiter. Do *not* mix vertical bars and backslashes in the same SEND statement.

  Note that on Windows, the backslash character is used as a directory separation character. If your filenames contain this character, you must use a vertical bar as an argument delimiter.
- All filenames can include a logical (for example, **RPT:rptctl**).
- The maximum length of each filename is 64.
- If you don't specify the *def_file* argument, ReportWriter searches for the report definition file as described in "Creating Report Definition Files" on page 14-2.
- If you don't specify the *chain_file* argument, ReportWriter won't chain to another program upon termination.
- If you specify the *chain_file* argument, ReportWriter displays a blinking "Working..." message when chaining to the other program. If you want to change the text of this message or clear it entirely, you can modify the text message file. For instructions see "The Synergy UI Toolkit Control Panel" in the "General Utilities" chapter of *Synergy Tools*.
- If you don't specify *main_file* or *text_file*, ReportWriter searches for the repository files as described in "Determining the repository files used" in the "Welcome to Repository" chapter of the *Repository User's Guide*.
- If you specify a different report definition name or Repository filenames using either the RPTUSR environment variable or the SEND/RECV message (as described in "Spawning ReportWriter" and "Chaining to ReportWriter" below), those filenames have precedence over the aforementioned default logic.
- If you don't specify *msg_file* or *lib_file*, ReportWriter attempts to open the message file as **SYNTXT:syntxt.ism** and the window library file as **RPTDAT:rptctl.ism**. If it cannot open a given file, ReportWriter attempts to open that file in the current directory.
- The report name is optional. If you pass a report name, ReportWriter generates the report and then either chains to the specified program or stops. If you pass a report name that doesn't exist, an error message is displayed, and then ReportWriter either chains to the specified program or stops.
- The output location and related arguments (output file and output format) are optional. If you specify an *output_location* value of 2 but do not specify an *output_file* argument, an input window prompts you for the output filename and output format. (See "Generating a Report File" on page 4-5 for a description of the input window.)

- If you are chaining to ReportWriter from a non-Synergy application, you must clear the screen before you chain.
- If you are chaining to ReportWriter on a multi-user system, we recommend that you pass the terminal number as the third argument in the Synergy DBL SEND statement. This prevents the possibility of the message being received by the wrong ReportWriter process.

Examples

Here are some examples of valid argument strings:

```
"RPT:reports.rpt|myfile.dbr"
```

```
"|myfile.dbr||||RPT:rptctl"
```

```
"RPT:reports.rpt\myfile.dbr\\\\\\myrpt\1"
```

Spawning ReportWriter

There are two methods for spawning ReportWriter.

- This method enables you to change your argument string dynamically and specify any of the optional arguments for ReportWriter within your application. Note that *argument_string* is explained beginning on page 16-24.

  Set the RPTUSR environment variable as follows:

  ```
  xcall setlog("RPTUSR", argument_string, sts)
  if (sts)
    xcall spawn("dbr RPT:rpt")
  ```

- This method is useful when you want to change your RPT environment temporarily, without setting and resetting RPT environment variables:

  1. Set the RPTUSR environment variable to your argument string before you run your application.
  2. Add the following line to your application:

     ```
     xcall spawn ("dbr RPT:rpt")
     ```

Chaining to ReportWriter

There are two methods for chaining to ReportWriter. Note that *argument_string* is explained beginning on page 16-24.

- To use the Synergy DBL SEND statement, modify your code as follows:

  ```
  send(argument_string, "RPTUSR")
  stop "RPT:rpt"
  ```

  In the SEND statement, “RPTUSR” is the message ID of ReportWriter.

- To use the RPTUSR environment variable, set it as follows:

  ```
  xcall setlog("RPTUSR", argument_string, sts)
  if (sts)
    stop "RPT:rpt"
  ```

17

Report Definition Language

Introduction to the Report Definition Language

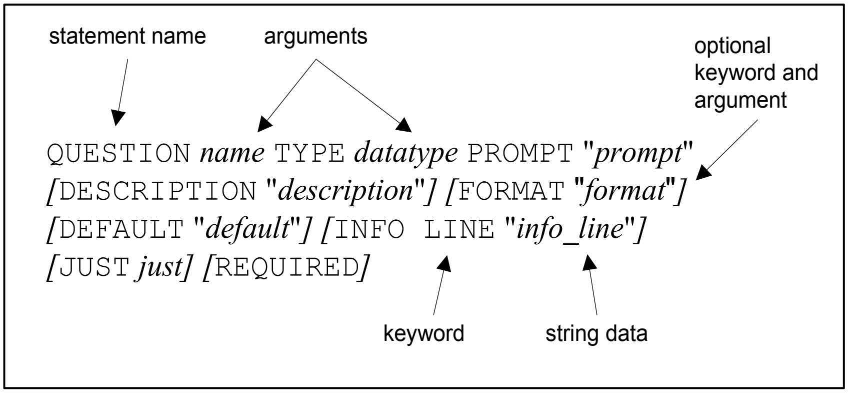
The Report Definition Language (RDL) describes the contents of a ReportWriter report definition. It was designed as a tool for documenting, analyzing, creating, and modifying reports.

- The ReportWriter utilities Generate Report Schema and Load Report Schema generate and interpret the Report Definition Language. Generate Report Schema converts the contents of the specified report into the Report Definition Language. Load Report Schema converts the contents of a Report Definition Language file into a new report.
- Report Definition Language files can be used to move reports from one system to another or to copy reports from one report definition file into another.
- Another use for the Report Definition Language is to create or modify reports. You can modify the Report Definition Language file created by the Generate Report Schema utility or create one of your own, and then reload it with the Load Report Schema utility.
- Report Definition Language output files are also very useful in analyzing reports or in creating a new report similar to an existing report.

Using Report Definition Language Statements

This section contains detailed descriptions of Report Definition Language statements. For each statement, you'll find syntax summaries and statement descriptions. We used the following conventions in documenting the syntax of the Report Definition Language statements:

- Statement names are shown in UPPERCASE.
- Arguments that you should replace with actual data are shown in *lowercase italics*.
- Optional keywords and their associated data are enclosed in *[*italic square brackets*]*.
- Keywords and keyword phrases are shown in UPPERCASE.
- Keyword string data is enclosed "in quotation marks."
- Arguments that may be repeated more than once are followed by an ellipsis (…).



```
QUESTION name TYPE datatype PROMPT "prompt"
[DESCRIPTION "description"] [FORMAT "format"]
[DEFAULT "default"] [INFO LINE "info_line"]
[JUST just] [REQUIRED]
```

Figure 17-1. Report Definition Language syntax

General usage rules

- You can specify statement names, keywords, and arguments in either uppercase or lowercase, except for quoted definition names, which must be uppercase. (ReportWriter converts all nonquoted data to uppercase on input.)
- When an RDL file is "loading," if required data has not been supplied, or if the data is invalid, an error message is logged and the entire report is ignored.
- You can specify keywords in any order. In this manual, we've listed keywords in the order in which they are generated by the Generate Report Schema utility.

- A keyword that is longer than a single physical word cannot span multiple lines. For example, the following is incorrect, because the INFO LINE keyword is split between two lines.

  ```
  QUESTION CITY TYPE ALPHANUMERIC SIZE 30
  PROMPT "City:" DEFAULT "*" INFO
  LINE "Enter name of city."
  ```

- Keyword file definition names must include the structure name, using the format *file.structure*.
- Keyword field definition names must be unique within a report. You may have to include the file and structure name to make the field name unique (for example, *file.structure.field*).
- Do not specify a negative value for a numeric keyword argument, except where noted. A negative value must be immediately preceded by a minus sign (for example, -3 or -10).
- Keyword string data must be enclosed in a set of matching double or single quotation marks (“ ” or ‘ ’) and cannot span multiple lines.
- String data can contain a quotation character if that character is different from the character that encloses the data. For example, the following string is valid:

  “Press ‘Enter’ to continue”

- Data that exceeds the maximum size allowed is truncated.

Recommended usage

To ensure that your Report Definition Language statements are loaded properly, we recommend you define them in the following order. The statements you’d use are in parentheses.

1. Define the report name and all files. (REPORT, FILE)
2. Define all temporary fields in the order in which you want them evaluated. (CALCULATION, ENVIRONMENT, QUESTION, SUBTOTAL, TEMPTEXT)
3. Define all selections and sorts. (SELECT, SORT)
4. Define each line in the order you want it to appear in the report, along with all of its fields and text. (FIELD, LINE, TEXT)
5. Define any miscellaneous report options. (MISCELLANEOUS)

Statement Syntax

Specification definitions

field_spec

The *field_spec* is a specification for a field in a report. A field specification may optionally contain subscript or range information.

- A simple *field_spec* has the format

  field [`FID` *fid]*

- A *field_spec* with optional subscripting has the format

  field[`[`*subs_value[,subs_value ...]*`]`*] [*`FID` *fid]*

- A *field_spec* with optional ranging has the format

  field[`(`*offset:length*`)`*] [*`FID` *fid]*

Arguments

field

A unique field name. If field alone does not identify a unique field, *file.field* or *file.structure.field* must be specified. Field can be a maximum of 30 characters and cannot contain spaces.

FID

(optional) Indicates that the owner file ID will follow. Specify this keyword if a file and structure combination has been selected more than once.

fid

The owner file identification number.

subs_value

(optional) Either a numeric literal or a nonranged, nonsubscripted field name to be used as a subscript value for an arrayed field. (Notice that if one or more *subs_value* arguments are present, they must be enclosed in square brackets. The arguments must be separated by a comma and can contain no embedded spaces.)

Because an arrayed field can have up to four dimensions, the field can have up to four subscript values.

offset

(optional) Either a numeric literal or a nonranged, nonsubscripted field name to be used as the base one offset position to range “from.”

length

(optional) Either a numeric literal or a nonranged, nonsubscripted field name to use as the "length" of the range.

Examples

```
ARCUST
ARRATE[3]
ARCUST(1:15)

ARTRANS.ARTRANS.ARCUST FID 1
```

conditional_spec

The *conditional_spec* is a specification for a condition or set of conditions in a report. It is used after the IF or the SELECT keyword.

field_spec compare value [`TRUEIFBLANK`*]*
[connect field_spec compare value [`TRUEIFBLANK`*]] […]*

Arguments

field_spec

A specification for a field. (See *field_spec* above.)

compare

The operator used to compare *field_spec* with *value*. Valid values are:

EQ	Equal to
NE	Not equal to
LT	Less than
LE	Less than or equal to
GT	Greater than
GE	Greater than or equal to

value

The *value* is either a literal or another field specification. If *value* is a literal value, it should be enclosed in quotation marks. (See the examples for the CALCULATION statement on page 17-8 and the LINE statement on page 17-8.)

TRUEIFBLANK

(optional) Indicates that the conditional evaluates to TRUE if the *field_spec* operand is blank or zero.

connect

(optional) The connector used if you're specifying two or more conditions. If you specify *connect*, you must also specify *field_spec*, *compare*, and *value*. Valid values are:

AND
OR

Up to five conditions can be connected together for all *conditional_specs*, except those used in selections, which can have up to 25.

Examples

```
ARRATE[3] EQ "400"
```

```
TEMP.QDATE GE TRANS.START_DATE
    AND TEMP.QDATE LE TRANS.END_DATE
```

```
SALES.REP EQ TEMP.QREF TRUEIFBLANK
```

CALCULATION – Describe a calculation field

CALCULATION *name* TYPE *type* SIZE *size* [PRECISION *dec_places*]
[IF *conditional_spec*] EXPRESSION "*expression*" [ALTERNATE "*alt_expression*"]
[DESCRIPTION "*description*"] [JUST *just*] [FORMAT "*format*"] [AFTER_SORT]

Arguments

name

The name of the temporary calculation field. It can have up to 15 characters and cannot contain spaces.

TYPE

Indicates that the data type will follow.

type

The data type of the calculation's result. Valid values are:

ALPHANUMERIC
NUMERIC
DATE
TIME

SIZE

Indicates that the length of the calculation's result will follow.

size

The length of the calculation's result. When using date or time fields in your calculation, the length should match the storage length of the base value in the expression.

PRECISION

(optional) Indicates that the number of decimal places in implied-decimal fields will follow.

dec_places

The number of characters to the right of the decimal in the implied-decimal calculation field. If the data type is implied-decimal, this attribute must be present, and it must be less than or equal to the size of the field. Otherwise, the field is ignored.

IF

(optional) Indicates that a conditional specification will follow.

conditional_spec

Specifies a condition or set of conditions in a report. It can have up to five conditions connected together. See "conditional_spec" on page 17-6 for more information.

EXPRESSION

(optional) Indicates that the calculation expression will follow.

expression

The calculation's expression. It can have up to 150 characters. See "Entering an expression" on page 8-10 for the syntax rules of expressions. The expression must be enclosed in quotation marks.

ALTERNATE

(optional) Indicates that the alternate calculation expression will follow.

alt_expression

The calculation's alternate expression. It can have up to 150 characters. See "Entering an expression" on page 8-10 for the syntax rules of expressions. The expression must be enclosed in quotation marks.

DESCRIPTION

(optional) Indicates that the description of the calculation field will follow.

description

A description of the calculation field. It can have up to 40 characters.

JUST

(optional) Indicates that a data justification argument will follow.

just

The justification of the calculation field. Valid values are:

LEFT
CENTER
RIGHT

The default is LEFT for alpha, date, time, and user type fields, and RIGHT for decimal and implied-decimal fields. CENTER alignment is allowed only for alpha and user fields.

FORMAT

(optional) Indicates that the format for this calculation field will follow.

format

The display format for the calculation field. It can have up to 40 characters.

AFTER_SORT

(optional) Indicates that the calculation field will be evaluated after the sort. If the calculation field is dependent on one or more subtotal access fields, it will automatically be evaluated after the sort.

Discussion

The CALCULATION statement describes a temporary calculation field to be created in a report.

A calculation field defines a mathematical expression. Calculation fields are temporary fields that are not found in the repository. To create this type of field, you must specify the field name, data type, data length, and calculation expression. You can also define an alternate expression based on a conditional.

The order in which you specify your calculation fields determines the order in which ReportWriter evaluates them. The maximum number of calculation fields that can be defined in a report is 99.

