DB2 Query Management Facility

Using DB2 QMF

Version 8 Release 1

SC18-7445-00

DB2 Query Management Facility

Using DB2 QMF

Version 8 Release 1

SC18-7445-00

Note!

Before using this information and the product it supports, be sure to read the general information under Appendix D, “Notices,” on page 369.

First Edition (January 2004)

This edition applies to IBM DB2 Query Management Facility, a feature of Version 8 Release 1 of IBM DB2 Universal Database Server for z/OS (DB2 UDB for z/OS), 5625-DB2, and to all subsequent releases and modifications until otherwise indicated in new editions.

Contents

About this book xi
How to order DB2 QMF books xi
How to send comments xi

Chapter 1. Getting acquainted with QMF . . 1
Tables, columns, and rows 1
QMF sample tables 2
Accessing your data. 3
QMF objects 4
Starting QMF 4
Starting from the QMF home panel . . . 4
Ending a QMF session 6
Issuing QMF commands 6
Entering a command on the command line 6
Entering commands using a function key . 6
Entering commands on the command prompt panel 7
Retrieving a command 7
Setting up and changing your QMF user profile 7
Saving and retrieving objects in the database . 9
Saving a QUERY, FORM, or PROC object 10
Saving a profile 11
Saving a DATA object as a table 11
Retrieving an object from the database . . 11
Using QMF help 12
Navigating in QMF help 12
Getting help after making an error . . . 14

Chapter 2. QMF in three quick lessons . . 17
Lesson One: Finding the data you need . . . 17
Lesson Two: Choosing specific data you want 19
Selecting the columns to display 19
Selecting the rows to display 20
Lesson Three: Customizing a report 23
Where do you go from here? 26

Chapter 3. Displaying a list of database objects 27
Displaying a list of database objects using the List key 27
Displaying a list of database objects using the LIST command 30
Choosing specific objects using selection symbols 31
Entering commands on the list of database objects 32
Commands you can use on the list of database objects. 32
Using a placeholder on the list of database objects 33
Correcting errors when you enter an incorrect command. 34
Returning to the list from another QMF panel 34

Chapter 4. Viewing the data in the database using prompted query 35
Prompted query panels 35
Starting prompted query. 38
Selecting tables and columns 38
Creating a column using expressions. . . . 40
Selecting rows 43
Narrowing row selection using multiple row conditions 45
Sorting the rows in a query 47
Displaying data from more than one table . . 49
Long table names 52
Eliminating duplicate rows from the report 53
Joining multiple tables 54
Joining multiple columns 57
Making your query reusable with substitution variables 58
Running a query and displaying a report . . 61
Saving a new query 62
Canceling a running query 63
Making changes to a saved query. 64
Retrieving a query from the database . . 64
Correcting a query that does not successfully run. 64
Adding information to a query. 65
Changing information in a query 65
Deleting information from a query . . . 66
Erasing a saved query. 66
Viewing the SQL equivalent of a prompted query 67
Converting a prompted query to an SQL query 67

Chapter 5. Viewing the data in the database using SQL statements 69
Entering an object name on more than one line in the SQL query window 69
Format of SQL queries 69
Starting a SQL query 70
Entering SQL statements and running the query 71
Selecting columns and tables 71
Creating a column using expressions. . . 72
Selecting rows 73
Selecting rows that have no data 73
Selecting rows using specific character values 74
Selecting rows using conditions 74
Selecting rows using opposite conditions 74
Narrowing row selection using selection symbols 75
Narrowing row selection using multiple row conditions 76
Selecting rows If both conditions are true 76
Selecting rows If either condition is true . 76
Grouping row conditions 77
Selecting rows using the IN predicate . . 77
Eliminating duplicate rows 78
Sorting the rows in a query 78
Adding or deleting lines in a SQL query . . 79
Displaying data from more than one table . . 79
Joining columns in two or more tables . . 79
Merging data from multiple tables into a single column 81
Creating a subquery to retrieve data from more than one table 84
Writing correlation names 87
Example 1 88
Example 2 88
Example 3 88
Working with a set of values using SQL column functions 89
Working with single data values using SQL scalar functions 90
Converting a value from one data type to another 91
Formatting dates and times 92
Allowing the database requestor to control date and time format 94
Isolating the day, month, or year portion of a date 94
Isolating the hour, minute, second, or microsecond portion of a time 95
Finding the length of a value 97
Displaying parts of a value 97
Replacing null values in the report with other values 98
Nesting SQL functions 98
Nesting scalar functions within scalar functions 99
Nesting scalar functions within column functions 99
Nesting column functions within scalar functions 99
Adding and subtracting dates and times . . 100
Rules for date/time addition 100
Rules for date/time subtraction 100
Finding the number of days between two dates 100
Accounting for months with different numbers of days 101
Using durations to represent date/time intervals 101
Incrementing and decrementing dates by durations 102
Subtracting dates 104
Incrementing and decrementing times by durations 105
Subtracting times 106
Incrementing and decrementing timestamps by durations 107
Using the concatenation operator 107
Rules for concatenation 107
Examples using concatenation 108
Making your query reusable with substitution variables 108
Saving a new query 110

Chapter 6. Customizing your reports . . 113
QMF form panels 113
Using the QMF default report format . . . 114
Changing the columns on your report . . . 115
Adding a new column to a report . . . 117
Changing the order in which columns are displayed 119
Changing column headings 119
Changing column widths and space between columns 121
Changing the alignment of the column heading and the data 122
Specifying punctuation for the values in a column 123

Specifying the way QMF uses values in a column 127
Adding subtotals to a report 128
Breaking on a value to add subtotals . . 128
Specifying text for the subtotal line on a report. 129
Adding page headings and footings . . . 131
Specifying fixed columns on a report . . . 132
Displaying a representative report before you select data. 134
Refining page headings and footings . . . 136
Using a global variable in a heading or footing 137
Adding the date, time, and page number to a heading or footing 138
Changing the alignment of page headings and footings 138
Adding break segments and text to your report. 139
Adding break heading and footing text to a report 140
Refining the format of your report with detail blocks 145
Specifying text to appear at the end of your report 147
Calculating values to use in a report . . . 149
Displaying a calculated value on a report 150
Displaying special conditions on your report. 152
Mixing tabular data with reformatted text 159
Showing totals across rows in a report . . . 160
Correcting errors on a form before displaying a report 162
Saving the report form 163
Resetting the values on a form panel . . . 163
Printing your report 164
CC (carriage control). 165
Creating a report containing a limited number of ordered entries 165

Chapter 7. Displaying your report as a chart. 167
QMF chart formats 167
Where QMF report data appears on a chart 167
How QMF spaces data along the X-axis 169
Where data appears on pie charts . . . 169
Chart data size limits 170
Displaying report data as a chart. 171
Changing a chart using QMF forms. . . . 172
Changing a chart in the ICU 176
Changing a chart format 177
Saving a chart format 177
Specifying a new default chart format . . . 178
Fixing problems with charts 178
Printing your chart 179

Chapter 8. Creating a procedure to run QMF commands. 181
Creating a linear procedure 181
Guidelines for writing linear procedures 183
Creating a procedure with logic 184
Guidelines for writing procedures with logic 185
Running a procedure 186
Sharing a procedure with other QMF users 187
Creating reusable procedures with substitution variables 188
Specifying values for variables on the RUN command 188
Specifying values for variables using global variables 188
Specifying values on the RUN command prompt panel 189
Using REXX variables in procedures with logic 190
Specifying REXX variables using SAY statements and PULL statements. . . . 191
Passing values to a procedure with logic 192
REXX variable and substitution variable differences 193
Using REXX error-handling instructions in procedures with logic 194
Branching to error-handling subroutines 194
Using messages with the REXX EXIT instruction 194
Calling REXX programs from a procedure with logic 196
Calling REXX programs without substitution variables 196
Calling REXX programs that contain substitution variables 196
Connecting to a remote location from a procedure 197
Writing a procedure that creates a query . . 199
Writing a template SQL statement . . . 199
Passing variables to the template query 199
Writing a procedure that sets global variables for the template query 201
Running procedures in batch 201

Writing batch-mode procedures 202
Restrictions 202
Using IMPORT/EXPORT commands . . 204
Using the EXIT command in QMF procedures 204
Effect of errors 205

Chapter 9. Making QMF objects reusable 207
Creating, changing, and deleting variables from the global variable list 207
Changing a variable value 208
Adding a new variable 209
Removing a variable 209
Using the global variable list and the CASE option 209
Creating, changing, and deleting global variables using commands 209

Chapter 10. Creating tables 211
Planning for your table 211
Creating a table 211
Saving and appending to a table 212
Creating a copy of a table 213
Creating a view of a table 213
Creating a synonym for a table or view 213
Creating an alias for a table or view . . 214
Deleting tables, views, synonyms, and aliases 214

Chapter 11. Maintaining the data in your tables 217
Using the Table Editor 217
Deciding when to save your data . . . 217
Adding rows 218
Specifying column default and null indicator 220
Redefining the column default and null indicators 221
Adding data to long fields 221
Using the previous row as a model . . . 222
Changing rows in a table using the Table Editor 222
Selecting the rows to display 223
Making changes to the rows in a table 224
Deleting rows from a table using the Table Editor 225
Ending a Table Editor session 225
Adding rows to a table using SQL statements 225
Using the QMF DRAW command to add rows 226
Writing your own query to add rows . . 226
Changing rows in a table using SQL statements 226
Using the QMF DRAW command to change rows 227
Writing your own query to change rows 227
Deleting rows from a table using SQL statements 227
Copying rows from one table into another using SQL statements 228
Adding a new column to a table using SQL statements 228
Working with BLOB, CLOB, and DBCLOB data 229
Greater than 32 KB support for LOB data type columns 229
Authorizing access to your tables using SQL statements 229
Giving users access to your tables . . . 229
Allowing users to update specific columns in your tables 230
Revoking access to a table 230
Entering date and time values by using QMF 231

Chapter 12. Exporting and importing objects 233
Exporting QMF objects 233
Exporting QMF objects into TSO 234
Exporting QMF objects into CMS . . . 234
Exporting QMF objects into CICS . . . 234
Exporting QMF reports for use on the Internet 235
Importing QMF objects 235
Importing QMF objects from TSO . . . 236
Importing QMF objects from CMS . . . 236
Importing QMF objects from CICS . . . 236

Chapter 13. Accessing data at a remote database 239
Accessing data at a remote database using Remote Unit of Work 239
Connecting to a remote database using the QMF CONNECT command 240
Connecting to a remote database using the DSQSDBNM program parameter . . 241
Viewing the current database location . 241
Reconnecting to a location 242

Accessing data at a remote database using distributed unit of work 244
Using QMF when connected to a database through remote unit of work 244
Data 244
QMF objects 244
Tips and techniques 245
CURRENT SQLID 245
Function keys and synonyms 245
Procedures, forms, and queries 245
Commands 245
Using the QMF CONNECT command to connect to databases 246
Example 1: How connecting to a new location affects your user ID 246
Example 2: Connecting to like databases 247
Example 3: Connecting to unlike databases 248
Example 4: Connecting to a new location using remote unit of work and distributed unit of work 249

Chapter 14. National Language support in QMF 251

Bilingual command support 251
Exporting and importing bilingual forms 251
Defining DBCS data 252
How DBCS data looks when displayed 252
How DBCS data changes the length of names and fields 253
Data types you can use with DBCS data 253
Using DBCS data in QMF 256
Using DBCS data in commands and procedures 256
Using DBCS data in Input fields 257
Using DBCS data in queries 257
Using DBCS in form panels 258
Exporting DBCS data 262
Importing DBCS data 262
Printing DBCS reports 262

Chapter 15. Using QMF with other products 265

Using DXT end user dialogs 265
Editing objects from outside QMF using ISPF 267
Using ISPF from QMF 269
Inserting a QMF report in a document . . . 270
Formatting the report 270
Inserting a report 271
Information about your editor 273
Restrictions on the document interface 274
Using the QMF document interface 274
Accessing QMF from a VM editor . . . 275
Accessing a VM editor from QMF . . . 276
Accessing QMF from a z/OS editor. . . 276
Accessing a z/OS editor from QMF. . . 278

Chapter 16. Using DB2 stored procedures with QMF 279

Developing the stored procedures to run from QMF 279
Running a CALL statement from the SQL QUERY panel 279
Format of the CALL statement 279
Examples of running a stored procedure . . 280
Running a stored procedure that returns no output parameters 280
Running a stored procedure that contains input and output parameters 281
Running a stored procedure that returns a result set. 282
Running a stored procedure that returns multiple result sets 283
Using QMF forms with result sets returned from stored procedures 284

Chapter 17. QMF sample tables 285

Q.APPLICANT 285
Q.INTERVIEW. 286
Q.ORG 287
Q.PARTS. 288
Q.PRODUCTS 288
Q.PROJECT. 289
Q.STAFF 290
Q.SUPPLIER 291

Appendix A. Query-by-example 293

Displaying the QBE query panel 293
Running and saving queries 293
Listing queries. 293
Drawing example tables 294
Presenting all the columns of a table . . . 295
Presenting certain columns of a table . . . 295
Changing the order of columns 296
Presenting certain rows of a table 296
Presenting rows that contain a certain value 296
Defining example elements 297
Writing expressions 298

Using unnamed columns in an example table 300
Adding conditions to the example table 302
Adding a CONDITIONS box 302
Adding a target table 304
Eliminating duplicate rows. 305
Presenting data from more than one table 306
Writing queries to be shared 309
Model query 309
Substitution variables 309
The USER variable 311
QMF commands specific to QBE 312
CONVERT command 312
DELETE command 314
DRAW command 314
ENLARGE command 316
REDUCE command 318
Keyword reference 320
ALL. — Display duplicate rows 321
AND — Present on two conditions 321
Two conditions on one row 321
AO., AO(n). — Sort rows in ascending order 322
Order by more than one column 322
AVG. — Calculate the average value . . 323
BETWEEN x AND y — Present values within a range 324
COUNT. — Count the number of values in a column. 326
D. — Delete rows from a table 326
DO., DO(n). — Sort rows in descending order 327
Order by more than one column 328
G. — Grouping data 329
I. — Insert rows into a table 330
IN (x,y,z) — Present certain values in a set 331
LIKE — Present on part of a value . . . 332
Like any single character (underscore) 332
Like any number of characters (percent sign) 332
Data-type dependencies. 333
MAX. — Calculate the maximum value 333
MIN. — Calculate the minimum value 334
NOT — Present on the opposite of the condition 335
NULL — Present rows with missing entries 337
Unknown values 338
OR — Present on either of two conditions 338
P. — Present data in a table 339
Present all the columns in a table . . . 339
Present some of the columns in a table 340
Present some of the rows in a table . . . 340
Present data from multiple tables . . . 341
Present data dependent on non-presented data 341
SUM. — Calculate the total 342
U. — Update a row in a table. 343
UNQ. — Eliminate duplicate rows . . . 344
USER — Present rows with a value equal to a user identification 345
+, –, *, / — Calculated values 345
=, ¬=, >, < — Equality and inequality . . 347
QBE exercises with solutions 348
Exercises. 348
Solutions to exercises 351

Appendix B. QMF functions that require specific support 365
QMF functions not available in CICS . . . 365

Appendix C. The DB2 QMF High Performance Option 367
QMF HPO/Manager. 367
QMF HPO/Compiler 367

Appendix D. Notices 369
Trademarks 371

Glossary of Terms and Acronyms . . . 373

Bibliography 387
CICS publications. 387
COBOL publications 387
DB2 Universal Database for z/OS publications. 387
Document Composition Facility (DCF) publications. 388
Distributed Relational Database Architecture (DRDA) publications. 388
Graphical Data Display Manager (GDDM) publications. 388
HIgh Level Assembler (HLASM) publications. 389
Interactive System Productivity Facility (ISPF) publications 389
OS/390 publications 389
OS PL/I publications 389
REXX publications 390

VM/ESA publications 390
VSE/ESA publications 390

Index 391

About this book

DB2 Query Management Facility (QMF) is a database application program that enables you to easily create, change, or retrieve data from a computer database. After retrieving the data, you can format it into reports or charts.

Using DB2 QMF is for new and occasional QMF users. Some knowledge of databases is helpful, but not necessary. This book introduces you to basic QMF tasks with examples that you can use and adapt for your own work. As you read, try out the examples in the book with QMF to produce the described results.

This book is designed to be used with *DB2 QMF Reference*, which contains detailed information about QMF commands and panels. More information about any topic is also available from QMF's extensive online help.

The first part of the book explains basic concepts involved in using QMF to work with databases. The second part of the book explains tasks you can perform with QMF with step-by-step instructions. The appendixes explain how to use QMF's Query-By-Example feature, show the sample tables that QMF provides, list QMF functions that require specific support, and describe the DB2 QMF High Performance Option (HPO).

Examples in this book were created by using QMF with a DB2 Server for VSE or VM™ database. The results you see in your own environment might be slightly different.

How to order DB2 QMF books

To order hard copies, contact your IBM representative or visit the IBM Publications Center on the world wide web at: http://www.elink.ibmlink.ibm.com/applications/public/applications /publications/cgibin/pbi.cgi. Or, you can call 1-800-879-2755 in the United States or any of its territories.

How to send comments

Your feedback is important in helping to provide the most accurate and high-quality information. If you have any comments about this book, go to http://www.ibm.com/software/data/qmf/support.html, and click on Feedback.

Chapter 1. Getting acquainted with QMF

Query Management Facility (QMF), lets you work with data that is stored in relational databases, including the following IBM® databases:

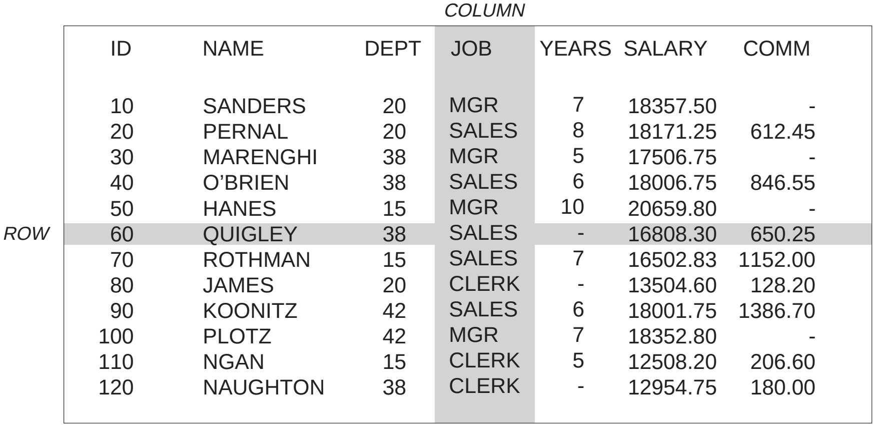
- DB2™ for z/OS
- DB2 Server for VSE and VM
- DB2 for iSeries®

This chapter discusses some basic concepts in using QMF. For example, it includes topics such as the following:

- Understanding how data is arranged in databases
- Starting and ending a QMF session
- Issuing commands
- Setting up your QMF profile
- Getting help while you are using QMF

Tables, columns, and rows

In QMF, data is arranged in tables. These tables have names; you must know the names of the tables that contain the data you need. The data in a table is arranged in columns and rows. Figure 1 shows an example.



ID	NAME	DEPT	JOB	YEARS	SALARY	COMM
10	SANDERS	20	MGR	7	18357.50	-
20	PERNAL	20	SALES	8	18171.25	612.45
30	MARENGHI	38	MGR	5	17506.75	-
40	O'BRIEN	38	SALES	6	18006.75	846.55
50	HANES	15	MGR	10	20659.80	-
60	QUIGLEY	38	SALES	-	16808.30	650.25
70	ROTHMAN	15	SALES	7	16502.83	1152.00
80	JAMES	20	CLERK	-	13504.60	128.20
90	KOONITZ	42	SALES	6	18001.75	1386.70
100	PLOTZ	42	MGR	7	18352.80	-
110	NGAN	15	CLERK	5	12508.20	206.60
120	NAUGHTON	38	CLERK	-	12954.75	180.00

Figure 1. Data in tables is arranged in columns and rows.

Columns:

- Appear vertically on the screen

- Contain data of the same kind
- Have names, which appear at the top as headings

Rows:

- Appear horizontally on the screen
- Contain different kinds of data about a single thing
- Have no names

In QMF, you normally refer to tables by using two-part names—a table identifier and an owner identifier, separated by a period. For example, the exercises in this book use a table named Q.STAFF, where `STAFF` identifies the table, and `Q` identifies the owner of that table. For a table named JOHN.ACCOUNTS, `ACCOUNTS` is the table identifier, and `JOHN` is the owner identifier. Normally, the person who creates a table is the owner of that table. QMF identifies the owner of the table by using the user ID of the individual who created it. The owner of a table can authorize others to access the information in the table. When referring to your own tables, you can leave out the owner identifier. QMF assumes that you are referring to a table that you own.

If your installation supports three-part names, also known as distributed unit of work, you can use a table from a remote location by including a location identifier. See your QMF administrator to find out whether your installation supports three-part names. For example, NEW_YORK.JOHN.ACCOUNTS refers to an ACCOUNTS table owned by JOHN and is located at a remote DB2 database known to your communications network as NEW_YORK. For more information on remote data access in the QMF environment, see the *DB2 QMF Reference*.

You do not need to use a location identifier with local tables. In this book, you use local tables with two-part names.

Column names: Columns are usually referred to by name. You will learn how to find out column names for tables in “Selecting tables and columns” on page 38 and “Selecting columns and tables” on page 71.

QMF sample tables

QMF provides six sample tables that you can use while learning QMF before you begin working with your own tables. The sample tables are used throughout this book as examples. They contain information about the J & H Supply Company, a fictitious electrical parts manufacturer.

Table 1 on page 3 describes what each sample table contains. You can see all the data in the tables in Chapter 17, “QMF sample tables,” on page 285.

Table 1. QMF provides six sample tables to use while learning the program.

Sample table name	Contains information about
Q.STAFF	The employees of the J & H Supply Company
Q.ORG	Organization of the J & H Supply Company by department (within division)
Q.PRODUCTS	Products produced by the J & H Supply Company
Q.PROJECT	J & H Supply Company projects
Q.SUPPLIER	Companies who supply materials to J & H Supply Co.
Q.PARTS	Materials supplied to J & H Supply Co.

Accessing your data

To get information from a database, you write a query, which is a request to the database to retrieve a specific set of data. Use QMF to "communicate" your request for information in three different ways. Each of the three ways has its own rules:

Prompted query

Prompted query is an easy-to-use query method that displays prompt panels to help you choose the specific information you are looking for. You do not need to know the syntax for your database request; it does the work for you by converting your request into a language the database can understand. It is especially suited for beginners and occasional QMF users. You can learn how to use Prompted Query to access your data in Chapter 4, "Viewing the data in the database using prompted query," on page 35.

SQL (Structured Query Language)

SQL is a powerful query language that lets you define, retrieve, change, and authorize access to data. It has a specific syntax you must follow so the database can process your request. It is especially suited for users who must work with large amounts of data, and who work with QMF frequently. You can learn how to use SQL to access in your data in Chapter 5, "Viewing the data in the database using SQL statements," on page 69.

Query-by-Example (QBE)

QBE is a graphical query method that lets you retrieve and change data with just a few keystrokes. Appendix A, "Query-by-example," on page 293 describes how to use Query-by-Example to create queries.

QMF objects

QMF stores information as QMF objects. Some objects, such as queries, are actually stored in the database. Other objects, such as reports and charts, exist only in temporary storage while you work on them.

There are seven QMF objects, as shown in Table 2.

Table 2. QMF stores information as seven kinds of QMF objects.

Object	Contains
QUERY	Specifications for selecting the data you want to display
DATA	Data you select using the query, or data you import from outside QMF
FORM	Specifications for displaying the data you select
REPORT	Formatted data produced when you run a query to retrieve data
CHART	Graphic display of formatted report data
PROCEDURE	A series of QMF commands that you can issue with a single RUN command- PROC is the abbreviation for PROCEDURE in QMF
PROFILE	Specifications for your QMF user session

Each QMF object in the database has an owner who is usually the person who saved the object in the database. You cannot use an object unless you either own it, or the owner has authorized you to use it, or you are a QMF administrator. A QMF administrator can use any object that is owned by anyone.

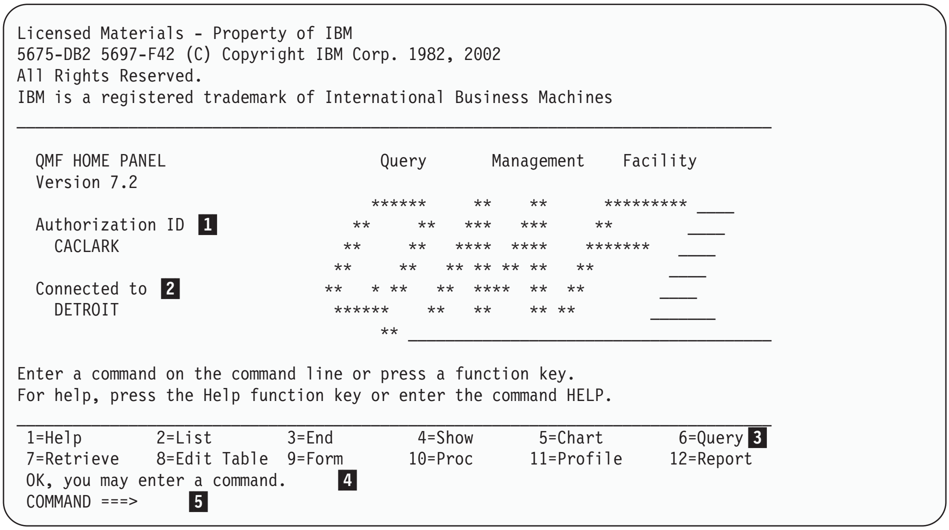
Starting QMF

Each enterprise customer usually has one or more people who perform set-up and maintenance tasks for QMF. These people are called QMF administrators.

Your QMF administrator can tell you how to start a QMF session and give you a user identification number or code word—your QMF user ID. Check with your QMF administrator if you have any questions about getting started with QMF.

Starting from the QMF home panel

After you start QMF, you will see the QMF Home panel:



```
Licensed Materials - Property of IBM
5675-DB2 5697-F42 (C) Copyright IBM Corp. 1982, 2002
All Rights Reserved.
IBM is a registered trademark of International Business Machines
_________________________________________________________________________________

  QMF HOME PANEL                       Query       Management    Facility
  Version 7.2
                                      ******     **    **      ********* ____
  Authorization ID 1                **     **   ***   ***     **        ____
    CACLARK                        **     **   ****  ****    *******   ____
                                  **     **   ** ** ** **   **        ____
  Connected to  2                **   * **   **  ****  **  **        ____
    DETROIT                       ******    **   **    ** **         _______
                                       ** _______________________________________

Enter a command on the command line or press a function key.
For help, press the Help function key or enter the command HELP.
_________________________________________________________________________________
 1=Help        2=List        3=End         4=Show       5=Chart        6=Query 3
 7=Retrieve    8=Edit Table  9=Form       10=Proc      11=Profile     12=Report
 OK, you may enter a command.      4
 COMMAND ===>      5
```

Figure 2. The QMF Home panel

On the Home panel you see:

1 Authorization ID

The user ID you used to log on to QMF

You can access only objects, such as queries, forms, or procedures created using this ID and objects to which this ID has been granted access. Any object you create during a session belongs to this user ID.

2 Connected to

The name of the database to which you are connected

3 Function keys

A function key performs a single operation

The function of each key depends on the panel you are looking at. You can reduce keystrokes by pressing a single function key to start the operation you want.

If your QMF administrator tailors your function keys, they may not be the same as the examples presented in this book.

Function keys that are labeled refer to the labels next to function keys at the bottom of the QMF panel.

4 The message line

On this line, QMF tells you what was accomplished by the last operation you started, or what you can do next.

5 The command line

If no function key is programmed to start the operation you want to do next, you can tell QMF what to do by entering a command on this line after the arrow. When you see “Enter a command”in this book, you are to type the command on the command line and press Enter. Some QMF panels display SCROLL ===> PAGE on the right side of the command line. You can type another value over the PAGE value to change the scrolling behavior when you press the Forward or Backward function keys. See *DB2 QMF Reference* or QMF’s online help for the Forward or Backward commands for more information.

Ending a QMF session

You can end a QMF session from the Home panel, or you can bypass the Home panel and end the session directly.

To end a QMF session from the Home panel:

1. From any QMF panel, press the End function key. The QMF Home panel appears.
2. Press the End key again to end the QMF session.

To bypass the Home panel and end a QMF session: Enter `EXIT` on the command line.

Issuing QMF commands

You can issue QMF commands in three ways:

- Enter a command on the command line
- Press a function key
- Specify a command on a prompt panel

Entering a command on the command line

You can enter a command on any QMF panel that has a command line. You can enter a command in full, or use the minimum unique abbreviation for any command that can be abbreviated.

To enter a command in full: To display a report that contains data from the sample table Q.STAFF, enter:

```
DISPLAY Q.STAFF
```

To enter a command using the minimum unique abbreviation: Enter the previous command as:

```
DI Q.STAFF
```

Entering commands using a function key

QMF commands can be entered by using a function key. Each QMF panel has a default set of function keys that can be changed to your preferences. The keys you see on your QMF panels might be different than the ones in this book.

Entering commands on the command prompt panel

Command prompt panels ask you for the required information that you need to enter a QMF command.

To see a command prompt panel: Enter the command and follow it with a space and a question mark.

Enter `RUN ?` to display the RUN Command Prompt panel.

To see a command prompt panel while QMF displays an error message: If you entered a command, but received an error message on the command line, press Enter while QMF displays the message. QMF displays the command prompt panel so you can enter the correct information.

Retrieving a command

To retrieve a command entered on the command line: Enter `RETRIEVE` or `?` to see the last command entered.

You can use multiple questions marks. `?` retrieves the last command you entered, `??` retrieves the command before that, and so on.

Setting up and changing your QMF user profile

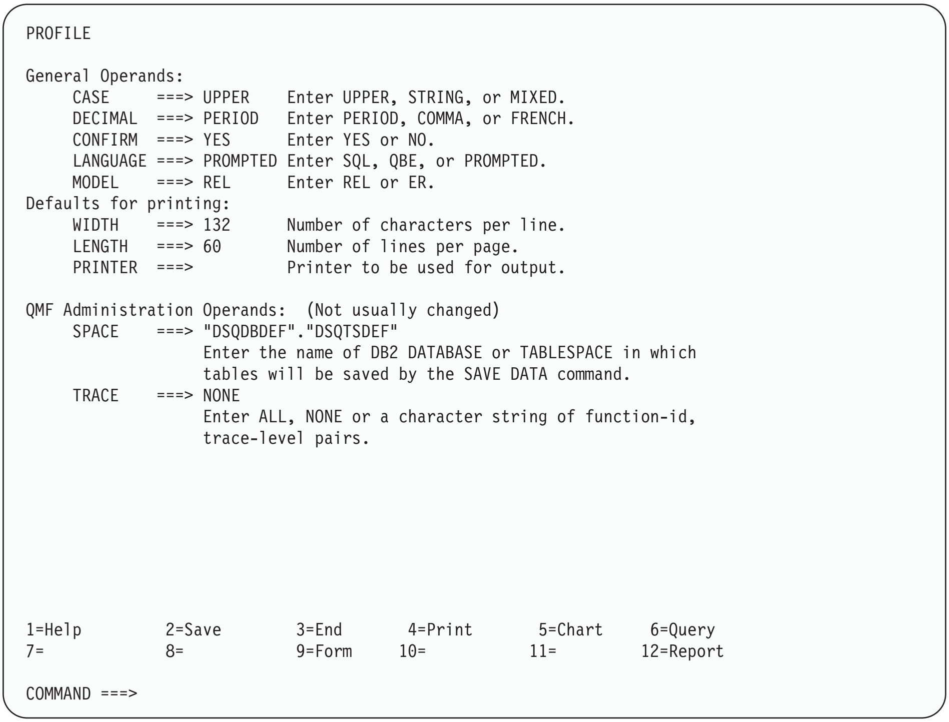
Your QMF user profile tells QMF how you want to display information and process commands, and what printer to use when you print reports or charts. Your user profile is the same as the system default when you first begin to use QMF. You can change the information in your profile to match your specific needs at any time.

To display your QMF user profile:

- Press the Profile function key.

  Or you can type the following command:
- Enter `SHOW PROFILE` (or `SH PROF`) on the command line.

The Profile panel displays, as shown in Figure 3 on page 8.



```
PROFILE

General Operands:
     CASE     ===> UPPER    Enter UPPER, STRING, or MIXED.
     DECIMAL  ===> PERIOD   Enter PERIOD, COMMA, or FRENCH.
     CONFIRM  ===> YES      Enter YES or NO.
     LANGUAGE ===> PROMPTED Enter SQL, QBE, or PROMPTED.
     MODEL    ===> REL      Enter REL or ER.
Defaults for printing:
     WIDTH    ===> 132      Number of characters per line.
     LENGTH   ===> 60       Number of lines per page.
     PRINTER  ===>          Printer to be used for output.

QMF Administration Operands:  (Not usually changed)
     SPACE    ===> "DSQDBDEF"."DSQTSDEF"
                   Enter the name of DB2 DATABASE or TABLESPACE in which
                   tables will be saved by the SAVE DATA command.
     TRACE    ===> NONE
                   Enter ALL, NONE or a character string of function-id,
                   trace-level pairs.









1=Help         2=Save        3=End       4=Print       5=Chart     6=Query
7=             8=            9=Form     10=           11=         12=Report

COMMAND ===>
```

Figure 3. The QMF Profile panel controls how information is displayed.

To change your QMF user profile: Position the cursor on any value in your QMF profile and type over it with the value you want.

If you press the End function key, the changes you make to your profile remain in effect only until you end your QMF session.

To save your QMF user profile: When you have all the desired values set the way you want, press the Save function key or enter `SAVE` or `SAVE PROFILE` on the command line. QMF stores the changed profile in the database, and uses the changed profile the next time you begin a QMF session.

You can change any of the profile values at any time. Table 3 shows some of your options for changing your profile.

Table 3. You can set your QMF profile with your preferences.

Profile value	Explanation
CASE ===> UPPER	QMF only recognizes commands in uppercase. If you want to enter information in mixed or lowercase, change your profile to show `CASE ===> UPPER`. QMF will convert your data entries to uppercase.

Table 3. You can set your QMF profile with your preferences. (continued)

Profile value	Explanation
DECIMAL ===> PERIOD	Although other indicators (such as a comma) are available, this book uses a period as a decimal point indicator.
CONFIRM ===> YES	When `CONFIRM ===> YES` is specified, QMF displays a confirmation panel before a command changes or replaces an object in the database. For the exercises in this book, be sure `YES` is specified.
LANGUAGE ===> PROMPTED	Choose `LANGUAGE ===> PROMPTED` when you want QMF to prompt you for the information you need to write a query. Choose `LANGUAGE ===> SQL` when you want to write queries directly in SQL. Choose `LANGUAGE ===> QBE` when you want to use the QMF Query-by-Example feature.

For more information about changing your profile, see the online help or the *DB2 QMF Reference* for the SET PROFILE command.

Saving and retrieving objects in the database

When you display or work with any QMF object, QMF places a copy of that object in a temporary storage area. There is one temporary storage area for each type of object, so you can only have one of each object in temporary storage at a time. The name of the temporary storage area is the same as the object type. QMF places QUERY objects in the QUERY temporary storage area, REPORT objects in the REPORT temporary storage area, and so on.

Changes that you make to objects in temporary storage do not change the actual objects stored in the database unless you save those objects when you finish working with them.

For example, you will lose changes to an object in temporary storage if you end a QMF session without saving it. You will also lose changes if you display another object of the same type before saving the one you are currently working with.

You cannot save a REPORT object. Instead, you save the query and form that produce the report. To save a CHART, you save the data and the chart format. You will learn more about saving charts in Chapter 7, “Displaying your report as a chart,” on page 167.

Saving a QUERY, FORM, or PROC object

You can save a query, form, or procedure by entering the SAVE command on the QMF command line in one of the following ways:

If you are on the QUERY, FORM, or PROC panel, and you want to save the currently displayed object, enter:

```
SAVE
```

If the object is an existing one that you loaded from the database, QMF saves it using its existing name.

If the object is new, QMF prompts you for a name for the object.

You can also enter the following command:

```
SAVE AS objectname
```

where *objectname* is the name you want to assign to the object.

If the object is an existing one that you loaded from the database, QMF saves it under the new name. The object stored in the database with the old name remains unchanged.

If you are on any QMF panel, and you want to save a currently loaded object even though it is not currently displayed, enter:

```
SAVE object
```

where *object* is the type of object you want to save. For example, if you are on the FORM panel, and you want to save the query that is currently loaded in temporary storage, enter `SAVE QUERY`.

If the object is an existing one that you loaded, QMF saves it using its existing name.

If the object is new, QMF prompts you for a name for the object.

If you are on any QMF panel, and you want to save a currently loaded object under a new name, enter:

```
SAVE object AS objectname
```

where *object* is the type of object you want to save, and *objectname* is the name you want to assign to the object.

You can use this command syntax for new or existing objects. If the object is an existing one, QMF saves the currently loaded object with the new name. The object stored in the database with the old name remains unchanged.

If you want to save an object and share it with other users, add the SHARE=YES parameter to the SAVE command you are using:

```
SAVE (SHARE=YES
SAVE AS objectname (SHARE=YES
SAVE object (SHARE=YES
SAVE object AS objectname (SHARE=YES
```

If you issue a SET GLOBAL command with the value DSQEC_SHARE=1 prior to issuing the SAVE command, you do not need to include the SHARE=YES parameter.

Saving a profile

You can have only one PROFILE object in the database. You can enter either of the following commands on the QMF command line of the PROFILE panel to save your profile:

```
SAVE
SAVE PROFILE
```

You can also enter `SAVE PROFILE` on the command line of any QMF panel.

Saving a DATA object as a table

A DATA object is not stored in the database, but is created for you to work with temporarily when you create and run a query. All data is stored in the database in tables. If you want to save the data in a DATA object, you must save it as a table.

To save a DATA object as a table, enter the following command on the QMF command line:

```
SAVE DATA AS tablename
```

where *tablename* is the name you want to assign to the new table.

Retrieving an object from the database

You can retrieve any object from the database after you save it.

To retrieve an object from the database, enter the following command on the QMF command line:

```
DISPLAY objectname
```

where *objectname* is the name of the specific object.

For example, to retrieve an object that is named MYQUERY, enter `DISPLAY MYQUERY`.

You can also enter the following command:

```
DISPLAY object objectname
```

where *object* is the type of object you want to retrieve, and *objectname* is the name of the specific object.

For example, to retrieve and display a procedure that is named MYPROC from the database, enter `DISPLAY PROC MYPROC`.

Using QMF help

You can display online help to learn about QMF. You can also display help for writing queries, formatting reports, editing a table, or creating procedures. QMF provides online help for tasks, commands, and error messages. QMF help lets you see information about what you are doing without leaving QMF. This information appears on the bottom half of your screen in a scrollable window. To see help information, press the Help function key from anywhere within QMF.

If you are a new or occasional QMF user, you might find the Learning About QMF menu helpful. This topic includes most of the task information in this book, *Using DB2 QMF*, in online form.

Navigating in QMF help

The QMF main help menu provides a list of general topics. From this menu, you can choose more specific topic panels. Figure 4 on page 13 shows how QMF help is organized.

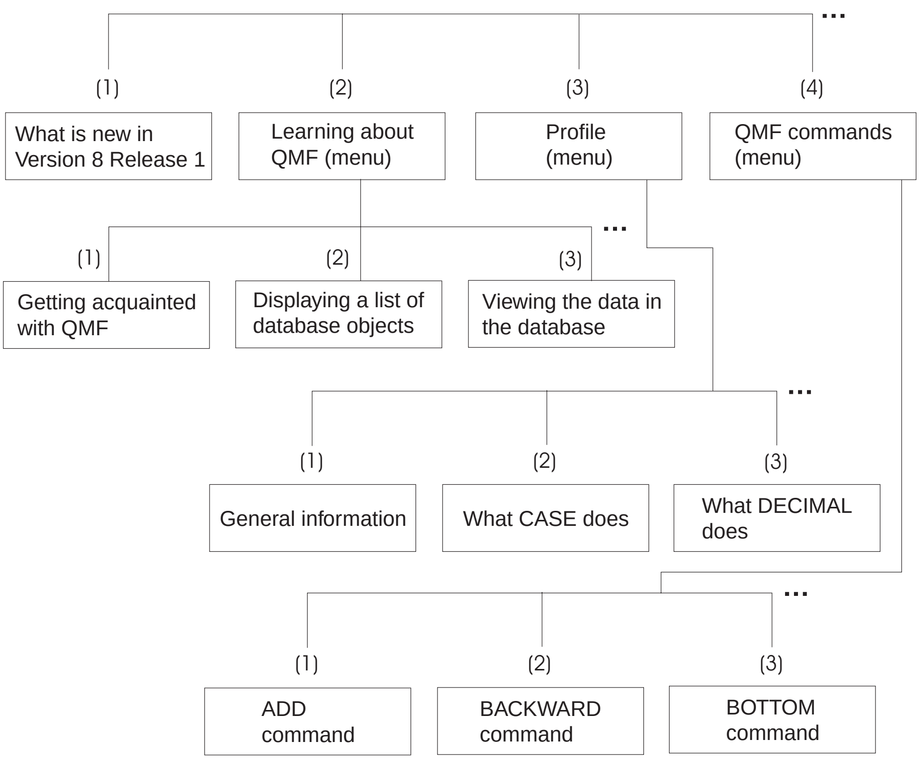


Figure 4. QMF help is organized from general to specific topics

QMF help varies slightly in how it allows you to access information, depending on what part of QMF you are using. When you press the Help function key:

- In most parts of QMF, you see a menu related to the QMF panel you are using. Select the topic or area of information that you want to see.
- In Prompted Query or the Table Editor (QMF's mode for adding or changing tables), you immediately see information specifically related to the part of the process you are working on.
- While you are creating an SQL query, you see a table of contents from which you can select the information you want to see.
- While you are working on form panels, you see information specifically related to the field on which you are working.
- In most parts of QMF, additional information related to a QMF error message will be shown.

The following functions are available on function keys to help you navigate through QMF help:

Exit Immediately removes all help panels and activates the underlying QMF panel

More Help
Shows you a menu of panels that are related to the help panel currently displayed (available for selected topics)

Menu Displays either the most recently displayed menu, or the menu for the underlying QMF panel

You can return through higher-level menus until you reach the Help main menu by repeatedly pressing the Menu function key.

Backward
Scrolls backward through the panel

Forward
Scrolls forward through the panel

Keys Lists the functions of the keys for the underlying QMF panel

Cancel
Removes one help panel at a time.

You can return to the underlying QMF panel by repeatedly pressing the Cancel function key.

Switch
On help panels for some specific topics, activates the underlying QMF panel- you can enter commands on the command line of the QMF panel while the help panel is still displayed

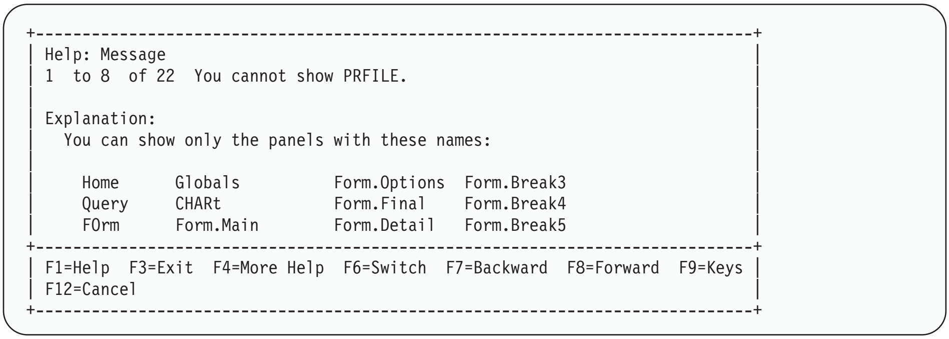
Getting help after making an error

When you misspell a command or try to run a query that is worded incorrectly, QMF presents a brief explanation of the error on the message line of the panel. If you need more information about the error, you can seek additional help by pressing the Help function key or by entering the HELP command on the command line. A panel will display a detailed explanation of the error and a suggestion on how to correct the error.

Type the command `SHOW PRFILE` in uppercase characters on the command line of the Home panel; you will see the error message:

```
You cannot show PRFILE.
```

To find out more, press the Help function key.



```
+------------------------------------------------------------------------------+
| Help: Message                                                                |
| 1  to 8  of 22  You cannot show PRFILE.                                      |
|                                                                              |
| Explanation:                                                                 |
|   You can show only the panels with these names:                             |
|                                                                              |
|     Home      Globals          Form.Options  Form.Break3                     |
|     Query     CHARt            Form.Final    Form.Break4                     |
|     FOrm      Form.Main        Form.Detail   Form.Break5                     |
+------------------------------------------------------------------------------+
| F1=Help  F3=Exit  F4=More Help  F6=Switch  F7=Backward  F8=Forward  F9=Keys |
| F12=Cancel                                                                   |
+------------------------------------------------------------------------------+
```

Figure 5. QMF displays an error message when it can not execute a command.

To scroll through the help panel, press the Forward function key. The help explains that the correct command is `SHOW PROFILE`.

If you want additional information, press the More Help function key. What you see depends on what you were doing before you first called for help. For example, if you were editing a table (in the Table Editor), pressing the More Help function key from the error message help panel would display the help panels associated with different aspects of the Table Editor. This is the same help that displays if you press the Help function key directly from the Table Editor. This list displays directly on top of the previous panel, the error message help panel, or any panel that is displayed from the Table Editor list.

You may also be presented with error messages as a result of a query being cancelled by the QMF Governor or the DB2 QMF High Performance Option Governor.

Chapter 2. QMF in three quick lessons

Working with QMF usually requires three basic tasks: finding data you need, selecting specific items from that data, and turning the data into a report. This chapter gives you three quick lessons on how to do these tasks using the data in the QMF sample tables.

As you read the other chapters in this book, you will learn methods that will refine the execution of these tasks. You can perform many of these tasks in a Windows environment by using the DB2 QMF for Windows feature.

Before starting the lessons, make sure that you are familiar with the QMF terms and concepts that are explained in Chapter 1, "Getting acquainted with QMF," on page 1.

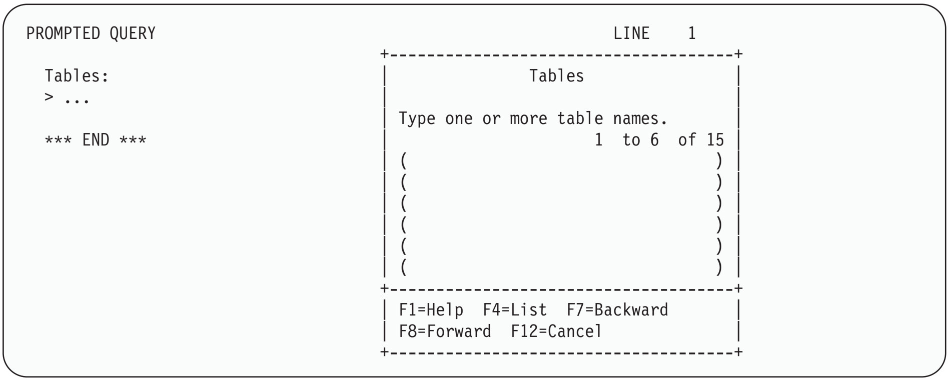
Lesson One: Finding the data you need

In this lesson, you will create a query to show the records for all the clerks in the Q.STAFF table. The Q.STAFF table is one of the sample tables that comes with QMF. Use the Prompted Query method; it is the easiest method to start with.

Select the Q.STAFF table.

To select a table:

1. On the command line of the QMF Home panel, enter:
   ```
   RESET QUERY (LANG=PROMPTED
   ```
   The Prompted Query panel displays the Tables panel:



```
PROMPTED QUERY                                        LINE    1
                                      +--------------------------------------+
  Tables:                             |               Tables                 |
  > ...                               |                                      |
                                      | Type one or more table names.        |
  *** END ***                         |                      1  to 6  of 15  |
                                      | (                                 )  |
                                      | (                                 )  |
                                      | (                                 )  |
                                      | (                                 )  |
                                      | (                                 )  |
                                      | (                                 )  |
                                      +--------------------------------------+
                                      | F1=Help  F4=List  F7=Backward        |
                                      | F8=Forward  F12=Cancel               |
                                      +--------------------------------------+
```

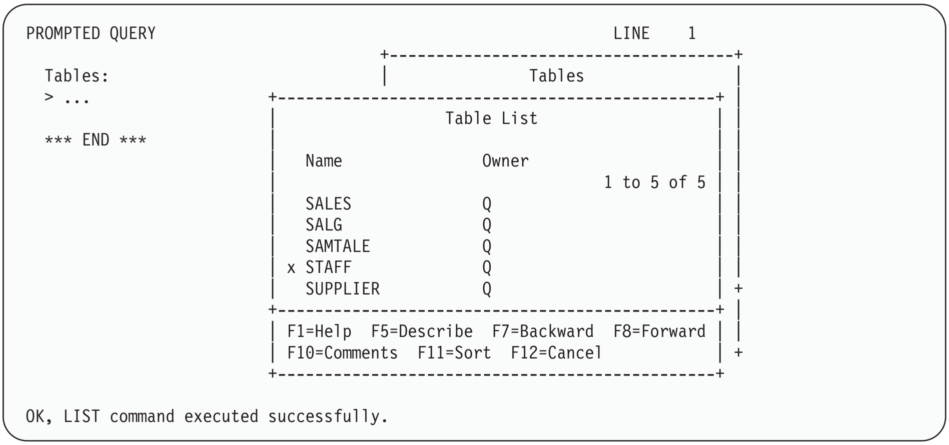
Figure 6. The Tables panel

Normally, you would type the name of the table you want to use, `Q.STAFF` on the Tables panel and press Enter. However, for this lesson you will choose the table from a list.

In addition, you will limit the list to only the tables that belong to user Q and that begin with the letter S. To do that, you use selection criteria. You will learn more about selection criteria in Chapter 3, “Displaying a list of database objects,” on page 27. In this case, you will use the characters `q.s` followed by a % sign.

2. Type `q.s%` on the first line of the Tables panel.
3. Press the List function key.

   The Table List panel displays with the names of all the tables that are owned by user Q and begin with S.
4. Move the cursor to **STAFF** and type `x`.



```
PROMPTED QUERY                                        LINE    1
                                      +--------------------------------------+
  Tables:                             |               Tables                 |
  > ...                   +-------------------------------------------------+ |
                          |                  Table List                     | |
  *** END ***             |                                                 | |
                          |   Name                Owner                     | |
                          |                                 1 to 5 of 5     | |
                          |   SALES               Q                         | |
                          |   SALG                Q                         | |
                          |   SAMTALE             Q                         | |
                          | x STAFF               Q                         | |
                          |   SUPPLIER            Q                         | +
                          +-------------------------------------------------+ |
                          | F1=Help  F5=Describe  F7=Backward  F8=Forward   | |
                          | F10=Comments  F11=Sort  F12=Cancel              | +
                          +-------------------------------------------------+

OK, LIST command executed successfully.
```

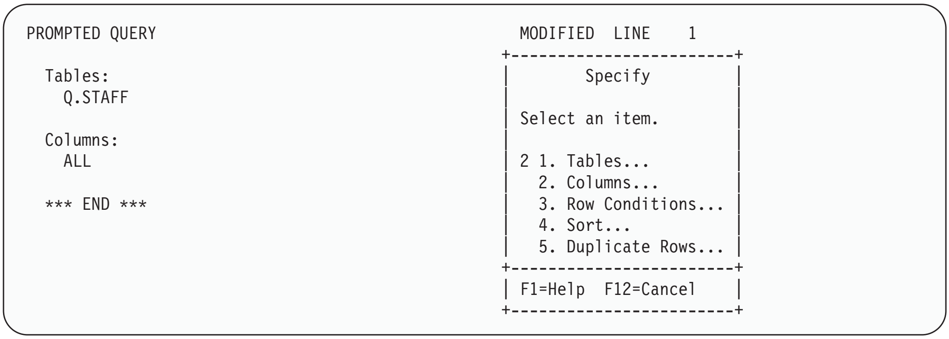
Figure 7. The Table List panel

5. Press Enter.

   The Tables panel displays again with Q.STAFF on the first line.
6. Press Enter to select the Q.STAFF table.

Q.STAFF is displayed under the Tables heading on the left side of the Prompted Query panel. This is called the echo area. The echo area shows you each part of the query as you create it.

The Specify panel is also displayed. Now that you have selected a table, you will use the Specify panel to create the rest of the query.



```
PROMPTED QUERY                                      MODIFIED  LINE    1
                                                  +------------------------+
  Tables:                                         |        Specify         |
    Q.STAFF                                       |                        |
                                                  | Select an item.        |
  Columns:                                        |                        |
    ALL                                           | 2 1. Tables...         |
                                                  |   2. Columns...        |
  *** END ***                                     |   3. Row Conditions... |
                                                  |   4. Sort...           |
                                                  |   5. Duplicate Rows... |
                                                  +------------------------+
                                                  | F1=Help  F12=Cancel    |
                                                  +------------------------+
```

Figure 8. The Specify panel

Lesson Two: Choosing specific data you want

After you locate the data that you want, you can choose specific items from the table. To do this, you select the columns and rows you want to display.

Selecting the columns to display

Columns contain data of the same kind for each individual entry in the table. For example, the column called JOB contains data about the job title of each person in the Q.STAFF table. You want to see several columns from the table for this lesson.

Choice 2, **Columns** is already selected, because selecting columns is usually the next step in creating a query.

If you look in the echo area on the Specify panel, you can see the word **ALL** appears directly under the **Columns** heading. If you do not select specific columns, QMF automatically selects all the columns in the table.

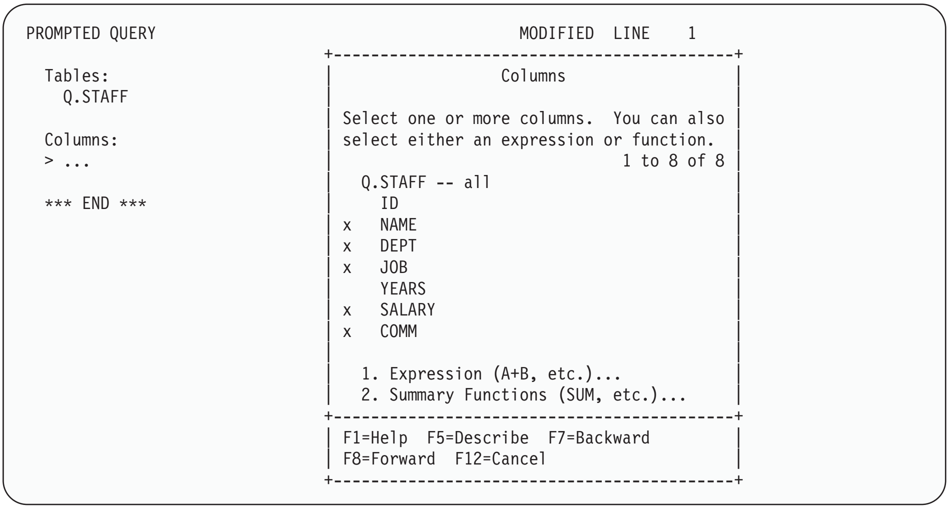
To select columns:

1. Press Enter.

   The Columns panel displays, which shows all the columns in the Q.STAFF table. You might need to scroll forward to see them all.

At the bottom of the panel, you will see selections for **Expression** and **Summary Functions**. You will not use expressions or summary functions in this lesson, but you will learn more about them in Chapter 4, "Viewing the data in the database using prompted query," on page 35.

2. Type x beside **NAME, DEPT, JOB, SALARY,** and **COMM**.



```
PROMPTED QUERY                                         MODIFIED  LINE    1
                                +--------------------------------------------+
  Tables:                       |                 Columns                    |
    Q.STAFF                     |                                            |
                                | Select one or more columns.  You can also  |
  Columns:                      | select either an expression or function.   |
  > ...                         |                              1 to 8 of 8   |
                                |   Q.STAFF -- all                           |
  *** END ***                   |     ID                                     |
                                | x   NAME                                   |
                                | x   DEPT                                   |
                                | x   JOB                                    |
                                |     YEARS                                  |
                                | x   SALARY                                 |
                                | x   COMM                                   |
                                |                                            |
                                |   1. Expression (A+B, etc.)...             |
                                |   2. Summary Functions (SUM, etc.)...      |
                                +--------------------------------------------+
                                | F1=Help  F5=Describe  F7=Backward          |
                                | F8=Forward  F12=Cancel                     |
                                +--------------------------------------------+
```

Figure 9. The Columns panel

3. Press Enter.

The columns you selected are displayed in the echo area, and the Specify panel displays again.

Selecting the rows to display

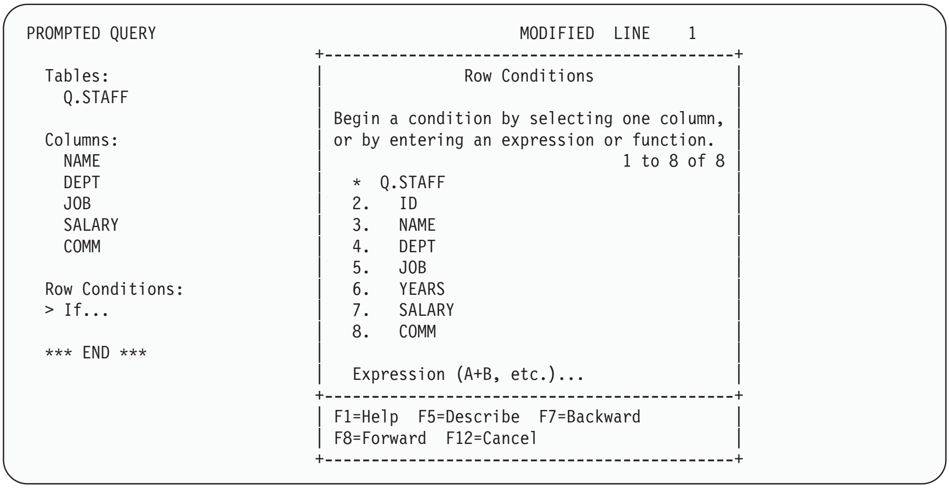
Next, you select the rows to display on your report. For this lesson, select only the rows for employees who are clerks.

In QMF, selecting specific rows to display is called creating a new condition.

To select rows:

1. Press Enter to select choice 3, **Row Conditions**.

   The Row Conditions panel displays, as shown in Figure 10 on page 21.



```
PROMPTED QUERY                                       MODIFIED  LINE    1
                               +---------------------------------------------+
  Tables:                      |               Row Conditions                |
    Q.STAFF                    |                                             |
                               | Begin a condition by selecting one column,  |
  Columns:                     | or by entering an expression or function.   |
    NAME                       |                                 1 to 8 of 8 |
    DEPT                       |   *  Q.STAFF                                |
    JOB                        |   2.   ID                                   |
    SALARY                     |   3.   NAME                                 |
    COMM                       |   4.   DEPT                                 |
                               |   5.   JOB                                  |
  Row Conditions:              |   6.   YEARS                                |
  > If...                      |   7.   SALARY                               |
                               |   8.   COMM                                 |
  *** END ***                  |                                             |
                               |   Expression (A+B, etc.)...                 |
                               +---------------------------------------------+
                               | F1=Help  F5=Describe  F7=Backward           |
                               | F8=Forward  F12=Cancel                      |
                               +---------------------------------------------+
```

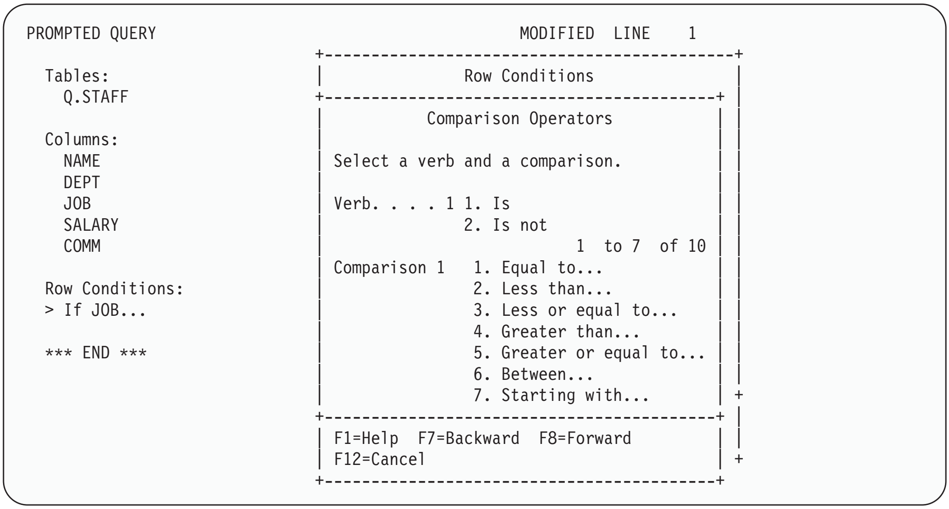
Figure 10. The Row Conditions panel

To create a row condition, select a column on which to base your row condition. You can use any column in the table, even if you are not displaying it on the report.

In this example, you want to display only rows where the job is clerk; select the **JOB** column.

2. Type 5 to select **JOB**.
3. Press Enter.

   The Comparison Operators panel displays:



```
PROMPTED QUERY                                       MODIFIED  LINE    1
                               +---------------------------------------------+
  Tables:                      |               Row Conditions                |
    Q.STAFF                    +-------------------------------------------+ |
                               |           Comparison Operators            | |
  Columns:                     |                                           | |
    NAME                       | Select a verb and a comparison.           | |
    DEPT                       |                                           | |
    JOB                        | Verb. . . . 1 1. Is                       | |
    SALARY                     |               2. Is not                   | |
    COMM                       |                            1  to 7  of 10 | |
                               | Comparison 1   1. Equal to...             | |
  Row Conditions:              |                2. Less than...            | |
  > If JOB...                  |                3. Less or equal to...     | |
                               |                4. Greater than...         | |
  *** END ***                  |                5. Greater or equal to...  | |
                               |                6. Between...              | |
                               |                7. Starting with...        | +
                               +-------------------------------------------+ |
                               | F1=Help  F7=Backward  F8=Forward          | |
                               | F12=Cancel                                | +
                               +-------------------------------------------+
```

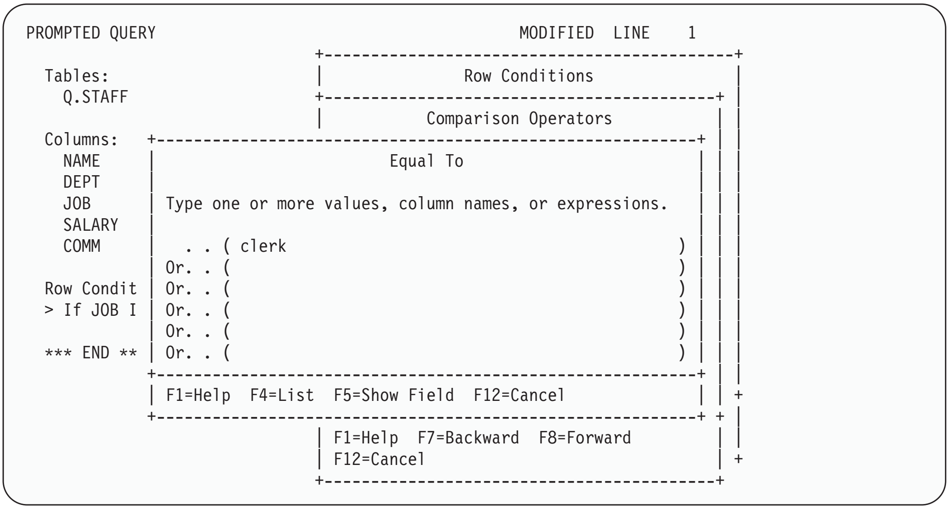
Figure 11. The Comparison Operators panel

On the Comparison Operators panel, create a row condition in which you only see rows where **JOB** is equal to clerk. To do this, select **1**, **Is**, and select **1**, **Equal to**. These choices are already selected for you.

4. Press Enter.

   The Equal To panel displays.

5. Type `clerk` on the first line of the Equal To panel.



```
PROMPTED QUERY                                       MODIFIED  LINE    1
                               +--------------------------------------------+
  Tables:                      |               Row Conditions               |
    Q.STAFF                    +------------------------------------------+ |
                               |           Comparison Operators           | |
  Columns:   +----------------------------------------------------------+ | |
    NAME     |                         Equal To                         | | |
    DEPT     |                                                          | | |
    JOB      | Type one or more values, column names, or expressions.   | | |
    SALARY   |                                                          | | |
    COMM     |   . . ( clerk                                          ) | | |
             | Or. . (                                                ) | | |
  Row Condit | Or. . (                                                ) | | |
  > If JOB I | Or. . (                                                ) | | |
             | Or. . (                                                ) | | |
  *** END ** | Or. . (                                                ) | | |
             +----------------------------------------------------------+ | |
             | F1=Help  F4=List  F5=Show Field  F12=Cancel              | | +
             +----------------------------------------------------------+ + |
                               | F1=Help  F7=Backward  F8=Forward         | |
                               | F12=Cancel                               | +
                               +------------------------------------------+
```

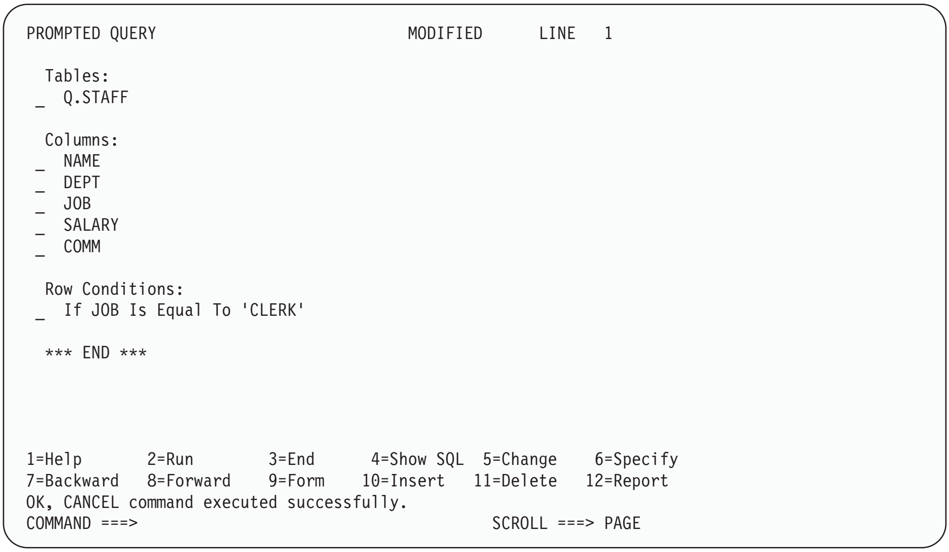
Figure 12. The Equal To panel

6. Press Enter.

   The row condition you created is displayed in the echo area, and the Specify panel displays again.

7. Press the Cancel function key to close the Specify panel.

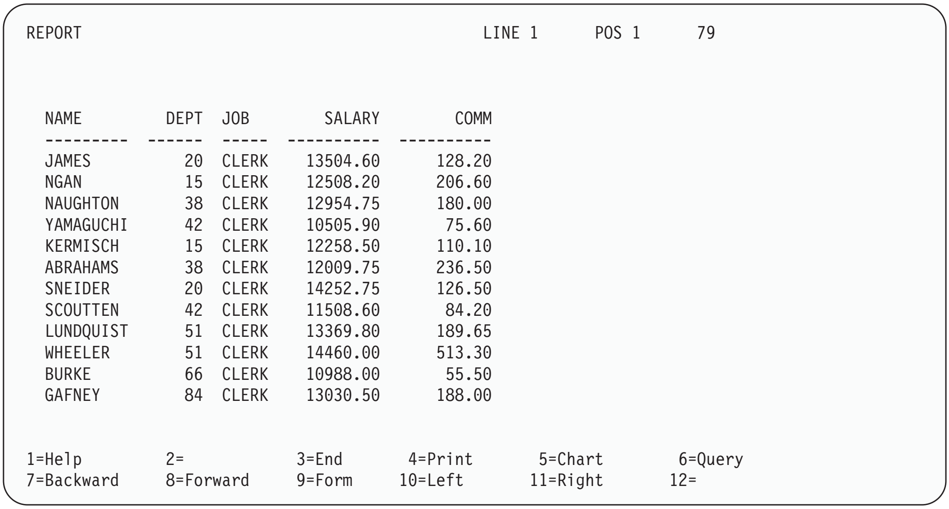
   The Prompted Query panel displays. Your query is displayed in the echo area, as shown in Figure 13 on page 23.



```
PROMPTED QUERY                           MODIFIED      LINE   1

  Tables:
 _  Q.STAFF

  Columns:
 _  NAME
 _  DEPT
 _  JOB
 _  SALARY
 _  COMM

  Row Conditions:
 _  If JOB Is Equal To 'CLERK'

  *** END ***




1=Help       2=Run        3=End      4=Show SQL  5=Change    6=Specify
7=Backward   8=Forward    9=Form    10=Insert   11=Delete   12=Report
OK, CANCEL command executed successfully.
COMMAND ===>                                  SCROLL ===> PAGE
```

Figure 13. QMF displays your query on the Prompted Query panel.

8. To run the query and display your data, press the Run function key. The following report displays:



```
REPORT                                    LINE 1      POS 1      79



  NAME        DEPT  JOB       SALARY        COMM
  ---------  ------  -----  ----------  ----------
  JAMES          20  CLERK    13504.60      128.20
  NGAN           15  CLERK    12508.20      206.60
  NAUGHTON       38  CLERK    12954.75      180.00
  YAMAGUCHI      42  CLERK    10505.90       75.60
  KERMISCH       15  CLERK    12258.50      110.10
  ABRAHAMS       38  CLERK    12009.75      236.50
  SNEIDER        20  CLERK    14252.75      126.50
  SCOUTTEN       42  CLERK    11508.60       84.20
  LUNDQUIST      51  CLERK    13369.80      189.65
  WHEELER        51  CLERK    14460.00      513.30
  BURKE          66  CLERK    10988.00       55.50
  GAFNEY         84  CLERK    13030.50      188.00


1=Help         2=           3=End       4=Print       5=Chart        6=Query
7=Backward     8=Forward     9=Form     10=Left       11=Right       12=
```

Figure 14. QMF displays the data from your query.

Lesson Three: Customizing a report

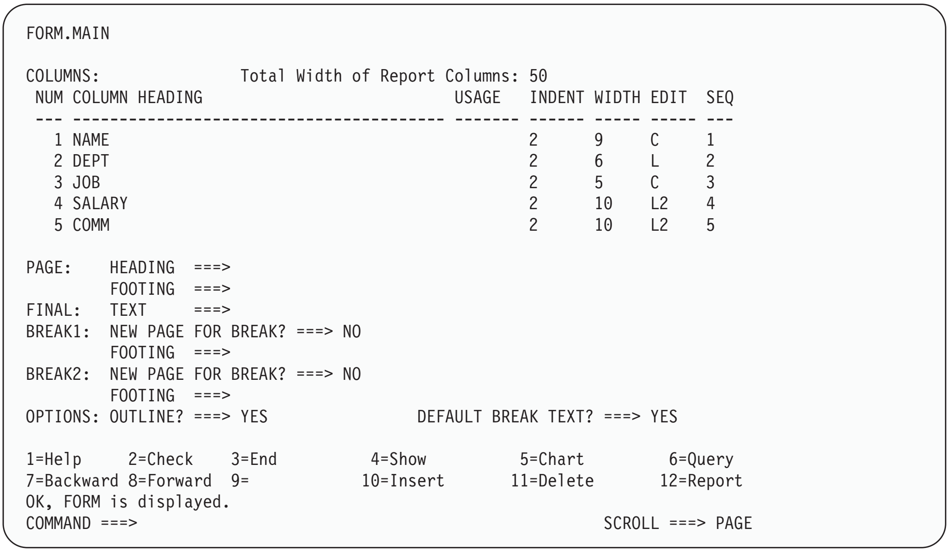
QMF displayed the data you retrieved in the previous lesson as a report. A report is a display of data that is formatted to make it easy to read or view.

When you run a query, QMF uses a default report format to display the report. You can change the appearance of your report by changing the default report format. In this lesson, you will learn how to change column headings and column widths, and add a page heading to the report you just displayed. You can perform many of these tasks from within a Windows environment by using the DB2 QMF for Windows feature.

