Computer Museum

Series 100/Condor®
20-3
User's Manual

{

HEWLETT PACKARD

NOTICE

The information contained in this document is subject to change without notice.

HEWLETT-PACKARD MAKES NO WARRANTY OF ANY KIND WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. Hewlett-Packard shall not be liable for errors contained herein or for incidental or consequential damages in connection with the furnishing, performance, or use of this material.

Hewlett-Packard assumes no responsibility for the use or reliability of its software on equipment that is not furnished by Hewlett-Packard.

This document contains proprietary information which is protected by copyright. All rights are reserved. No part of this document may be photocopied, reproduced, or translated to another program language without the prior written consent of Hewlett-Packard Company.

Printing History

New editions of this manual will incorporate all material since the previous edition. Update packages may be used between editions and contain replacement and additional pages to be merged into the manual by the user.

The manual printing date and part number indicate its current edition. The printing date changes when a new edition is printed. (Minor corrections and updates which are incorporated at reprint do not cause the date to change.)

First Edition - September 1983

Instructions For Using This Manual

The *Condor 20-3 User's Manual* describes the Condor 20-3
capabilities which are not covered in the *Condor 20-1 User's Manual*.
To use this manual, you should insert it into the 20-1 manual.
Remove the Title Page, Table of Contents, Index, and Chapter 6
from the 20-1 book. Then insert the 20-3 sections into their
appropriate locations.

HP Computer Museum
www.hpmuseum.net

Table of Contents

Chapter 1: Database Concepts

Chapter 2: Getting Started with Condor

Chapter 3: Defining a Database

Chapter 4: Entering and Updating Data

Chapter 5: Simple Inquiries and Reports

Chapter 6: Advanced Capabilities

Chapter 7: Database Utilities

Chapter 8: Using Databases with Other Applications

Chapter 9: Restructuring and Reorganizing Databases

Chapter 10: Database Command Procedures

Chapter 11: Transaction Processing

Chapter 12: Writing Reports

Chapter 13: Indexing a Database

Appendix A: Example Databases

Appendix B: Report Writer Example

Index

Chapter 6

RELATIONAL CAPABILITIES

For many questions that arise in daily business use, a simple inquiry or report, like those in the last chapter, will provide the needed information. For this kind of inquiry, all the information needed to provide an answer is located within a single database. But you will also have inquiries that require information from several different databases to be combined, related, and compared to arrive at an answer.

Condor excels at this kind of complex database manipulation. It is a relational database management system, which means that data from many different databases can be related and combined on demand. These relational database management capabilities are the topic of this chapter.

The Result Database

Many of the database commands in this chapter combine data from two databases to create a third database. This database contains the results of the database operation (such as a merge, selection, etc.), and is appropriately named the RESULT database. Figure 6-1 shows the role of the RESULT database:
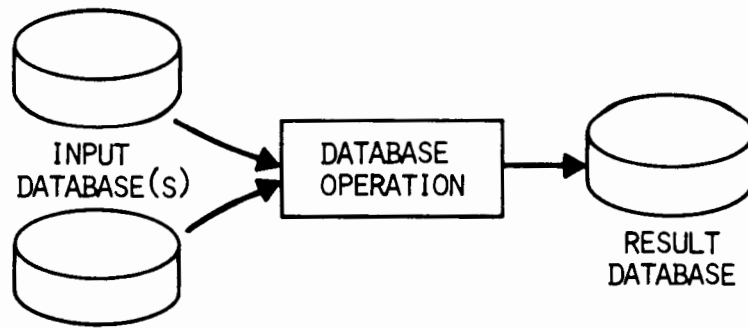
Figure 6-1. The RESULT Database

The RESULT database is always created on the **current disc drive**. It is automatically created by commands that produce a RESULT database.

Creating the RESULT database on a disc requires that a **Data Dictionary** already exists on that disc. This will automatically be true if any database has been defined or copied onto the disc. Chapter 9 fully explains the Data Dictionary and its function.

Selecting Database Records

One of the most common types of inquiries is "show me all the records that meet the following criteria ...". For example, you may want to select all the people in the **CARDFILE** database who live in Texas; or select all parts in an **INVENTORY** database where stock-on-hand is less than six months' supply.

This type of selection is done with the SELECT command:

    SELECT database

The SELECT command is very similar to the UPDATE and DISPLAY commands. Like those commands, it asks for a search condition, and searches the database for records that match the condition. But with the SELECT command, the records that match the search condition are placed in the **RESULT** database, so that you may process them further. Figure 6-2 illustrates the SELECT command.
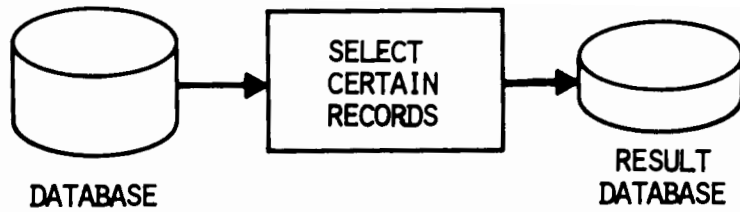
Figure 6-2. The SELECT Command

You can shorten the SEARCH command dialogue by including the search condition in the command, following the word "WHERE":

```
SELECT database WHERE search-condition
```

Example:

```
SELECT CARDFILE WHERE STATE IS AZ
```
    (records for Arizona residents are copied to the RESULT database)

```
PRINT RESULT BY NAME, CITY, STATE
```
    (print the RESULT database)

The rules for search conditions in the SELECT command are the same as those for the DISPLAY and UPDATE commands (see Chapter 4).

Saving the RESULT Database

Whenever you use a command that creates a **RESULT** database, the former contents of the **RESULT** database are replaced and lost. If you want to preserve the **RESULT** database, and return to it for later processing, you must save its contents. The SAVE command lets you save the **RESULT** database under another name:

SAVE database

The **RESULT** database is copied into the database named in the command. Figure 6-3 illustrates the effect of the **SAVE** command.

RESULT
DATABASE

DATABASE
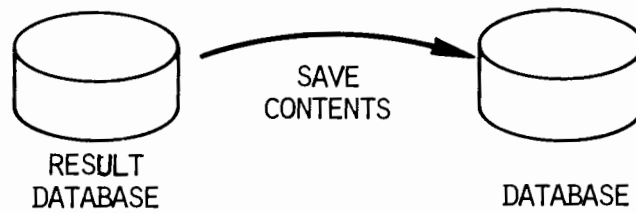
SAVE
CONTENTS

Figure 6-3. The SAVE Command

The SELECT and SAVE commands are often used together for complex inquiries that ask "what if" questions. A common inquiry technique is to successively narrow the search condition until an acceptably small number of records is found, then review the records selected. The following example illustrates this database inquiry technique.

Example:

Suppose we wish to search an EMPLOYEE database, to find employees eligible for the position of Manager of Engineering. Candidates should meet the following qualifications:

Mandatory conditions:
BS or MS degree
Management experience

Desirable conditions:
MSEE degree
Not currently at or above salary level
5 or more years experience with the company

First find those matching the mandatory conditions:
`SELECT EMPLOYEE WHOSE DEGREE IS BS* OR DEGREE IS MS*`

Data Base: EMPLOYEE 2820 Records.
Busy
Total records in Result Set = 152

Two SELECTs must be used because AND and OR cannot be combined in the same search condition:

`SELECT RESULT WHOSE POSITION > 1999`

Data Base: RESULT 152 Records
Busy
Total records in Result Set = 51

We assume position codes 2000 and above are management. Note that the RESULT database is used as the source of the search, and is overwritten with the result of the search. Candidates meeting the mandatory conditions are saved in case succeeding searches yield too few candidates and we wish to backtrack:

```
SAVE POTENTL
SELECT POTENTL WHOSE SALARY < 40000
```

Data Base: POTENTL 51 Records
Busy
Total records in Result Set = 30

Close, let's try for MSEE's:

```
SELECT RESULT WHOSE DEGREE IS MSEE
```

Data Base: RESULT 30 Records
Busy
Total records in Result Set = 2

Probably the best candidates. Find out who they are:

```
LIST RESULT BY NAME, DEPT
```

Now, backtrack to get secondary candidates:
```
SELECT POTENTL WHOSE SALARY < 40000 AND DEGREE IS BS?E

SELECT RESULT WHOSE HIRE.DATE < 6/1/76
LIST RESULT BY NAME, DEPT
```

As the example shows, the SELECT/SAVE combination allows powerful ad hoc inquiry, which can explore many different paths before reaching its conclusion.

Combining Two Databases

Data from two databases can be combined together into a single database, if both databases have the same structure (identical number of fields and record sizes). For example, clients who attended each of two different seminars may have been registered into two different databases, and you can combine the two databases to analyze the entire group of attendees.

Three commands permit the combining of records in two databases: APPEND, COMBINE, and JOIN.

The APPEND Command

The APPEND command adds the records of one database to another:

    APPEND database1 database2

Both databases must have an identical structure; i.e. they must contain the same number of data items, in the same order, with the same lengths. The records of database 2 are appended (added) to database1. Database2 remains unchanged by the command. Figure 6-4 illustrates the command.
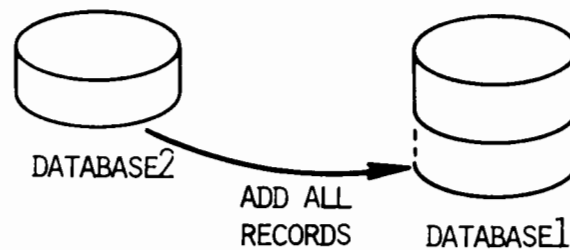
Figure 6-4. The APPEND Command

Example:

Orders for the day are contained in an ORDERS database. At the end of the day, these order records must be added to the other order records for the month, in the month-to-date orders database (MTDORDER), from which month-to-date reports are generated. The command:

```
APPEND MTDORDER ORDERS
```

appends the day's orders to the month-to-date database.

The COMBINE Command

The COMBINE command is similar to the **APPEND** command, but it leaves the data in both databases undisturbed. Instead, databases are combined into the **RESULT** database. Figure 6-5 illustrates the COMBINE command:
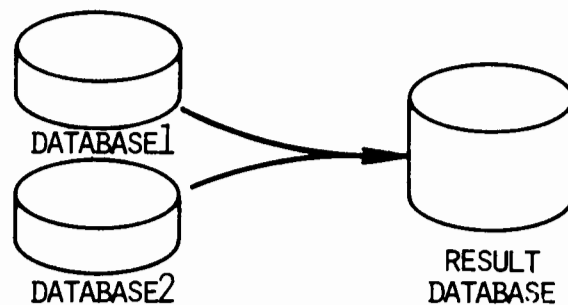
Figure 6-5. The COMBINE Command

The `COMBINE` command has the same form as the **APPEND** command:

```
COMBINE database1 database2
```

Again, both databases to be combined must have the identical structure. When the command is complete, the **RESULT** database will have the data from both databases and the database form and data item definitions from database1.