If your temporary calculation field does not have a unique name, it must be preceded by "TEMP.". For example, if your temporary field is called AVERAGE and one of the selected files also contains a field called AVERAGE, you would specify the temporary field like this:

```
TEMP.AVERAGE
```

Examples

- In the example below, the expression

  ```
  "ORDER.SHP_DATE - ORDER.ORD_DATE"
  ```

  is always calculated.

  ```
  CALCULATION TURNTIME TYPE NUMERIC SIZE 8
  EXPRESSION "ORDER.SHP_DATE - ORDER.ORD_DATE"
  ```

- In the following example, the expression is only calculated if the conditional is true. If the conditional is false, the numeric value 0 is assigned to the field. An alternate expression could have been defined for the false conditional.

  ```
  CALCULATION TURNTIME TYPE NUMERIC SIZE 8
  IF CUSTOMER.CUST_ID LT "100" AND
     CUSTOMER.CUST_ID GT "50"
  EXPRESSION "ORDER.SHP_DATE - ORDER.ORD_DATE"
  ```

ENVIRONMENT – Describe an environment field

```
ENVIRONMENT name TYPE type SIZE size [DESCRIPTION "description"]
[FORMAT "format"] [USERTEXT "user_text"] [PRECISION dec_places]
[STORED store_format] [JUST just]
```

Arguments

name

The name of the temporary environment field to be created. It can have up to 15 characters and cannot contain spaces.

TYPE

Indicates that the data type will follow.

type

The data type of the environment field. Valid values are:

ALPHANUMERIC
NUMERIC
DATE
TIME

SIZE

Indicates that the length of the environment field will follow.

size

The length of the environment field.

DESCRIPTION

(optional) Indicates that the description of the environment field will follow.

description

A description of the environment field. It can have up to 40 characters.

FORMAT

(optional) Indicates that the format for the environment field will follow.

format

The display format for the environment field. It can have up to 40 characters.

USERTEXT

(optional) Indicates that a user-defined text string will follow.

user_text

A user-defined text string associated with the environment field. It can have up to 40 characters.

PRECISION

(optional) Indicates that the number of decimal places in implied-decimal fields will follow.

dec_places

The number of characters to the right of the decimal point in an implied-decimal environment field. If the data type is implied-decimal, this attribute must be present, and it must be less than or equal to the size of the field. Otherwise, the field is ignored.

STORED

(optional) Indicates that the date storage format will follow.

store_format

Specifies the date storage format. Valid values are:

YYPP
YYJJJ
YYMMDD (default)
YYYYPP
YYYYJJJ
YYYYMMDD

JUST

(optional) Indicates that a data justification argument will follow.

just

The justification of the environment field. Valid values are:

LEFT
CENTER
RIGHT

The default is LEFT for alpha, date, time, and user type fields, and RIGHT for decimal and implied-decimal fields. CENTER alignment is not allowed for numeric, date, and time fields. RIGHT alignment is not allowed for alpha fields.

Discussion

The ENVIRONMENT statement describes a temporary environment field to be created in a report.

Environment fields are used to obtain data from the system environment. They are temporary fields that are not found in the repository. You must specify the field's name, data type, and size. The maximum number of environment fields that can be defined in a report is 99.

If your temporary environment field does not have a unique name, it must be preceded by "TEMP.". For example, if your temporary field is called SYSTEM and one of the selected files also contains a field called SYSTEM, you would specify the temporary field like this:

```
TEMP.SYSTEM
```

Examples

- In the example below, ENVIRON1 is the name of the field, and its size is 30. It is an alphanumeric field.

  ```
  ENVIRONMENT ENVIRON1 TYPE ALPHANUMERIC SIZE 30
  ```

- In the following example, ENVIRON2 is the name of the field and its size is 6. It is a date field with the storage format YYMMDD. The format in which the field will be displayed is “MM/DD/YY.” The associated user text string is “Start Date.”

  ```
  ENVIRONMENT ENVIRON2 TYPE DATE SIZE 6 FORMAT "MM/DD/YY"
  USERTEXT "Start Date" STORED YYMMDD
  ```

FIELD – Describe a field to be printed

```
FIELD field_spec [COLUMN column] [HEADER "header"] [JUST just]
[FORMAT "format"] [FMTTRUNC truncate] [STRIP] [STRIPCHR "strip_char"]
[TOTAL] [IF conditional_spec]
```

Arguments

field_spec

A specification for a field to be printed on the current line. See "field_spec" on page 17-5 for more information.

COLUMN

(optional) Indicates that the column position for this field will follow. If this keyword is not specified, the default column position is two spaces after the previous field.

column

The column position for this field. If this argument is specified, it must be the first argument after the *field_spec* argument. The value range is between 1 and (255 - *field size*).

HEADER

(optional) Indicates that the text for the field's header will follow. The default header is the field's header in the repository, or else the field's description as defined in the repository.

header

The text for the field's header. It can have up to 40 characters.

JUST

(optional) Indicates that a data justification argument will follow.

just

The justification of the field. Valid values are:

LEFT
CENTER
RIGHT

The default is LEFT for alpha fields and RIGHT for decimal and implied-decimal fields. CENTER alignment is not allowed for numeric fields; neither RIGHT nor CENTER alignment is allowed for text fields (multidimensional alpha fields).

FORMAT

(optional) Indicates that the format for this field will follow.

format

The display format for the field. It can have up to 40 characters.

FMTTRUNC

(optional) Indicates that the format truncation argument will follow.

truncate

Specifies how the format will be truncated if the format size is longer than the data size. Valid values are:

LEFT
RIGHT

STRIP

(optional) Strips trailing blanks from the field, bringing the next field closer. This argument uses the space character as the strip character.

STRIPCHR

(optional) Indicates the strip character will follow.

strip_char

The strip character that follows the blank-stripped field.

TOTAL

(optional) Totals this field at all sort levels and at the end of the report.

IF

(optional) Indicates that a conditional specification will follow.

conditional_spec

Specifies a condition or set of conditions that controls whether this field is printed in a report. See "conditional_spec" on page 17-6 for more information.

Discussion

The FIELD statement describes a field to be printed in a report.

A field can be printed on all line types in a report. The default starting position is three characters to the right of the previous field's last character. The maximum number of columns in the report is 255.

Examples

```
FIELD CUST_TAG.CUST_NAME COLUMN 1

FIELD SALES COLUMN 10 TOTAL

FIELD TEMP2 COLUMN 20 IF TURNTIME GT "30"
```

FILE – Describe a file to be read

```
FILE file.structure [ID id] [PARENT pid] [SEPARATE]
[RELATION name] | [[LINK_FIELD link_field] [LINK_NAME link_name]
[TO_KEY to_key] [LINK_SEGMENT segment ...]]
```

Arguments

file.structure

The name of a file and associated structure to be read from in a report.

ID

(optional) Indicates that the identification of *file.structure* will follow.

id

The identification number of this *file.structure* within a report. Each *file.structure* must contain a unique ID to enable secondary files to identify their parent's *file.structure*. After a *file.structure* (primary or secondary) is selected for reading while running ReportWriter, a new list of related *file.structures* is available. All secondary *file.structures* must come from these lists.

PARENT

(optional) Indicates that the identification of this *file.structure*'s parent *file.structure* will follow.

pid

The identification number of this *file.structure*'s parent. It must have been previously defined. The first *file.structure* is assumed to be the primary file, and any parent ID specified ignored.

SEPARATE

(optional) Indicates that this *file.structure* and all of its siblings (other *file.structures* that have been chosen from the same parent's list) should be used individually to create detailed records first from the first *file.structure*, then second from the second *file.structure*, and so on for the rest of the siblings. This flag must be set for the last sibling.

RELATION

(optional) Indicates that the name of the relation between this *file.structure* and its parent will follow.

name

The name of the relation between this *file.structure* and its parent. The name is a two-digit alphanumeric field that is obtained from the repository. This argument is not required if only one relation exists between this *file.structure* and its parent.

LINK_FIELD

(optional) Indicates that this *file.structure* is selected from its parent *file.structure*'s name link field.

link_field

The field in the parent *file.structure* that has the same name link as the first segment of this *file.structure*.

LINK_NAME

(optional) Indicates that this *file.structure* is selected from its parent *file.structure* name link field.

link_name

The name link that is used to build a name linked relation.

TO_KEY

(optional) Indicates that the name of the access key will follow.

to_key

The name of the access key in the file you're defining to which a name link relation has been established with the parent.

LINK_SEGMENT

(optional) Indicates that this name link relation can use a user-modified key. Nonlink base segments can be modified. If no LINK_SEGMENT argument follows a LINK_NAME argument, RDL will use the default name link method to build a link key. See "Determining the list of available secondary files based on name links" on page 18-3 for information about the default name link method.

segment

The key segment that could be a field name or a literal value. A name link may require more than one segment to establish a relation. You can list more than one segment (separated by spaces) for such a link relation.

Discussion

The FILE statement describes a file with an associated structure from which ReportWriter will read when a report is processed.

The files determine which structures can be read from when a report is being processed. The order in which they are listed in the report schema determines the order in which they will be processed by ReportWriter. The first file listed is assumed to be the primary file. The maximum number of files to read in a report is 49.

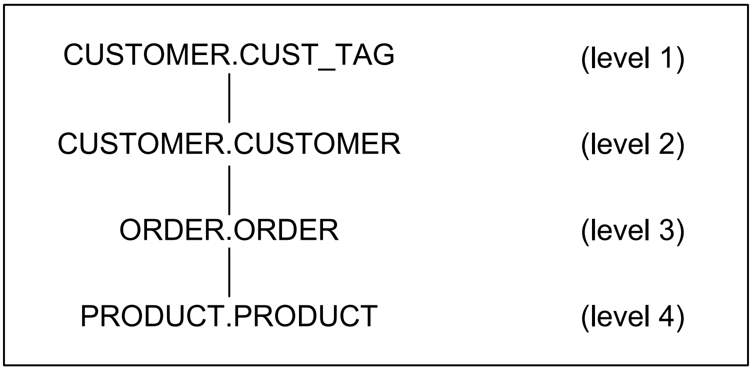
You can specify either a relation (with the RELATION keyword) or a name link relation (with the LINK_FIELD, LINK_NAME, TO_KEY, and LINK_SEGMENT keywords). You cannot specify both.

Examples

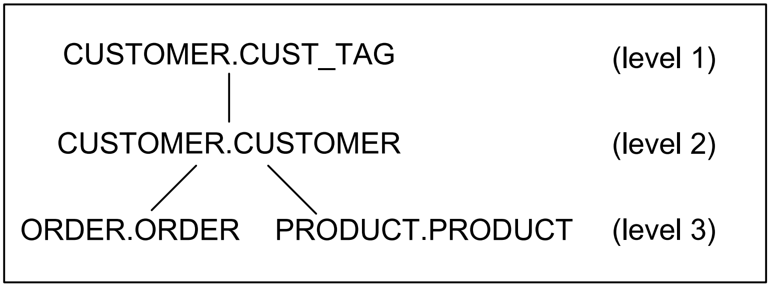
In the example below, CUSTOMER.CUST_TAG is the primary *file.structure* (ID = 1), and CUSTOMER.CUSTOMER, ORDER.ORDER, and PRODUCT.PRODUCT are secondary *file.structures*. Since there is more than one relation between ORDER.ORDER and its parent (CUSTOMER.CUSTOMER), the relation name (3) must be specified.

```
FILE CUSTOMER.CUST_TAG ID 1
FILE CUSTOMER.CUSTOMER ID 2 PARENT 1
FILE ORDER.ORDER       ID 3 PARENT 2 RELATION 3
FILE PRODUCT.PRODUCT   ID 4 PARENT 3
```

The example contains four levels of *file.structures*, as shown below.



However, if the parent of PRODUCT.PRODUCT were CUSTOMER.CUSTOMER, there would be only three levels, and PRODUCT.PRODUCT and ORDER.ORDER would be sibling *file.structures*, as illustrated below.



LINE – Describe a line

`LINE` *line_type options*

Arguments

line_type

The type of line. Valid values are:

REPORT_HEADER

REPORT_FOOTER

PAGE_HEADER

PAGE_FOOTER

PRE-BREAK

POST-BREAK

DETAIL

options

The following options for each line type:

REPORT_HEADER and **REPORT_FOOTER**

[`VISIBLE`*] [*`NOTVISIBLE`*] [*`NOTSEPARATE`*]*
[`NUMBLANK LINES` *blank_lines]*

PAGE_HEADER and **PAGE_FOOTER**

[`VISIBLE`*] [*`NOTVISIBLE`*] [*`NUMBLANK LINES` *blank_lines]*

PRE-BREAK and **POST-BREAK**

[`IF` *conditional_spec] [*`SUPPRESS BLANKLINE`*]*
`BREAKFIELD` *field_spec [*`SID` *sid]*

DETAIL

[`IF` *conditional_spec]*

VISIBLE

(optional) Includes the contents of a report header, report footer, page header, or page footer section in the report. The default is for both header sections to be included and both footer sections to be excluded.

NOTVISIBLE

(optional) Excludes the contents of a report header, report footer, page header, or page footer section from the report. The default is for both header sections to be included and both footer sections to be excluded.

NOTSEPARATE

(optional) Causes the report header and footer sections to be on the same page as other report text. The default is for the report header and footer sections to be on separate pages.

NUMBLANK LINES

(optional) Indicates that the number of blank lines to appear between the header and footer sections and the body of the report will follow if NOTSEPARATE is specified.

blank_lines

The number of lines of between the header and footer sections and the body of the report.

IF

(optional) Indicates that a conditional specification will follow.

conditional_spec

Specifies a condition or set of conditions in a report. This line will not appear in the report unless the conditional evaluates to true. See "conditional_spec" on page 17-6 for more information.

SUPPRESS BLANKLINE

(optional) Suppresses printing of a line in a report (the following line for a pre-break line or the preceding line for a post-break line) if that line is blank. (A line is considered blank if it contains no data or if all conditionals on all fields are false.) See "Suppressing a blank line" on page 10-12 for more information.

BREAKFIELD

(optional) Indicates that a break field argument will follow. This keyword is only valid for pre-break and post-break lines to designate at which sort level they should be printed.

field_spec

A specification for a sort field in a report. See "field_spec" on page 17-5 for more information.

SID

(optional) Indicates that a sort field identification number will follow. It is required when you are sorting on the same field more than once.

sid

The sort field identification number that indicates the sequential number of the sort field specified by *field_spec*. For example, if your SORT statement for the current report is

```
SORT company BY name BY name BY date
```

and you wanted to specify the second *name* break field, the sort field identification number would be 3 because the second *name* is the third sort field.

Discussion

The LINE statement describes the contents of a line within the report layout. At a minimum, you must specify the line type.

If there are several lines in your report, you must follow this order to ensure that your report is generated correctly:

Report header
Page header
Pre-break line
Detail line
Post-break line
Page footer
Report footer

All FIELD and TEXT statements are associated with the most recently defined line. If a page header line is the most recently defined line, any subsequent TEXT or FIELD statements will turn off the default field headers mode.

ReportWriter only generates the SID keyword and sort field identification numbers when you are sorting on the same field twice.