To change the appearance of a report:

1. From the report panel, press the Form function key to display the default report form.

   The FORM.MAIN panel displays:
   First, change some column names.



```
FORM.MAIN

COLUMNS:              Total Width of Report Columns: 50
 NUM COLUMN HEADING                           USAGE   INDENT WIDTH EDIT  SEQ
 --- ---------------------------------------- ------- ------ ----- ----- ---
   1 NAME                                             2      9     C     1
   2 DEPT                                             2      6     L     2
   3 JOB                                              2      5     C     3
   4 SALARY                                           2      10    L2    4
   5 COMM                                             2      10    L2    5

PAGE:    HEADING  ===>
         FOOTING  ===>
FINAL:   TEXT     ===>
BREAK1:  NEW PAGE FOR BREAK? ===> NO
         FOOTING  ===>
BREAK2:  NEW PAGE FOR BREAK? ===> NO
         FOOTING  ===>
OPTIONS: OUTLINE? ===> YES                DEFAULT BREAK TEXT? ===> YES

1=Help     2=Check    3=End          4=Show         5=Chart         6=Query
7=Backward 8=Forward  9=            10=Insert      11=Delete       12=Report
OK, FORM is displayed.
COMMAND ===>                                                  SCROLL ===> PAGE
```

Figure 15. The FORM.MAIN panel

   When you use the default report form, QMF assigns a name, usually the column name from the table, to each column on the report.

2. Move the cursor to the column name **NAME** and type `EMPLOYEE_NAME`.

   Be sure to type an underscore between each word in the column heading.

3. Type `DEPT_NUMBER` for **DEPT**.
4. Type `COMMISSIONS` for **COMM**.

   Next, you need to change the column width of the COMMISSIONS column so the title will fit.

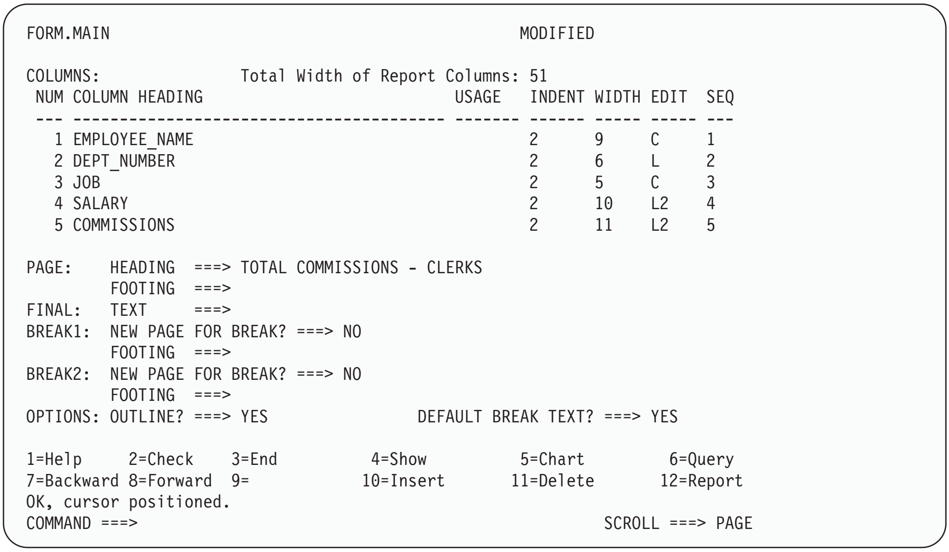
5. Move the cursor to the column width for COMMISSIONS and type 11.

   Finally, you will specify the text you want to display at the top of each page of your report.

6. Move the cursor to **PAGE: HEADING ==>** and type:

```
TOTAL COMMISSIONS - CLERKS
```

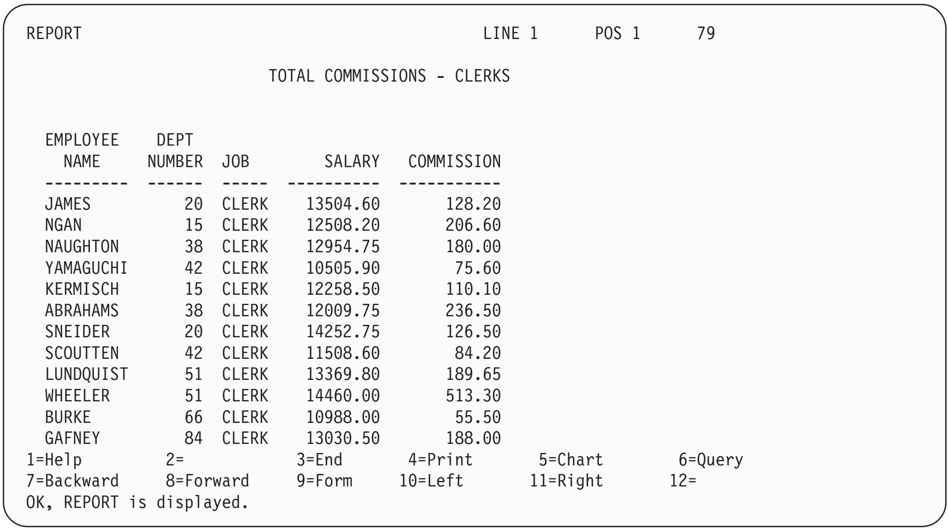
You have finished making your changes to the report format. The FORM.MAIN panel should look like this:



```
FORM.MAIN                                            MODIFIED

COLUMNS:               Total Width of Report Columns: 51
 NUM COLUMN HEADING                           USAGE   INDENT WIDTH EDIT  SEQ
 --- ---------------------------------------- ------- ------ ----- ----- ---
   1 EMPLOYEE_NAME                                    2      9     C     1
   2 DEPT_NUMBER                                      2      6     L     2
   3 JOB                                              2      5     C     3
   4 SALARY                                           2      10    L2    4
   5 COMMISSIONS                                      2      11    L2    5

PAGE:    HEADING  ===> TOTAL COMMISSIONS - CLERKS
         FOOTING  ===>
FINAL:   TEXT     ===>
BREAK1:  NEW PAGE FOR BREAK? ===> NO
         FOOTING  ===>
BREAK2:  NEW PAGE FOR BREAK? ===> NO
         FOOTING  ===>
OPTIONS: OUTLINE? ===> YES                DEFAULT BREAK TEXT? ===> YES

1=Help     2=Check    3=End          4=Show          5=Chart         6=Query
7=Backward 8=Forward  9=            10=Insert       11=Delete       12=Report
OK, cursor positioned.
COMMAND ===>                                                  SCROLL ===> PAGE
```

Figure 16. The FORM.MAIN panel shows the changes you want.

7. Press the Report function key to see the changed report:



```
REPORT                                           LINE 1      POS 1      79

                          TOTAL COMMISSIONS - CLERKS


  EMPLOYEE    DEPT
    NAME     NUMBER  JOB        SALARY   COMMISSION
  ---------  ------  -----  ----------  -----------
  JAMES          20  CLERK    13504.60       128.20
  NGAN           15  CLERK    12508.20       206.60
  NAUGHTON       38  CLERK    12954.75       180.00
  YAMAGUCHI      42  CLERK    10505.90        75.60
  KERMISCH       15  CLERK    12258.50       110.10
  ABRAHAMS       38  CLERK    12009.75       236.50
  SNEIDER        20  CLERK    14252.75       126.50
  SCOUTTEN       42  CLERK    11508.60        84.20
  LUNDQUIST      51  CLERK    13369.80       189.65
  WHEELER        51  CLERK    14460.00       513.30
  BURKE          66  CLERK    10988.00        55.50
  GAFNEY         84  CLERK    13030.50       188.00
1=Help         2=            3=End       4=Print       5=Chart        6=Query
7=Backward     8=Forward     9=Form     10=Left       11=Right       12=
OK, REPORT is displayed.
```

Figure 17. The report reflects the changes you made.

8. Press the End function key to return to the QMF Home panel.

Where do you go from here?

While working through these lessons, you have had a chance to look at many of the basic features of QMF. If you would like more detailed information on any of these features, see the following sections:

- For information on Prompted Query, see Chapter 4, “Viewing the data in the database using prompted query,” on page 35. For information on using SQL, see Chapter 5, “Viewing the data in the database using SQL statements,” on page 69.
- For information on creating and formatting reports, see Chapter 6, “Customizing your reports,” on page 113.
- For information on creating and formatting charts, see Chapter 7, “Displaying your report as a chart,” on page 167.
- For information on working with tables, see Chapter 10, “Creating tables,” on page 211 and Chapter 11, “Maintaining the data in your tables,” on page 217.

Chapter 3. Displaying a list of database objects

To quickly view information about database objects, display a list of those objects.

You can filter the objects in the list in many different ways. For example, you can limit objects to a certain type, such as queries, or to objects with names that contain a certain sequence of characters (for example, all objects with names beginning with ST).

QMF allows you to see only objects that you are authorized to use. These objects can include objects you saved in the database and objects that other users share with you.

If your database supports distributed unit of work, you can also display a list of tables that are located at a remote location. See your QMF administrator to find out if you have distributed unit of work. You can display QMF queries and forms in a Windows environment by using the DB2 QMF for Windows feature. See Appendix C, "The DB2 QMF High Performance Option," on page 367 for more information.

Displaying a list of database objects using the List key

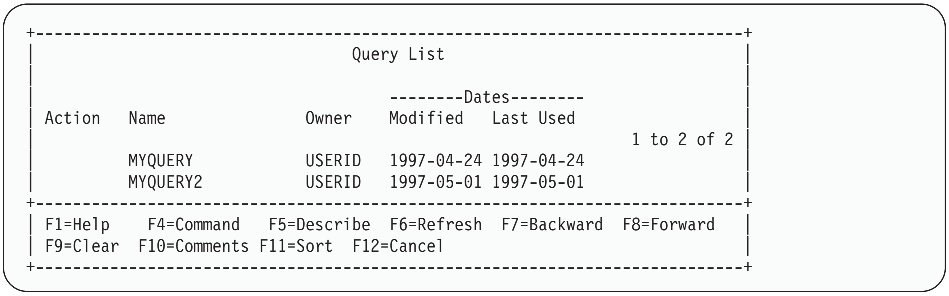
1. Type the object type (TABLES, QUERIES, PROCS, FORMS, QMF, or ALL) on the QMF command line.
2. Press the List function key. The list for the object type you specified displays.

   For example, to display a list of all the queries you own, type:

   ```
   QUERIES
   ```

   on the QMF command line and press the List function key.

   You will see a list that is similar to the one shown in Figure 18 on page 28.



```
+-----------------------------------------------------------------------------+
|                                  Query List                                 |
|                                                                             |
|                                      --------Dates--------                  |
| Action   Name              Owner    Modified   Last Used                    |
|                                                                 1 to 2 of 2 |
|          MYQUERY           USERID   1997-04-24 1997-04-24                   |
|          MYQUERY2          USERID   1997-05-01 1997-05-01                   |
+-----------------------------------------------------------------------------+
| F1=Help    F4=Command   F5=Describe  F6=Refresh  F7=Backward  F8=Forward    |
| F9=Clear  F10=Comments F11=Sort  F12=Cancel                                 |
+-----------------------------------------------------------------------------+
```

Figure 18. Displaying a list of Query objects

The name of the list depends on the type of objects displayed. This list is called a Query List because it contains only queries. A list containing more than one type of object is called an Object List.

You can type QMF commands in the Action area. Press the Comments function key to see a descriptive comment line for each object on the list. Press the Describe function key if the comment for that object is too long to display on the screen, or if you need more detailed information about an object. Press the Cancel function key to remove the list.

You cannot list REPORT or CHART objects, because they are not saved in the database; only the queries, or data and forms to produce them, are saved.

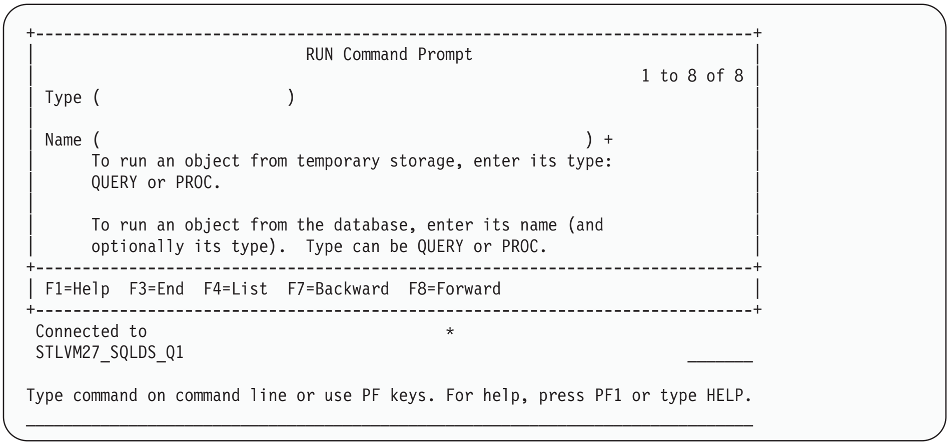
You can display a list of database objects from a command prompt panel for any field with a **+** sign next to it.

To display a list of database objects from a command prompt panel:

1. Move the cursor to the field with the **+** sign after it.

   For example, the **Name** field on the RUN Command Prompt panel has a **+** sign next to it; you can display a list of object names for this command. To see the RUN Command Prompt panel, on the QMF command line, enter:

   ```
   RUN ?
   ```

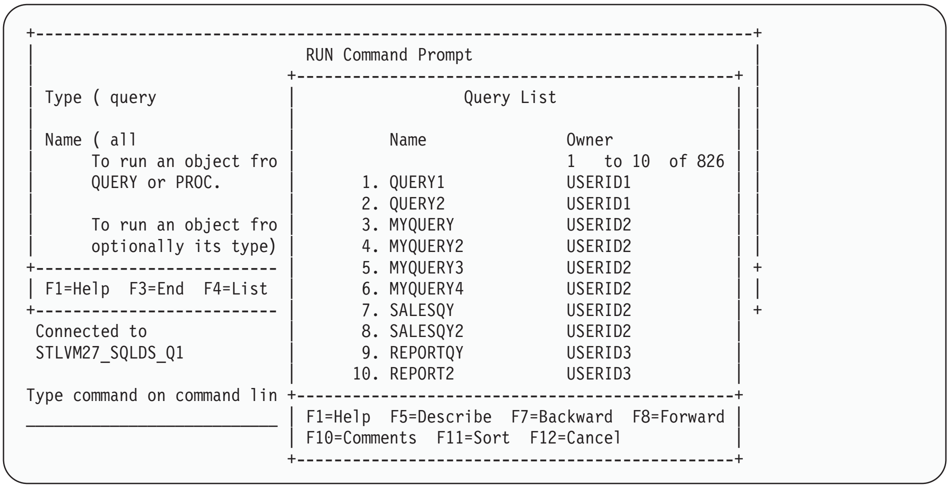


```
+------------------------------------------------------------------------------+
|                              RUN Command Prompt                              |
|                                                                  1 to 8 of 8 |
| Type (                    )                                                  |
|                                                                              |
| Name (                                                   ) +                 |
|      To run an object from temporary storage, enter its type:                |
|      QUERY or PROC.                                                          |
|                                                                              |
|      To run an object from the database, enter its name (and                 |
|      optionally its type).  Type can be QUERY or PROC.                       |
+------------------------------------------------------------------------------+
| F1=Help  F3=End  F4=List  F7=Backward  F8=Forward                            |
+------------------------------------------------------------------------------+
 Connected to                              *
 STLVM27_SQLDS_Q1                                                       _______

Type command on command line or use PF keys. For help, press PF1 or type HELP.
______________________________________________________________________________
```

Figure 19. You can list objects from the RUN Command Prompt panel.

You can also type information in the entry fields to choose the type of objects listed or to limit the objects that are shown by owner. For example, you can type QUERY in the Type field to see only query objects.

2. Press the List function key.
   The list displayed shows only the objects that are valid for the command you are issuing. For example, the valid objects for the RUN command are QUERY and PROC, so only those objects are displayed on the list.



```
+------------------------------------------------------------------------------+
|                              RUN Command Prompt                              |
|                            +------------------------------------------------+ |
| Type ( query               |                   Query List                   | |
|                            |                                                | |
| Name ( all                 |          Name               Owner              | |
|      To run an object fro  |                             1   to 10  of 826  | |
|      QUERY or PROC.        |       1. QUERY1             USERID1            | |
|                            |       2. QUERY2             USERID1            | |
|      To run an object fro  |       3. MYQUERY            USERID2            | |
|      optionally its type)  |       4. MYQUERY2           USERID2            | |
+--------------------------- |       5. MYQUERY3           USERID2            | +
| F1=Help  F3=End  F4=List   |       6. MYQUERY4           USERID2            | |
+--------------------------- |       7. SALESQY            USERID2            | +
 Connected to                |       8. SALESQY2           USERID2            |
 STLVM27_SQLDS_Q1            |       9. REPORTQY           USERID3            |
                             |      10. REPORT2            USERID3            |
Type command on command lin  +------------------------------------------------+
___________________________  | F1=Help  F5=Describe  F7=Backward  F8=Forward  |
                             | F10=Comments  F11=Sort  F12=Cancel             |
                             +------------------------------------------------+
```

Figure 20. QMF displays a list of query objects.

To select an object from the list, enter its number.

The Command Prompt panel displays again. The object that you selected displays in the appropriate field.

Displaying a list of database objects using the LIST command

The LIST command works like the List function key, except that you enter the command instead of pressing a key.

You can specify which objects to display by using keywords with the LIST command.

In DB2 QMF Version 8.1, the LIST command supports names longer than 18 characters. When using the LIST command in a QMF procedure, you must use single quotes to continue an authorization ID across more than one line.

Table 4 shows you how to use the LIST command to display specific objects in your list.

Table 4. Choose specific objects to display with the List command.

Objects you want in the list	**What you enter**	**Comments**
Objects you own	`LIST ALL`	Displays all the objects you own, including TABLES, QUERIES, PROCS, and FORMS
Objects of a specific object type that you own	`LIST` *`objecttype`*	For example, enter: `LIST QUERIES` to display a list of all the queries you have saved in the database
Objects another user owns and shares with you	`LIST` *`objecttype`* `(OWNER=`*`userid`*	For example, enter: `LIST TABLES (OWNER=KRISTI` to display a list of all the tables a person with user ID KRISTI shares with you

Table 4. Choose specific objects to display with the List command. (continued)

Objects you want in the list	What you enter	Comments
Objects at a remote location	`LIST` *`objecttype`* `(LOCATION=`*`location`*	If you are connected to a DB2 database that supports three-part names, you can display a list of tables and views at a remote location. For example, type `LIST TABLES (LOCATION=NEWYORK` to display a list of all tables in a database at the New York location. If your database does not support distributed unit of work, but does support remote unit of work, you can use the CONNECT command to connect to a remote database. Then, you can issue the LIST command to see tables and other QMF objects stored in the remote database. For information on connecting to a database in another location, see Chapter 13, "Accessing data at a remote database," on page 239.

Choosing specific objects using selection symbols

You can select specific objects or values to display on your list by using wildcards or selection symbols.

You can use selection symbols in both the commands you enter on the QMF command line and on command prompt panels. Table 5 shows the two kinds of selection symbols that QMF recognizes.

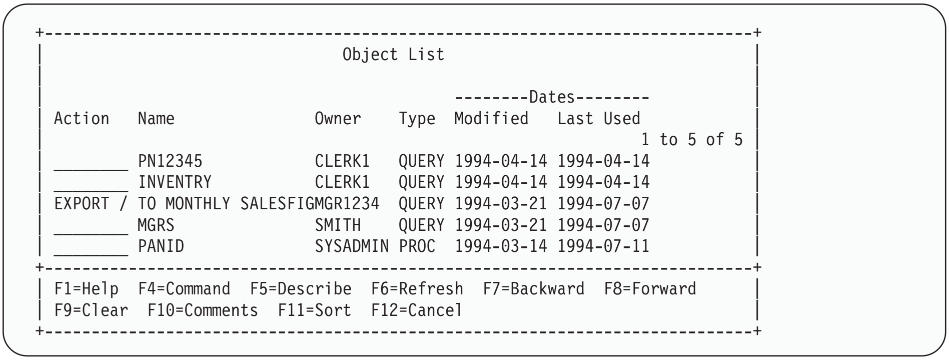
Table 5. QMF's selection symbols

Selection symbol	Fills in for	Examples
Underscore (_)	One character	To display a list of queries whose names start with REPORT but have an unknown character in the seventh position, enter: `LIST QUERIES (NAME=REPORT_`
Percent sign (%)	Zero or more characters	To display queries that are owned by users whose user IDs begin with MC, enter: `LIST QUERIES (OWNER=MC%`

Entering commands on the list of database objects

You can enter QMF commands and parameters in the **Action** area of the List panel for any object in the list, as shown in Figure 21. The command performs an action for that specific object.

Be careful to keep the entries in the list aligned. For example, if you press the Backspace key to correct a typing error, the entire line may shift to the left. Enter another space to keep the entries aligned in columns, or you will get an error message when you issue the command. If the command you are entering is longer than the Action area, you can continue typing the command across the entire width of the list panel.



```
+------------------------------------------------------------------------------+
|                                Object List                                   |
|                                                                              |
|                                            --------Dates--------             |
| Action   Name              Owner    Type  Modified   Last Used              |
|                                                                 1 to 5 of 5 |
| ________ PN12345           CLERK1   QUERY 1994-04-14 1994-04-14              |
| ________ INVENTRY          CLERK1   QUERY 1994-04-14 1994-04-14              |
| EXPORT / TO MONTHLY SALESFIGMGR1234  QUERY 1994-03-21 1994-07-07              |
| ________ MGRS              SMITH    QUERY 1994-03-21 1994-07-07              |
| ________ PANID             SYSADMIN PROC  1994-03-14 1994-07-11              |
+------------------------------------------------------------------------------+
| F1=Help  F4=Command  F5=Describe  F6=Refresh  F7=Backward  F8=Forward       |
| F9=Clear  F10=Comments  F11=Sort  F12=Cancel                                 |
+------------------------------------------------------------------------------+
```

Figure 21. You can enter commands directly on the Object list.

You can enter more than one command on the list. QMF runs commands by starting at the top of the list and moving to the bottom. Type an equal sign (=) to repeat a command for more than one object. Press the Clear function key to erase all the commands you typed on the list.

Commands you can use on the list of database objects

You can issue the following commands in the Action area of a list. Type `QMF` before any command to make sure that the QMF command runs instead of a command synonym. Your installation might have defined a command with the same name as a QMF command.

Command
: **What it does**

CONVERT
: Converts a prompted, QBE, or SQL query into an equivalent SQL query- the comments in the original query do not appear in the converted query

DISPLAY
: Retrieve an object from the database and display it on your terminal

EDIT Edit a table in the database by using the Table Editor

You can only use the EDIT command to edit a table from the database object list. If you want to edit a query or procedure, you must display it first.

ERASE
Delete an object from the database

EXIT End your QMF session

EXPORT
Export QMF objects that are stored in the database directly from the database into a file (CMS), data set (TSO and CICS- z/OS), or queue name (CICS)

IMPORT
Import QMF objects directly into the database from a file (CMS), data set (TSO and CICS- OS/390), or data queue name (CICS)

LAYOUT
Display the format of a report that is produced from a given form without using any data

You can use LAYOUT only with form objects, and only in an environment in which both REXX and ISPF are available.

PRINT
Print a database object

RUN Run a query or procedure that is stored in the database

SAVE Replace the object in the database with the object currently in temporary storage- for example, if you enter:

```
SAVE QUERY AS
```

next to a query on the database object list, QMF replaces that query in the database with the query currently in temporary storage

Using a placeholder on the list of database objects

You can use a slash (/) as a placeholder to represent the object type, owner, and name in a QMF command.

For example, entering the following command in the Action area for a table object:

```
EDIT / (MODE=ADD
```

means the same as entering:

```
EDIT TABLE owner.tablename (MODE=ADD
```

where *owner.tablename* is the owner and name of the table listed.

You can also use /T if you just want to specify the object type, or /N if you just want to specify the owner and name. The /T and /N placeholders are especially useful if you are issuing a command to run a user-written application that requires just the object type or just the object owner and name.

If you are displaying a list from a remote location, the placeholder symbols (/ and /N) include the location with the owner and name.

You can also display the prompt panel for a command with the object type and the object owner and name filled in. To do this, type the command followed by the / placeholder and a question mark.

For example, to display the RUN Command Prompt panel for the DEPTQUERY object, enter `RUN / ?` in the Action area next to the object. The RUN QUERY Command Prompt panels display. The first panel already has the object name and owner filled in. The next panel requests a form name to use in formatting the data from the query.

Correcting errors when you enter an incorrect command

QMF runs the commands you issue on the list of database objects from top to bottom. If QMF finds an incorrect command, it stops, displays an error message, and highlights the line that contains the error.

To correct an error:

1. Read the error message to see why the error occurred. If you need additional help, press the Help function key to get an explanation of the error message.
2. Press the spacebar to enter blanks over the incorrect command. If you want to delete all the commands on the list, press the Clear function key.
3. Type the correct command in the Action area, and press Enter to begin issuing commands again.

Commands that run successfully have an asterisk (*), followed by up to seven letters of the command displayed in the Action area. If the objects on the list change as a result of running the commands, press the Refresh function key to display the changed list.

Returning to the list from another QMF panel

When you issue commands from the list, another panel displays. For example, if you issue the RUN command for a query, QMF displays a report panel. If you issue the EDIT command for a table, QMF displays a Table Editor panel.

To return to the database object list from any panel, press the End function key on that panel.

Chapter 4. Viewing the data in the database using prompted query

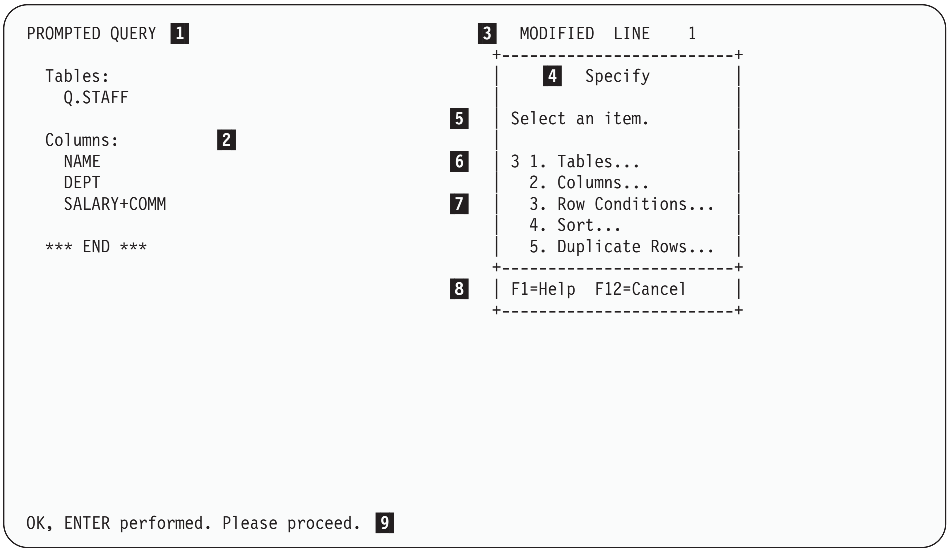
In this chapter, you will learn how to select and view data in the database by using QMF's Prompted Query. If you are new to QMF, or if you only use QMF occasionally, Prompted Query is an easy way to retrieve your data.

Prompted Query prompts you step-by-step through building a query. You do not need to know a query language, only which tables contain the data you want. You can also build QMF queries in a Windows environment by using the DB2 QMF for Windows feature. See Appendix C, "The DB2 QMF High Performance Option," on page 367 for more information.

If you completed the lessons in Chapter 2, "QMF in three quick lessons," on page 17, you are already familiar with Prompted Query.

Prompted query panels

Figure 22 shows the Prompted Query panel while a query is being built. Each area marked by a number in the figure is described in the list that follows.



```
PROMPTED QUERY                                  MODIFIED  LINE    1
                                             +-------------------------+
  Tables:                                    |       Specify           |
    Q.STAFF                                  |                         |
                                             | Select an item.         |
  Columns:                                   |                         |
    NAME                                     | 3 1. Tables...          |
    DEPT                                     |   2. Columns...         |
    SALARY+COMM                              |   3. Row Conditions...  |
                                             |   4. Sort...            |
  *** END ***                                |   5. Duplicate Rows...  |
                                             +-------------------------+
                                             | F1=Help  F12=Cancel     |
                                             +-------------------------+









OK, ENTER performed. Please proceed.
```

Figure 22. Building a prompted query

1 The name of the main panel. When you are using Prompted Query, this always says PROMPTED QUERY.

2 The echo area. All of the information that you specify in dialog panels is shown in there.

3 This area shows whether the screen you are looking at was changed in some way. If you have not made any changes, the word `MODIFIED` does not appear. The panel shows a line number that indicates the first line of the query that appears on the panel. When you scroll forward, the line number changes.

4 The Specify panel is a dialog panel. Use dialog panels to specify different kinds of information.

5 This field shows instructions that tell you what kind of information you can specify on this dialog panel, and how you can specify it.

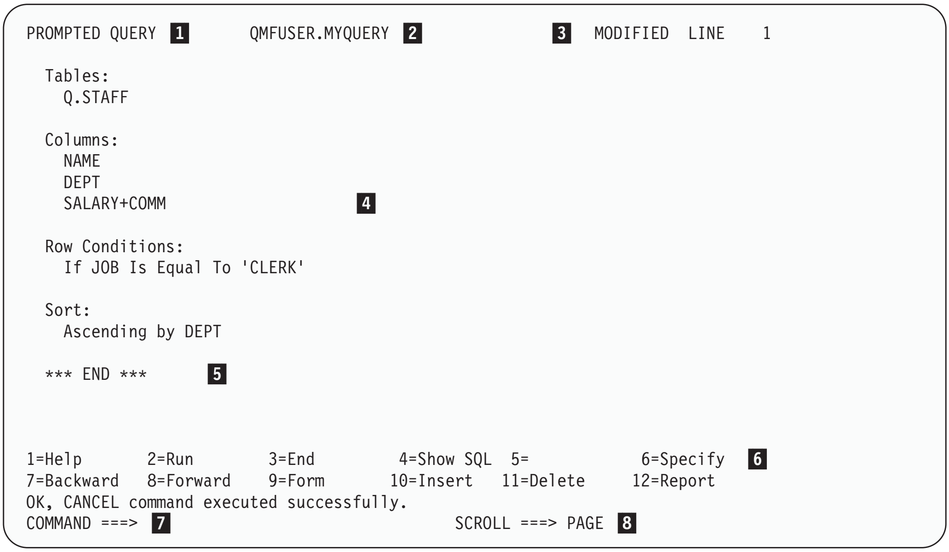
6 This is the area where you specify your choice. Different types of dialog panels have different ways to specify items. This dialog panel shows that your next item is `3. Row Conditions`.

7 This field shows the list of items you can choose from. You sometimes have to scroll forward to see all of the items on a list.

8 This field shows the active function keys. Use the function keys to issue certain commands, such as scrolling or getting help.

9 This is the message line. Whenever you issue a command in QMF, the message line indicates whether your command was successful.

Figure 23 on page 37 shows the main Prompted Query panel after the query is built.



```
PROMPTED QUERY 1      QMFUSER.MYQUERY 2             3 MODIFIED  LINE    1

  Tables:
    Q.STAFF

  Columns:
    NAME
    DEPT
    SALARY+COMM                     4

  Row Conditions:
    If JOB Is Equal To 'CLERK'

  Sort:
    Ascending by DEPT

  *** END ***       5



1=Help       2=Run        3=End         4=Show SQL  5=           6=Specify  6
7=Backward   8=Forward    9=Form       10=Insert   11=Delete     12=Report
OK, CANCEL command executed successfully.
COMMAND ===> 7                              SCROLL ===> PAGE 8
```

Figure 23. The built query

1 The name of the panel

2 The name of a saved query

If you saved a prompted query in the database, it has a two-part name: the user identification of the person who saved the query, and the name you gave the query when you saved it, separated by a period.

3 This shows that the panel was changed

4 This is the echo area- it currently shows the completed query that was built using the dialog panels

5 The word END means that there is no more information to look at- if END does not appear, you can scroll forward to look at the rest of your query

6 This field shows the active function keys

QMF assigns commonly-used commands to function keys. Instead of typing a command, you can just press the numbered function key with that label. Your function keys might have different commands assigned to them than the ones that are shown in the figure. The functions of the keys depend on the assignments your QMF administrator makes.

7 This is the command line- whenever you see `COMMAND ===>`, you can enter a QMF command

8 This is the scroll indicator- it shows how many lines QMF scrolls when you enter a forward or backward command

Rules for using Prompted Query: Here are some general rules for using Prompted Query:

- Your QMF profile must have the language operand set to `PROMPTED`; otherwise, each time you start a query, you must enter `RESET QUERY (LANGUAGE=PROMPTED`.
- You must type all entered data in uppercase, unless you set the CASE operand in your QMF profile to `UPPER`.
- When entering data, if a character string contains a special character, you must enclose the character string in quotation marks. For a list of special characters, see *DB2 QMF Reference*.
- You must put quotes around all graphic data that you enter.
- The name you give your query cannot be longer than 18 characters.

Starting prompted query

1. Make sure that the value of the Language field in your QMF user profile is PROMPTED. If you need more information about setting up your QMF user profile, see "Setting up and changing your QMF user profile" on page 7.
2. On the QMF command line, enter:

   ```
   RESET QUERY
   ```

   If you do not want to change your QMF profile, enter:

   ```
   RESET QUERY (LANGUAGE=PROMPTED
   ```

   each time you use the RESET command.

   The Prompted Query panel with the Tables panel displays.

Selecting tables and columns

To select tables and columns:

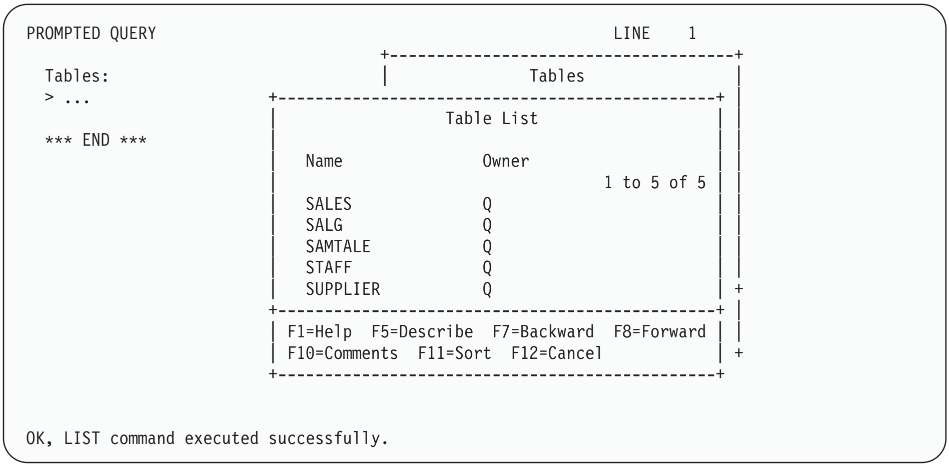
1. Enter the name of your table on the Tables panel. You can specify up to 15 tables.

   The name of the table shows in the echo area, and the Specify panel displays.

   To find the name of a table, you can list available tables:

   - On the Tables panel, press the List function key.

     You can specify search criteria to filter the list so it is smaller. For example, you can type `Q.S%` on the first line of the Tables dialog panel, and then press the List function key. QMF lists all the tables with names that start with Q.S. % represents a string of any length, containing any characters. The Table List panel displays.



```
PROMPTED QUERY                                                  LINE    1
                                      +-------------------------------------+
  Tables:                             |               Tables                |
  > ...                  +------------------------------------------------+ |
                         |                  Table List                    | |
  *** END ***            |                                                | |
                         |   Name              Owner                      | |
                         |                                    1 to 5 of 5 | |
                         |   SALES             Q                          | |
                         |   SALG              Q                          | |
                         |   SAMTALE           Q                          | |
                         |   STAFF             Q                          | |
                         |   SUPPLIER          Q                          | +
                         +------------------------------------------------+ |
                         | F1=Help  F5=Describe  F7=Backward  F8=Forward  | |
                         | F10=Comments  F11=Sort  F12=Cancel             | +
                         +------------------------------------------------+


OK, LIST command executed successfully.
```

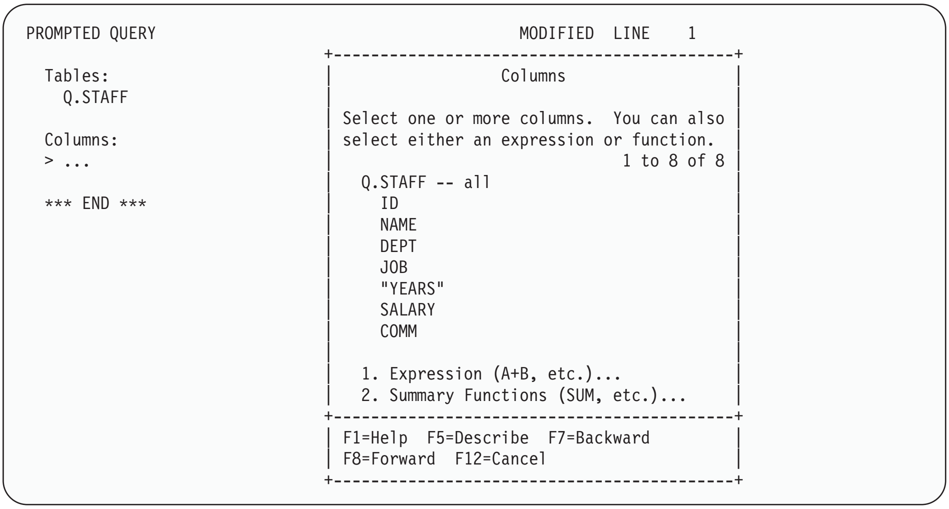
Figure 24. The Table List panel

You can press the Comments function key to see a descriptive comment for each object in the list. Press the Describe function key to see more detailed information about an object. Press the Sort function key to sort by name, type, or date. Press the Cancel function key to remove the list.

Choice 2, **Columns**, is already selected for you on the Specify panel if you have selected only one table. That is because selecting columns is usually the next step in creating a query. If you remove the **2**, Prompted Query selects all columns. If you look in the echo area, you can see that the word ALL appears directly under the Columns heading.

2. Press Enter.

   The Columns panel displays.



```
PROMPTED QUERY                                       MODIFIED  LINE    1
                                +--------------------------------------------+
  Tables:                       |                  Columns                   |
    Q.STAFF                     |                                            |
                                | Select one or more columns.  You can also  |
  Columns:                      | select either an expression or function.   |
  > ...                         |                                1 to 8 of 8 |
                                |   Q.STAFF -- all                           |
  *** END ***                   |     ID                                     |
                                |     NAME                                   |
                                |     DEPT                                   |
                                |     JOB                                    |
                                |     "YEARS"                                |
                                |     SALARY                                 |
                                |     COMM                                   |
                                |                                            |
                                |   1. Expression (A+B, etc.)...             |
                                |   2. Summary Functions (SUM, etc.)...      |
                                +--------------------------------------------+
                                | F1=Help  F5=Describe  F7=Backward          |
                                | F8=Forward  F12=Cancel                     |
                                +--------------------------------------------+
```

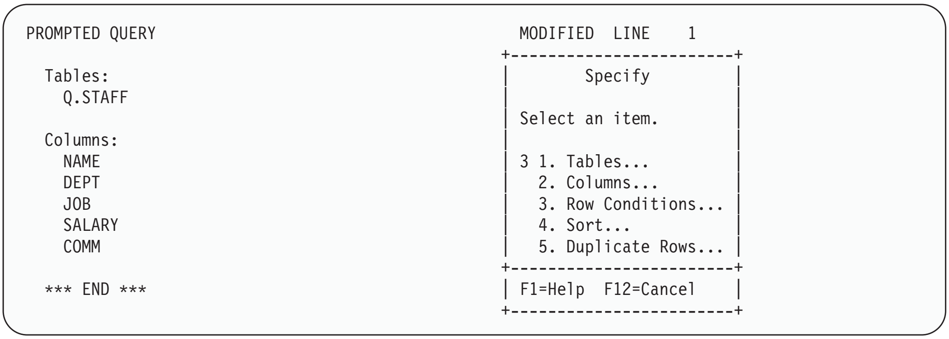
Figure 25. The Columns panel

3. Choose the columns you want displayed in your report by typing any character in the space before the column name.

   Use the Tab key to move past columns you do not want. Press the Forward or Backward function keys to display additional columns.

   If you want to see information about the columns before you select them, tab to the column name and press the Describe function key. The Column Description panel displays.

4. Press Enter. The columns you select are shown in the echo area, and the Specify dialog panel displays, with choice 3, **Row Conditions**.



```
PROMPTED QUERY                                       MODIFIED  LINE    1
                                                   +-------------------------+
  Tables:                                          |        Specify          |
    Q.STAFF                                        |                         |
                                                   | Select an item.         |
  Columns:                                         |                         |
    NAME                                           | 3 1. Tables...          |
    DEPT                                           |   2. Columns...         |
    JOB                                            |   3. Row Conditions...  |
    SALARY                                         |   4. Sort...            |
    COMM                                           |   5. Duplicate Rows...  |
                                                   +-------------------------+
  *** END ***                                      | F1=Help  F12=Cancel     |
                                                   +-------------------------+
```

Figure 26. QMF lists the columns you selected.

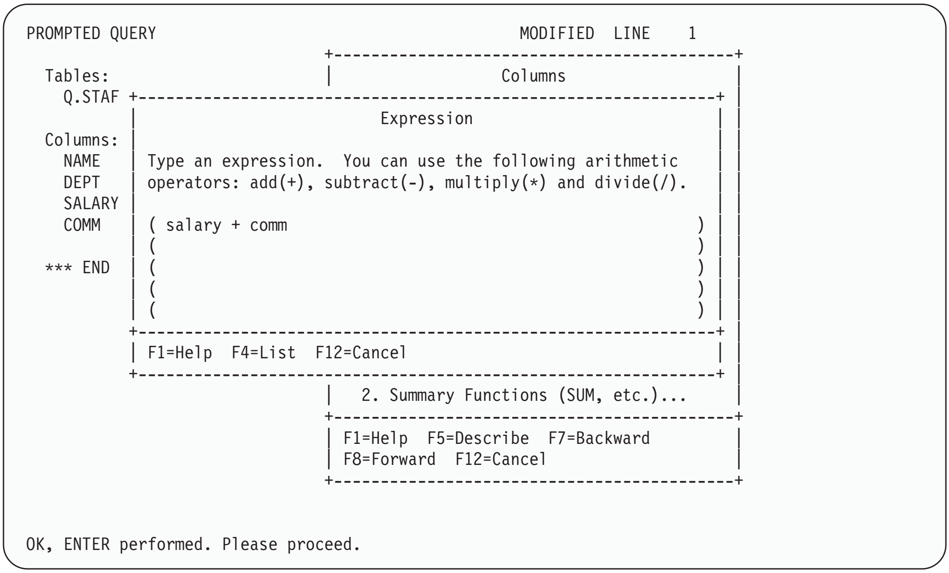
Creating a column using expressions

You can create a column in your report by defining an expression. An expression adds, subtracts, multiplies, or divides the values in two or more columns and places the results in a new column.

For example, you can create a total earnings column that is the sum of each employee's salary and commission.

To create a column:

1. Select the table, either by typing the name on the Tables panel or by selecting the table from the Table List.
2. Select the columns from the Columns panel by typing any character beside each one.
3. At the bottom of the Columns panel, select choice 1, **Expression (A+B, etc.)**.
4. Press Enter. The Expression panel displays.



```
PROMPTED QUERY                                     MODIFIED  LINE    1
                                +--------------------------------------------+
  Tables:                       |                  Columns                   |
    Q.STAF +--------------------------------------------------------------+ |
           |                        Expression                            | |
  Columns: |                                                              | |
    NAME   | Type an expression.  You can use the following arithmetic    | |
    DEPT   | operators: add(+), subtract(-), multiply(*) and divide(/).   | |
    SALARY |                                                              | |
    COMM   | ( salary + comm                                          )   | |
           | (                                                        )   | |
  *** END  | (                                                        )   | |
           | (                                                        )   | |
           | (                                                        )   | |
           +--------------------------------------------------------------+ |
           | F1=Help  F4=List  F12=Cancel                                 | |
           +--------------------------------------------------------------+ |
                                |   2. Summary Functions (SUM, etc.)...      |
                                +--------------------------------------------+
                                | F1=Help  F5=Describe  F7=Backward          |
                                | F8=Forward  F12=Cancel                     |
                                +--------------------------------------------+


OK, ENTER performed. Please proceed.
```

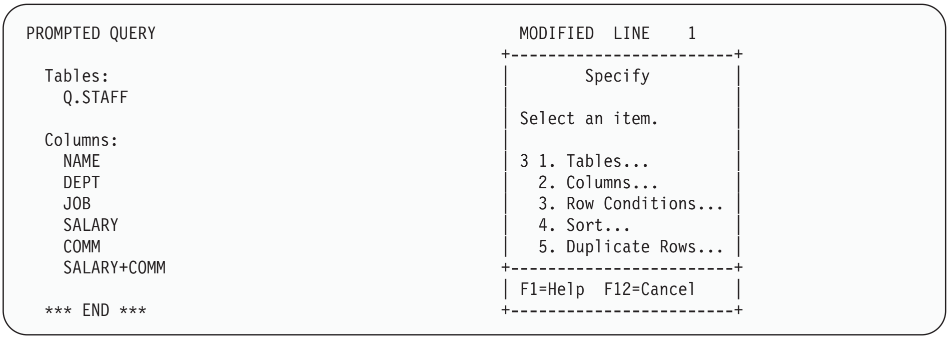
Figure 27. Enter the expression you want to use.

5. Enter your expression. For example, enter:

   ```
   SALARY + COMM
   ```

   You can create more complex expressions. For more information, press the Help function key on the Expressions panel. You can also press the List function key to see columns that you can use in your expression.

   The names of the tables and columns you selected are displayed in the echo area. The new column you created is also displayed under the heading **Columns**. The Specify panel displays again with choice 3, **Row Conditions**.



```
PROMPTED QUERY                                      MODIFIED  LINE    1
                                                  +------------------------+
  Tables:                                         |        Specify         |
    Q.STAFF                                       |                        |
                                                  | Select an item.        |
  Columns:                                        |                        |
    NAME                                          | 3 1. Tables...         |
    DEPT                                          |   2. Columns...        |
    JOB                                           |   3. Row Conditions... |
    SALARY                                        |   4. Sort...           |
    COMM                                          |   5. Duplicate Rows... |
    SALARY+COMM                                   +------------------------+
                                                  | F1=Help  F12=Cancel    |
  *** END ***                                     +------------------------+
```

Figure 28. The new column SALARY+COMM appears in the echo area.

You can also create columns that summarize a group of rows. With QMF's summary functions, you can calculate:

- Sum
- Average
- Minimum
- Maximum
- Count rows

To specify a summary function:

1. On the Columns panel, select choice 2, **Summary Functions (SUM, etc.)**.
2. Press Enter. The Summary Functions panel displays.
3. Select the functions you want to perform by typing a character in the space before the items.

   The Summary Function Items panel displays for any item you select except COUNT. The COUNT function does not need a column on which to act. In the echo area, COUNT is automatically assigned an asterisk, which indicates that it returns a count of the number of rows for each group.
4. Select the column on which you want the summary function to act.
5. Press Enter.

   QMF displays an error message if your expression is over 255 characters after the database processes it. If you see this message, return to the panel and specify your summary function so that the expression is less than or equal to 255 characters.

   The columns you selected are shown in the echo area, and the Specify panel displays again.

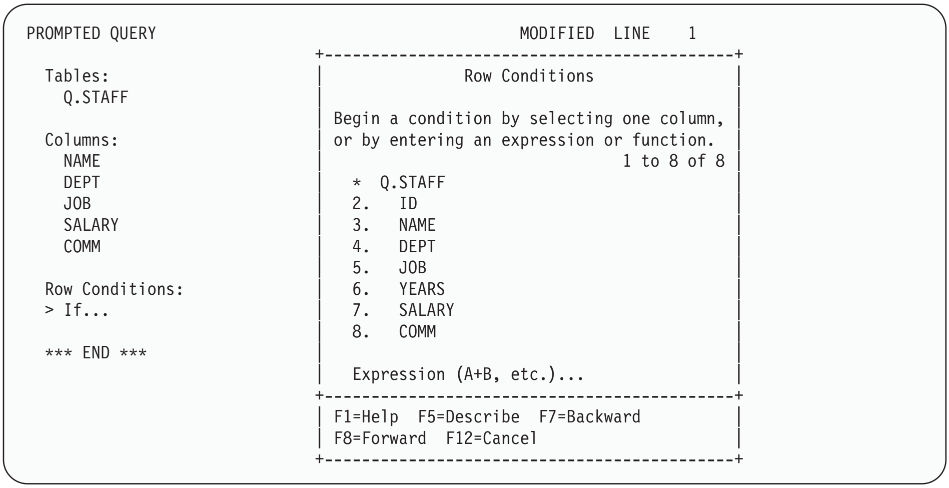
Selecting rows

You can select specific rows to display on your report. Selecting rows limits, or creates a subset of, the data in a table. You select rows by creating a row condition.

To select rows:

1. On the Specify panel, select choice 3, **Row Conditions**.

   The Row Conditions panel displays.

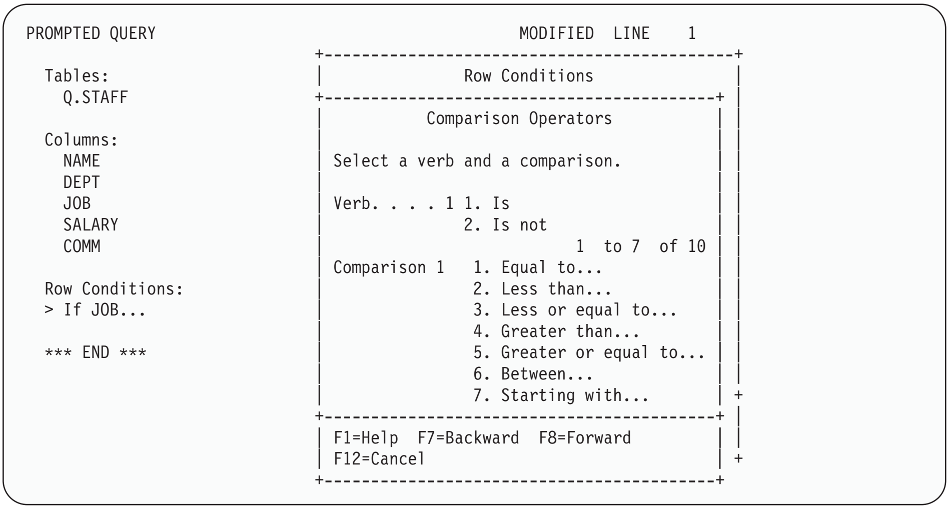


```
PROMPTED QUERY                                      MODIFIED  LINE    1
                               +---------------------------------------------+
  Tables:                      |              Row Conditions                 |
    Q.STAFF                    |                                             |
                               | Begin a condition by selecting one column,  |
  Columns:                     | or by entering an expression or function.   |
    NAME                       |                                 1 to 8 of 8 |
    DEPT                       |   *  Q.STAFF                                |
    JOB                        |   2.   ID                                   |
    SALARY                     |   3.   NAME                                 |
    COMM                       |   4.   DEPT                                 |
                               |   5.   JOB                                  |
  Row Conditions:              |   6.   YEARS                                |
  > If...                      |   7.   SALARY                               |
                               |   8.   COMM                                 |
  *** END ***                  |                                             |
                               |   Expression (A+B, etc.)...                 |
                               +---------------------------------------------+
                               | F1=Help  F5=Describe  F7=Backward           |
                               | F8=Forward  F12=Cancel                      |
                               +---------------------------------------------+
```

Figure 29. The Row Conditions panel

   To create a row condition, select a column that contains data from which you want to choose a limited selection. For example, you might want to see only the employees who are clerks. To do this, you select the rows where the value in the **JOB** column is `CLERK`. You can use any column in the table, even if you do not display it in the final report.

2. Enter the number of the column you want. The Comparison Operators panel displays.



```
PROMPTED QUERY                                       MODIFIED  LINE    1
                               +---------------------------------------------+
  Tables:                      |               Row Conditions                |
    Q.STAFF                    +-------------------------------------------+ |
                               |           Comparison Operators            | |
  Columns:                     |                                           | |
    NAME                       | Select a verb and a comparison.           | |
    DEPT                       |                                           | |
    JOB                        | Verb. . . . 1 1. Is                       | |
    SALARY                     |               2. Is not                   | |
    COMM                       |                           1  to 7  of 10  | |
                               | Comparison 1   1. Equal to...             | |
  Row Conditions:              |                2. Less than...            | |
  > If JOB...                  |                3. Less or equal to...     | |
                               |                4. Greater than...         | |
  *** END ***                  |                5. Greater or equal to...  | |
                               |                6. Between...              | |
                               |                7. Starting with...        | +
                               +-------------------------------------------+ |
                               | F1=Help  F7=Backward  F8=Forward          | |
                               | F12=Cancel                                | +
                               +-------------------------------------------+
```

Figure 30. The Comparison Operators panel

On the Comparison Operators panel, you complete the row condition by selecting the specific values you want to see from the column you previously selected. In this example, you want to select only rows where **JOB** is equal to clerk.

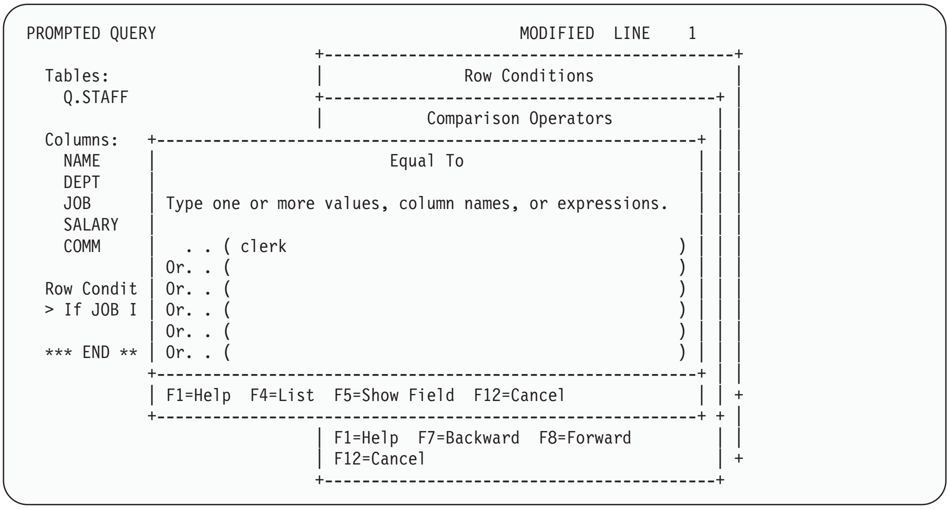
Select the verb and comparison operator that define the relationship between the column and the values you want to see on your report. For this example, you need the verb, **Is**, and comparison operator, **Equal to**, which are already selected.

3. Enter the number of the verb or comparison operator you want. You can scroll forward through the list of comparisons to find the one you want.
4. If the comparison operator you choose requires more information, a panel displays for you to enter a value with which to select the rows. In this example, the Equal To panel displays.

   You can specify more than one value on this panel. Enter the values on separate lines. If a character string contains a special character, such as a hyphen (**-**), enclose the character string in quotation marks. For example, if you are looking for an employee with a hyphenated name, such as Smith-Wiggins, type the name on the Equal To panel with single quotation marks around it:

   ```
   'Smith-Wiggins'
   ```

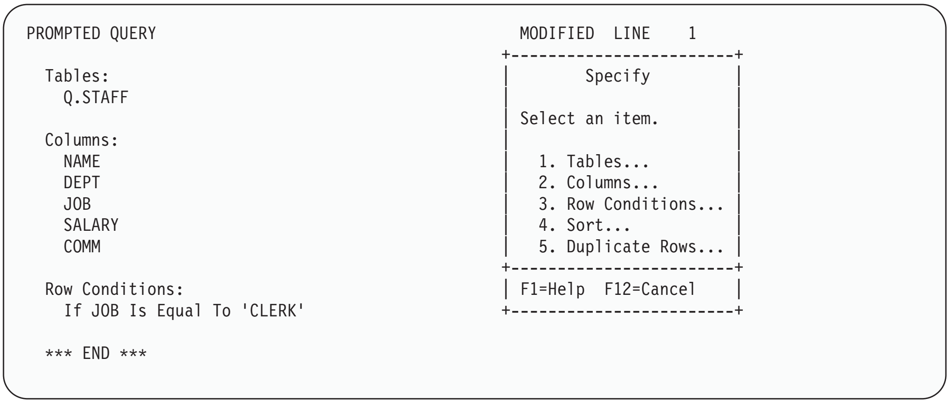
   Entering it this way ensures that the database will not interpret the hyphen as a minus sign, subtracting Wiggins from Smith. For a list of special characters, see *DB2 QMF Reference*.



```
PROMPTED QUERY                                       MODIFIED  LINE    1
                               +---------------------------------------------+
 Tables:                       |               Row Conditions                |
   Q.STAFF                     +-------------------------------------------+ |
                               |          Comparison Operators             | |
 Columns:   +---------------------------------------------------------+ | |
   NAME     |                        Equal To                         | | |
   DEPT     |                                                         | | |
   JOB      | Type one or more values, column names, or expressions.  | | |
   SALARY   |                                                         | | |
   COMM     |   . . ( clerk                                         ) | | |
            | Or. . (                                               ) | | |
 Row Condit | Or. . (                                               ) | | |
 > If JOB I | Or. . (                                               ) | | |
            | Or. . (                                               ) | | |
 *** END ** | Or. . (                                               ) | | |
            +---------------------------------------------------------+ | |
            | F1=Help  F4=List  F5=Show Field  F12=Cancel             | | +
            +---------------------------------------------------------+ + |
                               | F1=Help  F7=Backward  F8=Forward          | |
                               | F12=Cancel                                | +
                               +-------------------------------------------+
```

Figure 31. Enter the value to use in selecting rows.

The row condition you created is displayed in the echo area, and the Specify panel displays. Notice that in the echo area, Prompted Query has put single quotations around **CLERK** because it consists of character data.



```
PROMPTED QUERY                                       MODIFIED  LINE    1
                                                   +-------------------------+
 Tables:                                           |        Specify          |
   Q.STAFF                                         |                         |
                                                   | Select an item.         |
 Columns:                                          |                         |
   NAME                                            |   1. Tables...          |
   DEPT                                            |   2. Columns...         |
   JOB                                             |   3. Row Conditions...  |
   SALARY                                          |   4. Sort...            |
   COMM                                            |   5. Duplicate Rows...  |
                                                   +-------------------------+
 Row Conditions:                                   | F1=Help  F12=Cancel     |
   If JOB Is Equal To 'CLERK'                      +-------------------------+

 *** END ***
```

Figure 32. QMF displays the row condition you created.

Narrowing row selection using multiple row conditions

You can create more than one row condition to further filter the data you want to display on the report.

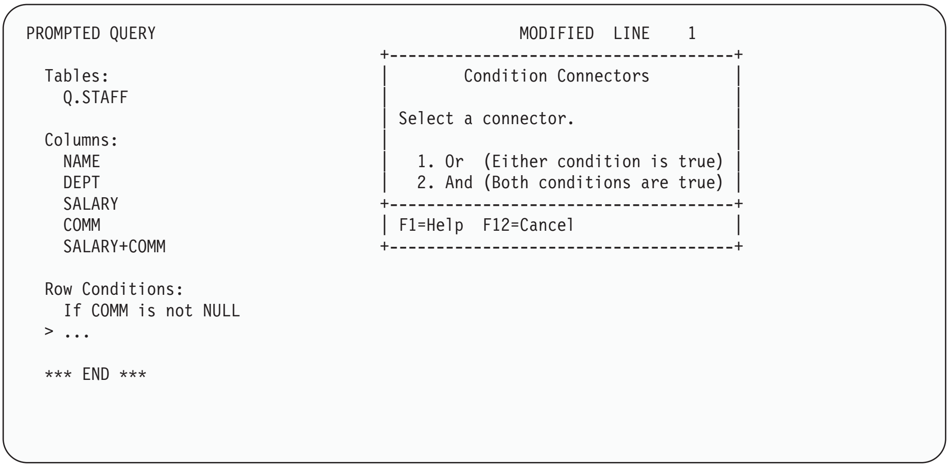
The following example shows how to select only rows for employees who earn a commission and whose total earnings are greater than $17,000.00. To select those rows, you need to create two row conditions: one to select only employees who earn a commission and another to select employees whose total earnings are greater than $17,000.00. Only the rows that meet both conditions appear on the report.

To create multiple row conditions:

1. Select the table and the columns for your report.
2. Press Enter to select choice 3, **Row Conditions**, on the Specify panel.
3. On the Row Conditions panel, enter the column on which you want to base your first row condition. For this example, select the **COMM** column.
4. On the Comparison Operators panel, enter the verb and the comparison operator for the row condition. For this example, select the verb **Is not** and the comparison operator **NULL**. This combination selects employees who earn any commission.

   You have created the first row condition. The row condition is displayed in the echo area, and the Specify panel displays with no choice selected.

   Now create the second row condition.
5. On the Specify panel, select choice 3, **Row Conditions**. The Condition Connectors panel displays.
6. Enter 1 to select rows that meet either of the conditions, or enter 2 to select rows that meet both of the conditions. For this example, enter 2 to select rows that meet both conditions.
   The Row Conditions panel displays.



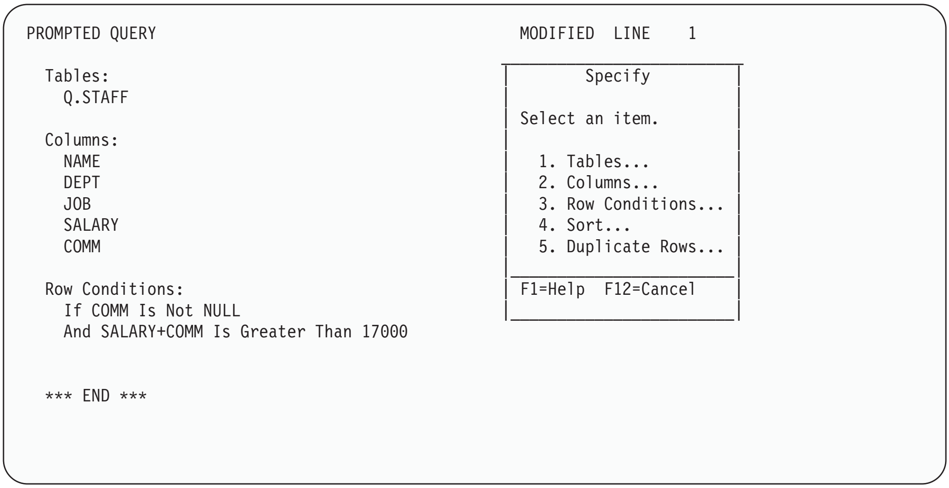
```
PROMPTED QUERY                                      MODIFIED  LINE    1
                                      +-------------------------------------+
  Tables:                             |        Condition Connectors         |
    Q.STAFF                           |                                     |
                                      | Select a connector.                 |
  Columns:                            |                                     |
    NAME                              |   1. Or  (Either condition is true) |
    DEPT                              |   2. And (Both conditions are true) |
    SALARY                            +-------------------------------------+
    COMM                              | F1=Help  F12=Cancel                 |
    SALARY+COMM                       +-------------------------------------+

  Row Conditions:
    If COMM is not NULL
  > ...

  *** END ***
```

Figure 33. The Condition Connectors panel

7. On the Row Conditions panel, enter the column on which you want to base your second row condition. For this example, select the choice **Expression (A+B, etc.)** at the bottom of the Row Conditions panel to create a **SALARY+COMM** column like the one you created in “Creating a column using expressions” on page 40.

8. On the Comparison Operators panel, enter the verb and the comparison operator for the row condition. For this example, select **Is** and **Greater than**.
9. If a panel displays for the comparison operator you selected, enter the value you want to use to select the rows. For this example, enter `17000` without commas or quotes.

   You have created the second row condition. Both row conditions are displayed in the echo area, and the Specify panel displays, with no choice selected for you.



```
PROMPTED QUERY                                     MODIFIED  LINE    1
                                                  _________________________
  Tables:                                         |        Specify         |
    Q.STAFF                                       |                        |
                                                  | Select an item.        |
  Columns:                                        |                        |
    NAME                                          |   1. Tables...         |
    DEPT                                          |   2. Columns...        |
    JOB                                           |   3. Row Conditions... |
    SALARY                                        |   4. Sort...           |
    COMM                                          |   5. Duplicate Rows... |
                                                  |________________________|
  Row Conditions:                                 | F1=Help  F12=Cancel    |
    If COMM Is Not NULL                           |________________________|
    And SALARY+COMM Is Greater Than 17000


  *** END ***
```

Figure 34. The query shows both row conditions you created.

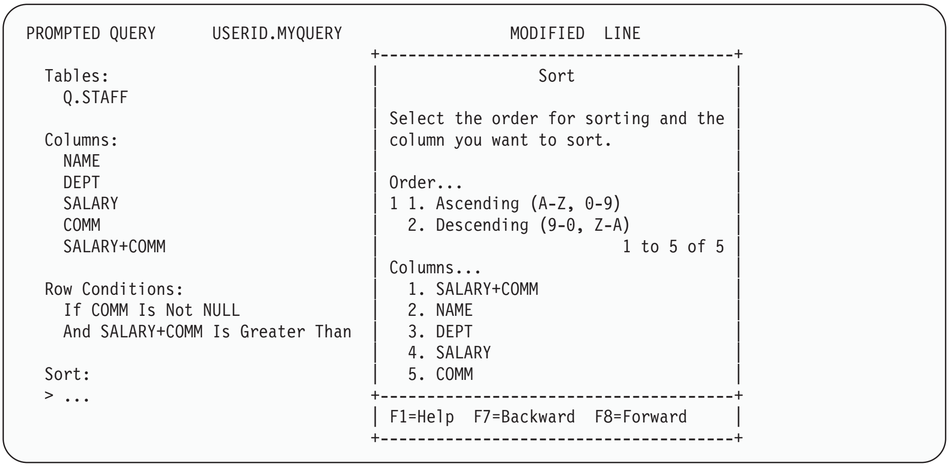
Repeat these steps to create more row conditions. You can create as many row conditions as you need to select the data you want.

Sorting the rows in a query

After you select the rows to display on your report, you can specify the way you want to sort the rows on the report. In this example, you will see how to sort the rows in ascending order by department number.

To sort rows:

1. On the Specify panel, select choice 4, **Sort**. The Sort panel displays with choice 1, **Ascending** already selected for you.



```
PROMPTED QUERY      USERID.MYQUERY                  MODIFIED  LINE
                                     +---------------------------------------+
  Tables:                            |                 Sort                  |
    Q.STAFF                          |                                       |
                                     | Select the order for sorting and the  |
  Columns:                           | column you want to sort.              |
    NAME                             |                                       |
    DEPT                             | Order...                              |
    SALARY                           | 1 1. Ascending (A-Z, 0-9)             |
    COMM                             |   2. Descending (9-0, Z-A)            |
    SALARY+COMM                      |                           1 to 5 of 5 |
                                     | Columns...                            |
  Row Conditions:                    |   1. SALARY+COMM                      |
    If COMM Is Not NULL              |   2. NAME                             |
    And SALARY+COMM Is Greater Than  |   3. DEPT                             |
                                     |   4. SALARY                           |
  Sort:                              |   5. COMM                             |
  > ...                              +---------------------------------------+
                                     | F1=Help  F7=Backward  F8=Forward      |
                                     +---------------------------------------+
```

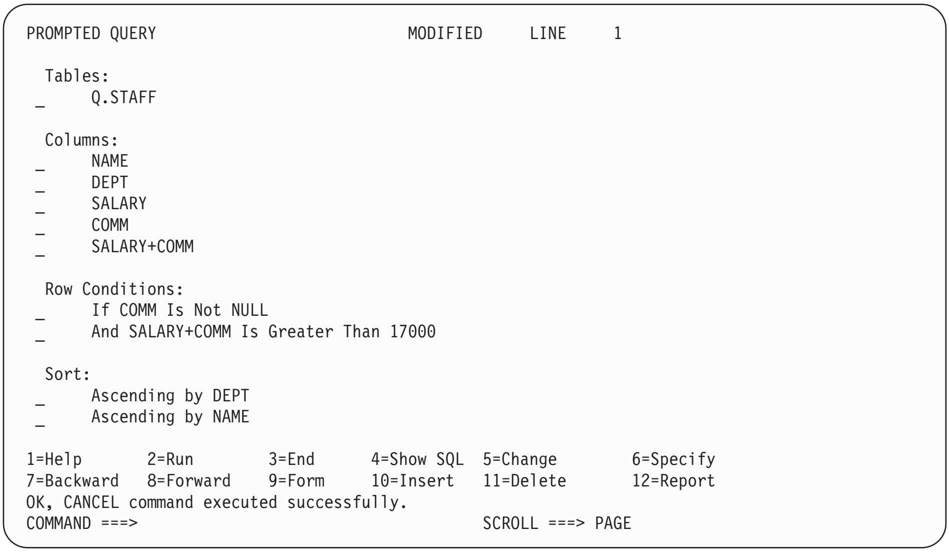
Figure 35. The Sort panel

2. Leave 1 selected if you want to sort the rows in ascending order, or type 2 to sort the rows in descending order.
3. Type the number of the column that contains the data you want to sort.
4. Press Enter.

The sort order you specified is displayed in the echo area, and the Specify panel displays.

Repeat these steps if you want to sort on other columns. For example, in this query, in addition to sorting by department number, you might also want to sort by name within department.

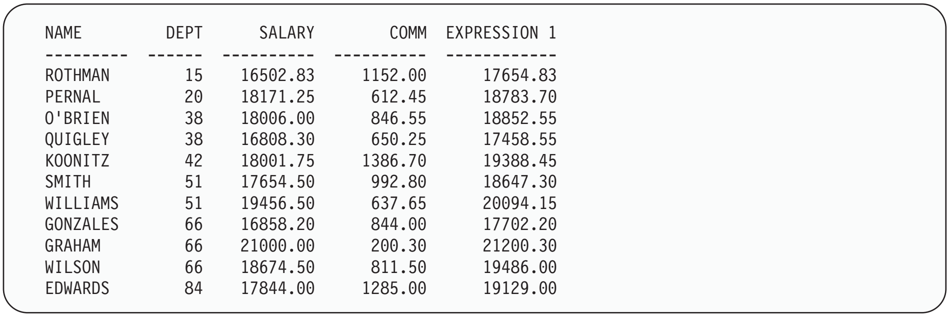
QMF displays the sort order you selected, as shown in Figure 36 on page 49.



```
PROMPTED QUERY                             MODIFIED     LINE     1

  Tables:
 _     Q.STAFF

  Columns:
 _     NAME
 _     DEPT
 _     SALARY
 _     COMM
 _     SALARY+COMM

  Row Conditions:
 _     If COMM Is Not NULL
 _     And SALARY+COMM Is Greater Than 17000

  Sort:
 _     Ascending by DEPT
 _     Ascending by NAME

1=Help       2=Run        3=End      4=Show SQL  5=Change        6=Specify
7=Backward   8=Forward    9=Form     10=Insert   11=Delete       12=Report
OK, CANCEL command executed successfully.
COMMAND ===>                                     SCROLL ===> PAGE
```

Figure 36. QMF displays the order in which rows will be sorted.

QMF displays the report in Figure 37 when you run this query. The rows are first sorted by department number, and then by employee name within each department.



```
  NAME        DEPT      SALARY        COMM  EXPRESSION 1
  ---------  ------  ----------  ----------  ------------
  ROTHMAN        15    16502.83     1152.00      17654.83
  PERNAL         20    18171.25      612.45      18783.70
  O'BRIEN        38    18006.00      846.55      18852.55
  QUIGLEY        38    16808.30      650.25      17458.55
  KOONITZ        42    18001.75     1386.70      19388.45
  SMITH          51    17654.50      992.80      18647.30
  WILLIAMS       51    19456.50      637.65      20094.15
  GONZALES       66    16858.20      844.00      17702.20
  GRAHAM         66    21000.00      200.30      21200.30
  WILSON         66    18674.50      811.50      19486.00
  EDWARDS        84    17844.00     1285.00      19129.00
```

Figure 37. The report is sorted by department number and employee name.

Displaying data from more than one table

With QMF you can display data from more than one table at a time. You can specify up to 15 tables from which to select data.

For example, suppose that you need salary data for each department within each corporate division.