Example:

People who attended two recent seminars were registered using two databases, named SEMINAR1 and SEMINAR2. Now, you want to generate statistical reports on total seminar attendance. The command:

```
COMBINE SEMINAR1 SEMINAR 2
```

generates a **RESULT** database, merging the two databases for analysis.

The JOIN Command

The **APPEND** and **COMBINE** commands combine databases with identical structure (i.e. both databases have the same data items, which are defined the same way). The `JOIN` command is used to combine selected data from two databases which do not have the same structure.

Consider the following example:

Regional order data is collected during the month in a REGIONS database, which also includes the Region Manager's name. At month-end, you wish to generate a mailing to the region managers, detailing their order performance. The REGIONS database does not contain a mailing address for the Region Managers. However, the CARDFILE database does contain their addresses. To generate the mailing, you need to combine the addresses from the CARDFILE database with the order data from the REGIONS database.

The `JOIN` command is designed for exactly this purpose.

The `JOIN` command operates by matching selected data items from the two databases to be combined. These data-items are called the "matching items", and are listed in the command:

    JOIN database1 database2 MATCHING data-item, ...

Example:

    JOIN REGIONS CARDFILE MATCHING NAME
    (data from the REGIONS and CARDFILE databases will be
    combined)

The `JOIN` command searches both databases, looking for a record in database1 and a record in database2 whose "matching items" match. (In the example, the command searches for a CARDFILE record and a REGIONS record that both contain the same NAME.) Each time the JOIN command finds such a pair of records, a new record is written to the **RESULT** database. The new record contains all the data items of database1 and all the data items of database2. (The matching fields, since they have the same value in both databases, are not repeated twice, but will only appear once in the **RESULT** database.)

Figure 6-6 illustrates the JOIN command.

6-10
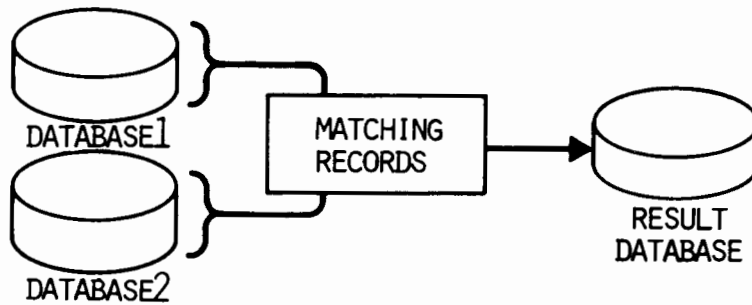
Figure 6-6. The JOIN Command

In the example above, the RESULT database would contain all the data items from the CARDFILE database (Street, City, Home Phone, etc.) and all the data items from the REGIONS database (Name, Order Dollars, etc.). Some of the data items may not be needed for subsequent processing. The PROJECT command (described in the next section) can be used to discard unneeded data items.

Example:

A report showing monthly orders by product and product line is required. The TABULATE command has already been used to create a RESULT database with one record per product, containing the required statistics. But product names and the product line code for each product is contained in a separate PRODUCTS database. The command:

    JOIN RESULT PRODUCTS MATCHING PRODUCT.NO

will produce a new RESULT database that includes the required information. The required report can be generated from this database.

Projecting a Database

When processing large databases, the time it takes to process Condor commands depends on how many records are in the database, and on how much information is contained in each database record. Large records require more time to process (and consume more space on the disc) than small records.

Often, only a few of the data items in a database are needed to answer an inquiry. The PROJECT command can be used to select only those data items, eliminating the others from the record. The resulting database (with fewer data items) is written to the RESULT database. Figure 6-7 illustrates the effects of the PROJECT command:
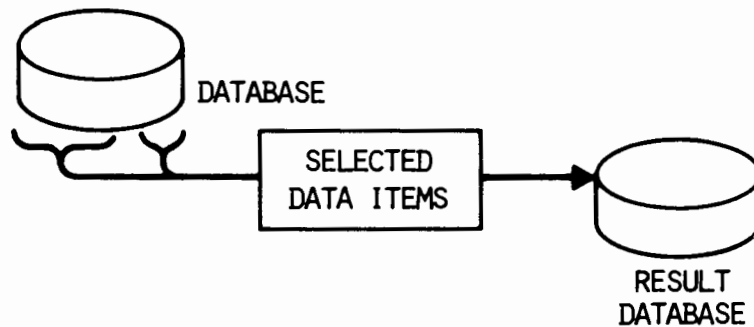
Figure 6-7. The PROJECT Command

The data items that are to be included in the RESULT database are named in the PROJECT command:

```
PROJECT database BY data-item1, data-item2, ...
```

Example:

An ORDERS database contains quite a lot of information about each order—Firm Name, Ship-to and Billing Addresses, etc. To product statistical reports on order dollars, order types, and order dates, it will be much more efficient to work with a database containing only that information. The command:

```
PROJECT ORDERS BY AMOUNT, DATE, ORDER.TYPE
```

creates a (much smaller) RESULT database that contains only the data items needed for the analysis. Subsequent SELECT, SORT, and PRINT commands will require less disc space and operate faster as a result.

Often several JOIN and PROJECT commands will be used in sequence to produce a RESULT database that is needed for a report. The following example illustrates this type of inquiry.

Example:

A report showing all the orders for the day, by customer, is required. The report is to show, for each order, the order number, the customer name, the product ordered, and the order amount. Subtotals are to be computed for each customer.

Three separate databases hold all the required information:

The ORDERS database contains, for each order:
ORDER.NO	(Order number)
CUST.NO	(Customer number)
PRODUCT.NO	(Product ordered)
QUANTITY	(Quantity ordered)
UNIT.PRICE	(Unit Price)
AMOUNT	(Amount of purchase)
— etc —	

The CUSTOMER database contains, for each customer:
CUST.NO	(Customer number)
FIRM	(Company name)
NAME	(Name of contact)
— etc —	

The PRODUCTS database contains, for each product:
PRODUCT.NO	(Product number)
PRODUCT.LINE	(Product line code)
DESCRIPTION	(Product description)
— etc —	

The following sequence of commands will produce the desired report:

JOIN ORDERS CUSTOMER BY CUST.NO
(Join these two databases to get a customer name for each customer number in the ORDERS database)

PROJECT RESULT BY ORDER.NO, FIRM, PRODUCT.NO, AMOUNT
(Discard unneeded data items from the join operation with this PROJECT command)

JOIN RESULT PRODUCTS BY PRODUCT.NO
(Join again, this time to get a description of each product ordered from the PRODUCTS database)

PROJECT RESULT BY ORDER.NO, FIRM, DESCRIPTION, AMOUNT
(Again, discard unneeded data items to make the following SORT go as rapidly as possible)

SORT RESULT BY FIRM
(Sort into order for printing)

TITLE "***** Daily Orders Summary *****"
(Title the report)

PRINT RESULT BY ORDER.NO, FIRM, DESCRIPTION, AMOUNT
AND COMPUTE TOTAL AMOUNT SUBTOTAL USING FIRM
(Finally, print the report)

Database Arithmetic

Data item values within a database can be changed in many different ways. The UPDATE command (described in Chapter 4) allows you to change values, one-by-one, for selected records in a database. Updating an entire database in this way would be very tedious, however. The COMPUTE and CHANGE commands allow you to change a data item value throughout an entire database with one command.

The COMPUTE Command

The COMPUTE command calculates a new value for a data item in every record of a database. The data item to be computed, and the formula for computing its new value, are both given in the command:

COMPUTE database ST data-item = expression

For each record in the database, the COMPUTE command performs the computation indicated, and assigns the resulting value to the data-item. The expression in the command must be an algebraic expression using addition (+), subtraction(-), multiplication (*), and division (/).

Examples:

```
COMPUTE ORDERS ST AMOUNT = QUANTITY * UNIT.PRICE
COMPUTE ORDERS ST TAX = AMOUNT * 6/100
```

The individual variables that are added, subtracted, multiplied and divided must be either numeric constants (e.g. 100) or numeric other data items from the same record. The expression is evaluated according to parentheses within the equation. If there are no parentheses, Condor performs multiplication and division before addition and subtraction. Dollars-and-cents constants in the expression should always have a decimal point and two decimal places. Constants or data items without decimal points are treated as cents when combined with dollars-type data items (i.e. 1067 = $10.67). Refer to the COMPUTE command in the Command Reference Section if you would like more details.

The CHANGE Command

The CHANGE command also changes the value of a data item in each record of a database. In this case, the new value is not computed, but is merely given in the CHANGE command itself. For example, you might want to change the date in each record of an ORDERS database to today's date. The CHANGE command lists the data item to be changed, and its new value:

    CHANGE database ST data-item IS value ...

The word "ST" in the command is an abbreviation for "SUCH THAT".

Examples:

CHANGE ORDERS ST DATE IS 02/15/82
    (the date for all orders is set to 02/15/82)

CHANGE EMPLOYEE ST STATUS IS ACTIVE
    (every employee is set on active status)

CHANGE EMPLOYEE ST STATUS IS ACTIVE AND DATE = 02/15/82
    (two changes are made in one command)

CHANGE EMPLOYEE ST STATUS1 IS ~STATUS2
    (Status1 is assigned the value of Status2)

As the examples show, several data items can be changed in a single CHANGE command simply by repeating the "data-item IS value" phrase with the word "AND". An equals sign can be used as an abbreviation for the word "IS".

The last example shows how one data item can be assigned the value of another data item in the same record. The "tilde" (~) is used to tell Condor that STATUS2 is a data item name, and that the value of this data item is to be assigned to the data item named STATUS1. (Note that if the ~ were not used, the command would cause STATUS1 to be assigned the value "STATUS2" as a literal, seven-letter word.)

The POST Command

The COMPUTE and CHANGE commands perform their arithmetic within each individual database record. Data item values within the record determine the new value for the data item being changed. Sometimes, data item values must be changed based on information from outside the database.

For example, suppose we were accumulating regional month-to-date order statistics in a REGIONS database, with one record per sales region. As each individual order is received during the day, it is entered into an ORDERS database. Each order includes information identifying the sales region from which it came. How can the REGIONS database be updated at the end of the day to reflect the day's orders?

The POST command performs this updating function. Using the POST command, records from one database (usually referred to as the "transaction" database) can be POSTED to another database (usually referred to as the **master** database), altering the contents of the master database. In the example, we would POST the ORDERS database to the REGIONS database, updating regional order statistics.

Figure 6-8 illustrates the operation of the POST command.
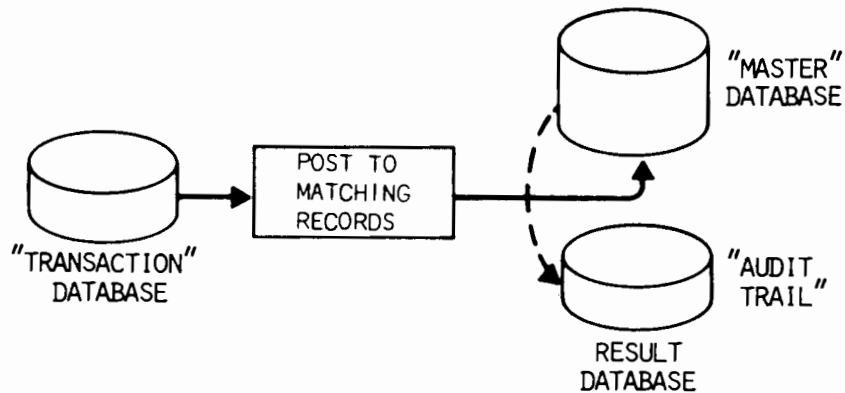
Figure 6-8. The POST Command

For each record in the **transaction** database, the POST command:

1. Finds the matching **master** database record, using one or more **matching fields. The matching fields must have the exact same structure** (same data type, length, etc.). In our example, the "matching field" is the REGION, which determines which master database record will be posted.