Examples

In the example below, the report header will default to the report name and will be on a separate page. The page header will default to containing the column headings for the detail line fields. The pre-break line will be printed in the report only if TEMP.STATE and CUST_TAG.STATE are equal, and the blank line that would normally appear in the event of a false conditional will be suppressed. The break level at which the pre-break line should be printed is defined by the TEMP.STATE field.

```
LINE REPORT_HEADER

LINE PAGE_HEADER
LINE PRE-BREAK IF CUST_TAG.STATE EQ TEMP.STATE
SUPPRESS BLANKLINE BREAKFIELD TEMP.STATE

LINE DETAIL
FIELD CUST_TAG.CUST_NAME COLUMN 1 HEADER "Customer name"
FIELD CUST_TAG.STATE COLUMN 20 HEADER "State"
FIELD ORD_ID COLUMN 24 HEADER "Order ID" JUST LEFT
FIELD ORD_ITEM COLUMN 32 HEADER "Product ID" JUST LEFT
```

MISCELLANEOUS – Describe miscellaneous report options

MISCELLANEOUS *[*LINES PERPAGE *lines] [*NUMBLANK LINES *blank_lines]*
*[*NUMBRKBLANK LINES *brk_blank_lines] [*BREAK COUNT*] [*NODATE ANDPAGNUM*]*
*[*NODETAIL LINES*] [*NORECORD COUNT*] [*NOFORMFEED*] [*NODASHED LINES*]*
*[*NOTOTAL DESCR*] [*REPORT DESCR "*description*"*]*

Arguments

LINES PERPAGE

(optional) Indicates that the number of lines per page will follow.

lines

The number of lines per page in the report. The default and the maximum is 66.

NUMBLANK LINES

(optional) Indicates that the number of blank lines between data records will follow.

blank_lines

The number of blank lines between data records. The default is 0.

NUMBRKBLANK LINES

(optional) Indicates that the number of blank lines between break sets will follow.

brk_blank_lines

The number of blank lines between break sets. The default is 2.

BREAK COUNT

(optional) Indicates that the report should include the number of breaks that occurred for each break (sort) level. If your report has only one break level, the break field count appears at the end of the report. If your report has more than one break level, a break field count appears at the end of each break level (except for the lowest level).

NODATE ANDPAGNUM

(optional) Indicates that no date or page number will be printed in the report's page headers.

NODETAIL LINES

(optional) Indicates that no detail lines will be printed in the report.

NORECORD COUNT

(optional) Indicates that the total record count in the report will not be printed.

NOFORMFEED

(optional) Indicates that blank lines will be issued for the remaining lines on a report page rather than issuing a form feed.

NODASHED LINES

(optional) Indicates that no dashed lines will be printed above a totaled field.

NOTOTAL DESCR

(optional) Indicates that the break and report summary total descriptions will not be printed.

REPORT DESCR

(optional) Indicates that the report summary description will follow.

description

The report summary description. The maximum size of the report description is 64.

Discussion

The MISCELLANEOUS statement defines global ReportWriter settings and can be set only once per report.

Examples

```
MISCELLANEOUS LINES PERPAGE 50
```

```
MISCELLANEOUS NOFORMFEED NODETAIL LINES
REPORT DESCR "Test Report Summary"
```

```
MISCELLANEOUS NODASHED LINES NOTOTAL DESCR
```

QUESTION – Describe a question field

QUESTION *name* TYPE *type* PROMPT "*prompt*" *[*DESCRIPTION "*description*"*]*
*[*FORMAT "*format*"*]* *[*DEFAULT "*default*"*]* *[*INFO LINE "*info_line*"*]* *[*JUST *just]*
*[*REQUIRED*]*

Arguments

name

The name of the temporary question field to be created. It can have up to 15 characters but cannot contain spaces.

TYPE

Indicates that the data type will follow.

type

The data type of the question field. It must be one of the following statements. (See the Discussion on page 17-26 for an explanation of each option.)

ALPHANUMERIC SIZE *size [*UPPERCASE*]*

NUMERIC SIZE *size*
*[*PRECISION *dec_places]* *[*NODECIMAL*]*
*[*NEGATIVE*]* *[*BLANKIFZERO*]*
DATE STORED *store_format* *[*DATE TODAY*]*

USER SIZE *size*
USERTYPE "*user_type_id*"
*[*USERTEXT "*user_text*"*]* *[*UPPERCASE*]*

TIME

PROMPT

Indicates that the prompt for the question field will follow.

prompt

The text of the question field's prompt to the user.

DESCRIPTION

(optional) Indicates that the description of the question field will follow.

description

A description of the question field. It can have up to 40 characters.

FORMAT

(optional) Indicates that the format for this question field will follow.

format

The display format for the question field. It can have up to 40 characters.

DEFAULT

(optional) Indicates that the default answer to the question field will follow.

default

The default answer to the question field. It can have up to 80 characters.

INFO LINE

(optional) Indicates that the information line text of the question field will follow.

info_line

The text that appears on the information line when this question field is displayed. It can have up to 80 characters.

JUST

(optional) Indicates that a data justification argument will follow.

just

The justification of the question field. It must be one of the following values:

LEFT
CENTER
RIGHT

The default is LEFT for alpha, time, and user type fields, and RIGHT for decimal, implied-decimal, and date fields. CENTER alignment is not allowed for numeric, date, and time fields. RIGHT alignment is not allowed for alpha and user fields.

REQUIRED

(optional) Specifies that a nonblank alpha or nonzero numeric entry is required in the field.

Discussion

The QUESTION statement describes a temporary question field to be created in a report. Question fields store answers that the user enters when the report is run. They are temporary fields that are not found in the repository. The temporary field's name, specific data type arguments and prompt are required. For alphanumeric and numeric data types, the length of the question field is required. For date data types, the *store_format* of the field is required. For user data types, the length and user type ID are required. The maximum number of question fields that can be defined in a report is 99.

If your temporary question field does not have a unique name, it must be preceded by “TEMP.” For example, if your temporary field is called STATE and one of the selected files also contains a field called STATE, you would specify the temporary field like this:

```
TEMP.STATE
```

The *type* argument options contain the following keywords and arguments:

Alphanumeric type

SIZE

Indicates that the length of the question field will follow.

size

The length of the question field.

UPPERCASE

Converts lowercase input characters to uppercase.

Numeric type

SIZE

Indicates that the length of the question field will follow.

size

The length of the question field.

PRECISION

(optional) Indicates that the number of decimal places in an implied-decimal field will follow.

dec_places

The number of characters to the right of the decimal point in an implied-decimal type question field. If the data type is implied-decimal, this attribute must be present, and it must be less than or equal to the size of the field. Otherwise, the field is ignored.

NODECIMAL

(optional) Indicates that the user does not need to type a decimal point when placing input in a decimal or implied-decimal question field.

NEGATIVE

(optional) Allows negative values in decimal and implied-decimal question fields.

BLANKIFZERO

(optional) Indicates that a decimal or implied-decimal question field will be left blank if the user enters a value of 0.

Date type

STORED

Indicates that the storage format for the date type question field will follow.

store_format

Specifies the storage format. Valid values are:

YYPP
YYJJJ
YYMMDD (default)
YYYYPP
YYYYJJJ
YYYYMMDD

DATE TODAY

(optional) Defaults the date to today's date if the user presses ENTER without entering anything in a blank question field.

User type

SIZE

Indicates that the length of the question field will follow.

size

The length of the question field.

USERTYPE

Indicates that the user-defined data type ID will follow.

user_type_id

The identification of the user-defined data type. It can have up to 30 characters.

USERTEXT

(optional) Indicates that a user-defined text string will follow.

user_text

(optional) A user-defined text string associated with the question field. It can have up to 80 characters.

UPPERCASE

(optional) Converts lowercase input characters to uppercase.

Time type

The time data type has no arguments.

Examples

- The question field defined below is an alphanumeric field of size 30. The prompt is “City:”, the answer defaults to all cities, and the text “Enter name of city.” appears on the information line when the prompt is displayed at report generation time.

```
QUESTION CITY TYPE ALPHANUMERIC SIZE 30 PROMPT "City:"
DEFAULT "*" INFO LINE "Enter name of city."
```

- The question field defined below is a numeric field of size 4. (In other words, 9999 is the maximum number that can be entered.) The user is required to enter a value for this field.

```
QUESTION CUST_ID TYPE NUMERIC SIZE 4 PROMPT "Customer ID:"
INFO LINE "Enter customer identification number." REQUIRED
```

- In the example below, the question field is a date field of size 6 (YYMMDD). The user is required to enter a value for this field, and the field will be displayed in the MM/DD/YYYY format if it is selected as a field to print.

```
QUESTION BEG_DATE TYPE DATE STORED YYMMDD
PROMPT "Beginning date:" FORMAT "MM/DD/YYYY"
INFO LINE "Enter the beginning date to read from." REQUIRED
```

- The question field below is a user-defined data type field of size 8. The identification of the user’s data type, “U_DATE1”, and the text “MM-DD-YYYY” is passed to the RPS_DATA_METHOD subroutine for display formatting.

```
QUESTION BEG_DATE TYPE USER SIZE 8 USERTYPE "U_DATE1"
USERTEXT "MM-DD-YYYY"
INFO LINE "Enter the beginning date to read from." REQUIRED
```

REPORT – Describe a report definition

```
REPORT "name"
```

Arguments

name

The name of a report.

Discussion

The REPORT statement describes a report definition.

One or more reports can be defined in a report schema file. The end of a report is signified by another REPORT statement or the end of the file. If another report with this name already exists in the report definition file, an error will be logged and the report will be ignored. Only the first 40 characters of *name* will be used, and *name* will be uppercase.

Examples

```
REPORT "Customer Profiles"
```

```
REPORT "Employee Profiles"
```

```
REPORT "Sales - Year To Date"
```

SELECT – Describe the fields to be compared in selecting records

`SELECT` *conditional_spec*

Arguments

conditional_spec

Specifies a condition or set of conditions in a report. The maximum number of conditions for a selection statement is 25. See “conditional_spec” on page 17-6 for more information.

Discussion

The SELECT statement describes the field(s) to be used and compared in selecting the records for a report.

A selection entry enables a user to choose which records should be included in a report.

Examples

In the following example, only records that have the same city and state as TEMP.CITY and TEMP.STATE will be included in the report.

```
SELECT CUST_TAG.CITY EQ TEMP.CITY AND
CUST_TAG.STATE EQ TEMP.STATE
```

In the example below, only records that have customer identification numbers greater than 200 and an item identification number of 13 will be included in the report.

```
SELECT CUST_ID GT "200" AND
ITEM_ID EQ "13"
```

SORT – Describe a sort field

```
SORT field_spec [REVERSE] [LINEBREAK] [PAGEBREAK] [MERGE]
[BY field_spec [REVERSE] [LINEBREAK] [PAGEBREAK] [MERGE]...]
```

Arguments

field_spec

A specification for a field in a report. See “field_spec” on page 17-5 for more information.

REVERSE

(optional) Reverses the sort order of the current sort field.

LINEBREAK

(optional) Inserts a line break in the report whenever the value of the current sort field changes.

PAGEBREAK

(optional) Inserts a page break in the report whenever the value of the current sort field changes.

MERGE

(optional) Allows multiple projection files to be treated as one for the purpose of sorting.

BY

(optional) Indicates that another sort field will follow.

Discussion

The SORT statement describes a sort field. Sort fields help control the organization of a report. The order in which the sort fields are specified determines the sort levels within a report. The maximum number of sort fields is 10.

Examples

- The report in the example below has one sort level (CUST_TAG.STATE). All records from the first (alphabetically sorted) state are printed, then all records from the second state, and so on.

  ```
  SORT CUST_TAG.STATE
  ```

- In the example below, the report has two sort levels: CUST_TAG.CITY and CUSTOMER.CUST_NAME. The second sort level also defines a line break, so after each change in CUSTOMER.CUST_NAME, a line break is printed. All records from the first (alphabetically sorted) city are printed, then all records from the second city are printed, and so forth. For each city, all records from the first (alphabetically sorted) customer name are printed, then all records from the second customer name, and so on.

  ```
  SORT CUST_TAG.CITY BY CUSTOMER.CUST_NAME LINEBREAK
  ```

SUBTOTAL – Describe a subtotal access field

```
SUBTOTAL name TOTALFIELD totalfield BREAKFIELD breakfield
TOTALTYPE type [DESCRIPTION "description"]
```

Arguments

name

The name of the temporary subtotal access field. It can have up to 15 characters but cannot contain spaces.

TOTALFIELD

Indicates that the field on which to total will follow.

totalfield

One of the following:

field_spec (no subscript or range)

A specification for a field in a report. The data type for this field must be numeric. See "field_spec" on page 17-5 for more information.

<COUNT>

Designates that the subtotal access field should contain the record count.

BREAKFIELD

Indicates that the sort level on which to total will follow.

breakfield

One of the following:

field_spec (no subscript or range)

A specification for a field in a report. This field must be a previously defined sort field with an associated line or page break. See "field_spec" on page 17-5 for more information.

<REPORT>

Designates that the subtotal access field should contain a total for the entire report.

TOTALTYPE

Indicates that the type of total will follow.

type

The type of total for a report. Valid values are:

COMPLETE	The total for the specified break level.
RUNNING	The current cumulative total for the given field at the specified level.

DESCRIPTION

(optional) Indicates that the description of the subtotal access field will follow.

description

A description of the subtotal access field. It can have up to 40 characters.

Discussion

The SUBTOTAL statement describes a temporary subtotal access field to be created in a report.

Subtotal access fields give access to a field's total at any sort level. They are temporary fields that are not found in the repository. To create a field of this type, you must specify the field name, total field, break field, and total type. The maximum number of subtotal access fields that can be defined in a report is 99.

If your temporary subtotal access field does not have a unique name, it must be preceded by "TEMP." For example, if your temporary field is called NUMBER and one of the selected files also contains a field called NUMBER, you would specify the temporary field like this:

```
TEMP.NUMBER
```

Examples

- In the example below, NUMRECS is the name of the field that contains a record count for the whole report. The value of NUMRECS is the same for each record in a break level (which in this case is the whole report).

```
SUBTOTAL NUMRECS TOTALFIELD <COUNT> BREAKFIELD <REPORT>
TOTALTYPE COMPLETE
```

- In the following example, INCOME is the name of the field that contains the totaled numeric value of the ITEMCOST field for the sort field level of CUST_ID. The value of INCOME is a cumulative total for each record in a break level (which in this case is each different CUST_ID).

```
SUBTOTAL INCOME TOTALFIELD ITEMCOST
BREAKFIELD CUST_ID TOTALTYPE RUNNING
```

TEMPTEXT – Describe a temporary text field

```
TEMPTEXT name TTEXT "text" [DESCRIPTION "description"]
```

Arguments

name

The name of the temporary text field. It can have up to 15 characters and cannot contain spaces.