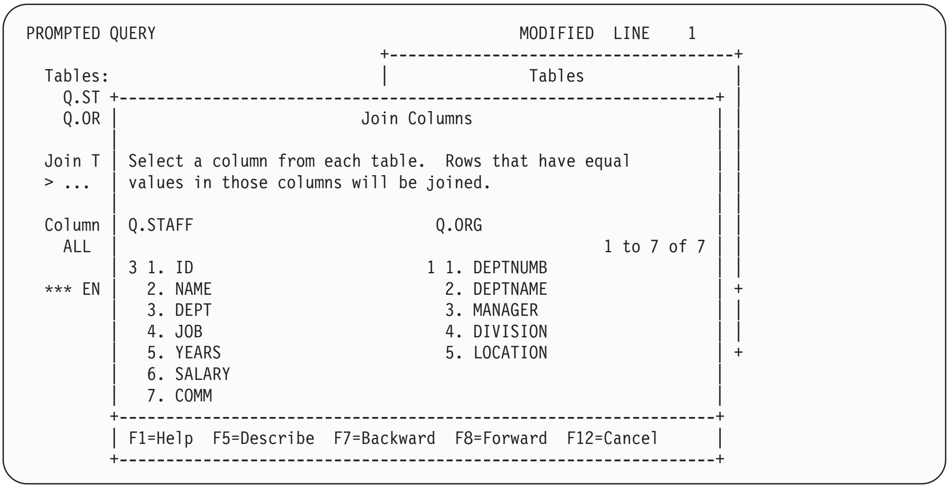
It is not likely that all of this data is in one table. If you check the sample tables in Chapter 17, “QMF sample tables,” on page 285, you see that department numbers are found in both the Q.ORG and Q.STAFF tables, division and department names are found only in the Q.ORG table, and salary data is found only in the Q.STAFF table.

You need to join the tables to get all the data you need. Joining tables means linking them by columns that contain the same kind of information. In this example, the DEPT column in the Q.STAFF table and the DEPTNUMB column in the Q.ORG table both contain department numbers. You can join the tables by these two columns.

When you join the Q.STAFF and Q.ORG tables by department number, QMF joins every row in Q.STAFF to every row in Q.ORG that has the same department number. By joining the tables, you produce a report that shows the names of all departments.

To join tables:

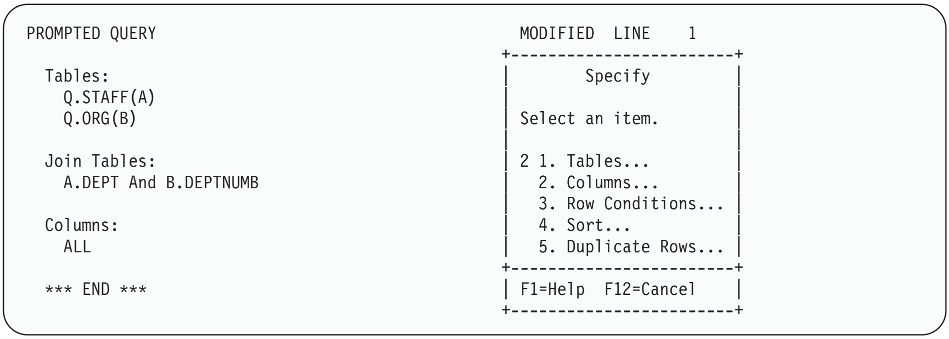
1. On separate lines on the Tables panel, enter the names of the tables from which you want to display data. The Join Columns panel displays. The columns from each table appear under separate headings.



```
PROMPTED QUERY                                       MODIFIED  LINE    1
                                      +-------------------------------------+
  Tables:                             |               Tables                |
    Q.ST +----------------------------------------------------------------+ |
    Q.OR |                          Join Columns                          | |
         |                                                                | |
  Join T | Select a column from each table.  Rows that have equal         | |
  > ...  | values in those columns will be joined.                        | |
         |                                                                | |
  Column | Q.STAFF                          Q.ORG                         | |
    ALL  |                                                    1 to 7 of 7 | |
         | 3 1. ID                         1 1. DEPTNUMB                  | |
  *** EN |   2. NAME                         2. DEPTNAME                  | +
         |   3. DEPT                         3. MANAGER                   | |
         |   4. JOB                          4. DIVISION                  | |
         |   5. YEARS                        5. LOCATION                  | +
         |   6. SALARY                                                    |
         |   7. COMM                                                      |
         +----------------------------------------------------------------+
         | F1=Help  F5=Describe  F7=Backward  F8=Forward  F12=Cancel      |
         +----------------------------------------------------------------+
```

Figure 38. The Join Columns panel

2. Select the column you want from the first table. For this example, type 3 to select the **DEPT** column from the Q.STAFF table.
3. Select the column you want from the second table. For this example, type 1 to select the **DEPTNUMB** column from the Q.ORG table.
4. Press Enter. The columns you used to join the tables are displayed in the echo area. The Specify panel displays, with choice 2, **Columns**, selected for you.



```
PROMPTED QUERY                                     MODIFIED  LINE    1
                                                 +------------------------+
  Tables:                                        |        Specify         |
    Q.STAFF(A)                                   |                        |
    Q.ORG(B)                                     | Select an item.        |
                                                 |                        |
  Join Tables:                                   | 2 1. Tables...         |
    A.DEPT And B.DEPTNUMB                        |   2. Columns...        |
                                                 |   3. Row Conditions... |
  Columns:                                       |   4. Sort...           |
    ALL                                          |   5. Duplicate Rows... |
                                                 +------------------------+
  *** END ***                                    | F1=Help  F12=Cancel    |
                                                 +------------------------+
```

Figure 39. QMF displays the names of the tables you selected to join.

5. Press Enter to select choice 2, **Columns**.

   The Columns panel displays with the names of the columns from all the tables you selected.

6. Enter x beside each column you want to display on the report. For this example, select the DEPTNAME column and the DIVISION column. The columns you select to display on the report appear in the echo area. The Specify panel displays, with choice 3, **Row Conditions**.

7. To get the total salary for each department, use QMF's summary functions. On the Specify panel, select choice 2, **Columns**. The Columns panel displays.

8. Select choice 2, **Summary Functions (SUM, etc.)** at the bottom of the panel. The Summary Functions panel displays.

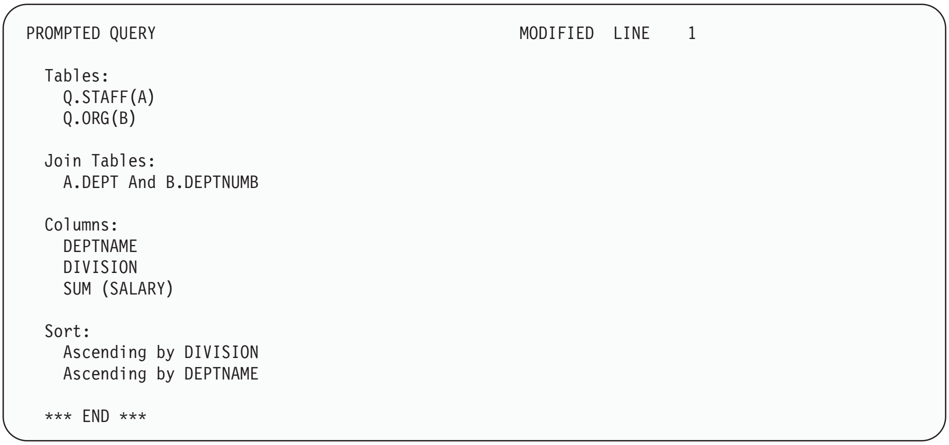
9. Select the summary function. For this example, select **Sum of**. The Summary Function Items panel displays.

10. Select the column you want the summary function to act on. For this example, select the SALARY column.

11. To finish the query, sort the rows by department name and division. On the Specify panel, select choice 4, **Sort**. The Sort panel displays.

12. For this example, select **Ascending** and the DIVISION column.

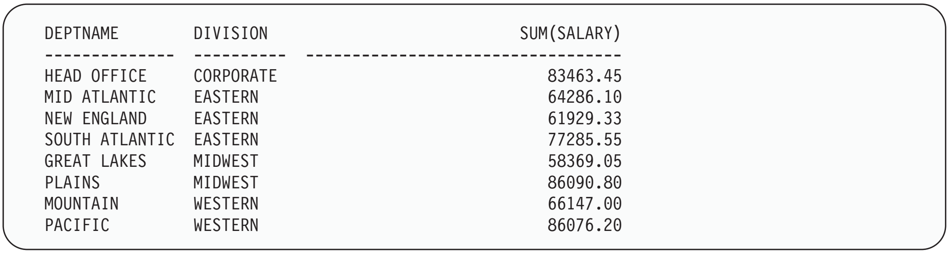
    Repeat these steps to sort the DEPTNAME column in ascending order.



```
PROMPTED QUERY                                        MODIFIED  LINE    1

  Tables:
    Q.STAFF(A)
    Q.ORG(B)

  Join Tables:
    A.DEPT And B.DEPTNUMB

  Columns:
    DEPTNAME
    DIVISION
    SUM (SALARY)

  Sort:
    Ascending by DIVISION
    Ascending by DEPTNAME

  *** END ***
```

Figure 40. QMF displays the finished query.

13. Press the Run function key to see the final report, showing the total salary data for each department within each division.

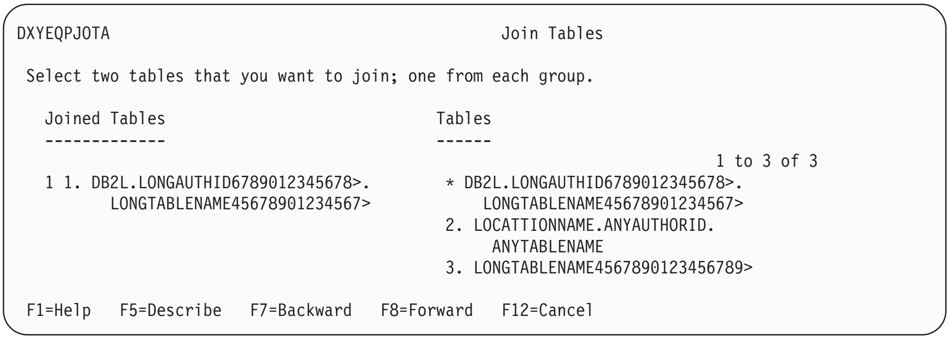


```
  DEPTNAME        DIVISION                           SUM(SALARY)
  --------------  ----------  ----------------------------------
  HEAD OFFICE     CORPORATE                             83463.45
  MID ATLANTIC    EASTERN                               64286.10
  NEW ENGLAND     EASTERN                               61929.33
  SOUTH ATLANTIC  EASTERN                               77285.55
  GREAT LAKES     MIDWEST                               58369.05
  PLAINS          MIDWEST                               86090.80
  MOUNTAIN        WESTERN                               66147.00
  PACIFIC         WESTERN                               86076.20
```

Figure 41. The report shows salary data for departments within divisions.

Long table names

In DB2 QMF Version 8.1, long table names are supported. If the table name is too long to display in the Join Tables panel, the table name is split into two lines. The location name and the authorization ID are on the first line with a ″>″ sign placed at position 29 followed by a dot separator. The table name is on the second line with a ″>″ sign placed at position 30 indicating that there is more text. The PF5 Describe key is used to display the full table name.

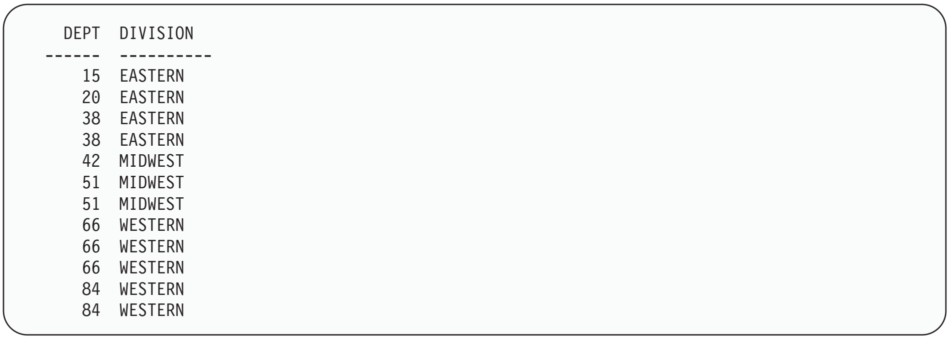


```
DXYEQPJOTA                                          Join Tables

 Select two tables that you want to join; one from each group.

   Joined Tables                             Tables
   -------------                             ------
                                                                           1 to 3 of 3
   1 1. DB2L.LONGAUTHID6789012345678>.        * DB2L.LONGAUTHID6789012345678>.
          LONGTABLENAME45678901234567>            LONGTABLENAME45678901234567>
                                              2. LOCATTIONNAME.ANYAUTHORID.
                                                   ANYTABLENAME
                                              3. LONGTABLENAME4567890123456789>

 F1=Help   F5=Describe   F7=Backward   F8=Forward   F12=Cancel
```

Figure 42. Join tables panel

Eliminating duplicate rows from the report

In Prompted Query, you can request that rows with duplicate information not appear in your report. For example, if you create a query to show all departments that have salespeople based on the sample tables, the report will show duplicate rows for each department with more than one salesperson.

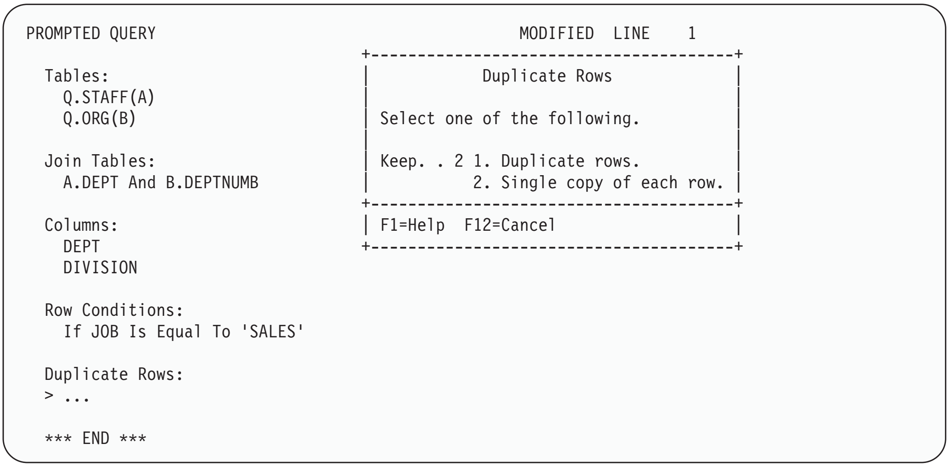


```
     DEPT  DIVISION
   ------  ----------
       15  EASTERN
       20  EASTERN
       38  EASTERN
       38  EASTERN
       42  MIDWEST
       51  MIDWEST
       51  MIDWEST
       66  WESTERN
       66  WESTERN
       66  WESTERN
       84  WESTERN
       84  WESTERN
```

Figure 43. The report shows duplicate rows for several departments.

To eliminate duplicate rows:

1. On the Specify panel, select **Duplicate Rows**. The Duplicate Rows panel displays.

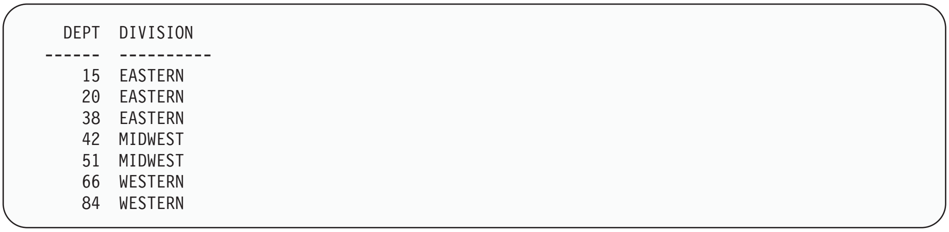


```
PROMPTED QUERY                                       MODIFIED  LINE    1
                                    +----------------------------------------+
  Tables:                           |            Duplicate Rows              |
    Q.STAFF(A)                      |                                        |
    Q.ORG(B)                        | Select one of the following.           |
                                    |                                        |
  Join Tables:                      | Keep. . 2 1. Duplicate rows.           |
    A.DEPT And B.DEPTNUMB           |           2. Single copy of each row.  |
                                    +----------------------------------------+
  Columns:                          | F1=Help  F12=Cancel                    |
    DEPT                            +----------------------------------------+
    DIVISION

  Row Conditions:
    If JOB Is Equal To 'SALES'

  Duplicate Rows:
  > ...

  *** END ***
```

Figure 44. The Duplicate Rows panel

2. Select **Single copy of each row**. The Prompted Query panel displays with your query in the echo area. The Specify panel does not display, because eliminating duplicate rows is the last step in creating a prompted query. Press the Specify key to display the Specify panel if you want to work on your query again.

   When you run the query, QMF displays a report that shows one row for each department, as shown in Figure 45.



```
    DEPT  DIVISION
  ------  ----------
      15  EASTERN
      20  EASTERN
      38  EASTERN
      42  MIDWEST
      51  MIDWEST
      66  WESTERN
      84  WESTERN
```

Figure 45. The report shows only one row for each department.

Joining multiple tables

This example uses the Q.SUPPLIER, Q.PARTS, and Q.PROJECT tables to create a query that shows the supplier name, the part name, the project number, and the start date for all the parts used by each project in the Q.PROJECT table.

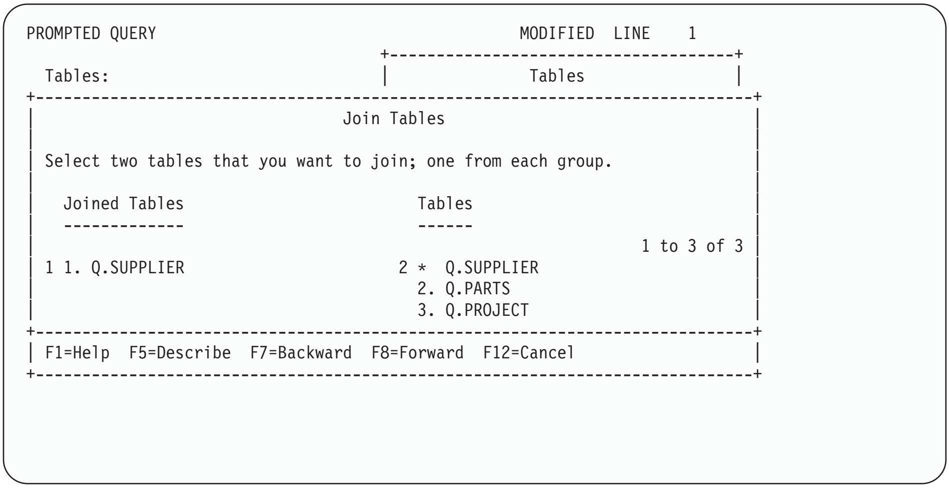
If you look at the sample table in Chapter 17, “QMF sample tables,” on page 285, you see the supplier name is in Q.SUPPLIER as COMPANY, the part name is in Q.PARTS as PARTNAME, and the project number is in Q.PROJECT as PROJNO. To get all the information for the report, you need to join these three tables.

To join multiple tables, you join two tables at a time by a common column. In this example, the ACCTNO column in Q.SUPPLIER and the SUPPNO column in Q.PARTS contain the same information, so you can join those two tables. Likewise, the PRODNUM column in Q.PARTS and the PRODNO column in Q.PROJECT contain the same information, so you can also join those tables.

To join multiple tables:

1. On separate lines on the Tables panel, enter the names of the tables from which you want to display data. For this example, enter `Q.SUPPLIER`, `Q.PARTS`, and `Q.PROJECT`.

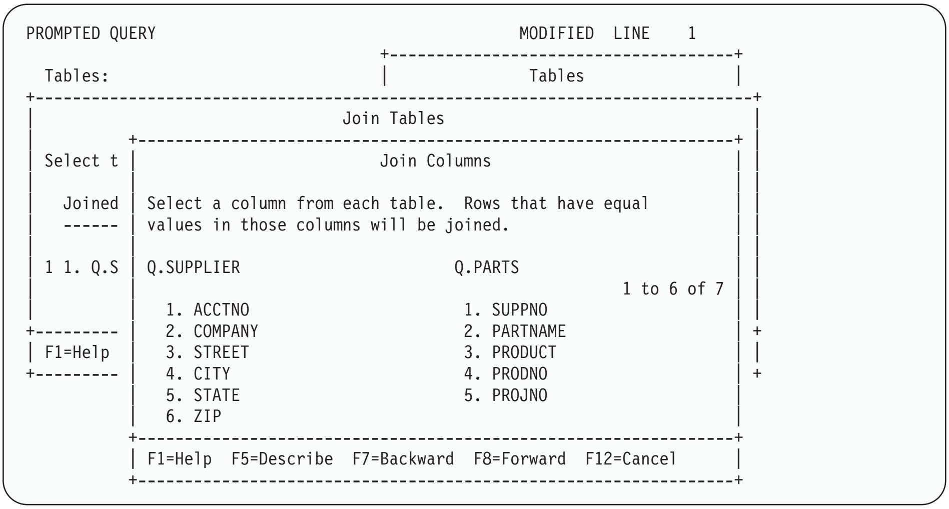
   The Join Tables panel displays. The first table you selected appears under the heading **Joined Tables**. Select the second table you want to join from the list under the heading **Tables**.



```
PROMPTED QUERY                                      MODIFIED  LINE    1
                                      +--------------------------------------+
  Tables:                             |               Tables                 |
+-----------------------------------------------------------------------------+
|                                 Join Tables                                 |
|                                                                             |
| Select two tables that you want to join; one from each group.               |
|                                                                             |
|   Joined Tables                        Tables                               |
|   -------------                        ------                               |
|                                                                  1 to 3 of 3 |
| 1 1. Q.SUPPLIER                     2 *  Q.SUPPLIER                          |
|                                       2. Q.PARTS                            |
|                                       3. Q.PROJECT                          |
+-----------------------------------------------------------------------------+
| F1=Help  F5=Describe  F7=Backward  F8=Forward  F12=Cancel                   |
+-----------------------------------------------------------------------------+
```

Figure 46. The Join Tables panel

2. From the **Joined Tables** heading, select the first table you want to join. For this example, select the Q.SUPPLIER table.
3. From the **Tables** heading, select the second table you want to join. For this example, select the Q.PARTS table.
4. Press Enter. The Join Columns panel displays.

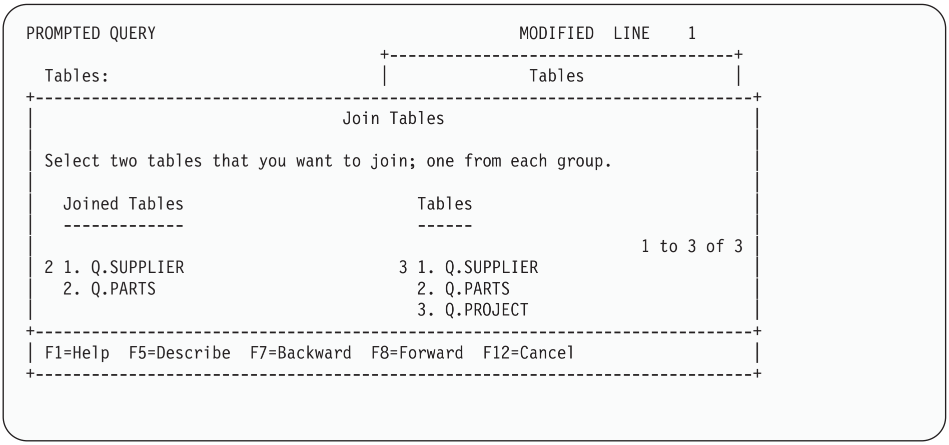


```
PROMPTED QUERY                                       MODIFIED  LINE    1
                                      +--------------------------------------+
  Tables:                             |               Tables                 |
+------------------------------------------------------------------------------+
|                                 Join Tables                                  |
|          +-----------------------------------------------------------------+ |
| Select t |                          Join Columns                           | |
|          |                                                                 | |
|   Joined | Select a column from each table.  Rows that have equal          | |
|   ------ | values in those columns will be joined.                         | |
|          |                                                                 | |
| 1 1. Q.S | Q.SUPPLIER                       Q.PARTS                        | |
|          |                                                    1 to 6 of 7  | |
|          |   1. ACCTNO                       1. SUPPNO                     | |
+--------- |   2. COMPANY                      2. PARTNAME                   | +
| F1=Help  |   3. STREET                       3. PRODUCT                    | |
+--------- |   4. CITY                         4. PRODNO                     | +
           |   5. STATE                        5. PROJNO                     |
           |   6. ZIP                                                        |
           +-----------------------------------------------------------------+
           | F1=Help  F5=Describe  F7=Backward  F8=Forward  F12=Cancel       |
           +-----------------------------------------------------------------+
```

Figure 47. The Join Columns panel

5. From the columns that are listed for the first table, select the column on which you want to join the tables. For this example, select **ACCTNO**.
6. From the columns that are listed for the second table, select the column that contains the same kind of data. For this example, select **SUPPNO**.
7. Press Enter.

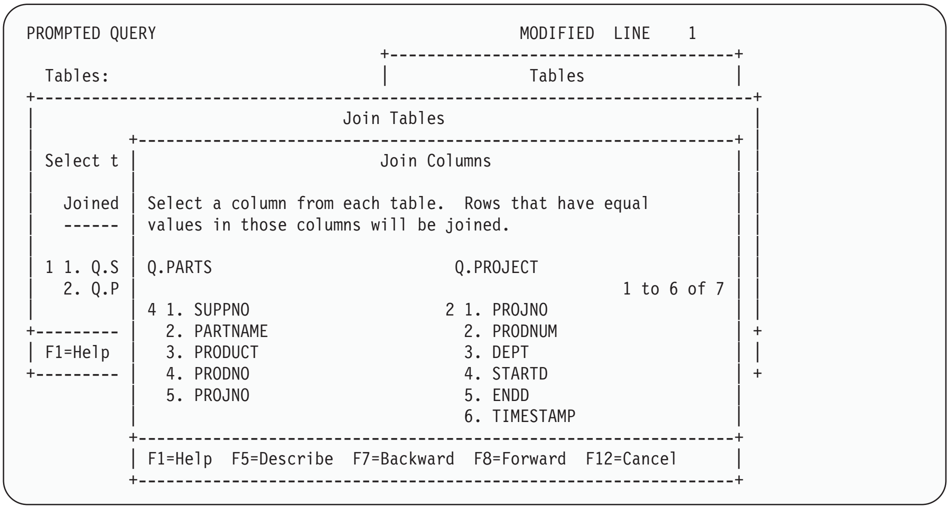
   You have joined the first two tables. The Join Tables panel displays again. This time, both tables you have joined appear under the heading **Joined Tables**. You select the last table to join with the other two from the list under the heading **Tables**. For this example, select **Q.PARTS** from the list under **Joined Tables**. Select **Q.PROJECT** from the list under **Tables**.
   The Join Columns panel displays again.



```
PROMPTED QUERY                                       MODIFIED  LINE    1
                                      +--------------------------------------+
  Tables:                             |               Tables                 |
+------------------------------------------------------------------------------+
|                                 Join Tables                                  |
|                                                                              |
| Select two tables that you want to join; one from each group.                |
|                                                                              |
|   Joined Tables                         Tables                               |
|   -------------                         ------                               |
|                                                                 1 to 3 of 3  |
| 2 1. Q.SUPPLIER                       3 1. Q.SUPPLIER                        |
|   2. Q.PARTS                            2. Q.PARTS                           |
|                                         3. Q.PROJECT                         |
+------------------------------------------------------------------------------+
| F1=Help  F5=Describe  F7=Backward  F8=Forward  F12=Cancel                    |
+------------------------------------------------------------------------------+
```

Figure 48. The Join Tables panel displays the tables you already joined.

8. Select the columns you want from each table. For this example, select **PRODNO** from the Q.PARTS table. Select **PRODNUM** from the Q.PROJECT table.
9. Press Enter.



```
PROMPTED QUERY                                   MODIFIED  LINE    1
                                    +-------------------------------------+
  Tables:                           |              Tables                 |
+---------------------------------------------------------------------------+
|                                Join Tables                                |
|          +-----------------------------------------------------------------+ |
| Select t |                        Join Columns                         | |
|          |                                                             | |
|   Joined | Select a column from each table.  Rows that have equal      | |
|   ------ | values in those columns will be joined.                     | |
|          |                                                             | |
| 1 1. Q.S | Q.PARTS                          Q.PROJECT                  | |
|   2. Q.P |                                               1 to 6 of 7   | |
|          | 4 1. SUPPNO                    2 1. PROJNO                  | |
+--------- |   2. PARTNAME                    2. PRODNUM                 | +
| F1=Help  |   3. PRODUCT                     3. DEPT                    | |
+--------- |   4. PRODNO                      4. STARTD                  | +
           |   5. PROJNO                      5. ENDD                    |
           |                                  6. TIMESTAMP               |
           +-----------------------------------------------------------------+
           | F1=Help  F5=Describe  F7=Backward  F8=Forward  F12=Cancel   |
           +-----------------------------------------------------------------+
```

Figure 49. QMF displays columns from the second set of tables.

You have joined the second and third tables.

If you have more tables to join, the Join Tables panel displays. If you have joined all your tables, the Specify panel displays, so you can select the columns to display on the report.

Joining multiple columns

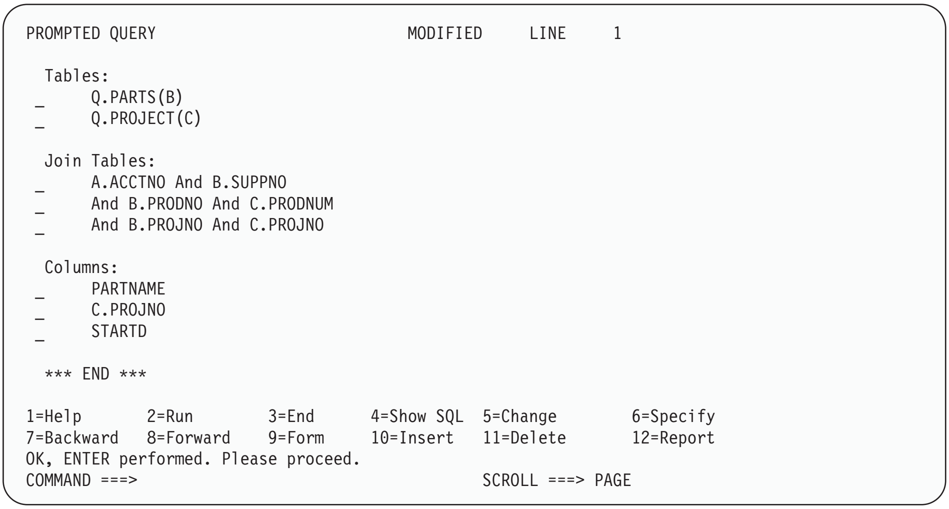
You might need to join tables on more than one column. For example, if you join the Q.PARTS table and the Q.PROJECT table on their PRODNO and PRODNUM columns alone, you might get a report showing some incorrect project numbers for the products. That is because the sample tables have more than one project number for some products. You need to join the Q.PARTS and Q.PROJECTS tables at both of the columns that share the same type of information. You must join the tables on the PROJNO columns as well as the PRODNO and PRODNUM columns.

To join multiple columns:

1. Create or display a query that joins two tables at a single column. For this example, display a query that joins the Q.PARTS and Q.PROJECT tables.
2. Press the Cancel function key to remove the Specify panel from the Prompted Query panel.

3. Move the cursor to the last pair of joined columns under the **Join Tables** heading.
4. Press the Insert function key. The Join Tables panel displays.
5. Select the tables you want to join on an additional column. For this example, select the Q.PARTS and the Q.PROJECT tables again. The Join Columns panel displays.
6. Select the columns you want to join. For this example, select the **PROJNO** columns from both tables.

   The query displays with the additional columns you joined.



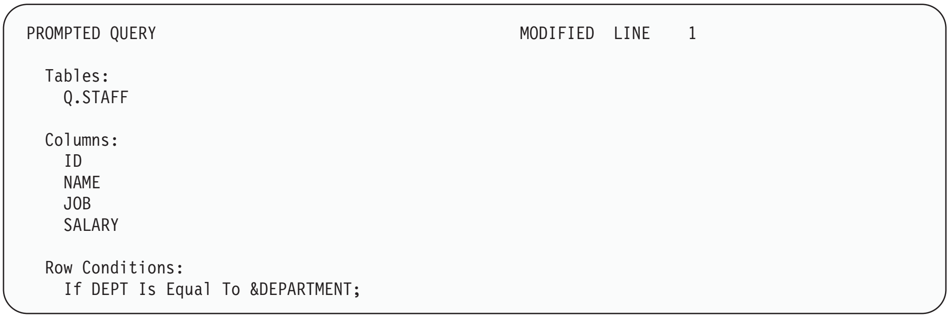
```
PROMPTED QUERY                           MODIFIED     LINE     1

  Tables:
 _     Q.PARTS(B)
 _     Q.PROJECT(C)

  Join Tables:
 _     A.ACCTNO And B.SUPPNO
 _     And B.PRODNO And C.PRODNUM
 _     And B.PROJNO And C.PROJNO

  Columns:
 _     PARTNAME
 _     C.PROJNO
 _     STARTD

  *** END ***

1=Help       2=Run        3=End      4=Show SQL  5=Change        6=Specify
7=Backward   8=Forward    9=Form     10=Insert   11=Delete       12=Report
OK, ENTER performed. Please proceed.
COMMAND ===>                                 SCROLL ===> PAGE
```

Figure 50. QMF shows that the two tables are joined at a second column.

Making your query reusable with substitution variables

When you specify substitution variables in a prompted query, you can use the same query to retrieve different information by supplying a new value for the variable each time you run the query.

The prompted query in Figure 51 on page 59 selects department data. By using a substitution variable (&DEPARTMENT) for the department number in the row condition, you can specify a different department number each time you run the query.



```
PROMPTED QUERY                                       MODIFIED  LINE    1

  Tables:
    Q.STAFF

  Columns:
    ID
    NAME
    JOB
    SALARY

  Row Conditions:
    If DEPT Is Equal To &DEPARTMENT;
```

Figure 51. This query uses a substitution variable for the DEPT name.

You can enter substitution variables on any Prompted Query panel where you can enter expressions.

You can specify values for substitution variables in any of the following ways:

- As a part of the RUN command
- From the RUN command prompt panel
- By setting a global variable

To specify a value as part of the RUN command: For example, to specify a value for the &DEPARTMENT variable, on the QMF command line enter:

```
RUN QUERY (&DEPARTMENT = 38
```

Enclose the value in parentheses if it contains one of the following special characters:

- Blank
- Comma
- Left or right parenthesis
- Single or double quotation mark
- Equal sign

For example:

```
RUN QUERY (&X=(DEPT,NAME,SALARY)
```

Type the text to specify a variable. You might have to enclose the text with quotes. For example, the following query has two variables. For the first variable, you specify a column name as the value; for the second, you specify text that contains a quotation mark.

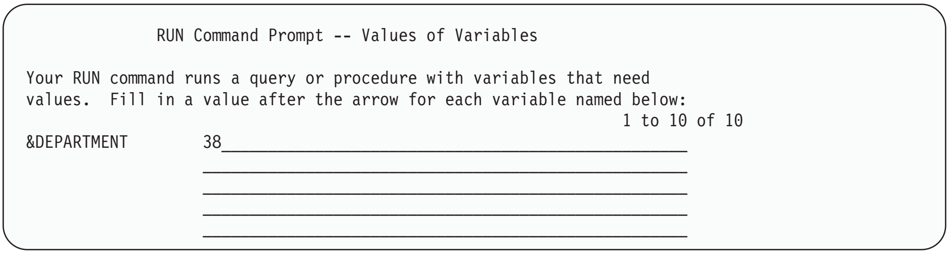
```
SELECT &X
FROM Q.STAFF
WHERE NAME=&Y
```

If the text itself contains quotation marks, add another set of quotation marks for each quotation mark:

```
RUN QUERY (&X=SALARY, &Y='O''BRIEN'
```

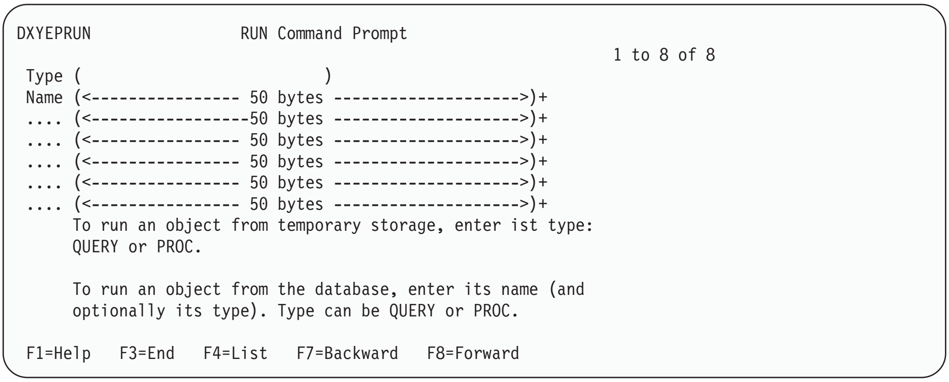
To specify a value on the RUN Command Prompt panel: If your query contains a variable, and you do not specify a value for the variable when you type the RUN command, the RUN Command Prompt panel displays.

The variables that need values appear on the panel. Type the values for the variables.



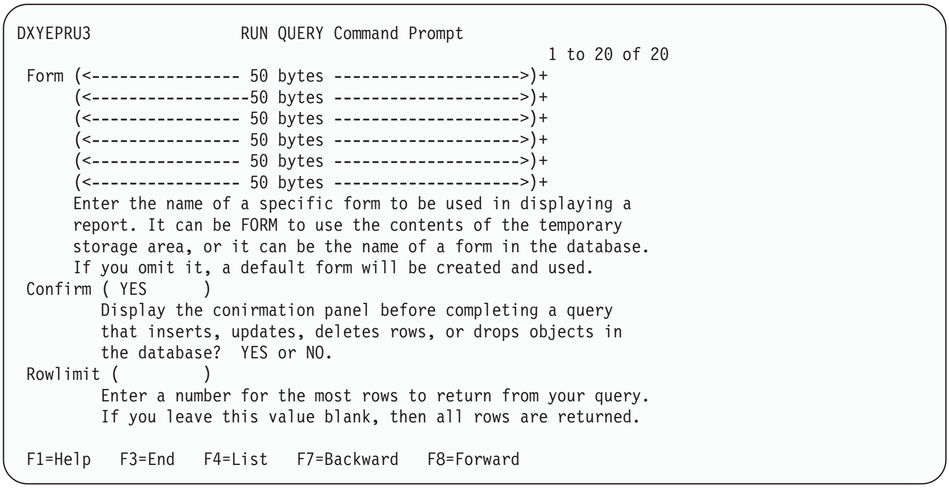
```
              RUN Command Prompt -- Values of Variables

Your RUN command runs a query or procedure with variables that need
values.  Fill in a value after the arrow for each variable named below:
                                                            1 to 10 of 10
&DEPARTMENT        38____________________________________________________
                   ______________________________________________________
                   ______________________________________________________
                   ______________________________________________________
                   ______________________________________________________
```

In DB2 QMF Version 8.1, the RUN command supports long owner and table names. The RUN command prompt panel now allows for the entry of long names for tables and views. The Name entry field is increased from 50 to 280 bytes which will accommodate an object name of the form: ″location(16)″.″authid(128)″.″object name(128)″. Below are examples of the new RUN command prompt screen and the RUN QUERY prompt screen:



```
DXYEPRUN                RUN Command Prompt
                                                            1 to 8 of 8
 Type (                           )
 Name (<---------------- 50 bytes -------------------->)+
 .... (<-----------------50 bytes -------------------->)+
 .... (<---------------- 50 bytes -------------------->)+
 .... (<---------------- 50 bytes -------------------->)+
 .... (<---------------- 50 bytes -------------------->)+
 .... (<---------------- 50 bytes -------------------->)+
      To run an object from temporary storage, enter ist type:
      QUERY or PROC.

      To run an object from the database, enter its name (and
      optionally its type). Type can be QUERY or PROC.

 F1=Help   F3=End   F4=List   F7=Backward   F8=Forward
```

Figure 52. RUN command prompt screen



```
DXYEPRU3                RUN QUERY Command Prompt
                                                                 1 to 20 of 20
 Form (<---------------- 50 bytes -------------------->)+
      (<-----------------50 bytes -------------------->)+
      (<---------------- 50 bytes -------------------->)+
      (<---------------- 50 bytes -------------------->)+
      (<---------------- 50 bytes -------------------->)+
      (<---------------- 50 bytes -------------------->)+
      Enter the name of a specific form to be used in displaying a
      report. It can be FORM to use the contents of the temporary
      storage area, or it can be the name of a form in the database.
      If you omit it, a default form will be created and used.
 Confirm ( YES      )
         Display the conirmation panel before completing a query
         that inserts, updates, deletes rows, or drops objects in
         the database?  YES or NO.
 Rowlimit (         )
         Enter a number for the most rows to return from your query.
         If you leave this value blank, then all rows are returned.

 F1=Help   F3=End   F4=List   F7=Backward   F8=Forward
```

Figure 53. RUN Query command prompt

To specify values for substitution variables using global variables: You can define global variables with the SET GLOBAL command. A global variable keeps its value until you reset it, or until you end the QMF session.

For example, to set a global variable value for the &DEPARTMENT variable, on the QMF command line enter:

```
SET GLOBAL (DEPARTMENT=38
```

You can specify up to 10 variable values. Separate the values with either commas or blanks.

For more information on defining global variables, see *DB2 QMF Reference*.

Running a query and displaying a report

When you run a query, QMF displays the data you select as a report.

You can run a query with only one table selected, and some or all columns selected.

To run a query:

1. From the Specify panel, press the Cancel function key. The Prompted Query panel displays your query.
2. Press the Run function key.

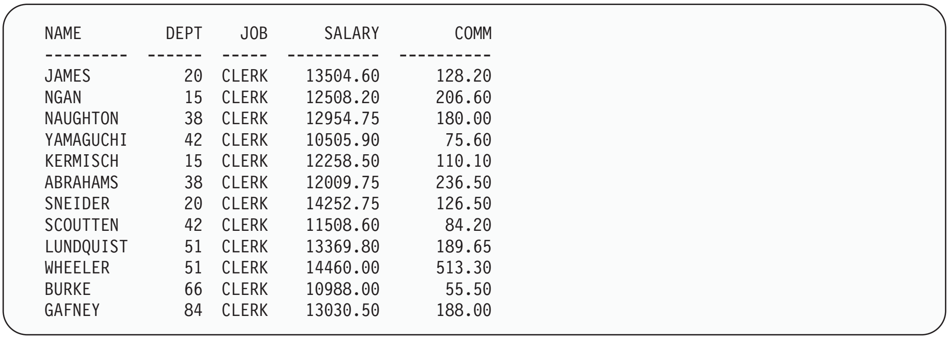
   You can also:

   Enter `RUN QUERY` on the QMF command line.

Viewing the data using prompted query

When QMF finishes running your query, it displays a report that shows all of the data that you selected.

If your query selects a large number of rows, you might need to scroll forward to see all the data.



```
NAME         DEPT   JOB      SALARY        COMM
---------  ------  -----  ----------  ----------
JAMES          20  CLERK    13504.60      128.20
NGAN           15  CLERK    12508.20      206.60
NAUGHTON       38  CLERK    12954.75      180.00
YAMAGUCHI      42  CLERK    10505.90       75.60
KERMISCH       15  CLERK    12258.50      110.10
ABRAHAMS       38  CLERK    12009.75      236.50
SNEIDER        20  CLERK    14252.75      126.50
SCOUTTEN       42  CLERK    11508.60       84.20
LUNDQUIST      51  CLERK    13369.80      189.65
WHEELER        51  CLERK    14460.00      513.30
BURKE          66  CLERK    10988.00       55.50
GAFNEY         84  CLERK    13030.50      188.00
```

Figure 54. QMF displays your data as a report.

3. If you want to make changes to the query, press the Query function key to return to the Prompted Query panel.

Saving a new query

You can save your query in the database after you create it. You can run a saved query and display the report again. You can add, delete, or change the information in a saved query.

To save a query: On the QMF command line on the Prompted Query panel, enter:

```
SAVE
```

QMF prompts you for the name to assign to the query.

You can also enter:

```
SAVE AS queryname
```

For example, to save your query in the database and name it MYQUERY, enter:

```
SAVE AS MYQUERY
```

To save a query and share it with other users, add the SHARE=YES parameter to the SAVE command you are using as follows:

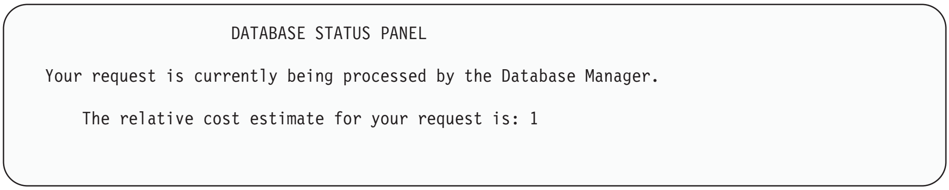
```
SAVE (SHARE=YES
SAVE AS queryname (SHARE=YES
```

QMF saves your query in the database. The Prompted Query panel displays with the name you gave the query. If you issue a SET GLOBAL command with the value DSQEC_SHARE=1 before issuing the SAVE command, the SHARE=YES parameter is not needed.

In some cases, a long report might not be complete when you attempt to save your query. When this happens, QMF cannot save the query until the report is complete. This could create performance problems. The global variable DSQEC_RESET_RPT allows you to predefine how you want QMF to handle this situation. See *DB2 QMF Reference* for more information.

Canceling a running query

There are occasions when you might want to cancel a running query. For example, you realize that your query will take too long to run. While a query is running, a database status panel like the one in Figure 55 shows you the relative "cost" for your query in terms of computer resources.



```
                    DATABASE STATUS PANEL

 Your request is currently being processed by the Database Manager.

     The relative cost estimate for your request is: 1
```

Figure 55. The Database Status panel

Canceling a query using the QMF command interrupt

To cancel a query using the QMF Command Interrupt:

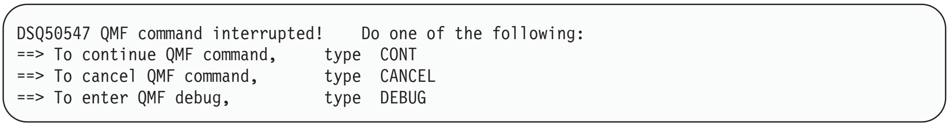
1. While the Database Status panel displays, press the PA1 key.

   See your QMF administrator if you need help finding this key on your terminal.

   The following message displays:

   ```
   DSQ50465 QMF command interrupted!  Clear screen and press ENTER.
   ```
2. Press the Clear function key.
3. Press Enter.

   The QMF Command Interrupt panel displays, as shown in Figure 56 on page 64.



```
DSQ50547 QMF command interrupted!    Do one of the following:
==> To continue QMF command,     type  CONT
==> To cancel QMF command,       type  CANCEL
==> To enter QMF debug,          type  DEBUG
```

Figure 56. The QMF Command Interrupt panel

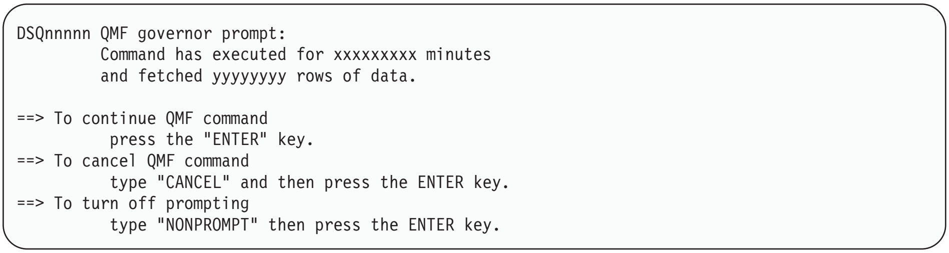
4. On the QMF command line, enter `CANCEL`.

QMF cancels the query.

Canceling a query by using the QMF governor prompt panel

Your installation might have a governor interrupt routine that automatically attempts to cancel a query that takes a long time to run or tries to retrieve too many rows. If a QMF Governor Prompt panel similar to the one in Figure 57 displays while you are trying to run a query, follow the instructions on the panel to cancel the query or continue.

If you continue, the QMF governor can still cancel your command.



```
DSQnnnnn QMF governor prompt:
         Command has executed for xxxxxxxxx minutes
         and fetched yyyyyyyy rows of data.

==> To continue QMF command
          press the "ENTER" key.
==> To cancel QMF command
          type "CANCEL" and then press the ENTER key.
==> To turn off prompting
          type "NONPROMPT" then press the ENTER key.
```

Figure 57. The Governor Prompt panel

Making changes to a saved query

After you save a query in the database, you can make changes to it. First, retrieve the query from the database, then make the changes.

Retrieving a query from the database

To retrieve a query from the database, on the QMF command line enter:

```
DISPLAY QUERY queryname
```

The Prompted Query panel displays with the query you requested from the database.

Correcting a query that does not successfully run

If you cannot successfully run a query, one or more of the database objects that are specified in the query probably has been updated. For example, a table name has changed, or a column has been deleted from a table since the last time you ran the prompted query. You cannot make the changes to the query from within QMF. You must convert or export the query to change it.

To correct the information in the query: Convert the prompted query to a SQL query, display the query, and make the changes. See "Converting a prompted query to an SQL query" on page 67 for information on converting a prompted query to an SQL query.

Adding information to a query

You can add information to a query before or after you save it. You can add specifications and you can change existing specifications.

To add a new specification to a query:

1. On the Prompted Query panel, press the Specify function key.
2. On the Specify panel, enter the number of the specification you want to add.

To add to an existing specification:

1. Move the cursor to the place in the query where you want to add information, and press the Insert function key. The appropriate panel displays.
2. For example, move the cursor to Row Conditions and press the Insert function key if you want to add a row condition. The Row Conditions panel displays.
3. Enter the information you want to add to the query.

   After you press Enter on the last panel, or press the Cancel function key to close the Specify panel, the Prompted Query panel displays with the information you added.

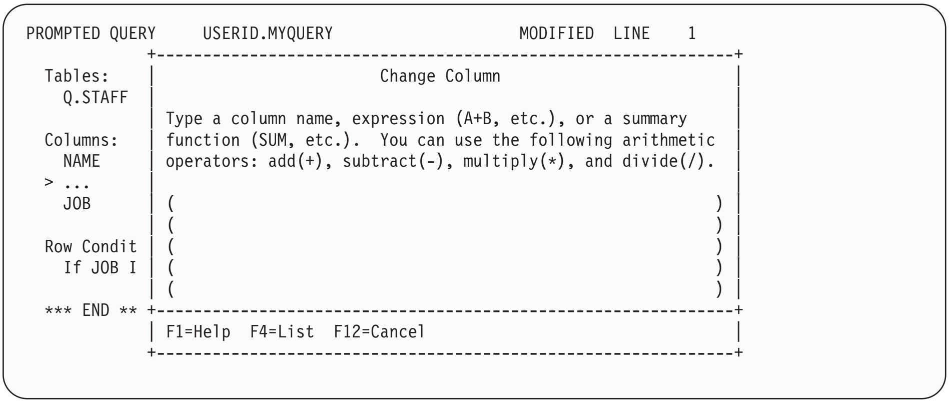
Changing information in a query

You can change column names, change row conditions, and sort information in a query you have saved. You cannot change table names, but you can delete a table and specify a new one.

To change information in a query:

1. Move the cursor to the information you want to change, and press the Change function key.

   The appropriate change panel, like the one in Figure 58 on page 66, displays.



```
PROMPTED QUERY     USERID.MYQUERY                    MODIFIED  LINE    1
             +---------------------------------------------------------------+
  Tables:    |                        Change Column                          |
    Q.STAFF  |                                                               |
             | Type a column name, expression (A+B, etc.), or a summary      |
  Columns:   | function (SUM, etc.).  You can use the following arithmetic   |
    NAME     | operators: add(+), subtract(-), multiply(*), and divide(/).   |
  > ...      |                                                               |
    JOB      | (                                                           ) |
             | (                                                           ) |
  Row Condit | (                                                           ) |
    If JOB I | (                                                           ) |
             | (                                                           ) |
  *** END ** +---------------------------------------------------------------+
             | F1=Help  F4=List  F12=Cancel                                  |
             +---------------------------------------------------------------+
```

Figure 58. QMF displays a panel where you specify changes to a query.

2. Enter the changes to the information.

The Prompted Query panel displays with the changed information in the echo area.

Deleting information from a query

You can delete any information from a query, including table names.

To delete information from a query:

Move the cursor to the line you want to delete and press the Delete function key.

Remember the following information when you are deleting tables or table joins from a query:

- When you delete a table from a query, QMF also deletes any table joins you created with that table.
- When you change a query so that two or more of the tables in the query are no longer joined, the Join Tables panel displays. You can link the tables by another common column.

Erasing a saved query

You can erase any query you have saved in the database.

To erase a query in the database: On the QMF command line, enter:

ERASE QUERY *queryname*

Only a QMF administrator can erase queries that are saved by others.

Viewing the SQL equivalent of a prompted query

You might want to see the SQL statements that make up a query you create with Prompted Query. For example, you might want to know if your prompted query is equivalent to another SQL query.

When you display the SQL equivalent of a prompted query, you cannot edit, run, or save the query you display.

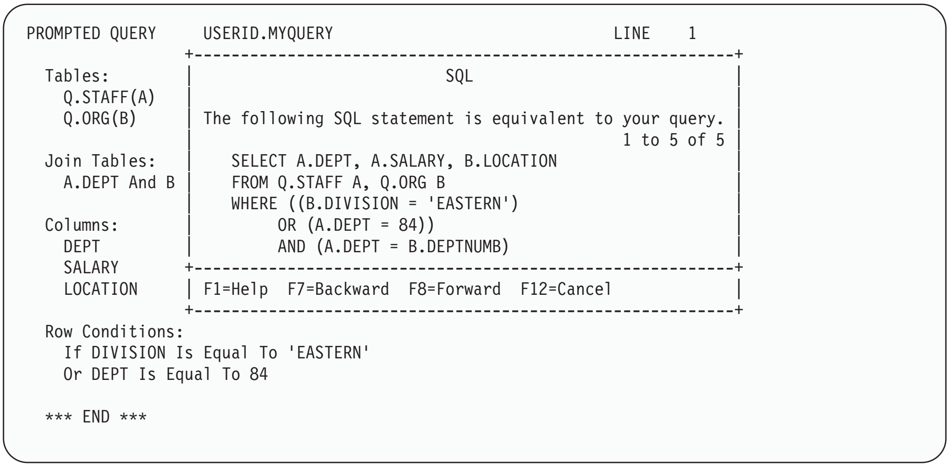
To display the SQL equivalent of a prompted query:

1. Display the prompted query on the Prompted Query panel.
2. Press the Show SQL function key.

   Or you can:

   Enter `SHOW SQL` on the command line.

   The SQL equivalent of the prompted query displays.



```
PROMPTED QUERY     USERID.MYQUERY                              LINE    1
                  +----------------------------------------------------------+
  Tables:         |                           SQL                            |
    Q.STAFF(A)    |                                                          |
    Q.ORG(B)      | The following SQL statement is equivalent to your query. |
                  |                                             1 to 5 of 5 |
  Join Tables:    |    SELECT A.DEPT, A.SALARY, B.LOCATION                   |
    A.DEPT And B  |    FROM Q.STAFF A, Q.ORG B                               |
                  |    WHERE ((B.DIVISION = 'EASTERN')                       |
  Columns:        |         OR (A.DEPT = 84))                                |
    DEPT          |         AND (A.DEPT = B.DEPTNUMB)                        |
    SALARY        +----------------------------------------------------------+
    LOCATION      | F1=Help  F7=Backward  F8=Forward  F12=Cancel             |
                  +----------------------------------------------------------+
  Row Conditions:
    If DIVISION Is Equal To 'EASTERN'
    Or DEPT Is Equal To 84

  *** END ***
```

Figure 59. QMF can display the SQL equivalent of a prompted query.

You will learn more about SQL in Chapter 5, “Viewing the data in the database using SQL statements,” on page 69.

Converting a prompted query to an SQL query

You can convert a prompted query to an SQL query. Converting a query is useful if you would like to expand a basic prompted query into a more complex query by using the SQL language.

After you convert a prompted query into an SQL query, you cannot convert it back into a prompted query. If you want to keep a copy of your original prompted query, be sure to save it in the database before you convert it to SQL.

To convert a prompted query to an SQL query:

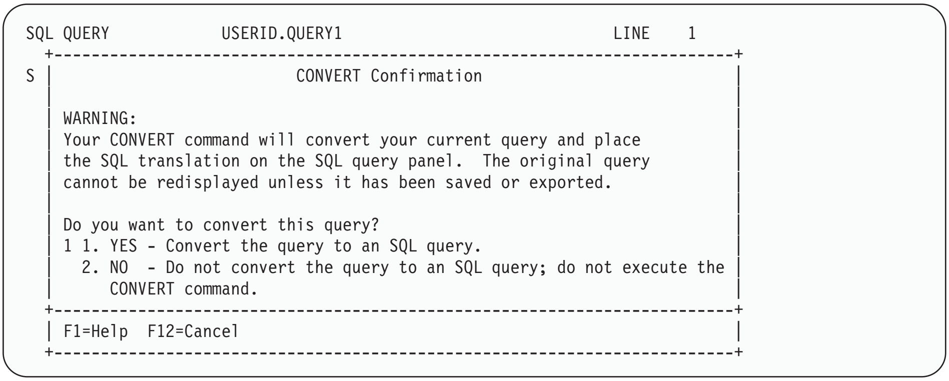
1. If the prompted query is saved in the database, display it by entering on the QMF command line:

   DISPLAY QUERY *queryname*

2. On the QMF command line, enter:

   ```
   CONVERT QUERY
   ```

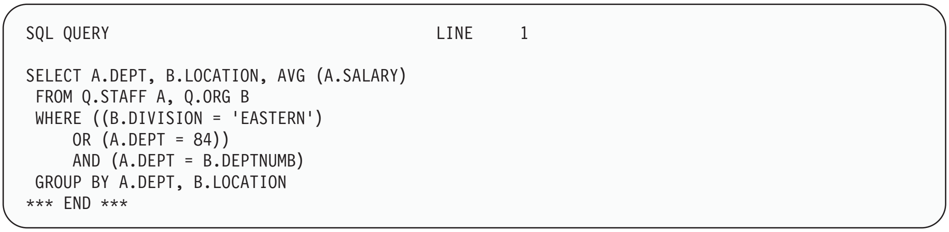
   The Convert Confirmation panel displays, if you specified YES for the Confirm option in your QMF user profile.



```
SQL QUERY            USERID.QUERY1                             LINE    1
  +-------------------------------------------------------------------------+
S |                          CONVERT Confirmation                           |
  |                                                                         |
  | WARNING:                                                                |
  | Your CONVERT command will convert your current query and place          |
  | the SQL translation on the SQL query panel.  The original query         |
  | cannot be redisplayed unless it has been saved or exported.             |
  |                                                                         |
  | Do you want to convert this query?                                      |
  | 1 1. YES - Convert the query to an SQL query.                           |
  |   2. NO  - Do not convert the query to an SQL query; do not execute the |
  |      CONVERT command.                                                   |
  +-------------------------------------------------------------------------+
  | F1=Help  F12=Cancel                                                     |
  +-------------------------------------------------------------------------+
```

Figure 60. The Convert Confirmation panel

3. Press Enter to accept choice 1, **Yes**. The SQL query displays.



```
SQL QUERY                                   LINE     1

SELECT A.DEPT, B.LOCATION, AVG (A.SALARY)
 FROM Q.STAFF A, Q.ORG B
 WHERE ((B.DIVISION = 'EASTERN')
     OR (A.DEPT = 84))
     AND (A.DEPT = B.DEPTNUMB)
 GROUP BY A.DEPT, B.LOCATION
*** END ***
```

Figure 61. QMF displays the SQL query.

   You can change the query by using SQL statements. You can also save the query in the database.

Chapter 5. Viewing the data in the database using SQL statements

In this chapter you will learn how to select and view the data in the database by using SQL statements. When you use SQL statements to select and view data, QMF does not prompt you for information as it does with Prompted Query. However, after you learn the basic rules for writing SQL queries, you might find it faster and simpler.

Entering an object name on more than one line in the SQL query window

The SQL query currently supports the entry of long names within an SQL statement. When an object name extends beyond a single line, delimited identifiers must be used to enclose the object name. A single SQL query line is limited to 79 bytes. The following example shows how to code an SQL statement that contains an object name across multiple lines:

```
SQL QUERY MODIFIED LINE 1

SELECT * FROM
"LOCA4567890123456"."OWNER6789001234567891234567890123456789012345678901234567
89012345678901234567890123456789012345678901234567890123456789012345678"."ONAME
6789012345678901234567890123456789012345678901234567890123456789012345678901234
56789012345678901234567890123456789012345678"
```

You can also write and run SQL queries in a Windows environment by using the DB2 QMF for Windows feature. See Appendix C, “The DB2 QMF High Performance Option,” on page 367 for more information.

When you enter SQL statements on the SQL Query panel, you need to know:

- The name of the table from which you want data
- The column names in the table
- The row conditions you want to specify
- The sequence in which you want the data to appear

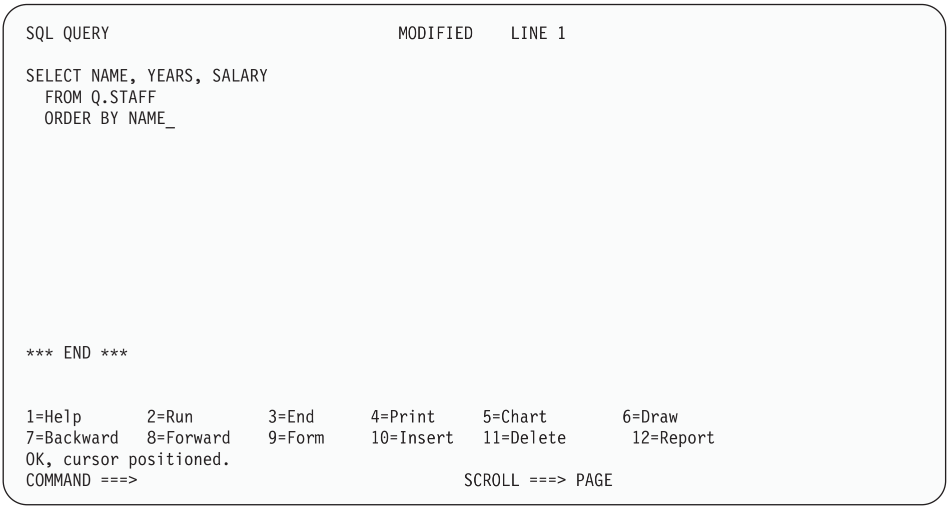
For more information about writing SQL queries, see the SQL reference manuals that come with your database management system.

Format of SQL queries

Many simple SQL queries use the following basic SQL statement:

```
SELECT columnname
 FROM tablename
 WHERE condition
 ORDER BY columnname
```

Figure 62 shows a basic SQL query. This query displays the employee names, years of service, and salaries from the Q.STAFF table.



```
SQL QUERY                              MODIFIED    LINE 1

SELECT NAME, YEARS, SALARY
  FROM Q.STAFF
  ORDER BY NAME_












*** END ***


1=Help       2=Run        3=End      4=Print     5=Chart        6=Draw
7=Backward   8=Forward    9=Form     10=Insert   11=Delete       12=Report
OK, cursor positioned.
COMMAND ===>                                   SCROLL ===> PAGE
```

Figure 62. A basic SQL query

Starting a SQL query

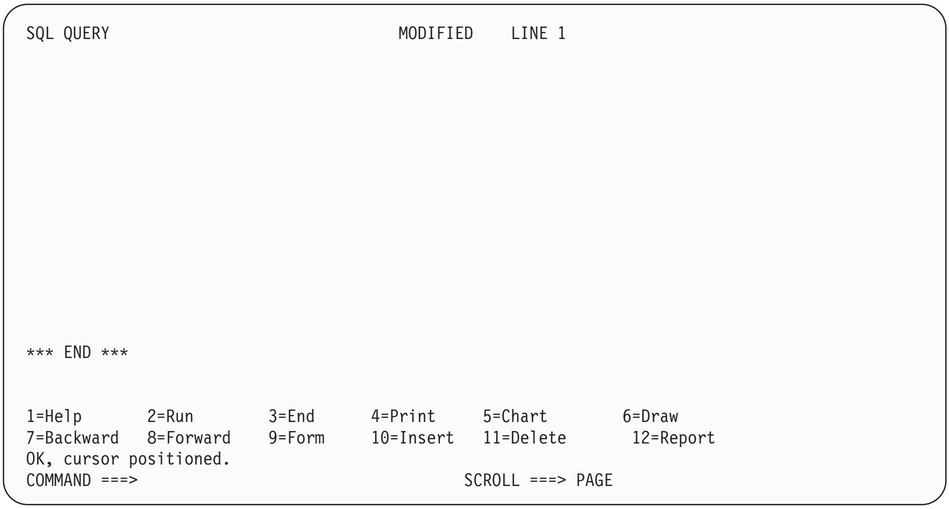
1. Make sure that the value of the Language field in your QMF user profile is SQL. If you need more information about setting up your QMF user profile, see “Setting up and changing your QMF user profile” on page 7.
2. On the QMF command line, enter:

   ```
   RESET QUERY
   ```

   If you do not want to change your QMF profile, you can enter the following command each time you use the RESET command:

   ```
   RESET QUERY (LANG=SQL
   ```

   The SQL query panel displays.



```
SQL QUERY                               MODIFIED    LINE 1














*** END ***


1=Help       2=Run       3=End      4=Print     5=Chart        6=Draw
7=Backward   8=Forward   9=Form     10=Insert   11=Delete       12=Report
OK, cursor positioned.
COMMAND ===>                                     SCROLL ===> PAGE
```

Figure 63. The SQL Query panel

Entering SQL statements and running the query

1. On the SQL Query panel, type the SQL statements you want.
2. To run the query, press the Run function key.

   Or you can:

   Enter `RUN QUERY` on the command line.

   A report displays, showing all the data that you selected.

You can cancel a query that is running. For information on canceling a query, see "Canceling a running query" on page 63.

For more information on running a query, see "Running a query and displaying a report" on page 61 and the topics that follow it.

Selecting columns and tables

The format of SQL queries requires you to select columns before you select tables.

To select columns: Type `SELECT`, followed by the names of the columns in the order that you want them to appear on the report. Use commas to separate the column names.

For example, to select the `DEPTNAME` and `DEPTNUMB` columns, type:

```
SELECT DEPTNAME, DEPTNUMB
```

To select all the columns to view on the report, type:

```
SELECT *
```

To find column names: If you know the table from which you want to select data, but do not know all the column names, you can use the Draw function key on the SQL Query panel.

1. On the QMF command line, type the name of the table whose columns you want to see.

   For example, to find the names of the columns in the Q.ORG table, type Q.ORG.
2. Press the Draw function key.

   QMF displays a query that selects all the columns for the table you specified:

   ```
   SELECT DEPTNUMB, DEPTNAME, MANAGER, DIVISION  -- Q.ORG
        , LOCATION                               -- Q.ORG
   FROM Q.ORG
   ```
3. Leave the query as is, or change it to select specific rows.

To select tables: Type FROM, followed by the name of the table from which you want to select data.

For example, to select the Q.ORG table, type:

```
FROM Q.ORG
```

If you need to see a list of tables, use the LIST TABLES command. For more information on the LIST TABLES command, see “Displaying a list of database objects using the LIST command” on page 30.

Creating a column using expressions

You can create a column for your report by adding, subtracting, multiplying, or dividing the values in two or more columns. Then, you can include the resulting value as a new column with the WHERE keyword.

For example, this statement creates a new column that is the total of each employee’s salary and commission:

```
SELECT NAME, SALARY + COMM
  FROM Q.STAFF
  WHERE SALARY + COMM > 20000
```

QMF displays this report when you run the query:

```
NAME       EXPRESSION 1
---------  ------------
WILLIAMS   20094.15
GRAHAM     21200.30
```

QMF names the calculated column as follows:

- **EXPRESSION 1** if you are using DB2 for VM or VSE
- **COL1** if you are using DB2 for z/OS
- **1** if you are using DB2 for AIX
- **0002** if you are using DB2 for iSeries

It names additional defined columns **EXPRESSION 2, EXPRESSION 3** (or **COL2, COL3**, or **2, 3**) and so on.

If you want to change the column headings, see “Changing column headings” on page 119.

After you define the new column, you can use it just as you would a column you select from a table.

For more information on using arithmetic expressions, see *DB2 QMF Reference*.

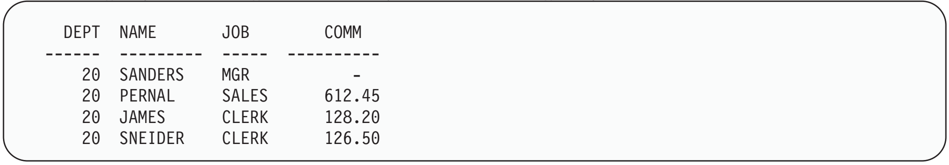
Selecting rows

There are times you will not want to view every row in a table. To select specific rows to view, use the WHERE keyword, followed by a condition. If you do not use the WHERE keyword, all the rows in the table are displayed.

For example, to select only the rows for employees who work in Department 20, type:

```
SELECT DEPT, NAME, JOB, COMM
  FROM Q.STAFF
  WHERE DEPT = 20
```

QMF displays this report when you run the query:



```
   DEPT  NAME       JOB        COMM
 ------  ---------  -----  ----------
     20  SANDERS    MGR            -
     20  PERNAL     SALES      612.45
     20  JAMES      CLERK      128.20
     20  SNEIDER    CLERK      126.50
```

Selecting rows that have no data

To select only rows that have no data, type:

```
WHERE columnname IS NULL
```

For example, to select employees with no commission, type:

```
WHERE COMM IS NULL
```

Selecting rows using specific character values

You can use character values to select the rows you want to view. Make sure that you enclose the data with single quotation marks.

For example:

```
SELECT NAME, JOB
  FROM Q.STAFF
  WHERE NAME = 'SANDERS'
```

Selecting rows using conditions

You can specify any of the following conditions when you select rows:

=	Equal to
>	Greater than
> =	Greater than or equal to
<	Less than
< =	Less than or equal to
¬=	Not equal to
<>	Not equal to

The following query selects employees who earn a commission greater than or equal to $1,000.00.

```
SELECT ID, COMM
  FROM Q.STAFF
  WHERE COMM >= 1000
```

The following query selects employees who earn a commission of at least $170.00, but not more than $220.00.

```
SELECT ID, COMM
  FROM Q.STAFF
  WHERE COMM BETWEEN 170 AND 220
```

For more information about the BETWEEN keyword, see the SQL reference manual for your database management system.

Selecting rows using opposite conditions

You specify the opposite of any condition by typing NOT before it.

If you specify >, <, or =, you must type NOT in front of the entire condition.

For example, type:

```
WHERE NOT YEARS = 10
```

If you specify a NULL, LIKE, IN, or BETWEEN condition, type NOT right before the condition keyword.

For example, type:

```
WHERE YEARS IS NOT NULL
```

```
WHERE YEARS IS NOT NULL
```

The following query selects employees whose salary is under $16,000.00 and over $22,000.00:

```
SELECT ID, NAME, SALARY
  FROM Q.STAFF
  WHERE SALARY NOT BETWEEN 16000 AND 22000
```

The following query selects employees whose salary is under $16,000.00 and who earn less than $500.00 in commissions:

```
SELECT ID, NAME, SALARY, COMM
  FROM Q.STAFF
  WHERE NOT SALARY > 16000 AND NOT COMM > 500
```

Narrowing row selection using selection symbols

To select rows using selection symbols, use the LIKE keyword in a WHERE clause, and the underscore and percent sign as selection symbols.

- Use an underscore (_) to fill in for one character.
- Use the percent sign (%) to fill in for zero or more characters.
- Any other character represents a single occurrence of itself.

For example, this query selects the rows for employees whose names end in SON.

```
SELECT NAME
  FROM Q.STAFF
  WHERE NAME LIKE '%SON'
```

This query selects the rows for employees whose names are five characters long and end in ES.

```
SELECT NAME
  FROM Q.STAFF
  WHERE NAME LIKE '___ES'
```

(The line '___ES' includes three underscores.)

```
NAME
---------
HANES
JAMES
JONES
```

You can use % more than once in an expression.

For example, the following query selects the rows for employees whose names contain an M and then an N. From the Q.STAFF sample table, this query selects MARENGHI, ROTHMAN, and MOLINARE.

```
WHERE NAME LIKE '78N%'
```

You can use the % and _ selection symbols in the same WHERE clause.

For example, the following query selects the rows for employees whose names have R as the second letter. From the Q.STAFF sample table, this query selects FRAYE and GRAHAM.

```
WHERE NAME LIKE '_R%'
```

You can use the NOT keyword with selection symbols to specify rows you do not want to select.

For example, the following query selects the rows for employees whose names do not begin with G.

```
WHERE NAME NOT LIKE 'G%'
```

Narrowing row selection using multiple row conditions

You can create multiple row conditions, and use the AND, OR, or IN keywords to connect the conditions.

Selecting rows If both conditions are true

If you want to select rows that meet both conditions, use the AND keyword to connect them.

The following query displays the ID, NAME, YEARS, and SALARY of employees in the Q.STAFF table that have both 10 years of service and earn more than $20,000.

```
SELECT ID, NAME, YEARS, SALARY
  FROM Q.STAFF
  WHERE YEARS = 10
    AND SALARY > 20000
```

Selecting rows If either condition is true

If you want to select rows that meet either condition, use the OR keyword to connect them.