2. POSTs the **master** database record, as requested in the POST command. Several different data items in the "master" database can be updated with a single POST command.

   The POST command can perform several different kinds of posting:

   ADD — the data item value from the **transaction** record is added to the corresponding data item in the "master" database record. You would use this option to accumulate total orders, for example.

   SUBTRACT — the data item value from the **transaction** record is subtracted from the corresponding data item in the "master" database record. You would use this option to process withdrawals from a total inventory count, for example.

   REPLACE — the data item value from the **transaction** record replaces the corresponding data item value in the **master** database record. You would use this option to replace the "last update" date in a master database, for example.

The databases, matching fields, posting operation, and data items for posting are all named in the POST command:

```
POST database1 database2 MATCHING data-item1, ...
posting-operation data-item2 ...
```

Database1 is the **master** database, and **database2** the **transaction** database for the posting operation. **Data-item1** specifies the matching field. The posting operation may be any of those listed above; the operation is carried out on the data-item(s) listed as **data-item2**.

Example:

Daily orders from the ORDERS database are to be posted to the REGIONS database. The command:

```
POST REGIONS ORDERS MATCHING REGION ADD AMOUNT
```

causes each order record to be matched against the correct regional record in the REGIONS database. The AMOUNT from the ORDERS record will be added to the AMOUNT in the REGIONS record. At the completion of the POST command, the REGIONS database is up-to-date with the day's new orders.

Example:

Withdrawals from a PARTS inventory database are entered into a TRANSACT database during the day. At the close of business, the PARTS inventory must be updated, and reports generated for ordering new parts. The command:

```
POST PARTS TRANSACT MATCHING PART-NO SUBTRACT
QUANTITY
```

causes each TRANSACT record to be matched with the master PARTS inventory record for the part withdrawn. The quantity on hand is decreased by the quantity of parts withdrawn. At the end of the POST command, the PARTS database has been updated to reflect the day's withdrawals.

You may use more than one matching field by listing several data-items, separated by commas, following the word **MATCHING** in the command. If several matching fields are listed, all the fields must match for the posting to occur. Similarly, the posting-operation may be applied to more than one data-item. You can also combine different posting options in the same command.

Examples:

```
POST PARTS TRANSACT MATCHING PART-NO, BIN-NO SUBTRACT
QUANTITY
```
(two matching data items)

```
POST PARTS TRANSACT MATCHING PART-NO SUBTRACT QUANTITY,
REPLACE DATE
```
    (two posting operations)

The POST command creates a RESULT database as a by-product of the POSTing operation. Before a **master** database record is updated, it is copied to the RESULT database. The RESULT database thus serves as an audit trail of the POSTing operation. This can be saved or printed using the appropriate command.

The POST command must search the **master** database for each record of the **transaction** database, which can be a time consuming task. The command will operate much more rapidly if both the **master** and **transaction** databases are sorted into sequence by matching fields before POSTing.

For more examples of the POST command, see the General Ledger example in Chapter 11.

Comparing Two Databases

Two databases can be compared to one another using the Condor COMPARE command. The compare command lists the individual data items that are to be compared, record by record:

```
COMPARE database1 database2 MATCHING data-item, ...
```

The COMPARE command works in the following way:

1. A record of database1 is read

2. A search is made through database2, to find a record whose data-items match those of the database1 record. All the data-items listed in the COMPARE command must match in order for the database2 record to be considered a "match".

3. If a matching record is found, the database1 record is added to the **RESULT** database.

4. Steps 1-3 are carried out for each record in database1.

5. At the end of the COMPARE command, the RESULT database will contain those database1 records for which matching database2 records were found.
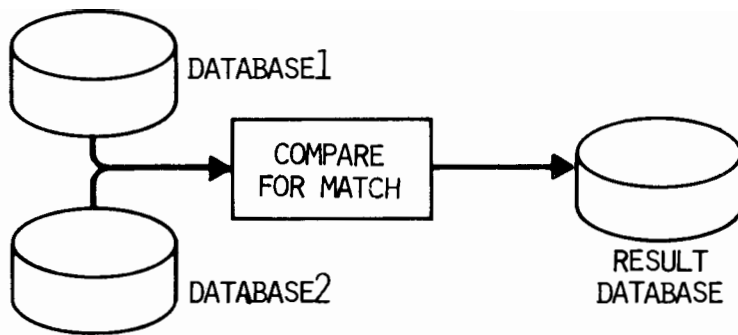
Figure 6-9. The COMPARE Command

Example:

Two databases of prospective customers have been produced from two different sources. We wish to compare the databases, to find duplicates. The command:

```
COMPARE LIST1 LIST2 MATCHING NAME, FIRM
```

will produce a RESULT database containing all the records in LIST1 that had a duplicate record in LIST2 (i.e., the same NAME and FIRM).

The COMPARE command can also be used to find the records in a database that do NOT match any records in another database:

```
COMPARE database1 database2 NOT MATCHING data-
item,...
```

The COMPARE...NOT MATCHING command follows exactly the same five-step process described for the COMPARE command. However, the records placed in the **RESULT** database will be the records from database1 for which no matching database2 record can be found.

Example:

Daily orders have been entered into an ORDERS database, and a data-item in each record tells from which sales region the order came. Before posting the REGIONS database with the ORDERS, we should insure that only valid regions have been entered into the ORDERS database records. The command:

```
COMPARE ORDERS REGIONS NOT MATCHING REGION
```

will place in the **RESULT** database any ORDERS record which does not have a matching region in the REGIONS database. These records should be corrected before posting.

Note that both databases in a COMPARE command must contain the data-items to be compared, and the data items must be defined the same way (data item name, type and size) in both databases. However, the other data items in the two databases may be quite different.

The order of the databases in the COMPARE command is also important. Note that

```
COMPARE ORDERS REGIONS MATCHING REGION
```

is not the same as

```
COMPARE REGIONS ORDERS MATCHING REGION
```

In the first example, records from the ORDERS database will be placed in the **RESULT** database; in the second, records from the REGIONS database will be placed in the **RESULT** database.

More examples of the COMPARE command can be found in the General Ledger example of Chapter 11.

Deleting Database Records

Records of a database can be deleted one-by-one through the UPDATE command. However, if many records are to be deleted, this is a time consuming process. The DELETE command lets you delete many records from a database with a single command:

```
DELETE database WHERE search-condition
```

The command searches the database named for records that match a search condition. Only those records that match the search condition are deleted. The search-condition follows the same rules described for the search-conditions in the UPDATE, DISPLAY and SELECT commands (see Chapter 4).

The DELETE command has far-reaching effects, since it can delete many records at a time, and the data in deleted records is no longer accessible. Care should therefore be used when using the DELETE command.

Note that deleted records are logically deleted, but they will still occupy space in the database and will still be included in the record count for the database. Sorting the database will collect together all the space occupied by the deleted records, making it available for other records and decreasing the record count accordingly.

Examples:

```
DELETE CARDFILE WHERE REV-DATE LT 01/01/75
```

(deletes CARDFILE records not revised since 1974)

```
DELETE CARDFILE WHERE STATE IS MI
```

(deletes Michigan residents from the CARDFILE database)

Chapter Summary

- The relational commands in Condor allow multiple databases to be related "on the fly" to respond to complex inquiries.

- The SELECT command selects specific records from a database and places them in the **RESULT** database.

- The SAVE command saves the **RESULT** database for later use.

- The APPEND command adds the records of one database to another.

- The COMBINE command combines two databases into a **RESULT** database.

- The JOIN command matches records from two databases, based upon matching fields, and combines their data items in a **RESULT** database.

- The PROJECT command creates a **RESULT** database that contains only selected data items from the original database.

- The CHANGE command changes a data-item value in every record of a database.

- The COMPUTE command computes a new value for a data-item from an algebraic expression.

- The POST command posts **transaction** records to a **master** database, updating selected data items.

- The COMPARE command compares two databases, placing in the **RESULT** database the records that do or do not match.

- The DELETE command deletes selected records from a database.

Chapter 12

WRITING REPORTS

In the preceding chapters, you learned about commands that produce simple reports: LIST, PRINT, STAX, TABULATE, and TITLE. Outside of building a form file, you cannot, however, specify the format of a report using these commands. To specify the format of a report, use the **Report Writer** included in Condor 20-3 (accessed by entering the REPORT command).

With the Report Writer, you control of all aspects of formatting a report:

- Special formats and columnar formats may be combined in a single report.

- Column headings of one or more lines may be specified independent of the data item names.

- The printing width and format of the data items may be selected or changed at will.

- Labels, text, controlled section breaks, and other special features, such as computed items and totals, may be included in a report specification.

In addition to formatting, Report Writer in Condor 20-3 enables you to print and list reports. Several report specifications may be created for a single database, or one report specification may be used with several different databases.

This chapter will teach you how to use the Report Writer to create, modify, and describe a report specification as well as print reports. The chapter also explains how to invoke the Report Writer using or not using its command menu, and how to access the Report Writer through command procedures.

All of the details necessary for creating a report specification are contained in this chapter. Appendix B shows a step-by-step example of how to create the Warehouse report which is discussed later in this chapter.

About Using the Report Writer

The Report Writer produces a range of reports, including:

- Columnar reports
- One record per page, full page formats
- Reports requiring subtotals, totals, and/or averages
- Mailing labels
- Reports which require section breaks

The concept of the Report Writer is straightforward: you create a report specification for a database and that specification is used to print a report. You can save the report specification so that the report can be printed at any time.

The Report Writer references a database's .DEF (definition) file during the creation of a report specification. Its .DAT (data) file is utilized during report printing. Report Writer never refers to the database .FRM (form) file.

In the creation of a report specification, two files are produced: **database.RPT** and **filename.FRM**. You are asked to name both of them.

database.RPT — This will be your saved report specification. Whenever revising, printing, or describing a report, this file will be referenced.

filename.FRM — This will be a report form which you build as a model to create a report specification. It should be saved in case the report specification must be re-created. **Caution: This form is different from the database form and should not be given the same name. Assigning the same name would cause the database form to be lost, forcing you to create it again.**

The Report Writer is set up to allow you to quickly define a report specification and then edit it to meet your exact requirements. Designing a report with the Report Writer is similar to using the `DEFINE` command to define a database. You begin by creating a model of the report form on your screen. Then, the Report Writer uses the report form and responses from you to specify the format and contents of the report.