TTEXT

Indicates that the text of the temporary text field will follow.

text

The text of the temporary text field. It can have up to 80 characters.

DESCRIPTION

(optional) Indicates that a description of the text field will follow.

description

A description of the text field. It can have up to 40 characters.

Discussion

The TEMPTEXT statement describes a temporary text field to be created in a report.

A text field defines a text string to print. Temporary text fields are not found in the repository. You must specify the temporary text field's name and text. The maximum number of text fields that can be defined in a report is 99.

If your temporary text field does not have a unique name, it must be preceded by "TEMP.". For example, if your temporary field is called CITY and one of the selected files also contains a field called CITY, you would specify the temporary field like this:

```
TEMP.CITY
```

Examples

- In the following example, TEMP1 is the name of the temporary text field and "*" is the text for this field.

  ```
  TEMPTEXT TEMP1 TTEXT "*"
  ```

- In the example below, TEMP2 is the name of the temporary text field and "<-------" is the text for this field. The description of the field is "After Deadline."

  ```
  TEMPTEXT TEMP2 TTEXT "<-------"
  DESCRIPTION "After Deadline"
  ```

TEXT – Describe text to be printed

```
TEXT "text" COLUMN column
```

Arguments

text

Text to be printed on the current line.

COLUMN

Indicates that the column position for the text will follow.

column

The column position for the text.

Discussion

The TEXT statement describes text to be printed in a report. Text can be printed on the following types of lines:

REPORT_HEADER
PAGE_HEADER
PAGE_FOOTER
REPORT_FOOTER

To print text on a break or detail line, use a temporary text field.

The maximum number of columns in the report is 255.

Examples

```
TEXT "The grand total is: " COLUMN 40
```

18

Miscellaneous ReportWriter Information

Selecting Files to Read

When you create a report, the first file you select is the *primary* file. All other files that you select are *secondary* files. This section describes how ReportWriter determines the list of available primary and secondary files. It also describes how ReportWriter determines the list of secondary files based on name links stored in the repository cross-reference file.

ReportWriter only includes files that are not flagged as temporary. See the "Defining a New File" in the "Working with Files" chapter of the *Repository User's Guide* for information about flagging file definitions as temporary.

Determining the list of available primary files

ReportWriter reads the repository and then lists all files that have one or more structures assigned to them. Each such file is listed once in the Available Files list. If the file you select has more than one structure assigned to it, the Available Structures list is then displayed. From this list you can select the appropriate structure. For example, if both

Structure : CMCLNT   Description : Customer client

and

Structure : MSCLNT   Description : Master client

are assigned to the CLIENT file, these two entries appear in the Available Structures list:

Customer client
Master client

Determining the list of available secondary files

Once a primary file is selected, you can display the list of available secondary files. Secondary files are related to the primary file, either directly or indirectly.

ReportWriter reads the repository to find all relations that are defined for the current structure. For example, using the items listed above, if you select

CLIENT.Master client

as your primary file, ReportWriter will look up all relations defined for structure MSCLNT. If the relations defined for MSCLNT are as follows:

From key	To structure	To key
CLNTID	ARCUST	CUSTID
CONTID	CMCONT	CONTID

ReportWriter will look up all files assigned to all “to” structures. These files will comprise the list of available secondary files. (These files are directly related to the primary file.) If ARCUST is assigned to the files AR92 (Accounts receivable 92) and AR93 (Accounts receivable 93), and CMCONT is assigned to the file CONTCT (Contact), the list of available files will be as follows:

Accounts receivable 92
Accounts receivable 93
Contact

Each file is listed once in the Available Files list. If the file you select has more than one structure assigned to it, the Available Structures list is then displayed. From this list you can select the appropriate structure.

Additionally, if the selected structure is used in more than one relation between it and the structure of the primary file, a third list—the Available Relations list—is displayed. This list displays the “from key” and “to keys” involved in the relations. From this list you can select the appropriate relationship between the primary and secondary file.

Once a secondary file has been selected, you can display the list of files related to it or redisplay those related to the primary file. Files linked to the secondary file are considered to be indirectly related to the primary file.

The primary file can also appear as a secondary file; this will happen if one of the secondary files defines a relation back to the primary file.

Determining the list of available secondary files based on name links

Once a primary file is selected, you can display the list of secondary files that are available through name links, rather than those based on predefined relations. ReportWriter creates a foreign “from” key and a temporary relation in order to access these secondary files.

Name links are established in Repository and allow “name linked” fields to be used to access related data. Each field (and template) in the repository has a “name link” flag. This flag determines whether the field name itself should be used to find matches or if an alternate name link should be used. The alternate name link is the name of the field’s template or the name of the template’s parent if the name link flag is set.

The list of possible name link relations for a given repository exists in the repository cross-reference file, which is generated by the Repository Generate cross-reference utility. Each record in this file links a field with an access key in another structure.

When you select Add file via name link, ReportWriter reads the cross-reference file in order to display a list of fields in the current file that have name links to other files. These fields are displayed in the Link Fields from *filename* list.

When you select one of the link fields, ReportWriter reads the cross-reference file to find all access keys in all structures to which this field has a name link. ReportWriter then looks up all files assigned to these structures. These files will comprise the list of available secondary files.

Each file is listed once in the Available Files list. If the file you select has more than one name linked structure assigned to it, the Available Structures list is displayed. From this list you can select the appropriate structure.

Additionally, if more than one record was found in the cross-reference file linking the link field with the selected structure, a fourth list—the Available Relations list—is displayed. This list displays the “link field” and the “to” key (the access key) involved in each possible relation. From this list you can select the relationship that ReportWriter should construct between the primary and secondary file.

Constructing temporary relations

When you select a secondary file (based on a name link) ReportWriter constructs both a foreign key (the “from” key) and a temporary relation, linking this “from” key to the selected “to” key. Depending on the link field selected and the “to” key used, the following situations are possible:

- If the link field is the first and only segment of the “to” key, ReportWriter constructs a single-segment, nonmodifiable foreign key. (Note that the link field name and the “to” key field name will not necessarily be the same. The reason the link is possible is that both fields have the same name link.)
- If the link field is the second segment of the “to” key and the first segment is the tag field, ReportWriter creates a two-segment, nonmodifiable foreign key of which the first segment is a literal (the tag comparison value) and the second segment is the link field. The tag must be defined as a **Field** type tag and must use the EQ (equal) connector.
- If the link field is the first segment of the “to” key and is followed by other segments, ReportWriter constructs a foreign key that matches as many segments as possible, based on name links. If any segments in the “to” key do not have corresponding name-link matching fields in the “from” structure, you must enter literal values for the remaining segments or clear the segments so that ReportWriter does a partial match.

Using external key segments

Repository allows for a special kind of relationship between files, called an external relation. An external relation involves three or more files, where one file is accessed by a key composed of segments from the remaining files. For example, the item type from one file (file A), along with the item number from a second file (file B), can be used to access the item ID in a third file (file C).

When you create a report, the list of available files will contain all three files. You will want to select file A first. After you select file A, the list of available files will only include file B. File C will *not* be displayed, because it requires file B. Once you select file B, file C will be displayed, and you can select it.

Similar logic applies when you try to delete files from the report. If you've selected all three files, you cannot delete file B until file C has been deleted.

Using non-indexed files

ReportWriter currently supports two non-indexed file types: ASCII and relative. Since ReportWriter does random access on all secondary files, ASCII files can be used only as primary files. Relative files can be used as both primary and secondary files.

In order for ReportWriter to access relative files properly, an access key must be defined for those files in Repository. A relative file can have only one access key: the record number. For convenience, when you create a structure whose file type is relative, Repository automatically creates an access key with the proper attributes. The key is named RECORD_NUMBER, and it has an ascending sort order, allows no duplicates, and has one segment of type R (record number). In addition to the access key created for you, you can also define foreign keys for a relative file.

When using a relative file as your primary file, you have two methods of accessing other files:

- If you use the RECORD_NUMBER access key as the "from" key in the relation, as ReportWriter processes sequentially through the file, the current record's position will be used as the key value for accessing the related file.
- You can also use a foreign key. When sequentially processing the file, the current record is used to construct the foreign key, and its value is used as the key for accessing the related file.

Because a secondary file (the "to" part of a relation) can be accessed only via access keys, the only method of accessing a relative secondary file is via the record number. The key value is constructed from the primary file (whether it be a field, an external key, a literal, or a relative file's record number), and it is then used as the number of the record to read from the secondary file.

Understanding Multiple Projections

Hierarchical relationships

One or more secondary files linked indirectly to the primary file define a hierarchical relationship (File A → File B → File C).

When hierarchical relationships are processed, one detail record is created for each possible combination of matching key values in all files. For example, given these key values:

File A	File B	File C
1	1a	1a
2	1b	1b
	2a	1c
	2b	2a
		2b

these detail records would be generated:

```
A1, B1a, C1a
A1, B1a, C1b
A1, B1a, C1c
A1, B1b, C1a
A1, B1b, C1b
A1, B1b, C1c
A2, B2a, C2a
A2, B2a, C2b
A2, B2b, C2a
A2, B2b, C2b
```

In hierarchical relationships, the total number of detail records created for a given primary file key value is the number of records for that key value in each of the secondary files, multiplied together. For instance, in the example above, the total number of detail records for a primary file value of “1” is 6 (the number of matching “1” records from File B multiplied by the number of matching “1” records from File C, or 2 * 3 = 6).

Tree relationships

One or more secondary files linked directly to the primary file create a tree relationship (File A → File B and File A → File C).

When tree relationships are processed, one detail record is created for each occurrence of a matching key value in either of the secondary files. For example, given these key values:

	File A	
	1	
	2	
File B		**File C**
1a		1a
1b		1b
2a		1c
2b		2a
		2b

these detail records would be generated:

```
A1,  B1a,    C1a
A1,  B1b,    C1b
A1,  <null>, C1c
A2,  B2a,    C2a
A2,  B2b,    C2b
```

In tree relationships, the total number of detail records created for a given primary file key value is the maximum number of records with a matching key value in any one of the secondary files. For instance, in the example above, the total number of detail records for a primary file value of “1” is 3. (The maximum number of matching “1” records from any one of the secondary files is 3, from File C.)

Multiple projections enable you to alter the default detail record creation logic for tree relationships.

The requirements for using multiple projections are as follows:

- Two or more secondary files must be selected.
- These secondary files must be on the same level of the tree.
- The relationship between the primary file and the secondary files must be one-to-many.

Multiple projections enable you to create detail records from one file, then from the next file, for a given key value. For instance, in the example above, you could create detail records as follows:

```
A1,  B1a,     <null>
A1,  B1b,     <null>
A1,  <null>,  C1a
A1,  <null>,  C1b
A1,  <null>,  C1c
A2,  B2a,     <null>
A2,  B2b,     <null>
A2,  <null>,  C2a
A2,  <null>,  C2b
```

Notice that the total number of detail records created for a given primary file key value is the total of the number of records with a matching key value in all secondary files. For instance, in the example above, the total number of detail records for a primary file value of “1” is 5 (the number of matching “1” records from File B, plus the number of matching “1” records from File C, or 2 + 3 = 5).

When using the above method of detail record creation, the %GOTREC intrinsic function will tell you whether or not a record was found in a given file. You can assign the 0/1 value from this intrinsic to a temporary field via a calculation and then use the field in a record selection criterion or in a print or calculation conditional.

When using multiple projections to process the records in the secondary files separately, you can also specify common sort fields for each of the secondary files and then merge them. See “Merging sort fields” on page 10-6 for more information.

Let’s suppose you have two secondary files, a sales history file and a current year sales file. Possible uses for this alternate method of record processing are as follows:

- To generate all current sales data for a client, followed by all history data
- To generate subtotals for the current sales data and/or subtotals for the history data
- To merge the current and past sales data together into one sorted report

Understanding the ReportWriter Processing Flow

Because ReportWriter presents an intuitive interface for creating reports, you may not care to know its internal processing sequence under most circumstances. On the other hand, the order of events may play an important role when you're designing more complex reports. Under such circumstances, we hope you'll find the following outline useful.

Build phase

When you elect to run a report, ReportWriter must first build an executable code structure from the report definition. During this phase, the following occurs:

1. The RW_INITBLD_METHOD routine is called.

   This "user hook" routine enables you to initialize any global data or perform any other one-time initial processing required by the other "user hook" routines.

2. The files for the report are opened.

   For each filename to open, ReportWriter calls the RPS_FILNAM_METHOD routine to perform any runtime mapping of the filename before attempting the open. At this point, if the file type is user-defined, the RPS_OPEN_METHOD routine is called.

3. ReportWriter determines whether sorting is necessary and what access key and initial find values are optimal.

4. The report definition structures are translated into executable code.

Generation phase

After building the report, ReportWriter generates the report from the files to read and the code that was created during the Build Phase. During the Generation Phase, the following occurs:

1. Environment variables are loaded.

   The RW_ENV_METHOD routine is called once for each environment field to load its contents.

2. Question fields are processed.

   ReportWriter builds input windows from the question field parameters and accepts input.

3. Any sorting is performed.

   If sorting is necessary, each set of records is read from the related files (see "Reading sets of records from related files" on page 18-11), and an intermediate file is created from all fields being accessed. This file is then sorted and becomes the new (and only) input file for the report. Also included in this file are the results of any calculation fields that are not dependent on subtotal access fields and that were not explicitly specified to occur after sorting. If any of the file types are user-defined, the routine RPS_READS_METHOD is used to retrieve the data.

4. The report's output file is opened.

   If the report is being displayed, the screen is accessed. If a report file is being printed or generated, the routine RW_PRTOPEN_METHOD is called.

5. The first record set is read.

   If a sort was performed, this is just the first record from the sorted intermediate file. If no sort was performed, the first set of records from the related files are read in. (See "Reading sets of records from related files" on page 18-11.) Any calculation fields that depend on subtotal access fields, or that were specified to occur after sorting, are calculated. If no record set is available, skip to step 12 below.

6. The report header is printed.

   If visible, the report header lines are passed individually to the RW_HEADER_METHOD routine for any runtime changes and are then printed.

7. Paging is performed.

   If a new page is needed (and this is always true at the beginning of the report), a page break occurs. If there was a previous page, any visible page footer is first printed. Each line of the footer is passed to the RW_FOOTER_METHOD routine for any runtime modifications and is then printed. A new page is ejected, and any visible page header is printed. Each line of the page header is passed to the RW_HEADER_METHOD routine for any runtime modifications and is then printed.

8. A pre-break is processed.

   If necessary, subsequent record sets are processed to accumulate any subtotal access fields that are "Complete" subtotals, and the position in the file(s) is reset. Any subtotal-dependent calculation fields are recalculated. All appropriate pre-break print lines are printed (which is all of them the first time).