The following query displays the same columns in the Q.STAFF table, but selects employees who have either 10 years of service or earn more than $20,000.00.

```
SELECT ID, NAME, YEARS, SALARY
  FROM Q.STAFF
  WHERE YEARS = 10
    OR SALARY > 20000
```

Grouping row conditions

You can use AND and OR statements together to connect conditions. Use parentheses to indicate which conditions you want checked first. The conditions inside the parentheses are checked first, and then the conditions outside the parentheses.

If you do not use parentheses, NOT is applied before AND, and AND is applied before OR.

For example, when you run this query:

```
SELECT NAME, ID, DEPT
  FROM Q.STAFF
  WHERE (JOB='SALES' AND COMM > 1200) OR YEARS > 10
```

QMF Displays this report:

```
NAME           ID    DEPT
---------  ------  ------
KOONITZ        90      42
JONES         260      10
GRAHAM        310      66
EDWARDS       340      84
```

When you run the same query with the parentheses moved:

```
SELECT NAME, ID, DEPT
  FROM Q.STAFF
  WHERE JOB='SALES' AND (COMM > 1200 OR YEARS > 10)
```

QMF Displays this report:

```
NAME           ID    DEPT
---------  ------  ------
KOONITZ        90      42
GRAHAM        310      66
EDWARDS       340      84
```

Selecting rows using the IN predicate

You can use one IN statement to replace multiple OR statements.

Both of the following queries select the same rows to view on the report:

```
SELECT ID, NAME, YEARS, SALARY
  FROM Q.STAFF
  WHERE DEPT = 38 OR DEPT = 20 OR DEPT = 42
```

```
SELECT ID, NAME, YEARS, SALARY
  FROM Q.STAFF
  WHERE DEPT IN (38, 20, 42)
```

And

Use NOT with the IN statement to specify rows you do not want to select, as in the following example:

```
WHERE DEPT NOT IN (15, 20, 38)
```

Eliminating duplicate rows

Use the DISTINCT keyword to eliminate duplicate rows from a report.

The following query displays each department in which some employee is a salesperson. Even if a department has more than one salesperson, QMF displays the department number only once on the report.

```
SELECT DISTINCT DEPT
  FROM Q.STAFF
  WHERE JOB = 'SALES'
  ORDER BY DEPT
```

Sorting the rows in a query

To specify the way you want to sort the rows, use the ORDER BY keyword. Follow ORDER BY with the name of the column, or columns, on which you want to sort the rows. QMF sorts the rows in ascending order unless you specify descending order.

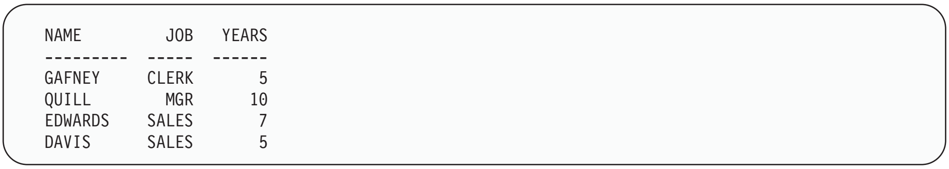
For example, the following query displays the rows in ascending orders by job:

```
SELECT NAME, JOB, YEARS
 FROM Q.STAFF
 WHERE DEPT = 84
 ORDER BY JOB
```

If you sort rows by more than one column, the first column is ordered first, the second column is ordered within the order of the first column, and so on.

This query displays the rows in ascending order by job, and orders the years within job in descending order.

```
SELECT NAME, JOB, YEARS
  FROM Q.STAFF
  WHERE DEPT=84
  ORDER BY JOB, YEARS DESC
```



```
NAME         JOB  YEARS
---------  -----  ------
GAFNEY     CLERK       5
QUILL        MGR      10
EDWARDS    SALES       7
DAVIS      SALES       5
```

Adding or deleting lines in a SQL query

You can add new lines to an SQL query, or delete lines you no longer need. You can change the query before or after you run it.

To add lines:

1. If you have not saved the query in the database, display the query again by entering `SHOW QUERY` on the QMF command line. Or, you can display the query by pressing the Query function key. If you saved the query in the database, enter `DISPLAY QUERY` *queryname*.
2. Move the cursor above the place where you want to add information.
3. Press the Insert function key. QMF displays a blank line.
4. Type the information on the blank line. For this example, add a condition to select only employees from department 38.

```
SELECT NAME, YEARS, SALARY
  FROM Q.STAFF
  WHERE DEPT=38
  ORDER BY NAME
```

To delete a line:

Move the cursor to the line you want to delete, and press the Delete function key.

Displaying data from more than one table

You can include information from more than one table or view by doing one of the following:

- Joining tables or views by a common column
- Merging data from two or more tables or views into a single column
- Creating a subquery to retrieve data from several tables

Joining columns in two or more tables

The SQL statement in Figure 64 on page 80 uses data from the Q.STAFF and Q.ORG tables to select all the clerks in the Eastern division.

If you check the sample tables in Chapter 17, “QMF sample tables,” on page 285, you see that the department numbers are found in both tables, the division name is in the Q.ORG table, and the job title is in the Q.STAFF table. In Q.ORG, the department number is in the DEPTNUMB column, and in Q.STAFF, the department number is in the DEPT column. You will join the tables by these two columns.

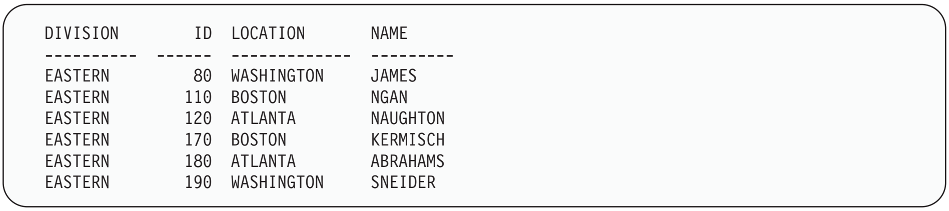
Specify all the columns you want to display on the report in the SELECT clause. Use the FROM clause to Specify the tables you want to join. Specify the columns whose values are equal in the WHERE clause, separated by an equal(=) sign.

```
SELECT DIVISION, ID, LOCATION, NAME
  FROM Q.STAFF, Q.ORG
  WHERE DIVISION = 'EASTERN'
    AND JOB='CLERK'
    AND DEPTNUMB = DEPT
  ORDER BY ID
```

Figure 64. This SQL query joins the Q.STAFF and Q.ORG tables.

The report in Figure 65 displays when you run the query:

If you do not specify a common column when you join two tables, each row in the first table is joined to each row in the second table. The resulting report might contain duplicate data and might be very large.



```
DIVISION        ID  LOCATION       NAME
----------  ------  -------------  ---------
EASTERN         80  WASHINGTON     JAMES
EASTERN        110  BOSTON         NGAN
EASTERN        120  ATLANTA        NAUGHTON
EASTERN        170  BOSTON         KERMISCH
EASTERN        180  ATLANTA        ABRAHAMS
EASTERN        190  WASHINGTON     SNEIDER
```

Figure 65. The report shows the data from both tables.

The columns in the tables you are joining might have the same name. Use one of the following methods to distinguish between columns with the same name:

- Add a qualifier to the column name
- Specify a correlation name to identify a column with a particular table

Distinguishing between column names with qualifiers

You can add a qualifier to identical column names to identify the table from which you selected the column.

For example, to distinguish between the PRODNUM column in the Q.PRODUCTS table, and the PRODNUM column in the Q.PROJECT table, add the following qualifiers to the column names:

- Add Q.PRODUCTS to the PRODNUM column from the PRODUCTS table
- Add Q.PROJECT to the PRODNUM column from the PROJECT table

The SQL statement in Figure 66 on page 81 selects all the product numbers in both the Q.PRODUCTS and Q.PROJECT tables, the project numbers, departments, and product prices.

You only need to specify one of the duplicate column names when you select columns because the two columns are combined in the report. Use a qualifier for duplicate column names everywhere you refer to them in the query.

```
SELECT PROJNO, Q.PRODUCTS.PRODNUM, DEPT, PRODPRICE
 FROM Q.PROJECT, Q.PRODUCTS
 WHERE Q.PRODUCTS.PRODNUM < 100 AND
 Q.PRODUCTS.PRODNUM = Q.PROJECT.PRODNUM
```

Figure 66. This SQL query selects data from two columns with the same name.

Distinguishing between column names with correlation names

Correlation names are used to identify the tables or views from which you selected columns when more than one column has the same name.

For example, to distinguish between the PRODNUM column in the Q.PRODUCTS table and the PRODNUM column in the Q.PROJECTS table, specify a correlation name of P for Q.PROJECT and a correlation name of S for Q.PRODUCTS.

Use the correlation name as a prefix to the column name wherever you refer to that column. The following query shows examples of using correlation names:

```
SELECT PROJNO, S.PRODNUM, DEPT, PRODPRICE
  FROM Q.PROJECT P, Q.PRODUCTS S
  WHERE S.PRODNUM < 100 AND
  S.PRODNUM = P.PRODNUM
```

Merging data from multiple tables into a single column

You can merge data from two or more tables into a single column on a report by using the keyword UNION. First, create two or more queries to select the data you want to merge, then specify the keyword UNION between the queries.

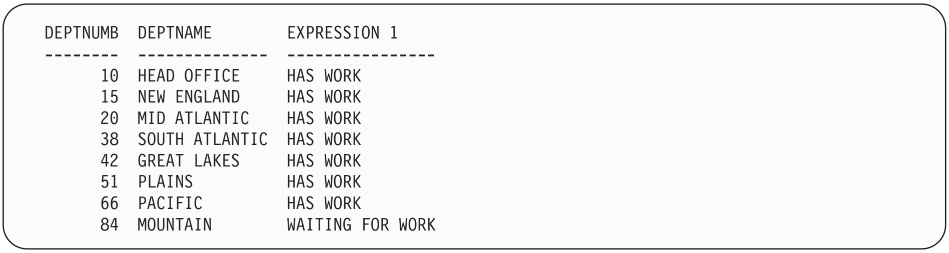
In Figure 67 on page 82, the first query selects the department name and number from the Q.ORG table and creates a new column that displays the words WAITING FOR WORK. The second query selects the department name and number from the Q.PROJECT and Q.ORG tables, and creates a new column that displays the words HAS WORK. The database determines the name of the new column, unless you change it using QMF forms.

Select the same number of columns for each query. Corresponding columns must be the same general data type, and must both either allow null values or not allow null values. If you want to order the columns, specify a column number, because the names of the columns you are merging are probably different. If you want to display duplicate rows on the report, specify UNION ALL instead of UNION.

```
SELECT DEPTNUMB, DEPTNAME, 'WAITING FOR WORK'
    FROM Q.ORG
    WHERE DEPTNUMB NOT IN (SELECT DEPT FROM Q.PROJECT)
UNION
SELECT O.DEPTNUMB, O.DEPTNAME, 'HAS WORK'
    FROM Q.PROJECT P, Q.ORG O
    WHERE P.DEPT = O.DEPTNUMB
ORDER BY 1
```

Figure 67. This SQL query merges data from two columns into one.

QMF displays the following report when you run the query, showing the department names and numbers and their status information on the same report.



```
DEPTNUMB  DEPTNAME        EXPRESSION 1
--------  --------------  ----------------
      10  HEAD OFFICE     HAS WORK
      15  NEW ENGLAND     HAS WORK
      20  MID ATLANTIC    HAS WORK
      38  SOUTH ATLANTIC  HAS WORK
      42  GREAT LAKES     HAS WORK
      51  PLAINS          HAS WORK
      66  PACIFIC         HAS WORK
      84  MOUNTAIN        WAITING FOR WORK
```

Figure 68. The report shows the two new columns merged into one.

You can specify the order in which you want to merge the columns from multiple tables. Specifying order is important when you use UNION and UNION ALL. Use parentheses to indicate which table's columns you want merged first. The conditions inside the parentheses are checked first before the conditions outside the parentheses.

For example, this query produces Report A in Figure 69 on page 83:

```
(SELECT ID, NAME, SALARY
    FROM Q.STAFF
    WHERE SALARY>12000
UNION ALL
SELECT ID, NAME, SALARY
    FROM Q.STAFF
    WHERE DEPT=38)
UNION
SELECT ID, NAME, SALARY
    FROM Q.STAFF
    WHERE JOB='SALES'
```

If you move the parentheses, the same query produces Report B in Figure 69:

```
SELECT ID, NAME, SALARY
    FROM Q.STAFF
    WHERE SALARY>12000
UNION ALL
(SELECT ID, NAME, SALARY
    FROM Q.STAFF
    WHERE DEPT=38
UNION
SELECT ID, NAME, SALARY
    FROM Q.STAFF
    WHERE JOB='SALES')
```

```
          REPORT A                             REPORT B
    ID  NAME           SALARY           ID  NAME           SALARY
------  ---------  ----------       ------  ---------  ----------
    10  SANDERS      18357.50           20  PERNAL       18171.25
    20  PERNAL       18171.25           30  MARENGHI     17506.75
    30  MARENGHI     17506.75           40  O'BRIEN      18006.00
    40  O'BRIEN      18006.00           60  QUIGLEY      16808.30
    50  HANES        20659.80           70  ROTHMAN      16502.83
    60  QUIGLEY      16808.30           90  KOONITZ      18001.75
    70  ROTHMAN      16502.83          120  NAUGHTON     12954.75
    80  JAMES        13504.60          150  WILLIAMS     19456.50
    90  KOONITZ      18001.75          180  ABRAHAMS     12009.75
   100  PLOTZ        18352.80          220  SMITH        17654.50
   110  NGAN         12508.20          280  WILSON       18674.50
   120  NAUGHTON     12954.75          300  DAVIS        15454.50
   140  FRAYE        21150.00          310  GRAHAM       21000.00
   150  WILLIAMS     19456.50          320  GONZALES     16858.20
   160  MOLINARE     22959.20          340  EDWARDS      17844.00
   170  KERMISCH     12258.50           10  SANDERS      18357.50
   180  ABRAHAMS     12009.75           20  PERNAL       18171.25
   190  SNEIDER      14252.75           30  MARENGHI     17506.75
   210  LU           20010.00           40  O'BRIEN      18006.00
   220  SMITH        17654.50           50  HANES        20659.80
   230  LUNDQUIST    13369.80           60  QUIGLEY      16808.30
   240  DANIELS      19260.25           70  ROTHMAN      16502.83
   250  WHEELER      14460.00           80  JAMES        13504.60
   260  JONES        21234.00           90  KOONITZ      18001.75
   270  LEA          18555.50          100  PLOTZ        18352.80
   280  WILSON       18674.50          110  NGAN         12508.20
   290  QUILL        19818.00          120  NAUGHTON     12954.75
   300  DAVIS        15454.50          140  FRAYE        21150.00
   310  GRAHAM       21000.00          150  WILLIAMS     19456.50
   320  GONZALES     16858.20          160  MOLINARE     22959.20
   340  EDWARDS      17844.00          170  KERMISCH     12258.50
   350  GAFNEY       13030.50          180  ABRAHAMS     12009.75
```

Figure 69. The two reports show the differences in merging order.

The first query selects employees whose salaries are greater than $12,000.00 and all employees from Department 38. Then, it eliminates any duplicate entries by selecting only employees who work in sales and are not in Department 38 or making more than $12,000.00 a year.

The second query creates duplicate entries because it first selects employees from Department 38 and employees from outside Department 38 who work in sales. Then, it adds employees whose salaries are more than $12,000.00.

Creating a subquery to retrieve data from more than one table

You can add subqueries to your query to retrieve a value or set of values from one table so you can select data to display from another table. A subquery is a complete query that appears in the WHERE or HAVING clause of another query.

You can specify up to 16 subqueries within a single query, and you can specify subqueries within a subquery. Subqueries run from last to first within the overall query.

Rules for creating a subquery:

- Enclose the subquery in parentheses.
- Specify only one column or expression in a subquery unless you are using IN, ANY, ALL, or EXISTS.
- A subquery cannot contain a BETWEEN or LIKE clause.
- A subquery cannot contain an ORDER BY clause.
- A subquery in an UPDATE query cannot retrieve data from the same table in which data is to be updated.
- A subquery in a DELETE query cannot retrieve data from the same table in which data is to be deleted.

The following query displays the names and IDs of employees who work in Boston. The subquery (in parentheses) finds the department number for the location of BOSTON in the Q.ORG table. Then, the main query selects the names of the employees in that department from the Q.STAFF table.

```
SELECT NAME, ID
  FROM Q.STAFF
  WHERE DEPT=(SELECT DEPTNUMB
              FROM Q.ORG
              WHERE LOCATION='BOSTON')
```

In the next example, the subquery and main query retrieve data from the same table. The subquery calculates the average salary for all the employees in the Q.STAFF table. Then, the main query selects the salespeople whose salaries are equal to or greater than the average salary.

```
SELECT ID, NAME, SALARY
  FROM Q.STAFF
  WHERE JOB = 'SALES' AND
    SALARY >= (SELECT AVG(SALARY)
               FROM Q.STAFF)
```

Retrieving more than one value with a subquery

Usually a subquery selects only one column and returns only one value to the query. However, you can create a subquery that returns a set of values using the ANY or ALL keywords used with the comparison operators =, ¬=, >, >=, <, or <=. In addition, just as you use the IN keyword in place of multiple OR statements in a query, you can also use IN in place of the ANY keyword in a subquery.

The query in Figure 70 selects any employee who works in the Eastern division. The subquery finds the department numbers in the Eastern division, and then the main query selects the employees who work in any of these departments.

Use the ANY keyword for this query; it is likely that the subquery will find more than one department in the Eastern Division. If you use the ALL keyword instead of the ANY keyword, no data is selected because no employee works in all departments of the Eastern division.

```
SELECT NAME, ID
  FROM Q.STAFF
  WHERE DEPT = ANY
      (SELECT DEPTNUMB
       FROM Q.ORG
       WHERE DIVISION='EASTERN')
```

Figure 70. This SQL query contains a subquery using the ANY keyword.

The query in Figure 71 on page 86 selects the department with the highest average salary. The subquery finds the average salary for each department, and then the main query selects the department with the highest average salary.

Use the ALL keyword for this subquery. The department selected by the query must have an average salary greater than or equal to all the average salaries of the other departments.

```
SELECT DEPT, AVG(SALARY)
  FROM Q.STAFF
  GROUP BY DEPT
  HAVING AVG(SALARY) >= ALL
        (SELECT AVG(SALARY)
         FROM Q.STAFF
         GROUP BY DEPT)
```

Figure 71. This SQL query contains a subquery using the ALL keyword.

The query in Figure 72 selects all salespeople (and their salaries) who work for managers earning more than $20,000 a year. The subquery finds the managers who earn more than $20,000 a year, and the main query selects the salespeople who work for those managers.

Use the IN keyword for this subquery because you need to find values from more than one department.

```
SELECT ID, NAME, SALARY
  FROM Q.STAFF
  WHERE JOB = 'SALES'
    AND DEPT IN
       (SELECT DISTINCT DEPT
        FROM Q.STAFF
        WHERE JOB = 'MGR'
          AND SALARY > 20000)
```

Figure 72. This SQL query contains a subquery using the IN keyword.

Checking for rows that satisfy a condition

In the previous examples, you learned how to use a subquery to return a value to the query. You can also use a subquery to check for rows that satisfy a certain row condition using a WHERE EXISTS clause.

The query in Figure 73 on page 87 selects employees from the Q.STAFF table who have a salary of less than $14,000, and who work in a department where at least one other employee with the same job earns a salary greater than $14,000. The subquery checks for other employees in the department with the same job, but who earn a salary greater than $14,000.

```
SELECT NAME, DEPT, JOB, SALARY
  FROM Q.STAFF S
  WHERE S.SALARY < 14000 AND
        EXISTS (SELECT * FROM Q.STAFF
        WHERE S.DEPT=DEPT AND SALARY >14000
        AND S.JOB=JOB)
  ORDER BY S.DEPT
```

Figure 73. This subquery checks for rows that satisfy a condition.

You can specify NOT IN in a subquery to select information from one table when corresponding information does not exist in the other table.

Specifying a correlation name in a subquery

You can specify a correlation name in a subquery to evaluate every row that is selected by the query for the condition that is specified in the subquery.

The query in Figure 74 selects the department, name, and salary of the employees who have the highest salary in their departments. The subquery calculates the maximum salary for each department that is selected by the main query. The correlation name, Y, compares each row that is selected by the query to the maximum salary that is calculated for the department in the subquery.

```
SELECT DEPT, NAME, SALARY
  FROM Q.STAFF Y
  WHERE SALARY = (SELECT MAX (SALARY)
                  FROM Q.STAFF
                  WHERE DEPT = Y.DEPT)
```

Figure 74. This subquery specifies a correlation name.

Writing correlation names

The correlation name must be unique within the query and must appear in two places:

- In the FROM clause of the main query
- In the WHERE clause of the subquery

Names used for correlation names in queries are arbitrary. Choose any name up to 128 characters long. It must not duplicate other words in the query or any SQL reserved word.

If you use correlation names and several table names, separate the items in the list with commas. For example:

```
FROM Q.ORG XXX, Q.APPLICANT, Q.STAFF YYY
```

You might need correlation names even without a subquery, as in the following example.

Example 1

This query lists employees whose salaries are greater than their managers' salaries, and selects twice from Q.STAFF.

```
SELECT X.ID, X.NAME, X.SALARY, Y.SALARY
  FROM Q.STAFF X, Q.STAFF Y
  WHERE X.DEPT = Y.DEPT
    AND Y.JOB = 'MGR'
    AND X.SALARY > Y.SALARY
```

Selecting twice from Q.STAFF is necessary so that the DEPT of each person can be matched with every other DEPT in the table to discover which employees work for which managers.

The WHERE condition selects employees from both tables who are in the same department, and selects employees in the Y version of the table who are managers. Then, it selects employees whose salaries are greater than their managers' salaries.

Example 2

This query lists employees who earn the largest commission in each location.

```
SELECT LOCATION, ID, NAME, COMM
  FROM Q.STAFF, Q.ORG ZZZ
  WHERE DEPT=DEPTNUMB
    AND COMM = (SELECT MAX(COMM)
               FROM Q.STAFF, Q.ORG
               WHERE DEPT=DEPTNUMB
                 AND LOCATION = ZZZ.LOCATION)
```

In this query, the subquery first finds the largest commission within a given location. The main part of the query finds who within that location earned that commission. Because the query names two tables, it includes a correlation name that indicates which table contains the LOCATION column.

Example 3

This query lists employees whose salaries are greater than their managers' salaries. Another version of this query that is shown in Example 1 selects twice from Q.STAFF without using a subquery. A query that joins two or more tables will probably run faster than a similar query that uses a subquery.

```
SELECT ID, NAME, SALARY
  FROM Q.STAFF EMP
  WHERE SALARY > (SELECT SALARY
    FROM Q.STAFF
WHERE ID = (SELECT MANAGER
              FROM Q.ORG
              WHERE DEPTNUMB = EMP.DEPT))
```

Working with a set of values using SQL column functions

A column function produces a single value for a group of rows. For example, if an SQL SELECT clause asks for the value SUM(SALARY), QMF returns only one value, the sum. The following query shows the use of the column function SUM:

```
SELECT SUM(SALARY)
  FROM Q.STAFF
  WHERE DEPT = 38
```

QMF returns this report:

```
              COL1
------------------
          77285.55
```

The report examples in this chapter and the next chapter appear if your current location is DB2. If your current location is DB2 Server for VSE or VM, your reports might look different.

The column functions are:

AVG Finds the average of the values in a particular column, or a set of values that are derived from one or more columns- the column or expression that is summarized must contain numeric data

MAX Finds the maximum value in a particular column, or a set of values that are derived from one or more columns- MAX applies to all data types

MIN Finds the minimum value in a particular column, or a set of values that are derived from one or more columns- MIN applies to all data types

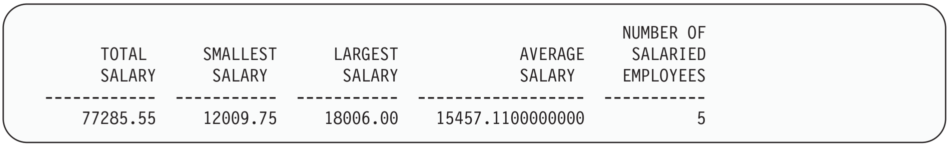
SUM Finds the sum of the values in a particular column, or a set of values that are derived from one or more columns- the column or expression that is added must contain numeric data

COUNT
Finds the number of rows that satisfy the search condition, or finds the number of distinct values in a particular column

The SELECT clause of the SQL statement in Figure 75 uses the five column functions. The SQL statement produces the report that is shown in Figure 76.

```
SELECT SUM(SALARY), MIN(SALARY), MAX(SALARY),
  AVG(SALARY), COUNT(*)
  FROM Q.STAFF
  WHERE DEPT = 38
```

Figure 75. This SQL query uses the QMF column functions.



```
                                                                NUMBER OF
          TOTAL      SMALLEST      LARGEST            AVERAGE    SALARIED
         SALARY        SALARY       SALARY             SALARY   EMPLOYEES
   ------------  -----------  -----------  ------------------  -----------
       77285.55      12009.75     18006.00   15457.1100000000            5
```

Figure 76. The report demonstrates the results of QMF column functions.

In this case, we changed the column headings on the form panel to make them more descriptive.

If you use column functions in an SQL statement where there is no GROUP BY clause, every occurrence of a column name must have a column function so the query can return a single row.

Working with single data values using SQL scalar functions

You can use scalar functions to:

- Convert a value from one data type to another
- Process date/time values
- Manipulate parts of character or graphic strings
- Avoid null values

For more information on scalar functions, see the SQL reference manual for your database management system.

Scalar functions produce a single scalar value for each row that satisfies the search condition in your query. For example, if you replace SUM in the SELECT clause of the SQL statement in Figure 75 with the scalar function HEX, five rows are returned—one hexadecimal value for each row satisfying the search condition:

```
SELECT HEX(SALARY)
  FROM Q.STAFF
  WHERE DEPT = 38
```

```
SALARY
--------
1750675C
1800600C
1680830C
1295475C
1200975C
```

Conversion functions, date/time functions, and string functions are subsets within the set of scalar functions.

Converting a value from one data type to another

The scalar functions DECIMAL, DIGITS, FLOAT, HEX, INTEGER, and VARGRAPHIC allow you to convert a value from one data type to another.

The DECIMAL function returns a decimal representation of a number.

- A numeric expression is an expression that returns a value of any numeric data type.
- A precision integer is an integer constant with a value in the range of 1 to 31.
- A scale integer is an integer constant in the range of 0 to the precision-integer value.

The DIGITS function returns values without a decimal point.

The FLOAT function returns a floating-point representation of a number.

The HEX function uses the hexadecimal numeration system.

The INTEGER function returns an integer representation of a number.

The VARGRAPHIC function converts a mixed single-byte and double-byte character string to a pure double-byte character string. VARGRAPHIC returns a varying length graphic string (data type VARGRAPHIC) result.

The first or only argument of each of these functions is an expression that gives the value to convert.

For example, when you run this SQL statement:

```
SELECT SALARY,           --SALARY
DECIMAL(SALARY,9,3),     --COL1
DIGITS(SALARY),          --COL2
FLOAT(SALARY),           --COL3
HEX(NAME),               --COL4
VARGRAPHIC(JOB)          --COL5
  FROM Q.STAFF
  WHERE DEPT = 10
```

QMF produces this report:

```
    SALARY          COL1  COL2          COL3  COL4                COL5
----------  ------------  -------  ----------  ------------------  ----------
  22959.20     22959.200  2295920   2.296E+04  D4D6D3C9D5C1D9C5    -M-G-R
  20010.00     20010.000  2001000   2.001E+04  D3E4                -M-G-R
  19260.25     19260.250  1926025   1.926E+04  C4C1D5C9C5D3E2      -M-G-R
  21234.00     21234.000  2123400   2.123E+04  D1D6D5C5E2          -M-G-R
```

Formatting dates and times

The date, time, and timestamp scalar functions change the data type of their arguments to the associated date/time data type.

The DATE function returns a date from a value. The argument must be a timestamp, a date, or a string representation of a date.

In the following SQL statement, the argument for DATE is a timestamp:

```
SELECT PROJNO, DATE(TIMESTAMP)
  FROM Q.PROJECT
  WHERE PROJNO = '1401'
```

The query produces this report:

```
PROJNO  DATE
------  ----------
1401    1994-12-18
```

The TIME function returns a time from a value. The argument must be a time, a timestamp, or a string representation of a time. When you run the following SQL statement:

```
SELECT PRODNUM, TIME(TIMESTAMP)
  FROM Q.PROJECT
  WHERE YEAR(STARTD) = 1996
```

QMF produces this report, where TIME shows the time portions of three timestamps in the Q.PROJECT table:

```
PRODNUM  TIME
-------  --------
     10  10.14.44
     50  10.15.01
    150  10.22.23
```

The TIMESTAMP function returns a timestamp from a value or a pair of values. If only one argument is specified, it must be a timestamp, a string representation of a timestamp, a character string of length 8, or a character string of length 14. If the value is a character string of length 14, it must be in the form *yyyymmddhhmmss*, where *yyyy* is the year, *mm* is the month, *dd* is the day, *hh* is the hour, *mm* is the minute, and *ss* is the second.

If a second optional argument is specified, it must be a time or a string representation of a time, and the first argument must be a date or a string representation of a date. For example, for this statement:

```
TIMESTAMP (CURRENT DATE, '10.00.00')
```

QMF produces a timestamp that represents 10 a.m. today.

The CHAR function returns a string representation of a date/time value. CHAR changes the value of its argument (a date or time value) to the CHAR data type. The result of CHAR is a fixed-length character string representation of a date/time value in the format that is specified by its optional second argument. If the first argument is a date or a time, the second argument must be USA, ISO, JIS, EUR, or LOCAL. LOCAL refers to an installation default format. If you omit the second argument, the date or time format is ISO.

When you run the following SQL statement, with a second argument of USA for CHAR:

```
SELECT TEMPID, CHAR(INTDATE, USA)
  FROM Q.INTERVIEW
  WHERE MANAGER = 140
```

QMF produces this report:

```
        INTERVIEW
TEMPID     DATE
------  ----------
   420  04/07/1990
   490  09/30/1990
```

Table 6 and Table 7 show examples of DATE and TIME formats in USA, ISO, JIS, and EUR. In these tables, USA refers to United States of America format, ISO refers to International Standards Organization format, JIS refers to Japanese Industrial Standard format, and EUR refers to European format.

Table 6. DATE formats

Date format	Edit code	Example
USA	TDMx	12/15/1998
ISO, JIS	TDYx	1998-12-15
EUR	TDDx	15.12.1998

Table 7. TIME formats

Time format	Edit code	Example
USA	TTUx	01:25 PM
ISO, EUR	TTSx	13.25.10
JIS	TTSx	13:25:10

Allowing the database requestor to control date and time format

The default edit codes for formatting date (TD) and time (TT) allow the database requestor to control how these values are displayed. Whatever format is in effect at the database requestor is used by QMF. For example, if you are using the TD edit code in QMF and the database requestor specifies USA format for the DATE and TIME fields then the date is displayed as MM/DD/YYYY (TDMx). If the database requestor specifies Japanese Industrial Standard format for the DATE and TIME fields then the date is displayed as YYYY-MM-DD (TDYx).

Isolating the day, month, or year portion of a date

The DAY function returns the day part of a value. The argument must be a date, timestamp, or a decimal number that is interpreted as a duration of years, months, or days. (For a complete description of durations, see "Using durations to represent date/time intervals" on page 101.) The following SQL statement produces a report showing on which day of the month an interview occurs:

```
SELECT TEMPID, DAY(INTDATE)
  FROM Q.INTERVIEW
  WHERE MANAGER = 270
```

This report shows that the interviews take place on the fifth day of the month:

```
            DAY OF
 TEMPID      MONTH
 ------  ---------
    400          5
    470          5
```

The MONTH function returns the month part of a value. The argument must be a date, timestamp, or a decimal number that is interpreted as a duration of years, months, or days. The following SQL statement produces a report showing on which month of the year an interview occurred:

```
SELECT MANAGER, DISP, MONTH(INTDATE)
  FROM Q.INTERVIEW
```

This report shows the months in which certain managers interviewed prospective employees and the disposition of each interview.

```
MANAGER  DISP         MONTH
-------  ------  -----------
    270  NOHIRE           2
     10  HIRE             2
    140  HIRE             4
    290  NOHIRE           4
    160  HIRE             3
     50  HIRE             9
    100  HIRE            10
    270  HIRE             2
    160  NOHIRE           3
    140  NOHIRE           9
```

The YEAR function returns the year part of a value. YEAR works like DAY and MONTH. The argument must be a date, timestamp, or a decimal number that is interpreted as a duration of years, months, or days. When you run the following SQL statement:

```
SELECT PROJNO, YEAR(ENDD)
  FROM Q.PROJECT
  WHERE PRODNUM = 190
```

QMF produces this report:

```
            YEAR OF
PROJNO   COMPLETION
------  -----------
1404           1999
1410           2000
```

This report shows the year of project end dates for a given product. It disregards the day and month.

Isolating the hour, minute, second, or microsecond portion of a time

The HOUR function returns the hour part of a value. The argument must be a time, timestamp, or a decimal number that is interpreted as a time. When you run the following SQL statement:

```
SELECT TEMPID, ENDTIME
  FROM Q.INTERVIEW
  WHERE TEMPID = 400
```

QMF produces this report:

```
TEMPID  ENDTIME
------  --------
   400  15.12.00
```

ENDTIME shows hours, minutes, and seconds. For example, when you run this SQL statement:

```
SELECT TEMPID, HOUR(ENDTIME)
  FROM Q.INTERVIEW
  WHERE TEMPID = 400
```

QMF produces this report, which shows only the hour portion of ENDTIME:

```
              ENDING
TEMPID         HOUR
------  -----------
   400           15
```

The MINUTE function returns the minute part of a value. The argument must be a time, timestamp, or a decimal number that is interpreted as a duration of hours, minutes, or seconds.

When you run the following SQL statement:

```
SELECT TEMPID, MINUTE(ENDTIME)
  FROM Q.INTERVIEW
  WHERE TEMPID = 400
```

QMF produces this report:

```
              ENDING
TEMPID        MINUTE
------  -----------
   400           12
```

The SECOND function returns the seconds part of a value. The argument must be a time, timestamp, or a decimal number that is interpreted as a duration of hours, minutes, or seconds. For example, when you run the following SQL statement:

```
SELECT TEMPID, SECOND(ENDTIME)
  FROM Q.INTERVIEW
  WHERE TEMPID = 400
```

QMF produces this report:

```
              ENDING
TEMPID        SECOND
------  -----------
   400            0
```

The MICROSECOND function returns the microsecond part of a value. The argument can be only a timestamp. For example, when you run the following SQL statement:

```
SELECT PROJNO, MICROSECOND(TIMESTAMP)
  FROM Q.PROJECT
  WHERE PROJNO = '1409'
```

QMF produces this report:

```
PROJNO        COL1
------  -----------
1409        149572
```

Finding the length of a value

The LENGTH function returns the length of a value. The length of a character string is

- The number of bytes for a graphic string
- The number of DBCS characters for a numeric value
- The number of bytes used to represent the value
- The number of bytes in the internal representation of the value for a date/time value

The following SQL statement shows the length of a timestamp, which is 10. When you run this statement:

```
SELECT TIMESTAMP, LENGTH(TIMESTAMP)
  FROM Q.PROJECT
  WHERE DEPT = 51
```

QMF produces this report:

```
TIMESTAMP                        LENGTH
--------------------------  -----------
1994-12-18-10.22.23.000001           10
1996-03-13-12.22.14.201966           10
```

Displaying parts of a value

The SUBSTR function returns a substring of a string. The format of SUBSTR is:

```
SUBSTR(M,N,L)
```

Where:

- M represents a character string or a graphics string to manipulate
- N represents the position of the first character of the desired substring
- L represents the length of the substring to select

The following SQL statement selects a column that contains a last name and a column that contains the first initial of the first name. When you run this statement:

```
SELECT LASTNAME, SUBSTR(FIRSTNAME,1,1)
  FROM Q.INTERVIEW
  WHERE MANAGER = 140
```

QMF produces this report:

```
LASTNAME   INIT
---------  ----
MONTEZ     R
GASPARD    P
```

Replacing null values in the report with other values

The VALUE function is the substitution of a non-null value (specified in the second argument) for each null value found in the column designated by the first argument. You must specify two or more arguments, and the data types of the arguments must be comparable. The following SQL statement selects a column that contains null values. When you run this statement:

```
SELECT COMM
  FROM Q.STAFF
  WHERE YEARS = 5
```

QMF produces the following report:

```
      COMM
----------
         -
    206.60
         -
    806.10
    188.00
```

When you use VALUE with a second argument of 0 in the SELECT clause of an SQL statement, the null values are replaced with 0.00 because the data type is DECIMAL. For example, when you run this statement:

```
SELECT VALUE(COMM, 0)
  FROM Q.STAFF
  WHERE YEARS = 5
```

QMF produces this report:

```
      COMMISSION
----------------
            0.00
          206.60
            0.00
          806.10
          188.00
```

Nesting SQL functions

You can nest built-in column and scalar functions within other functions in the following ways:

- Nest scalar functions within other scalar functions
- Nest scalar functions within column functions

- Nest column functions within scalar functions

You cannot nest column functions within other column functions.

Nesting scalar functions within scalar functions

Suppose you want to know the month and day of interview for all applicants interviewed by manager 140, and you want the result in USA format. When you run this query:

```
SELECT SUBSTR((CHAR(INTDATE, USA)),1,5)
  FROM Q.INTERVIEW
  WHERE MANAGER = 140
```

QMF produces this report:

```
DATE
----------
04/07
09/30
```

Nesting scalar functions within column functions

If a column function's argument is a scalar function, the scalar function must include a reference to a column. For example, if you want to know the last year that any project will begin and the last year that any project will complete, you can run this query:

```
SELECT MAX(YEAR(STARTD)), MAX(YEAR(ENDD))
  FROM Q.PROJECT
```

QMF produces this report:

```
     LATEST        LATEST
      START      COMPLETION
   -----------  -----------
          1999         2000
```

Nesting column functions within scalar functions

Suppose that you want to know the year in which the last project in department 20 will be initiated. If you run this query:

```
SELECT YEAR(MAX(STARTD))
  FROM Q.PROJECT
  WHERE DEPT = 20
```

QMF produces this report:

```
    LAST
  PROJECT
   START
----------
      1997
```

Adding and subtracting dates and times

Addition and subtraction are the only arithmetic operators that you can apply to date/time values. You can increment or decrement a date, time, or timestamp by a duration. You can subtract a date from a date, or a time from a time. You cannot subtract a timestamp from a timestamp.

Rules for date/time addition

If a date/time value is the operand of an addition, the other operand must be a duration. The operands of date/time addition must be as follows:

If one operand is a:	**The other operand must be:**
Date	A duration of years, months, or days
Time	A duration of hours, minutes, or seconds
Timestamp	Any valid duration

Rules for date/time subtraction

Subtracting two date/time values is different from subtracting a duration from a date/time value. The operands of date/time subtraction must be as follows:

If the first operand is a:	**The second operand must be:**
Date	A date, string representation of a date, or a duration of years, months, or days
Time	A time, string representation of a time, or duration of hours, minutes, or seconds
Timestamp	A duration. A timestamp can be only the first operand of subtraction.

If the second operand is a:	**The first operand must be:**
Date	A date or string representation of a date
Time	A time or string representation of a time

Because character strings cannot be subtracted, a string representation of a date or time value cannot be subtracted from another string representation of a date or time value. For example, the following expression is not valid:

```
'1998-01-01' - '1997-01-01'
```

However, if you convert one of the strings to a date or time, the expression is valid. For example, the following expression is valid:

```
DATE('1998-01-01') - '1997-01-01'
```

Finding the number of days between two dates

The DAYS function calculates the number of days between one date and another. You can do this with an equation such as this one:

```
DAYS (future date) - DAYS (&DATE)
```

&DATE supplies the current date.

The DAYS function returns an integer representation of a date. The result of DAYS is the number of days since December 31, 0000. (There is no year 0000. This convention ensures that all days in the range of years 0001 to 9999 are included.) The argument can be a date, a timestamp, or a string representation of a date. For example, if you run this statement:

```
DAYS('0002-01-03')
```

The result is 368 days.

The DAYS function allows you to be more precise in date/time arithmetic. See *DB2 QMF Reference* for a more detailed explanation.

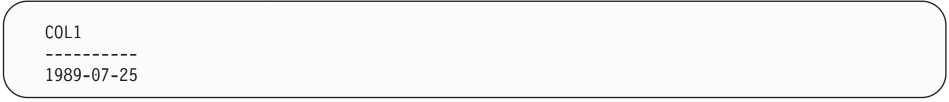
Accounting for months with different numbers of days

Because of inconsistencies in the number of days in the months of the year, adding a month to a given date does not always result in the same day of the next month. The result of adding one month to January 31 cannot be February 31. Adding a month to a given date results in the same day of the next month when such a day exists. If it does not exist, adding a month to a given date results in the last day of the next month. For example, if you add one month to January 31, the result is February 28 (or February 29 in a leap year).

Use days to avoid inconsistencies in date arithmetic caused by months. For example, to increment a date by the difference between two dates, you can use an SQL statement like this:

```
SELECT DATE(DAYS('1988-01-05') + DAYS(ENDD) - DAYS(STARTD))
  FROM Q.PROJECT
  WHERE PROJNO = '1408'
```

QMF produces this report:



```
COL1
----------
1989-07-25
```

Using durations to represent date/time intervals

A duration is a number that represents an interval of time. The number can be a constant, a column name, a function, or an expression.

A duration represents any number of years, months, days, hours, minutes, seconds, or microseconds. The unit is expressed by a keyword that follows the number. In the expression STARTD+25 YEARS, the duration is 25 YEARS.

You can use a duration only in an expression that involves a date or time value. For example, STARTD+25 YEARS+1 MONTH is a valid expression. (STARTD is a column in Q.PROJECT that gives the start date of a project.) STARTD+(25 YEARS+1 MONTH) is not a valid expression, because (25 YEARS+1 MONTH) does not include a date or time value within the parentheses.

YEAR(ENDD - STARTD) < 3 YEARS is not valid because you cannot use the duration, 3 YEARS, as an operand of comparison. A valid way of coding this is YEAR(ENDD - STARTD) < 3.

Subtracting one date from another date results in a duration that is expressed by the number of years, months, and days. Subtracting one time from another time results in a duration that is expressed in the number of hours, minutes, and seconds. See “Subtracting dates” on page 104 and “Subtracting times” on page 106 for the exact format of these results.

Incrementing and decrementing dates by durations

Suppose that you want to know what the start date for project 1404 would be if you delayed it one year. You would increment the current start date (1991-01-04) by using a duration of 1 year. For example, when you run this SQL statement:

```
SELECT STARTD + 1 YEAR
  FROM Q.PROJECT
  WHERE PROJNO = '1404'
```

QMF produces this report:

```
COL1
----------
1998-01-04
```

The month of the result is the same as the month of the date you are incrementing. The day of the result is the same as that of the date incremented, unless the result is February 29 of a year that is not a leap year. In that case, the day is February 28.

If you want to know what the end date of project 1404 (currently slated for 1993-06-30) would be if you finish the project two months ahead of schedule, run this SQL statement using the duration of 2 months:

```
SELECT ENDD - 2 MONTHS
  FROM Q.PROJECT
  WHERE PROJNO = '1404'
```

QMF produces this report:

```
COL1
----------
1999-04-30
```

QMF counts only months (calendar pages) and years (if necessary). The day of the result is the same as the day of the date you are decrementing, unless the result would be an date that is not valid. In that case, the day part of the result is the last day of the month.

To find out what the start date of project 1407 would be if the project is started 30 days early, run the following SQL statement using the duration of 30 days:

```
SELECT STARTD - 30 DAYS
  FROM Q.PROJECT
  WHERE PROJNO = '1407'
```

QMF produces this report:

```
COL1
----------
1997-11-12
```

Suppose that you want to know what the end date for project 1407 would be if you delay the project by 2 years and 11 months. For example, if you run this SQL statement:

```
SELECT ENDD + 2 YEARS + 11 MONTHS
  FROM Q.PROJECT
  WHERE PROJNO = '1407'
```

QMF produces this report:

```
COL1
----------
2003-05-15
```

Suppose, rather than increment by the two years and eleven months duration used in the preceding example, you want to decrement by the same duration. You want to know the project end date if the project is finished 2 years and 11 months ahead of schedule. For example, if you run this SQL statement:

```
SELECT ENDD - 2 YEARS - 11 MONTHS
  FROM Q.PROJECT
  WHERE PROJNO = '1407'
```

QMF produces this report:

```
COL1
----------
1997-07-15
```

Subtracting dates

The result of date subtraction is a duration that is expressed in years, months, and days between the two dates. A negative result is possible with date subtraction.

Suppose that you want to know how many years, months, and days project 1407 will take to complete. If you run the following SQL statement:

```
SELECT ENDD - STARTD
  FROM Q.PROJECT
  WHERE PROJNO = '1407'
```

QMF produces this report:

```
       COL1
-----------
      20603
```

The result is a numeric representation of the duration: 2 years, 6 months, and 3 days. You can treat this result like any other numeric value. The duration format for dates is *yyyymmdd*, where *yyyy* represents years, *mm* represents months, and *dd* represents days. Leading zeros are always truncated in the results.

To find out the number of weeks project 1405 would take to complete, run the following SQL statement:

```
SELECT (DAYS(ENDD) - DAYS(STARTD))/7
  FROM Q.PROJECT
  WHERE PROJNO = '1405'
```

QMF produces this report:

```
       COL1
-----------
        130
```

The result is a duration of 130 weeks.

Suppose that you would like to know, in terms of years, how many years it takes to complete project 1403. If you run this SQL statement:

```
SELECT (DAYS(ENDD)-DAYS(STARTD))/365.24
  FROM Q.PROJECT
  WHERE PROJNO='1403'
```

QMF produces this report:

```
              COL1
------------------
              3.31
```

Making durations easier to read

Suppose that you run the following SQL statement:

```
SELECT ENDD-STARTD
  FROM Q.PROJECT
  WHERE PROJNO='1403'
```

QMF produces this report:

```
       COL1
-----------
      30327
```

The result of this date subtraction is a duration of 3 years, 3 months, and 27 days.

To get results that are easier to read in a report, run the following SQL statement:

```
SELECT YEAR(ENDD - STARTD), MONTH(ENDD - STARTD), DAY(ENDD - STARTD)
  FROM Q.PROJECT
  WHERE PROJNO='1403'
```

QMF produces a report like this:

```
      YEARS       MONTHS         DAYS
-----------  -----------  -----------
          3            3           27
```

QMF changes the column headings on the QMF form to make the report more meaningful.

Incrementing and decrementing times by durations

Adding a duration to a time or subtracting a duration from a time results in a time. The next example increments a time by a duration. To find the start time for an interview if the interview starts 2 hours, 30 minutes, and 45 seconds late, use the following SQL statement:

```
SELECT STARTTIME + 2 HOURS + 30 MINUTES + 45 SECONDS
  FROM Q.INTERVIEW
  WHERE TEMPID = 400
```

QMF produces this report:

```
COL1
--------
15.30.45
```

Adding 24 hours to the time 00.00.00 results in 24.00.00. However, adding 24 hours to any other time results in the same time as the time you are incrementing.

The next example decrements a time by a duration. To find out what time an interview would end if it ended 1 hour, 20 minutes, and 20 seconds early, use the following SQL statement:

```
SELECT ENDTIME - 1 HOUR - 20 MINUTES - 20 SECONDS
  FROM Q.INTERVIEW
  WHERE TEMPID = 410
```

QMF produces this report:

```
COL1
--------
14.57.40
```

Subtracting times

If you subtract two times, the result is a duration that represents the number of hours, minutes, and seconds between the two times. A negative result is possible when subtracting two times.

If you want to know how much time an interview for a person with temporary ID 410 took, use this SQL statement:

```
SELECT ENDTIME - STARTTIME
  FROM Q.INTERVIEW
  WHERE TEMPID = 410
```

QMF produces this report:

```
     COL1
---------
    11800
```

The result is a numeric representation of the duration: 1 hour, 18 minutes, and 0 seconds. You can treat this result like any other numeric value. The duration format for time is *hhmmss*, where *hh* represents hours, *mm* represents minutes, and *ss* represents seconds. QMF always removes leading zeros from the result.

Incrementing and decrementing timestamps by durations

The result of adding a duration to a timestamp or subtracting a duration from a timestamp is a timestamp. The following example increments a timestamp by a duration of 30 microseconds:

```
SELECT TIMESTAMP + 30 MICROSECONDS
  FROM Q.PROJECT
  WHERE PROJNO = '1409'
```

QMF produces this report:

```
COL1
--------------------------
1996-03-13-09.12.57.149602
```

Suppose that you want to know what the timestamp will be for project 1409 if you add a duration of 2 years, 1 month, and 2 hours to the project's existing timestamp. Use the following SQL statement:

```
SELECT TIMESTAMP + 2 YEARS + 1 MONTH + 2 HOURS
  FROM Q.PROJECT
  WHERE PROJNO = '1409'
```

QMF produces this report:

```
COL1
--------------------------
1998-04-13-11.12.57.149572
```

Using the concatenation operator

Use the concatenation operator (||) to join two values of an expression into a single string. In some non-English, single-byte character sets, the || can display as !! (exclamation marks) or other special characters.

Rules for concatenation

The following rules apply to using the concatenation operator:

- The operands you concatenate must all be either character strings or graphic strings.
- The length of the result is the sum of the lengths of the operands.
- The data type of the result is:
  - VARCHAR when all operands are CHAR or when one or more operands is VARCHAR.
  - VARGRAPHIC when all operands are GRAPHIC or when one or more operands are VARGRAPHIC.
- If either operand is null, the result is the null value. (To avoid null values, use the VALUE scalar function that is described at page 98.)

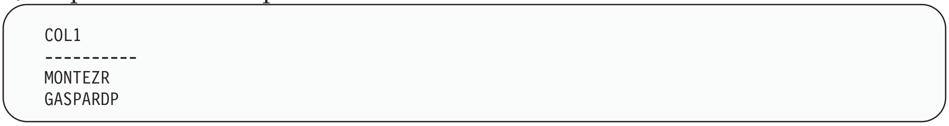
- You cannot specify concatenation in a LIKE clause.
- You cannot specify concatenation in the SET clause of an UPDATE query.

Examples using concatenation

In the SELECT clause of the following SQL statement, the concatenation operator is used with the SUBSTR scalar function to join the first character of FIRSTNAME with LASTNAME. When you run this query:

```
SELECT LASTNAME||SUBSTR(FIRSTNAME,1,1)
  FROM Q.INTERVIEW
  WHERE MANAGER = 140
```

QMF produces this report:



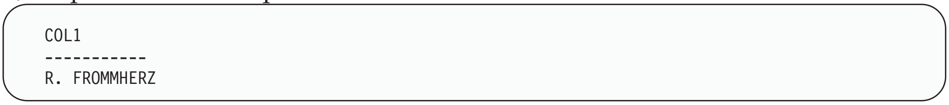
```
COL1
----------
MONTEZR
GASPARDP
```

There is no space between the last name and the initial, because none was provided for when concatenation was done. This is true because the data types for the columns FIRSTNAME and LASTNAME are VARCHAR.

The next example concatenates a substring of the first name with a period and a space, and then with the last name. When you run this SQL statement:

```
SELECT SUBSTR(FIRSTNAME,1,1)||'. '||LASTNAME
  FROM Q.INTERVIEW
  WHERE TEMPID = 400
```

QMF produces this report:



```
COL1
-----------
R. FROMMHERZ
```

Making your query reusable with substitution variables

When you specify substitution variables in an SQL query, you can use the same query to retrieve different information by supplying a new value for the variable each time you run the query.

The following query selects department data. By using a substitution variable (&DEPARTMENT) for the department number in the row condition, you can specify a different department number each time you run the query.

```
SELECT ID, NAME, JOB, SALARY
FROM Q.STAFF
WHERE DEPT=&DEPARTMENT
```

You can specify values for substitution variables in any of the following ways:

- As a part of the RUN command
- From the RUN command prompt panel
- By setting a global variable

To specify a value as part of the RUN command

For example, to specify a value for the &DEPARTMENT variable, on the QMF command line, enter:

```
RUN QUERY (&DEPARTMENT = 38
```

Enclose the value in parentheses if it contains one of the following special characters:

- Blank
- Comma
- Left or right parenthesis
- Single or double quotation mark
- Equal sign

For example:

```
RUN QUERY (&X=(DEPT,NAME,SALARY)
```

To specify text for a variable, just type the text. You might have to enclose the text with quotes, depending on whether it would require quotes if you entered it directly into the query. For example, the following query has two variables. For the first you specify a column name as the value; for the second, you specify text that contains a quotation mark.

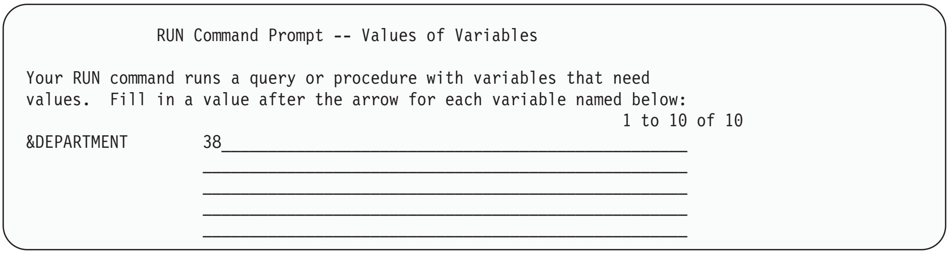
```
SELECT &X
FROM Q.STAFF
WHERE NAME=&Y
```

If the text itself contains quotation marks, add another set of quotation marks for each quotation mark:

```
RUN QUERY (&X=SALARY, &Y='O''BRIEN'
```

To specify a value on the RUN Command Prompt panel: If your query contains a variable, and you do not specify a value for the variable when you type the RUN command, the RUN Command Prompt panel displays.

The prompt panel displays the variables that need values. Type the values for the variables.



```
             RUN Command Prompt -- Values of Variables

Your RUN command runs a query or procedure with variables that need
values.  Fill in a value after the arrow for each variable named below:
                                                              1 to 10 of 10
&DEPARTMENT       38__________________________________________________
                  ____________________________________________________
                  ____________________________________________________
                  ____________________________________________________
                  ____________________________________________________
```

To specify values for substitution variables using global variables: You can define global variables with the SET GLOBAL command. A global variable keeps its value until you reset it, or until you end the QMF session.

For example, to set a global variable value for the &DEPARTMENT variable, on the QMF command line, enter:

```
SET GLOBAL (DEPARTMENT=38
```

You can specify up to 10 variable values. Separate the values with commas or with blanks.

For more information on defining global variables, see*DB2 QMF Reference*.

Saving a new query

You can save your query in the database after you create it. You can run a saved query and display the report again. You can also add, delete, or change the information in a saved query.

To save a query: On the QMF command line of the SQL Query panel, enter:

```
SAVE
```

QMF prompts you for the name you want to assign to the query.

You can also enter the following:

```
SAVE AS queryname
```

For example, to save your query in the database and name it MYQUERY, enter:

```
SAVE AS MYQUERY
```

To save a query and share it with other users, add the SHARE=YES parameter to the SAVE command you are using as follows:

```
SAVE (SHARE=YES
SAVE AS queryname (SHARE=YES
```

QMF saves your query in the database. The SQL Query panel displays with the name you gave the query. If you issue a SET GLOBAL command with the value DSQEC_SHARE=1 prior to issuing the SAVE command, the SHARE=YES parameter is not needed.

To retrieve a query from the database, enter:

```
DISPLAY QUERY queryname
```

Chapter 6. Customizing your reports

In this chapter you will learn how to change the appearance of your report by changing the default report format.

QMF form panels

Change the appearance of your reports by changing report information on the nine QMF form panels. You specify a different part of your report information on each panel. You can also customize reports in a Windows environment by using the DB2 QMF for Windows feature. See Appendix C, "The DB2 QMF High Performance Option," on page 367 for more information.

Figure 77 shows the QMF form panels and their purposes.

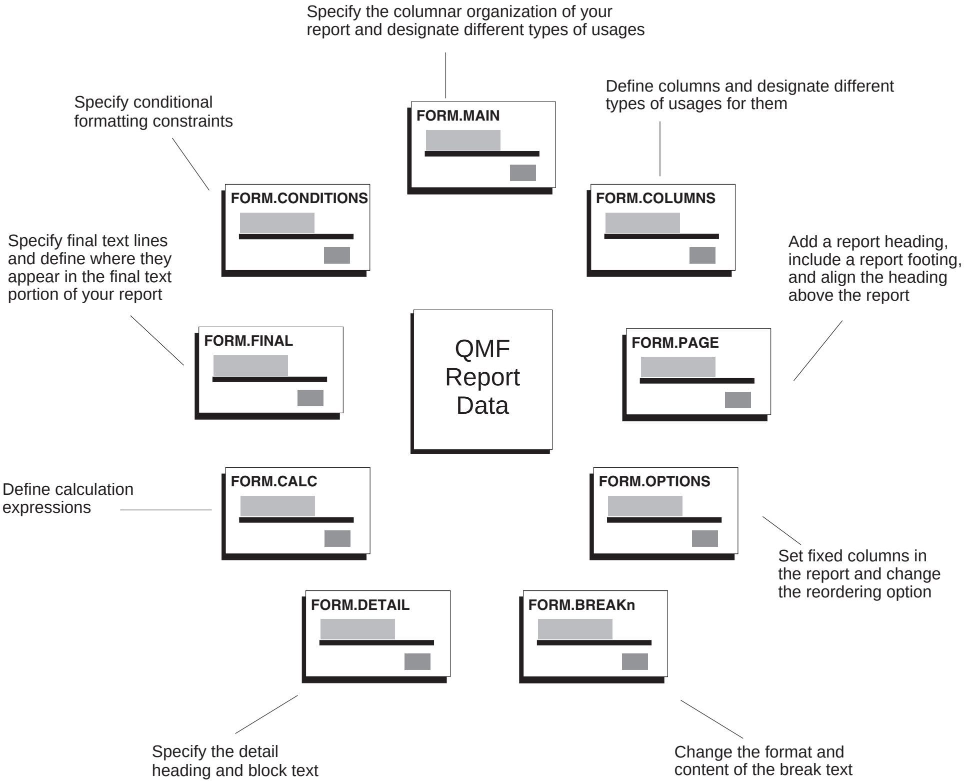

Specify the columnar organization of your report and designate different types of usages
FORM.MAIN
Specify conditional formatting constraints
FORM.CONDITIONS
Define columns and designate different types of usages for them
FORM.COLUMNS
Specify final text lines and define where they appear in the final text portion of your report
FORM.FINAL
QMF Report Data
Add a report heading, include a report footing, and align the heading above the report
FORM.PAGE
Define calculation expressions
FORM.CALC
FORM.OPTIONS
Set fixed columns in the report and change the reordering option
FORM.DETAIL
FORM.BREAKn
Specify the detail heading and block text
Change the format and content of the break text

Figure 77. QMF creates reports from forms you fill in.

Display form panels by using either the SHOW command or the DISPLAY command. For example, to display FORM.MAIN, enter one of the following commands:

```
SHOW FORM.MAIN        or     SH F.M
DISPLAY FORM.MAIN     or     DI FORM.MAIN
```

Or, you can enter:

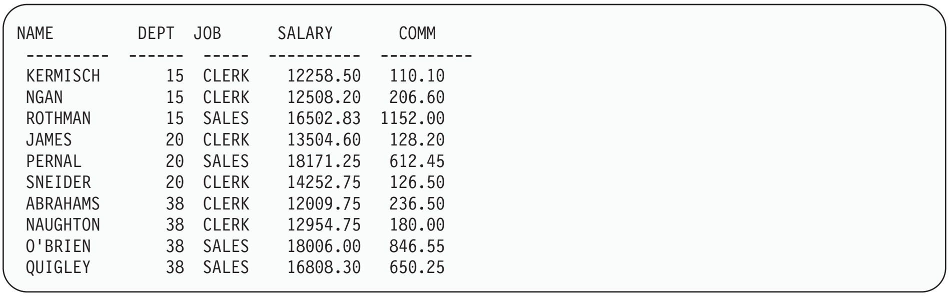
```
FORM.MAIN   or    F.M
```

Then, press the Show function key. You can use the abbreviated form panel name when using the SHOW command. For a complete list of the shortened form panel names, enter:

```
SHOW ?
```

Using the QMF default report format

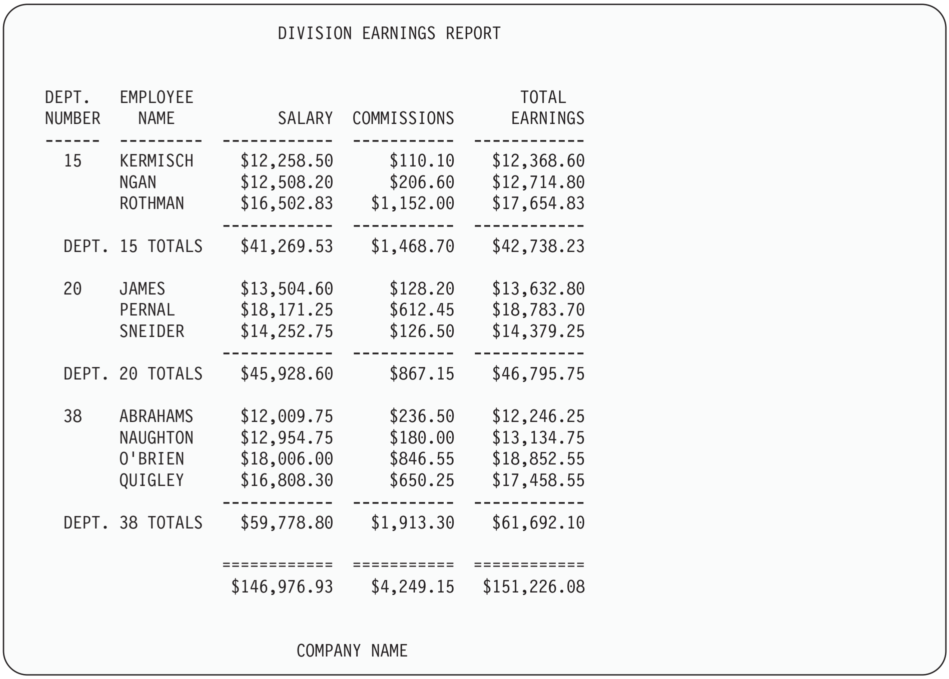
When you display a report by using the default report format, the report looks like the one in Figure 78.



```
NAME        DEPT  JOB      SALARY       COMM
 ---------  ------  -----  ----------  ----------
 KERMISCH       15  CLERK    12258.50   110.10
 NGAN           15  CLERK    12508.20   206.60
 ROTHMAN        15  SALES    16502.83  1152.00
 JAMES          20  CLERK    13504.60   128.20
 PERNAL         20  SALES    18171.25   612.45
 SNEIDER        20  CLERK    14252.75   126.50
 ABRAHAMS       38  CLERK    12009.75   236.50
 NAUGHTON       38  CLERK    12954.75   180.00
 O'BRIEN        38  SALES    18006.00   846.55
 QUIGLEY        38  SALES    16808.30   650.25
```

Figure 78. QMF uses a default report format like this one.

By tailoring the report format, you can use the same data to produce the report in Figure 79 on page 115.



```
                         DIVISION EARNINGS REPORT


 DEPT.   EMPLOYEE                                     TOTAL
 NUMBER    NAME          SALARY  COMMISSIONS       EARNINGS
 ------  ---------  ------------  -----------  ------------
   15    KERMISCH     $12,258.50      $110.10    $12,368.60
         NGAN         $12,508.20      $206.60    $12,714.80
         ROTHMAN      $16,502.83    $1,152.00    $17,654.83
                    ------------  -----------  ------------
   DEPT. 15 TOTALS    $41,269.53    $1,468.70    $42,738.23

   20    JAMES        $13,504.60      $128.20    $13,632.80
         PERNAL       $18,171.25      $612.45    $18,783.70
         SNEIDER      $14,252.75      $126.50    $14,379.25
                    ------------  -----------  ------------
   DEPT. 20 TOTALS    $45,928.60      $867.15    $46,795.75

   38    ABRAHAMS     $12,009.75      $236.50    $12,246.25
         NAUGHTON     $12,954.75      $180.00    $13,134.75
         O'BRIEN      $18,006.00      $846.55    $18,852.55
         QUIGLEY      $16,808.30      $650.25    $17,458.55
                    ------------  -----------  ------------
   DEPT. 38 TOTALS    $59,778.80    $1,913.30    $61,692.10

                    ============  ===========  ============
                     $146,976.93    $4,249.15   $151,226.08


                           COMPANY NAME
```

Figure 79. A customized report can show the same data in different ways.

Changing the columns on your report

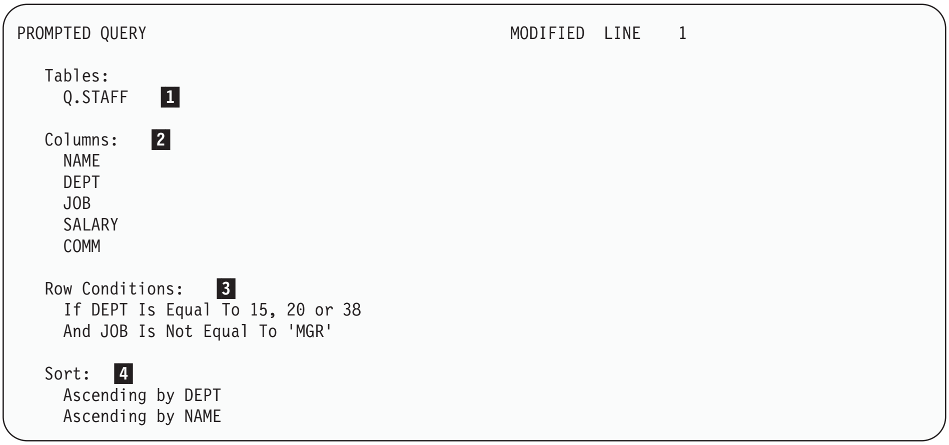
The first step in producing the report shown in Figure 79 is to change the appearance of the columns. To change columns, display the FORMS panels for the report.

To display the FORMS panels for a report:

1. Retrieve the data to display on the report by using either a prompted query or an SQL query.

   Figure 80 on page 116 shows the prompted query for the first few examples in this chapter. You can save this query and use it wherever you need it for the examples in this chapter. For example, on the QMF command line, enter:

   ```
   SAVE QUERY AS NEWQUERY
   ```



```
PROMPTED QUERY                                      MODIFIED  LINE    1

   Tables:
     Q.STAFF    1

   Columns:    2
     NAME
     DEPT
     JOB
     SALARY
     COMM

   Row Conditions:    3
     If DEPT Is Equal To 15, 20 or 38
     And JOB Is Not Equal To 'MGR'

   Sort:   4
     Ascending by DEPT
     Ascending by NAME
```

Figure 80. This query produces the data for the report.

1 QMF retrieves the data from the Q.STAFF table

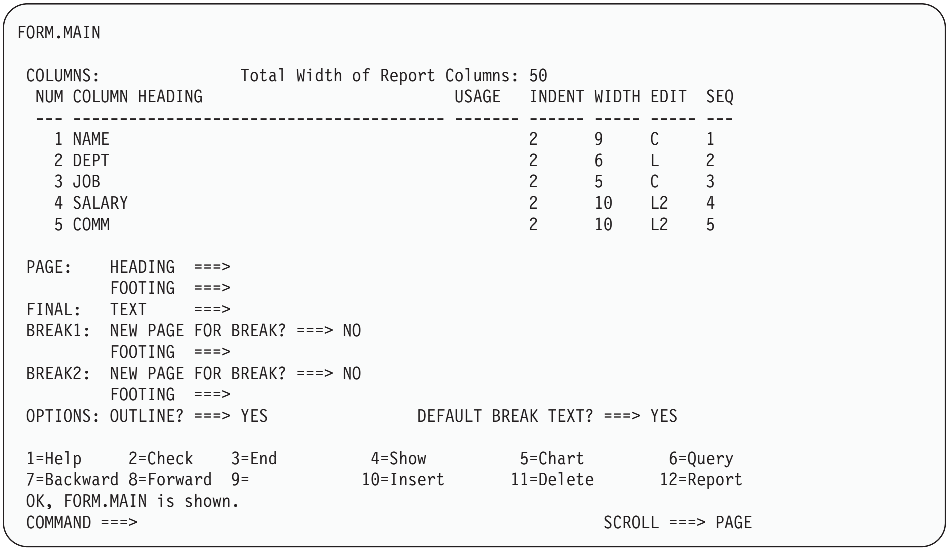
2 QMF displays these columns on the report

3 The employees are nonmanagers from departments 15, 20, and 38

4 QMF orders the rows by department number, and then by name

2. Run the query to display the report.
3. On the QMF command line, enter `SHOW FORM.MAIN`.

   The FORM.MAIN panel displays with the default report format for this report:



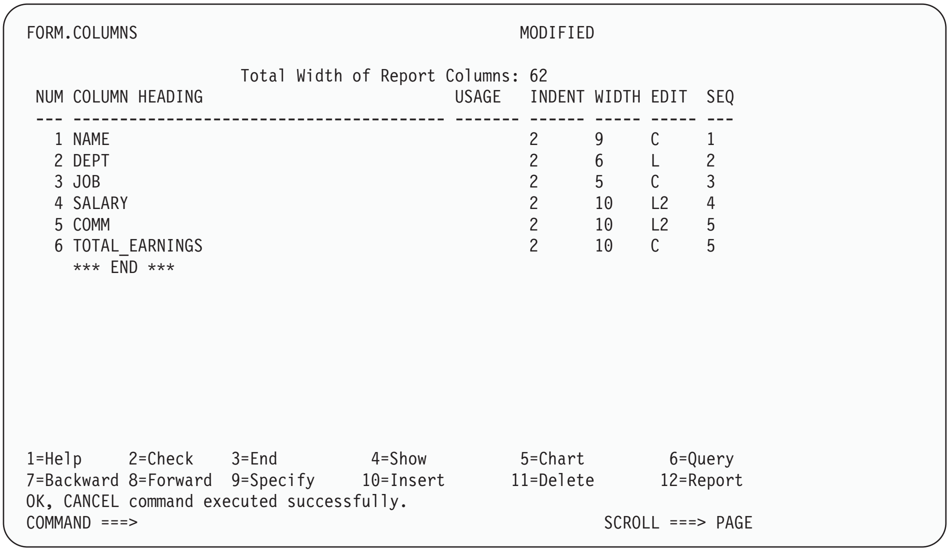
```
FORM.MAIN

 COLUMNS:                Total Width of Report Columns: 50
  NUM COLUMN HEADING                           USAGE   INDENT WIDTH EDIT  SEQ
  --- ---------------------------------------- ------- ------ ----- ----- ---
    1 NAME                                     2      9     C     1
    2 DEPT                                     2      6     L     2
    3 JOB                                      2      5     C     3
    4 SALARY                                   2      10    L2    4
    5 COMM                                     2      10    L2    5

 PAGE:    HEADING  ===>
          FOOTING  ===>
 FINAL:   TEXT     ===>
 BREAK1:  NEW PAGE FOR BREAK? ===> NO
          FOOTING  ===>
 BREAK2:  NEW PAGE FOR BREAK? ===> NO
          FOOTING  ===>
 OPTIONS: OUTLINE? ===> YES                DEFAULT BREAK TEXT? ===> YES

 1=Help     2=Check    3=End          4=Show         5=Chart         6=Query
 7=Backward 8=Forward  9=            10=Insert      11=Delete       12=Report
 OK, FORM.MAIN is shown.
 COMMAND ===>                                                  SCROLL ===> PAGE
```

Figure 81. You can change column appearance on the QMF FORM.MAIN panel.

You can make changes to the columns on the FORM.MAIN panel. However, in this example, you will see how to use the FORM.COLUMNS panel to make all the changes to the columns on your report.

4. Enter `SHOW FORM.COLUMNS`.

The FORM.COLUMNS panel displays with the default column information for this report.



```
FORM.COLUMNS                                          MODIFIED

                       Total Width of Report Columns: 62
 NUM COLUMN HEADING                           USAGE   INDENT WIDTH EDIT  SEQ
 --- ---------------------------------------- ------- ------ ----- ----- ---
   1 NAME                                             2      9     C     1
   2 DEPT                                             2      6     L     2
   3 JOB                                              2      5     C     3
   4 SALARY                                           2      10    L2    4
   5 COMM                                             2      10    L2    5
   6 TOTAL_EARNINGS                                   2      10    C     5
     *** END ***









1=Help     2=Check    3=End          4=Show          5=Chart         6=Query
7=Backward 8=Forward  9=Specify     10=Insert       11=Delete       12=Report
OK, CANCEL command executed successfully.
COMMAND ===>                                                   SCROLL ===> PAGE
```

Figure 82. The FORM.COLUMNS panel

Adding a new column to a report

You can define a new column on the FORM.COLUMNS panel similar to the way you define a new column in a query.