A report specification may be revised after it has been created by using the revise option of the Report Writer. Page length and margins can be altered, lines can be repositioned within the report, and text and data items can be repositioned within a line. However, you will not be able to insert or delete lines of text or data or correct spelling errors. If a spelling error is detected on the report form (.FRM) file when creating a report, that form will have to be changed and the report specification created over again.

When you are ready to view the report with data in it, you may send the report to the printer (print the report) or to the screen (list the report). In addition, you may choose a particular set of data records from your database to be included in the report. This is done by specifying a WHERE condition in the Report Writer, similar to the WHERE condition in the `DELETE, DISPLAY, SELECT,` and `UPDATE` commands.

The best way to understand the Condor Report Writer is by using it to create an actual report specification. Figure 12-1 shows a sample warehouse inventory report organized by division and warehouse. It has titles, category headings, subtotals, and a grand total, along with basic information. Figure 12-2 shows the screen format of the Warehouse database used to create this report and its attribute summary. You will probably want to create your own warehouse database so that you can follow the examples given for using the Report Writer.

```
                           WAREHOUSE UPDATE REPORT

                                                  PAGE    1

Item     Item          Prev.                     Current  Unit    Total
Number   Description    On Hand   Shipped   Received  On Hand  Price   Value

DIVISION 1         WAREHOUSE 1

  1      SHOVELS        25          7         4        22     5.80    127.60
  2      TROWELS        20        100        60       -20     3.33    -66.60
  5      RED STONES     87         91        23        19     4.71     89.49
  7      YELLOW STONES  97          7        14       104     4.58    476.32
  8      BLACK STONES   50         19        23        54     4.88    263.52

                           WAREHOUSE  1        TOTAL           890.33


DIVISION 1         WAREHOUSE 2

  1      SHOVELS        27          9        14        32     5.80    185.60
  2      TROWELS        15          6         0         9     3.33     29.97
  5      RED STONES     93         75        12        30     4.71    141.30
  7      YELLOW STONES  47         14        11        44     4.58    201.52
  8      BLACK STONES    3          8        31        26     4.88    126.88

                           WAREHOUSE  2        TOTAL           685.27


                  DIVISION  1        TOTAL                    1575.60


DIVISION 2         WAREHOUSE 1

 11      BLUE CEMENT    12         10         2         4     3.40     13.60
 12      YELLOW CEMENT  30         10         5        25     4.50    112.50
  8      BLACK STONES   50          1        34        83     4.50    373.50

                           WAREHOUSE  1        TOTAL           499.60


DIVISION 2         WAREHOUSE 2

 11      BLUE CEMENT    40         12         3        31     3.40    105.40

                           WAREHOUSE  2        TOTAL           105.40


                  DIVISION  2        TOTAL                     605.00


                  GRAND  TOTAL                               2180.60
```

Figure 12-1. Warehouse Inventory Report

SCREEN FORMAT OF WAREHOUSE DATABASE

[DIVNBR] _   [WHSENBR] _
 (division number)   (warehouse number)

[NO] __   [ITEM] _____
 (item number)   (item description)

[PREQTY] __   [SHIPPED] ___   [RCVED] __
 (previous quantity)   (number shipped)   (number received)

[PRICE] _____

****

            Number of underscores in screen format

                    [DIVNBR]   1
                    [WHSENBR]   1
                    [NO]   2
                    [ITEM]   13
                    [PREQTY]   2
                    [SHIPPED]   3
                    [RCVED]   2
                    [PRICE]   6

****

Attribute summary of Data Base WAREHSE

1.DIVNBR: N,1,1,5,"   "
2.WHSENBR: N,1,0,5,"   "
3.NO: N,1,0,127,"   "
4.ITEM: AN,13,0,13,"   "
5.PREQTY: N,1,0,127,"   "
6.SHIPPED: N,1,0,127,"   "
7.RCVED: N,1,0,127,"   "
8.PRICE: $,3,.00,327.67,"   "

Record Size (Bytes) = 23
Total Records = 0

Figure 12-2. Warehouse Database Screen Format

Invoking the Report Writer
By Way of the Menu

NOTE

All examples assume that drive B contains the databases to be used with the Report Writer. If you have placed your databases on another drive, substitute that drive letter wherever **B:** or **B>>** is shown in these examples.

Once you have set up your discs and have created your database, you are ready to begin using the Report Writer. First, set the current drive to the one with your data disc by typing the following:

   B: `Return`

With your Report Writer disc in drive A:, Condor will respond with the prompt: **B>>**. Type the following:

   REPORT database-name `Return`

If the database does not already exist, or if it is not listed in the data dictionary, the following error message is displayed:

   Title not found in dictionary
   REPORT ABORTED

If you receive this message, confirm that you have your database on the disc in your current drive. Once you have ensured this and performed LOGDISK (described in Chapter 7), type the **REPORT** command again.

If the database does exist and is listed in the data dictionary, the system displays the menu shown in Figure 12-3. In our example, we assume that a database named **WAREHSE** already exists.

```
************************************************************
*                                                        *
*                                                        *
*            CONDOR SERIES 20 RDBMS REPORT WRITER        *
*                    Version 2.xx**xx                    *
*                                                        *
*                    Choose option                       *
*                                                        *
*     Create New Report Specification        (C)         *
*     Describe existing Report Specification (D)         *
*     Revise existing Report Specification   (R)         *
*     Print or Display Database Report        (P)         *
*                                                        *
*     Enter option or End <C/R>:                         *
*                                                        *
************************************************************
```

Figure 12-3. Report Writer Menu

At this point, you must decide whether you want to create a new report specification, describe (look at) an existing report specification, revise an existing report specification, or print a database report. Once you have decided, go on to the appropriate section.

Creating a Report Specification

To create a report specification, press the letter C in the Report Writer menu.

The process of creating a report specification is divided into two parts: 1) building a report model form, and 2) generating a report specification around the model form. Building a report form is much like creating a database form. Generating the report specification is a matter of responding to questions asked by Report Writer. You may select a long or short method of creating a report specification.

You are first asked if you want to use the **short method** of creating the report specification:

Do you want to use the short method (Y/N)?

Answer Y if you wish to use the short method of creating you report specification. Answer N if you would rather use the long method.

Short Method: This method is used to quickly define a report specification. You are only asked to specify: 1) The line type of each model line on your report form, 2) Details about calculated items, and 3) Control items for section breaks. After quickly defining the report specification, you are given the option to immediately revise it, so you may define special line spacing and other forms of fine tuning. Alternatively, you may choose to not revise at this point, but rather print portions of the report to check the layout. If you desire revisions after printing, you may do that by entering the Revise option at the Report Writer menu.

Long Method: This method asks many more questions and allows you to completely define your report specification as well as refine it during the creation process. After defining the report layout using the long method, you will probably want to print portions of the report and revise it if the results are not satisfactory. **This method must be used if you wish to print reports with greater than 80 columns** (see the sections titled **Report Writer Specifications** and **Helpful Hints** near the end of this chapter, when you wish to create such a report).

Creating a Report Form

The next question Report Writer asks you is:

    Do you wish to create a new report format (Y/N)?

A report format is the model form used to create a report specification. If you specify N, Report Writer asks you for the name of a report model form that already exists. Upon supplying the name of the report form, you are given the option to modify it or create a report specification around the unchanged form.

Since you have not previously created a report form, type Y for "yes". The screen changes, giving you a blank screen with the same softkeys that you used in creating a database form.

A report form is similar to a database screen form, as follows:

- Data items are identified by placing them in brackets.

- Text, page numbers, and other types of information are typed in without brackets.

However, unlike a database screen form, underscores are not used.

Lines created within the report form are called model lines. Each model line tells Report Writer which combination of data items, text, and other information you want printed on that line.

To produce the sample report seen in Figure 12-1, enter the model
lines exactly as they appear in Figure 12-4. If you feel that a grid
on the screen will help you lay out the report form, press
the `Grid` function key. The `Refresh Screen` function key erases the
grid while leaving the items you have typed.

```
                       WAREHOUSE UPDATE REPORT

                                            PAGE    $PAGE

Item     Item       Prev.                   Current  Unit    Total
Number   Description On Hand  Shipped   Received On Hand  Price   Value
DIVISION [DIVNBR] WAREHOUSE [WHSENBR]
   [NO] [ITEM]              [PREQTY] [SHIPPED]    [RCVED] [CURQTY] [PRICE] [VALUE]

                             WAREHOUSE [WHSENBR] TOTAL          [VALUE]
                        DIVISION [DIVNBR] TOTAL                 [VALUE]
                     GRAND TOTAL                                [VALUE]
```

Figure 12-4

Only one model line in the report form is used for each different
type of line in the final report, even though the same model lines
may be printed more than once per page. You can create 24
different model lines, one for each line available on the Series 100
screen, more than enough for most reports. Each individual word
or item in a model line becomes a specification, which can be
moved within the line and printed in a variety of formats.

After you finish creating the report form, check the spelling and
make sure no text or data items have been left out. A quick,
efficient check just after you create the form will save you time; to
correct a misspelled data item or title, you must return to the form
and correct the model line, and then respecify the report. After the
final check, press the `End` function key to save the report form.

Indicating Line Printing Specifications

The next few sections provide information you need to build an entire report specification. Appendix B shows step-by-step, how to create the report specification for the Warehouse database.

After creating the report form, you begin structuring the report specification. For every model line in your form, you are asked to define its line type after seeing the following menu:

```
*************************************************************

              Describe when this line is to be printed

      First Page only                              [FP]
      Last Page only                               [LP]
      Every Page Heading                           [PH]
      Once Every Line                              [EL]
      On Break-Before Section                      [BS]
      On Break-After Section                       [AS]
      Enter code                                   [PH]


*************************************************************
```

The options on this menu are used as follows:

[FP] — Lines printed on the first page only are normally used to describe a report.

[LP] — These lines are printed only on the last page of the report and are commonly used to print summary material and grand totals.

[PH] — Page headings normally appear at the top of each page. They may include the system default values. $PAGE prints the current page number. $TODAY is the current data as set by the DATE command before entering the Report Writer. Page headings normally contain titles or other information, reporting periods, and column headings. A report may have more than one page heading model line, so multiple-line titles are possible.

[EL] — This indicates that the model line is to be printed once for each valid database record and is used to print the detailed contents of the report that are repeating. It is possible to have more than one **[EL]** model line in the report form, creating a pattern of lines that will be printed once for each database record. You may limit the records that Report Writer uses. (This is explained in the section **Printing a Report**, found later in this chapter.)

[BS] — A **Before Section** line is often used as a subtitle describing the group of lines that follow it. For instance, in our WAREHSE database report, sections pertain to particular warehouses and divisions. A **Before Section** must have one or more control items to indicate when the section is changed. A control item is a data item from the database. Each time the control item changes, a section break occurs, and the **Before Section** line is printed. More than one **Before Section** line can exist, each with its own set of control items. Up to 8 sets of control items are allowed.