9. The record set is processed.

   If a new page is needed, paging is performed as described in step 7 above. "Fields to print" lines are printed. If any of the printed fields are defined as type user-defined, the RPS_DATA_METHOD routine is called to format the output. The contents of all fields accessed in post-break print lines are copied off.

10. The next record set is read.

    As described in step 5 above, the next record from the intermediate file, or the next appropriate set of records from the files, is returned. Subtotal-dependent or post-sort calculations are performed. If no next record set is available, skip to step 12 below.

11. Breaks are checked.

    If a break field has changed, any appropriate counts, subtotals, or post-break print lines are printed and subtotal accumulators are cleared. If the break elicits a new page, return to step 7 above. If the break does not require a page break, return to step 8 above. If no break has occurred, return to step 9 above.

12. End processing is performed.

    A post-break for the report is processed as described in step 11 above, using grand totals. Any page footer is printed as described in step 7. If a report footer is visible, each line of the footer is passed to RW_FOOTER_METHOD for any runtime modifications and then printed. The output file is closed by calling RW_PRTCLOSE_METHOD. The input file(s) are also closed, calling RPS_CLOSE_METHOD if necessary.

Reading sets of records from related files

If the report data must be sorted, record sets are read from the input files before the sort occurs. If no sorting is required, the record sets are read in as the report output records are processed.

Reading the first record set

1. A record is read from the primary file. If no more records exist, return immediately, indicating “no more records” to ReportWriter.
2. Any selection criteria that are dependent only on the primary file are evaluated. If the results (combined with other selection criteria whose results are unknown) allow ReportWriter to exclude this record, return to step 1.
3. For the next related file, read any related record in the order in which the records are related. If no related record is found, or if we are currently processing multiple projections separately for another parallel file, the record will be set to blanks, and %GOTREC for the file will return 0. Any selection criteria that are dependent only on this file (or on this file and a prior file) are evaluated. If the results, combined with previously evaluated selection criteria and the unknown results of other selection criteria, allow ReportWriter to exclude this record set, return to step 1. Otherwise, repeat this step until all related files have been processed.
4. After all related files have been processed, any calculation fields that are not dependent on subtotal access fields and that have not been explicitly specified to occur after sorting are computed. Any remaining unevaluated selection criteria are evaluated. If the results, combined with previously evaluated selection criteria, logically exclude this record set, return to step 1. Otherwise, the record set is complete.

Reading subsequent record sets

These steps occur when reading subsequent record sets, starting with the last file relation level.

1. Get the next related record(s) at this level.

   - If we’re processing multiple projections separately, read the next related record from the current file in this relation level. If no next related record exists, set the record to blanks, set the return value of %GOTREC for this file to 0, and repeat this step for the next file at this relation level. If no more files exist at this level, repeat step 1 for the parent file. If there is no parent file, indicate “no more records” to ReportWriter.

- If we're not processing multiple projections separately, read the next related records from all files in this relation level. If no next related record exists in a file, set its record to blanks and its return value for %GOTREC to 0. If no related records exist for any of the files at this relation level, repeat step 1 for the parent file. If there is no parent file, return a "no more records" message to ReportWriter.

2. Any selection criteria that are dependent only on this file relation level (or on this level and a prior level) are evaluated. If ReportWriter can exclude the record set based on these criteria, return to step 1.
3. Follow steps 3 and 4 as described for the first record set above.

Understanding ReportWriter Optimization

ReportWriter optimizes its performance by doing the minimum amount of I/O to your data files and by sorting only when necessary.

By default, reports are processed in primary key order on the primary file. To process the primary file in a different order, sort on that field.

If the sort key specifications (all sort fields combined) match the initial segment(s) of an access key on the primary file, the access key is used instead of sorting.

Additionally, if you specify selection criteria that use the access key that's being used to read the file, the following optimizations are performed:

- If an EQ, GE, or GT selection is done on initial segment(s) of the key, these criteria are not immediately preceded by an OR and not followed by *any* OR, and the values being tested against are literals, an initial key value is constructed so that a READ to the first desired record can be performed.
- If an EQ, LE, or LT selection is done on initial segment(s) of the key, these criteria are not immediately preceded by an OR and not followed by *any* OR, and the values being tested against are literals, no further records are processed when that test fails.

If the sort key specifications do *not* match any of the access keys, sorting takes place. However, because selections are performed before sorting, if you specify selection criteria that use the primary access key, the optimizations described above are performed.

When using a relative file as the primary file, the access key being used to read the file is the record number. If you don't want to process every record in the primary file, you can optimize the performance by defining selection criteria that test the record number. For example, if the record number is your customer ID, and you only want to process records from ID 20 through ID 50, define a calculation field that is assigned the %RECNUM intrinsic function (see page 8-13). Then define selection criteria that compare the calculation field (the current record number) against the values 20 and 50.

Appendices

Appendix A: Maximums

Lists all size and number maximums permitted by ReportWriter.

Appendix B: Date and Time Formats

Lists the date and time display formats that ReportWriter supports.

Appendix C: Error Messages

Lists error messages that may appear in ReportWriter, along with an explanation of the problem that caused each error to occur.

Appendix D: Data Formats

Explains Synergy DBL data formatting.

Appendix E: Distributed Shortcuts

Lists the ReportWriter shortcuts as they are originally distributed.

Appendix F: Environment Variables

Describes the environment variables used by ReportWriter.

A

Maximums

Maximum Values Permitted by ReportWriter	
Item	**Maximum**
Calculation fields per report	99
Conditionals per detail or break line	1
Conditionals per field or calculation	1
Conditionals per report	99
Criteria per conditional	5
Enumerated fields per report	99
Environment field user text (size in characters)	40
Environment fields per report	99
Expression in calculation field (size in characters)	150
Field header (size in characters)	40
Field that can be accessed (size in characters)	99
Fields per detail line	99
Fields per header/footer line	99
Fields to print per post-break line	99
Fields to print per pre-break line	99
Fields to print per report (detail record)	990
Files per report	50
Format string (size in characters)	40
Lines per detail record	10

Maximum Values Permitted by ReportWriter (Continued)	
Item	**Maximum**
Lines per split (^) field header	3
Pre- or post-break print lines per break	1
Question field default value (size in characters)	80
Question field information line (size in characters)	80
Question field prompt (size in characters)	40
Question fields per report	99
Range references per report	300
Report name (size in characters)	40
Report/page headers/footers, 10 rows x 255 columns (size in characters)	2550
Reports	no limit
Selection criteria per report	25
Sort field breaks per report	10
Sort fields per report	10
Subscript references per report	300
Subtotal access fields per report	99
Temporary field description (size in characters)	40
Temporary field name (size in characters)	15
Temporary fields (total of all types of temporary fields)	495
Text field (size in characters)	80
Text fields per report	99
Width of a report (in characters)	255

B

Date and Time Formats

This appendix lists the date and time display formats supported by ReportWriter.

Date Display Formats

Date display formats supported by ReportWriter:

#01	MM/DD/YYYY
#02	MM/DD/YY
#03	MM-DD-YYYY
#04	MM-DD-YY
#05	Mon/DD/YYYY
#06	Mon/DD/YY
#07	Mon-DD-YYYY
#08	Mon-DD-YY
#09	YYYY/MM/DD
#10	YY/MM/DD
#11	YYYY-MM-DD
#12	YY-MM-DD
#13	YYYY/Mon/DD
#14	YY/Mon/DD
#15	YYYY-Mon-DD
#16	YY-Mon-DD
#17	DD/MM/YYYY
#18	DD/MM/YY
#19	DD-MM-YYYY
#20	DD-MM-YY
#21	DD/Mon/YYYY
#22	DD/Mon/YY
#23	DD-Mon-YYYY
#24	DD-Mon-YY
#25	PP/YYYY
#26	PP/YY
#27	PP-YYYY
#28	PP-YY

where

MM is the one- or two-digit month.
Mon is the three-letter abbreviation for the month.
DD is the one- or two-digit day.
YYYY is the year, including the century.
YY is the last two digits of the year.
PP is the period.

The format size and the display size don't have to match. For example, if you choose a two-digit year format for a field that's stored as a four-digit year, ReportWriter automatically omits the century from the display format.

Time Display Formats

Time display formats supported by ReportWriter:

#01	12:MM
#02	24:MM
#03	12:MMm
#04	12:MM:SS
#05	24:MM:SS
#06	12:MM:SSm

where

12 is the hour in 12-hour notation.
24 is the hour in military (24-hour) notation.
MM is the minute.
SS is the second.
m is the meridian a or p (ante or post).

C

Error Messages

This appendix lists error messages that may appear in ReportWriter, along with explanations of the problems that may have caused the error. If you receive messages that are not listed in this appendix, contact your Synergy/DE Developer Support engineer at at 800.366.3472 (in the U.S. and Canada) or 916.635.7300.

***Filename* is not a version 7 or version 10 repository.**

Repository version 10 can only open repository files that are in version 7 format or version 10 format. (Version 7 format was used by Repository versions 7, 8, and 9; it can be converted to version 10 format by the version 10 Repository.) It is likely that the specified repository predates version 7 and therefore can be neither opened nor converted by version 10. If this is the case, you must use the repository conversion program to convert it. Refer to the Repository release notes (REL_RPS.TXT) for instructions.

Partial expression **can only be used in a subscript or ranged reference.**

The "," character is only valid when specifying a subscript value for an arrayed field or when ranging a field.

Partial expression **cannot be used as a unary operator in an expression.**

Only "+" and "–" can be used as unary operators (to designate positive and negative). The specified operator requires two operands.

Partial expression **must follow a field name.**

The "[" character must follow a field name. It designates the subscripting of an arrayed field.

Alpha field not allowed: *field name*

You cannot specify an alpha field as part of an arithmetic operation in a calculation expression.

Alpha field not allowed: *partial expression*

Alpha fields can only be used in calculation expressions when doing assignments, such as CALC_FLD = ALPHA_FLD.

Alpha fields cannot be right-justified.

You cannot right-justify an alpha field. Alpha fields can only be left-justified or centered.

Ambiguous field reference: *field name*

Two or more fields in the current set of report files have the same name. You must qualify the field name by preceding it with the filename. If more than one structure is assigned to that file and the same field name exists in two or more of the structures, you must also add the structure name (using the format *filename.structure name.field name*).

Ambiguous file/structure reference: *partial expression*

You have specified a field name in your expression that is not unique enough; another field in another file has the same name. Specify the *filename.field name* combination or the *filename.structure name.field name* combination if necessary.

An arrayed field cannot be referenced for subscripting.

You cannot use an arrayed field as a subscript for another arrayed field. You can only use a numeric literal or a nonarrayed numeric field name for the subscript.

Associated structures for file *filename* not found.

Refer to the message "Structure *structure name* not found" on page C-15.

Cannot abandon changes after field modifications.

When modifying a report or page header or footer, if you add, delete, or edit a field, you can no longer abandon your changes.

Cannot add a sort field immediately before a merged field.

By definition, a merged sort field is merged with the sort field that precedes it. You've attempted to insert a new sort field between two sort fields being merged. Insert the new sort field either before the merged sort fields or at the end of the sort fields list.

Cannot change break status on merged field.

You've attempted to set a break on a merged field. By definition, a merged field cannot be a break field; however, the field being merged with *can* be a break field.

Cannot change sort order on merged field.

When sort fields are merged together, their sort order must be the same. To change the sort order for a set of merged sort fields, remove the merged status, change the sort order, and then reset the merge status for all fields in the set.

Cannot generate report: page size *x* too small for *y* lines per page.

You have attempted to generate a report, but the page size (specified in the Miscellaneous input window) is less than the combined number of header, footer, and detail lines. You must increase the page size before you can generate the report.

Cannot merge break field.

You've attempted to merge a break field with the field that precedes it. A merged field cannot be a break field; however, the field being merged with *can* be a break field.

Cannot merge first sort field.

By definition, merging a sort field attaches it to the field that precedes it. Therefore, the first sort field in the list cannot be merged.

Cannot move merged field.

Because a merged sort field is closely associated with the field to which it is merged, it cannot be moved.

Cannot open main repository file.

ReportWriter cannot find either the repository main file specified by the RPSMFIL logical or the **rpsmain.ism** file in either the directory specified by the RPSDAT logical or the current directory. Another possible explanation is that you don't have the necessary system privileges to access this file.

Cannot open report definition file *filename*.

ReportWriter cannot find either the report definition file specified by the RPTRFIL logical or the **reports.rpt** file in either the directory specified by the RPTDAT logical or the current directory. Another possible explanation is that you don't have the necessary system privileges to access this file.

Cannot open ReportWriter window library.

ReportWriter cannot find its window library file in the directory specified by the RPTDAT logical or the current directory. Another possible explanation is that you don't have the necessary system privileges to access this file.

Cannot open repository text file.

ReportWriter cannot find either the repository text file specified by the RPSTFIL logical or the **rpstext.ism** file in either the directory specified by the RPSDAT logical or the current directory. Another possible explanation is that you don't have the necessary system privileges to access this file.

Colon out of place in expression: *partial expression*

A colon is in an invalid location in the expression. Colons are only allowed within a field range specification (to separate the offset from the length) and within a %SUM intrinsic function specification.

Criterion specification is incomplete.

You have not entered all of the required information. When you define a selection criterion or conditional, you must specify the field name and the comparison operator. If other criteria are already defined, you must also specify the connector.

Decimal type expected: *partial expression*

You have specified the %SUM intrinsic function in a calculation expression, but you have not supplied a decimal field or literal for both the *begin* and *end* arguments.

Division by zero attempted.

While the report is being generated, a calculation field's arithmetic expression is attempting to divide by zero, which is undefined. If your expression contains fields that might have data values of 0, use a calculation conditional to evaluate the expression only when the data field is nonzero.

Error occurred while scanning the repository.

A fatal I/O error occurred while ReportWriter was trying to read from the repository. ReportWriter will terminate.

Field *field name* cannot be deleted. It is used in a subtotal access field.

The temporary field you are trying to delete is used in a subtotal access field definition. Before you can delete it, you must remove it from all subtotal access fields that reference it.

Field *field name* is computed after selections and cannot be used here.

The field you are specifying in a selection criterion is a calculation field and has been designated as being computed after the selections are processed. (This means that you have either flagged the calculation field as being forced to occur after selections and sorting, or that the calculation field uses a subtotal access field, which by definition forces it to be computed after selections and sorting.)

Field *field name* is not defined as a break field.

The field you're trying to specify as the break level for a subtotal access field is not defined as a break field. Break fields are sort fields that have been assigned a break status. To use the given field as the break level, you must first select it as a sort field and then set the break status. Press the List selections shortcut to display a list of available break fields.

Field *field name* not defined.

The field name you specified in a calculation expression is invalid. Use the List selections shortcut to display the list of fields that are available from the current report files.