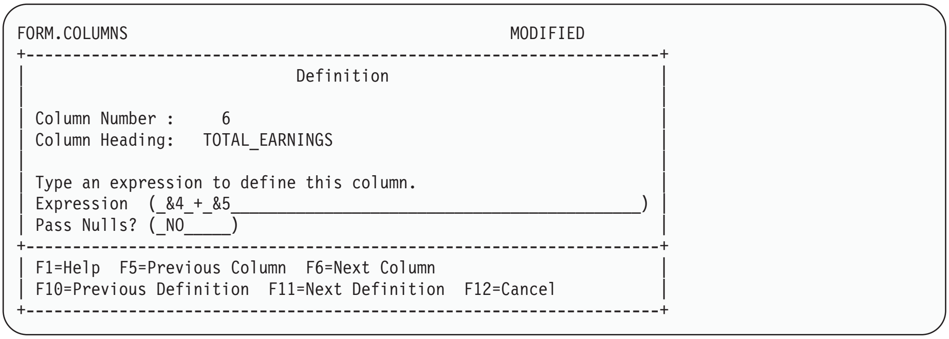
For this example, add a total earnings (SALARY+COMMISSION) column for each employee.

To add a new column:

1. On the FORM.COLUMNS panel, move the cursor to the column name after which you want to add the new column. In this example, move the cursor to the line for the COMM column.
2. Press the Insert function key. A new column number displays.
3. Type the column heading for the new column. For this example, type `TOTAL_EARNINGS`. Use an underscore to split the heading between two lines when it appears on the report.
4. Press the Specify function key. The Specify panel displays.

If you want to skip the Specify panel, type SPECIFY DEFINITION on the QMF command line. Then, move the cursor to the column you want to define and press Enter. The Definition panel for the column displays.

5. Select **Definition**.
6. Press Enter.

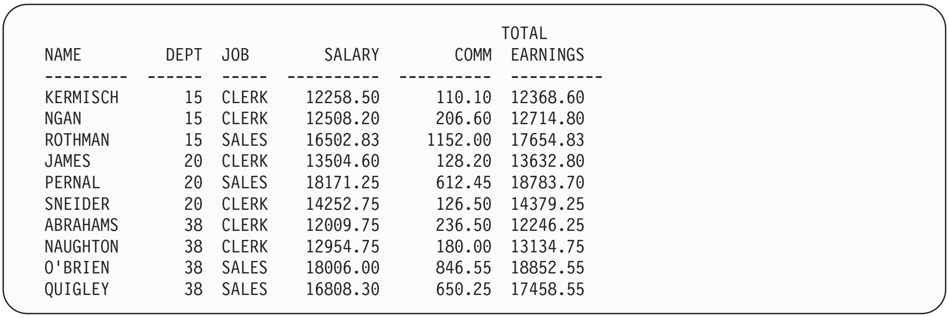


```
FORM.COLUMNS                                         MODIFIED
+---------------------------------------------------------------------+
|                             Definition                              |
|                                                                     |
| Column Number :     6                                               |
| Column Heading:   TOTAL_EARNINGS                                    |
|                                                                     |
| Type an expression to define this column.                           |
| Expression  (_&4_+_&5_____________________________________________) |
| Pass Nulls? (_NO_____)                                              |
+---------------------------------------------------------------------+
| F1=Help  F5=Previous Column  F6=Next Column                         |
| F10=Previous Definition  F11=Next Definition  F12=Cancel            |
+---------------------------------------------------------------------+
```

Figure 83. The Definition panel

7. Type the expression you want to use to define this column. For this example, type &4+&5, which stands for: the value in this column is equal to the value in column 4 plus the value in column 5 (SALARY+COMM).
8. Leave NO in the **Pass Nulls** field to process null values for the expression in this example.

   You can process null values if you are defining a column by using REXX execs. For more information on passing nulls and writing REXX execs, see *DB2 QMF Reference*.
9. Press Enter; you have finished defining the new column.
10. Press the Cancel function key to close the Specify panel.
11. Press the Report function key to display the changed report.



```
                                                    TOTAL
   NAME         DEPT  JOB        SALARY        COMM  EARNINGS
   ---------  ------  -----  ----------  ----------  ----------
   KERMISCH       15  CLERK    12258.50      110.10  12368.60
   NGAN           15  CLERK    12508.20      206.60  12714.80
   ROTHMAN        15  SALES    16502.83     1152.00  17654.83
   JAMES          20  CLERK    13504.60      128.20  13632.80
   PERNAL         20  SALES    18171.25      612.45  18783.70
   SNEIDER        20  CLERK    14252.75      126.50  14379.25
   ABRAHAMS       38  CLERK    12009.75      236.50  12246.25
   NAUGHTON       38  CLERK    12954.75      180.00  13134.75
   O'BRIEN        38  SALES    18006.00      846.55  18852.55
   QUIGLEY        38  SALES    16808.30      650.25  17458.55
```

Figure 84. The changed report shows the new column.

Changing the order in which columns are displayed

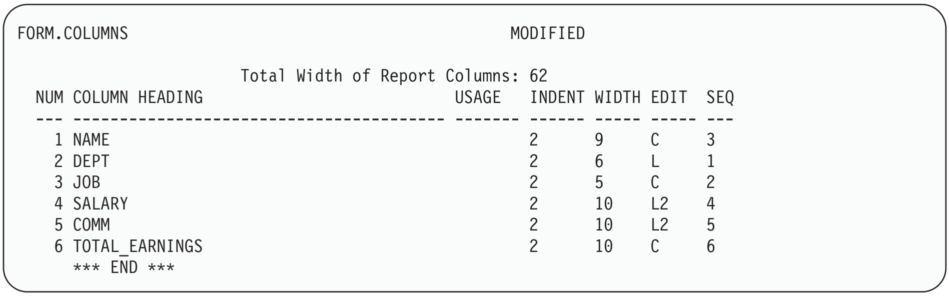
Change the order in which the columns are displayed in your report by changing the sequence, or **SEQ**, field for that column.

For this example, change the order of the columns to DEPT, JOB, NAME, SALARY, COMM, and TOTAL EARNINGS.

To change column order:

1. Type the new sequence number over the existing one.

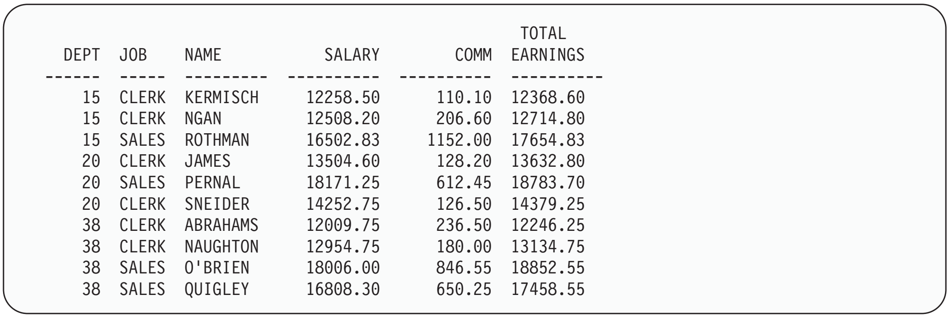
   In this example, type the following:
   - 3 for the NAME sequence number
   - 1 for the DEPT sequence number
   - 2 for the JOB sequence number
   - 6 for the TOTAL_EARNINGS sequence number



```
FORM.COLUMNS                                          MODIFIED

                       Total Width of Report Columns: 62
  NUM COLUMN HEADING                           USAGE   INDENT WIDTH EDIT  SEQ
  --- ---------------------------------------- ------- ------ ----- ----- ---
    1 NAME                                             2      9     C     3
    2 DEPT                                             2      6     L     1
    3 JOB                                              2      5     C     2
    4 SALARY                                           2      10    L2    4
    5 COMM                                             2      10    L2    5
    6 TOTAL_EARNINGS                                   2      10    C     6
      *** END ***
```

Figure 85. Change the order of columns by changing the SEQ field.

2. Press the Report function key to see the changed report.



```
                                                   TOTAL
     DEPT  JOB    NAME          SALARY        COMM  EARNINGS
   ------  -----  ---------  ----------  ----------  ----------
       15  CLERK  KERMISCH     12258.50      110.10  12368.60
       15  CLERK  NGAN         12508.20      206.60  12714.80
       15  SALES  ROTHMAN      16502.83     1152.00  17654.83
       20  CLERK  JAMES        13504.60      128.20  13632.80
       20  SALES  PERNAL       18171.25      612.45  18783.70
       20  CLERK  SNEIDER      14252.75      126.50  14379.25
       38  CLERK  ABRAHAMS     12009.75      236.50  12246.25
       38  CLERK  NAUGHTON     12954.75      180.00  13134.75
       38  SALES  O'BRIEN      18006.00      846.55  18852.55
       38  SALES  QUIGLEY      16808.30      650.25  17458.55
```

Figure 86. The changed report shows the columns in a new order.

Changing column headings

When you display a report by using the default report format, QMF assigns each column a name. Usually, this name is the column name or label from the table where you retrieve the data. Each column that you define in a query is given the name COL or EXPRESSION followed by a number so that each column name is unique. If you are using DB2 for AIX, QMF identifies columns you define by only a number.

You can change column headings on the FORM.COLUMNS panel.

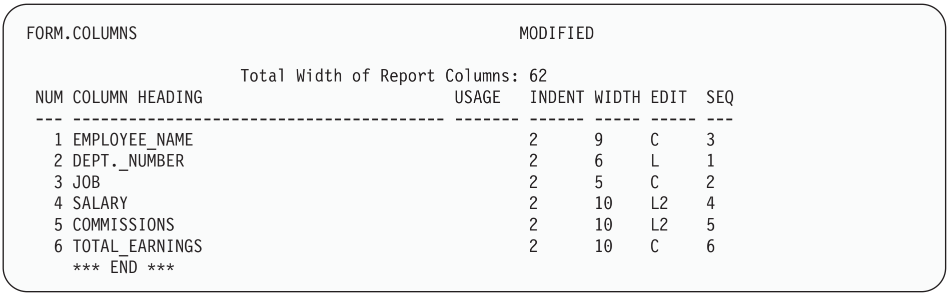
In this example, change the headings for the NAME, DEPT, and COMM columns.

To change column headings:

1. Type the new heading over the existing heading. Use an underscore to split the heading between two lines.

   For this example, type:

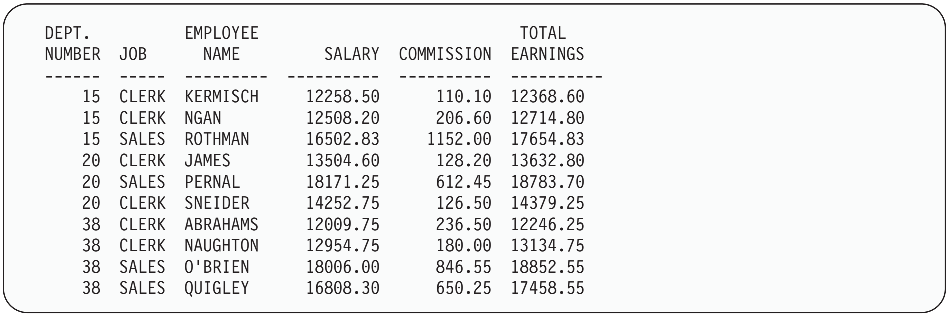
   - `EMPLOYEE_NAME` over NAME
   - `DEPT._NUMBER` over DEPT
   - `COMMISSIONS` over COMM



```
FORM.COLUMNS                                         MODIFIED

                      Total Width of Report Columns: 62
 NUM COLUMN HEADING                           USAGE   INDENT WIDTH EDIT  SEQ
 --- ---------------------------------------- ------- ------ ----- ----- ---
   1 EMPLOYEE_NAME                                    2      9     C     3
   2 DEPT._NUMBER                                     2      6     L     1
   3 JOB                                              2      5     C     2
   4 SALARY                                           2      10    L2    4
   5 COMMISSIONS                                      2      10    L2    5
   6 TOTAL_EARNINGS                                   2      10    C     6
     *** END ***
```

Figure 87. Change the column headings on FORM.COLUMNS.

2. Press the Report function key to see the changed report.



```
DEPT.         EMPLOYEE                             TOTAL
NUMBER  JOB     NAME         SALARY  COMMISSION  EARNINGS
------  -----  ---------  ----------  ----------  ----------
    15  CLERK  KERMISCH     12258.50      110.10  12368.60
    15  CLERK  NGAN         12508.20      206.60  12714.80
    15  SALES  ROTHMAN      16502.83     1152.00  17654.83
    20  CLERK  JAMES        13504.60      128.20  13632.80
    20  SALES  PERNAL       18171.25      612.45  18783.70
    20  CLERK  SNEIDER      14252.75      126.50  14379.25
    38  CLERK  ABRAHAMS     12009.75      236.50  12246.25
    38  CLERK  NAUGHTON     12954.75      180.00  13134.75
    38  SALES  O'BRIEN      18006.00      846.55  18852.55
    38  SALES  QUIGLEY      16808.30      650.25  17458.55
```

Figure 88. The changed report shows the new column headings.

Changing column widths and space between columns

In the report shown in Figure 88 on page 120, the last letter of the COMMISSIONS column heading does not appear because the column is not wide enough. You can change the column width and the spacing between columns on the FORM.COLUMNS panel.

To change column width or spacing:

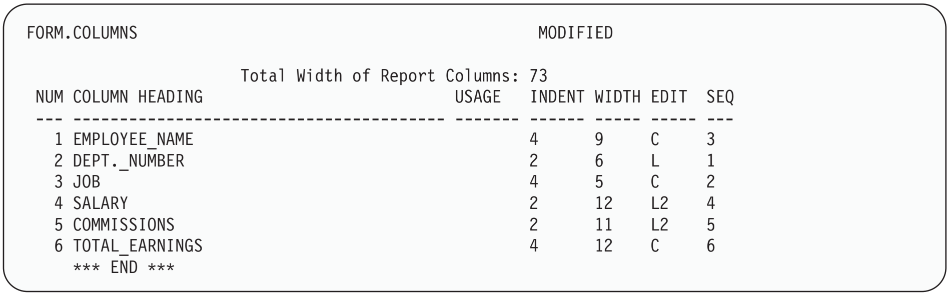
1. Move the cursor to the column whose width you want to change and type the new width under the **WIDTH** field. Be sure to include room for punctuation, such as dollar signs, commas, and decimal points.

   In this example, change the following:

   - 12 for the column width of the SALARY column
   - 11 for the column width of the COMMISSIONS column
   - 12 for the column width of the TOTAL_EARNINGS column

2. To change the amount of space between columns of data, move the cursor to the column you want to move to the right. Then, type the new spacing under the **INDENT** field.

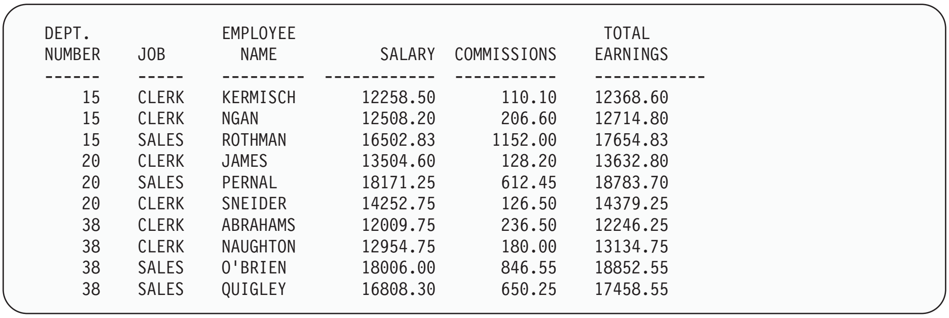
   For this example, type 4 for the spacing of the EMPLOYEE_NAME, JOB, and TOTAL_EARNINGS columns.



```
FORM.COLUMNS                                           MODIFIED

                       Total Width of Report Columns: 73
 NUM COLUMN HEADING                           USAGE   INDENT WIDTH EDIT  SEQ
 --- ---------------------------------------- ------- ------ ----- ----- ---
   1 EMPLOYEE_NAME                                    4      9     C     3
   2 DEPT._NUMBER                                     2      6     L     1
   3 JOB                                              4      5     C     2
   4 SALARY                                           2      12    L2    4
   5 COMMISSIONS                                      2      11    L2    5
   6 TOTAL_EARNINGS                                   4      12    C     6
     *** END ***
```

Figure 89. Change the width and spacing of columns on FORM.COLUMNS.

3. Press the Report function key to see the changed report.



```
DEPT.              EMPLOYEE                                         TOTAL
NUMBER    JOB       NAME          SALARY  COMMISSIONS    EARNINGS
------    -----    ---------   ------------  -----------    ------------
    15    CLERK    KERMISCH       12258.50       110.10    12368.60
    15    CLERK    NGAN           12508.20       206.60    12714.80
    15    SALES    ROTHMAN        16502.83      1152.00    17654.83
    20    CLERK    JAMES          13504.60       128.20    13632.80
    20    SALES    PERNAL         18171.25       612.45    18783.70
    20    CLERK    SNEIDER        14252.75       126.50    14379.25
    38    CLERK    ABRAHAMS       12009.75       236.50    12246.25
    38    CLERK    NAUGHTON       12954.75       180.00    13134.75
    38    SALES    O'BRIEN        18006.00       846.55    18852.55
    38    SALES    QUIGLEY        16808.30       650.25    17458.55
```

Figure 90. The changed report shows the new widths and spacing of columns.

Changing the alignment of the column heading and the data

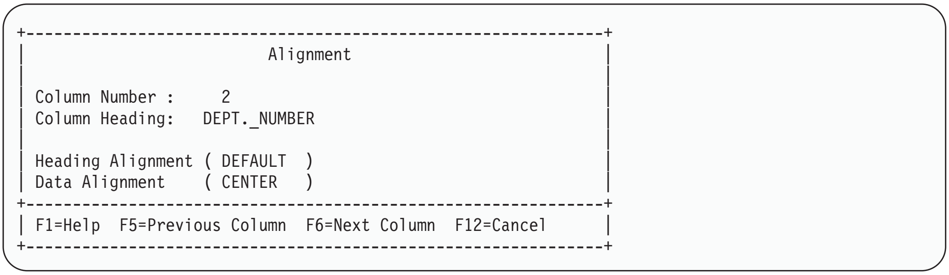
You change the alignment of the column heading and the data in much the same way you define a new column.

For this example, change the data alignment to CENTER for the DEPT._NUMBER column.

To change column alignment:

1. On the FORM.COLUMNS panel, move the cursor to the column whose alignment you want to change. For this example, move the cursor to the line for the DEPT NUMBER column.
2. Press the Specify function key. The Specify panel displays.

   If you want to skip the Specify panel, type `SPECIFY ALIGNMENT` on the QMF command line. Then, move the cursor to the column whose alignment you want to change and press Enter.
3. Select **Alignment**. The Alignment panel for the column displays.
4. Type the alignment for the data in the column (LEFT, CENTER, RIGHT, or DEFAULT). For this example, type `CENTER`.

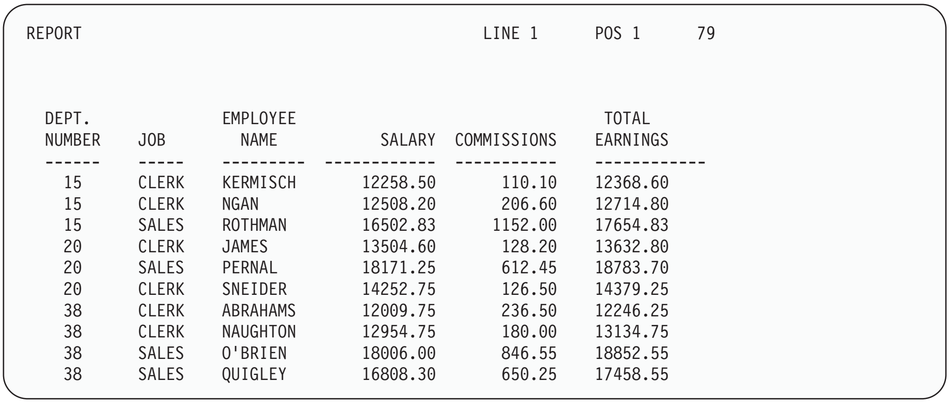


```
+--------------------------------------------------------------+
|                          Alignment                           |
|                                                              |
| Column Number :     2                                        |
| Column Heading:   DEPT._NUMBER                               |
|                                                              |
| Heading Alignment ( DEFAULT  )                               |
| Data Alignment    ( CENTER   )                               |
+--------------------------------------------------------------+
| F1=Help  F5=Previous Column  F6=Next Column  F12=Cancel      |
+--------------------------------------------------------------+
```

Figure 91. The Alignment panel

5. Press Enter. You have finished changing the alignment of the column.

6. Press the Cancel function key to close the Specify panel.
7. Press the Report function key to display the changed report.



```
REPORT                                           LINE 1      POS 1      79



  DEPT.              EMPLOYEE                                 TOTAL
  NUMBER    JOB        NAME           SALARY  COMMISSIONS    EARNINGS
  ------    -----    ---------  ------------  -----------    ------------
    15      CLERK    KERMISCH       12258.50       110.10    12368.60
    15      CLERK    NGAN           12508.20       206.60    12714.80
    15      SALES    ROTHMAN        16502.83      1152.00    17654.83
    20      CLERK    JAMES          13504.60       128.20    13632.80
    20      SALES    PERNAL         18171.25       612.45    18783.70
    20      CLERK    SNEIDER        14252.75       126.50    14379.25
    38      CLERK    ABRAHAMS       12009.75       236.50    12246.25
    38      CLERK    NAUGHTON       12954.75       180.00    13134.75
    38      SALES    O'BRIEN        18006.00       846.55    18852.55
    38      SALES    QUIGLEY        16808.30       650.25    17458.55
```

Figure 92. The changed report shows the centered column.

Specifying punctuation for the values in a column

You can use QMF's edit codes to specify how to punctuate the values in a column. An edit code is a set of characters that tells QMF how to format and punctuate the data in a specific column of a report. You specify edit codes on the FORM.COLUMNS panel. QMF assigns a default edit code that is based on the data type of the column to each column in a report. You can change the default. For example, you can use an edit code to specify a currency symbol, a decimal point, and commas for numeric data.

You can follow a numeric data edit code (such as **L**, **D**, **P**, or **K**) with a number that indicates the number of decimal places to use for that data. This number can range from 0 to 99. For example, L2 means to display a numeric value by using the L edit code, and allow 2 digits after a decimal.

Here are some common edit codes:

C Character data—specifies no punctuation

L Numeric data—specifies a decimal point and negative sign, if they occur

D Numeric data—specifies a currency symbol and a separator for groups of three digits, as well as a decimal point and negative sign, if they occur

P Numeric data—specifies numeric data as a percentage by using the % symbol, as well as a decimal point and negative sign, if they occur

K Numeric data—supplies a minus sign for negative values, a separator for groups of three digits, and decimal placement

M Metadata— specifies that metadata will be shown and the Descriptor Area will be displayed in character format rather than actual column data. The Descriptor Area consists of the data type and the length of the data. Edit code ′M′ displays the LOB data types (CLOB, DBCLOB, and BLOB) and the defined length field by default.

Note: If the column length of specified form width is less than the amount needed to display the complete Descriptor Area, the Descriptor Area will be truncated to fit into the column space. Edit code ′M′ will not alter the width of the column. If the user changes to or from edit code ′M′ within the form, the unaltered normal result set will be displayed. Edit code ′M′ does not modify the data row.

If a user wants to view the actual LOB data, he can modify FORM.MAIN or FORM.COLUMNS, and change the column edit code to ′C′ or ′CW′ to display character data. The maximum horizontal width of a report is 32 KB. If the user wants to display LOB data greater than 32 KB, he must specify a column wrapping edit code. If a wrapping edit code is not specified, the LOB data that does not fit within the report will not be displayed; no message will be issued. if a column with the ′M′ edit code is null, no metadata will be displayed, and a null indicator will be displayed.

Suppressing zero values

With numeric data edit codes, you can also choose to use a **Z** edit code in the second position to suppress zero values in a report. For example, **DZ** indicates numeric data, zero suppression, with a currency symbol, a separator for groups of three digits, and a decimal point and negative sign, if they occur.

Specifying a currency symbol

With the **D** edit code, you can also choose to use a **C** edit code in the second or third position. The **C** edit code causes QMF to use the currency symbol that you specify with the DSQDC_CURRENCY global variable.

For example, **DC** indicates numeric data, with the currency symbol that is specified with the DSQDC_CURRENCY global variable, a separator for groups of three digits, and a decimal point and negative sign, if they occur.

Note that if you use both **Z** and **C** with the **D** edit code, **C** must follow **Z**.

For additional information about edit codes, see *DB2 QMF Reference*.

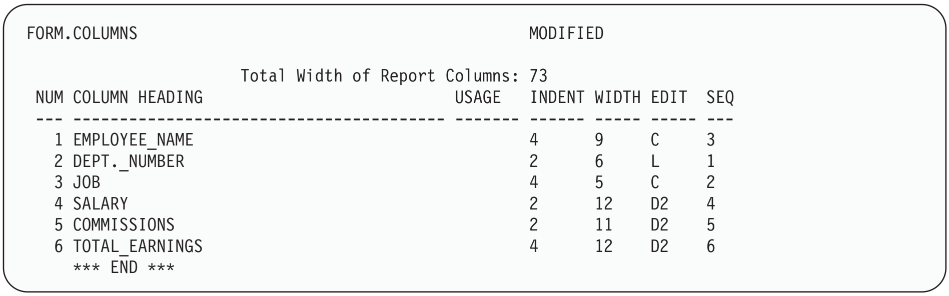
Changing edit codes

In this example, you will change the edit codes for the SALARY, COMMISSIONS, and TOTAL_EARNINGS columns to display the values as dollar amounts.

To change edit codes:

1. On the FORM.COLUMNS panel, move the cursor under the **EDIT** field for the column you want to change.
2. Type the new edit code.

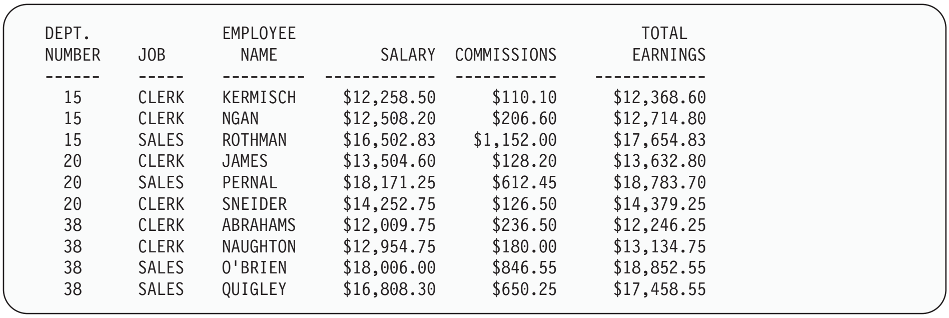
   In this example, type D2 in the SALARY, COMMISSIONS, and TOTAL_EARNINGS columns. The D2 edit code tells QMF to punctuate the values in these columns with a currency symbol, and to allow two digits after the decimal.



```
FORM.COLUMNS                                         MODIFIED

                       Total Width of Report Columns: 73
 NUM COLUMN HEADING                           USAGE   INDENT WIDTH EDIT  SEQ
 --- ---------------------------------------- ------- ------ ----- ----- ---
   1 EMPLOYEE_NAME                                    4      9     C     3
   2 DEPT._NUMBER                                     2      6     L     1
   3 JOB                                              4      5     C     2
   4 SALARY                                           2      12    D2    4
   5 COMMISSIONS                                      2      11    D2    5
   6 TOTAL_EARNINGS                                   4      12    D2    6
     *** END ***
```

Figure 93. Change the way columns are punctuated on FORM.COLUMNS.

3. Press the Report function key to see the changed report.



```
DEPT.            EMPLOYEE                                         TOTAL
NUMBER    JOB      NAME          SALARY  COMMISSIONS           EARNINGS
------    -----  ---------  ------------  -----------      ------------
  15      CLERK  KERMISCH     $12,258.50      $110.10        $12,368.60
  15      CLERK  NGAN         $12,508.20      $206.60        $12,714.80
  15      SALES  ROTHMAN      $16,502.83    $1,152.00        $17,654.83
  20      CLERK  JAMES        $13,504.60      $128.20        $13,632.80
  20      SALES  PERNAL       $18,171.25      $612.45        $18,783.70
  20      CLERK  SNEIDER      $14,252.75      $126.50        $14,379.25
  38      CLERK  ABRAHAMS     $12,009.75      $236.50        $12,246.25
  38      CLERK  NAUGHTON     $12,954.75      $180.00        $13,134.75
  38      SALES  O'BRIEN      $18,006.00      $846.55        $18,852.55
  38      SALES  QUIGLEY      $16,808.30      $650.25        $17,458.55
```

Figure 94. The changed report shows the dollar sign punctuation.

The default currency symbol displays. You can specify a different currency symbol to use in the report by using the currency symbol edit code.

To change the currency symbol:

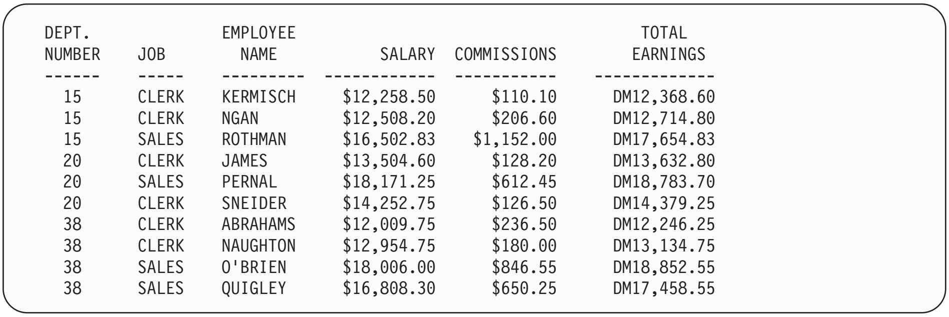
1. On the command line, enter the following to define a new currency symbol:

   ```
   SET GLOBAL (DSQDC_CURRENCY = DM
   ```

   where DM is the currency symbol you want to use.

The currency symbol can be a string with a length from 1 to 18 bytes.

2. On the FORM.COLUMNS panel, change the column width for the TOTAL _EARNINGS column to 13.
3. Change the edit code for TOTAL_EARNINGS to **DC2**. The edit code **C** causes QMF to display the currency symbol you defined with the SET GLOBAL (DSQDC_CURRENCY = DM command.
4. Press the Report function key to see the changed report.



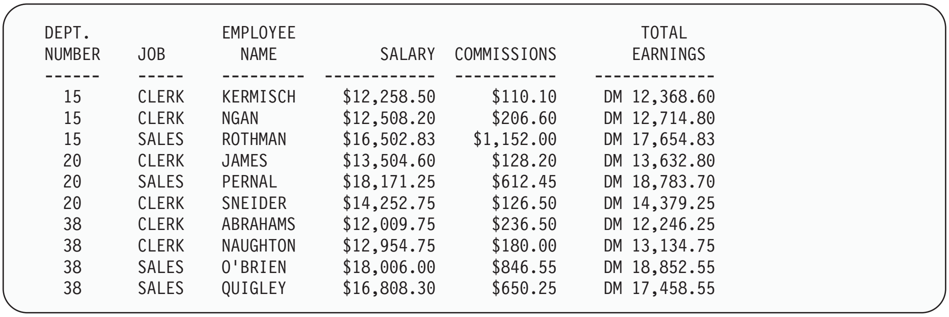
```
DEPT.            EMPLOYEE                                        TOTAL
NUMBER    JOB      NAME          SALARY  COMMISSIONS          EARNINGS
------    -----    ---------  ------------  -----------    -------------
  15      CLERK    KERMISCH     $12,258.50      $110.10      DM12,368.60
  15      CLERK    NGAN         $12,508.20      $206.60      DM12,714.80
  15      SALES    ROTHMAN      $16,502.83    $1,152.00      DM17,654.83
  20      CLERK    JAMES        $13,504.60      $128.20      DM13,632.80
  20      SALES    PERNAL       $18,171.25      $612.45      DM18,783.70
  20      CLERK    SNEIDER      $14,252.75      $126.50      DM14,379.25
  38      CLERK    ABRAHAMS     $12,009.75      $236.50      DM12,246.25
  38      CLERK    NAUGHTON     $12,954.75      $180.00      DM13,134.75
  38      SALES    O'BRIEN      $18,006.00      $846.55      DM18,852.55
  38      SALES    QUIGLEY      $16,808.30      $650.25      DM17,458.55
```

Figure 95. The changed report shows the Deutch mark punctuation.

If you want a space between the DM currency symbol and the currency values, reissue the SET GLOBAL command as follows:

```
SET GLOBAL (DSQDC_CURRENCY = 'DM '
```

5. Press the Report function key to see the changed report.



```
DEPT.            EMPLOYEE                                        TOTAL
NUMBER    JOB      NAME          SALARY  COMMISSIONS          EARNINGS
------    -----    ---------  ------------  -----------    -------------
  15      CLERK    KERMISCH     $12,258.50      $110.10     DM 12,368.60
  15      CLERK    NGAN         $12,508.20      $206.60     DM 12,714.80
  15      SALES    ROTHMAN      $16,502.83    $1,152.00     DM 17,654.83
  20      CLERK    JAMES        $13,504.60      $128.20     DM 13,632.80
  20      SALES    PERNAL       $18,171.25      $612.45     DM 18,783.70
  20      CLERK    SNEIDER      $14,252.75      $126.50     DM 14,379.25
  38      CLERK    ABRAHAMS     $12,009.75      $236.50     DM 12,246.25
  38      CLERK    NAUGHTON     $12,954.75      $180.00     DM 13,134.75
  38      SALES    O'BRIEN      $18,006.00      $846.55     DM 18,852.55
  38      SALES    QUIGLEY      $16,808.30      $650.25     DM 17,458.55
```

Figure 96. The changed report shows the Deutch mark punctuation.

To change the currency symbol to a dollar sign, issue this command:

```
SET GLOBAL (DSQDC_CURRENCY = $
```

Specifying the way QMF uses values in a column

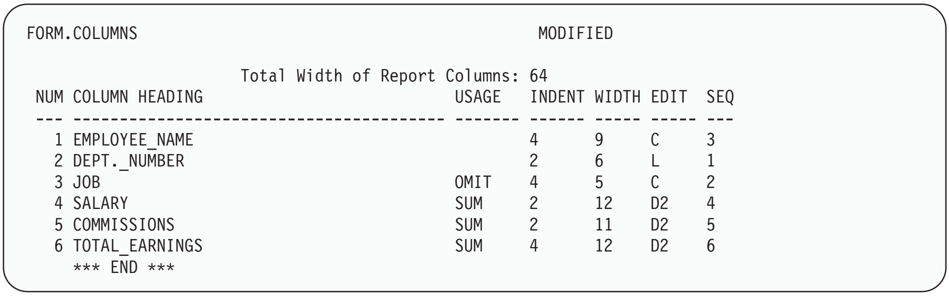
On the FORM.COLUMNS panel, you can specify usage codes to display the values in a column in a way that is meaningful to you. A usage code is a set of characters that tells QMF what to do with the values in a column when displaying them in the report.

For example, you can sum a column of numbers and display the total, or break the report at certain values to calculate subtotals.

In this example, you will sum a column of numbers, omit a column from the report, and calculate subtotals.

To specify a usage code:

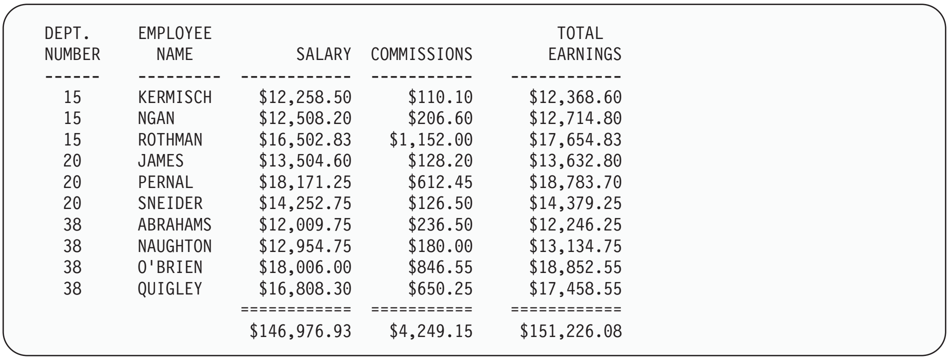
1. Move the cursor to the **USAGE** field for that column.
2. Type the usage code. For this example, type:
   - `SUM` in the **USAGE** field for the SALARY, COMMISSIONS, and TOTAL_EARNINGS columns.
   - `OMIT` for the JOB column.



```
FORM.COLUMNS                                          MODIFIED

                       Total Width of Report Columns: 64
 NUM COLUMN HEADING                           USAGE   INDENT WIDTH EDIT  SEQ
 --- ---------------------------------------- ------- ------ ----- ----- ---
   1 EMPLOYEE_NAME                                    4      9     C     3
   2 DEPT._NUMBER                                     2      6     L     1
   3 JOB                                      OMIT    4      5     C     2
   4 SALARY                                   SUM     2      12    D2    4
   5 COMMISSIONS                              SUM     2      11    D2    5
   6 TOTAL_EARNINGS                           SUM     4      12    D2    6
     *** END ***
```

Figure 97. Change the way QMF displays columns with usage codes.

3. Press the Report function key to display the changed report.



```
DEPT.     EMPLOYEE                                       TOTAL
NUMBER      NAME          SALARY  COMMISSIONS          EARNINGS
------    ---------  ------------  -----------     ------------
  15      KERMISCH     $12,258.50      $110.10       $12,368.60
  15      NGAN         $12,508.20      $206.60       $12,714.80
  15      ROTHMAN      $16,502.83    $1,152.00       $17,654.83
  20      JAMES        $13,504.60      $128.20       $13,632.80
  20      PERNAL       $18,171.25      $612.45       $18,783.70
  20      SNEIDER      $14,252.75      $126.50       $14,379.25
  38      ABRAHAMS     $12,009.75      $236.50       $12,246.25
  38      NAUGHTON     $12,954.75      $180.00       $13,134.75
  38      O'BRIEN      $18,006.00      $846.55       $18,852.55
  38      QUIGLEY      $16,808.30      $650.25       $17,458.55
                     ============  ===========     ============
                      $146,976.93    $4,249.15      $151,226.08
```

Figure 98. The changed report sums the columns and omits the JOB column.

For information on other usage codes, see the *QMF™ Reference*.

Adding subtotals to a report

Adding subtotals to a report can help make the report easier to read and easier to understand.

Breaking on a value to add subtotals

To include subtotals in the report, specify a BREAK usage code that tells QMF where to make a break in the report and calculate a subtotal.

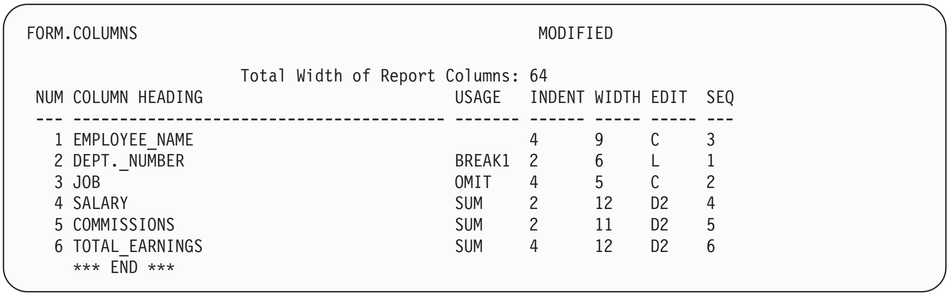
The BREAKn usage code divides the report whenever the value of the associated column changes. Control breaks are the points at which a report breaks. Control columns are the columns that govern control breaks. You can have up to six breaks in a report. To indicate a break, you type the word BREAK and follow it with a number from 1 to 6.

In this example, you add a break to one column to divide the report every time the department number changes.

Because you want to show subtotals by department (calculate a subtotal whenever the department number changes), specify the break usage code in the DEPT_NUMBER column. Therefore, the DEPT_NUMBER column is the control column.

To add subtotals:

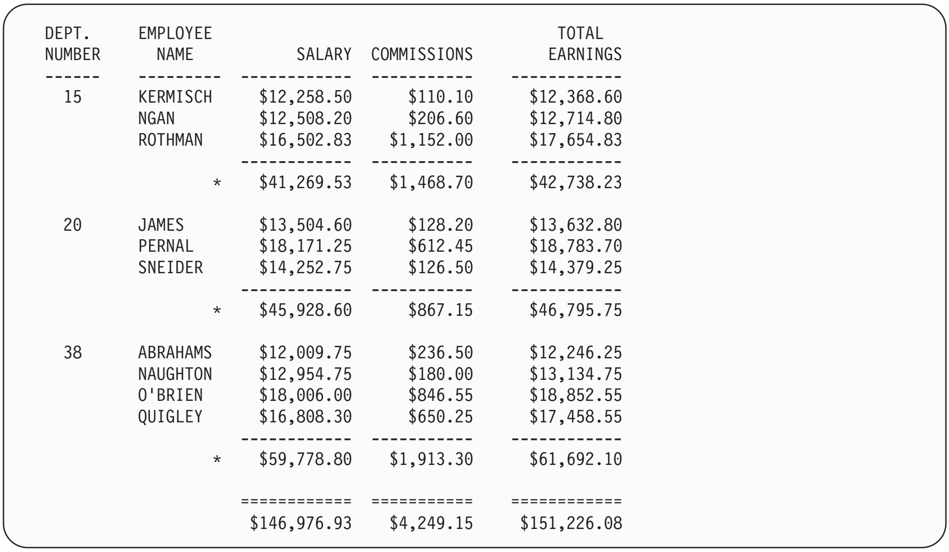
1. On the FORM.COLUMNS panel, move the cursor to the **USAGE** field for the column you want to use as the control column.
2. For this example, type `BREAK1` for the DEPT_NUMBER column.



```
FORM.COLUMNS                                          MODIFIED

                       Total Width of Report Columns: 64
 NUM COLUMN HEADING                           USAGE   INDENT WIDTH EDIT  SEQ
 --- ---------------------------------------- ------- ------ ----- ----- ---
   1 EMPLOYEE_NAME                                    4      9     C     3
   2 DEPT._NUMBER                             BREAK1  2      6     L     1
   3 JOB                                      OMIT    4      5     C     2
   4 SALARY                                   SUM     2      12    D2    4
   5 COMMISSIONS                              SUM     2      11    D2    5
   6 TOTAL_EARNINGS                           SUM     4      12    D2    6
     *** END ***
```

Figure 99. Create breaks for subtotals in your report on FORM.COLUMNS.

3. Press the Report function key to display the changed report.



```
  DEPT.     EMPLOYEE                                         TOTAL
  NUMBER      NAME            SALARY  COMMISSIONS          EARNINGS
  ------    ---------   ------------  -----------      ------------
    15      KERMISCH      $12,258.50      $110.10        $12,368.60
            NGAN          $12,508.20      $206.60        $12,714.80
            ROTHMAN       $16,502.83    $1,152.00        $17,654.83
                        ------------  -----------      ------------
                    *     $41,269.53    $1,468.70        $42,738.23

    20      JAMES         $13,504.60      $128.20        $13,632.80
            PERNAL        $18,171.25      $612.45        $18,783.70
            SNEIDER       $14,252.75      $126.50        $14,379.25
                        ------------  -----------      ------------
                    *     $45,928.60      $867.15        $46,795.75

    38      ABRAHAMS      $12,009.75      $236.50        $12,246.25
            NAUGHTON      $12,954.75      $180.00        $13,134.75
            O'BRIEN       $18,006.00      $846.55        $18,852.55
            QUIGLEY       $16,808.30      $650.25        $17,458.55
                        ------------  -----------      ------------
                    *     $59,778.80    $1,913.30        $61,692.10

                        ============  ===========      ============
                         $146,976.93    $4,249.15       $151,226.08
```

Figure 100. The changed report shows a subtotal after each department.

For more information on specifying control breaks, see*DB2 QMF Reference*.

Specifying text for the subtotal line on a report

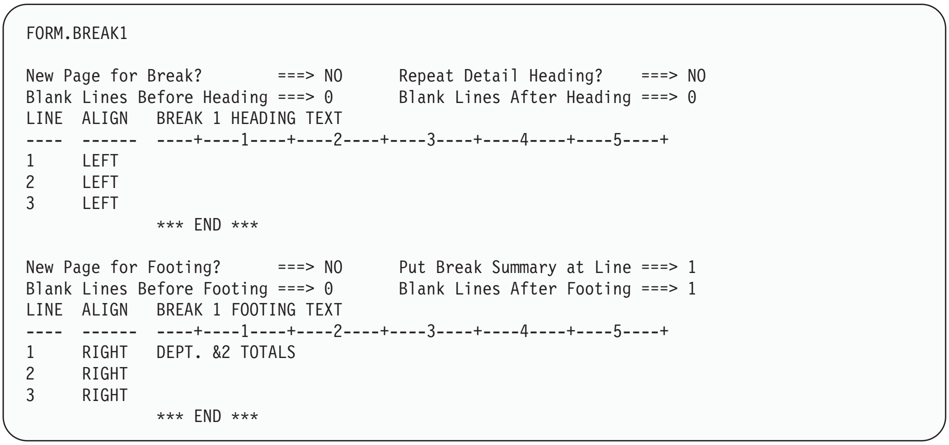
You can specify the text you want to display at each subtotal line in your report. If you do not specify the text, asterisks display at each subtotal line.

To specify text for subtotal lines:

1. On the QMF command line, enter:

   ```
   SHOW FORM.BREAK1
   ```

   The FORM.BREAK1 panel displays.



```
FORM.BREAK1

New Page for Break?        ===> NO      Repeat Detail Heading?    ===> NO
Blank Lines Before Heading ===> 0       Blank Lines After Heading ===> 0
LINE  ALIGN   BREAK 1 HEADING TEXT
----  ------  ----+----1----+----2----+----3----+----4----+----5----+
1     LEFT
2     LEFT
3     LEFT
              *** END ***

New Page for Footing?      ===> NO      Put Break Summary at Line ===> 1
Blank Lines Before Footing ===> 0       Blank Lines After Footing ===> 1
LINE  ALIGN   BREAK 1 FOOTING TEXT
----  ------  ----+----1----+----2----+----3----+----4----+----5----+
1     RIGHT   DEPT. &2 TOTALS
2     RIGHT
3     RIGHT
              *** END ***
```

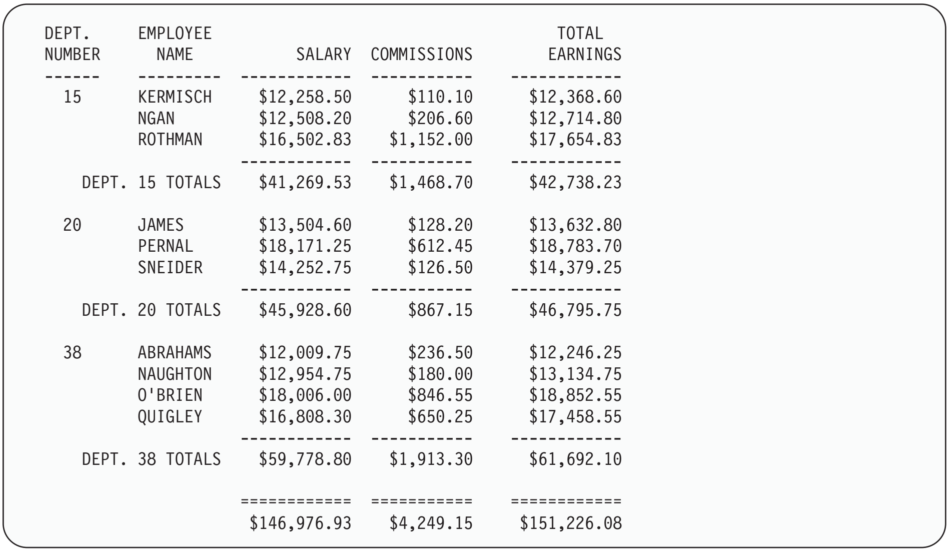
Figure 101. Enter the subtotal text in the BREAK 1 FOOTING TEXT field.

2. Type the text you want to display at each subtotal line in the **BREAK 1 FOOTING TEXT** field.

   For this example, type `DEPT. &2 TOTALS`.

   The **&2** is a form variable that tells QMF to display the current value in column 2 for each subtotal line. Column 2 is the DEPT_NUMBER column, so the current department number displays as part of the text for each subtotal line.

3. Press the Report function key to see the changed report.



```
DEPT.     EMPLOYEE                                      TOTAL
NUMBER      NAME          SALARY  COMMISSIONS         EARNINGS
------    ---------  ------------  -----------     ------------
  15      KERMISCH     $12,258.50      $110.10       $12,368.60
          NGAN         $12,508.20      $206.60       $12,714.80
          ROTHMAN      $16,502.83    $1,152.00       $17,654.83
                     ------------  -----------     ------------
    DEPT. 15 TOTALS    $41,269.53    $1,468.70       $42,738.23

  20      JAMES        $13,504.60      $128.20       $13,632.80
          PERNAL       $18,171.25      $612.45       $18,783.70
          SNEIDER      $14,252.75      $126.50       $14,379.25
                     ------------  -----------     ------------
    DEPT. 20 TOTALS    $45,928.60      $867.15       $46,795.75

  38      ABRAHAMS     $12,009.75      $236.50       $12,246.25
          NAUGHTON     $12,954.75      $180.00       $13,134.75
          O'BRIEN      $18,006.00      $846.55       $18,852.55
          QUIGLEY      $16,808.30      $650.25       $17,458.55
                     ------------  -----------     ------------
    DEPT. 38 TOTALS    $59,778.80    $1,913.30       $61,692.10

                     ============  ===========     ============
                      $146,976.93    $4,249.15      $151,226.08
```

Figure 102. The changed report shows subtotal text after each department.

For information on specifying text for additional form breaks, see *DB2 QMF Reference*.

Adding page headings and footings

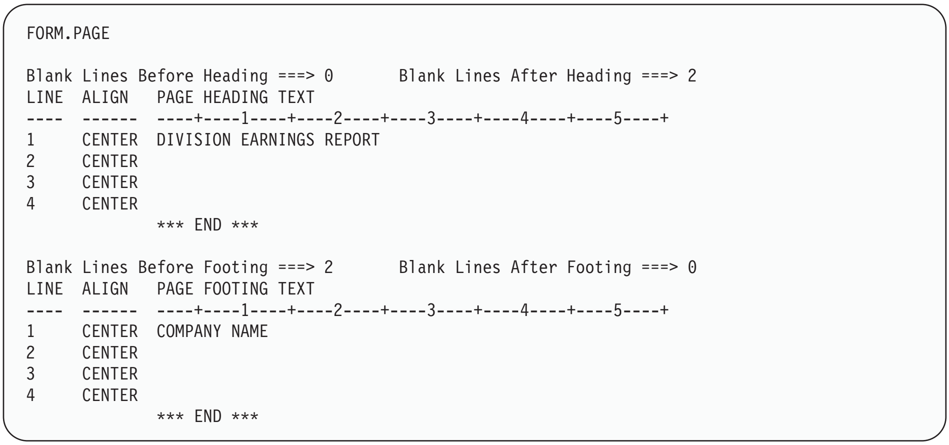
You can display headings and footings at the top and bottom of online and printed reports.

To add page headings and footings*:*

1. On the QMF command line, enter:

   ```
   SHOW FORM.PAGE
   ```

   The FORM.PAGE panel displays. Figure 103 shows a sample FORM.PAGE panel.



```
FORM.PAGE

Blank Lines Before Heading ===> 0      Blank Lines After Heading ===> 2
LINE  ALIGN   PAGE HEADING TEXT
----  ------  ----+----1----+----2----+----3----+----4----+----5----+
1     CENTER  DIVISION EARNINGS REPORT
2     CENTER
3     CENTER
4     CENTER
              *** END ***

Blank Lines Before Footing ===> 2      Blank Lines After Footing ===> 0
LINE  ALIGN   PAGE FOOTING TEXT
----  ------  ----+----1----+----2----+----3----+----4----+----5----+
1     CENTER  COMPANY NAME
2     CENTER
3     CENTER
4     CENTER
              *** END ***
```

Figure 103. Add page headings and footings on the FORM.PAGE panel.

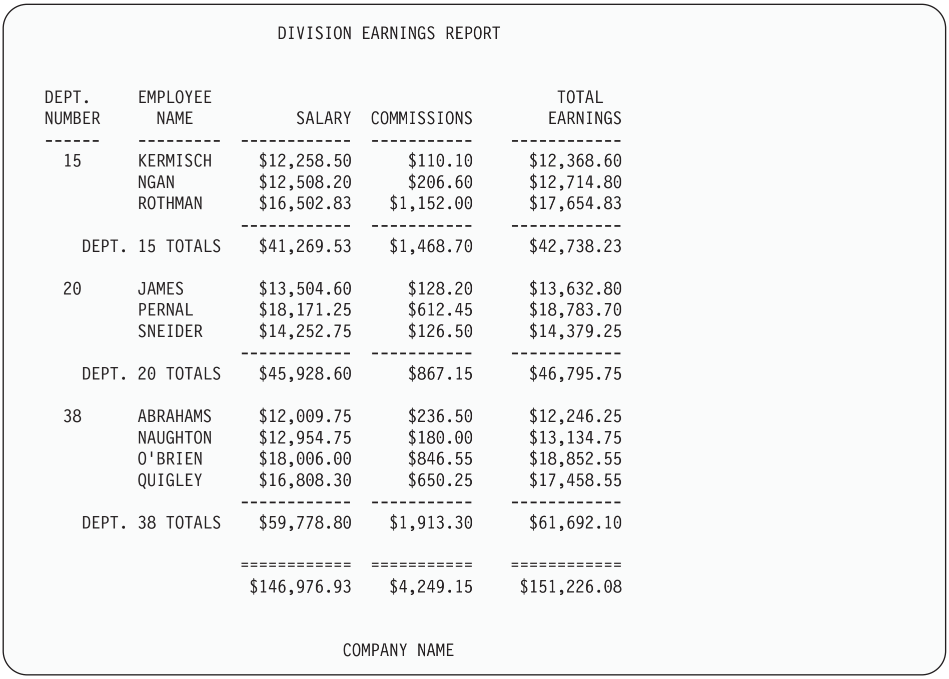
2. Move the cursor to the **PAGE HEADING TEXT** field.
3. Type the text you want to display at the top of each page in the report. For this example, type `DIVISION EARNINGS REPORT`.

   You can specify either left or right justification of the text, or align the text at a specific column.

   If all the heading text for one line does not fit in the space allowed for that line, type `APPEND` in the ALIGN column of the next line. Then, change its line number to the same number as the preceding line, and continue typing the text. Be sure to indent the appended text at least one space.

   If you need another line, move the cursor to the line above the place you want the new line and press the Insert function key. You can specify up to 999 lines of text.

4. Type the text you want to display at the bottom of each page for the report in the **PAGE FOOTING TEXT** field. For this example, type `COMPANY NAME`
5. Press the Report function key to see the changed report.



```
                             DIVISION EARNINGS REPORT


 DEPT.     EMPLOYEE                                    TOTAL
 NUMBER      NAME          SALARY  COMMISSIONS       EARNINGS
 ------    --------- ------------  -----------    ------------
   15      KERMISCH    $12,258.50      $110.10      $12,368.60
           NGAN        $12,508.20      $206.60      $12,714.80
           ROTHMAN     $16,502.83    $1,152.00      $17,654.83
                     ------------  -----------    ------------
     DEPT. 15 TOTALS   $41,269.53    $1,468.70      $42,738.23

   20      JAMES       $13,504.60      $128.20      $13,632.80
           PERNAL      $18,171.25      $612.45      $18,783.70
           SNEIDER     $14,252.75      $126.50      $14,379.25
                     ------------  -----------    ------------
     DEPT. 20 TOTALS   $45,928.60      $867.15      $46,795.75

   38      ABRAHAMS    $12,009.75      $236.50      $12,246.25
           NAUGHTON    $12,954.75      $180.00      $13,134.75
           O'BRIEN     $18,006.00      $846.55      $18,852.55
           QUIGLEY     $16,808.30      $650.25      $17,458.55
                     ------------  -----------    ------------
     DEPT. 38 TOTALS   $59,778.80    $1,913.30      $61,692.10

                     ============  ===========    ============
                      $146,976.93    $4,249.15     $151,226.08


                                    COMPANY NAME
```

Figure 104. The changed report displays the page heading and footing.

You have finished making changes to this report.

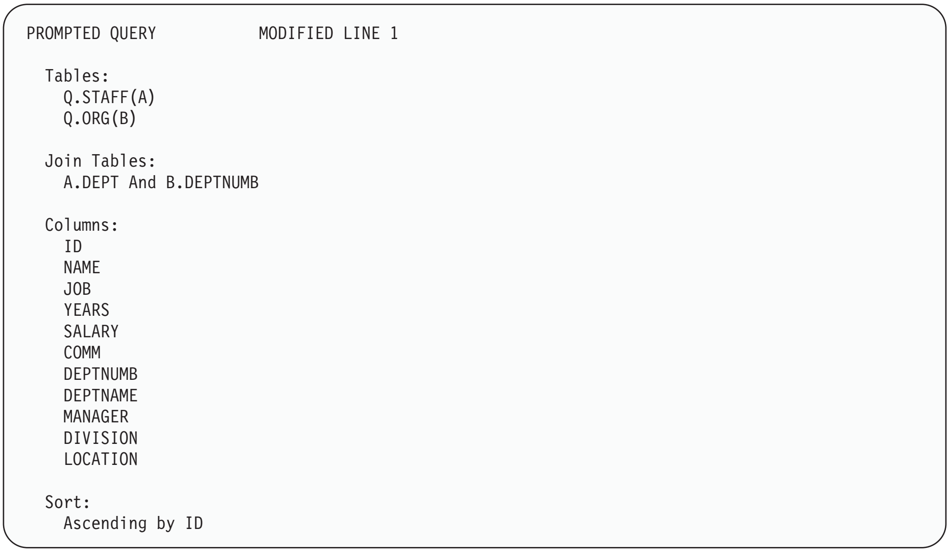
Specifying fixed columns on a report

Using forms panels, you can specify fixed columns on a report.

In an online report, the fixed columns remain in place on the left of the screen when you press the Left or Right function keys. A vertical line | separates the fixed area from the scrollable portion of the report.

On a printed report, QMF repeats the fixed columns on the left side of each page.

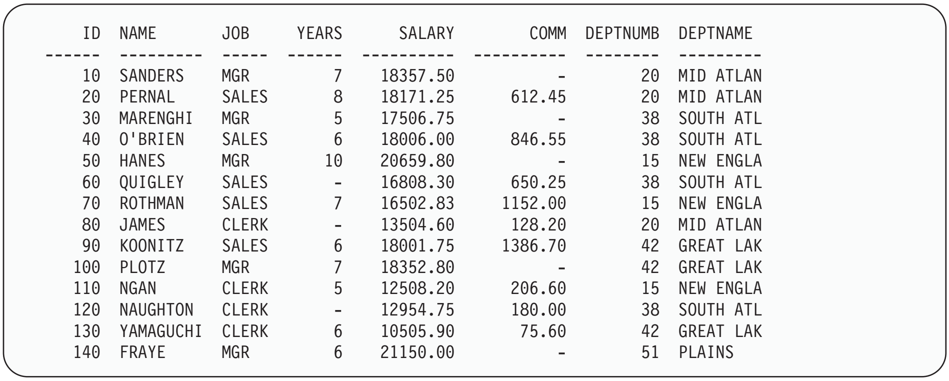
For this example, use the query in Figure 105 on page 133.



```
PROMPTED QUERY          MODIFIED LINE 1

  Tables:
    Q.STAFF(A)
    Q.ORG(B)

  Join Tables:
    A.DEPT And B.DEPTNUMB

  Columns:
    ID
    NAME
    JOB
    YEARS
    SALARY
    COMM
    DEPTNUMB
    DEPTNAME
    MANAGER
    DIVISION
    LOCATION

  Sort:
    Ascending by ID
```

Figure 105. Use this query to see how fixed columns affect a report.

The query creates this report:



```
      ID  NAME       JOB     YEARS      SALARY        COMM  DEPTNUMB  DEPTNAME
  ------  ---------  -----  ------  ----------  ----------  --------  ---------
      10  SANDERS    MGR         7    18357.50           -        20  MID ATLAN
      20  PERNAL     SALES       8    18171.25      612.45        20  MID ATLAN
      30  MARENGHI   MGR         5    17506.75           -        38  SOUTH ATL
      40  O'BRIEN    SALES       6    18006.00      846.55        38  SOUTH ATL
      50  HANES      MGR        10    20659.80           -        15  NEW ENGLA
      60  QUIGLEY    SALES       -    16808.30      650.25        38  SOUTH ATL
      70  ROTHMAN    SALES       7    16502.83     1152.00        15  NEW ENGLA
      80  JAMES      CLERK       -    13504.60      128.20        20  MID ATLAN
      90  KOONITZ    SALES       6    18001.75     1386.70        42  GREAT LAK
     100  PLOTZ      MGR         7    18352.80           -        42  GREAT LAK
     110  NGAN       CLERK       5    12508.20      206.60        15  NEW ENGLA
     120  NAUGHTON   CLERK       -    12954.75      180.00        38  SOUTH ATL
     130  YAMAGUCHI  CLERK       6    10505.90       75.60        42  GREAT LAK
     140  FRAYE      MGR         6    21150.00           -        51  PLAINS
```

Figure 106. The right side of the report is not visible.

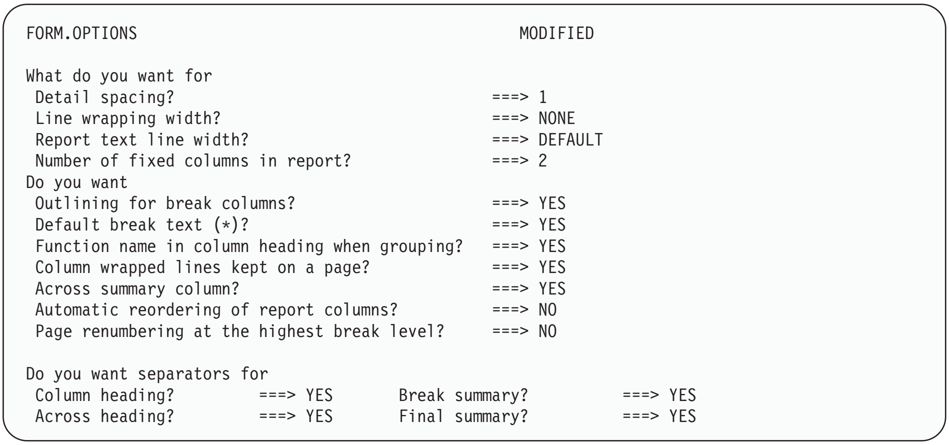
When you press the Right function key to see the rest of the information, you can no longer see the ID and NAME fields. You cannot tell which information belongs to which employee.

To specify fixed columns on a report:

1. On the QMF command line, enter:

   ```
   SHOW FORM.OPTIONS
   ```

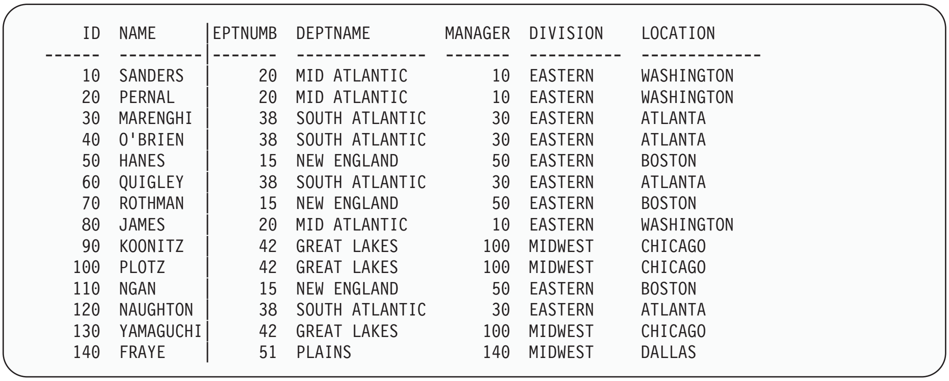
   The FORM.OPTIONS panel displays.



```
FORM.OPTIONS                                         MODIFIED

What do you want for
 Detail spacing?                                  ===> 1
 Line wrapping width?                             ===> NONE
 Report text line width?                          ===> DEFAULT
 Number of fixed columns in report?               ===> 2
Do you want
 Outlining for break columns?                     ===> YES
 Default break text (*)?                          ===> YES
 Function name in column heading when grouping?   ===> YES
 Column wrapped lines kept on a page?             ===> YES
 Across summary column?                           ===> YES
 Automatic reordering of report columns?          ===> NO
 Page renumbering at the highest break level?     ===> NO

Do you want separators for
 Column heading?         ===> YES      Break summary?          ===> YES
 Across heading?         ===> YES      Final summary?          ===> YES
```

Figure 107. Specify the number of columns you want to remain fixed.

2. Move the cursor to the **Number of fixed columns in report?** field.
3. Type the number of columns you want to remain fixed. For this example, you want the ID and NAME columns to display at all times. Type 2 in the **Number of fixed columns in report?** field. Press the Report function key to see the changed report. Press the Right function key to display the rest of the information. The columns you specified as fixed remain displayed on the screen.



```
      ID  NAME     |EPTNUMB  DEPTNAME        MANAGER  DIVISION    LOCATION
  ------  ---------|-------  --------------  -------  ----------  -------------
      10  SANDERS  |     20  MID ATLANTIC         10  EASTERN     WASHINGTON
      20  PERNAL   |     20  MID ATLANTIC         10  EASTERN     WASHINGTON
      30  MARENGHI |     38  SOUTH ATLANTIC       30  EASTERN     ATLANTA
      40  O'BRIEN  |     38  SOUTH ATLANTIC       30  EASTERN     ATLANTA
      50  HANES    |     15  NEW ENGLAND          50  EASTERN     BOSTON
      60  QUIGLEY  |     38  SOUTH ATLANTIC       30  EASTERN     ATLANTA
      70  ROTHMAN  |     15  NEW ENGLAND          50  EASTERN     BOSTON
      80  JAMES    |     20  MID ATLANTIC         10  EASTERN     WASHINGTON
      90  KOONITZ  |     42  GREAT LAKES         100  MIDWEST     CHICAGO
     100  PLOTZ    |     42  GREAT LAKES         100  MIDWEST     CHICAGO
     110  NGAN     |     15  NEW ENGLAND          50  EASTERN     BOSTON
     120  NAUGHTON |     38  SOUTH ATLANTIC       30  EASTERN     ATLANTA
     130  YAMAGUCHI|     42  GREAT LAKES         100  MIDWEST     CHICAGO
     140  FRAYE    |     51  PLAINS              140  MIDWEST     DALLAS
```

Figure 108. The first two columns remain fixed when you press the Right function key.

Displaying a representative report before you select data

Before you select any data, you can use the LAYOUT command to display a representation of the report a form will produce.

Displaying a representation is helpful if you want to test or change a form without running a query. Also, you can display a representation to help you remember the report produced by a particular form.

Your installation might not support use of the LAYOUT command for the following reasons:

- ISPF is not available when you run QMF.
- Your QMF administrator has not made the default command synonyms available.

Check with your QMF administrator before you try to use this command.

On a report representation, data displays as either a string of letters (character data) or a string of numbers (numeric data).

To display a representative report for a form saved in the database: On the QMF command line, enter:

```
LAYOUT FORM formname
```

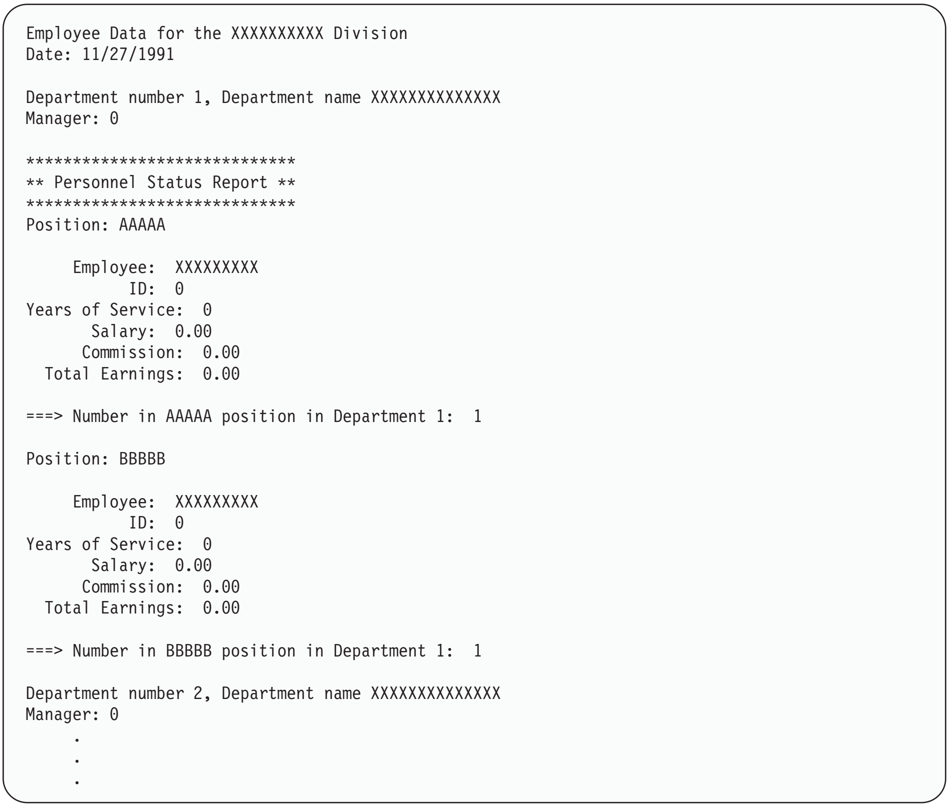
For example, for a saved form that is named FORM5, enter:

```
LAYOUT FORM FORM5
```

To display a representative report for a form in temporary storage: Enter:

```
LAYOUT FORM
```

Figure 109 on page 136 shows an example of a representative report.



```
Employee Data for the XXXXXXXXXX Division
Date: 11/27/1991

Department number 1, Department name XXXXXXXXXXXXXX
Manager: 0

*****************************
** Personnel Status Report **
*****************************
Position: AAAAA

     Employee:  XXXXXXXXX
           ID:  0
Years of Service:  0
       Salary:  0.00
      Commission:  0.00
  Total Earnings:  0.00

===> Number in AAAAA position in Department 1:  1

Position: BBBBB

     Employee:  XXXXXXXXX
           ID:  0
Years of Service:  0
       Salary:  0.00
      Commission:  0.00
  Total Earnings:  0.00

===> Number in BBBBB position in Department 1:  1

Department number 2, Department name XXXXXXXXXXXXXX
Manager: 0
     .
     .
     .
```

Figure 109. A representative report shows how data will be displayed.

The first control break, on the DEPT column (a numeric column), is represented by a **1** for the first department and a **2** for the second department. The second control break, on the JOB column (a character column), is represented by **AAAAA** for the first job title and **BBBBB** for the second job title.

Refining page headings and footings

In addition to specifying heading and footing text, you can:

- Include form variables, such as &DATE and &TIME, and global variables in the heading and footing text
- Control the placement of the page heading and footing text
- Indicate the number of blank lines to appear before and after the page heading and footing text

Use the FORM.PAGE panel to refine heading and footing text.

Using a global variable in a heading or footing

This example uses the SQL query in Figure 110. The query selects and joins columns from the Q.STAFF and Q.ORG tables.

```
SELECT ID, NAME, DEPT, JOB, YEARS, SALARY,
       COMM, DEPTNUMB, DEPTNAME, MANAGER,
       DIVISION, LOCATION
  FROM Q.STAFF, Q.ORG
  WHERE DEPT=DEPTNUMB
    AND DIVISION = &DIVISION
  ORDER BY DEPT, JOB, LOCATION
```

Figure 110. This query joins columns from the Q.STAFF and Q.ORG tables.

In addition, you will use a global variable to specify the division. Global variables allow you to save a QMF object and use it multiple times for different purposes without having to change it.

For more information on using global variables in queries, see *DB2 QMF Reference*.