[AS] — **After Section** lines are identical to **Before Section** lines, except thay are printed after the group of lines that make up a section. They are printed prior to any **Before Section** lines beginning a new section and are most often used to accumulate and print subtotals, averages, minimums, maximums, and other types of information.

Item-type Specifications

Reports can have four different types of items:

Item Type	Use
TEXT	Titles, headings, descriptive text
SYSTEM	$PAGE, $TODAY
DATA	Read from database
CALCULATED or COMPUTED	Calculated by Report Writer

Report Writer can tell which type an item is, so you will not be asked about it.

Text items are placed in a report exactly as shown in the report form and are **not** placed in square brackets ([ ]) on the model line. They are used for titles, headings, and as labels for system values, data items, and calculated items within the report.

System items consist of $PAGE and $TODAY keywords. $PAGE prints the current page number in a report. $TODAY prints the date which was set by the DATE command before entering Report Writer. These values are **not** placed in square brackets and normally appear in **page heading** lines.

Data items are read from the database file. They have the same name as the item in the database from which they are to be read. In the report form, they are placed within square brackets. Data items normally make up the bulk of a report, and appear in **before section, after section,** or **every line** model lines.

Calculated items are calculated by Report Writer from data items or constants, according to a formula which you supply. They are placed within square brackets, and **cannot** have the same name as a database data item. In the WAREHSE example, CURQTY and VALUE are calculated items.

Other types of information are used to specify how an item is printed in the report:

Starting Column

>Short Method. Report Writer does not ask you for this data. All starting column positions will be determined directly from the model form. If items are expected to overlap, Report Writer will determine a recommended starting column and use it.

>Long Method. Report Writer asks you to supply the starting column for each item. A default value is displayed in square brackets ([ ]) and is based on the position of the item in the model line. To select the default, press ⎡Return⎤ . If items will overlap because of their lengths, Report Writer will issue a warning which recommends a starting position.

Read from the Database or Computed

>Short Method. If an item name is not found in the database, Report Writer will ask if it is to be a computed item, with a Y or N response required. If Y, you will be asked to specify the item length, item type (date, numeric, or dollar), the equation for the item (discussed in the section titled **Computed Item Equations**), and its printing format.

>Long Method. If an item name is found in the database, Report Writer will ask if the item is to be obtained from the database. A Y or N response is required. If you answer Y, item length, type, and format are requested. If the item name is **not** found in the database, the procedure follows that of the short method above.

Printing Width

>This specification is only asked in the long method or when specifying a computed item. The default for a database item is the number of underscores in the database screen format. A text item's default is the number of characters that appear in the item. For computed items, the default is the number of characters in the name of the item. For example, since the name of the item [TOTAL] is five characters, it will have a default printing width of 5 if TOTAL is a computed item.

Printing Format

Data items and computed items can be printed in several formats. This is also only asked in the long method or when working with a computed item. These formats are listed below, with descriptions and examples of printed output.

[SD]SYSTEM DEFAULT — Prints as in the LIST command.

[LJ]LEFT JUSTIFY — Aligns printing along the left margin of the printing width.
> "EXAMPLE   "

[RJ]RIGHT JUSTIFY — Aligns printing along the right margin of the printing width.
> "   EXAMPLE"

[NP]NOT PRINTED — Useful for accumulating subtotals without printing them, or for temporarily removing items from a report.

[LZ]LEADING ZEROS — Aligns numbers along the right margin of the printing width and fills the preceding positions with zeros. Useful for numbers used as codes.
> "00000001234"

[BZ]BLANK IF ZERO — Prints blanks instead of "0" or "0.00".

[FD]FLOATING DOLLAR SIGN — Places a dollar sign to the left of the number. If the number is negative, the minus sign is printed to the left of the dollar sign.
> " $23.50"
> "   −$23.50"

Totals, Subtotals, Averages, Min, and Max

In a report, **totals** and **averages** will be **computed items.** If your total or average item appears in **after section** or **last page** type lines, you may only use the same computed item that appeared on **every line** type lines. If you do not want to print the calculation on every line, you can specify the item to be printed with the ⌜NP⌟ format (see **Printing Formats** above). In the WAREHSE example, VALUE is used to accumulate subtotals and totals.

Choices for computed item types are:

VAL Print the item's most recent value.

SUB Print the subtotal. The subtotal is reset to zero whenever the section changes.

TOT Print the accumulated total. This total is **not** reset to zero at every section change.

MIN Print the minimum value found in the section.

MAX Print the maximum value found in the section.

AVG Print the average value for the section.

Computed Item Equations

Report Writer must have an equation or formula for each computed item. This equation must have the form:

```
itemname = term1 opr. term2 opr. ...
```

itemname is the name of the computed item. Only one item name can appear to the left of the equal sign.

The right side must have at least 2 terms and 1 operator, but no more than 32 terms and operators combined. Terms may be either numeric constants or data items read from the database. Operators are:

+ addition
- subtraction
* multiplication
/ division

A data item used in a formula need not be part of the report specification, but it must exist in the database file. Equations are evaluated from left to right without regard for parentheses.

Section Breaks

You learned how to designate section breaks earlier in this chapter in the section entitled **Indicating Line Printing Specifications**. The [BS] and [AS] line types allow you to group a section of common database values (for example: all orders for a product number, all transactions for particular dates, etc.). Each line that is printed within a section will most likely be an [EL] line type.

Use of Form Feeds

After you finish defining what the report will look like, Report Writer asks if you want it to send a form feed at the end of a report. If you are going to make multiple copies, this is the only way to make sure each copy begins at the top of the next page.

The form feed cannot be changed by the Revise option. Therefore, it is recommended that you select it unless you are **certain** it is unnecessary. Note that the top-of-form must also be properly set on your printer.

Report Specification Naming

Report Writer asks you for the name of the report specification that you have created. The default is the name of the database, creating a file called **database-name.RPT**. You may specify a different name if you want, but it cannot exceed 8 characters.

If there is already a report specification with the same name, Report Writer asks for permission to overwrite the existing specification with the new one. The following message appears:

```
filename.RPT exists already
Do you wish to destroy and re-create (Y/N)?
```

Answer Y if you wish to overwrite the old file. If you answer N, you are then asked to supply the name of a file where the report specification may be saved.

If a specification with a different name already exists for the current database, Report Writer links the new report specification to the database in the data dictionary without destroying the old report specification. Thus, a database can have more than one report specification, but only one report specification can be linked to the database at any one time.

Multiple reports can be accessed either by updating the data dictionary (data.dic) and changing the contents of the **REPORT** data item, or by entering the same database in the data dictionary several times, each time with a different title and report specification. This is illustrated in Figure 12-5 by a sample listing of the DIC command.

Title	Data	Form	Def	Index	Report
DATABASE	DATABASE	DATABASE	DATABASE		DATABASE
REPORT1	DATABASE	DATABASE	DATABASE		DATABASE
REPORT2	DATABASE	DATABASE	DATABASE		DATABASE

Figure 12-5. Sample Listing of the DIC Command

After naming the report specification, Report Writer will ask:

```
Do you wish to save the report form? (Y/N)
```

The report form is the model form that was built and used to create the report specification. It is advised that you save it, in case the report must be respecified. The name under which you save the report form **MUST NOT** be the same as that of the database. The report form will be a .FRM file and would overwrite the .FRM file that already exists for that database.

12-18

Fine Tuning the Report Specification

If you used the short method of creating the report specification, you are given the option to make changes right after the creation process. Report Writer asks if you want to make a revision; if you respond in the affirmative, the system automatically takes you into the report specification revision process. (See the section entitled **Revising a Report Specification** for details on the revision process.) Since the short method of creating a report specification uses many defaults from the model form, you may want to alter the line spacing, item positioning, and/or heading and text positioning.

You may want to print the report before fine tuning it. If so, specify that you don't want to revise the report specification at this moment. If your report looks good, you will probably want to describe the report specification and save it with your system documentation. If you are not satisfied with the appearance of the report, you may Revise the report specification.

Revising a Report Specification

You may revise information that is contained in your report specification. Enter the revision process by one of the following ways:

1. Type an R (for the Revise function) when at the Report Writer menu.

2. After using the short method of creating a report specification, answer in the affirmative when asked if a revision is desired.

3. Supply the [R] option when entering Report Writer (detailed in the section **Using Report Writer Without Going Through the Menu**).

When revising a report specification, the Report Writer does not make reference to the model report form, so you cannot add or delete information on the form. If you make changes to the report form, you must re-create the report specification. You will also be unable to change an item's type (Text, Data, etc.), but this causes no problem since Report Writer can already tell which type an item is. If you desire to change an item's type, you must re-create the specification.

During the revision process, you will be shown information contained in your report specification, line by line. The cursor stops at the end of each line, called specification lines, so that you may make revisions. If the information is satisfactory, enter either [Return] or the down-cursor key (both provide the same function here). If you wish to revise information on the line, position the cursor over the information to be revised by using the back-cursor key, and revise the line by typing new information over the item(s) to be changed. Move to the next specification by pressing [Return] or the down-cursor key.

If you decide you want to move back up to the previous specification line, use the up-cursor key. When the report items are displayed and you enter the up-cursor key to step back to a previous item, a back slash (\ ) will appear before the previous specification line is displayed. If you are at the first specification line of a model line and press the up-cursor key, the Report Writer will step back to the start of specifications for the previous model line.

In order to allow you maximum flexibility during report specification revision, the Report Writer performs only minimal checks for accuracy. It is possible for you to enter information during the revision process which will cause errors when you run the report. The most common error occurs when item starting positions are altered; this may result in the item values overlapping when a report is run.

The last screen during the revision process will display equations for Computed Items. You may change the equations at this time if you desire. After all specification have been displayed and revised, press any key to return to the Report Writer menu.

Describing a Report Specification

You may get a description of the report specification at any time by selecting the D (Describe) option from the Report Writer menu. Report Writer asks if the output is to be sent to the printer:

Send output to printer (Y/N)?

If you answer Y, the description will be printed (as in the example in Appendix B). If you need to stop printing before the entire description has been printed, type **Control-C** (hold the ⌈CTRL⌉ key and press the ⌈c⌉ key). It will take a few seconds for the printer to stop printing information already obtained from the computer.

If you answer N, the report specification description will be displayed in the same manner as when you were Revising the report specification. However, since you cannot change information during the Describe option, all information about each line is displayed on the screen at once. Press any key to see the next model line. After all model lines have been described, the equations for computed items are displayed. Press any key at this point to return to the Report Writer menu. (See Appendix B for a description of the WAREHSE report specification.)

Printing a Report

Once you have the report specification created and revised the way you want it, you may print a report by selecting the P (Print) option from the Report Writer menu. You are asked:

    Select all records for report (Y/N)?

If you answer N, you will be asked for selection criteria:

    Enter selection criteria or End <C/R>:

Answer in the form of a WHERE condition (e.g., PRICE < 10.00). You will then be asked:

    Send output to printer (Y/N)?

If you answer N, the report appears on the display. Since it's difficult to read the report as it scrolls up the screen, you may wish to stop the screen from scrolling by typing a **Control-S** (hold the ⌈CTRL⌉ key down and press the ⌈S⌉ key). To resume the display after a Control-S, type **Control-Q**. If during the process of listing the report to the screen, you wish to terminate printing, type a **Control-C**. You are then prompted to press any key to continue. Once you press a key, you are returned to the Report Writer menu.