Field *field name* not found in file *filename*.

Refer to the message “Structure *structure name* not found” on page C-15.

Field cannot be deleted. It is printed on a detail line.

The temporary field you’re trying to delete was selected as a field to print. Before you can delete it, you must remove it from the list of fields to print.

Field cannot be deleted. It is printed on a page footer line.

The temporary field you’re trying to delete is used in a page footer. Before you can delete it, you must remove it from the page footer.

Field cannot be deleted. It is printed on a page header line.

The temporary field you’re trying to delete is used in a page header. Before you can delete it, you must remove it from the page header.

Field cannot be deleted. It is printed on a post-break line.

The temporary field you’re trying to delete is used in a post-break print line. Before you can delete it, you must remove it from all post-break print lines.

Field cannot be deleted. It is printed on a pre-break line.

The temporary field you’re trying to delete is used in a pre-break print line. Before you can delete it, you must remove it from all pre-break print lines.

Field cannot be deleted. It is printed on a report footer line.

The temporary field you’re trying to delete is used in a report footer. Before you can delete it, you must remove it from the report footer.

Field cannot be deleted. It is printed on a report header line.

The temporary field you’re trying to delete is used in a report header. Before you can delete it, you must remove it from the report header.

Field cannot be deleted. It is used as a range offset or length.

The temporary field you’re trying to delete is used as the range offset or length for another field. This ranged field is either specified as a field to print, used in a calculation expression, used in a selection statement, used as a sort field, and/or used in a field to print, calculation, or break line conditional. Before you can delete the field, you must remove all such range references.

Field cannot be deleted. It is used as a sort field.

The temporary field you’re trying to delete was selected as a field on which to sort. Before you can delete it, you must remove it from the list of sort fields.

Field cannot be deleted. It is used as a subscript.

The temporary field you're trying to delete is used as a subscript for an arrayed field. This arrayed field is either specified as a field to print, used in a calculation expression, used in a selection statement, used as a sort field, and/or used in a fields to print, calculation, or break line conditional. Before you can delete it, you must remove the subscripted field from all such references.

Field cannot be deleted. It is used in a calculation expression.

The temporary field you're trying to delete is used in a calculation expression. Before you can delete it, you must remove it from any and all calculations.

Field cannot be deleted. It is used in a criterion definition.

The temporary field you're trying to delete is used in a selection criterion or conditional. Before you can delete it, you must remove it from any and all selection criteria and conditionals.

Field size cannot be changed. It is selected to print.

You've attempted to change the size of a temporary field that is selected to print on a detail line, break line, or header or footer line. You must remove the field from all print lines before changing its size.

Field used as break level in subtotal field *field name*. Cannot clear break.

You've tried to remove the break status from a sort field that is used in the definition of a subtotal access field. You must remove the field from all subtotal access field definitions before you can clear its break status.

Field used with %BRKCNT in calculation field *field name*. Cannot clear break.

You've tried to remove the break status from a sort field that is used in the %BRKCNT intrinsic within a calculation field. You must remove the field from all %BRKCNT intrinsics before you can clear its break status.

Fields for structure *structure name* not found.

Refer to the message "Structure *structure name* not found" on page C-15.

File *filename* not found.

Refer to the message "Structure *structure name* not found" on page C-15.

File cannot be deleted. *Filename* is linked to it.

This error message is displayed if the following scenario occurs. File A is related to File B in the repository. Both files have been selected as files to read. You are currently trying to delete File A. Because File A is required to access File B, you cannot delete it. Similarly, suppose File A is related to File B, and File B is related to File C. Since File A depends on File B to access File C, you cannot delete File B before deleting File C.

File cannot be deleted. Field *field name* is printed on a detail line.

A field from this file was selected as a field to print. Before you can delete the file, you must remove the field from the list of fields to print.

File cannot be deleted. Field *field name* is printed on a page footer line.

A field from this file is used in a page footer. Before you can delete the file, you must remove the field from the page footer.

File cannot be deleted. Field *field name* is printed on a page header line.

A field from this file is used in a page header. Before you can delete the file, you must remove the field from the page header.

File cannot be deleted. Field *field name* is printed on a post-break line.

A field from this file was selected as a field to print on a post-break line. Before you can delete the file, you must remove the field from all post-break print lines.

File cannot be deleted. Field *field name* is printed on a pre-break line.

A field from this file was selected as a field to print on a pre-break line. Before you can delete the file, you must remove the field from all pre-break print lines.

File cannot be deleted. Field *field name* is printed on a report footer line.

A field from this file is used in a report footer. Before you can delete the file, you must remove the field from the report footer.

File cannot be deleted. Field *field name* is printed on a report header line.

A field from this file is used in a report header. Before you can delete the file, you must remove the field from the report header.

File cannot be deleted. Field *field name* is used as a range offset or length.

A field from this file is used as the range offset or length for another field. This ranged field is either specified as a field to print, used in a calculation expression, used in a selection statement, used as a sort field, and/or used in a fields to print, calculation, or break line conditional. Before you can delete the file, you must remove all such range references.

File cannot be deleted. Field *field name* is used as a sort field.

A field from this file was selected as a field on which to sort. Before you can delete the file, you must remove the field from the list of fields to sort.

File cannot be deleted. Field *field name* is used as a subscript.

A field from this file is used as a subscript for an arrayed field. This arrayed field is either a field to print, used in a calculation expression, used in a selection statement, used as a sort field, and/or used in a fields to print, break line, or calculation conditional. Before you can delete the file, you must remove the field being used as a subscript from all such references.

File cannot be deleted. Field *field name* is used in a calculation expression.

A field from this file is used in a calculation expression. Before you can delete the file, you must remove the field from any and all calculations.

File cannot be deleted. Field *field name* is used in a criterion definition.

A field from this file is used in a selection criterion or conditional. Before you can delete the file, you must remove the field from any and all selection criteria and conditionals.

File cannot be deleted. Field *field name* is used in a subtotal access field.

A field from this file is used in a subtotal access field definition. Before you can delete the file, you must remove the field from all subtotal access fields that reference it.

File cannot be deleted. It is used in a multiple projection.

You've attempted to delete a secondary file that is associated with another secondary file (its sister file) in a multiple projection. You must set the multiple projection status of the sister file to "Together" before deleting the current file.

File cannot be deleted. Structure *structure name* is referenced.

This error message is displayed if the following scenario occurs. File A is related to both File B and File C in the repository. File A's structure defines a key that contains an external segment from File B's structure. File A then uses this key for the relation to File C. You are currently trying to delete File B. Because File B is essential for the relation to File C, you cannot delete File B before deleting File C.

File/structure not found: *partial expression*

You've specified an invalid file definition or file/structure definition name as the argument in the %GOTREC intrinsic function. View your list of Files to Read for the valid file/structure definition names.

File/structure specification expected: *partial expression*

You've specified the %GOTREC intrinsic function, but a file/structure specification was not found within the parentheses.

Function argument expected: *partial expression*

You've specified the %SUM or %GOTREC intrinsic function without specifying the proper argument(s). See %SUM on page 8-13 and %GOTREC on page 8-14 for information on the required arguments.

Key *key name* not found for structure *structure name*.

Refer to the message "Structure *structure name* not found" on page C-15.

Key of reference *key name* not found for structure *structure name*.

Refer to the message "Structure *structure name* not found" on page C-15.

Invalid comparison field type.

The data type of the field you've specified as the compare value in a selection criteria statement does not match that of the field being compared.

Invalid date specification.

You've entered an invalid date as a comparison value for a date field.

Invalid field *field name* referenced.

You've specified an invalid field name in a print or calculation conditional, as a subscript for an arrayed field, or as a field range offset or length.

Invalid name link relation.

You've skipped a segment definition when modifying the NAME_LINK_KEY in the input window. Segment definitions must be contiguous.

Invalid precision specification.

The precision value you specified is larger than the length of the field. (The number of decimal places must be less than or equal to the field length.)

Invalid report version *version* cannot be loaded with the current ReportWriter.

You've attempted to access a report that has not been converted to the current revision level. Refer to your online release notes for information on converting existing ReportWriter reports.

Invalid selection criteria specification.

Your selection criteria contain an invalid wildcard specification. For example, entering one or more asterisks (*) and nothing else would be considered invalid.

Invalid subscript field type.

You can only use numeric type data fields as subscripts for an arrayed field. (Date and time fields are also decimal fields; however, their use as subscripts is rare.)

Invalid temporary field name.

You've specified a field name that is invalid. Valid field names begin with a letter and can contain letters, numbers, underscores, and dollar signs.

Invalid time specified.

You've either specified a negative value in a question field while a report is being generated, or your data file contains a negative time value. Time values must be greater than zero.

Invalid value.

In your selection criteria, you've attempted to compare a numeric field with an alphanumeric literal.

Linked structure *structure name* not found.

Refer to the message "Structure *structure name* not found" on page C-15.

Maximum number of break line fields (99) has been selected.

You've tried to select more than 99 fields to print on your pre-break or post-break print line. You cannot select any more fields to print until you delete one or more fields from your list of fields to print.

Maximum number of conditions (5) has been defined.

You've tried to define more than five conditions. You cannot add any more conditions to the conditional until you delete one or more existing conditions.

Maximum number of detail lines (10) has been defined.

You've tried to create more than 10 detail lines. You cannot create any more lines until you delete one or more existing lines.

Maximum number of fields to print (990) has been selected.

You've tried to select more than 990 fields to print. You cannot select any more fields to print until you delete one or more fields from your list of fields to print.

Maximum number of fields to print per line (99) has been selected.

You've tried to select more than 99 fields to print on the current detail line. You cannot select any more fields to print until you delete one or more fields from the current line.

Maximum number of files (50) has been selected.

You’ve tried to select more than 50 files to read. You cannot select any more files until you delete one or more files from your list.

Maximum number of individual temporary fields (99) has been defined.

You’ve tried to create more than 99 calculation, question, text, environment, or subtotal access fields. You cannot create any more fields of that type until you delete one or more existing ones.

Maximum number of page header lines (10) has been defined.

You’ve tried to create a new detail line, but the number of detail lines in combination with the number of split headers has reached the allowed maximum. For example, if you have five detail lines, all of which have a field with a split heading, you have a total of 10 header lines. ReportWriter tries to maintain a one-to-one correspondence between header and detail lines. To add additional detail lines, edit the page header in some way. Doing so flags the header as “dirty,” and ReportWriter no longer tries to maintain the one-to-one correspondence between header and detail lines.

Maximum number of ranged fields (300) has been defined.

Each time a range is specified for a field, the number of ranged fields is increased by 1. You’ve reached the maximum number of ranged fields. You cannot range any more fields in the report until you delete one or more existing ranged fields.

Maximum number of selections (25) has been defined.

You’ve tried to define more than 25 selections. You cannot define any more selections until you delete one or more existing selections.

Maximum number of sort fields (10) has been selected.

You’ve tried to select more than 10 sort fields. You cannot select any more sort fields until you delete one or more fields from your list of fields to sort.

Maximum number of subscripted fields (300) has been defined.

Each time an arrayed field is used in a calculation, in a selection, in a conditional, or as a field to print, the count of subscripted fields is increased by 1. You’ve reached the maximum number of subscripted fields. You cannot subscript any more arrayed fields in the report until you delete one or more existing usages.

Maximum report width (255 columns) has been reached.

The total length of the fields you’ve selected to print on the current line is longer than 255 columns. You cannot select any more fields to be printed on that line until you delete one or more fields from the line.

Merged sort fields must be the same type and length.

You’ve selected a sort field to be merged with the preceding one. When merging two sort fields, the fields must be the same data type and length.

Merged sort fields must have the same sort order.

You’ve selected a sort field to be merged with the preceding sort field, but they are not sorted in the same order.

Missing “)” or “]” in expression: *partial expression*

ReportWriter has found an open parenthesis (( ) without a matching close parenthesis ( )) or a left square bracket ([ ) without a matching right square bracket ( ]) in the expression.

Missing operand: *partial expression*

You’ve specified an operator without the corresponding operand(s).

Missing operator in expression: *partial expression*

You’ve specified two operands in the expression without specifying an operator. Select the List operators menu entry to display a list of valid operators.

Multiple projections not possible for the relation to this file.

When selecting a file on which to perform multiple projections, the relationship defined by the other file (the “From” file) must be one to many. In other words, the key in this file used in the relation must allow duplicates.

No break fields defined.

You’ve selected the List selections function while on the Break level field of a subtotal access field definition. This option attempts to display a list of sort fields that are defined as break fields for the current report. Currently, no sort fields have their break status set. You must have at least one break field defined before you can access a subtotal for a level other than the entire report.

No enumerated information found. Field will be processed as numeric.

A field has been defined as enumerated (E), but the enumerated information cannot be found in the repository. Rather than the numeric value in the field being used as an index to a text string, the numeric value will be processed explicitly.

No files have been defined in the repository.

ReportWriter obtains its list of available files from Repository. You cannot create a report in ReportWriter unless one or more files is defined in the repository and all files have a structure assigned to them. Currently, either no files are defined or the defined files do not have structures assigned to them.

No input files have been defined for this report.

You must select one or more report files before ReportWriter can perform the requested function. Select the Files to read function.

No name link relations available.

You have selected Add file via name link from the Files to read function and one of the following situations is true: the Repository cross-reference file **rpsxref.ism** (or the file pointed to by RPSXFIL) is not found, or the cross-reference file does exist, but it doesn't contain any relations based on field name links.

No previous file at the same read level.

When you select a file on which to perform multiple projections, there must already be at least one file to read on the same level. "Same level" refers to the levels of the tree in a hierarchical relationship. For example, if File A relates to File B, File A relates to File C, and File C relates to File D, files B and C are at the same level, while File D is not.

Null value not allowed in expression: *partial expression*

One of the following error conditions is present in the calculation expression: no expression was entered, no value was specified between parentheses or brackets, or an operator was followed by either nothing or a close parenthesis.

Numeric fields cannot be centered.

You cannot center a numeric field. Numeric fields can only be left- or right-justified.

Operator (*operator*) not allowed with alpha operand.

You've specified a mathematical operator in conjunction with an alpha operand. Alpha operands can only be used in calculation fields to assign their value to a temporary field.

Percent sign out of place: *partial expression*

You've specified the intrinsic function character ("%") in an unexpected location in the calculation field expression.

Question fields cannot inherit attributes from an integer field.

You have attempted to inherit the characteristics of a question field from a field whose type is integer. This is not allowed.

Quoted string encountered in expression: *partial expression*

You cannot use quoted strings in calculation expressions that perform numeric operations.

Range offset plus length *literal* must be within the field size.

The range offset and length values must be equal to or less than the size of the field, and the range offset plus the length must be less than or equal to the size of the field.

Range specification is incomplete.