To set a global variable:

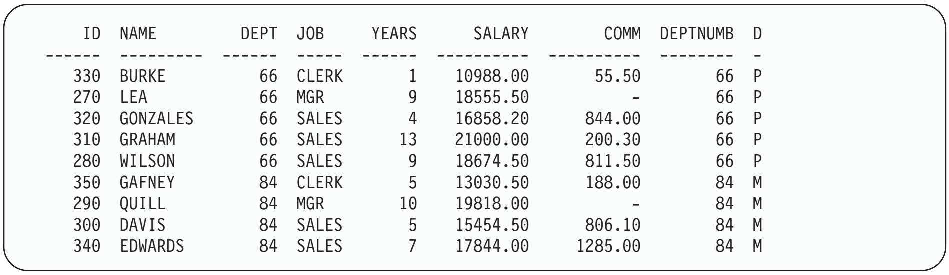
1. On the QMF command line, enter:

   `SET GLOBAL (`*`varname=value`*

   For this example, enter:

   `SET GLOBAL (DIVISION = '''WESTERN'''`

   You must reset the global variable using the SET GLOBAL command each time you start a new QMF session. If you do not set global variables before you run your query, QMF displays a panel that prompts you to enter values for the variables.
2. Run the query to display the default report. Figure 111 shows the report for this example.



```
    ID  NAME        DEPT  JOB    YEARS      SALARY        COMM  DEPTNUMB  D
------  ---------  ------  -----  ------  ----------  ----------  --------  -
   330  BURKE         66  CLERK       1    10988.00       55.50        66  P
   270  LEA           66  MGR         9    18555.50           -        66  P
   320  GONZALES      66  SALES       4    16858.20      844.00        66  P
   310  GRAHAM        66  SALES      13    21000.00      200.30        66  P
   280  WILSON        66  SALES       9    18674.50      811.50        66  P
   350  GAFNEY        84  CLERK       5    13030.50      188.00        84  M
   290  QUILL         84  MGR        10    19818.00           -        84  M
   300  DAVIS         84  SALES       5    15454.50      806.10        84  M
   340  EDWARDS       84  SALES       7    17844.00     1285.00        84  M
```

Figure 111. The default report has no page headings or footings.

3. On the QMF command line, enter:

   ```
   SHOW FORM.PAGE
   ```

   The FORM.PAGE panel displays.
4. In line 1 of the **PAGE HEADING TEXT** field, change the alignment to LEFT, and type the text you want to display as the page heading.

   For this example, type:

   ```
   EMPLOYEE DATA FOR THE &11 DIVISION
   ```
5. Press the Report function key to see the changed report.

Adding the date, time, and page number to a heading or footing

You can display the date, time, or page number on a page heading or footing by using form variables.

You can use the following variables in a report:

&DATE
Adds the current date to a heading or footing when you run the report

&TIME
Adds the current time to a heading or footing when you run the report

&PAGE
Adds the current page number to a heading or footing when you run the report

When the date, time, or page number is displayed in a page heading or footing, it is not displayed at the bottom of the page of the printed report.

To add the date, time, or page number: You can also specify text before or after the form variable. In this example, to add the date to the second line of the report heading, change the alignment to LEFT, and type `Date: &DATE` in the **PAGE HEADING TEXT** field on the second line of the page heading. For more information on form variables, see *DB2 QMF Reference*.

Changing the alignment of page headings and footings

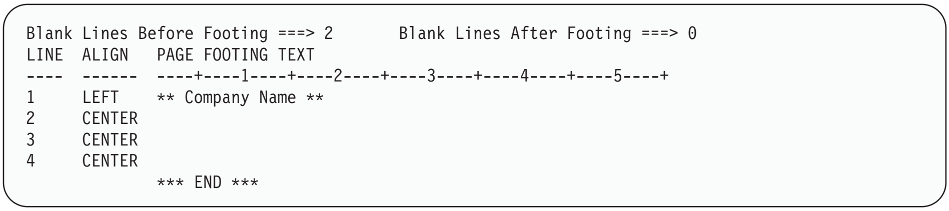
The default alignment for page headings and footings is centered (CENTER), but you can change the alignment on the FORM.PAGE panel.

In this example, change the alignment of the page footing to the left margin of the report.

To change the alignment of a page heading or footing:

1. Move the cursor to the **ALIGN** field for the line you want to change.

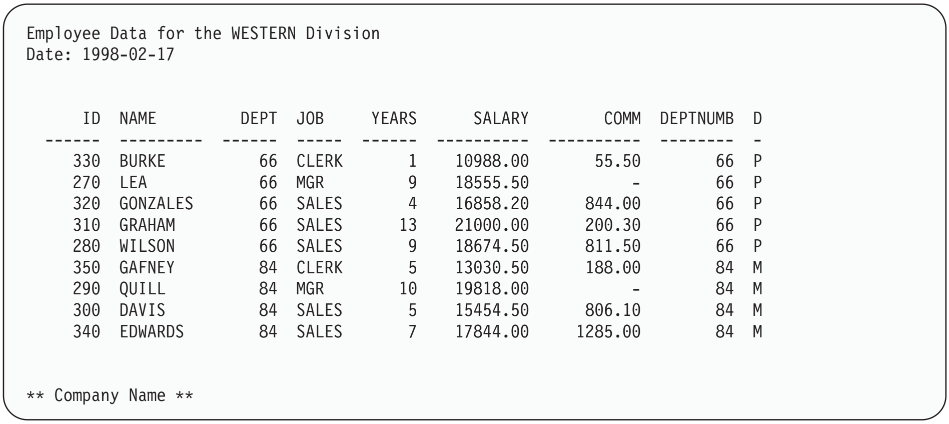
2. Type the new alignment value. For this example, change the alignment for line 1 of the page footing to `LEFT`, and type `** Company Name **` as the footing text.



```
Blank Lines Before Footing ===> 2       Blank Lines After Footing ===> 0
LINE  ALIGN   PAGE FOOTING TEXT
----  ------  ----+----1----+----2----+----3----+----4----+----5----+
1     LEFT    ** Company Name **
2     CENTER
3     CENTER
4     CENTER
              *** END ***
```

Figure 112. Change the alignment of page headings and footings on FORM.PAGE.

3. Press the Report function key to see the changed report.



```
Employee Data for the WESTERN Division
Date: 1998-02-17


      ID  NAME         DEPT  JOB     YEARS      SALARY        COMM  DEPTNUMB  D
  ------  ---------  ------  -----  ------  ----------  ----------  --------  -
     330  BURKE          66  CLERK       1    10988.00       55.50        66  P
     270  LEA            66  MGR         9    18555.50           -        66  P
     320  GONZALES       66  SALES       4    16858.20      844.00        66  P
     310  GRAHAM         66  SALES      13    21000.00      200.30        66  P
     280  WILSON         66  SALES       9    18674.50      811.50        66  P
     350  GAFNEY         84  CLERK       5    13030.50      188.00        84  M
     290  QUILL          84  MGR        10    19818.00           -        84  M
     300  DAVIS          84  SALES       5    15454.50      806.10        84  M
     340  EDWARDS        84  SALES       7    17844.00     1285.00        84  M


** Company Name **
```

Figure 113. The changed report shows the headings and footings aligned left.

Adding break segments and text to your report

You can add break segments to your report by specifying BREAKn usage codes for columns.

In addition, you can use the six FORM.BREAKn panels to do the following:

- Specify the break heading text lines and footing text lines for your report.
- Control the placement of the break heading and footing text.
- Indicate the number of blank lines that appear before and after the break heading and footing.
- Specify whether you want a new page at each break or footing.
- Specify whether you want to repeat the detail heading after the break heading.
- Use form or global variables to place additional information at breaks.
- Place the break summary at a specific line.

- Place the results of calculation expressions that are specified on FORM.CALC in the BREAK footings of your report.

  See “Calculating values to use in a report” on page 149 for more information on specifying calculation expressions.

Adding break heading and footing text to a report

You can add heading and footing text for each break segment in your report by using the FORM.BREAK panels.

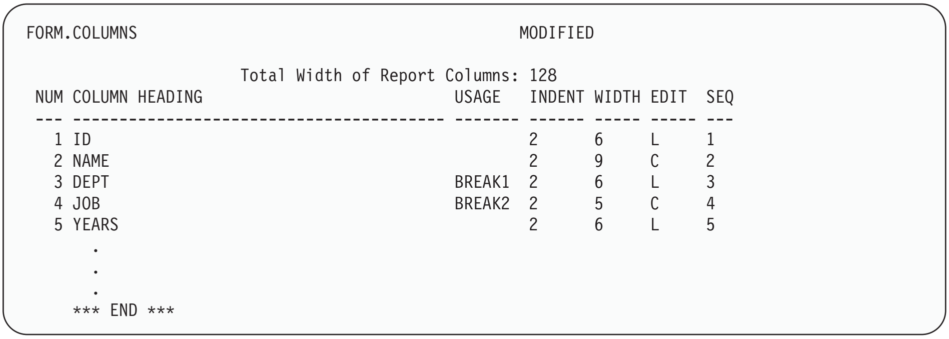
In the following example, you add break heading text for BREAK1 and break heading and footing text for BREAK2.

To add break text:

1. On the QMF command line, enter:

   ```
   SHOW FORM.COLUMNS
   ```

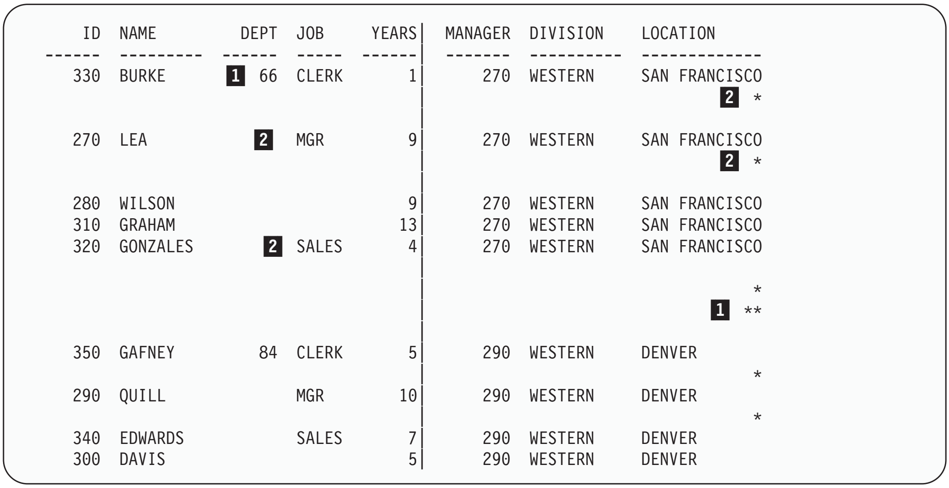
   The FORM.COLUMNS panel displays.
2. In the **USAGE** field, specify up to six breaks for the columns in the report. For this example, type `BREAK1` for DEPT and `BREAK2` for JOB.



```
FORM.COLUMNS                                         MODIFIED

                       Total Width of Report Columns: 128
 NUM COLUMN HEADING                           USAGE   INDENT WIDTH EDIT  SEQ
 --- ---------------------------------------- ------- ------ ----- ----- ---
   1 ID                                               2      6     L     1
   2 NAME                                             2      9     C     2
   3 DEPT                                     BREAK1  2      6     L     3
   4 JOB                                      BREAK2  2      5     C     4
   5 YEARS                                            2      6     L     5
       .
       .
       .
     *** END ***
```

Figure 114. Specify breaks at the DEPT and JOB columns.

3. Press the Report function key to see the changed report.



```
    ID  NAME         DEPT  JOB     YEARS|  MANAGER  DIVISION    LOCATION
------  ---------  ------  -----  ------|  -------  ----------  -------------
   330  BURKE       1  66  CLERK       1|      270  WESTERN     SAN FRANCISCO
                                        |                                2  *
                                        |
   270  LEA             2  MGR         9|      270  WESTERN     SAN FRANCISCO
                                        |                                2  *
                                        |
   280  WILSON                         9|      270  WESTERN     SAN FRANCISCO
   310  GRAHAM                        13|      270  WESTERN     SAN FRANCISCO
   320  GONZALES         2  SALES       4|      270  WESTERN     SAN FRANCISCO
                                        |
                                        |                                   *
                                        |                               1  **
                                        |
   350  GAFNEY         84  CLERK       5|      290  WESTERN     DENVER
                                        |                                   *
   290  QUILL              MGR        10|      290  WESTERN     DENVER
                                        |                                   *
   340  EDWARDS            SALES       7|      290  WESTERN     DENVER
   300  DAVIS                          5|      290  WESTERN     DENVER
```

Figure 115. The changed report shows breaks after each department and job.

1	Shows a first-level break
2	Shows a second-level break

4. On the QMF command line, enter:

   ```
   SHOW FORM.BREAKn
   ```

   Where *n* is the break level. For this example, enter:

   ```
   SHOW FORM.BREAK1
   ```

   The FORM.BREAK panel for the break level you specified displays. You specify break heading and footing text on this panel.
5. Leave the defaults, or type new values for the **New Page for Break**, **Blank Lines Before Heading**, **Repeat Detail Heading?**, and **Blank Lines After Heading** fields.

   For this example, type 2 for **Blank Lines After Heading**.
6. Type the break text for each line under the **BREAK 1 HEADING TEXT** heading. You can use form variables in the text.

   In this example, type `Department number &3, Department name &9` for the first line of text, and `Manager:  &10` for the second line of text.

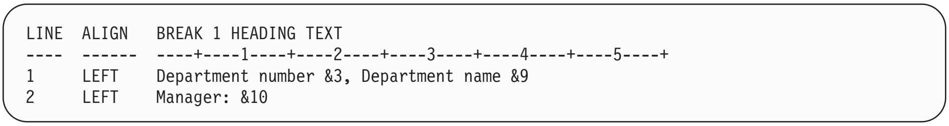
   Line 1 contains the form variable &3 for department number, and &9 for department name. Line 2 contains the form variable &10 for the ID number of the manager for the department.

   You can specify either left or right justification of the text, or align the text at a specific column.

   If all the break text for one line does not fit in the space allowed on that line, type `APPEND` in the ALIGN column of the next line. Then, change its line number to the same number as the preceding line, and continue typing the text. Be sure to indent the appended text at least one space.

If you need another line, move the cursor to the line above the place you want the new line and press the Insert function key. You can specify up to 999 lines of text.

For this example, leave the default of `LEFT`.



```
LINE  ALIGN   BREAK 1 HEADING TEXT
----  ------  ----+----1----+----2----+----3----+----4----+----5----+
1     LEFT    Department number &3, Department name &9
2     LEFT    Manager: &10
```

Figure 116. Specify headings and footings for report breaks on FORM.BREAK.

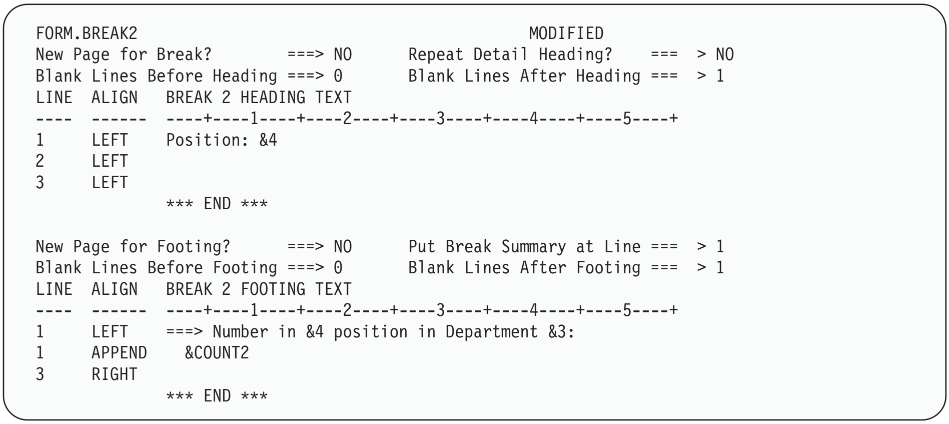
7. Specify text for the break footing in the same way you specify text for the break heading.

   For this example, do not specify footing text for the level-one break.

   For the second break, on the JOB column, enter both heading and footing text.
8. On the QMF command line, enter:

   `SHOW FORM.BREAK2`
9. For this example, type `Position: &4` for the first line of heading text.
10. Change the alignment to**LEFT** and type `===> Number in &4 position in Department &3:` for the first line of footing text.
11. Change the line number in the **LINE** field for the second line of footing text from 2 to 1.
12. Type `APPEND` in the **ALIGN** field for the second line of footing text.
13. Type `&COUNT2` in the **BREAK 2 FOOTING TEXT** field. Be sure to leave a space at the beginning of the second line of text. These two lines appear as one line on the report. The variable &COUNT2 is an aggregation variable that counts the number of values for column 2, NAME.

    Figure 117 on page 143 shows the completed FORM.BREAK2 panel for this example.

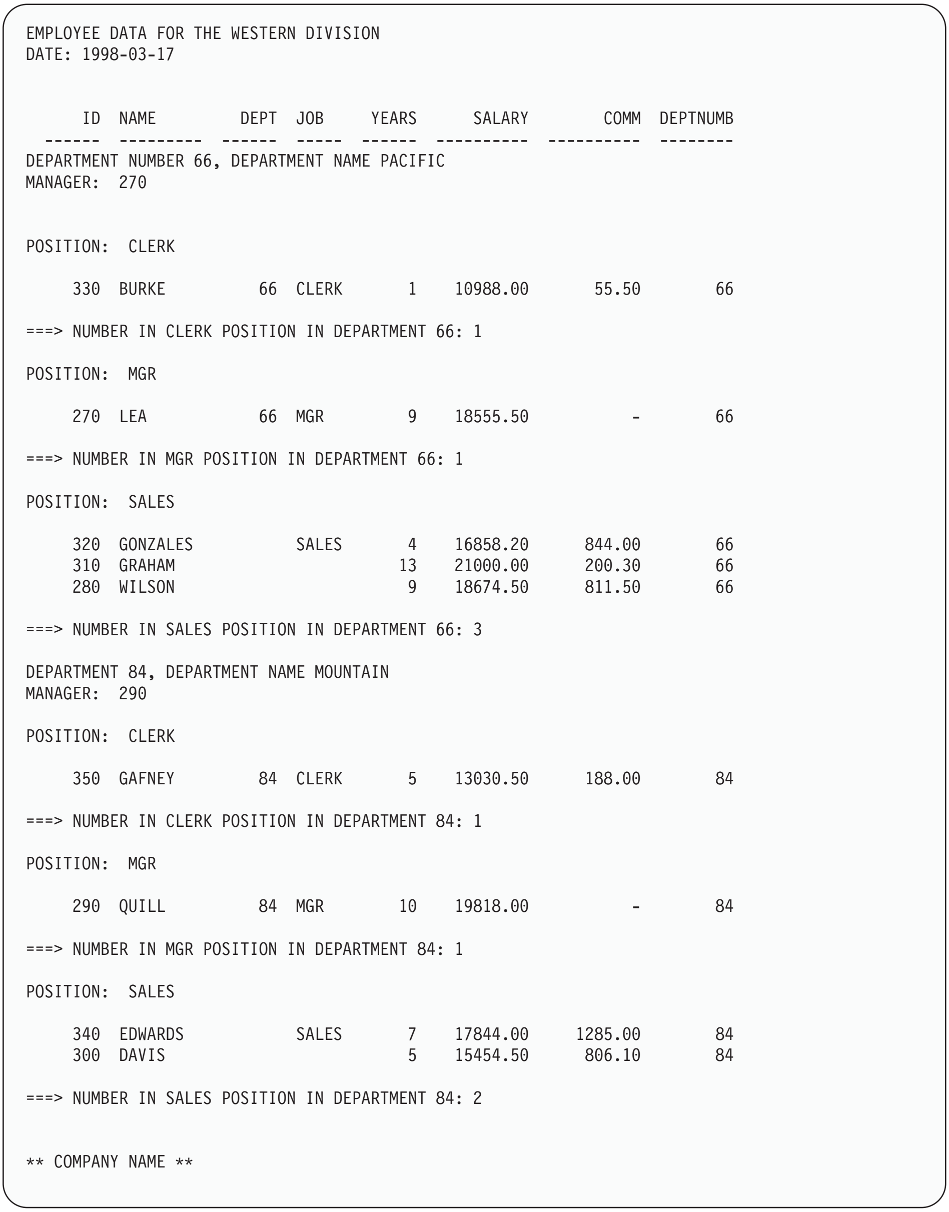


```
FORM.BREAK2                                           MODIFIED
New Page for Break?        ===> NO      Repeat Detail Heading?    ===  > NO
Blank Lines Before Heading ===> 0       Blank Lines After Heading ===  > 1
LINE  ALIGN   BREAK 2 HEADING TEXT
----  ------  ----+----1----+----2----+----3----+----4----+----5----+
1     LEFT    Position: &4
2     LEFT
3     LEFT
              *** END ***

New Page for Footing?      ===> NO      Put Break Summary at Line ===  > 1
Blank Lines Before Footing ===> 0       Blank Lines After Footing ===  > 1
LINE  ALIGN   BREAK 2 FOOTING TEXT
----  ------  ----+----1----+----2----+----3----+----4----+----5----+
1     LEFT    ===> Number in &4 position in Department &3:
1     APPEND    &COUNT2
3     RIGHT
              *** END ***
```

Figure 117. Specify text for the second break on FORM.BREAK2.

For more information on all the fields on the BREAK panels, see *DB2 QMF Reference*.

14. Press the Report function key to see the changed report.Figure 118 on page 144 shows how the changed report should look.



```
EMPLOYEE DATA FOR THE WESTERN DIVISION
DATE: 1998-03-17


      ID  NAME         DEPT  JOB     YEARS      SALARY        COMM  DEPTNUMB
 ------  ---------  ------  -----  ------  ----------  ----------  --------
DEPARTMENT NUMBER 66, DEPARTMENT NAME PACIFIC
MANAGER:  270


POSITION:  CLERK

     330  BURKE         66  CLERK      1    10988.00       55.50        66

===> NUMBER IN CLERK POSITION IN DEPARTMENT 66: 1

POSITION:  MGR

     270  LEA           66  MGR        9    18555.50           -        66

===> NUMBER IN MGR POSITION IN DEPARTMENT 66: 1

POSITION:  SALES

     320  GONZALES           SALES      4    16858.20      844.00        66
     310  GRAHAM                       13    21000.00      200.30        66
     280  WILSON                        9    18674.50      811.50        66

===> NUMBER IN SALES POSITION IN DEPARTMENT 66: 3

DEPARTMENT 84, DEPARTMENT NAME MOUNTAIN
MANAGER:  290

POSITION:  CLERK

     350  GAFNEY        84  CLERK      5    13030.50      188.00        84

===> NUMBER IN CLERK POSITION IN DEPARTMENT 84: 1

POSITION:  MGR

     290  QUILL         84  MGR       10    19818.00           -        84

===> NUMBER IN MGR POSITION IN DEPARTMENT 84: 1

POSITION:  SALES

     340  EDWARDS           SALES      7    17844.00     1285.00        84
     300  DAVIS                         5    15454.50      806.10        84

===> NUMBER IN SALES POSITION IN DEPARTMENT 84: 2


** COMPANY NAME **
```

Figure 118. The changed report shows first and second level break text.

Refining the format of your report with detail blocks

You can reformat and add text to your report with detail blocks. A detail block is a set of specifications that tells QMF to put in whatever special formatting you want for one row of the data retrieved by your query.

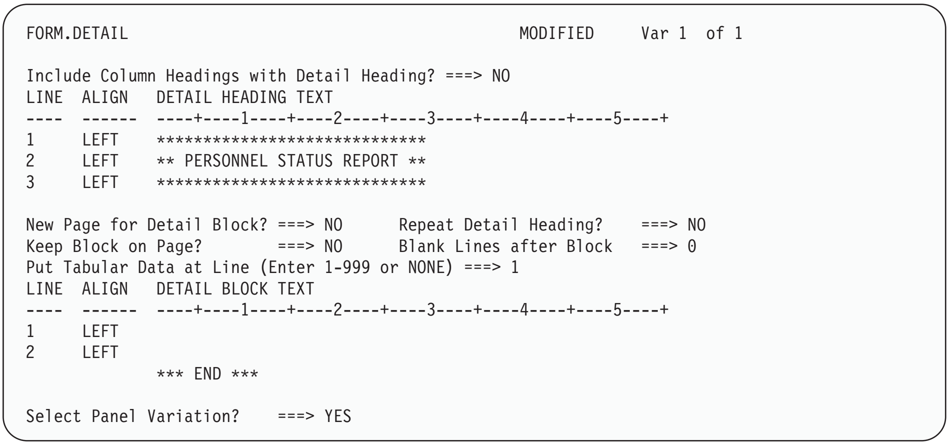
Specify detail blocks on the FORM.DETAIL panel. You can use the panel to do the following:

- Format the detail heading and block text in your report.
- Enter your own text for the detail heading to either replace the column headings or combine it with the column headings.
- Enter your own block of detail text and specify its location anywhere within the detail block of the report.
- Use data from the columns selected in your query by using form column variable names or aggregation variables. Place that data at any location within the detail block.
- Specify the placement of tabular data.
- Mix tabular (column) data with free-flowing text (detail block text)
- Omit tabular data from your report.
- Place the results of calculation expressions, specified on FORM.CALC, in the detail block text of your report. See "Calculating values to use in a report" on page 149 for an example of how to specify calculation expressions.

In the following example, you use the FORM.DETAIL panel to reformat the report in Figure 118 on page 144. You will also remove the column headings from the report, omit tabular data (columns), and add headings for each subsection. When you finish, the report will look like Figure 121 on page 147.

To specify detail blocks:

1. On the QMF command line, enter:
   ```
   DISPLAY FORM.DETAIL
   ```
   The FORM.DETAIL panel displays.



```
FORM.DETAIL                                          MODIFIED     Var 1  of 1

Include Column Headings with Detail Heading? ===> NO
LINE  ALIGN   DETAIL HEADING TEXT
----  ------  ----+----1----+----2----+----3----+----4----+----5----+
1     LEFT    *****************************
2     LEFT    ** PERSONNEL STATUS REPORT **
3     LEFT    *****************************

New Page for Detail Block? ===> NO      Repeat Detail Heading?    ===> NO
Keep Block on Page?        ===> NO      Blank Lines after Block   ===> 0
Put Tabular Data at Line (Enter 1-999 or NONE) ===> 1
LINE  ALIGN   DETAIL BLOCK TEXT
----  ------  ----+----1----+----2----+----3----+----4----+----5----+
1     LEFT
2     LEFT
              *** END ***

Select Panel Variation?    ===> YES
```

Figure 119. Specify text for the detail heading on FORM.DETAIL.

2. Because the final report is not in tabular format (columns and rows), you do not want to display the column headings. Type NO in the **Include Column Headings with Detail Heading?** field for this example.

   If you leave YES, the column headings display immediately after any detail heading text in the report.

3. Type the detail heading text for each line in the **DETAIL HEADING TEXT** field. You can use form variables in the text.

   For this example, type the text as it is shown on the FORM.DETAIL panel in Figure 119. You need to insert an extra line and change its line number.

   Next, rearrange the data from each column into a vertical list, rather than a tabular arrangement of columns and rows. Give each column value a new label to replace the column headings. Specify the format of the detail block on the bottom half of the FORM.DETAIL panel.

4. Leave the defaults, or type new values for the **New Page for Detail Block?**, **Repeat Detail Heading?**, **Keep Block on Page?**, and **Blank Lines After Block** fields.

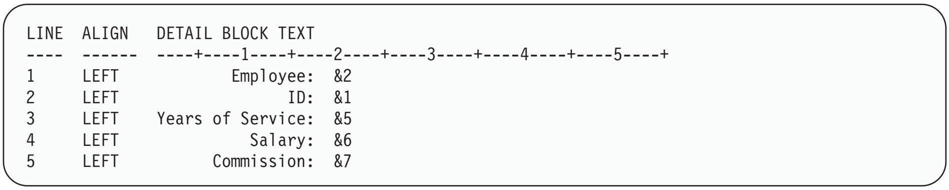
   For this example, type 1 for **Blank Lines After Block**.

5. Type a line number if you want to include tabular data in the report, or type NONE if you want to remove all the tabular data.

   For this example, type NONE, to remove all the tabular data from the report.

6. Type the detail block text for each line in the **DETAIL BLOCK TEXT** field. Use form variables and text to provide values for the column headings and column data on the report.

   For this example, type the text as it is shown on the following FORM.DETAIL panel.

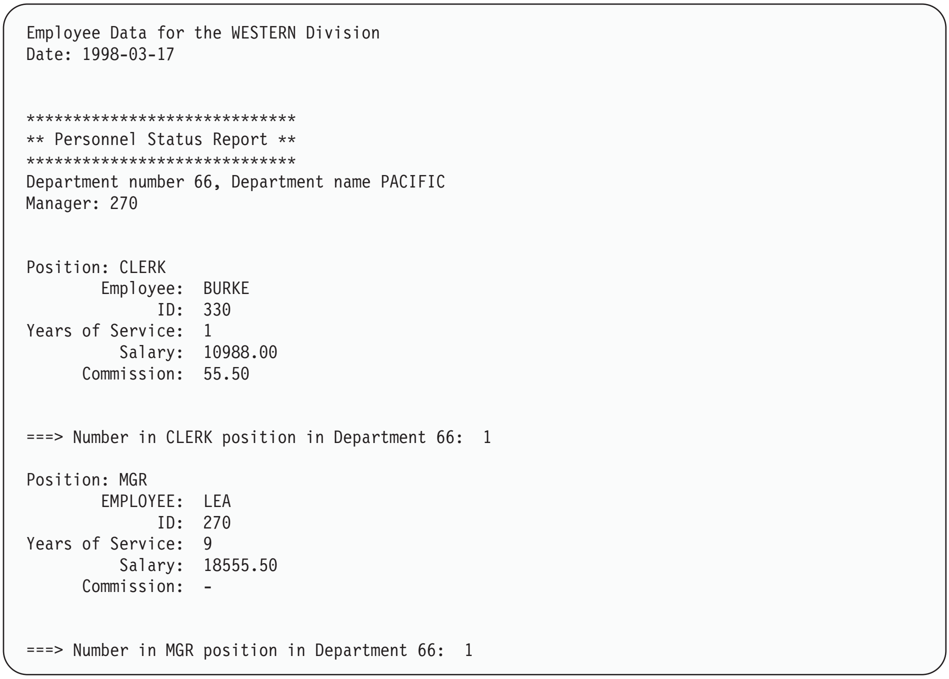


```
LINE  ALIGN   DETAIL BLOCK TEXT
----  ------  ----+----1----+----2----+----3----+----4----+----5----+
1     LEFT            Employee:  &2
2     LEFT                  ID:  &1
3     LEFT    Years of Service:  &5
4     LEFT              Salary:  &6
5     LEFT          Commission:  &7
```

Figure 120. Specify replacement column headings with detail block text.

You can create variations of the detail blocks to use with different conditions in a report format with panel variations. For more information on creating panel variations, see *DB2 QMF Reference*.

7. Press the Report function key to see the changed report.



```
Employee Data for the WESTERN Division
Date: 1998-03-17


*****************************
** Personnel Status Report **
*****************************
Department number 66, Department name PACIFIC
Manager: 270


Position: CLERK
        Employee:  BURKE
              ID:  330
Years of Service:  1
          Salary:  10988.00
      Commission:  55.50


===> Number in CLERK position in Department 66:  1

Position: MGR
        EMPLOYEE:  LEA
              ID:  270
Years of Service:  9
          Salary:  18555.50
      Commission:  -


===> Number in MGR position in Department 66:  1
```

Figure 121. The changed report shows the results of reformatting.

Specifying text to appear at the end of your report

You can place text at the end of your report for any purpose you want. For example, you can explain items in the report, or you can include information that summarizes the data, such as totals or averages. Specify the final text for your report on the FORM.FINAL panel.

Use the FORM.FINAL panel to do the following:

- Specify final text for your report.
- Control the position of the final text in the report.

- Specify that the final text begins on a new page.
- Specify the number of blank lines that appear before the text.
- Specify the line number at which the final summary begins.
- Place the results of calculation expressions, specified on FORM.CALC, in your report final text.

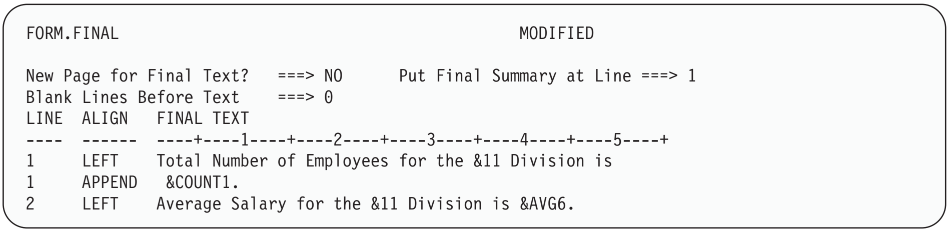
For this example, add final text to the report that shows the total number of employees for the Western Division and the average of their salaries.

To specify final text:

1. On the QMF command line, enter:

   `SHOW FORM.FINAL`

   The FORM.FINAL panel displays.



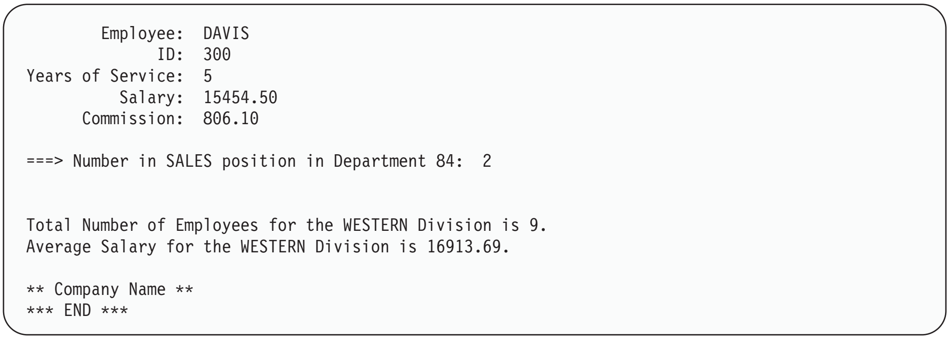
```
FORM.FINAL                                          MODIFIED

New Page for Final Text?   ===> NO      Put Final Summary at Line ===> 1
Blank Lines Before Text    ===> 0
LINE  ALIGN   FINAL TEXT
----  ------  ----+----1----+----2----+----3----+----4----+----5----+
1     LEFT    Total Number of Employees for the &11 Division is
1     APPEND   &COUNT1.
2     LEFT    Average Salary for the &11 Division is &AVG6.
```

Figure 122. Specify text for the end of a report on FORM.FINAL.

2. Leave the defaults, or type new values for the **New Page for Final Text?**, **Put Final Summary at Line**, and **Blank Lines Before Text** fields.

   For this example, leave the defaults for these fields.
3. Type the final text for each line under the **FINAL TEXT** field. You can use form variables in the text.

   For this example, change the alignment to **LEFT**, and type `Total Number of Employees for the &11 Division is` for the first line. Specify APPEND for alignment for the next line, change the line number to 1, and type `&COUNT1`. Be sure to leave a space before `&COUNT1`. For the next line of final text, change the alignment to **LEFT**, and type `Average Salary for the &11 Division is &AVG6`.
4. Press the Report function key to display the changed report.



```
        Employee:  DAVIS
              ID:  300
Years of Service:  5
          Salary:  15454.50
      Commission:  806.10

===> Number in SALES position in Department 84:  2


Total Number of Employees for the WESTERN Division is 9.
Average Salary for the WESTERN Division is 16913.69.

** Company Name **
*** END ***
```

Figure 123. The final text displays at the end of your report.

Calculating values to use in a report

Note to CICS Users

You cannot calculate values to use in a report in CICS.

The values in your reports can come from the following:

- Data that you import or have stored in the database
- Calculations that are performed in a query
- Calculations that are performed within a QMF form

You can specify calculations in a form that are similar to the calculations that are performed in a query. QMF evaluates calculations in a form by using the REXX language. Calculations can take advantage of all the built-in REXX functions. You can also specify REXX execs you write in a form. However, calculations in a form can affect performance.

You can create calculations to use in a report in one of the following ways:

- Define an expression that calculates a value.
- Create a REXX exec to return a value.

DB2 QMF Reference describes expressions in detail. For more information on REXX execs, see *OS/390 V2R10.0 TSO/E REXX Reference* (for TSO) or *VM System Product Interpreter Reference* (for CMS). Your installation might not support use of calculations and REXX functions. Check with your QMF administrator before you try to use calculations in a report.

Displaying a calculated value on a report

You can display a calculated value in detail block text, break footing text, and final text on a report.

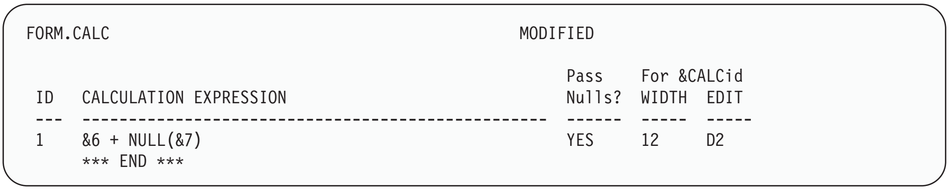
In this example, you will define an expression that adds an employee's salary and commission. This expression is similar to the one used in a query in "Creating a column using expressions" on page 40. Then, you will display the result in the detail block text on the Personnel Status Report you created and changed in previous examples.

To display a calculated value:

1. On the QMF command line, enter:

   ```
   SHOW FORM.CALC
   ```

   The FORM.CALC panel displays.



```
FORM.CALC                                          MODIFIED

                                                        Pass    For &CALCid
 ID   CALCULATION EXPRESSION                            Nulls?  WIDTH  EDIT
 ---  ------------------------------------------------  ------  -----  -----
 1    &6 + NULL(&7)                                     YES     12     D2
      *** END ***
```

Figure 124. Specify an expression to calculate a value on FORM.CALC.

2. Type an ID number for the expression. You can use any number from 1 through 999.

   In this example, type 1 for ID.
3. Type the expression, using form variables to specify the columns, in the **CALCULATION EXPRESSION** field.

   In this example, type `&6 + NULL(&7)`, which means to add the values in columns 6 (SALARY) and 7 (COMM).

   Because some of the commission values in the sample tables are null, they appear as a hyphen in the report. REXX cannot perform an arithmetic operation on data that contains both numeric values and nulls. The REXX NULL EXEC looks for the nulls in the data and replaces them with a specified value. In this case, it replaces nulls with zeros.

   When you write a REXX exec, make sure that you make it available to QMF by placing it on an accessible disk or specifying the correct data set. Here is the NULL EXEC for this example:

   ```
   /* REXX EXEC to substitute 0 in place of nulls */
   parse arg in1
     if in1 = "DSQNULL" then
       value = 0
     else value = in1
   return value
   ```

4. In the **PASS NULLS** field, type YES to process nulls for this example.
5. In the **WIDTH** field, type 12 to accommodate the number of characters expected in the result of the calculation.
6. In the **EDIT** field, type the edit code for the result of this calculation. Because you want to display total earnings as a dollar value, type D2.

   See "Specifying punctuation for the values in a column" on page 123 for information about edit codes.

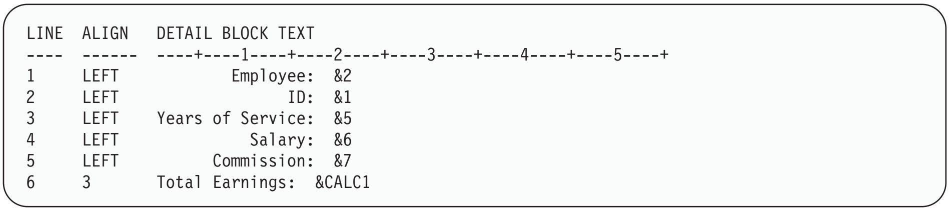
   After you define the expression, you can use the FORM.DETAIL panel to define how you want to display the result of the calculation on the report.
7. On the QMF command line, enter:

   ```
   SHOW FORM.DETAIL
   ```

   The FORM.DETAIL panel displays.
8. Type the new line of text in the **DETAIL BLOCK TEXT** field.

   For this example, type Total Earnings:&CALC1. The variable &CALC1 corresponds to the calculation expression you created on the FORM.CALC panel.

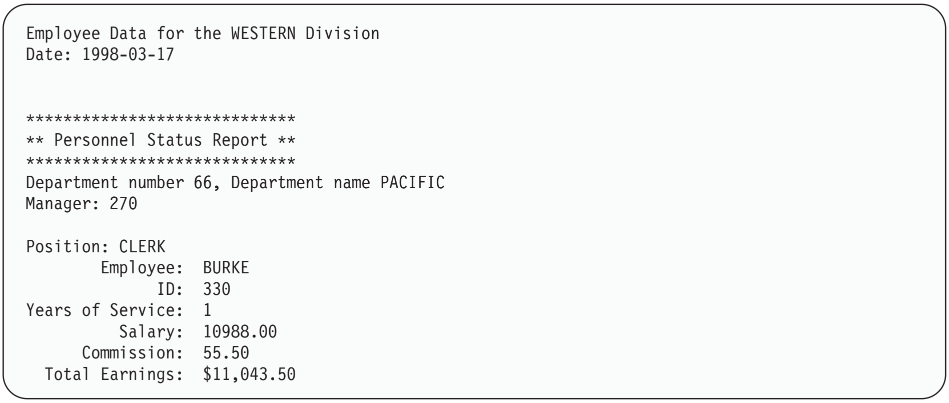
   Change the line number and alignment for the new line of text. For this example, change the line number to 6 and change the alignment to 3. This means that you want this line of detail block text to begin in column 3.



```
LINE  ALIGN   DETAIL BLOCK TEXT
----  ------  ----+----1----+----2----+----3----+----4----+----5----+
1     LEFT            Employee:  &2
2     LEFT                  ID:  &1
3     LEFT    Years of Service:  &5
4     LEFT              Salary:  &6
5     LEFT          Commission:  &7
6     3       Total Earnings:  &CALC1
```

Figure 125. Specify where a calculated value appears with detail block text.

9. Press the Report function key to see the changed report.



```
Employee Data for the WESTERN Division
Date: 1998-03-17


*****************************
** Personnel Status Report **
*****************************
Department number 66, Department name PACIFIC
Manager: 270

Position: CLERK
        Employee:  BURKE
              ID:  330
Years of Service:  1
          Salary:  10988.00
      Commission:  55.50
  Total Earnings:  $11,043.50
```

Figure 126. The calculated value appears next to Total Earnings in the report.

In this example, the value for **Total Earnings** comes from the &CALC1 variable. You can also define **Total Earnings** as a new column by using the same expression and REXX exec. Then, you can specify the value on the FORM.DETAIL panel by using the form variable &*n*, where *n* is the column number given to the new column.

For information on defining a column, see "Adding a new column to a report" on page 117.

Displaying special conditions on your report

You can also define a calculation that identifies a special condition on your report by using either an expression or a REXX exec.

In this example, you will use calculated values to identify two special conditions on a report, one to identify employees who deserve a commission bonus, and one to identify employees who need a raise.

For more information on REXX execs, see *OS/390 V2R10.0 TSO/E REXX Reference* (for TSO) or *VM System Product Interpreter Reference* (for CMS). Your installation might not support use of calculations and REXX functions. Check with your QMF administrator before you try to use calculations in a report.

Identifying a special condition using a REXX exec

In the first part of this example, you create a condition that prints the text `*** Commission Bonus ***` on the Personnel Status Report for all employees with a commission greater than or equal to $800.00.

Because you want to be able to specify a different commission amount to qualify for the bonus each time you run the report, write a REXX exec that allows you to specify the commission amount when you display the report.

To use a REXX program to identify a special condition:

1. On the QMF command line, enter:

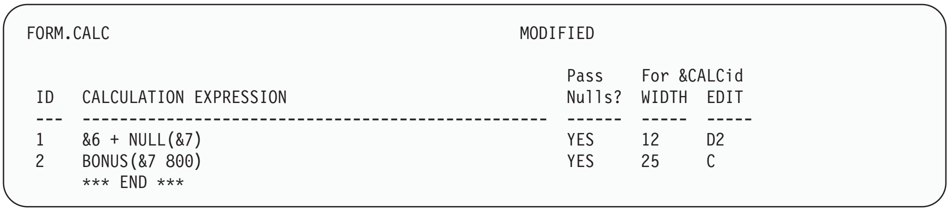
   ```
   SHOW FORM.CALC
   ```

   The FORM.CALC panel displays.
2. In the **ID** field, type an ID number for the expression. You can use any number from 1 through 999. Because you already have an expression from a previous example, type 2.
3. In the **CALCULATION EXPRESSION** field, type the expression, using form variables to specify the columns.

   For this example, type BONUS(&7 800). BONUS is a REXX exec you write that checks the value in the COMMISSION column (&7) to see if it is greater than or equal to the amount you specify for the commission bonus (800). If the value in the column qualifies for the commission bonus, the words *** Commission Bonus *** display on the report.

   Here is the BONUS program for this example:

   ```
   /* REXX BONUS */
   /* program to flag employees whose commission levels warrant a bonus */

   parse arg commission commission_level
   retvalue = ' '
   if (commission ¬= "DSQNULL") & (commission >= commission_level) then
   retvalue = '*** Commission Bonus ***'
   return retvalue
   ```
4. In the **WIDTH** field, type 24 to accommodate the number of characters in the text string *** **Commission Bonus** ***.
5. In the **EDIT** field, type the edit code C to treat the text string as character data.



```
FORM.CALC                                            MODIFIED

                                                          Pass    For &CALCid
 ID   CALCULATION EXPRESSION                              Nulls?  WIDTH  EDIT
 ---  ----------------------------------------------     ------  -----  -----
 1    &6 + NULL(&7)                                       YES     12     D2
 2    BONUS(&7 800)                                       YES     25     C
      *** END ***
```

Figure 127. Specify an expression using a REXX EXEC on FORM.CALC.

   Now that you have specified the calculation expression, use detail block text to specify the placement of the text string.

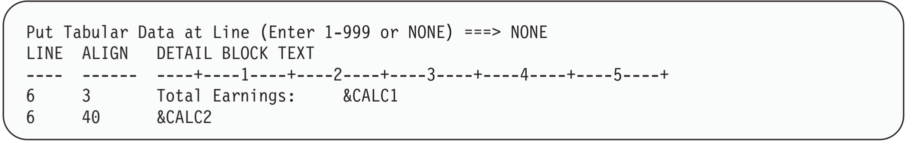
6. On the QMF command line, enter:

   ```
   SHOW FORM.DETAIL
   ```

   The FORM.DETAIL panel displays.

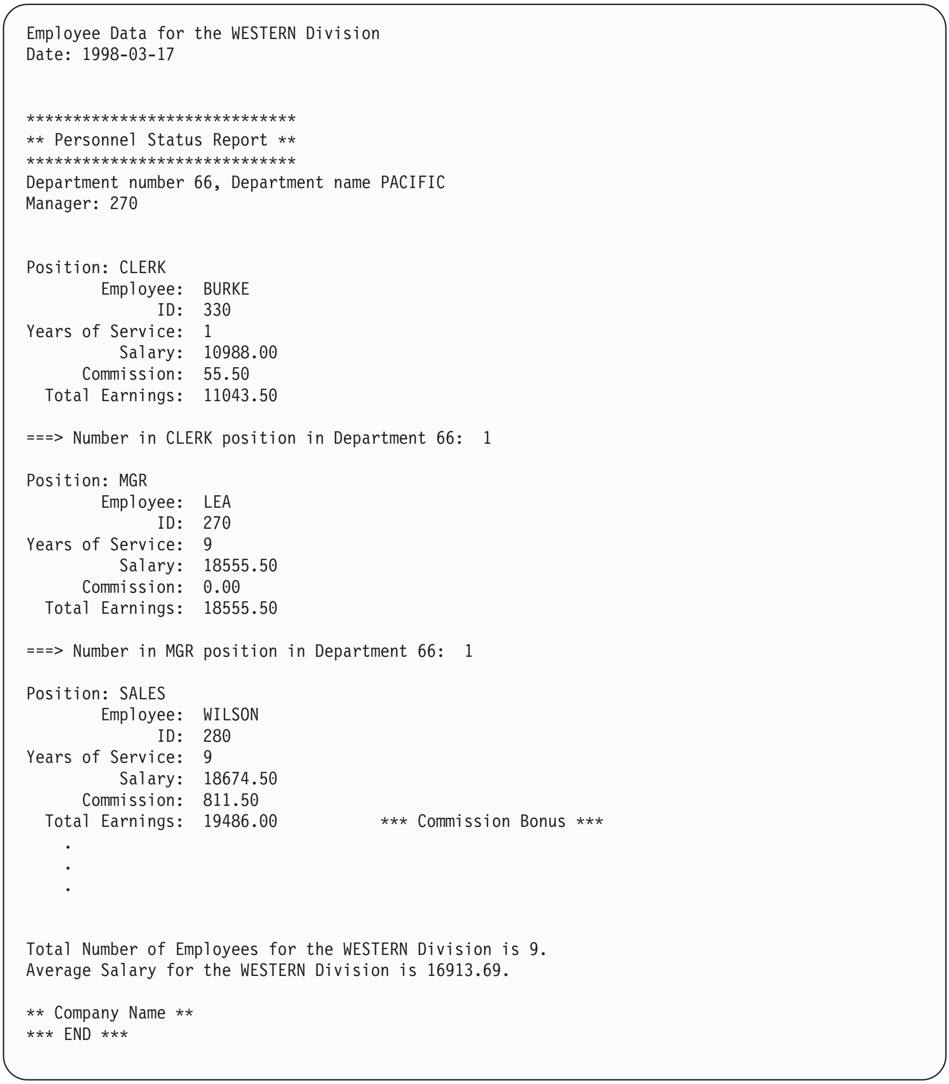
7. Type the information for the detail block text. For this example, you want to display the result of the calculation (&CALC2) on the same line as the total earnings value (6), in column 40.

   If the text you want to display on the report exceeds the report text line width, you can increase the report text line width on the FORM.OPTIONS panel. For information about the fields on the FORM.OPTIONS panel, see *DB2 QMF Reference*.

   ```
   Put Tabular Data at Line (Enter 1-999 or NONE) ===> NONE
   LINE  ALIGN   DETAIL BLOCK TEXT
   ----  ------  ----+----1----+----2----+----3----+----4----+----5----+
   6     3       Total Earnings:     &CALC1
   6     40      &CALC2
   ```

8. Press the Report function key to see the changed report.



```
Employee Data for the WESTERN Division
Date: 1998-03-17


*****************************
** Personnel Status Report **
*****************************
Department number 66, Department name PACIFIC
Manager: 270


Position: CLERK
        Employee:  BURKE
              ID:  330
Years of Service:  1
          Salary:  10988.00
      Commission:  55.50
  Total Earnings:  11043.50

===> Number in CLERK position in Department 66:  1

Position: MGR
        Employee:  LEA
              ID:  270
Years of Service:  9
          Salary:  18555.50
      Commission:  0.00
  Total Earnings:  18555.50

===> Number in MGR position in Department 66:  1

Position: SALES
        Employee:  WILSON
              ID:  280
Years of Service:  9
          Salary:  18674.50
      Commission:  811.50
  Total Earnings:  19486.00           *** Commission Bonus ***
    .
    .
    .


Total Number of Employees for the WESTERN Division is 9.
Average Salary for the WESTERN Division is 16913.69.

** Company Name **
*** END ***
```

Figure 128. The changed report shows the conditional text for bonuses.

Identifying a special condition using an expression

In this example, you create a report that displays some flag text whenever the data meets certain conditions. This example is very similar to the previous example that uses the FORM.CALC panel and a REXX exec. This time, however, you will use the FORM.CONDITIONS panel with the FORM.DETAIL panel. Using this method, you can format your report in different ways depending on conditions you want QMF to test for. You specify an expression for any tests or conditions on the FORM.CONDITIONS panel. Then you associate a FORM.DETAIL panel with each test. When the test evaluates to true for the data in a particular row, QMF formats the report the way you specify on the associated FORM.DETAIL panel. When the test does not evaluate to true, you can specify a different format on another FORM.DETAIL panel.

In this example, you create a condition that prints the text `*** Needs Raise ***` in the report for all employees with salary plus commissions less than $17,000.00.

To use an expression to identify a special condition:

1. Run the query and display the report.

   For this example, use this query:

   ```
   SELECT ID, NAME, JOB, DEPT, SALARY, COMM
     FROM Q.STAFF
   ```
2. On the QMF command line, enter:

   ```
   SHOW FORM.CONDITIONS
   ```

   The FORM.CONDITIONS panel displays.
3. Type an ID number for the expression. You can use any number from 1 through 999.

   For this example, you need to create two conditions; one for employees who earn greater than or equal to $17,000.00, and one for employees who earn less than $17,000.00.

   Type `1` for the first expression, and `2` for the second expression.
4. In the **CONDITIONAL EXPRESSION** field, type the expression, using form variables to specify the columns.

   For this example, type:

   ```
   &5 + NULL(&6) >= 17000.00
   ```

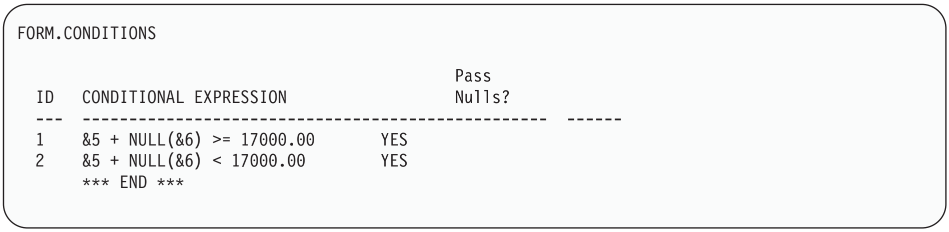
   This is the first condition. It means find each row for which the employee's total earnings (SALARY+COMM) are greater than or equal to $17,000.00.

   Type:

   ```
   &5 + NULL(&6) < 17000.00
   ```

   This is the second condition. It means find each row for which the employee's total earnings are less than $17,000.00.
5. Type `YES` in the **PASS NULLS** field to process nulls for this example.

If you want to process nulls, you can create a REXX exec like NULL to substitute a 0 (or whatever value is appropriate) for the null. Then, you can use that value in the calculation.



```
FORM.CONDITIONS

                                               Pass
  ID   CONDITIONAL EXPRESSION                  Nulls?
  ---  --------------------------------------------------  ------
  1    &5 + NULL(&6) >= 17000.00       YES
  2    &5 + NULL(&6) < 17000.00        YES
       *** END ***
```

Figure 129. Specify conditional expressions for employees' raises.

Now that you have defined the conditions you want to display on the report, use the FORM.DETAIL panel to create a report variation for each condition. This type of report formatting is called conditional formatting.

You can select each report variation unconditionally, or associate it with a condition such as those you just entered on the FORM.CONDITIONS panel.

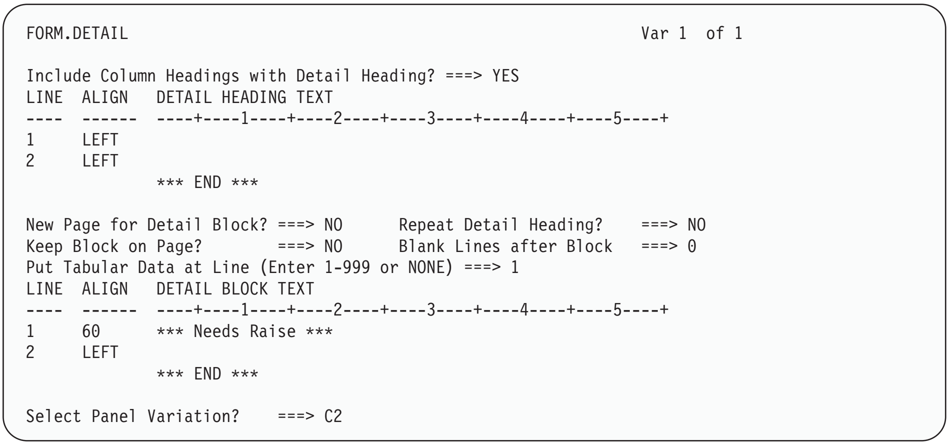
6. On the QMF command line, enter:

   ```
   SHOW FORM.DETAIL
   ```

   The FORM.DETAIL panel displays.
7. In the **Select Panel Variation** field, type `C1` to associate this report format variation with the first condition on the FORM.CONDITIONS panel, where the employee makes greater than or equal to $17,000.00. For this variation, do not enter any detail text.
8. Create a report variation for the second condition. For this example, move the cursor to **Var 1 of 1**.
9. Type `2` over the `1`.
10. Press Enter.

    Or you can enter `NEXT` on the QMF command line.

    A second FORM.DETAIL panel displays.
11. In the **Select Panel Variation** field, type `C2` to associate this report format variation with the second condition on the FORM.CONDITIONS panel, where the employee makes less than $17,000.00.
12. For this example, type `*** Needs Raise ***` for the first line of detail block text. Change the alignment to column 60. QMF displays the text on the report when this condition is true.



```
FORM.DETAIL                                                       Var 1  of 1

Include Column Headings with Detail Heading? ===> YES
LINE  ALIGN   DETAIL HEADING TEXT
----  ------  ----+----1----+----2----+----3----+----4----+----5----+
1     LEFT
2     LEFT
              *** END ***

New Page for Detail Block? ===> NO      Repeat Detail Heading?    ===> NO
Keep Block on Page?        ===> NO      Blank Lines after Block   ===> 0
Put Tabular Data at Line (Enter 1-999 or NONE) ===> 1
LINE  ALIGN   DETAIL BLOCK TEXT
----  ------  ----+----1----+----2----+----3----+----4----+----5----+
1     60      *** Needs Raise ***
2     LEFT
              *** END ***

Select Panel Variation?    ===> C2
```

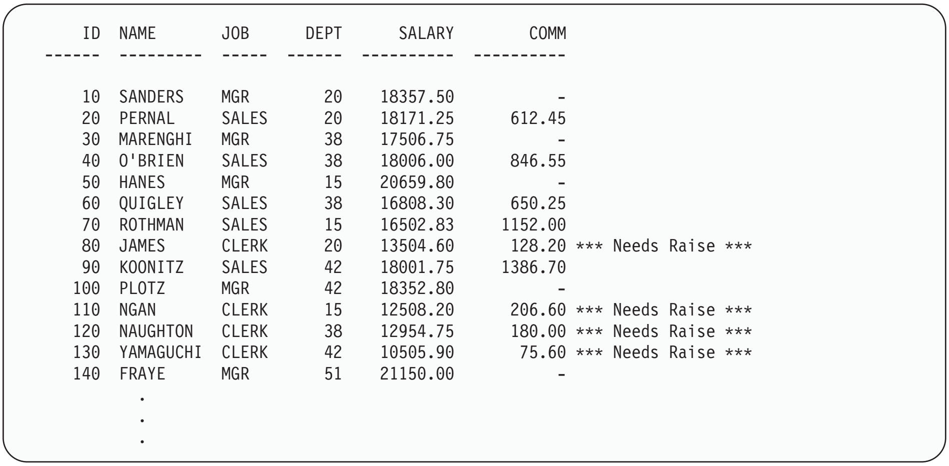
Figure 130. Specify detail block text to appear when the condition is true.

13. Change the width of the report so that the detail block text appears. On the QMF command line, enter:

    ```
    SHOW FORM.OPTIONS
    ```

14. In the **Report text line width** field, type `80` to change the report width to 80 columns.

15. Press the Report function key to see the changed report.



```
    ID  NAME       JOB     DEPT      SALARY        COMM
------  ---------  -----  ------  ----------  ----------

    10  SANDERS    MGR        20    18357.50           -
    20  PERNAL     SALES      20    18171.25      612.45
    30  MARENGHI   MGR        38    17506.75           -
    40  O'BRIEN    SALES      38    18006.00      846.55
    50  HANES      MGR        15    20659.80           -
    60  QUIGLEY    SALES      38    16808.30      650.25
    70  ROTHMAN    SALES      15    16502.83     1152.00
    80  JAMES      CLERK      20    13504.60      128.20 *** Needs Raise ***
    90  KOONITZ    SALES      42    18001.75     1386.70
   100  PLOTZ      MGR        42    18352.80           -
   110  NGAN       CLERK      15    12508.20      206.60 *** Needs Raise ***
   120  NAUGHTON   CLERK      38    12954.75      180.00 *** Needs Raise ***
   130  YAMAGUCHI  CLERK      42    10505.90       75.60 *** Needs Raise ***
   140  FRAYE      MGR        51    21150.00           -
          .
          .
          .
```

Figure 131. The changed reports shows employees who need a raise.

Mixing tabular data with reformatted text

You can mix tabular data and reformatted text within blocks of text.

To mix tabular and detail block text:

1. Run the query and display the report.

   For this example, run the following SQL query:

   ```
   SELECT ACCTNO, COMPANY, NOTES
     FROM Q.SUPPLIER
   ```

2. On the QMF command line, enter:

   ```
   SHOW FORM.COLUMNS
   ```

   For this example, make the following changes on the FORM.COLUMNS panel:

   a. Specify an OMIT usage code for all columns except the NOTES column, because the NOTES column is the only column you will be displaying as tabular data.

   b. Change the edit code for the NOTES column to CT, to allow text wrapping within the column, and the width to 40.

3. On the QMF command line, enter:

   ```
   SHOW FORM.DETAIL
   ```

4. Make the following changes on the FORM.DETAIL panel:

   a. In the **Include Column Headings with Detail Headings?** field, type `NO` so that the column headings will not appear after the detail heading text.

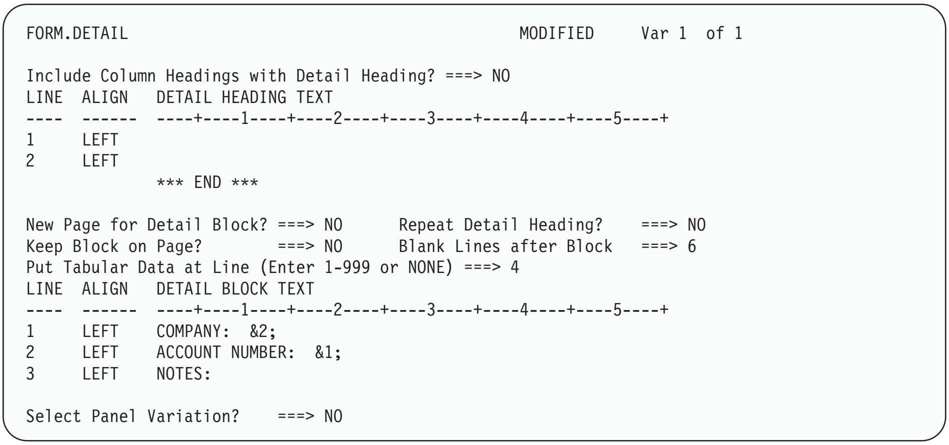
   b. In the **Blank Lines after Block** field, type `6`.

   c. Specify the line number on which you want to display the tabular data in the **Put Tabular Data at Line** field. Make sure you specify different line numbers for the detail block text and the tabular data. Otherwise, one will overlay the other when you display the report. For this example, type `4`.

   d. Type the information for the detail block text. For this example, type `Company:  &2` for the first line. Type `Account Number:  &1` for the second line. Type `Notes:` for the third line.

      You do not have to specify a form variable for the tabular data; the data will follow the last line of detail block text.

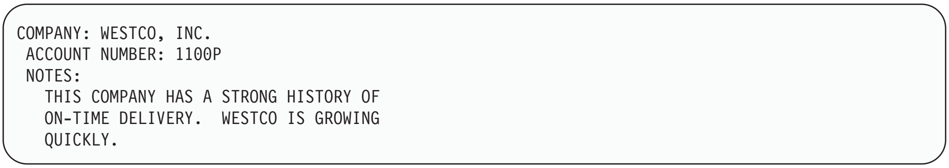
   For more information on specifying detail block text, see “Refining the format of your report with detail blocks” on page 145.



```
FORM.DETAIL                                          MODIFIED     Var 1  of 1

Include Column Headings with Detail Heading? ===> NO
LINE  ALIGN   DETAIL HEADING TEXT
----  ------  ----+----1----+----2----+----3----+----4----+----5----+
1     LEFT
2     LEFT
              *** END ***

New Page for Detail Block? ===> NO      Repeat Detail Heading?    ===> NO
Keep Block on Page?        ===> NO      Blank Lines after Block   ===> 6
Put Tabular Data at Line (Enter 1-999 or NONE) ===> 4
LINE  ALIGN   DETAIL BLOCK TEXT
----  ------  ----+----1----+----2----+----3----+----4----+----5----+
1     LEFT    COMPANY:  &2;
2     LEFT    ACCOUNT NUMBER:  &1;
3     LEFT    NOTES:

Select Panel Variation?    ===> NO
```

Figure 132. Specify placement of tabular data with detail block text.

5. Press the Report function key to see the changed report.



```
COMPANY: WESTCO, INC.
 ACCOUNT NUMBER: 1100P
 NOTES:
   THIS COMPANY HAS A STRONG HISTORY OF
   ON-TIME DELIVERY.  WESTCO IS GROWING
   QUICKLY.
```

Figure 133. The changed report shows tabular data mixed with a detail block.

When you omit some columns from the report, as in this example, you reduce the automatic total width of the report. Make sure that your report is wide enough to include all of the detail block text. Change the width of your report by changing the **Report text line width** field on the FORM.OPTIONS panel.

Showing totals across rows in a report

You can display a report that gives a total or average across the rows in a report by using column usage codes. Totals and averages are examples of QMF's aggregation functions, which are any functions that summarize the data in a column. You can also specify other aggregation usages, such as standard deviation, percent, or cumulative totals. For more information on aggregation functions, see *DB2 QMF Reference*.

In this example, you use the ACROSS, GROUP, SUM, and OMIT usage codes to create a report that summarizes the salary, commission, and total earnings for each job description within every department.

To summarize report data:

1. Run the query to display the report:

For this example, run the following SQL query:

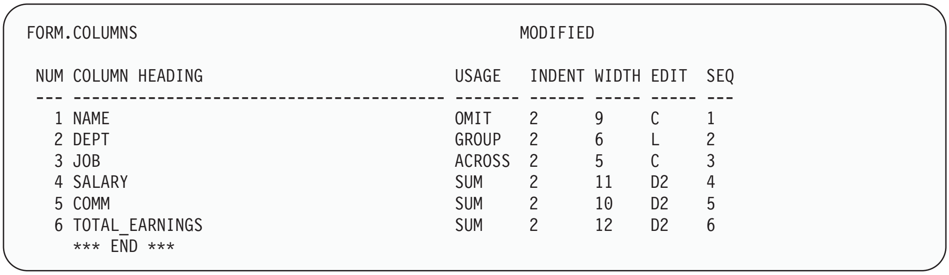
```
SELECT NAME, DEPT, JOB, SALARY, COMM, SALARY + COMM
  FROM Q.STAFF
  WHERE DEPT IN (15, 20, 38) AND JOB <> 'MGR'
  ORDER BY DEPT, JOB
```

2. On the QMF command line, enter:

   ```
   SHOW FORM.COLUMNS
   ```

   The FORM.COLUMNS panel displays.
3. Type any changes to the column names in the **COLUMN HEADING** field.

   For this example, change the column name that is created by the query to `TOTAL_EARNINGS`.
4. Type the usage codes and other changes for the columns. For this example, make the following changes:
   a. Specify `GROUP` for the DEPT column to group your data by department. You must group your data by at least one column. Be sure to order your data by that column.
   b. Specify `ACROSS` for the JOB column to summarize salary, commission, and total earnings by job.
   c. Specify `SUM` for the SALARY, COMM, and TOTAL_EARNINGS columns.
   d. Specify `OMIT` for the NAME column, because you do not want to display it on the report. When you use the GROUP usage code, if you leave a column usage blank, that column does not appear on the report.
   e. Specify `D2` in the **EDIT** field for the SALARY, COMM, and TOTAL_EARNINGS columns.
   f. Change the width of the SALARY column to 11.



```
FORM.COLUMNS                                              MODIFIED

 NUM COLUMN HEADING                         USAGE   INDENT WIDTH EDIT  SEQ
 --- -------------------------------------- ------- ------ ----- ----- ---
   1 NAME                                   OMIT    2      9     C     1
   2 DEPT                                   GROUP   2      6     L     2
   3 JOB                                    ACROSS  2      5     C     3
   4 SALARY                                 SUM     2      11    D2    4
   5 COMM                                   SUM     2      10    D2    5
   6 TOTAL_EARNINGS                         SUM     2      12    D2    6
     *** END ***
```

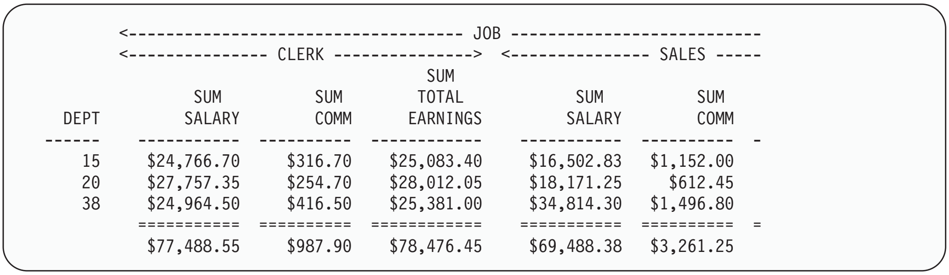
Figure 134. Use GROUP and ACROSS usage codes to summarize data in a report.

5. On the QMF command line, enter:

   ```
   SHOW FORM.OPTIONS
   ```

The FORM.OPTIONS panel displays.

6. In the **Automatic reordering of report columns?** field, type YES. If you do not make this change, your report displays with a warning at the top.
7. Press the Report function key to see the changed report.
8. For this example, press the Right function key to see the summary column.



```
        <------------------------------------ JOB ---------------------------
        <--------------- CLERK --------------->  <--------------- SALES -----
                                         SUM
                SUM          SUM        TOTAL            SUM          SUM
  DEPT         SALARY        COMM      EARNINGS         SALARY        COMM
------    -----------  ----------  ------------    -----------  ----------  -
    15     $24,766.70     $316.70    $25,083.40     $16,502.83   $1,152.00
    20     $27,757.35     $254.70    $28,012.05     $18,171.25     $612.45
    38     $24,964.50     $416.50    $25,381.00     $34,814.30   $1,496.80
          ===========  ==========  ============    ===========  ==========  =
           $77,488.55     $987.90    $78,476.45     $69,488.38   $3,261.25
```

Figure 135. Pressing the Right function key displays the rest of the summarized data.

Correcting errors on a form before displaying a report

You can use the CHECK command to check for errors on a form panel before you run the report.

To check a form panel:

1. Display the form panel you want to check for errors. You can display any form panel for a particular form. QMF checks for errors on that panel and all the other panels for that form as well.

   QMF checks for two types of errors:

   - Errors that you must correct before you display the report
   - Warnings that you do not have to correct, but that can cause unexpected results when you display the report
2. On a form panel, press the Check function key.

   Or, you can enter CHECK on the QMF command line.

   If QMF finds an error on a panel, it displays that panel with the field in error highlighted. The message on the message line describes the error.
3. Correct the field in error.

   Press the Help function key to see more information about the error and instructions to correct it.
4. Press the Check function key or enter CHECK to see any remaining errors.

After you correct any errors, QMF displays any warning conditions when you issue the CHECK command. The steps for correcting warning conditions are the same as for correcting errors.

Saving the report form

If you want to display a report in the same format again, you can save your form in the database in one of the following ways:

If you are on the FORM panel, enter:

```
SAVE
```

QMF prompts you for the name you want to assign to the form.

You can also enter:

```
SAVE AS formname
```

Where *formname* is the name you want to assign to the form.

If you are on a panel other than the FORM panel, enter:

```
SAVE FORM
```

QMF prompts you for the name you want to assign to the form.

You can also enter:

```
SAVE FORM AS formname
```

If you want to save your form and share it with other users, add the SHARE=YES parameter to the SAVE command you are using as follows:

```
SAVE (SHARE=YES
SAVE AS formname (SHARE=YES
SAVE FORM (SHARE=YES
SAVE FORM AS formname (SHARE=YES
```

QMF saves your form in the database. If you issue a SET GLOBAL command with the value DSQEC_SHARE=1 prior to issuing a SAVE command, you do not need to use the SHARE=YES parameter.

To use this report form again when you run a query, enter:

```
RUN QUERY queryname (FORM=formname
```

The data in a query must fit the form you use, or the report will not display.

Resetting the values on a form panel

You can reset the values on a form panel to the default values. This ability is useful if you entered values that do not produce the results you want, so that you can start again from the default values.

To reset values:

- To reset values on all form panels, on the QMF command line, enter:

  ```
  RESET FORM
  ```

  When you enter this command from the FORM.MAIN panel, you do not need to specify the object type as FORM. The object type defaults to FORM when the RESET command is entered from a form panel.
- To reset the values on a specific form panel, enter:

  ```
  RESET FORM.panelname
  ```

  For example, to reset the values on the FORM.COLUMNS panel, enter:

  ```
  RESET FORM.COLUMNS
  ```

  When you enter this command from any specific form panel, you do not need to specify the object type as FORM.*panelname*. The object type defaults to FORM*panelname* when the RESET command is entered from a specific form panel.

Remember that you can only reset the values on the FORM panels before you save the form in the database.

For more information on resetting the values on form panels, see *DB2 QMF Reference*.

Printing your report

You can print your report to paper. The report must be in temporary storage. The rules for printing QMF reports vary depending on which operating system you use and how you set up your printer. See your QMF administrator if you need help printing reports. You can also print your report in a Windows environment by using the DB2 QMF for Windows feature. See Appendix C, “The DB2 QMF High Performance Option,” on page 367 for more information.