If you answer Y, you will be asked:

    Repeat first few lines for forms alignment (Y/N)?

This feature allows you to verify that the paper is properly aligned in the printer. Enter Y, if you want to check paper alignment. You are then asked for the number of copies of the report to be printed:

    Enter number of copies to print (1-99) [1]:

The default in Report Writer is one copy. If you want multiple copies of the report to be printed, enter the number of copies desired. As is shown in the prompt, you may specify up to 99 copies.

12-22

If you had entered Y to the forms alignment question, the first few lines of the report will be printed, and you will be asked:

```
Forms alignment OK (Y/N)?
```

If you are not satisfied with the alignment of the paper, answer N to the question and realign the paper in the printer. The first few lines of the report will be printed again, and the question will be repeated. When you are satisfied with the forms alignment, answer Y to the question, and the rest of the report will be printed, from that point on.

Using Report Writer Without Going Through the Menu

Throughout this chapter, we have described how to use the Report Writer functions by going through the Report Writer menu. You may also run Report Writer without going through the menu by entering an option on the command line. The available options are:

[C] Create new Report Specification
[D] Describe existing Report Specification
[P] Print a Report
[PF] Print a Report, invoking forms alignment check
[R] Revise existing Report Specification
[L] List a Report on the Video Display

The Report Writer is invoked from the standard Condor prompt (e.g., B>> if drive B is the current drive). Examples of accessing the Report Writer in this manner are as follows:

```
B>>REPORT WAREHSE [C]              (to create report spec)
B>>REPORT ORDERS [R]               (to revise report spec)
B>>REPORT GLJOURNL [P]             (to print report)
```

Where WAREHSE, ORDERS, and GLJOURNL are names of databases.

You can also specify selection criteria for printing when invoking the Report Writer in this manner. To do this, list the selection criteria before the print option. Examples:

```
B>>REPORT WAREHSE WHERE DIVNBR < 3 [P]
B>>REPORT PRODUCTS WHERE ALL [PF]
```

Using the Report Writer in a Command Procedure

The Report Writer can also be invoked from a Condor command procedure, as outlined in Chapter 10. The command line will appear just as those which do not use the Report Writer menu, shown above. The same options are also used.

When **printing** a report from a command procedure, you will likely want it to run unattended. You may suppress the Report Writer prompts by specifying a WHERE condition and entering the [P] option on the command line. If you want all records to be selected, specify WHERE ALL, ALL, or *. If you enter the [PF] option instead of the [P] option, the alignment question will be asked, and the command procedure will not run unattended.

Note also that the TITLE is cleared when the Report Writer is started. If you want the paper to go to the top of a new page before a report is printed, use the TITLE TOP command in the command procedure before the REPORT command.

Report Writer Specifications and Helpful Hints

Maximum Number of Report Specifications Allowed

The total number of individual specifications in a report is limited, depending on the number of data items in the database definition. Remember, each individual word or item in a model line becomes a specification.

Because the total number of specifications is limited, you may want to combine separate words or phrases of titles and other text into a single specification, rather than have each word be an independent specification. To do this, place quotes around text in your report form. Text in quotes will be broken up into a series of 15-character specifications. This reduces the total number of specifications in the report and makes it easier to position the model line.

Length and Width of Paper
On Which Report Is Printed

Normally, paper which is 8.5 inches wide by 11 inches long is used for printing a report, and a printer which prints 10 characters per inch of page width and 6 lines per inch of page length is assumed. If you use this page size and printer, the default values in the Report Writer will work correctly.

The following table lists the default values:

Item of Information	Range	Default
Page Length (lines)	(1–127)	[66]
Left Margin (column)	(1–132)	[1]
Right Margin (column)	(1–127)	[80]
Top Margin (line)	(1–127)	[1]
Bottom Margin (line)	(1–127)	[60]

If you are using paper of a different size or a printer with different characteristics, you may want to replace the default values. Values for several common paper sizes and printer types are listed below. Use these values instead of the default values if you use these paper/printer combinations. If your paper/printer combination is not listed, find those that have similar characteristics to yours and combine the length and margin specifications to meet your needs.

Paper/Printer Details	Pg. Length	Rt. Margin	Bot. Margin
11-inch long paper, 8 lines/inch printer	88	80	80
8.5-inch wide paper, 12 char/inch printer	66	96	80
14-inch wide paper (132 column computer paper)	66	132	80
14-inch long (legal size) paper, 6 lines/ inch printer	78	80	72

Reports Wider than 80 Columns

A column width of 80 characters is assumed when using the short method of creating a report specification. For reports which exceed 80 characters in width, **you must use the long method.**

To specify a report wider than 80 columns, set the right margin at least 80 characters beyond the left margin. Report Writer recognizes the difference between the margins and allows you to print with a width greater than 80 if this information is specified at the beginning of the report specification **while using the long method.** (Note: If you do not specify the extended width at this time, you must re-create the report specification.)

On your report model form, you should either:

- Pack the fields and information into a single 80-column line on your report form, expanding them into their proper printing positions when you revise the report specification; or

- Use a second line, and let the items or text wrap around onto the second line.

When the report is specified to be greater than 80 columns, model lines in the report form automatically wrap around to the next line. The wrap continues until the next model line fills the available space for column width.

Double backslashes (" \ \ ") are useful in formatting your model form and report specification. When " \ \ " is encountered on a model line, the remainder of that line is ignored by Report Writer and text is stopped from wrapping. The next model line on the form is treated as a new report specification line.

Line Spacing/Positioning

When using the long method of creating a report specification, you are asked to specify the position and spacing for each line of the report. If the short method is used, you must revise the report specification to position and space the lines. Line positioning falls into three categories: 1) absolute line positioning, 2) relative line positioning, and 3) page lookahead.

Absolute Line Positioning

The prompts appear as follows:

Long Method:

```
Line number to skip to BEFORE printing line [ 0 ]:
Line number to skip to AFTER printing line [ 0 ]:
```

Short Method:

```
Line #          Line #
Before          After

   0               0
```

Absolute line positioning allows you to specify the exact place on the page that a report line is printed. For instance, if you specify that Report Writer is to skip to line 3 before printing a heading line, that heading line is always printed on line 3 of a page. Absolute line positioning is useful for ensuring that page headings and similar material appear in the same position on all pages.

When using the long method, if you specify an absolute line position, you will not be prompted for relative line position and page lookahead for that line.

CAUTION

If the report line is of the [EL] "every line" type, entering a
number (other than 0) for an absolute position causes Report Writer
to skip to a new page for every record in the database.

Relative Line Positioning

The prompts appear as follows:

Long Method:

```
Number of blank lines to insert BEFORE
    printing line [ 0 ]:
Number of blank lines to insert AFTER
    printing line [ 0 ]:
```

Short Method:

```
Space           Space
Before          After

  0               0
```

Relative line positioning specifies the number of blank lines to
appear before or after a line. This type of positioning is used
most often with **before section, after section, page heading,**
and **last page** model lines.

Page Lookahead

The prompts appear as follows:

Long Method:

```
Start new page if fewer than ____ lines remaining
```

Short Method:

```
Look
Ahead

0
```

Page lookahead is a special feature that lets you prevent
information being split across two pages, and also lets you
begin each new section on a new page.

Use this function by specifying the number of lines that must remain on a page for the line to be printed. If fewer lines remain, Report Writer goes to the next page, prints page header information, and then prints the lines on the new page.

To start each new section on a new page, specify a number which is greater than 1 in the **before section** line.

To keep groups of **every line** information together on one page, use this feature in the first **every line** specification.

Chapter Summary

- Report Writer allows you to specify the format and contents of database reports.

- Report Writer is accessed from the Condor prompt (B>>).

    1. Options may be selected after the Report Writer menu is displayed.

    2. Options may be specified in the REPORT command.

- A report is defined with two files:

    1. a REPORT MODEL FORM file

    2. a REPORT SPECIFICATION file

- A report model form is created in the same manner as in the DEFINE and FORMAT commands.

- **Short** and **Long** methods are available for defining a report specification. The long method queries you for more information and allows you to create reports which are wider than 80 columns.

- Report Writer can extract items from a database or calculate numbers from a given equation of items and/or constants.

- A report specification may be Revised or Described after it is created.

- Report Writer may be accessed using a command procedure.

Chapter 13

INDEXING YOUR DATABASE

In earlier chapters, you learned how to seek database records through functions like DISPLAY, DELETE, SELECT, and UPDATE. This is done by specifying a WHERE condition. You also learned to extract information from a database using LIST and PRINT commands. Before listing or printing, you sorted the database to organize its records in the proper sequence. In all of these cases, Condor finds its data by performing a sequential search (record-by-record). A sequential search is fine for small databases or those that don't change much, but in other cases, a database index is very helpful.

An index serves the same purpose for a database as a card catalog does for a library. Just as a card in the catalog is used to locate a book rapidly, an index is used to locate a record of information quickly. A card contains the title and location of a book and, similarly, an index contains the **key** and location of a record. The **key** consists of the specific values of one or more data items in a record. Data items used to generate a key are specified when indexing a database.

There are numerous reasons that you might want to index a database:

1. You desire quick access to specified records.

2. You wish to see reports and listings sorted in key item sequence and do not wish to re-sort the database file each time a report is produced.

3. You wish to place values automatically in order when entering into a database.

4. You desire to keep records in order when appending, reading, deleting, and/or updating records in your database.

5. You want to prevent duplicate items in a database.

Condor allows you to use one to eight database items as index keys. If you specify a single key for a database, then just that data item will be used to quickly locate records satisfying the given search criteria. If multiple keys are given for a database, then Condor uses all of the index items to locate specific records in the database. For example, if a database is indexed by both CITY and ZIPCODE, Condor can quickly locate records by CITY and, within CITY, by ZIPCODE.

This chapter will show you how to index a database, check for duplicate keys, and use Condor commands in conjunction with Condor indexing capabilities. The chapter will also describe how an index is updated and show you how to invalidate an index.

Setting Up or Re-creating a Database Index

An index may be set up for a new database or for an existing database. You also have the ability to re-create an index.

Indexing a New Database

You may establish an index for a database when you first DEFINE the database. After you create a database, you are asked:

    Do you wish to create an index for this Database (Y/N)?

To set up an index at this time, answer Y. Whether you answer Y or N, you are asked if you would like a printed copy of the database attribute summary:

    Do you want a printed copy of the data definitions
    (Y/N)?

Answer Y to send the attribute summary to the printer. Answer N to display the summary only on the screen.

If you had earlier specified that you wish to create an index for this database, you are next prompted for the item(s) which will be index keys:

> `Enter name of data-item(s) which is to be the key:`

Enter 1 to 8 database items that you would like to use as keys. Separate each item by a comma. The combined length of these items **must not exceed 127 characters.**

Next, you are asked whether you would like to permit duplicate keys to be entered into the database. (The concept of duplicate keys is explained later in this chapter.)