You have not entered all of the required information. When you range a field, you must specify the range offset and the range length.

Range value *literal* must be in the range 1 to 99.

The largest field size that can be accessed is 99 characters. Therefore, the value specified for the range length must be between 1 and 99, inclusive.

Relation *relation name* is not for structure *structure name*.

Refer to the message "Structure *structure name* not found" on page C-15.

Relation linkage corrupted. Report cannot be loaded.

Refer to the message "Structure *structure name* not found" on page C-15.

Relation records for *structure name* not found.

Refer to the message "Structure *structure name* not found" on page C-15.

Report cannot be deleted. It is being used by another user.

Only one user can load or delete a report at a time, and someone else is currently using the requested report.

Report cannot be saved into an existing report.

When you modify an existing report, you can save it (copy it) under another name. The name you have specified is the name of an existing report. Enter another name.

Report cannot be saved. It is being used by another user.

Only one user can load or delete a report at a time, and someone else is currently using the requested report.

Report control structure overflow.

The report you're trying to generate is too complex for ReportWriter. Although you haven't exceeded any individual program maximums, the amount of memory required to store the current report definition is too extensive.

Report is incomplete. Please select fields to print.

You cannot display the current report until you have selected fields to print using the Fields to print function, selected pre- or post-break fields to print, or selected a field to print in a header or footer. (If you have not selected your report files, you must do that first.)

Report not found.

You've attempted an operation on a report that cannot be found in the current report definition file. Press the List selections shortcut to display a list of available reports.

Selected field name *field name* will not fit in the expression.

By adding this field, you will exceed the maximum size of an expression (150 characters). Therefore, you cannot add this field. If any of the field names used in the expression are prefixed with their file and/or structure names, and if these field names are unambiguous, you can remove the prefixes to decrease the size of the expression.

Sort field cannot be deleted. It is merged with the next field.

Merged sort fields must be deleted in a "last in, first out" order. In other words, you must delete them from the bottom up.

Structure *structure name* not found.

The report definition and the repository are out of sync. This could have happened in one of two ways: the repository was modified after a report definition was saved, or, while a report was being designed, the repository was modified.

Once a repository is defined, it should not be modified! Modifications to a repository that is being used by ReportWriter can invalidate report definitions.

Structure *structure name* not found. Cannot load fields.

Refer to the message "Structure *structure name* not found" on page C-15.

Subscript specification is incomplete.

You have not entered all the required information. When you use an arrayed field, you must specify a subscript value for each dimension of the array.

Subscript value *literal* must be in the range 1 to *max array elements*.

The literal value used to specify an arrayed field subscript is outside the bounds of the array.

%SUM requires array beg:end: *partial expression*

You have specified the %SUM intrinsic function but have not included the required arguments. You must specify the name of a one-dimensional array and two numeric fields or literals indicating the first and last elements to include in the sum.

Tagged field not found.

Refer to the message “Structure *structure name* not found” on page C-15.

Temporary field *field name* already exists. Please enter another name.

You entered the name of an existing temporary field. Temporary field names must be unique.

Temporary field definition is incomplete.

You have not entered all of the required information. You must specify the following information for each type of temporary field:

Temporary field	Required information
Calculation	Name, type, length, and expression
Environment	Name, type, and length
Question	Name, type, length, and prompt
Subtotal access	Name, field to total, and break level
Text	Name

Too many close parentheses in expression: *partial expression*

The number of close parentheses ( )) exceeds the number of open parentheses (( ).

Too many lines to view in the report format. It has been truncated.

The report format that you view is divided into three sections: report header, report page, and report footer. The report page section includes the page header, date and page number line, data lines, pre- and post-break lines, and page footer. The maximum viewable size of each section is 16 lines. Because the size of your report page exceeds 16 lines, this section of the report format that you’re viewing has been truncated.

Too many total fields. Program will terminate.

The number of fields to be totaled requires more memory than is currently available.

Unrecognized character in expression: *partial expression*

The expression contains an invalid character. An expression can only contain the following characters: all numeric and alphanumeric characters (alphanumeric characters that aren’t in quotation marks are assumed to be field specifications); operators (**+**, **–**, *****, and **/**); parentheses, brackets, and commas (for array subscripting); and quoted strings for alpha expressions.

Unrecognized intrinsic function reference: *partial expression*

You've specified the intrinsic function character (%) without supplying a valid intrinsic function name.

Wrong number of dimensions specified: *partial expression*

An arrayed field has either no subscript specifications or the incorrect number of subscript values. The number of array subscript values that you specify must be equal to the number of field dimensions.

D

Data Formats

The format specification is a sequence of characters that depicts how the contents of a field will look. The characters listed in the table below have special meanings. Other characters that appear in a format string are inserted directly into the field.

If the format size (number of data characters) is larger than the field size, an alpha field is left-justified and the format string truncated, while a decimal field is right-justified and the format string truncated. If the format size is smaller than the field size, the data is truncated.

Character	Meaning
@	Represents a single alpha character.
X	Represents a single digit. It causes a digit from the source data to be inserted into the specified position in the field. Digits are extracted from the source data beginning at the right and continuing to the left; the rightmost X in the format is loaded with the rightmost digit in the source data. If there are more X characters than there are significant digits in the source data, the leftmost X positions are loaded with zeros.
Z	Represents a single digit. Z differs from X if there are more Z characters than significant digits to be transferred from the source data. The Z position is loaded with a blank if no X character or period appears to the left of it in the format. If a period appears to the left but not to the right, the Z position is loaded with a zero. A zero is also loaded if an X appears to the left of the Z.
*	Represents a digit position. The * is similar to an X, except that when there are no more significant digits, the position is loaded with an asterisk rather than a zero.
money sign	Represents a digit position. If there are more significant digits to be transferred, the position is loaded with the next digit. If no more significant digits remain to be transferred, the position is loaded with a blank. The rightmost money character is changed to a dollar sign when field loading is concluded. The default money sign is “$”, but you can use the Synergy DBL MONEY subroutine to change it to any character. See MONEY in the “System-Supplied Subroutines and Functions” chapter of the *Synergy DBL Language Reference Manual*.

Character	Meaning
–	Causes a minus sign to be inserted at the corresponding position in the field if the – is the first or last character of a format. If the source data is positive, a blank will be inserted. If the – is used within the format, a hyphen is inserted at the corresponding position in the field.
.	Causes a decimal point to be inserted at the corresponding position in the field. In addition, any Z that appears to the right of the decimal point is treated as an X.
,	Causes a comma to be inserted at the corresponding destination position, but only if one of the following conditions is true: ▸ More significant source digits remain to be transferred. ▸ There is an X character to the left of the comma.

Using European numeric formatting

The RPTEURO environment variable tells ReportWriter to apply European formatting conventions, instead of American conventions, to numeric values. If RPTEURO is set to any value, commas are used as decimal points, and periods are used as separators. See RPTEURO on page F-9 for more information.

E

Distributed Shortcuts

This appendix lists of some of the ReportWriter shortcuts as they appear in your original distribution.

Distributed Shortcuts

Menu entry	VT-style shortcut	PC-style shortcut
Abandon	CTRL+A	CTRL+A
Add field (file, selection, sort field)	INS	INS
Add field (in header or footer)	F7	F7
Add fields (sort fields)	CTRL+K	CTRL+K
Add line	CTRL+N	CTRL+N
Arrange field position	CTRL+N	CTRL+N
Beginning of line	CTRL+B	CTRL+B
Change break status	CTRL+B	CTRL+B
Change field attributes	CTRL+E	CTRL+E
Change ranging	CTRL+W	CTRL+W
Change subscripting	CTRL+H	CTRL+H
Clear break status	F12	F12
Clear field	CTRL+X	CTRL+X
Command mode	CTRL+F	CTRL+F
Copy selection	F18	SHIFT+F8
Create new line	F13	SHIFT+F3
Delete character	CTRL+X	CTRL+X
Delete current line	CTRL+V	CTRL+V

Distributed Shortcuts (Continued)		
Menu entry	**VT-style shortcut**	**PC-style shortcut**
Delete current (detail) line	F14	SHIFT+F4
Delete field (file, selection, sort field)	REM	DEL
Delete word	CTRL+U	CTRL+U
Display report	PF2	F2
Down one line	↓	↓
Edit	RETURN	RETURN
End of line	CTRL+V	CTRL+V
Examine relation	CTRL+E	CTRL+E
Exit	PF4	F4
Fields to create	F9	F9
Fields to print	F10	F10
Find	FIND	HOME
First entry	CTRL+F	CTRL+F
Help	PF1	F1
Insert/overstrike mode	CTRL+K	CTRL+K
Last entry	CTRL+L	CTRL+L
Left one character	←	←
Left 10 characters	CTRL+L	CTRL+L
List selections	F7	F7
Move left	←	←
Move right	→	→
Next entry	↓	↓
Next line	TAB	TAB
Next page	PGDN	PGDN

Distributed Shortcuts (Continued)		
Menu entry	**VT-style shortcut**	**PC-style shortcut**
Next report file	TAB	TAB
Post-break line	F14	SHIFT+F4
Pre-break line	F13	SHIFT+F3
Previous entry	↑	↑
Previous line	F8	F8
Previous page	PGUP	PGUP
Previous report file	F7	F7
Reorder fields (selections, sort fields)	F17	SHIFT+F7
Reset field	CTRL+U	CTRL+U
Right one character	→	→
Right 10 characters	TAB	TAB
Save loaded report	F20	SHIFT+F10
Scroll left	←	←
Scroll right	→	→
Show field names	F8	F8
Specify conditions	CTRL+G	CTRL+G
Specify line conditions	F19	SHIFT+F9
Strip trailing blanks	CTRL+B	CTRL+B
Toggle data line/break lines	F9	F9
Toggle view	CTRL+V	CTRL+V
Total field	CTRL+J	CTRL+J
View	PF3	F3

F

Environment Variables

RPSDAT – Directory containing Repository data files

WT WN U V

The RPSDAT environment variable defines the directory where Repository data files are located.

Value

The path, including the device, for the directory that contains the Repository data files.

Discussion

The default directory for Repository data files is the RPSDAT subdirectory below the directory where the Repository was installed. The RPSDAT directory contains the default repository files, **rpsmain.ism** and **rpstext.ism**, and the optional cross-reference file, **rpsxref.ism**.

Setting location

- On Windows, the [synergy], [dbr], or [*myprog*] section of **synergy.ini** (where *myprog* is any **.dbr** file).
- On UNIX, the **setsde** file.
- On OpenVMS, the **SYNERGY_STARTUP.COM** file (or **ACTIVATE_SDE.COM** for alternate installations).

Examples

On OpenVMS,

```
$ DEFINE/SYS/EXEC RPSDAT DKA0:[SYNERGY.RPS.RPSDAT]
```

RPSMFIL – Repository main file

WT	WN	U	V

The RPSMFIL environment variable specifies the full path and filename of the Repository main file.

Value

The full path and filename of the Repository main file.

Discussion

If the environment variable RPSMFIL is not set, Repository looks for the **rpsmain.ism** file in the path specified by RPSDAT. If RPSDAT is not set, Repository looks for **rpsmain.ism** in the current directory.

Setting location

The environment. On Windows, this environment variable can also be set in the [synergy], [dbr], or [*myprog*] section of **synergy.ini** (where *myprog* is any **.dbr** file).

Examples

On Windows, if set in the **synergy.ini** file,

```
[synergy]
RPSMFIL=c:\home\rpsmain.ism
```

RPSTFIL – Repository text file

WT	WN	U	V

The RPSTFIL environment variable specifies the full path and filename of the Repository text file.

Value

The full path and filename of the Repository text file.

Discussion

If the environment variable RPSTFIL is not set, Repository looks for the **rpstext.ism** file in the path specified by RPSDAT. If RPSDAT is not set, Repository looks for **rpstext.ism** in the current directory.

Setting location

The environment. On Windows, this environment variable can also be set in the [synergy], [dbr], or [*myprog*] section of **synergy.ini** (where *myprog* is any **.dbr** file).

Examples

On UNIX,

```
RPSTFIL=/usr/rpstext.ism    ;export RPSTFIL
```

RPSXFIL – Repository cross-reference file

WT	WN	U	V

The RPSXFIL environment variable specifies the full path and filename of the Repository cross-reference file.

Value

The full path and filename of the Repository cross-reference file.

Discussion

If the environment variable RPSXFIL is not set, Repository looks for the **rpsxref.ism** file in the path specified by RPSDAT. If RPSDAT is not set, Repository looks for **rpsxref.ism** in the current directory.

See "Generating a Cross-Reference File" in the "Utility Functions" chapter in the *Repository User's Guide* for more information about Repository cross-reference files.

Setting location

The environment. On Windows, this environment variable can also be set in the [synergy], [dbr], or [*myprog*] section of **synergy.ini** (where *myprog* is any **.dbr** file).

Examples

On OpenVMS,

```
$ DEFINE RPSXFIL DKA0:[REPORTS]RPSXREF.ISM
```

RPT – ReportWriter directory

WT	WN	U	V

The RPT environment variable specifies the directory that contains your ReportWriter distribution. It is required for normal operation of the ReportWriter program and utilities.

Value

The path, including the device, for the directory that contains the ReportWriter distribution files.

Setting location

- On Windows, the [synergy], [dbr], or [*myprog*] section of **synergy.ini** (where *myprog* is any **.dbr** file).
- On UNIX, the **setsde** file.
- On OpenVMS, the **SYNERGY_STARTUP.COM** file (or **ACTIVATE_SDE.COM** for alternate installations).

Examples

On UNIX,

```
RPT=/usr/synergy/rpt      ; export RPT
```

RPTDAT – Directory containing ReportWriter data files

WT	WN	U	V

The RPTDAT environment variable specifies the directory where ReportWriter data files are located.

Value

The path, including the device, for the directory that contains the ReportWriter data files.

Discussion

The default directory for ReportWriter data files is the RPTDAT subdirectory below the directory where ReportWriter was installed. The RPTDAT directory contains the default report definition file, **reports.rpt**, and the window library file, **rptctl.ism**.

Setting location

- On Windows, the [synergy], [dbr], or [*myprog*] section of **synergy.ini** (where *myprog* is any **.dbr** file).
- On UNIX, the **setsde** file.
- On OpenVMS, the **SYNERGY_STARTUP.COM** file (or **ACTIVATE_SDE.COM** for alternate installations).

Examples

On OpenVMS,

```
$ DEFINE RPTDAT DKA0:[SYNERGY.RPT.RPTDAT]
```

RPTDATE – Date input order

WT	WN	U	V

The RPTDATE environment variable specifies the input order for dates in ReportWriter. Setting RPTDAT affects the default date order that is used when dates are entered in ReportWriter input fields, because it changes the UI Toolkit global variable **g_date_order**. Consequently, if you are using ReportWriter's external subroutine interface, it will affect your application.