To print a report: On the QMF command line, enter:

```
PRINT REPORT (PRINTER=printer
```

Where *printer* is the printer nickname your QMF administrator set up for you.

When you initiate the PRINT command from a report panel, you do not need to specify the object type in the command. From a report panel, the object type for the PRINT command defaults to REPORT.

For more information on the PRINT command, see *DB2 QMF Reference*.

CC (carriage control)

In QMF V8.1, CC (carriage control) is a new key word which has these constraints:

- When the key word option PRinter = Printer name, the CC option is ignored.
- When the key word option PRinter = Blank, the CC option can be set from the command line or from the PRINT panel prompt.

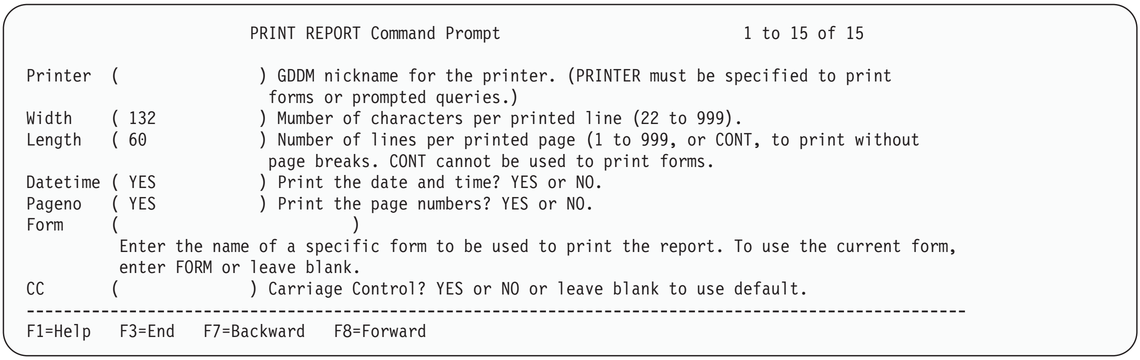
From the command line, you would type

```
PRINT REPORT (CC = x
```

where x can be Y, N, or Not specified.

- Y- leave the carriage control characters in place
- N- remove the carriage control characters; shift the report to the left by one space
- Not specified- the CC will be set with the value from global variable DSQEC_CC

CC can also be set from the PRINT panel prompt:



```
                          PRINT REPORT Command Prompt                          1 to 15 of 15

Printer  (               ) GDDM nickname for the printer. (PRINTER must be specified to print
                          forms or prompted queries.)
Width    ( 132           ) Mumber of characters per printed line (22 to 999).
Length   ( 60            ) Number of lines per printed page (1 to 999, or CONT, to print without
                          page breaks. CONT cannot be used to print forms.
Datetime ( YES           ) Print the date and time? YES or NO.
Pageno   ( YES           ) Print the page numbers? YES or NO.
Form     (                          )
          Enter the name of a specific form to be used to print the report. To use the current form,
          enter FORM or leave blank.
CC       (              ) Carriage Control? YES or NO or leave blank to use default.
-----------------------------------------------------------------------------------------------------
F1=Help   F3=End   F7=Backward   F8=Forward
```

Figure 136. Setting CC from the PRINT panel prompt

Creating a report containing a limited number of ordered entries

You can create a report that contains a limited number of ordered entries. To do so, first construct a SQL query that selects and orders entries as appropriate to your objective. Then, run the query and specify the row limit value that gives the desired result. The key elements are the ORDER BY clause of the SQL statement, and the ROWLIMIT parameter of the RUN QUERY command. For example, to create a report that contains the five managers with the most years of service, you could use the following query and QMF command.

SQL query:

```
SELECT NAME, YEARS
  FROM Q.STAFF
 WHERE JOB='MGR'
 ORDER BY YEARS DESC
```

QMF command:

```
RUN QUERY (ROWLIMIT=5
```

Resulting report:

```
NAME        YEARS
---------  ------
JONES          12
QUILL          10
HANES          10
LU             10
LEA             9
```

Chapter 7. Displaying your report as a chart

You can display your tabular data in a wide variety of charts. You can change QMF chart formats, or create new chart formats. You can also use charting tools in Windows ″suites″ and other graphical tools from any Windows environment that is supported by the DB2 QMF for Windows feature. See Appendix C, “The DB2 QMF High Performance Option,” on page 367 for more information.

QMF can send your report data to the Interactive Chart Utility (ICU), which displays the data as a chart.

You do not need to learn everything about the ICU to create charts. You can create many basic charts by using only the QMF interface to the ICU.

Your installation might not support the use of charts. Check with your QMF administrator before you try to create charts.

QMF chart formats

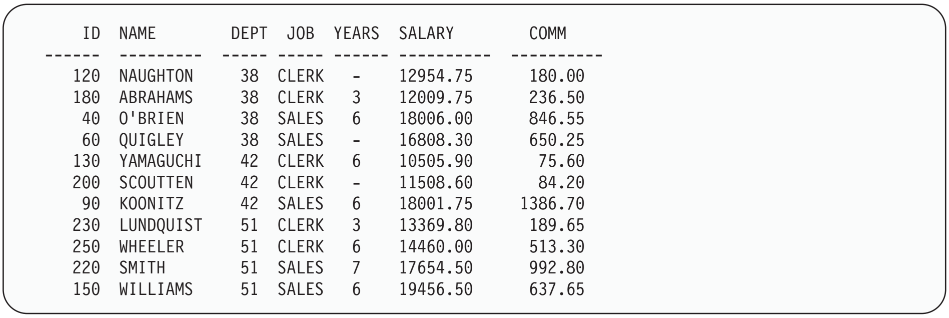
QMF provides the following chart formats. To use a chart format, specify its name as a parameter when you type the `DISPLAY CHART` command.

BAR (the QMF default chart)
PIE
LINE
TOWER
TABLE
POLAR
HISTOGRAM
SURFACE
SCATTER

For more information on QMF’s chart types, see *DB2 QMF Reference*.

Where QMF report data appears on a chart

Compare the report in Figure 137 on page 168 and the bar chart in Figure 138 on page 168 to see how QMF displays report data on a chart. QMF created both the report and bar chart by using the default report form.



```
    ID  NAME        DEPT  JOB   YEARS  SALARY       COMM
------  ---------  ----- ----- ------ ----------  ----------
   120  NAUGHTON     38  CLERK    -    12954.75      180.00
   180  ABRAHAMS     38  CLERK    3    12009.75      236.50
    40  O'BRIEN      38  SALES    6    18006.00      846.55
    60  QUIGLEY      38  SALES    -    16808.30      650.25
   130  YAMAGUCHI    42  CLERK    6    10505.90       75.60
   200  SCOUTTEN     42  CLERK    -    11508.60       84.20
    90  KOONITZ      42  SALES    6    18001.75     1386.70
   230  LUNDQUIST    51  CLERK    3    13369.80      189.65
   250  WHEELER      51  CLERK    6    14460.00      513.30
   220  SMITH        51  SALES    7    17654.50      992.80
   150  WILLIAMS     51  SALES    6    19456.50      637.65
```

Figure 137. This report shows employee data.

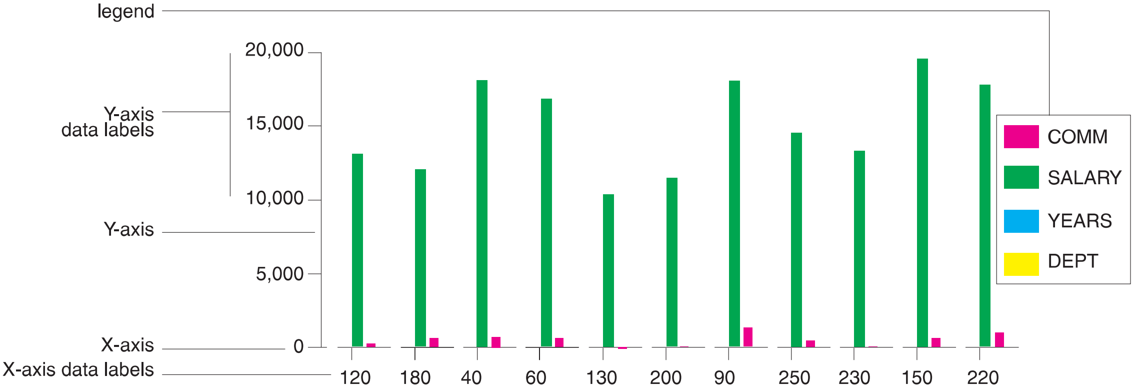


Figure 138. This bar chart shows the same employee data as the report.

You specify the following parts of a chart using a QMF form:

- *chart heading*
- *X-axis*
- *X-axis data labels*
- *Y-axis*
- *Y-axis data labels*
- *legend*

In general, report data appears on a chart according to the following rules:

Chart heading
Report heading

X-axis data
First (leftmost) column of the report. If you define a GROUP or BREAK column, the data in that column appears on the X-axis.

X-axis data labels
Values in the leftmost column, or the GROUP or BREAK column.

Y-axis data
Remaining numeric columns.

Y-axis data labels
Values in the remaining numeric columns.

Legend
Column headings for the Y-axis data.

How QMF spaces data along the X-axis

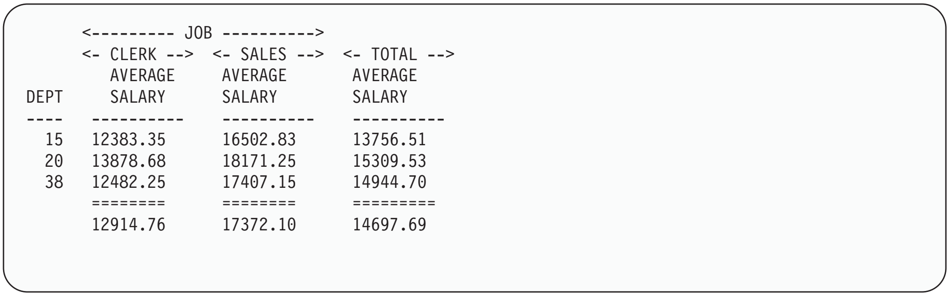
The following rules apply to how QMF spaces data along the X-axis for the chart formats that it provides:

- QMF plots numeric data from a single report column by its actual numeric value.
- QMF spaces non-numeric data from a single report column at even intervals.
- QMF spaces numeric data or non-numeric data from multiple report columns at even intervals.

The QMF-provided chart formats for bar, tower, and polar charts space both numeric values and non-numeric values at even intervals. If you specify one of these chart types in the ICU, rather than using the QMF DISPLAY command, your data might be spaced unevenly along the X-axis.

Where data appears on pie charts

Pie charts do not have a typical X-axis and Y-axis. Compare the report in Figure 139 and the pie chart in Figure 140 on page 170 to see how QMF displays data on a pie chart.



```
      <--------- JOB ---------->
      <- CLERK -->  <- SALES -->  <- TOTAL -->
         AVERAGE     AVERAGE       AVERAGE
DEPT     SALARY      SALARY        SALARY
----   ----------    ----------    ----------
  15   12383.35      16502.83      13756.51
  20   13878.68      18171.25      15309.53
  38   12482.25      17407.15      14944.70
       ========      ========      =========
       12914.76      17372.10      14697.69
```

Figure 139. This report shows department salary averages.

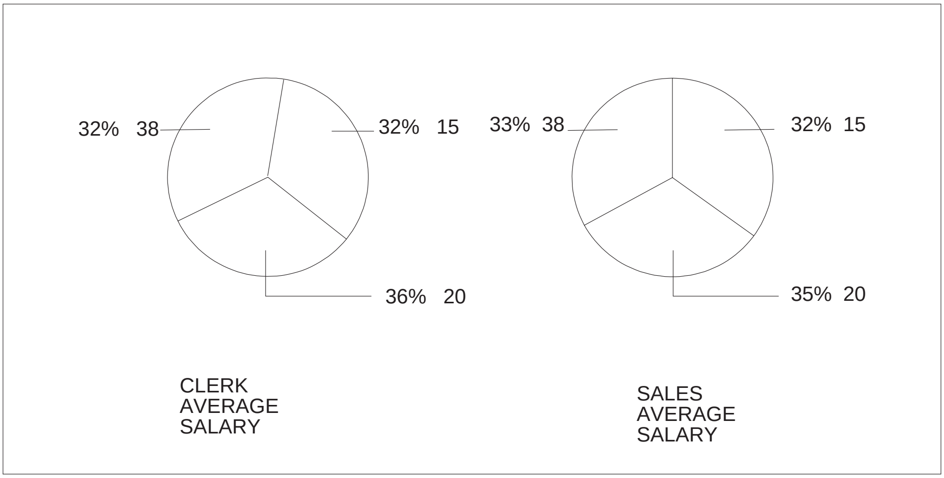


Figure 140. The pie chart displays the same data.

Each numeric column (called Y-data columns) produces a separate pie. QMF displays column data as labels attached to the pie slices. It displays column headings as titles for the pies.

Because each numeric column on a report produces a separate pie, you probably will not want to use a pie chart to display data from reports with more than two numeric columns.

Chart data size limits

The following size limits apply to chart data:

- 132 bytes for chart title, X-axis label, and legend label
- 999 X-data values (report data rows)
- 999 Y-data groups (report data columns)
- 8,192 Y-data values (number of X-data rows times the number of Y-data columns per row)

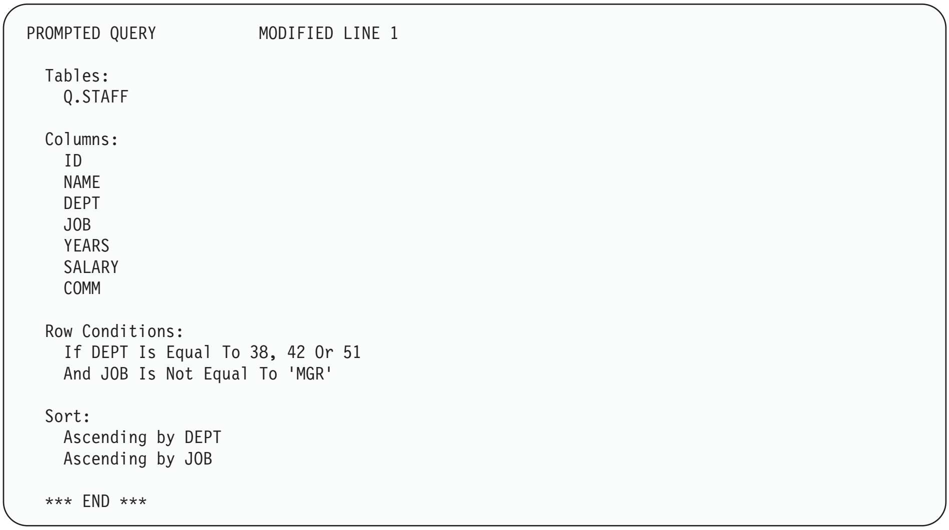
QMF defines the last restriction. The others are ICU limits that are validated by QMF.

There are no data size limits when using the DB2 QMF for Windows feature in a Windows environment. Your Windows applications may have data size limits for charting and graphing; their product documentation should be consulted accordingly. See Appendix C, "The DB2 QMF High Performance Option," on page 367 for more information about the DB2 QMF for Windows feature.

Displaying report data as a chart

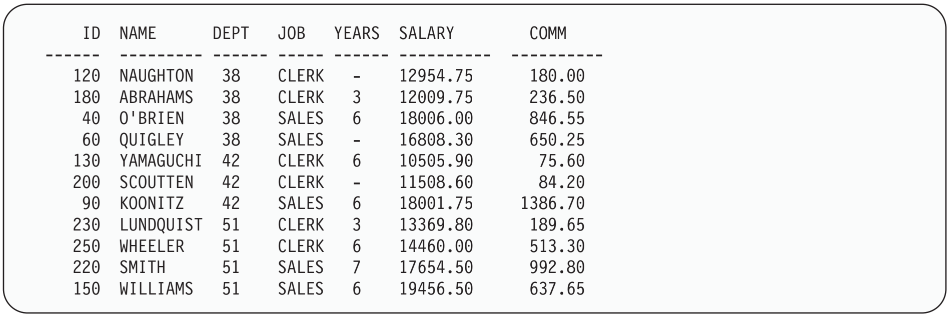
1. Run the query to display the report.

   For this example, run the following prompted query:
   To display this report using the default chart format:



```
PROMPTED QUERY          MODIFIED LINE 1

  Tables:
    Q.STAFF

  Columns:
    ID
    NAME
    DEPT
    JOB
    YEARS
    SALARY
    COMM

  Row Conditions:
    If DEPT Is Equal To 38, 42 Or 51
    And JOB Is Not Equal To 'MGR'

  Sort:
    Ascending by DEPT
    Ascending by JOB

  *** END ***
```

Figure 141. Use this query to produce the charts for the examples.



```
      ID  NAME      DEPT   JOB   YEARS  SALARY        COMM
  ------  --------- ------ ----- ------ ----------  ----------
     120  NAUGHTON   38    CLERK   -    12954.75      180.00
     180  ABRAHAMS   38    CLERK   3    12009.75      236.50
      40  O'BRIEN    38    SALES   6    18006.00      846.55
      60  QUIGLEY    38    SALES   -    16808.30      650.25
     130  YAMAGUCHI  42    CLERK   6    10505.90       75.60
     200  SCOUTTEN   42    CLERK   -    11508.60       84.20
      90  KOONITZ    42    SALES   6    18001.75     1386.70
     230  LUNDQUIST  51    CLERK   3    13369.80      189.65
     250  WHEELER    51    CLERK   6    14460.00      513.30
     220  SMITH      51    SALES   7    17654.50      992.80
     150  WILLIAMS   51    SALES   6    19456.50      637.65
```

Figure 142. The query produces this report for the examples.

2. Press the Chart function key.

   Or you can enter:

   ```
   DISPLAY CHART
   ```

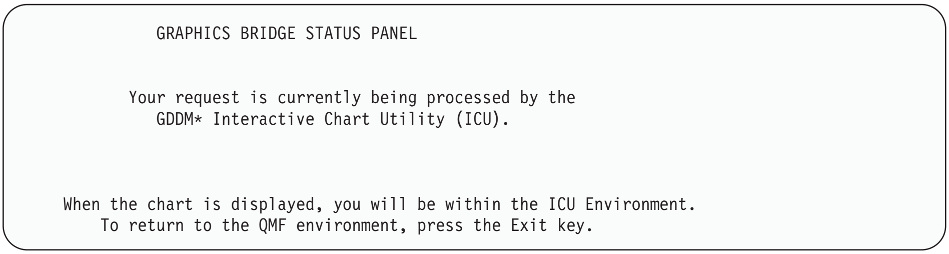
   If you do not specify a chart type, QMF creates the chart by using the GDDM default chart type. The following examples assume that the GDDM default chart type is bar.

   To specify a different chart type, enter:

```
DISPLAY CHART (ICUFORM=charttype
```

For the QMF-provided chart types, see "QMF chart formats" on page 167.

While the ICU is creating the chart, you see a panel like the one in Figure 143.
The chart displays.



```
                GRAPHICS BRIDGE STATUS PANEL


             Your request is currently being processed by the
                GDDM* Interactive Chart Utility (ICU).



      When the chart is displayed, you will be within the ICU Environment.
          To return to the QMF environment, press the Exit key.
```

Figure 143. The Graphics Bridge Status panel

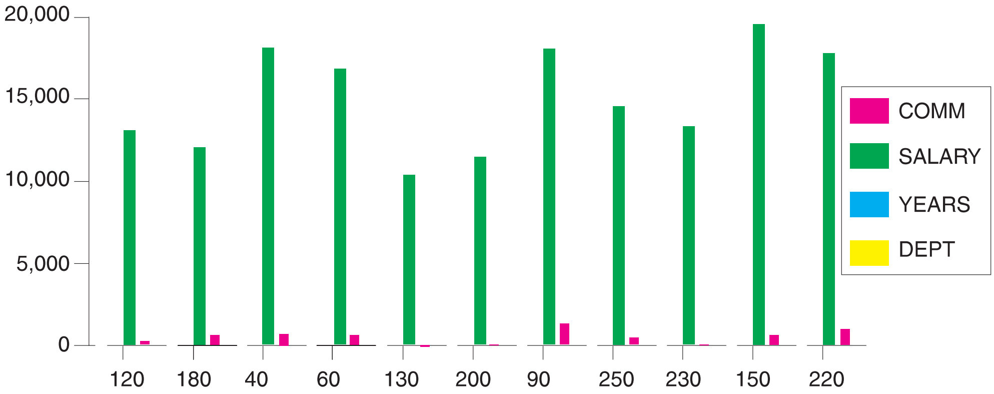


Figure 144. The default bar chart displays.

QMF creates this chart by using the default report form and the default chart format. You can change the appearance of a chart by changing the QMF form and by changing the ICU values.

3. Press the Home function key, then the Exit function key to return to the QMF panel where you issued the `DISPLAY CHART` command.

Changing a chart using QMF forms

You can change the format and content of your chart by using QMF form panels. For example, you can make the following types of changes on a form panel:

- Use the OMIT usage code for columns you do not want to display on the chart.

- Use the GROUP usage code to group all the entries in a particular column and display the grouped data on the chart.
- Use the AVERAGE usage code to calculate the average value for grouped data and display the average on the chart.
- Change the **PAGE HEADING** field to change the chart heading.
- Change the chart legend by changing column headings.

Use QMF forms to change any chart property that is determined by the report data. Table 8 identifies changes you can make to charts by using QMF forms.

Table 8. Quick reference for changing a chart with QMF forms

Change	Object	Comments
Chart type	CHART	Use ICUFORM parameter
X-axis data label text	FORM	Leftmost column (BREAK or GROUP)
X-axis data label length	FORM	WIDTH of leftmost column
Y-axis chart data	FORM	Numeric data columns (not OMIT)
Legend text*	FORM	Column headings of Y-data
Legend case	PROFILE	Select uppercase or string for the CASE option
Chart heading text	FORM	Page heading
Chart heading case	PROFILE	Select uppercase or string for the CASE option

* Legend text for pie charts is an exception. The data from the report column that normally appears as labels along the X-axis appears as labels attached to the pie slices.

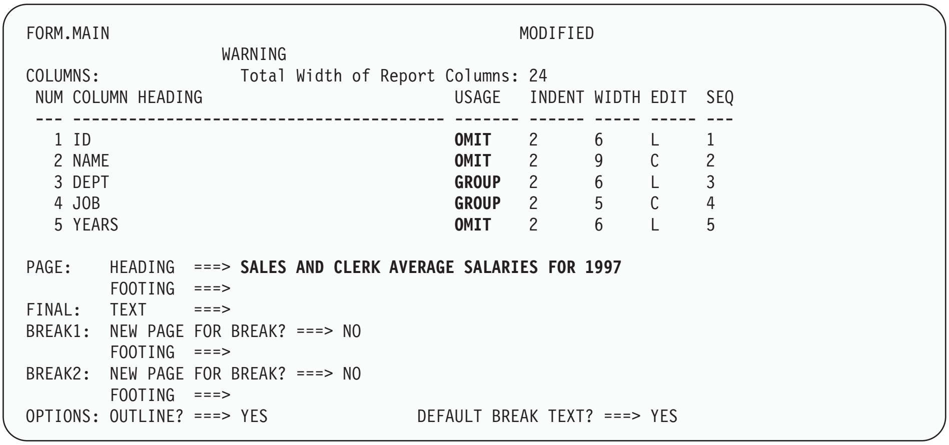
For this example, you will modify the QMF default form from the previous example to display a bar chart that shows average salaries for clerks and salespeople by department.

To change a chart using forms:

1. Display the FORM panel you need to change.

   In this example, you make all the form changes on the FORM.MAIN panel. Enter `SHOW FORM` on the QMF command line, or press the Show function key.
2. Type the changes to the form.

   For this example, type the changes that are shown on the FORM.MAIN panel in Figure 145 on page 174.



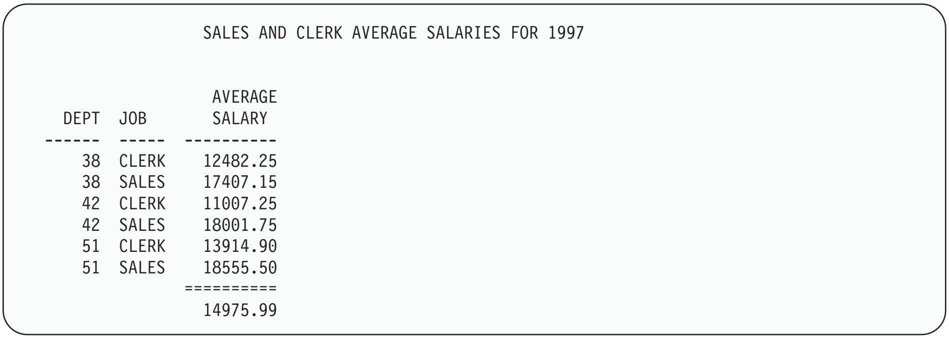
```
FORM.MAIN                                            MODIFIED
                     WARNING
COLUMNS:               Total Width of Report Columns: 24
 NUM COLUMN HEADING                          USAGE   INDENT WIDTH EDIT  SEQ
 --- ---------------------------------------- ------- ------ ----- ----- ---
   1 ID                                      OMIT    2      6     L     1
   2 NAME                                    OMIT    2      9     C     2
   3 DEPT                                    GROUP   2      6     L     3
   4 JOB                                     GROUP   2      5     C     4
   5 YEARS                                   OMIT    2      6     L     5

PAGE:    HEADING  ===> SALES AND CLERK AVERAGE SALARIES FOR 1997
         FOOTING  ===>
FINAL:   TEXT     ===>
BREAK1:  NEW PAGE FOR BREAK? ===> NO
         FOOTING  ===>
BREAK2:  NEW PAGE FOR BREAK? ===> NO
         FOOTING  ===>
OPTIONS: OUTLINE? ===> YES               DEFAULT BREAK TEXT? ===> YES
```

Figure 145. Make changes to the chart format on FORM.MAIN.

If the sixth and seventh columns do not appear on the sample panel, scroll forward to see them. Here is the information you change for those columns:

```
NUM  COLUMN HEADING         USAGE

  6  SALARY                 AVERAGE
  7  COMMISSION             OMIT
```

3. Press the Report function key to see the changed report.



```
                   SALES AND CLERK AVERAGE SALARIES FOR 1997


                    AVERAGE
  DEPT  JOB         SALARY
------  -----  ----------
    38  CLERK    12482.25
    38  SALES    17407.15
    42  CLERK    11007.25
    42  SALES    18001.75
    51  CLERK    13914.90
    51  SALES    18555.50
               ==========
                 14975.99
```

Figure 146. The report reflects the changes you make on FORM.MAIN.

4. Enter `DISPLAY CHART`, or press the Chart function key to display your new chart.

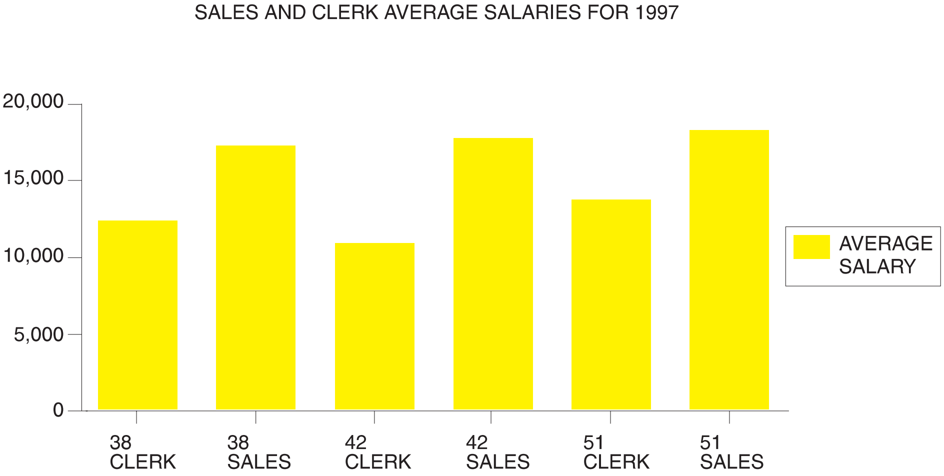
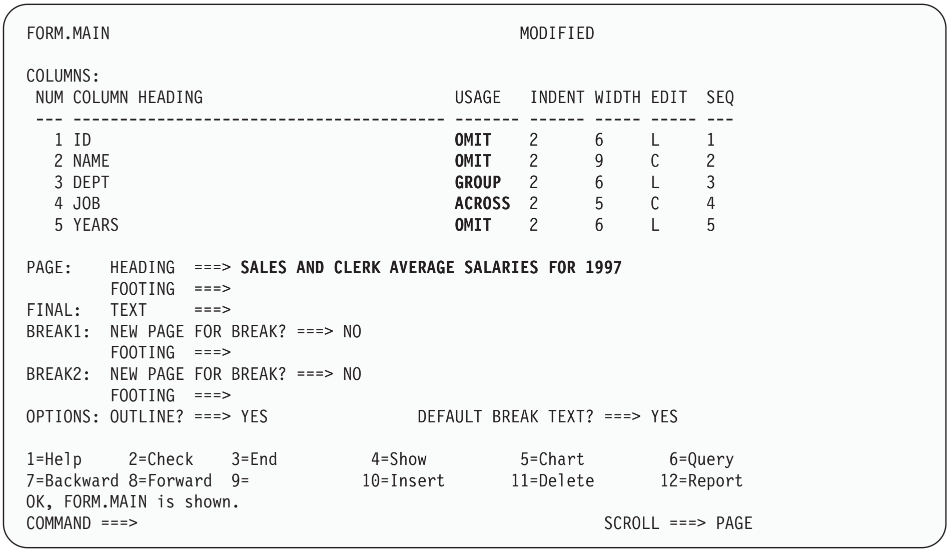


Figure 147. The chart reflects the same changes.

You can change the QMF form to display the same data in a way that makes it easier to compare the salaries in a chart.

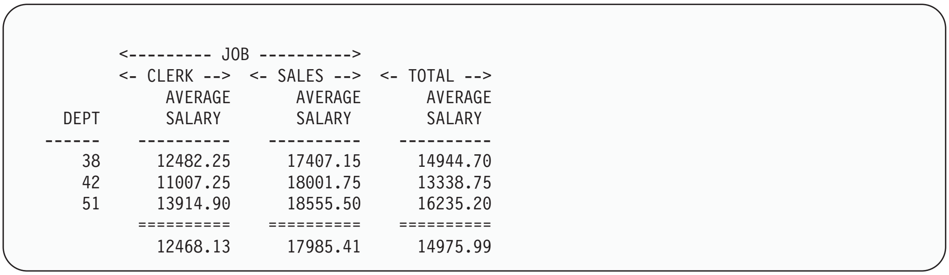
For this example, type the information on the FORM.MAIN panel.Figure 148 shows the information to type.



```
FORM.MAIN                                              MODIFIED

COLUMNS:
 NUM COLUMN HEADING                           USAGE   INDENT WIDTH EDIT  SEQ
 --- ---------------------------------------- ------- ------ ----- ----- ---
   1 ID                                       OMIT    2      6     L     1
   2 NAME                                     OMIT    2      9     C     2
   3 DEPT                                     GROUP   2      6     L     3
   4 JOB                                      ACROSS  2      5     C     4
   5 YEARS                                    OMIT    2      6     L     5

PAGE:    HEADING  ===> SALES AND CLERK AVERAGE SALARIES FOR 1997
         FOOTING  ===>
FINAL:   TEXT     ===>
BREAK1:  NEW PAGE FOR BREAK? ===> NO
         FOOTING  ===>
BREAK2:  NEW PAGE FOR BREAK? ===> NO
         FOOTING  ===>
OPTIONS: OUTLINE? ===> YES               DEFAULT BREAK TEXT? ===> YES

1=Help     2=Check    3=End          4=Show          5=Chart         6=Query
7=Backward 8=Forward  9=            10=Insert       11=Delete       12=Report
OK, FORM.MAIN is shown.
COMMAND ===>                                                SCROLL ===> PAGE
```

Figure 148. Group data for the chart on FORM.MAIN.

The report looks like Figure 149 on page 176.



```
           <--------- JOB ---------->
           <- CLERK -->  <- SALES -->  <- TOTAL -->
                AVERAGE       AVERAGE       AVERAGE
  DEPT          SALARY        SALARY        SALARY
------       ----------    ----------    ----------
    38         12482.25      17407.15      14944.70
    42         11007.25      18001.75      13338.75
    51         13914.90      18555.50      16235.20
             ==========    ==========    ==========
               12468.13      17985.41      14975.99
```

Figure 149. The report shows average salaries for sales and clerks.

Within each department, the report displays the average salary for clerks and for salespeople in a different column. Each column is a bar on the chart. The TOTAL column and the final summary line do not appear on a chart. The chart looks like Figure 150.

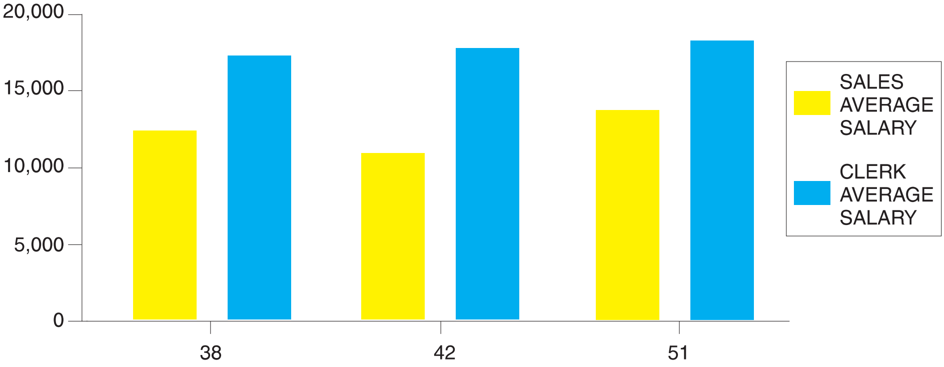


Figure 150. The chart makes comparing average salaries easier.

Changing a chart in the ICU

The changes you make to a chart by using QMF forms usually affect the data you want to display on the chart. The changes you make to a chart in the ICU usually affect the format of the chart itself.

Use the ICU to change chart presentation properties, such as color, position, and size. Table 9 identifies changes you make to charts in the ICU.

Table 9. Quick reference for changing a chart in the ICU

Change	ICU Home panel	Select
Axis titles	AXIS OPTIONS	X-axis; function key for Y-axis
Legend position	HEADING, LEGEND, AND LAYOUT	Legend Position and Format

Table 9. Quick reference for changing a chart in the ICU (continued)

Change	ICU Home panel	Select
Color and size of legend text	DATA ENTRY & IMPORT	Group Name Attributes
Chart size	HEADING, LEGEND, AND LAYOUT	Chart Dimensions
Color and appearance of data	DATA ENTRY & IMPORT	Data Attributes
Position of data along X-axis	DATA ENTRY & IMPORT	Data Interpretation
X-axis label color and size	DATA ENTRY & IMPORT	Data Label Attributes
Chart heading color and size	HEADING, LEGEND, AND LAYOUT	Chart Heading (answer YES)
Heading and axis positions	HEADING, LEGEND, AND LAYOUT	Heading and Axis Positions

Changing a chart format

You can change a chart format in the ICU, such as moving the legend or changing the size of the pie in a pie chart.

To change a chart format:

1. On the QMF command line, enter:
   ```
   DISPLAY CHART (ICUFORM=charttype
   ```
   Where *charttype* is the type of the chart you want to change. The chart of the type you specified displays.
2. Press the Home function key; the ICU Home panel displays.
3. Select the menu for the chart properties you want to change.
4. Make the appropriate selections for the changes.

Saving a chart format

1. Return to the ICU Home panel or any panel that has a Save function key or Save/Load function key.
2. Press the Save or Save/Load key. The Save and Load Chart panel displays.
3. In the **What do you want to do?** field, type 3.
4. Because you want to save only the chart format, type 1 in the **Which part of chart?** field.
5. In the **Format** field under **Filename**, type the name of the chart, for example, MYTOWER.

6. Press Enter.

To display that chart again from the QMF command line, type its name for the ICUFORM parameter. For example, enter:

```
DISPLAY CHART (ICUFORM=MYTOWER
```

Specifying a new default chart format

You can change the default chart format from one QMF-provided chart format to another.

In this example, you change the default chart format from BAR to LINE.

To specify a new default chart format:

1. On the QMF command line, enter:

   ```
   DISPLAY CHART (ICUFORM=charttype
   ```

   Where *charttype* is the chart type of the chart you want to use as the default.

   For this example, enter:

   ```
   DISPLAY CHART (ICUFORM=LINE
   ```
2. Set the REPLACE option to `YES`, and save the chart as DSQCFORM (the QMF default report format) on the ICU Save panel.

Fixing problems with charts

When you display your QMF report data as a chart in the ICU, you may not see exactly what you expect. Here are some tips on how to fix chart problems:

Chart does not display X-axis labels or Y-axis labels

Do one of the following:

- Use the QMF form to truncate the labels by reducing the column widths for those particular columns.
- Use the ICU to make the labels smaller or set them at an angle (or both) using the ICU menu for data label attributes.

Chart does not display all the pies for a pie chart

There is not enough space on your screen to display all the pies at a reasonable size. In the ICU, reduce the margins of your chart by using the menu that is associated with headings, legends, and layout.

Omitted data value labels

The data labels cannot fit on your chart. Do one of the following tasks:

- In QMF, reorder the data in your query so that QMF can group values appropriately. This requires fewer labels on the X-axis.
- In the ICU, reduce the margins of your chart.
- In the ICU, put the labels in a legend rather than attach them to the pie slices. Use the menu that allows you to specify chart options for each chart type.

Wrong spacing of X-axis data

In this case, the position of data along the X-axis is either spaced at equal intervals and you want it spaced according to numeric value, or just the opposite. Do one of the following tasks:

- In QMF, specify an alternate chart type by using a different QMF-provided chart format. Each chart format provides the type of X-axis that is most typically used with its given chart type.
- In the ICU, change the way the data is spaced along the X-axis by using the menu for data interpretation.

Printing your chart

You can print your chart on paper. The chart must be in temporary storage. The rules for printing QMF charts vary depending on which operating system you use and how you set up your printer. See your QMF administrator if you need help printing charts.

To print a chart: On the QMF command line, enter:

```
PRINT CHART (PRINTER=printer
```

Where *printer* is the printer nickname your QMF administrator set up for you.

When you initiate the PRINT command from a chart panel, you do not need to specify the object type in the command. From a chart panel, the object type for the PRINT command defaults to CHART.

For more information on the PRINT command, see *DB2 QMF Reference*.

Chapter 8. Creating a procedure to run QMF commands

You can create two types of procedures to run QMF commands. Create a linear procedure to run a series of QMF commands. Create a procedure with logic to run a series of QMF commands under program control. The commands are run based on REXX logic you add to the procedure.

Use linear procedures if you are using QMF in the CICS environment. If you are using QMF in the CMS or TSO environments, you can also use REXX statements and functions to create procedures with logic.

You can also build procedures that include QMF objects and commands in a Windows environment supported by the DB2 QMF for Windows feature. You build these procedures by using simple Windows application macro languages and application building toolkits. These include nearly all Windows suites, applications, and development environments today. See Appendix C, "The DB2 QMF High Performance Option," on page 367 for more information.

Creating a linear procedure

For this example, create a linear procedure to do the following:

- Select commission data from the Q.STAFF table
- Save the commission data in a separate table in the database
- Print a report that displays commissions for salespeople

To create a linear procedure:

1. Create and save the query and form.

   In this example, REPT4QRY is the name of the saved query, and REPT4FORM is the name of the saved form. Here is the query for this example:

   ```
   SELECT NAME, ID, COMM
   FROM Q.STAFF
   ```

2. On the QMF command line, enter:

   ```
   RESET PROC
   ```

   The PROC panel displays.

3. Type the QMF commands you want this procedure to run in the order you want them to run.

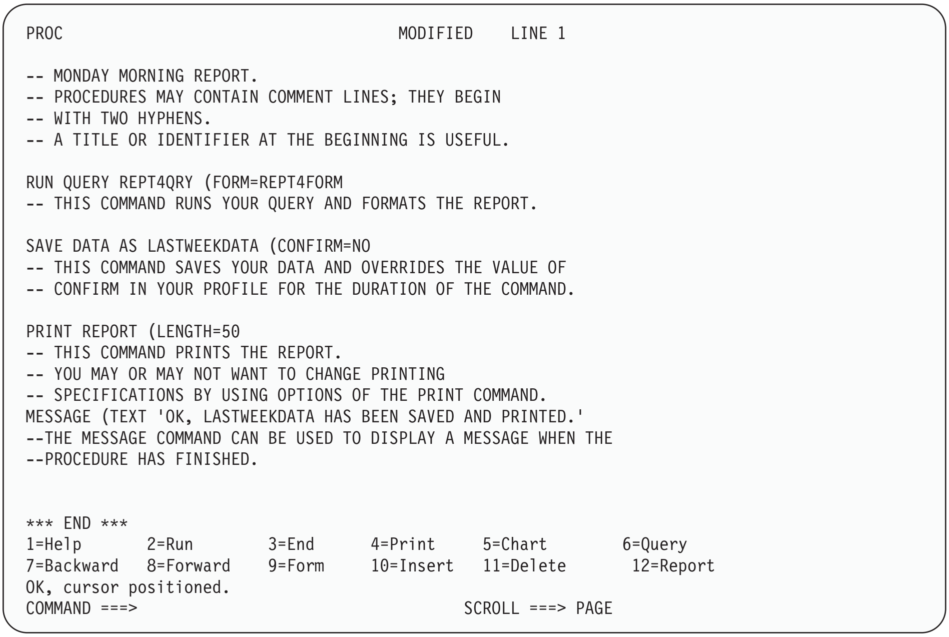
   If you want to display and interact with panels just as you would if you entered a command on the QMF command, type `INTERACT` before the command name.

For more information on the INTERACT command, see *DB2 QMF Reference* .

4. Type comment lines if you need them.
5. To insert lines in a procedure, move the cursor to the line you want to precede the new line, and press the Insert function key.

   Or, you can type `INSERT` on the QMF command line, move the cursor to the line you want to precede the new line, and press Enter.
6. To delete lines from a procedure, move the cursor to the line you want to delete and press the Delete function key.

   Or, you can type `DELETE` on the QMF command line, move the cursor to the line you want to delete, and press Enter.



```
PROC                                MODIFIED    LINE 1

-- MONDAY MORNING REPORT.
-- PROCEDURES MAY CONTAIN COMMENT LINES; THEY BEGIN
-- WITH TWO HYPHENS.
-- A TITLE OR IDENTIFIER AT THE BEGINNING IS USEFUL.

RUN QUERY REPT4QRY (FORM=REPT4FORM
-- THIS COMMAND RUNS YOUR QUERY AND FORMATS THE REPORT.

SAVE DATA AS LASTWEEKDATA (CONFIRM=NO
-- THIS COMMAND SAVES YOUR DATA AND OVERRIDES THE VALUE OF
-- CONFIRM IN YOUR PROFILE FOR THE DURATION OF THE COMMAND.

PRINT REPORT (LENGTH=50
-- THIS COMMAND PRINTS THE REPORT.
-- YOU MAY OR MAY NOT WANT TO CHANGE PRINTING
-- SPECIFICATIONS BY USING OPTIONS OF THE PRINT COMMAND.
MESSAGE (TEXT 'OK, LASTWEEKDATA HAS BEEN SAVED AND PRINTED.'
--THE MESSAGE COMMAND CAN BE USED TO DISPLAY A MESSAGE WHEN THE
--PROCEDURE HAS FINISHED.


*** END ***
1=Help       2=Run        3=End      4=Print     5=Chart        6=Query
7=Backward   8=Forward    9=Form     10=Insert   11=Delete       12=Report
OK, cursor positioned.
COMMAND ===>                                   SCROLL ===> PAGE
```

Figure 151. Enter your procedure on the PROC panel.

7. To save the procedure in the database, enter:

   ```
   SAVE
   ```

   QMF prompts you for the name you want to assign to the procedure.

   You can also enter:

   SAVE AS *procname*

   For this example, enter:

   ```
   SAVE AS MONDAY
   ```

Guidelines for writing linear procedures

Keep the following guidelines in mind when you write linear procedures:

- A linear procedure can contain QMF commands, comment lines beginning with two hyphens (--), and blank lines.
- Use complete names for commands, options, and values, rather than abbreviated names.
- Do not specify a command of more than 2,000 characters (or the equivalent in DBCS). QMF stops running a procedure when it finds a command of over 2,000 characters.
- You can include comments on the same line as a command, but place them after the command.

  If a command spans more than one line, type **+** at the beginning of the continuation line. **+** is the continuation character. For example:

  ```
  RUN QUERY (&&VAR1 = 'THIS IS A VALUE FOR VAR1.'  &&VAR2 = 'THIS
  +IS A VALUE FOR VAR2.'
  ```

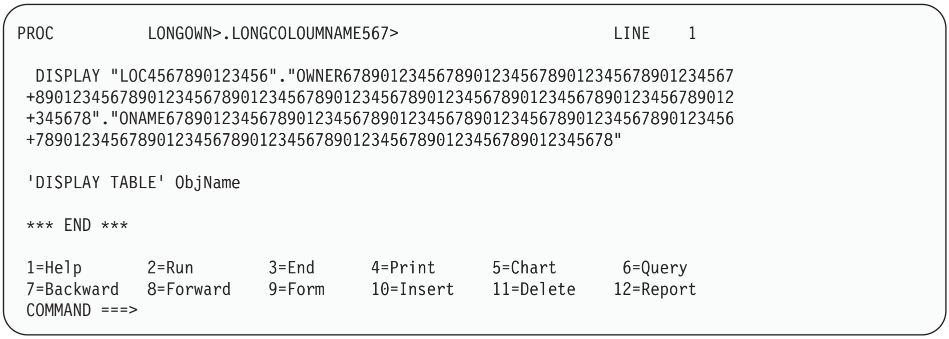
  QMF does not insert a space between the last character of the first line and the first character of the second line, unless either of the following is true:

  – The command includes an open quote

  – You included a space at the end of the first line

  You cannot use a continuation character in a comment line, command keyword, or substitution variable name. You can use a continuation character in a substitution variable value, if you enclose the value in single quotes.

  You can type comments and blank lines between continuation lines.
- Long names support: To continue an object name across one or more lines, it must be enclosed within double quotes. See the screen below.



```
PROC          LONGOWN>.LONGCOLOUMNAME567>                      LINE    1

  DISPLAY "LOC4567890123456"."OWNER678901234567890123456789012345678901234567
 +890123456789012345678901234567890123456789012345678901234567890123456789012
 +345678"."ONAME6789012345678901234567890123456789012345678901234567890123456
 +78901234567890123456789012345678901234567890123456789012345678"

 'DISPLAY TABLE' ObjName

 *** END ***

 1=Help       2=Run        3=End      4=Print      5=Chart       6=Query
 7=Backward   8=Forward    9=Form     10=Insert    11=Delete    12=Report
 COMMAND ===>
```

Figure 152. Linear procedures long name support

Creating a procedure with logic

You can print the same commissions report as in the previous example, but add REXX logic to check whether the day is Monday. If it is Monday, the procedure can automatically print the report.

The rules and the structure of procedures with logic follow those of any REXX program. For more information about the REXX procedural language, see either of the following publications:

- *VM System Product Interpreter Reference*
- *TSO Extensions REXX Reference*

To create a procedure with logic:

1. Create and save the query and form.
2. Enter:

   ```
   RESET PROC
   ```

   The PROC panel displays.
3. Type a REXX comment line as the first line of the procedure. REXX comment lines begin with **/*** and end with ***/**.
4. Type the QMF commands you want the procedure to run in the order you want them to run.

   Because QMF does not convert any text in a procedure, type all QMF commands in uppercase, or they do not run.

   Enclose all QMF commands in quotes, otherwise any QMF command identical to a REXX command (such as EXIT) is processed as a REXX command.

   If you want to display and interact with panels just as you would if you entered a command on the QMF command, type `INTERACT` before the command name.

   For more information on the INTERACT command, see *DB2 QMF Reference*.
5. Type the logic statements for the procedure. You can use any REXX function in a procedure with logic.

   You can also include internal functions for arithmetic operations, character manipulation, data conversion, and information gathering, and you can write your own external functions.
6. Type REXX comment lines (instead of QMF comment lines) if you need them.
7. Type a REXX exit statement at the end of the procedure.

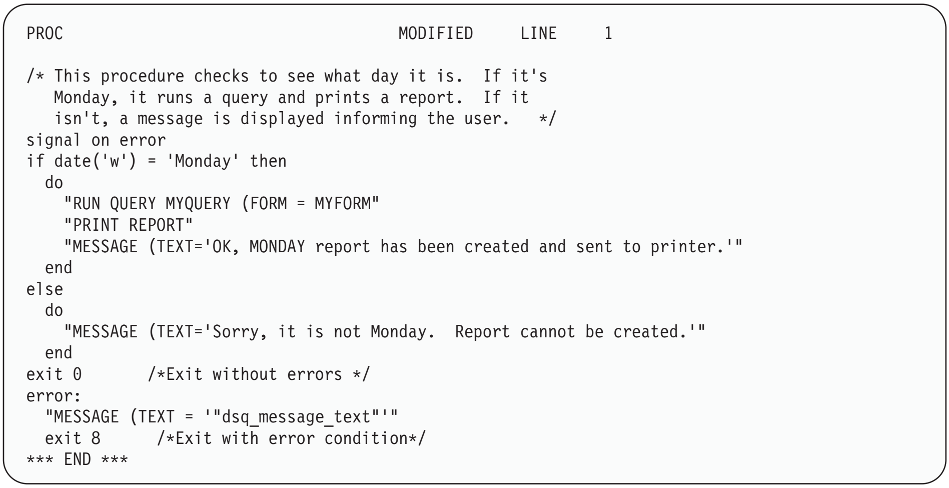
The procedure in Figure 153 has two exit statements. One has an exit code of 0, meaning that the procedure ran successfully. The other has a return code of 8, meaning that an error occurred while the procedure was running.

8. To insert lines in a procedure, move the cursor to the line you want to precede the new line, and press the Insert function key.

   Or, you can type `INSERT` on the QMF command line, move the cursor to the line you want to precede the new line, and press Enter.
9. To delete lines from a procedure, move the cursor to the line you want to delete and press the Delete function key.

   Or, you can type `DELETE` on the QMF command line, move the cursor to the line you want to delete, and press Enter.
10. To save the procedure in the database, enter:

    SAVE AS *procname*



```
PROC                                    MODIFIED     LINE     1

/* This procedure checks to see what day it is.  If it's
   Monday, it runs a query and prints a report.  If it
   isn't, a message is displayed informing the user.   */
signal on error
if date('w') = 'Monday' then
  do
    "RUN QUERY MYQUERY (FORM = MYFORM"
    "PRINT REPORT"
    "MESSAGE (TEXT='OK, MONDAY report has been created and sent to printer.'"
  end
else
  do
    "MESSAGE (TEXT='Sorry, it is not Monday.  Report cannot be created.'"
  end
exit 0       /*Exit without errors */
error:
  "MESSAGE (TEXT = '"dsq_message_text"'"
  exit 8      /*Exit with error condition*/
*** END ***
```

Figure 153. This procedure produces a commission report on Mondays.

In the procedure that is shown in Figure 153, the REXX DATE function provides the day of the week. The rest of the procedure includes QMF commands that are run depending on the day of the week.

Guidelines for writing procedures with logic

Keep the following guidelines in mind when you write procedures with logic:

- A procedure with logic can contain QMF commands, REXX logic statements, and comment lines.
- Use the complete names for commands, options, and values, rather than the abbreviated names.

- Do not specify a command of more than 2,000 characters (or the equivalent in DBCS). QMF stops running a procedure when it finds a command of over 2,000 characters.
- You can include comments on the same line as a command, but place them after the command.
- If a command spans more than one line, type a comma as a continuation character at the end of the first line. For example:

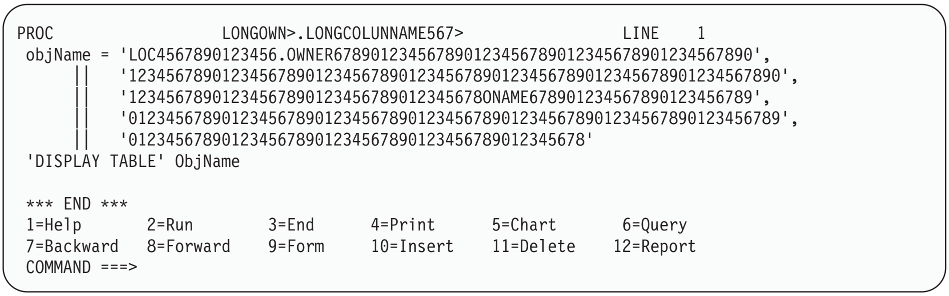
  ```
  "RUN QUERY MYQUERY (&&DEPT=38, ",
  "&&DIV='EASTERN'"
  ```

  Because this statement is a QMF command that is split into two lines, both lines are enclosed in quotes. The continuation character is placed at the end of the first line, outside the quotes.

  You cannot use a continuation character in a comment line, command keyword, or substitution variable name. You can use a continuation character in a substitution variable value, if you enclose the value in single quotes.

  You can type comments between continuation lines.

  Long names support: Follow the REXX syntax rules for continuation and concantenation. Here is an example of a long object name:



```
PROC                  LONGOWN>.LONGCOLUNNAME567>                LINE    1
 objName = 'LOC4567890123456.OWNER678901234567890123456789012345678901234567890',
       ||   '12345678901234567890123456789012345678901234567890123456789012345678890',
       ||   '1234567890123456789012345678ONAME67890123456789012345678 9',
       ||   '0123456789012345678901234567890123456789012345678901234567890123456789',
       ||   '01234567890123456789012345678901234567890012345678'
 'DISPLAY TABLE' ObjName

 *** END ***
 1=Help       2=Run        3=End      4=Print      5=Chart       6=Query
 7=Backward   8=Forward    9=Form     10=Insert    11=Delete    12=Report
 COMMAND ===>
```

Figure 154. Long names support for a procedure with logic

The proc object name at the top of the screen might be truncated if the authorization ID is longer than 8 characters or the object name is longer than 18 characters. Enter the SHOW NAME command from the command line to display a pop-up screen that contains the complete object name. See the SHOW NAME command for more details.

Running a procedure

To run a procedure, on the QMF command line, enter:

```
RUN PROC procname
```

The QMF commands you specify in a linear procedure run in the order they appear in the procedure. The QMF commands you specify in a procedure with logic run in the order that is specified by the logic of the procedure.

If the QMF commands in the procedure run a query or display a query or form, they change the contents of the temporary storage areas DATA, FORM, or QUERY. This occurs just as if you entered each command separately on the QMF command line.

If an error occurs while a linear procedure is running, QMF stops running the procedure. The PROC panel displays the command that contains the error the top of the panel. The error message at the bottom of the screen provides information on correcting the error.

If an error occurs while a procedure with logic is running, the logic of the procedure determines when the procedure ends and what displays. For more information, see "Using REXX error-handling instructions in procedures with logic" on page 194.

If ISPF is available on your system, you can use the QMF batch application to run the procedure while you are doing other work at your terminal. For more information, see *DB2 QMF Reference*.

Sharing a procedure with other QMF users

You can share a procedure with other QMF users just as you do other QMF objects, by saving it with the SHARE=YES parameter. Make sure that you also save any other QMF objects you specify in the procedure with the SHARE=YES parameter.

You can also set the DSQEC_SHARE global variable to share objects with other users globally. To set this global variable to allow other users to share your objects, enter this command:

```
SET GLOBAL (DSQEC_SHARE=1
```

To check whether an object is shared, use the LIST command to display the object. Move the cursor to the object name and press the Describe function key. If the object is shared, the value in the **Restricted** field on the object description panel is `No`. For more information on displaying a list of database objects, see Chapter 3, "Displaying a list of database objects," on page 27.

Make sure that you qualify the name of every shared object in a procedure with your user ID or the user ID of the person who owns it. Doing this ensures that other people use the correct procedure, in case they have procedures with identical names.

Creating reusable procedures with substitution variables

You can use substitution variables in &proclibs and procedures with logic, just as you can in queries.

A substitution variable is any variable that you can use in a QMF command; QMF manages these variables for you. A substitution variable is always preceded by an ampersand (**&**).

You can supply a value for a variable in the following ways:

- On the RUN command
- On a prompt panel
- On the SET GLOBAL command

Specifying values for variables on the RUN command

You can assign a value to a substitution variable by using the RUN command:

- In your linear procedure:
  ```
  RUN PROC SCHEDULE (&&TYPE='VACATION'
  ```
- In your procedure with logic:
  ```
  "RUN PROC SCHEDULE (&&TYPE='VACATION'"
  ```

You enclose the variable value `VACATION` in single quotes because the value is a character string. Precede the variable with **&&** to set the value on the RUN statement, or with **&** if the procedure prompts you for the value.

This value for the substitution variable is active only within the procedure that defines it. The value is not active in any procedure or module called from the defining procedure.

In the previous example, the value of `&&TYPE` is available only to the procedure that is called SCHEDULE.

Specifying values for variables using global variables

You can specify values for substitution variables by defining global variables with the SET GLOBAL command. A global variable keeps its value until you reset it, or until you end the QMF session.

For example, to set a global variable value for the &DEPARTMENT variable, enter:

```
SET GLOBAL (DEPARTMENT=38
```

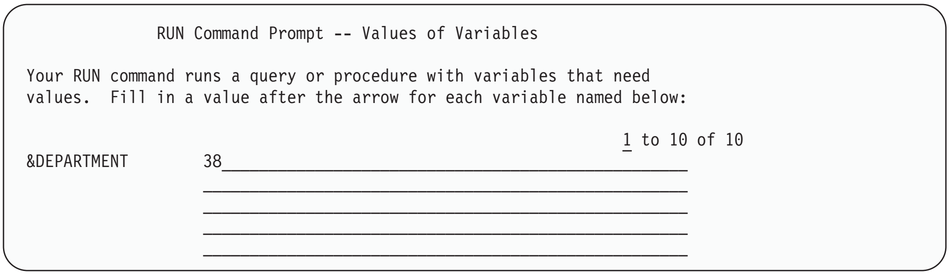
You can specify up to 10 variable values. Separate the values with commas or with blanks.

You can use the SET GLOBAL command to prompt for all the values in your procedure at the same time, as in the following example:

```
"SET GLOBAL (LASTNAME=&LASTNAME,DEPT_NUM=&DEPT_NUM";
```

Specifying values on the RUN command prompt panel

If you run a procedure that contains a substitution variable, and you do not assign a value to the variable using a global variable or on the RUN command, QMF displays a RUN command prompt panel. You can specify the value for the variable on this panel.



```
             RUN Command Prompt -- Values of Variables

Your RUN command runs a query or procedure with variables that need
values.  Fill in a value after the arrow for each variable named below:

                                                        1 to 10 of 10
&DEPARTMENT       38______________________________________________
                  ________________________________________________
                  ________________________________________________
                  ________________________________________________
                  ________________________________________________
```

Figure 155. Enter a value for a substitution variable.

This value for the substitution variable is active only within the procedure that defines it. The value is not active in any procedure or module called from the defining procedure.

In a linear procedure, QMF scans the procedure for substitution variables and resolves them before it processes any commands. It prompts you for all variable values before the procedure runs.

In a procedure with logic, QMF does not prompt you for variable values until REXX encounters the statement that contains the variables. For example, if your procedure with logic includes three statements that contain variables that QMF must prompt you for, QMF prompts you three times—once for each statement.

If you want a procedure with logic to prompt you for all the necessary variable values at one time, as the linear procedure does, use a dummy procedure. Suppose you want to be prompted once for the substitution variables LASTNAME and DEPT_NUM, which occur on two different lines in your procedure with logic as shown in Figure 156 on page 190.

```
/* This procedure runs two queries, displaying the report after each */
/* procedure has run.                         */

"RUN QUERY REG_QUERY (&&LASTNAME=&LASTNAME";
"INTERACT"
"RUN QUERY REG2_QUERY (&&DEPT_NUM=&DEPT_NUM";
```

Figure 156. This procedure requires two substitution variables.

Add the following line to the beginning of your procedure with logic, immediately following the comment lines:

```
"RUN PROC PROMPT_ME (&LASTNAME, &DEPT_NUM";
```

Where PROMPT_ME is a procedure with logic containing a comment line and no instructions, as shown in Figure 157.

```
/* PROMPT_ME is a dummy proc used by other procedures. */

"RUN PROC PROMPT_ME (&LASTNAME, &DEPT_NUM";
"RUN QUERY REG_QUERY (&&LASTNAME=&LASTNAME";
"INTERACT"
"RUN QUERY REG2_QUERY (&&DEPT_NUM=&DEPT_NUM";
```

Figure 157. This procedure prompts you to enter the substitution variables.

Using REXX variables in procedures with logic

You can use REXX variables in a procedure with logic. The values for these variables are known only within the procedure in which you defined them.

You can do the following:

- Copy a REXX variable to a QMF variable with the SET GLOBAL command
- Copy a global variable to a REXX variable with the GET GLOBAL command
- Use REXX variables in your REXX statements

For more information on REXX variables, see the REXX reference manual for your system. For details on the GET GLOBAL and SET GLOBAL commands, see *DB2 QMF Reference*.

QMF also provides a group of REXX variables for the callable interface that QMF sets after processing each QMF command. These variables provide important information about the results of each command. You can use them in your procedures with logic. For example, DSQ_RETURN_CODE is QMF's return code, and DSQ_MESSAGE_ID is QMF's completion message. For more information about these variables, see *Developing DB2 QMF Applications*.

Specifying REXX variables using SAY statements and PULL statements

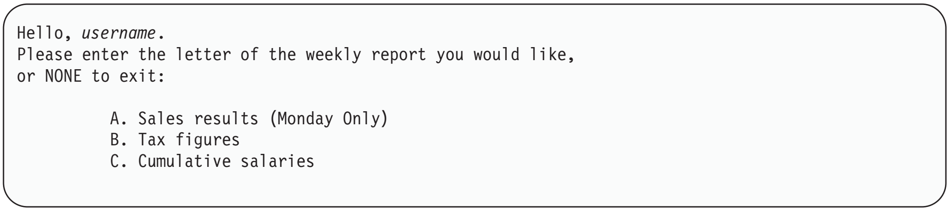
In a procedure with logic, you can use REXX SAY and PULL statements to prompt for variable values.

Use a SAY statement, or a sequence of SAY statements, to display text on the screen. For example, if you use the SAY statements in Figure 158:

```
say 'Hello,' whoisuser'.'
say 'Please enter the letter of the weekly report you would like, '
say 'or NONE to exit:'
say
say '              A. Sales results (Monday Only)'
say '              B. Tax figures'
say '              C. Cumulative salaries'
```

Figure 158. SAY statements prompt users to enter text.

The following appears:



```
Hello, username.
Please enter the letter of the weekly report you would like,
or NONE to exit:

          A. Sales results (Monday Only)
          B. Tax figures
          C. Cumulative salaries
```

Figure 159. The user prompts appear on the screen.

Specify a REXX PULL statement to retrieve the input from the screen and place it in the REXX variable answer as shown in Figure 160 on page 192.

```
/*    This procedure can produce any of three weekly reports
      regularly produced by the Acme Company—Sales, Tax,
      Cumulative Salaries, Inventory.  It prompts the user
      for the type of report wanted, runs the necessary
      queries, and checks for errors.                      */

arg report .               /* get any arguments from RUN PROC   */
ok = 'NO'                    /*   set variable for do loop        */
"GET GLOBAL (WHOISUSER = DSQAO_CONNECT_ID"     /*  identify user  */

if report = '' then          /*  check to see if no arg entered     */

/* if no arg entered, prompt user until A,B,C, or NONE is entered */
  do until ok = 'YES'

    say 'Hello,' whoisuser'.'
    say 'Please enter the letter of the weekly report you would like, '
    say 'or None to exit:'
    say
    say '                A. Sales results (Monday Only)'
    say '                B. Tax figures'
    say '                C. Cumulative salaries'

    pull answer                /*   get answer from user     */
    answer = strip(answer)   /* strip any leading or trailing blanks  */

    if answer = 'NONE' then exit 3     /*  exit immediately if NONE   */
    if pos(answer,'ABC') ¬= 0 then ok = 'YES'  /* if invalid value,   */
  end                                /* keep prompting.     */
else answer = report
```

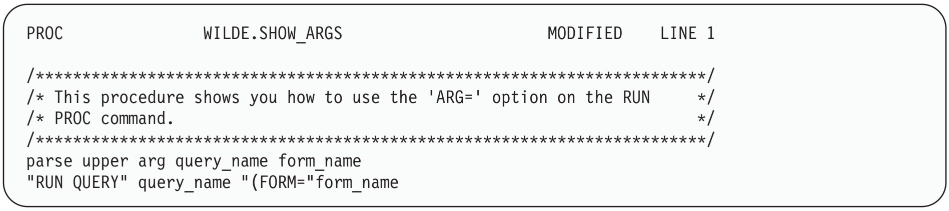
Figure 160. PULL statements take user input from the screen.

The exit code 3 was selected here to indicate the exit condition when the user enters None. As with any exit code, you choose the number to indicate an exit condition.

Passing values to a procedure with logic

For procedures with logic, use the ARG option on the RUN PROC command to pass arguments, or values, to a procedure with logic. You can also use the ARG option to pass values between procedures.

Use the ARG option when you are running a procedure that contains a REXX PARSE ARG statement or ARG statement, as in Figure 161 on page 193.



```
PROC              WILDE.SHOW_ARGS                   MODIFIED    LINE 1

/************************************************************************/
/* This procedure shows you how to use the 'ARG=' option on the RUN     */
/* PROC command.                                                        */
/************************************************************************/
parse upper arg query_name form_name
"RUN QUERY" query_name "(FORM="form_name
```

Figure 161. The ARG option passes values to a procedure with logic.

The RUN command for this procedure is:

```
RUN PROC SHOW_ARGS (ARG=(query_name form_name)
```

REXX variable and substitution variable differences

Table 10 shows the differences between REXX variables and substitution variables. It also shows how each is used in a procedure with logic.

Table 10. REXX variables versus substitution variables in a procedure with logic

REXX variables	Substitution variables
Name is made up of alphanumeric characters in lowercase or uppercase. `what_2_do`	Name must begin with an ampersand (&), followed by alphanumeric and special characters `&DEPARTMENT`
Can be used in REXX statements: `if progname = '' then`	Can be used in QMF commands: `"RUN QUERY MYQUERY (FORM = &FORMNAME"`

Can be given a value on the RUN PROC command using the QMF ARG parameter and the REXX ARG parameter:
`RUN PROC MYPROC (ARG=MONDAY`
`arg whichday`
	Can be given a value on the RUN PROC command:
	`"RUN PROC MYPROC (&&FORMNAME = MYFORM"`

Can be given a value using a QMF global variable and the QMF GET GLOBAL command: `"GET GLOBAL (WHO_IS_IT = DSQAO_CONNECT_ID"`	Automatically assigned a value by QMF at the time the command is run if a global variable by that name has been set (if the substitution variable has not already been given a value)
Can be used to set a global variable value using the QMF SET GLOBAL command: `"SET GLOBAL (JOBTYPE =" JOBVAR`	Cannot be used to set a global variable value

Using REXX error-handling instructions in procedures with logic

You can use REXX error-handling techniques, such as the REXX SIGNAL instruction, in a procedure with logic. In addition, you can use QMF commands and variables with the REXX EXIT instruction to help clarify nonzero return codes.

Branching to error-handling subroutines

The REXX signal on error instruction tells REXX to leave the current line and branch to a label marked *error* when a nonzero return code is encountered. This instruction requires two parts:

- Signal on error

  After every command, REXX puts the return code of the command in a variable that is called `rc`.

  If a command has a nonzero return code, REXX branches to the *error* label.

> **Note to TSO users and CMS users**
>
> *Signal on error* returns errors from the QMF REXX procedure (ADDRESS QRW) command environment, but not the REXX callable interface.

- *Error* label

  The signal on error instruction requires that you provide a label that the procedure can branch to if it encounters a nonzero return code. The label precedes your error-handling code. The return code is in the variable `rc`. You can use this variable to branch to another subroutine, or you can use it in your EXIT instruction, as shown in Figure 162.

```
/*  error handling code for a procedure with logic */
error:
  exit rc
```

Figure 162. QMF exits with a non-zero return code.

Using messages with the REXX EXIT instruction

You can use the REXX EXIT instruction to exit a procedure with logic. QMF always issues a message when it finishes running a procedure with logic. If you use the EXIT instruction, the message you see depends on these factors:

- Whether the last QMF command encountered an error
- Whether the return code was zero

Table 11 on page 195 shows which message you see based on the given conditions.

Table 11. Messages returned from QMF commands in procedures

Nonzero return code from the last QMF command	Procedure return code	Message at completion of procedure
No	0	`OK, your procedure was run`
No	nonzero	`The return code from your procedure was 8`
Yes	0	The error message provided by QMF
Yes	nonzero	The error message provided by QMF

An error message takes precedence over the return code message if you have an incorrect QMF command and a nonzero return code.

If you want to show the error message from the last command and exit with a nonzero return code, use the MESSAGE command as in Figure 163.

```
"MESSAGE (TEXT='"dsq_message_text"'"
exit rc
```

Figure 163. Specify MESSAGE to see the error message from the last command.

The variable `dsq_message_text` is a QMF-provided REXX variable. You can use the MESSAGE command and the `dsq_message_text` variable to store and display a message after further processing has occurred, as shown in Figure 164.

```
 /* Monthly report                                           */
   Signal on error
   "DISPLAY TABLE JUNE_INFO"
   "PRINT REPORT"
   Exit(0);
 Error:
   Original_msg = dsq_message_text    /* Saves error message. */
   "RUN PROC GENERAL_RECOVERY"        /* This proc generates  */
                                      /* new dsq_message_text. */
   "MESSAGE (TEXT='" Original_msg "'" /* Display original error msg. */
   Exit(8);
```

Figure 164. The MESSAGE command displays the original error message.

For more information on the MESSAGE command, see *DB2 QMF Reference*.

Calling REXX programs from a procedure with logic

You might have procedures that call applications. When you call your REXX callable interface application from a procedure with logic, be careful about the number of ampersands you specify for the substitution variables in your application. This is important if the program being called contains a RUN command with substitution variables, as in `RUN QUERY WEEKLY_Q (&&DEPT=58`.

Calling REXX programs without substitution variables

If your REXX program does not contain an embedded RUN command that includes substitution variables, use one of the following methods to start your program:

- The ADDRESS instruction

  This instruction establishes a command environment. For more information on command environments, see *DB2 QMF Reference*. If you want to call a program that is named PANDA from within the CMS environment, enter this command:

  ```
  ADDRESS CMS "PANDA"
  ```

- The CALL instruction

  This instruction starts a program. For the program named PANDA, the command is:

  ```
  CALL PANDA
  ```

- A function

  You also can call the program PANDA as a function:

  ```
  ANSWER = PANDA()
  ```

For more information on any of these commands, see the REXX reference manual for your system.

You might consider removing the substitution variables from the RUN command if you want to call your programs by using one of the REXX invocation calls. In that case, QMF prompts the user for the variables.

Calling REXX programs that contain substitution variables

If your REXX application contains a QMF RUN command with a substitution variable, you must start it using either `CMS program_name` or `TSO program_name`.

Whether you are running a procedure with logic or a callable interface program that is invoked by a procedure with logic, commands come into QMF the same way. In this context, the callable interface program becomes a logical extension of the procedure itself.

Consider the command:

```
RUN QUERY WEEKLY_Q (&DEPT=58
```

In a procedure with logic, use two ampersands on the substitution variable to pass the variable to the query:

```
"RUN QUERY WEEKLY_Q (&&DEPT=58"
```

If a substitution variable has only one ampersand, QMF resolves the variable for the procedure itself, and cannot pass the variable to the query.

If you call a REXX callable interface application from a procedure with logic, and that application contains the command `RUN QUERY WEEKLY_Q (&DEPT=58`, QMF resolves the variable just as it would for the calling procedure. Because the statement contains only one ampersand, the variable is not passed to the query.

To pass variables to QMF from a REXX callable interface application that is called by a procedure with logic, you have three choices:

- Use the CMS or TSO command to call the application.

  When you call the application, QMF does not process any substitution variables it encounters. In the preceding command, &DEPT=58 is passed to the query, where the substitution variable is resolved.
- Treat all substitution variables in your application as though you were using them in a procedure with logic.

  Add an ampersand to every substitution variable so the procedure with logic does not resolve it.
- Use global variables.

  You can define global variables at the start of your application and use them throughout your QMF session.

Connecting to a remote location from a procedure

The QMF CONNECT command lets you connect to another user ID or to a remote DB2 or DB2 Server for VSE or VM database to use the remote unit of work support. You can use this command with a linear procedure or a procedure with logic.

You cannot use the CONNECT command from DB2 for VSE. However, you can use DB2 for VSE as a server and connect to it from either DB2 or DB2 for VM.