> `Do you wish to permit duplicate keys (Y/N)?`

Answering Y allows duplicate keys to be added to the database.

Indexing an Existing Database

You can also index an established database. Simply enter the INDEX command along with the database name and Key Data Items, and the data in the database will be indexed. Indexing an existing database with very few records is a fairly short process. It may take a little more time to index a database with a large number of records, since each record must be reviewed and an index established for it.

The format of the INDEX command is as follows:

> `INDEX database USING d1,d2,...,d8 [option]`

D1,d2,...,d8 are called Key Data Items. You may specify up to eight data items from your database to be used as keys. The combined length of all Key Data Items **must not exceed 127 characters.**

Available options ([option]) are:

> [D] Permit Duplicate Keys
> [U] Update Index

If you do not specify key data items or an option within the INDEXcommand, you are prompted for them with the following messages:

```
Enter name of data-item(s) which is to be the key:

Do you wish to permit duplicate keys (Y/N)?
```

Re-creating an Index

If database is already indexed, you have the ability to re-create the index. You may wish to do this:

- Change key data items that are indexed, and/or

- Allow/disallow duplicate key entries

Use the INDEX command to re-create an index. The database must already exist and be indexed.

Example:

Type: INDEX database

Since the database is already indexed, Condor issues the following message:

```
Index exists already>Options: Destroy(D),
 Re-create(R) or End <C/R>:
```

Type: R

Condor asks for key data items:

```
Enter name of data-item(s) which is to be the key:
```

Enter up to 8 key data items. Finally, you are asked if duplicate entries are to be allowed:

```
Do you wish to permit duplicate keys (Y/N)?
```

Duplicate Keys

In many cases, you do not want to allow duplicate keys in a database index. For example, a personnel database should probably not allow duplicate employee numbers. No two customer numbers should be the same in a customer address database. For these cases, you do not need to specify non-duplicate keys when setting up an index, as Condor defaults to that.

However, there are cases, when you want to permit duplicate keys. If you are indexing a purchase order database by VENDOR, you may have more than one purchase order with the same VENDOR. In order to tell the INDEX command to allow duplicate keys, specify the [D] option with the INDEX command, or answer Y to the duplicate key prompt when creating a database.

Updating an Index

An index is automatically updated when you use ENTER, APPEND, READ, EMPTY, and DELETE commands with a database. It is also automatically modified when you use UPDATE to delete data in the database. **A data item used as a key cannot be altered by a CHANGE, COMPUTE, POST, or UPDATE command.**

Each time a database is used, the index is checked to make sure it is current with the data. If it is found not to be current, an error message is displayed, and the command is aborted. The [U] option may be used to update an existing index. It takes about the same amount of time to update an index as it does to create an index on the existing database.

Invalidating an Index

If a database index is no longer needed, you may invalidate it in one of two ways:

- Delete the index file from the dictionary using the INDEX command.

- Re-sort or reorganize a database.

Deleting the Index File

A database index (.IDX) file may be deleted from the dictionary by using the INDEX command and choosing the option to destroy the index. The database must already exist and be indexed.

Example:

Type: INDEX database

Condor responds with the following prompt:

Index exists already>Options: Destroy(D),
 Re-create(R) or End <C/R>:

Type: D

Condor lets you know when it has finished destroying the index file by displaying:

Done.

In order to preserve the database, this is the only way that an index file should be destroyed.

Invalidating the Index with SORT or REORG

Use of the SORT or REORG commands also invalidates an index. By changing the sequence of the records in the database, the index is no longer accurate. To reconstruct the index, use the INDEX command with the [U] option.

Commands That Affect Indexed Databases

Since indexes are used primarily to provide fast access to the data in a database, the commands on which you can specify a WHERE condition (DISPLAY, DELETE, SELECT, and UPDATE) check to see if the database is indexed, and, if it is, whether the index can be used to make record access faster. This is done by determining if Key Items are used in the WHERE condition and there are no OR clauses.

The DELETE and SELECT commands impose an additional restriction on index usage. In order for these commands to take advantage of indexed access, the WHERE condition must specify a particular value (IS or = condition) or a limited range (surrounded by both GT and LT conditions).

The LIST and PRINT commands will also utilize an index. If a database is indexed, the information can be shown in key sequence by specifying the [I] (Index) option within the LIST or PRINT command.

Chapter Summary

- The INDEX command allows you to index a database for:
  1. Quick record access
  2. Listing and printing of a database in key item sequence without re-sorting
  3. Preventing duplicate items

- Data item names in a database are used as keys.

- New or existing databases may be indexed.

- Duplicate keys may be allowed by using the [D] option in the INDEX command.

- An index may be invalidated by using the INDEX command or by re-sorting or reorganizing the database.

- A database index is automatically updated with many Condor commands. The [U] option allows you to update an index at any time.

Appendix B

REPORT WRITER
EXAMPLE

This appendix is supplied to step you through a complete example of creating a report specification using the short method. Once you have created a report specification using the short method, you will be able to use the long method as well. The example contained is the same report found in Chapter 12, the Warehouse Report (Figure B-1).

Before beginning, you will want to create the WAREHSE database and database form as in Figure B-2. Use the command DEFINE WAREHSE. You may defer entering records into the database until a later time. (If you need assistance in creating the database and form, see Chapter 3.

```
                      WAREHOUSE UPDATE REPORT

                                           PAGE    1

Item    Item         Prev.                 Current  Unit    Total
Number  Description  On Hand  Shipped   Received  On Hand  Price   Value

DIVISION 1        WAREHOUSE 1

   1    SHOVELS        25        7          4       22    5.80    127.60
   2    TROWELS        20      100         60      -20    3.33    -66.60
   5    RED STONES     87       91         23       19    4.71     89.49
   7    YELLOW STONES  97        7         14      104    4.58    476.32
   8    BLACK STONES   50       19         23       54    4.88    263.52

                        WAREHOUSE 1        TOTAL            890.33


DIVISION 1        WAREHOUSE 2

   1    SHOVELS        27        9         14       32    5.80    185.60
   2    TROWELS        15        6          0        9    3.33     29.97
   5    RED STONES     93       75         12       30    4.71    141.30
   7    YELLOW STONES  47       14         11       44    4.58    201.52
   8    BLACK STONES    3        8         31       26    4.88    126.88

                        WAREHOUSE 2        TOTAL            685.27


                   DIVISION 1        TOTAL              1575.60


DIVISION 2        WAREHOUSE 1

  11    BLUE CEMENT    12       10          2        4    3.40     13.60
  12    YELLOW CEMENT  30       10          5       25    4.50    112.50
   8    BLACK STONES   50        1         34       83    4.50    373.50

                        WAREHOUSE 1        TOTAL            499.60


DIVISION 2        WAREHOUSE 2

  11    BLUE CEMENT    40       12          3       31    3.40    105.40

                        WAREHOUSE 2        TOTAL            105.40


                   DIVISION 2        TOTAL               605.00


                   GRAND TOTAL                          2180.60
```

Figure B-1

SCREEN FORMAT OF WAREHOUSE DATABASE

[DIVNBR] _   [WHSENBR] _
 (division number)   (warehouse number)

[NO] __   [ITEM] _____
 (item number)   (item description)

[PREQTY] __   [SHIPPED] ___   [RCVED] __
 (previous quantity)   (number shipped)   (number received)

[PRICE] _____

****

            Number of underscores in screen format

                    [DIVNBR]   1
                    [WHSENBR]   1
                    [NO]   2
                    [ITEM]   13
                    [PREQTY]   2
                    [SHIPPED]   3
                    [RCVED]   2
                    [PRICE]   6

****

Attribute summary of Data Base WAREHSE

1.DIVNBR: N,1,1,5,"   "
2.WHSENBR: N,1,0,5,"   "
3.NO: N,1,0,127,"   "
4.ITEM: AN,13,0,13,"   "
5.PREQTY: N,1,0,127,"   "
6.SHIPPED: N,1,0,127,"   "
7.RCVED: N,1,0,127,"   "
8.PRICE: $,3,.00,327.67,"   "

Record Size (Bytes) = 23
Total Records = 0

                    Figure B-2

Creating the Warehouse Report

Conventions used in the example that follows:

1. All information to be supplied by you is highlighted in bold characters.

2. Report Writer prompts are printed without bold characters.

3. Instructional comments are displayed with bold characters surrounded by parentheses.

B>> REPORT WAREHSE

```
           CONDOR SERIES 20 RDBMS REPORT WRITER
                   Version 1.xx**xx

                      Choose option

        Create New Report Specification       (C)
        Describe existing Report Specification (D)
        Revise existing Report Specification   (R)
        Print or Display Database Report       (P)

        Enter option or End <C/R>: C


Busy


Do you wish to use the short method (Y/N)? Y


Do you wish to create a new report format (Y/N)? Y
```

(The screen will clear. Duplicate the form in Figure B-3 and press the End softkey.)

```
                    WAREHOUSE UPDATE REPORT

                                         PAGE     $PAGE

Item     Item        Prev.                    Current  Unit     Total
Number   Description On Hand  Shipped  Received On Hand Price    Value
DIVISION [DIVNBR] WAREHOUSE [WHSENBR]
  [NO] [ITEM]            [PREQTY] [SHIPPED]  [RCVED] [CURQTY] [PRICE] [VALUE]

                            WAREHOUSE [WHSENBR] TOTAL           [VALUE]
                        DIVISION [DIVNBR] TOTAL                 [VALUE]
                    GRAND TOTAL                                 [VALUE]
```

Figure B-3

B-4

Do you wish to use the short method (Y/N)? Y

(Report Writer gives you a second chance to specify or not specify the short method after you have created a new report format or revised an existing one.)

>1.   WAREHOUSE UPDATE REPORT

    Describe when this line is to be printed:
        First Page only   (FP)
        Last Page only   (LP)
        Every Page Heading   (PH)
        Once Every Line   (EL)
        On Break-Before Section (BS)
        On Break-After Section   (AS)
        Enter code   [PH]: [Return]

(The above line is a page heading. It is to be printed at the top of every page.)

OK (Y/N)? Y

(If you incorrectly specified when the line is to be printed, you may enter "N" to any of the "OK (Y/N)" prompts and alter your response.)

>2.   PAGE $PAGE

    Describe when this line is to be printed:
        First Page only   (FP)
        Last Page only   (LP)
        Every Page Heading   (PH)
        Once Every Line   (EL)
        On Break-Before Section (BS)
        On Break-After Section   (AS)
        Enter code   [PH]: [Return]

OK (Y/N)? Y

>3. Item   Item   Prev.   Current   Unit   Total

   Describe when this line is to be printed:

      First Page only   (FP)
      Last Page only   (LP)
      Every Page Heading   (PH)
      Once Every Line   (EL)
      On Break-Before Section   (BS)
      On Break-After Section   (AS)

      Enter code   [PH]: `Return`

   OK (Y/N)? Y

>4. Number Description On Hand Shipped
   Received On Hand Price Value

   Describe when this line is to be printed:

      First Page only   (FP)
      Last Page only   (LP)
      Every Page Heading   (PH)
      Once Every Line   (EL)
      On Break-Before Section   (BS)
      On Break-After Section   (AS)

      Enter code   [PH]: `Return`

   OK (Y/N)? Y

>5. DIVISION [DIVNBR] WAREHOUSE [WHSENBR]

   Describe when this line is to be printed:

      First Page only   (FP)
      Last Page only   (LP)
      Every Page Heading   (PH)
      Once Every Line   (EL)
      On Break-Before Section   (BS)
      On Break-After Section   (AS)

      Enter code   [PH]   :BS `Return`

  The above line is to be printed before every section.)