Value

One of the following values:

1	DDMMYY
2	YYMMDD

Discussion

If RPTDATE is not set, the default input order—MMDDYY—is used.

If you are accessing ReportWriter from your application by external subroutine interface, using RPTDAT can affect the value of **g_date_order** in your application.

Setting location

The environment. On Windows, this environment variable can also be set in the [synergy], [dbr], or [*myprog*] section of **synergy.ini** (where *myprog* is any **.dbr** file).

Examples

On OpenVMS,

```
$ DEFINE RPTDATE 2
```

RPTEURO – European formatting

WT	WN	U	V

The RPTEURO environment variable specifies that European formatting conventions will be applied to numeric values.

Value

One of the following values:

0	MMDDYY or MMDDYYYY format
1	DDMMYY or DDMMYYYY format
2	YYMMDD or YYYYMMDD format

All other values are treated as 0.

Discussion

If RPTEURO is set to any value, ReportWriter uses commas as decimal points and periods as separators in numeric values.

When using RPTEURO, you must place a space between the literals and the comma in a subscript specification. This rule applies to subscripts specified within ReportWriter input fields and report schemas. For example:

```
"FIELD[1 , 2]"
```

We recommend using the Synergy DBL LOCALIZE subroutine instead of RPTEURO in conjunction with ReportWriter's external subroutine interface. See LOCALIZE in the "System-Supplied Subroutines and Functions" chapter of the *Synergy DBL Language Reference Manual*.

If the date is a period date, RPTEURO is ignored.

Setting location

The environment. On Windows, this environment variable can also be set in the [synergy], [dbr], or [*myprog*] section of **synergy.ini** (where *myprog* is any **.dbr** file).

Examples

On UNIX,

```
RPTEURO=1     ; export RPTEURO
```

RPTLIB – ReportWriter header file and shared library location

WT	WN	U	V

The RPTLIB environment variable specifies the location of the ReportWriter header file (**reports.def**) and shared library (**synrpt.elb**, or **SYNRPT.EXE** on OpenVMS).

Value

The directory in which the ReportWriter header file and shared library reside.

Discussion

RPTLIB should point to the rpt\lib directory under the directory where Synergy/DE was installed.

Setting location

- On Windows, the [synergy] section of **synergy.ini**.
- On UNIX, the **setsde** file.
- On OpenVMS, the **SYNERGY_STARTUP.COM** file (or the **ACTIVATE_SDE.COM** file for alternate installations).

Examples

On UNIX,

```
RPTLIB=/synergy/rpt/lib    ;export RPTLIB
```

RPTRFIL – ReportWriter report definition file

WT	WN	U	V

The RPTRFIL environment variable specifies the full path and filename of the ReportWriter report definition file.

Value

The full path and filename of the ReportWriter report definition file.

Discussion

If the environment variable RPTRFIL is not set, ReportWriter looks for the **reports.rpt** file in the path specified by RPTDAT. If RPTDAT is not set, ReportWriter looks for **reports.rpt** in the current directory.

Setting location

The environment. On Windows, this environment variable can also be set in the [synergy], [dbr], or [*myprog*] section of **synergy.ini** (where *myprog* is any **.dbr** file).

Examples

On UNIX,

```
RPTRFIL=/usr/reports.ism    ; export RPTRFIL
```

RPTTUT – ReportWriter tutorial data directory

WT WN U V

The RPTTUT environment variable specifies the directory that contains the ReportWriter tutorial files.

Value

The path for the directory in which the ReportWriter tutorial files are located.

Discussion

The repository associated with the tutorial uses RPTTUT to locate the example data files. Do not modify this setting. It is unset automatically when you exit ReportWriter.

Setting location

- On Windows, in the [rpt] section of **synergy.ini**. (This is set when ReportWriter is installed.)
- On UNIX, by **tutorial**.
- On OpenVMS, by **@tutorial.com**.

Examples

On Windows,

```
[rpt]
RPTTUT=%SYNERGYDE%\rpt\tutor
```

RPTUSR – ReportWriter argument string

WT	WN	U	V

The RPTUSR environment variable specifies the argument string used when chaining or spawning ReportWriter.

Value

See “Spawning and Chaining to ReportWriter” on page 16-24 for the syntax and a discussion of the ReportWriter argument string.

Setting location

The environment. On Windows, this environment variable can also be set in the [synergy], [dbr], or [*myprog*] section of **synergy.ini** (where *myprog* is any **.dbr** file).

Examples

On UNIX,

```
RPTUSR="RPT:reports.rpt|myprog.dbr"    ; export RPTUSR
```

RWUSRLIB – ELB for user-overloadable routines

WT	WN	U	V

The RWUSRLIB environment variable specifies the ELB in which user-overloadable subroutines and functions reside.

Value

The full path and filename of the ELB or shared image in which the user-overloadable routines reside.

Discussion

On OpenVMS, ReportWriter calls OPENELB on the shared image that is specified with RWUSRLIB before attempting to call the routines.

If you are using the external subroutine interface, you can pass the ELB name as a parameter to %RW_INIT instead of specifying it with RWUSRLIB.

See "Using User-Overloadable Routines" on page 15-3 for information on writing and using user-overloadable routines with ReportWriter.

Setting location

The environment. On Windows, this environment variable can also be set in the [synergy], [dbr], or [*myprog*] section of **synergy.ini** (where *myprog* is any **.dbr** file).

Examples

On Windows, if set in the **synergy.ini** file,

```
[synergy]
RWUSRLIB=c:\Apps\elbs\myUserRoutines.elb
```

SYNCENTURY – Two-digit year used to determine default century

WT		U	V

The SYNCENTURY environment variable defines a sliding window for Synergy/DE default century methods by specifying a two-digit year to be used as a defining point. Years prior to the specified year use one century, while years the same as or later than the specified year use a different century.

Value

A two-digit year in the range 00–99.

Discussion

ReportWriter enables you to define a method that is called whenever a default century is required with the %RW_CENTURY_METHOD function (see page 15-34). If no century method is registered, but SYNCENTURY is defined, an internal century method is called. If SYNCENTURY is not defined or is invalid, ReportWriter uses 19 as the century for all dates stored in YYMMDD, YYJJJ, and YYPP formats.

The default century method used by ReportWriter provides a sliding window in which the default century for each side of the window is determined by SYNCENTURY and the current year:

- If the current two-digit year falls between 0 and SYNCENTURY, years between 0 and SYNCENTURY use the current system century, while years between SYNCENTURY (inclusive) and 99 use the previous century.
- If the current two-digit year falls between SYNCENTURY (inclusive) and 99, years between SYNCENTURY (inclusive) and 99 use the current system century, while years between 0 and SYNCENTURY use the next century.

The table below shows the translated four-digit year when SYNCENTURY is set to 50:

Current year	Year entered by user	Translated four-digit year
1998	0	2000
1998	49	2049
1998	50	1950
1998	99	1999
2000	0	2000

Current year	Year entered by user	Translated four-digit year
2000	49	2049
2000	50	1950
2000	99	1999

SYNCENTURY is also used by UI Toolkit and *xf*ODBC.

Selection and sorting of dates in YYMMDD, YYJJJ, or YYPP formats is not optimized if SYNCENTURY is set or if %RW_CENTURY_METHOD is specified.

Setting location

The environment. On Windows, this environment variable can also be set in the [synergy], [dbr], or [*myprog*] section of **synergy.ini** (where *myprog* is any **.dbr** file).

See also

%RW_CENTURY_METHOD on page 15-34.

Examples

On Windows, if set in the **synergy.ini** file,

```
[synergy]
SYNCENTURY=50
```

SYNRPT – Location of the SYNRPT.EXE shared image

			V

The SYNRPT logical points to **RPTLIB:SYNRPT.EXE** for use in activating the shared image that contains the ReportWriter shared library.

Value

The full path and filename of the **SYNRPT.EXE** shared image.

Setting location

The **SYNERGY_STARTUP.COM** file found in SYS$MANAGER (or **ACTIVATE_SDE.COM** for alternate installations).

Examples

```
$ DEFINE/SYS/EXEC SYNRPT RPTLIB:SYNRPT.EXE
```

Glossary

access key
Represents a true key in a data file and is used to specify relationships between files.

arrayed field
A field definition from the repository that represents a group of fields, each of the same size and data type. Arrayed fields can have between one and four dimensions.

blank strip
To remove all trailing blanks from a field, bringing the next field closer. ReportWriter enables you to specify a strip character to be displayed between the blank-stripped field and the field that follows it.

break field
A sort field on which a report break and possibly a page break is set in ReportWriter.

break summary line
A line in your report that contains the break description, the number of records printed in the break set, and, if one or more fields are being totaled, the word "total" and the subtotal amounts.

byte
The smallest area of memory addressable by ReportWriter. One byte contains one character.

calculation field
A temporary field that describes a mathematical expression whose result will be stored in that field.

cross-reference file
A file that contains name link associations, which ReportWriter uses to provide access to related files and data structures.

conditional
Defines the fields and conditions that ReportWriter will use and compare in determining whether to print a field for a particular report record, which of two calculations to perform for a given calculation field, or whether or not to print a pre- or post-break line or field.

detail line
One line of a report record. A report record can have a maximum of 10 detail lines.

enumerated field
A field that designates a numeric value that is stored in a file and that represents an alpha string. It is specified in conjunction with a selection list or an allow list in Repository.

environment field

A temporary field used to obtain static data from the environment. It can be used to retrieve shell or operating system values, global control record information, or any other data that remains static for the duration of the report.

external key

Enables a key to be composed of a segment from the defining structure and one or more fields from another structure or structures.

field

A named area of computer memory used to store a specific type of data.

field header

The text that appears at the top of the column for a field (if you choose to print page headers) on each page of a report. The default field header is the header that was specified during the field definition phase of the Repository application.

file

An area of the disk where groups of similar data (called records) are stored.

fixed font

A font in which every character has the same width.

foreign key

A field on a file that is a key on another file. It can be thought of as “pointing” straight to a particular record on that other file.

format

The way a field will be displayed in your report. Global formats are defined in the repository and are available for use by any field in ReportWriter. Predefined formats for date and time fields are built into every repository.

global format

See format.

information line

A single line at the bottom of the screen body that ReportWriter uses to display messages and general information.

inherited question field

A question field whose characteristics were copied from an existing field.

input window

A window that can contain text, input fields, and buttons, in which the user enters information.

intrinsic function

An internally defined function that can be used in a calculation expression.

key

A field in a file that may be used for extremely fast access to that file. A key may be composed of discontiguous data segments from within the record.

literal A specific value that represents itself (as opposed to a variable). Both numbers and text can be literal values.

method A program that you write and that ReportWriter calls when necessary.

modifiable list A list of entries that you can edit. You can also add, delete, and possibly move entries in the list.

multiple projections A method of processing secondary files to create detail records. It enables you to create records first from one file and then from the second file, instead of combining data from both files.

name link A way of associating fields with access keys in other structures. These associations can then be used by ReportWriter to access related files.

non-modifiable list A list of entries that you can select but not change, add, or delete.

numeric type A term that encompasses the decimal, implied-decimal, and integer types.

page footer The text that appears at the bottom of each page of your report. The maximum size of a page footer is 10 lines.

page header The text that appears at the top of each page of your report. The maximum size of a page header is 10 lines, which default to the header lines from your report record.

post-break line A line that is printed at the end of a report break, after the break summary line.

pre-break line A line that is printed at the beginning of a report break.

precision The number of places after the decimal point in an implied-decimal field.

primary file The first report file selected for your report. It determines what other files will be available for selection, based on the relations that were established in the repository.

proportional font A font in which different characters have different widths (like the font used to write this sentence).

question field A temporary field used to hold answers obtained from the user when a report is run.

ranging Enables you to designate a portion of a field for ReportWriter to use (for sorting, selecting, or printing), instead of the entire field. If your field is longer than 99 characters, ranging is invoked automatically.

record
A unit of data (consisting of one or more fields) into which files are subdivided. Each record in the file contains different data but has the same field layout.

report break
A sort field setting that causes the report to generate a new page when the value of that field changes. A subtotal will be computed for any field that will be totaled at the end of the report. (You can also set report breaks that don't generate a new page.)

report definition
A report.

report definition file
A file that contains one or more report definitions.

report file
A file definition/structure combination that has been selected for use in your report.

report footer
The text that appears at the end of your report. The maximum size of the report footer is 10 lines, and it can either be on the same page as the end of the report or on a separate page.

report format
The report header, date and page number line, page header, report record layout, pre- and post-break lines, page footer, and report footer. You can view these items at any time while designing a report.

report header
The text that appears at the beginning of your report. The maximum size of a report header is 10 lines, and it can either be on the same page as the start of the report or on a separate page.

report record
The collection of fields from your selected report files that you've chosen to print in your report.

Repository
The Synergy/DE application that orders and defines data structures, files, and attributes.

repository
The centralized location where your data definitions are stored. If you're a ReportWriter user, your repository should already be set up for you.

RPS_*xxx*_METHOD subroutines
Subroutines that you write in order to overload certain functionality built into ReportWriter. These subroutines enable you to provide or modify the processing of user-defined file types, file definition open filenames, and user-defined data types.

selection criteria
The fields and conditions that ReportWriter will use and compare in selecting the records for your report. You can specify up to 25 conditions (connected by AND and OR).

selection window

A window containing a choice of one or more entries that can be selected (usually by highlighting them and pressing ENTER).

sister file

One of a pair of files in a tree of relationships in which both files are selected to read and both are on the same level in the tree.

sort fields

The fields on which your report will be sorted. You can select up to 10 sort fields for a report.

strip character

The character that will follow a blank-stripped field. For example, you would probably select “**,**” as the strip character for a blank-stripped city field. The default strip character is a blank space.

structure

A record definition or compilation of field and key characteristics for a particular file or files.

subscripted field

The specification of a particular element of an arrayed field. Whenever an arrayed field is selected in ReportWriter (for printing, selection criteria, or a calculation expression), you are prompted to specify which element of the array to use. This subscript value may be a literal or another field name.

subtotal access field

A temporary field that gives you access to a field’s running total or complete total at any break level.

temporary field

A general term used to describe calculation, text, question, environment, and subtotal access fields. All temporary fields are created when the report is defined and reside in the temporary file.

temporary file

A report file that ReportWriter creates when you’ve created one or more temporary fields. It is available to the user as another report file from which fields can be selected for printing, sorting, subscripting, ranging, selecting, and use in expressions.

text field

A temporary field that defines a static text string to be printed on report records.

user-overloadable routine

A routine that you write and register so that ReportWriter calls it automatically at the appropriate time.

XCALL interface

Refers to ReportWriter’s external subroutine interface.

Index