   OK (Y/N)? Y

   Which data-item is to cause a section break
   to occur: **DIVNBR**
   Enter any other or <C/R>: **WHSENBR**
   Enter any other or <C/R>: `Return`

>6. [NO] [ITEM] [PREQTY] [SHIPPED] [RCVED] [CURQTY]
   [PRICE] [VALUE]

   Describe when this line is to be printed:
       First Page only   (FP)
       Last Page only   (LP)
       Every Page Heading   (PH)
       Once Every Line   (EL)
       On Break-Before Section   (BS)
       On Break-After Section   (AS)
       Enter code   [BS]   :EL [Return]
   OK (Y/N)? Y

Item 6: [CURQTY]

   Describe Type of Variable:
       Numeric   (N)
       Dollar   ($)
       Date   (J)
       Enter Code   [N]: [Return]
   [CURQTY] = PREQTY – SHIPPED + RCVED

   Define printing width - Enter column size [6] :3
   (Above is the formula value for CURQTY. CURQTY is the
   Current Quantity on hand. If you press [Return] without
   supplying the formula for this or any other computed item,
   Report Writer asks you to enter a calculation expression.)

Item 8: [VALUE]

   Describe Type of Variable:
       Numeric   (N)
       Dollar   ($)
       Date   (J)
       Enter Code   [N]: $ [Return]
   [VALUE] = CURQTY * PRICE
   Define printing width - Enter column size [5] :7
   (VALUE is the dollar value of the current inventory.)

>7. WAREHOUSE [WHSENBR] TOTAL   [VALUE]

    Describe when this line is to be printed:

        First Page only   (FP)
        Last Page only   (LP)
        Every Page Heading   (PH)
        Once Every Line   (EL)
        On Break-Before Section   (BS)
        On Break-After Section   (AS)

        Enter code   [BS]   :AS Return

    OK (Y/N)? Y

    Which data-item is to cause a section break to occur:
    WHSENBR
    Enter any other or <C/R>: Return

Item 2: [WHSENBR]

    Describe Value to Print:

        Average value   (AVG)
        Maximum value   (MAX)
        Minimum value   (MIN)
        Subtotal   (SUB)
        Accumulated total   (TOT)
        Last value read   (VAL)

        Enter Code   [SUB]   :VAL

Item 4: [VALUE]

    Define printing width - Enter column size [5]: 7

    Describe Value to Print:

        Average value   (AVG)
        Maximum value   (MAX)
        Minimum value   (MIN)
        Subtotal   (SUB)
        Accumulated total   (TOT)
        Last value read   (VAL)

        Enter Code   [SUB]:   Return

>8. DIVISION [DIVNBR] TOTAL   [VALUE]

    Describe when this line is to be printed:

        First Page only   (FP)
        Last Page only   (LP)
        Every Page Heading   (PH)
        Once Every Line   (EL)
        On Break-Before Section   (BS)
        On Break-After Section   (AS)

        Enter code   [AS]: Return

    OK (Y/N)? Y

    Which data-item is to cause a section break to occur:
    DIVNBR
    Enter any other or <C/R>: Return

Item 2: [DIVNBR]

    Describe Value to Print:

        Average value   (AVG)
        Maximum value   (MAX)
        Minimum value   (MIN)
        Subtotal   (SUB)
        Accumulated total   (TOT)
        Last value read   (VAL)

        Enter Code   [SUB]   :**VAL**

Item 4: [VALUE]

    Define printing width - Enter column size [5]: 7

    Describe Value to Print:

        Average value   (AVG)
        Maximum value   (MAX)
        Minimum value   (MIN)
        Subtotal   (SUB)
        Accumulated total   (TOT)
        Last value read   (VAL)

        Enter Code   [SUB]: Return

>9. GRAND TOTAL   [VALUE]

    Describe when this line is to be printed:
        First Page only   (FP)
        Last Page only   (LP)
        Every Page Heading   (PH)
        Once Every Line   (EL)
        On Break-Before Section   (BS)
        On Break-After Section   (AS)
        Enter code   [EL]   :LP [Return]
    OK (Y/N)? Y

Item 3: [VALUE]
    Define printing width - Enter column size [5]: 7

    Describe Value to Print:
        Average value   (AVG)
        Maximum value   (MAX)
        Minimum value   (MIN)
        Subtotal   (SUB)
        Accumulated total   (TOT)
        Last value read   (VAL)

        Enter Code   [TOT]: [Return]

Enter name of output report [WAREHSE]: [Return]
Busy

Do you wish to save the report form created (Y/N)? Y

Enter filename of report form or End <C/R>:
WHSERPT.FRM

(This is your report model form. It is advised that you save it, in case the report must be respecified. Always, the name under which you save the form MUST NOT be the same as the database form name. To use the same name would destroy the database form used for data entry and updating.)

*** You Have Finished Creating the Report Specification ***

Now, anytime you wish to print a report of the data in your database, simply select the [P] (for print) option in the Report Writer menu.

B-10

Since the short method doesn't ask you to specify line spacing for the report, you probably want to revise the specification so that it appears just like the report in Figure B-1. The report specification should end up looking like the description which immediately follows in this appendix.

Also, before you print, you should do two things:

1. Make sure you have data in your database.

2. Sort your database by the control items which are used in your report specification to designate section breaks. If your report appears to have incorrect section changes, it is probably not properly sorted.

```
Report Specification for WAREHSE

    Page Format

    Physl    Left    Right    Top    Bottom
    Lines   Margin   Margin  Margin  Margin

     66       1       80       1      60
```

>1. WAREHOUSE UPDATE REPORT

```
    Print on
    (PH) - Every Page Heading

        Line #  Line #  Space   Space   Look
        Before  After   Before  After   Ahead

          1       0       0       0       0

        Item                    Type  Format   Column  Length  Print
        WAREHOUSE               TEXT  SD         24      9      VAL
        UPDATE                  TEXT  SD         34      6      VAL
        REPORT                  TEXT  SD         41      6      VAL
```

>2. PAGE $PAGE

Print on
(PH) - Every Page Heading

Line # Before	Line # After	Space Before	Space After	Look Ahead
0	0	1	0	0

Item		Type	Format	Column	Length	Print
PAGE		TEXT	SD	40	4	VAL
$PAGE		TEXT	SD	47	5	VAL

>3. Item Item Prev. Current Unit Total

Print on
(PH) - Every Page Heading

Line # Before	Line # After	Space Before	Space After	Look Ahead
0	0	1	0	0

Item		Type	Format	Column	Length	Print
Item		TEXT	SD	1	4	VAL
Item		TEXT	SD	9	4	VAL
Prev.		TEXT	SD	22	5	VAL
Current		TEXT	SD	52	7	VAL
Unit		TEXT	SD	63	4	VAL
Total		TEXT	SD	71	5	VAL

>4. Number Description On Hand
Shipped Received On Hand Price Value

Print on
(PH) - Every Page Heading

Line # Before	Line # After	Space Before	Space After	Look Ahead
0	0	0	0	0

Item	Type	Format	Column	Length	Print
Number	TEXT	SD	1	6	VAL
Description	TEXT	SD	9	11	VAL
On	TEXT	SD	22	2	VAL
Hand	TEXT	SD	25	4	VAL
Shipped	TEXT	SD	31	7	VAL
Received	TEXT	SD	42	8	VAL
On	TEXT	SD	52	2	VAL
Hand	TEXT	SD	55	4	VAL
Price	TEXT	SD	63	5	VAL
Value	TEXT	SD	71	5	VAL

>5. DIVISION [DIVNBR] WAREHOUSE [WHSENBR]

```
Print on
(BS) - Break Before Section
Control Items
DIVNBR WHSENBR
```

Line # Before	Line # After	Space Before	Space After	Look Ahead
0	0	1	1	0

Item	Type	Format	Column	Length	Print
DIVISION	TEXT	SD	1	8	VAL
DIVNBR	DATA	SD	10	1	VAL
WAREHOUSE	TEXT	SD	19	9	VAL
WHSENBR	DATA	SD	29	1	VAL

>6. [NO] [ITEM] [PREQTY] [SHIPPED] [RCVED] [CURQTY]
   [PRICE] [VALUE]

```
Print on
(EL) - Every Line
```

Line # Before	Line # After	Space Before	Space After	Look Ahead
0	0	0	0	0

Item	Type	Format	Column	Length	Print
NO	DATA	SD	3	2	VAL
ITEM	DATA	SD	8	13	VAL
PREQTY	DATA	SD	23	2	VAL
SHIPPED	DATA	SD	32	3	VAL
RCVED	DATA	SD	44	2	VAL
*CURQTY	DATA	SD	52	3	VAL
PRICE	DATA	SD	63	6	VAL
*VALUE	DATA	SD	71	7	VAL

>7. WAREHOUSE [WHSENBR] TOTAL [VALUE]

Print on
(AS) – Break After Section
Control Items
WHSENBR

Line # Before	Line # After	Space Before	Space After	Look Ahead
0	0	1	1	0

Item	Type	Format	Column	Length	Print
WAREHOUSE	TEXT	SD	37	9	VAL
WHSENBR	DATA	SD	47	1	VAL
TOTAL	TEXT	SD	57	5	VAL
*VALUE	DATA	SD	71	7	SUB

>8. DIVISION [DIVNBR] TOTAL [VALUE]

Print on
(AS) – Break After Section
Control Items
DIVNBR

Line # Before	Line # After	Space Before	Space After	Look Ahead
0	0	1	1	0

Item	Type	Format	Column	Length	Print
DIVISION	TEXT	SD	28	8	VAL
DIVNBR	DATA	SD	37	1	VAL
TOTAL	TEXT	SD	46	5	VAL
*VALUE	DATA	SD	71	7	SUB

B-14

>9. GRAND TOTAL [VALUE]

Print on
(LP) - Last Page Only

Line # Before	Line # After	Space Before	Space After	Look Ahead
0	0	1	0	0

Item	Type	Format	Column	Length	Print
GRAND	TEXT	SD	23	5	VAL
TOTAL	TEXT	SD	29	5	VAL
*VALUE	DATA	SD	71	7	TOT

*   COMPUTED DATA-ITEMS

CURQTY=PREQTY - SHIPPED + RCVED
VALUE=CURQTY * PRICE

Done.

INDEX
TO 20-3 USER MANUAL

A

B

C

D

H

I

J

K

L

M

MS™-DOS is a trademark of Microsoft Incorporated.

Index-14

T

U

V

W

X