In the following example, suppose that you are an administrator in Miami, and you want to write a procedure that:

- Connects to a remote location (DALLAS)
- Issues a series of QMF commands
- Produces a report

- Reconnects to the originating location (MIAMI)

The procedure looks like the one that is shown in Figure 165.

```
CONNECT TO DALLAS                          -- SQL executed in Dallas
RUN PROC GENERATE_REPORT (FORM=GEN_FORM    -- Issue QMF commands
PRINT REPORT                               -- Report printed in Miami
CONNECT TO MIAMI
```

Figure 165. This procedure uses the CONNECT command.

Make sure that you store the procedure at the current location, in the same database to which you are connected when you issue the RUN PROC command. When you connect to a new location, QMF re-initializes your profile, except for the value of TRACE. It also re-initializes command synonyms and function keys to the values at the new (current) location.

When you write procedures that use the QMF CONNECT command to access remote databases, keep the following guidelines in mind:

- If you are connected to a remote database and issue a RUN PROC command, that procedure and all the objects used in that procedure must be stored at the remote database.
- All QMF commands in the procedure run in QMF temporary storage at the system where QMF is running (the local system). However, all objects used by these QMF commands (such as queries, procedures, or forms) must be defined in the database at the current location (the remote system).
- All commands that affect the database (for example, SQL statements, QMF queries, or EDIT TABLE updates) run at the current location.
- If the procedure contains system-specific commands (CICS, CMS, or TSO), these commands run at the system where QMF is running (the local system).

  If your procedures contain system-specific commands that do not run on the system where QMF runs, your procedure will not run successfully.
- Any files or data sets that are used in a system-specific command must exist on the system where QMF is running (the local system).

For more information about using the QMF CONNECT command and remote unit of work support, see the *DB2 QMF Reference* manual.

Writing a procedure that creates a query

The example in this section shows you how to write a procedure with logic to "fill in" a template SQL statement to create a query.

The sample procedure:

- Checks the day of the week
- Sets the values of the variables passed to the query if the day is Friday
- Runs the query

Writing a template SQL statement

You can write a template SQL statement that can accept different values for the column names and row conditions. For this scenario, create the following query and save it as SENIORSTAFF:

```
SELECT &SELECT1
FROM Q.STAFF
WHERE &COND1
```

This query allows the user or a procedure to specify the column names and row conditions just before running the query.

Using a procedure, you can assign values to the QMF query substitution variables (&SELECT1 and &COND1) by using one of the following procedures:

- Pass the substitution variable values to the query on the RUN QUERY command. To write this type of procedure, see "Passing variables to the template query."
- Set global variable values. To write this type of procedure, see 201.

Both of the procedures that are described in this scenario produce the same results.

Passing variables to the template query

You can write a procedure that sets REXX variable values and passes these values to your template SQL statement. The QMF procedure in Figure 166 on page 200 passes the substitution variable values to the query on the RUN QUERY command.

```
/*  REXX PROC  */
IF DATE('W') = 'Friday' THEN
  DO
    sel = '(NAME, JOB, SALARY, COMM)'
    con1 = "((SALARY > 15000) OR (JOB = 'MGR'))"
  END
ELSE
  DO
    sel = '*'
    con1 = '(DEPT=51)'
  END

"RUN QUERY SENIORSTAFF (&&SELECT1 ="sel",&&COND1 ="con1
```

Figure 166. The procedure passes values on the RUN QUERY command.

Because this procedure assigns values to the substitution variables (SELECT1 and COND1) on the RUN QUERY command, you must use a double ampersand before the variable names to tell REXX that these variables are assigned in the procedure, but not used in the procedure.

If you use only one ampersand before the variable name, as in this statement:

```
"RUN QUERY (&SELECT1 ="sel",&COND1 ="con1
```

QMF assumes that the variables are procedure variables, rather than variables to be passed to the query, and prompts you for their values when you run the procedure.

In the following lines from this procedure, the procedure assigns a character string to a REXX variable:

```
    con1 = "((SALARY > 15000) OR (JOB = 'MGR'))"

    con1 = '(DEPT=51)'
```

These values are then passed to the query on the RUN QUERY command. The values in the first REXX variable assignment, SALARY and JOB, are enclosed in double parentheses because the character strings passed to the query contain single parentheses and an equal sign. For the complete rules about using parentheses around character strings that are passed on a RUN command, see the *DB2 QMF Reference* manual.

When you run this procedure on a Friday, the procedure sets the substitution variables and passes the values to the query so that QMF runs the following query:

```
SELECT NAME, JOB, SALARY, COMM
FROM Q.STAFF
WHERE (SALARY > 15000) OR (JOB='MGR')
```

If you run this procedure on any day other than Friday, QMF runs the following query:

```
SELECT *
FROM Q.STAFF
WHERE DEPT = 51
```

Writing a procedure that sets global variables for the template query

You can write a procedure that sets global variable values according to REXX logic. These values are then available to the template query when the procedure issues the QMF RUN QUERY command.

The procedure in Figure 167 sets the query variables as global variables. The results are the same as those explained in "Passing variables to the template query" on page 199.

```
/*  REXX PROC  */

IF DATE('W') = 'Friday' THEN
  DO
    "SET GLOBAL (SELECT1 = 'NAME, JOB, SALARY, COMM'"
    "SET GLOBAL (COND1 = '(SALARY > 15000) OR (JOB = 'MGR'')'"
  END
ELSE
  DO
    "SET GLOBAL (SELECT1 = '*'"
    "SET GLOBAL (COND1 = '(DEPT = 51)'"
  END

"RUN QUERY SENIORSTAFF"
```

Figure 167. The procedure sets query variables as global variables.

Running procedures in batch

> **Note to CICS Users**
>
> Because ISPF does not run in the CICS environment, you cannot use the QMF BATCH command.

In QMF batch mode, you can run both linear procedures and procedures with logic in the z/OS and VM environments while you perform other work at your terminal. You can run procedures in batch at any time, and you do not have to interact with QMF while the procedure is running. ISPF is required to use the QMF BATCH application.

Creating a procedure to run QMF commands

To run a procedure in batch mode, first create and save the procedure, just as you would to run it interactively. Then use the QMF batch application, which simplifies batch processing. The application prepares and submits the batch job from information that you enter on the batch prompt panel. You only need to know the name of the procedure and a few details about the batch machine on your system. However, it still might be necessary to contact your information center to have the application tailored to your needs.

Writing batch-mode procedures

The rules for writing batch-mode procedures are more restrictive than those for writing interactive procedures. The restrictions avoid situations in which user interaction is required. Before discussing these restrictions, you need to know two new terms:

- The *main procedure* is the one that is identified on the ISPSTART command that starts QMF for batch mode.
- A *subordinate procedure* is one that is called directly from the main procedure or from another subordinate procedure.

Restrictions

The following restrictions apply equally to main and subordinate procedures unless otherwise indicated.

- Do not write incomplete commands.

  In batch mode, QMF has no way of prompting you for the complete command.
- Do not try to directly access command prompt panels. (Do not issue commands that use the question mark to obtain the command prompt panels.)
- Do not issue commands that might cause the display of confirmation panels.

  These are commands that erase, update, or replace database objects, or that replace exported files. A confirmation panel asks you whether you want to make a change. In batch mode, QMF has no way of handling such prompts.

  You can still issue commands that erase or change data objects, but you must inhibit the confirmation prompt.

  To inhibit the confirmation panel, include CONFIRM=NO or issue the command:

  ```
  SET PROFILE (CONFIRM=NO
  ```
- Avoid situations that might display the incomplete data prompt.

  QMF has no way to prompt you in batch mode.
- Save the main batch procedure, specifying SHARE=YES. If you have issued a SET GLOBAL command with the value DSQEC_SHARE=1 prior to this SAVE command, you do not need to specify the SHARE=YES parameter.

If you are using a National Language Feature (NLF): You are writing a sequence of QMF commands that the NLF must understand. The verbs and keywords in the commands must be the translated versions of their English-language counterparts: ANZEIGEN for DISPLAY, for example, in a German batch mode procedure, and PROZEDUR for PROC.

Example for VM

The following main procedure illustrates some of the restrictions on batch procedures in the VM environment:

```
CONNECT userid (PASSWORD = mypass
RUN MYQUERY (FORM = myform
SAVE DATA AS MYTABLE (CONFIRM = no
CMS CP SP PRT TO USERID
PRINT REPORT
CMS CP SP PRT CLOSE
```

CONNECT
Gives the CMS batch machine the same authorization (via a password) as the user ID associated with submitting the batch work- that user ID must be authorized to connect to DB2 Server for VM and have a password in SYSTEM.SYSUSERAUTH

RUN Runs a stored query with a stored form

SAVE Saves the data in the database

CMS CP SP PRT
Sends output to a user ID instead of a printer

PRINT
Prints a report based on the query results

CMS CP SP PRT CLOSE
Ends printing

Example for z/OS

The following main procedure illustrates some of the restrictions on batch procedures for the z/OS environment:

```
SET PROFILE (CONFIRM=NO
RUN QUERYA (&&LICENSE='007'
PRINT REPORT (PRINTER='
SAVE DATA AS TABLEA
RUN PROCA (&&TABLE=TABLEA
EXIT
```

SET Eliminates the possible display of confirmation panels- in batch mode, such a display produces an error

RUN QUERYA
Passes the value 007 to QUERYA for the substitution variable &LICENSE- if QUERYA contained other substitution variables, the run would fail

The object names in this command are not qualified with the owner's name. Their owner is, therefore, the person for whom the procedure is being run; that is, the person whose logon ID appears as the USER parameter on the JOB card.

PRINT
Prints a report based on the query results- the output goes to the DSQPRINT data set

SAVE Saves the data in the database

The SAVE command does not need to contain CONFIRM=NO because of the SET PROFILE command at the start of the procedure. If the DATA object is too large for the storage that is reserved for it, the SAVE command might end the procedure through the incomplete-data prompt condition.

RUN PROCA
Runs a procedure that does something with TABLEA (the table that was just created or replaced by the SAVE command)

The name of this table is passed to the procedure through the &TABLE parameter. This command fails if the procedure called has other substitution variables not set.

EXIT Ends the procedure and QMF

Using IMPORT/EXPORT commands

When you export an object and later import it, refer to the data set name consistently. Always refer to it using the unqualified or fully qualified name. Otherwise, problems can arise.

Using the EXIT command in QMF procedures

QMF stops after the EXIT command runs.

A procedure also stops after it runs the command on its bottom line. If this command is not EXIT, one of three things happens:

- For a subordinate procedure, control is returned to the calling procedure without ending QMF. This is true in both batch and interactive modes.
- For a main procedure in batch mode, QMF is ended.
- For a main procedure in interactive mode, control returns to the user, in QMF (unless the procedure is an initial procedure).

Ending a main procedure in batch mode always ends QMF. This is why the sample batch procedure does not need the EXIT command.

Effect of errors

Any error encountered while running a linear procedure ends the procedure. The logic in your procedure handles any errors encountered while running a procedure. For more information, see “Using REXX error-handling instructions in procedures with logic” on page 194.

Chapter 9. Making QMF objects reusable

In QMF, a global variable keeps its value from the time it is set until you either reset it or end your QMF session. Use global variables to assign changing values to substitution variables in queries, procedures, and forms. You can also use global variables to change certain behavioral aspects of your QMF session, such as displaying confirmation panels in the Table Editor.

Each global variable has a name and a value. “Creating, changing, and deleting global variables using commands” on page 209 discusses limitations for the lengths of names and values. Some variable names are reserved for use by QMF. These names begin with the letters DSQ.

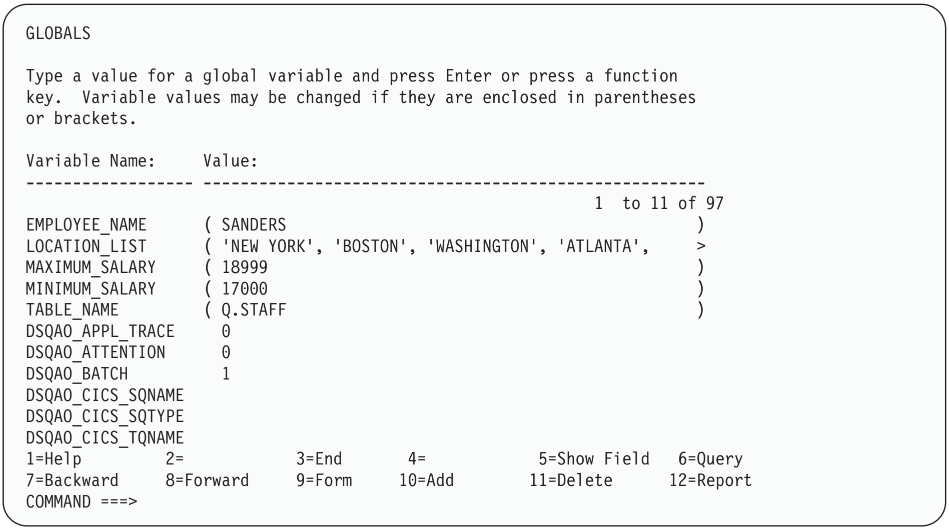
Variable values used in queries cannot begin with dashes because the database misinterprets them. The command for viewing global variables is SHOW GLOBALS. This command lists global variables and their values. From the global variable list, you can change or delete an existing global variable or add a new one.

You can also use the SET GLOBAL and RESET GLOBAL commands from the QMF command line to set and delete global variables without displaying the global variable list.

This chapter describes how to use the global variable list and the SET GLOBAL and RESET GLOBAL commands. For information and examples on other aspects of using variables in queries, forms, or procedures, see “Making your query reusable with substitution variables” on page 58, “Making your query reusable with substitution variables” on page 108, “Using a global variable in a heading or footing” on page 137, and “Specifying values for variables using global variables” on page 188.

Creating, changing, and deleting variables from the global variable list

The easiest way to display, change, add, or delete global variables is with the SHOW GLOBALS command. When you enter `SHOW GLOBALS` on the QMF command line, QMF displays a global variable list panel similar to Figure 168 on page 208.



```
GLOBALS

Type a value for a global variable and press Enter or press a function
key.  Variable values may be changed if they are enclosed in parentheses
or brackets.

Variable Name:     Value:
------------------ ------------------------------------------------------
                                                          1  to 11 of 97
EMPLOYEE_NAME      ( SANDERS                                            )
LOCATION_LIST      ( 'NEW YORK', 'BOSTON', 'WASHINGTON', 'ATLANTA',     >
MAXIMUM_SALARY     ( 18999                                              )
MINIMUM_SALARY     ( 17000                                              )
TABLE_NAME         ( Q.STAFF                                            )
DSQAO_APPL_TRACE     0
DSQAO_ATTENTION      0
DSQAO_BATCH          1
DSQAO_CICS_SQNAME
DSQAO_CICS_SQTYPE
DSQAO_CICS_TQNAME
1=Help         2=            3=End       4=           5=Show Field   6=Query
7=Backward     8=Forward     9=Form     10=Add       11=Delete      12=Report
COMMAND ===>
```

Figure 168. The Globals panel

The global variable list panel uses one row on the screen for each global variable. The variable name appears on the left, and up to the first 50 characters of the variable value appear on the right. Variables you defined appear in alphabetic order first. Then, QMF DSQ variables appear in alphabetical order.

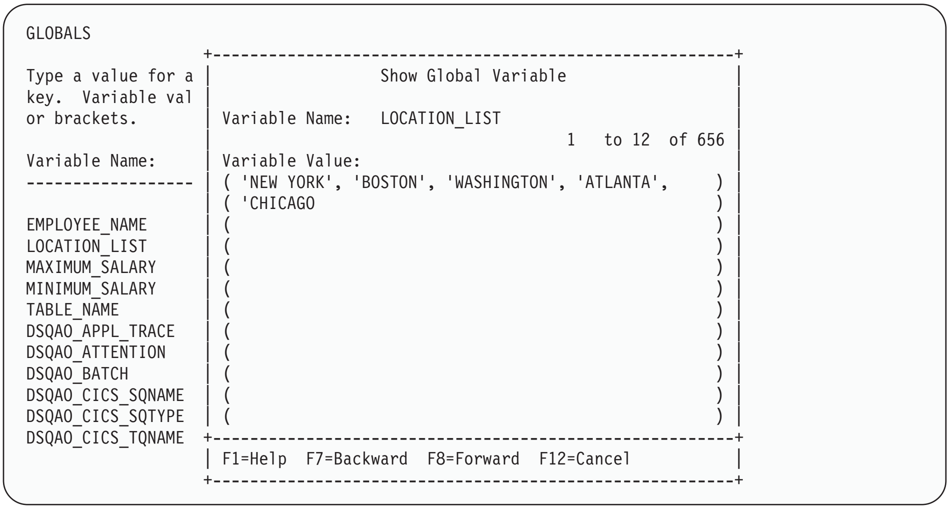
Global variables added on the SHOW GLOBALS panel can have a length of up to 32,768 characters. Variable values that are longer than a single line are indicated by a greater-than (>) sign to the right of the value.

Changing a variable value

Variable values that you can change appear within parentheses. To change a variable value, type over the displayed value and press Enter.

Some DSQ variables have a restricted set of acceptable values. For example, the variable DSQDC_COST_EST (which controls the display of the database cost estimate) must have a value of either 0 or 1. See the global variable tables in*DB2 QMF Reference* or *Developing DB2 QMF Applications* for more information.

If the variable value is too long to be displayed completely (indicated by a greater-than sign (>) in the right margin), or if you want to change a variable to a value greater than 50 bytes, move the cursor to the line containing the variable name. Then, press the Show Field function key. This displays the Show Global Variable panel, and the entire variable value appears in a scrollable area.



```
GLOBALS
                    +--------------------------------------------------------+
Type a value for a  |                   Show Global Variable                 |
key.  Variable val  |                                                        |
or brackets.        | Variable Name:   LOCATION_LIST                         |
                    |                                        1   to 12  of 656 |
Variable Name:      | Variable Value:                                        |
------------------  | ( 'NEW YORK', 'BOSTON', 'WASHINGTON', 'ATLANTA',     ) |
                    | ( 'CHICAGO                                           ) |
EMPLOYEE_NAME       | (                                                    ) |
LOCATION_LIST       | (                                                    ) |
MAXIMUM_SALARY      | (                                                    ) |
MINIMUM_SALARY      | (                                                    ) |
TABLE_NAME          | (                                                    ) |
DSQAO_APPL_TRACE    | (                                                    ) |
DSQAO_ATTENTION     | (                                                    ) |
DSQAO_BATCH         | (                                                    ) |
DSQAO_CICS_SQNAME   | (                                                    ) |
DSQAO_CICS_SQTYPE   | (                                                    ) |
DSQAO_CICS_TQNAME   +--------------------------------------------------------+
                    | F1=Help  F7=Backward  F8=Forward  F12=Cancel           |
                    +--------------------------------------------------------+
```

Figure 169. The Show Global Variable panel

To change a variable value in the Show Field panel, type over the displayed value and press Enter. To close the window without changing the variable value, press the Cancel function key.

Adding a new variable

To add a new variable from the global variable list, press the Add function key. This displays the Add Global Variable panel that contains empty entry fields for the variable name and its value. To create a new variable, fill in the name and value fields and press Enter. To close the window without creating the variable, press the Cancel function key.

Removing a variable

To delete a global variable from the global variable list, move the cursor to the line that contains the variable name and press the Delete function key.

You cannot delete variables that begin with DSQ.

Using the global variable list and the CASE option

When you change or add a global variable from the global variable list, the name and value might be changed to uppercase, depending on the CASE option in your QMF profile. The variable name is changed to uppercase if the CASE option is set to UPPER or STRING. The variable value is changed only if the CASE option is set to UPPER.

Creating, changing, and deleting global variables using commands

You can set and delete global variables from the QMF command line by using these commands:

SET GLOBAL

Lets you create or change up to ten global variables

For example, to set a new global variable JOBTYPE with the value SECRETARY, enter the following command on the QMF command line:

```
SET GLOBAL (JOBTYPE='SECRETARY'
```

In QMF Version 7.2, the SET GLOBAL command has been modified. It now can copy from another global value:

```
SET GLOBAL(Variablename = &Variablename
```

If you use linear syntax for the SET GLOBAL command, the maximum length of the value is 55 characters. If you use the extended syntax for this command, the maximum length is 32,768 characters. For more information on the extended syntax of the SET GLOBAL command, see *Developing DB2 QMF Applications*.

RESET GLOBAL

Lets you delete some or all of your global variables- to delete a global variable, enter:

```
RESET GLOBAL (JOBTYPE
```

To delete all the global variables you created, enter:

```
RESET GLOBAL ALL
```

For full command syntax and other specific information on using QMF commands on global variables, see *DB2 QMF Reference*.

Chapter 10. Creating tables

You can create your own tables by using SQL statements. The examples in this chapter show you how. The syntax of the SQL statements that are shown might vary slightly depending on what database management system you use. For the exact syntax, see the SQL reference manual for your database management system.

You can only create tables at your current location. To create tables at a remote location, use the CONNECT command. The remote location becomes the current location, from which you can create tables.

You can also create tables in a Windows environment by using the DB2 QMF for Windows feature. See Appendix C, "The DB2 QMF High Performance Option," on page 367 for more information.

Planning for your table

Make sure that you are authorized to create tables. Before you create a table, you need the following information:

- The *spacename* (called *tablespacename* in DB2 and *dbspacename* in DB2 Server for VSE or VM) into which you can save your tables- check with your QMF administrator for this information
- The name of the table
- The columns you want to include in the table, and the data type of each column

Creating a table

To create a table, use the SQL CREATE TABLE statement:

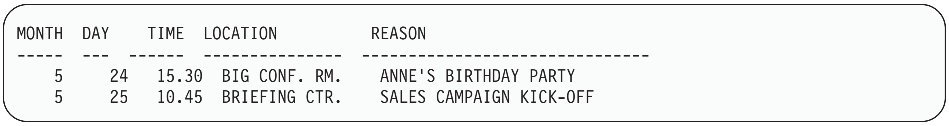
```
CREATE TABLE tablename
  (columnname  datatype definition,
   columnname  datatype definition)
   IN spacename
```

Where:

- *tablename* is the name of the table
- *columnname* is the name of a column
- *datatype* is the data type of the data you are using in that column
- *definition* (optional) describes whether the column can contain nulls

- *spacename* is the dbspacename (DB2 Server for VSE) or tablespacename (DB2) where you store the table

The example in Figure 170 shows you how to create a table for an appointment calendar. The table name is CALENDAR. There are columns for the month, day, time, location, and reason for the event.



```
MONTH  DAY    TIME  LOCATION          REASON
-----  ---  ------  ---------------  ------------------------------
    5    24   15.30  BIG CONF. RM.    ANNE'S BIRTHDAY PARTY
    5    25   10.45  BRIEFING CTR.    SALES CAMPAIGN KICK-OFF
```

Figure 170. This table contains data for an appointment calendar.

Specifying NOT NULL prevents you from entering an appointment without a MONTH, DAY, TIME, and LOCATION. Specify a data type (character, numeric, or date/time) for each column. You must specify *spacename* when you create a query.

There are often several ways to specify columns and data types for a table. In this example, you can combine the MONTH and DAY columns into one column and use the DATE data type. Or, you can use the TIME data type for the time column.

If you use DATE and TIME data types, your CREATE TABLE statement looks like the one that is shown here:

```
CREATE TABLE CALENDAR
  (CALDATE   DATE   NOT NULL,
   TIME      TIME   NOT NULL,
   LOCATION  VARCHAR(15)  NOT NULL,
   REASON    VARCHAR(36))
IN space-name
```

For more information on data types, see the SQL reference manual for your database management system.

Saving and appending to a table

To save your table in the database, enter:

```
SAVE DATA AS tablename
```

If you want to append the table to an existing table, enter:

```
SAVE DATA AS tablename (ACTION=APPEND
```

Where *tablename* is the name of the table to which you want to append the new table.

For example, to append a table that is called NEWAPPTS to the existing table CALENDAR, enter:

```
DISPLAY TABLE NEWAPPTS
SAVE DATA AS CALENDAR (ACTION=APPEND
```

The new table must have the same number of columns and the same data types as the existing table.

Creating a copy of a table

You can create a table by copying the data from an existing table into a new table. Use SQL statements or QMF commands to create a copy of a table.

To create a copy of a table using QMF commands:

1. Enter `DISPLAY TABLE` *`tablename`* where *tablename* is the name of the table you want to copy.

   For example, if you want to create a new table from a copy of the CALENDAR table, enter `DISPLAY TABLE CALENDAR`.

   The table you want to copy displays.
2. Enter `SAVE DATA AS` *`tablename`*, where *tablename* is the name of the new table.

   For example, enter `SAVE DATA AS MYCAL`, to name the new table MYCAL.

   QMF creates a new table with the same data as the old table. In this example, MYCAL and CALENDAR both exist in the database with the same data.

Creating a view of a table

You can create a view that includes some or all of the columns in one or more tables. You can use a view just as you would a table. When you update the tables or tables you used to create the view, the view is also updated. When you update the view, the tables are updated.

You might find it useful to create a view of a table when you want a portion of a table to remain hidden. Creating a view from portions of several tables can simplify query development because you only have to specify that view instead of selecting several tables and joining them.

The following example shows how to create a view of the CALENDAR table, called MYCAL, with the REASON column omitted.

```
CREATE VIEW MYCAL
  (CALDATE, TIME, LOCATION)
AS SELECT CALDATE, TIME, LOCATION
  FROM CALENDAR
```

Creating a synonym for a table or view

You can create a synonym for the name of a table or view by using the CREATE SYNONYM statement. Then when you refer to that table or view, you will not have to specify the fully-qualified table name.

For example, to create a synonym for the CALENDAR table, enter:

```
CREATE SYNONYM CALEN FOR CALENDAR
```

Depending on your database configuration, you might need to specify an owner qualifier when you specify the table. In that case, enter:

```
CREATE SYNONYM CALEN FOR userid.CALENDAR
```

Now you can use the synonym where you previously specified the table name. In the previous example, you can specify CALEN instead of CALENDAR.

If you share a query that uses a synonym, the users you share it with must define the same synonym before they can run the query.

Creating an alias for a table or view

If you have CREATEALIAS privilege or SYSADM or SYSCTRL authority, you can create an alias for the name of a table or view by using the CREATE ALIAS statement.

For example, to create an alias for the CALENDAR table, enter:

```
CREATE ALIAS CALEN FOR CALENDAR
```

Depending on your database configuration, you might need to specify an owner qualifier when you specify the table. In that case, enter:

```
CREATE ALIAS CALEN FOR userid.CALENDAR
```

You can use an alias in the same way you use a synonym. The difference between a synonym and an alias, however, is that a synonym can be used only by its owner, and an alias can be used by its owner and other users.

If you share a query that uses an alias, the users you share it with do not have to define the same alias before they can run the query.

Deleting tables, views, synonyms, and aliases

You can use either the QMF ERASE command or the SQL DROP statement to erase tables, views, synonyms, and aliases from the database.

For example, to use the QMF ERASE command to delete the CALENDAR table, enter:

```
ERASE TABLE CALENDAR
```

To use the SQL DROP statement to erase the same table, run this query:

```
DROP TABLE CALENDAR
```

When you use either the DROP statement or the ERASE command to delete a table from the database, any views or synonyms you created from them are also dropped.

To erase a table, you must be the owner of the table or have DBADM authority.

To erase a view or an alias, you must be the owner or have SYSADM or SYSCTRL authority.

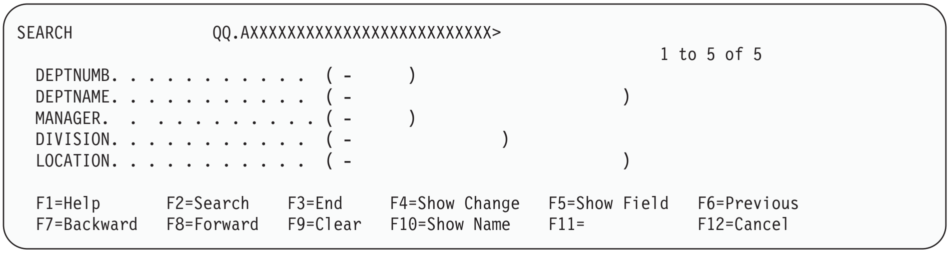
To erase a synonym, you must be the owner of the synonym.

Chapter 11. Maintaining the data in your tables

After you create your tables, you will want to add to or make changes to the data in them. Using the QMF Table Editor or SQL statements, you can easily make updates to the information in your tables.

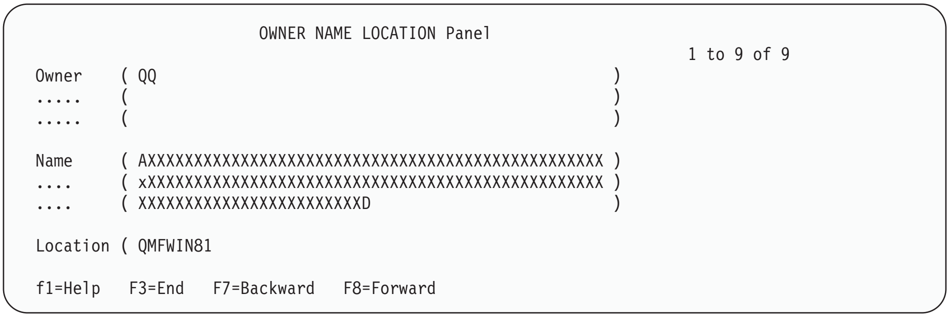
Using the Table Editor

For DB2 QMF Version 8.1, PF Key 10 is assigned *Show Name* on the EDIT TABLE dialog for both the ADD and SEARCH modes. When pressed, PF10 displays the complete name of the table being edited as opposed to the truncated table name on the Edit Table screen:



```
SEARCH               QQ.AXXXXXXXXXXXXXXXXXXXXXXXXXX>
                                                                     1 to 5 of 5
  DEPTNUMB. . . . . . . . . . .  ( -      )
  DEPTNAME. . . . . . . . . . .  ( -                       )
  MANAGER.  .  . . . . . . . . . ( -      )
  DIVISION. . . . . . . . . . .  ( -               )
  LOCATION. . . . . . . . . . .  ( -                       )

  F1=Help       F2=Search    F3=End     F4=Show Change   F5=Show Field   F6=Previous
  F7=Backward   F8=Forward   F9=Clear   F10=Show Name    F11=            F12=Cancel
```

Figure 171. Edit Table screen



```
                          OWNER NAME LOCATION Panel
                                                                        1 to 9 of 9
  Owner    ( QQ                                                   )
  .....    (                                                      )
  .....    (                                                      )

  Name     ( AXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX )
  ....     ( xXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX )
  ....     ( XXXXXXXXXXXXXXXXXXXXXXXXD                          )

  Location ( QMFWIN81

  f1=Help   F3=End   F7=Backward   F8=Forward
```

Figure 172. Show Name screen

Deciding when to save your data

When you start a Table Editor session, you can specify whether you want to save each addition or change in the database as you make it, or whether you want to hold all your additions or changes and save them when you end the Table Editor session.

You specify when you want to save additions or changes by using the SAVE keyword when you type the EDIT TABLE command that begins a Table Editor session.

If you want to save additions or changes as you make them, use SAVE=IMMEDIATE. This option is only available if your database management system supports CURSOR HOLD. See your QMF administrator to find out if you can use the SAVE=IMMEDIATE option.

If you want to hold additions or changes and save them at the end of the Table Editor session, use SAVE=END. Because SAVE=END is the default for the EDIT TABLE command, you do not have to type anything if you want to hold additions or changes. You will see examples of how to enter the EDIT TABLE command in the sections that follow.

If you specified that you want to see confirmation panels (CONFIRM=YES) either in your QMF user profile or when you began this Table Editor session, you will see different confirmation panels depending on when you decide to save the data.

Adding rows

To add rows to a table using the Table Editor:

1. Do one of the following tasks, depending on where you are starting:
   - From the QMF Home panel, type:

     ```
     tablename (MODE=ADD
     ```

     then, press the Edit Table function key.
   - From any other QMF panel with a command line, enter:

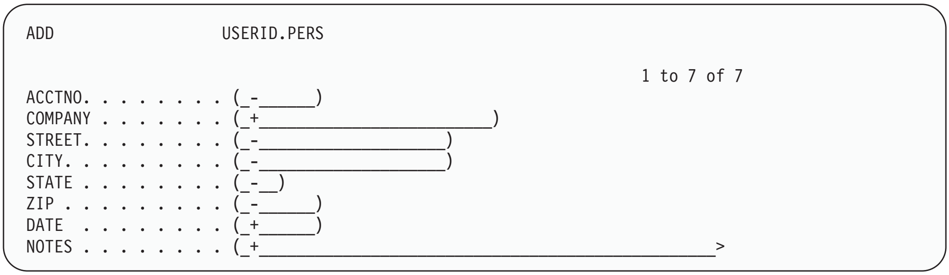
     ```
     EDIT TABLE tablename (MODE=ADD
     ```

   For example, to add rows to the PERS table from the QMF Home panel, type `PERS (MODE=ADD`, and press the Edit Table key.

   To enter the same command and save each addition as you make it, enter:

   ```
   PERS (MODE=ADD SAVE=IMMEDIATE
   ```

   The Table Editor ADD panel displays, showing the name of each column in the table, followed by an entry field where you enter new data for that column.



```
ADD                 USERID.PERS

                                                 1 to 7 of 7
ACCTNO. . . . . . . . (_-______)
COMPANY . . . . . . . (_+______________________)
STREET. . . . . . . . (_-_________________)
CITY. . . . . . . . . (_-_________________)
STATE . . . . . . . . (_-__)
ZIP . . . . . . . . . (_-______)
DATE  . . . . . . . . (_+______)
NOTES . . . . . . . . (_+_____________________________________________>
```

Figure 173. The Table Editor ADD panel

On this panel:

- The name of the table you are editing and the user ID of the table owner appear at the top of the panel.
- The columns displayed on this panel make up one row in the table.
- A null indicator (not the same as zero or blank) or a column default indicator (if available for the column) is displayed in each field to indicate that nothing has been entered. These indicators are configurable. See “Specifying column default and null indicator” on page 220 for more information.

  A blank or a zero means that a blank or zero value has been entered for that column.
- The scroll indicator tells you how many columns are in a row, and how many columns appear on the panel.

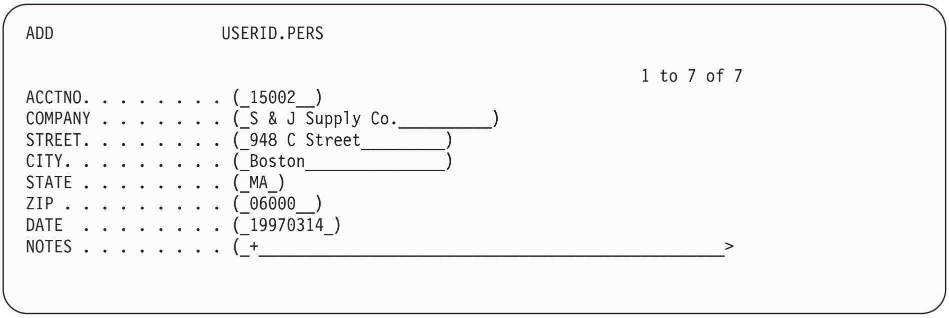
To move a specific column to the top of the panel, type its number in the first position of the scroll indicator. Press the Forward key to see the rest of the columns. Table Editor panels do not have a command line, so press the appropriate function key for the command you want to issue.

You can display the default settings for the fields by using the Show Field PF key. This is helpful if you have typed over the original values and forgotten what they were.

2. Type the information in each field as it appears in Figure 174 on page 220.

   Use the Tab key to move from field to field.

   If you need to know what values are valid for a field, press the Show Field function key.



```
ADD                  USERID.PERS


                                                                  1 to 7 of 7
ACCTNO. . . . . . . . . (_15002__)
COMPANY . . . . . . . . (_S & J Supply Co.__________)
STREET. . . . . . . . . (_948 C Street_________)
CITY. . . . . . . . . . (_Boston_______________)
STATE . . . . . . . . . (_MA_)
ZIP . . . . . . . . . . (_06000__)
DATE  . . . . . . . . . (_19970314_)
NOTES . . . . . . . . . (_+__________________________________________________>
```

Figure 174. Type data for your table in the fields on the panel.

3. Press the Add function key after entering all the data for the row.

If you specified that you want to save each row when you press the Add function key (SAVE=IMMEDIATE), QMF adds the new row to the table.

If you specified that you want to hold all the rows and save them when you end the Table Editor session (SAVE=END), the new row is held temporarily until you end the Table Editor session.

The Table Editor ADD panel is reset as is shown in Figure 173 on page 219.

Specifying column default and null indicator

QMF allows you to specify a default indicator for columns that can support a default or a null indicator for columns that support null. For example, when you type the column default indicator in a table editor field that supports it, QMF uses the default value for the field. If you specify the column default indicator for a column that has the system date defined as its default, QMF uses the system date in that column. Table 12 describes the column default and null indicators.

Table 12. Column default and null indicators

Indicator	Character initially provided with QMF	Global variables that define the indicator character
Column Default	+	DSQCP_TEDFLT, DSQCP_TEDFLT_DBCS
Null	–	DSQCP_TENULL, DSQCP_TENULL_DBCS

If you specify the column default for a column that does not have a default, QMF returns an error message, and prompts you to correct your entry. Possible reasons QMF might not be able to detect a default value are:

- No default exists for the column

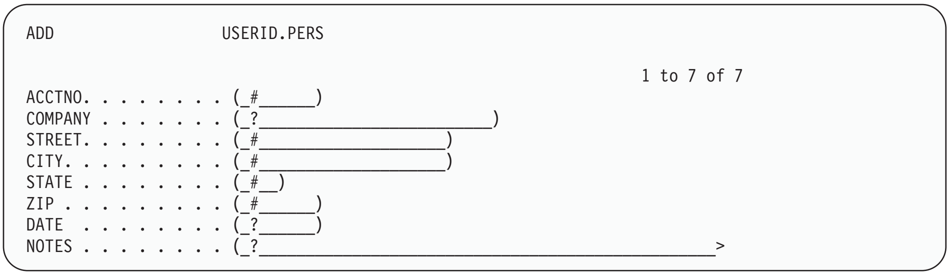
- The table you are editing has a 3–part name that refers to a remote location
- The table you are editing is a view located at a DB2 Common Server (Version 2.1.1 or higher) or DB2 for OS/390 (V4 or higher.)

Redefining the column default and null indicators

You can redefine the column default and null indicators from the command line by using the SET GLOBAL command. For example, to change the column default indicator to "?" and the null indicator to "#", enter the following at the command line:

```
SET GLOBAL (DSQCP_TEDFLT='?', DSQCP_TENULL='#')
```

Once you have issued this command, the table editor session illustrated in Figure 173 on page 219 would look like this:



```
ADD                  USERID.PERS

                                                                 1 to 7 of 7
ACCTNO. . . . . . . . . (_#______)
COMPANY . . . . . . . . (_?_________________________)
STREET. . . . . . . . . (_#____________________)
CITY. . . . . . . . . . (_#____________________)
STATE . . . . . . . . . (_#__)
ZIP . . . . . . . . . . (_#______)
DATE  . . . . . . . . . (_?______)
NOTES . . . . . . . . . (_?_________________________________________________>
```

Figure 175. The Table Editor ADD panel with new values for the column default and null indicators

Adding data to long fields

If a field is followed by a greater-than symbol **>** rather than a right parenthesis, the entire field is longer than 50 characters. If the information you need to type for this field is more than 50 characters, QMF provides a way to display the entire field.

To add data:

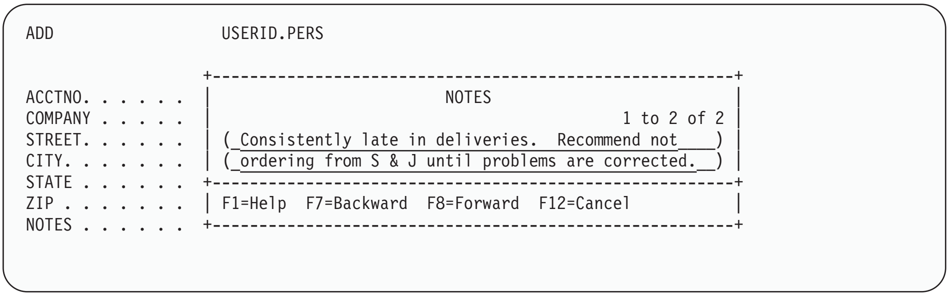
1. Move the cursor to the field you want to display.

   In the PERS sample table, the NOTES field is longer than 50 characters.
2. Press the Show Field function key.

   The Show Field panel for the field displays.

   The valid values for the field appear at the bottom of the panel on the message line.
3. Type the data for the field.

   When you get to the end of one line, just continue typing. Your data will automatically wrap to the next line.



```
ADD                 USERID.PERS

                    +-------------------------------------------------------+
ACCTNO. . . . . .   |                        NOTES                          |
COMPANY . . . . .   |                                             1 to 2 of 2 |
STREET. . . . . .   | (_Consistently late in deliveries.  Recommend not____) |
CITY. . . . . . .   | (_ordering from S & J until problems are corrected.__) |
STATE . . . . . .   +-------------------------------------------------------+
ZIP . . . . . . .   | F1=Help  F7=Backward  F8=Forward  F12=Cancel          |
NOTES . . . . . .   +-------------------------------------------------------+
```

Figure 176. You can enter more data in long fields with the Show Field key.

4. Press Enter to save the data in the field.

The Table Editor ADD panel displays with the first 50 characters of the field displayed.

Using the previous row as a model

If the row you want to add contains much of the same information as the previous row, you can save keystrokes and time by using the previous row as a model.

To copy the previous row:

1. Press the Previous function key.

   The last row you entered displays on the Table Editor ADD panel.
2. Type the information for the new row over the information that is displayed on the panel.

   Make sure that you erase any remaining old information from each field you change.

Changing rows in a table using the Table Editor

Before you begin making changes to a table, make sure the text case (UPPER, LOWER, MIXED) you specify for this session is the same as the text in the table.

For information on changing the text case, see “Setting up and changing your QMF user profile” on page 7.

To make changes to the data in a table:

1. Do one of the following, depending on where you are starting:
   - From the QMF Home panel, type:

     ```
     tablename (MODE=CHANGE
     ```

     Then, press the Edit Table function key.
   - From any other QMF panel with a command line, enter:

```
    EDIT TABLE tablename (MODE=CHANGE
```

For example, to change rows in the PERS table from a QMF command line, enter:

```
EDIT TABLE PERS (MODE=CHANGE
```

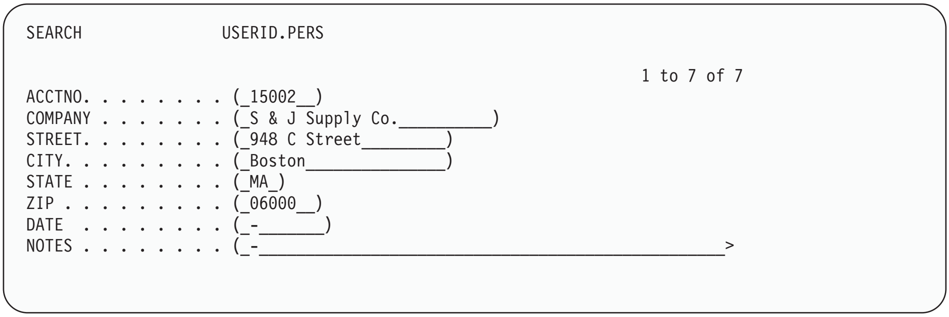
To enter the same command and save each addition as you make it, enter:

```
EDIT TABLE PERS (MODE=CHANGE SAVE=IMMEDIATE
```

The Table Editor Search panel displays.

Selecting the rows to display

The Table Editor SEARCH panel shows the name of each column in the table, followed by an entry field where you can enter search criteria to select the rows you want to change.



```
SEARCH              USERID.PERS

                                                             1 to 7 of 7
ACCTNO. . . . . . . . (_15002__)
COMPANY . . . . . . . (_S & J Supply Co.__________)
STREET. . . . . . . . (_948 C Street_________)
CITY. . . . . . . . . (_Boston_______________)
STATE . . . . . . . . (_MA_)
ZIP . . . . . . . . . (_06000__)
DATE  . . . . . . . . (_-_______)
NOTES . . . . . . . . (_-_________________________________________________>
```

Figure 177. The Table Editor SEARCH panel

To select the rows:

1. Type the criteria you want to use to select the rows to change. Leave a null in any field for which you are not specifying selection criteria. Press the Clear function key to clear all the fields and set them to the null or column default indicator. Press the Show Field function key to see the data type for a column.

   If you want to select all the rows in the table, press Enter.

   If you want to select a specific set of rows to change, you can use the underscore (_) and the percent sign (%) as selection symbols to specify selection criteria for any column that contains character or graphic data.

   - Use an underscore to fill in for one character.
   - Use the percent sign to fill in for zero or more characters.
2. Press the Search function key.

To search on ROWID, you must specify a valid hexadecimal value for an existing ROWID. You cannot update a ROWID value in a table; ROWID values are managed dynamically by DB2.

The Table Editor Change panel displays with the first row you selected.

Making changes to the rows in a table

1. On the Table Editor Change panel, type the changes to that row.

   You can change information in any field that is enclosed in parentheses. In the example that is shown here, you can change the information in any field but the ACCTNO field.

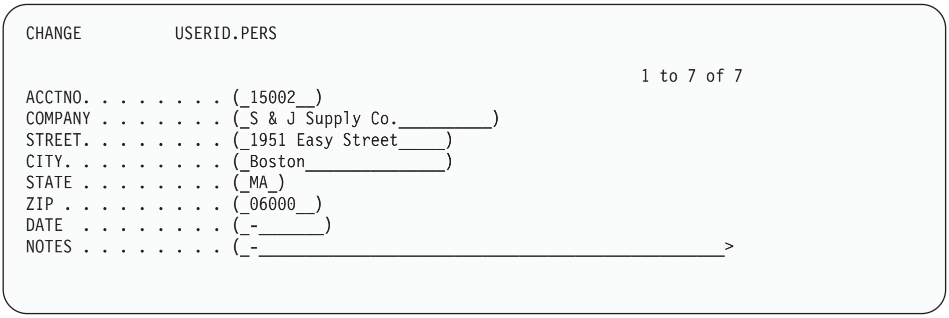
   You can display the default settings for the fields by using the Show Field PF key.

   To change a value to the default value for a field, if a default value is available, type the default indicator in the field.

   To remove changes you type and return the original data to the fields, press the Refresh function key.

   To see the next row without making changes to this row, press the Next function key.

   To select another set of rows, press the Show Search function key.
2. Press the Change function key.



```
CHANGE          USERID.PERS


                                                                  1 to 7 of 7
ACCTNO. . . . . . . . (_15002__)
COMPANY . . . . . . . (_S & J Supply Co.__________)
STREET. . . . . . . . (_1951 Easy Street_____)
CITY. . . . . . . . . (_Boston_______________)
STATE . . . . . . . . (_MA_)
ZIP . . . . . . . . . (_06000__)
DATE  . . . . . . . . (_-_______)
NOTES . . . . . . . . (_-__________________________________________________>
```

Figure 178. Change data on the Table Editor CHANGE panel.

If you specified that you want to save each row when you press the Change function key (SAVE=IMMEDIATE), QMF adds the changes to the table.

If you specified that you want to hold all the rows and save them when you end the Table Editor session (SAVE=END), the new row is held temporarily until you end the Table Editor session.

If there are more rows to display from the set you selected, the Table Editor Change panel displays.

If there are no more rows that are left to display, the Table Editor Search panel displays.

Deleting rows from a table using the Table Editor

1. Make sure that the row you want to delete appears on the Table Editor Change panel.
2. Press the Delete function key.

If you specified that you want to save each row when you press the Delete function key (SAVE=IMMEDIATE), QMF deletes the rows from the table.

If you specified that you want to hold all the rows and save them when you end the Table Editor session (SAVE=END), QMF holds the deleted row temporarily until you end the Table Editor session.

If there are more rows to display from the set you selected, the Table Editor Change panel displays.

If there are no more rows that are left to display, the Table Editor Search panel displays.

Ending a Table Editor session

To end a Table Editor session, do one of the following tasks:

- Press the End function key. If you specified SAVE=END when you began the Table Editor session, QMF saves the held rows in the database.
- Press the Cancel function key. You can only cancel a Table Editor session if you specified SAVE=END. QMF does not save any held rows in the database.

The QMF panel from which you began the Table Editor session displays.

If you want to see the changed table, enter:

```
DISPLAY TABLE tablename
```

For more information on all the Table Editor commands, see *DB2 QMF Reference*.

Adding rows to a table using SQL statements

There are two ways to add rows to a table by using SQL statements:

- Use the QMF DRAW command to create a query that adds the data to the table.
- Use SQL statements to create your own query to add the data to the table.

Queries that add data to a table are called insert queries.

Using the QMF DRAW command to add rows

1. On the QMF command line, enter:

   ```
   RESET QUERY (LANGUAGE=SQL
   ```

   The SQL query panel displays.
2. On the QMF command line, enter:

   ```
   DRAW tablename (TYPE=INSERT
   ```

   The INSERT query template for the table displays.
3. Under **ENTER VALUES BELOW**, type the data for each column.
4. Press the Run function key.

QMF adds the new row to the table.

Repeat these steps to add additional rows to the table.

Writing your own query to add rows

You can write your own insert query by using SQL statements.

To create an insert query, use the SQL INSERT statement. The syntax of the INSERT statement is:

```
INSERT INTO tablename
  VALUES (value1, value2, value, ...)
```

Where:

- *Tablename* is the name of the table to which you are adding data
- *value1, value2, value3* is the data you are adding to each column.

To write an insert query:

1. Enter:

   ```
   RESET QUERY (LANGUAGE=SQL
   ```

   The SQL query panel displays.
2. Use the SQL INSERT statement to add data to each column.

   If you do not specify data for a column, QMF adds a null value.
3. Press the Run function key to run the query.

QMF adds the new row to the table.

Repeat these steps to add additional rows to the table.

Changing rows in a table using SQL statements

There are two ways to change rows in a table by using SQL statements:

- Use the QMF DRAW command to create a query that updates the data in the table.
- Use SQL statements to create your own query to update the data in the table.

Queries that update data in a table are called update queries.

Using the QMF DRAW command to change rows

1. Enter:
   ```
   RESET QUERY (LANGUAGE=SQL
   ```
   The SQL query panel displays.
2. Enter:
   ```
   DRAW tablename (TYPE=UPDATE
   ```
   The UPDATE query template for the table displays.
3. Under **ENTER VALUES BELOW**, type the data for each column.
4. Press the Delete function key to delete any rows you are not changing. Make sure that there is no comma in front of the first column name.
5. Press the Run function key to run the query.

QMF updates the table.

Repeat these steps to update additional rows in the table.

Writing your own query to change rows

You can also write your own update query by using SQL statements.

To change rows using SQL:

1. On the QMF command line, enter:
   ```
   RESET QUERY (LANGUAGE=SQL
   ```
   The SQL query panel displays.
2. Use the SQL UPDATE statement to change rows.
3. Press the Run function key to run the query.

QMF makes the updates to the table.

Repeat these steps to make additional updates to the table.

Deleting rows from a table using SQL statements

Use the SQL DELETE statement to create a query to delete one or more rows from a table.

For example, the following query deletes the row for employee number 410 from the MYSTAFF table:

```
DELETE FROM MYSTAFF
  WHERE ID = 410
```

This query deletes all rows that are associated with department 38:

```
DELETE FROM MYSTAFF
  WHERE DEPT = 38
```

Copying rows from one table into another using SQL statements

You can use an insert query to copy certain rows and columns from an existing table into another table.

You can add the rows to an existing table, or you can specify a new table name and create a new table to contain the rows you specify.

For example, the following insert query adds the ID number, name, department, and job columns for all employees in department 38 in the Q.STAFF table to the MYSTAFF table:

```
INSERT INTO MYSTAFF (ID, NAME, DEPT, JOB)
SELECT ID, NAME, DEPT, JOB
  FROM Q.STAFF
  WHERE DEPT = 38
```

After running this query, QMF adds five new rows to the MYSTAFF table. For these employees, the YEARS, SALARY, and COMM columns contain null values, because QMF does not select these columns in the query. If you want to include all the data for a row, you must select all the columns in the table.

Adding a new column to a table using SQL statements

Use the ALTER TABLE SQL statement to add a new column to a table.

For example, to add a NOTES column to the CALENDAR table, run the following SQL statement:

```
ALTER TABLE CALENDAR
ADD NOTES VARCHAR(40)
```

NOTES is the name of the new column, VARCHAR is the data type, and 40 is the number of characters in the column.

If you are storing the table in a DB2 database, you can specify a default value other than null for the column. If you are storing the table in a DB2 Server for VSE or VM database, the default value must be null for the column.

For more information on the ALTER TABLE SQL statement, see *DB2 Universal Database for z/OS SQL Reference* for your database management system.

Working with BLOB, CLOB, and DBCLOB data

DB2 QMF Version 8.1 can display BLOB, CLOB, and DBCLOB data and the defined length field with the new edit code ’M’. LOB data types BLOB, CLOB, and DBCLOB can now be displayed in a QMF report or table without having to be casted as VARCHAR or VARGRAPHIC data types. The size of a LOB data row ranges in size from 0 bytes to 2 GB minus one byte.

To view actual LOB data, users can modify FORM.MAIN or FORM.COLUMNS and change the column edit code to ’C’ or ’CW’ to display character data. The maximum horizontal width of a report is 32 KB. To display LOB data greater than 32 KB, users must specify a column wrapping edit code. If a wrapping edit code is not specified, the LOB data that does not fit within the report will not be displayed; no message will be issued.

For more information on new edit code ’M’, see “Specifying punctuation for the values in a column” on page 123

Greater than 32 KB support for LOB data type columns

The maximum horizontal width of a report is 32 KB. If a user wants to display LOB data larger than 32 KB, he will have to specify a column wrapping edit code. If a wrapping edit code is not specified, the LOB data that does not fit within the report will not be displayed; no message will be issued.

Authorizing access to your tables using SQL statements

After you create a table and add data to it, you can use SQL statements to specify how you want other users to access the information. For example, you can authorize users to make changes to your tables, or you can limit their access so they can only view the data.

Giving users access to your tables

You can give users authority to do any of the following with your tables:

- View the data in a table
- Add new rows to the table
- Change the rows in the table
- Delete rows from a table

For example, to give a user with user ID LINDSAY the authority to view, add, change, and delete the data in the CALENDAR table, run the following query:

```
GRANT ALL ON TABLE CALENDAR
TO LINDSAY
```

To allow LINDSAY the authority to view the data in the PERS table, run this query:

```
GRANT SELECT ON TABLE PERS
TO LINDSAY
```

To allow authority to the MYSTAFF table to remote users, run this query:

```
GRANT ALL ON TABLE MYSTAFF
TO PUBLIC AT ALL LOCATIONS
```

Note to CICS Users

You can only allow users authority to view data in tables at remote locations.

Allowing users to update specific columns in your tables

You can give another user authority to update specific columns in your tables.

The following example shows you how to give LINDSAY the authority to update the LOCATION column in the CALENDAR table.

To give user authority for specific columns:

1. Run this query to allow another user to view the data in the query and to select rows to change:

   ```
   GRANT SELECT ON TABLE CALENDAR TO LINDSAY
   ```

2. Run this query to allow another user to update a specific column in the table:

   ```
   GRANT UPDATE(LOCATION) ON TABLE CALENDAR TO LINDSAY
   ```

   For more information on the GRANT SQL keyword, see the SQL reference manual for your database management system.

Revoking access to a table

You can revoke access to a table. For example, to prevent LINDSAY from deleting rows from the CALENDAR table, run this query:

```
REVOKE DELETE ON CALENDAR
FROM LINDSAY
```

For more information on the REVOKE SQL keyword, see the SQL reference manual for your database management system.

Entering date and time values by using QMF

There are some additional considerations when using SQL statements to insert or update date and time values by using QMF. QMF application programs are precompiled with date and time options of the International Standards Organization (ISO), which represents date as *yyyy-mm-dd* and time as *hh.mm.ss*. For more information on using SQL statements to insert or update date and time values, see *DB2 QMF Reference*.

If you insert a date or time value into a character column using a special register such as CURRENT DATE or CURRENT TIME, the character string representation of the value is in the ISO format.

To insert the value in a format other than ISO, you can use a statement like this:

```
INSERT INTO date_table
  SELECT CHAR(CURRENT DATE, EUR)
  FROM any_table
  WHERE any_table.unique_column = 'unique_value'
```

Where *date_table* is the name of the table into which you want to insert the current date value, *any_table* is any table (preferably not one that is subject to changes) with a column that contains unique values, and *'unique_value'* is a value of the unique column. In these examples, *date_table* has one character column, which contains the character representation of a date value.

To insert the default ISO format, you can enter an SQL statement like the following example:

```
INSERT INTO date_table
  VALUES( CURRENT DATE )
```

To update a character column with the CURRENT DATE or CURRENT TIME value in a format other than ISO, use a statement such as the following example:

```
UPDATE date_table
  SET date_column = CHAR(CURRENT DATE, EUR)
  WHERE (clause identifying row to be updated)
```

(Where *date_column* is a column of the date type).

Chapter 12. Exporting and importing objects

Use the QMF EXPORT and IMPORT commands to share your objects with other users on your system, or to modify the object using a QMF application.

You can also import and export objects from Microsoft® Windows® environments using the QMF feature, DB2 QMF High Performance Option. See Appendix C, "The DB2 QMF High Performance Option," on page 367 for more information.

Exporting QMF objects

Use the QMF EXPORT command to export a QMF object into a sequential CMS file, a TSO data set, or a CICS data queue.

You can export QMF database objects from either temporary storage or from the database depending on the object type, as is shown in Table 13.

Table 13. You can export all kinds of QMF objects.

Data object	Export from database	Export from temporary storage
TABLE	X	
QUERY	X	X
FORM	X	X
PROCEDURE	X	X
DATA		X
REPORT		X
CHART		X

You can export reports and charts from temporary storage, but you cannot import them into temporary storage.

When you export a QMF object from an object panel, you do not need to specify the object type for the export. For example, if you enter an export command from a form panel, the form currently displayed in the panel is exported with the default object type FORM. If you enter an export command from a chart panel, the chart currently displayed in the panel is exported with the default object type CHART. If you attempt to export from a panel that does not have a valid object type for export, and do not specify a valid object type, QMF prompts you for a valid object type.

Exporting QMF objects into TSO

To export a QMF object from temporary storage to a TSO data set, enter:

```
EXPORT objecttype TO dataset
```

For example, to export a query from temporary storage to a data set that is named REPORTX, enter:

```
EXPORT QUERY TO REPORTX
```

To export a QMF object in the database to a data set, enter:

```
EXPORT objecttype objectname TO dataset
```

For example, to export a query that is named MYREP4Q in the database to a data set that is named RPT4Q, enter:

```
EXPORT QUERY MYREP4Q TO RPT4Q
```

You can use either a fully qualified or partially qualified name in TSO.

For more information on using TSO data set names with the EXPORT command, see the *QMF Reference*.

Exporting QMF objects into CMS

To export a QMF object in temporary storage to a CMS file, enter:

```
EXPORT objecttype TO filename
```

For example, to export a query in temporary storage to a file, named REPORTX, enter:

```
EXPORT QUERY TO REPORTX
```

If you do not specify a file type or file mode, QMF uses the object type, in this case QUERY, as file type, and A as the file mode.

To export a QMF object in the database to a file, enter:

```
EXPORT objecttype objectname TO filename
```

For example, to export a query that is named MYREP4Q in the database to a file that is named RPT4Q, enter:

```
EXPORT QUERY MYREP4Q TO RPT4Q
```

Exporting QMF objects into CICS

To export a QMF object in temporary storage to a CICS data queue, enter:

```
EXPORT objecttype TO queuename (queuetype=TS/TD
```

For example, to export a query in temporary storage to a data queue that is named REPORTX, and a queue type of TS, enter:

```
EXPORT QUERY TO REPORTX
```

To export a QMF object in the database to a data queue, enter:

```
EXPORT objecttype objectname TO dataqueue (queuetype=TS/TD
```

For example, to export a query that is named MYREP4Q in the database to a data queue that is named RPT4Q, and a queue type of TS, enter:

```
EXPORT QUERY MYREP4Q TO RPT4Q
```

Exporting QMF reports for use on the Internet

You can export reports for use on the Internet by specifying the HTML parameter with your EXPORT REPORT command.

To export an HTML report to a TSO data set, enter:

```
EXPORT REPORT TO dataset (DATAFORMAT=HTML
```

To export an HTML report to a CMS file, enter:

```
EXPORT REPORT TO filename filetype filemode (DATAFORMAT=HTML
```

To export an HTML report to a CICS data queue, enter:

```
EXPORT REPORT TO queuename (QUEUETYPE=TS|TD DATAFORMAT=HTML
```

See *DB2 QMF Reference* for more details on the EXPORT REPORT command. The resulting report contains HTML code that allows your report to be viewed with a web browser.

Importing QMF objects

Use the QMF IMPORT command to bring a file, data set, or data queue back into a QMF temporary storage area or the database.

You can import QMF database objects into either temporary storage or the database depending on the object type, as is shown in Table 14.

Table 14. You can import all kinds of QMF objects.

Data object	Import to database	Import to temporary storage
TABLE	X	
QUERY	X	X
FORM	X	X
PROC	X	X
DATA		X

Importing QMF objects from TSO

To import a TSO data set into QMF temporary storage, enter:

```
IMPORT objecttype FROM dataset
```

For example, to import a query in a data set that is named REPORTX to temporary storage, enter:

```
IMPORT QUERY FROM REPORTX
```

To import a QMF object in a data set to the database, enter:

```
IMPORT objecttype objectname FROM dataset
```

For example, to import a query that is named MYREP4Q from a data set that is named RPT4Q to the database, enter:

```
IMPORT QUERY MYREP4Q FROM RPT4Q
```

You can use either a fully qualified or partially qualified name in TSO.

For more information on using TSO data set names with the IMPORT command, see the *QMF Reference*.

Importing QMF objects from CMS

To import a QMF object in a CMS file to temporary storage, enter:

```
IMPORT objecttype FROM filename
```

For example, to import a query in a file that is named REPORTX to temporary storage, enter:

```
IMPORT QUERY FROM REPORTX
```

If you do not specify a file type or file mode, QMF uses the object type, in this case QUERY, as file type, and A as the file mode.

To import a QMF object in a file to the database, enter:

```
IMPORT objecttype objectname FROM filename
```

For example, to import a query that is named MYREP4Q from a file that is named RPT4Q to the database, enter:

```
IMPORT QUERY MYREP4Q FROM RPT4Q
```

Importing QMF objects from CICS

To import a QMF object in a CICS data queue to temporary storage, enter:

```
IMPORT objecttype FROM queuename (queuetype=TS/TD
```

For example, to import a query in a data queue that is named REPORTX, with a queue type of TS, to temporary storage, enter:

```
IMPORT QUERY FROM REPORTX
```

To import a QMF object from a data queue to the database, enter:

```
IMPORT objecttype objectname FROM dataqueue (queuetype=TS/TD
```

For example, to import a query that is named MYREP4Q from a data queue that is named RPT4Q, and a queue type of TS, to the database, enter:

```
IMPORT QUERY MYREP4Q FROM RPT4Q
```

For more information on the EXPORT command and the IMPORT command, see *DB2 QMF Reference*.

Chapter 13. Accessing data at a remote database

You can access data at a remote DB2 database using QMF. Then, you can create reports and charts to view the data on your local system. You can connect to the remote database when you start QMF or during a QMF session. You can connect two like (for example, DB2 UDB for z/OS to DB2 UDB for z/OS) or two unlike (for example, DB2 UDB for z/OS to DB2 Server for VM) databases.

When you are connected to a remote database, you access data and objects in the same way that you access them at a local database. QMF continues to use programs that reside at the system on which you are running QMF.

QMF provides two ways to access data at remote locations:

- Remote unit of work access for DB2 UDB for z/OS or DB2 Server for VM or VSE databases
- Distributed unit of work access between DB2 UDB for z/OS databases

You can connect to multiple remote databases simultaneously from Windows environments by using the DB2 QMF for Windows feature. See Appendix C, "The DB2 QMF High Performance Option," on page 367 for more information.

ROWID and LOB data types are supported in DB2 UDB for OS/390 beginning with Version 6. After a connection from an application requester that does not support ROWID and LOB data, the result is unpredictable when you attempt to access data containing these data types.

Accessing data at a remote database using Remote Unit of Work

Using remote unit of work, you can access data at either a remote DB2 UDB for z/OS database or a remote DB2 Server for VM or VSE database. The remote database is called the server. To use remote unit of work to access the data, you must first connect to the remote database. You can connect to a remote database in either of the following ways:

- Using the QMF CONNECT command during a QMF session
- Using the DSQSDBNM program parameter when you start a QMF session

You can change authorization IDs while attached to remote locations.

Connecting to a remote database using the QMF CONNECT command

Use the QMF CONNECT command to connect to a remote database during a QMF session.

You can issue the CONNECT command from:

- The command line
- Within a procedure (linear or with logic)
- The callable or command interface

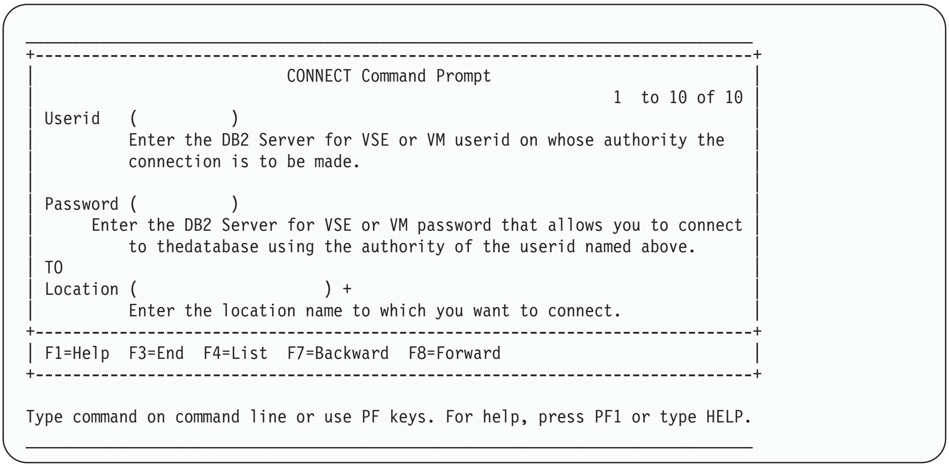
For more information on procedures, see Chapter 8, “Creating a procedure to run QMF commands,” on page 181. For more information on the callable or command interface, see *Developing DB2 QMF Applications*. Before connecting to the remote database, QMF completes any work (for example, a large report) at the current location.

To use the CONNECT command:

1. If you need help with the syntax of the CONNECT command, enter:

   ```
   CONNECT ?
   ```

   The CONNECT Command Prompt panel displays:



```
_____________________________________________________________________________
+---------------------------------------------------------------------------+
|                           CONNECT Command Prompt                          |
|                                                          1  to 10 of 10  |
| Userid   (         )                                                      |
|          Enter the DB2 Server for VSE or VM userid on whose authority the |
|          connection is to be made.                                        |
|                                                                           |
| Password (         )                                                      |
|      Enter the DB2 Server for VSE or VM password that allows you to connect|
|          to thedatabase using the authority of the userid named above.   |
| TO                                                                        |
| Location (                  ) +                                           |
|          Enter the location name to which you want to connect.           |
+---------------------------------------------------------------------------+
| F1=Help  F3=End  F4=List  F7=Backward  F8=Forward                         |
+---------------------------------------------------------------------------+

Type command on command line or use PF keys. For help, press PF1 or type HELP.
_____________________________________________________________________________
```

2. Type the information you need to connect to the remote database.

   If a plus sign appears after the Location field, you can press the List function key to display a list of database names.

   If you use QMF in the VM environment, the list contains only databases that are specified in the communications directories. It does not necessarily contain all the databases to which you can connect. In VSE it is the DBName directory. If a database is not specified here, a connection cannot occur.

If you select a database from the list but can not connect to it, check that:

- You have the authority to connect to the database
- The database location supports remote unit of work
- The database is actually up and running

For more information on the CONNECT command, see *DB2 QMF Reference*.

Connecting to a remote database using the DSQSDBNM program parameter

To use the DSQSDBNM program parameter to specify the database to which you want to connect when you start QMF, enter:

```
QMFn D=DBNAME
```

Where `n` is the language identifier for the session you are starting, and `DBNAME`, which must be entered in uppercase, is the name of the database to which you are connecting.

For example, to start an English-language session, and to connect to a database that is named Detroit, enter:

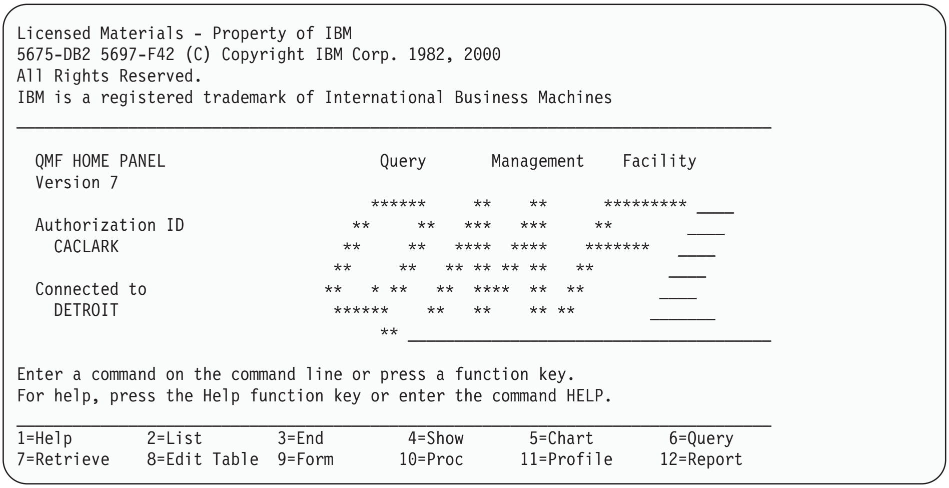
```
START QMFE D=DETROIT
```

Viewing the current database location

QMF provides several ways for you to view the name of the database to which you are currently connected. Viewing the current database name can help you orient yourself if you are accessing data in more than one location.

Viewing the current database location on the QMF home panel

When you connect to a new database, that location name is displayed below the **Connected to** heading on the QMF Home panel:



```
Licensed Materials - Property of IBM
5675-DB2 5697-F42 (C) Copyright IBM Corp. 1982, 2000
All Rights Reserved.
IBM is a registered trademark of International Business Machines
_________________________________________________________________________________

  QMF HOME PANEL                       Query       Management    Facility
  Version 7
                                      ******     **    **      ********* ____
  Authorization ID                  **     **   ***   ***     **        ____
    CACLARK                        **     **   ****  ****    *******   ____
                                  **     **   ** ** ** **   **        ____
  Connected to                   **   * **   **  ****  **  **        ____
    DETROIT                       ******    **   **    ** **         _______
                                       ** _______________________________________

Enter a command on the command line or press a function key.
For help, press the Help function key or enter the command HELP.
_________________________________________________________________________________
1=Help        2=List        3=End         4=Show       5=Chart        6=Query
7=Retrieve    8=Edit Table  9=Form       10=Proc      11=Profile     12=Report
```

Viewing the current database location using a global variable

If the location name does not appear on the QMF Home panel (for example, if you lose your connection to the database, or if you are connected to a DB2 subsystem without a location name), you can view the database location name by displaying the global variable DSQAO_CONNECT_LOC. Enter:

```
SHOW GLOBAL
```

The GLOBALS panel displays with a list of all your QMF global variables. The value of the DSQAO_CONNECT_LOC variable is the location name.

For a list of all QMF global variables, see *DB2 QMF Reference*.

Viewing the current database location using QMF's governor exit

You can view the location name by displaying the XCBCLOC field in the QMF control block DXEXCBA.

Reconnecting to a location

You can reconnect to a remote database if the connection is lost. The steps you need to take to reconnect may vary depending on how you are issuing your QMF commands. You can issue QMF commands in any of the following ways:

- Enter QMF commands interactively (by entering a command on the QMF command line or by using a function key)
- Run either a linear procedure or a procedure with logic that contains QMF commands
- Run a batch application that contains QMF commands

Reconnecting if you are entering QMF commands interactively

If you are entering QMF commands interactively and the connection to the remote location is lost, the Lost Connection Prompt panel displays.

To reconnect to the remote database:

1. Type 1 to reconnect to the remote database, or type 2 to exit QMF.

   If you type 1 to reconnect to the remote database, the CONNECT Command Prompt panel displays.

   The name of the location to which you were previously connected appears in the **TO Location** field. If you are using DB2 Server for VSE or VM, your user ID appears in the **Userid** field.
2. Enter the information you need to reconnect to the remote database.

   If QMF cannot connect to the location you specify, the CONNECT Command Prompt panel displays so you can try to connect again.

Reconnecting if you are running QMF commands in a procedure

If you are running QMF commands in a linear procedure and the connection to the remote location is lost, the procedure ends. If you are running the procedure interactively, the Lost Connection Prompt panel displays so you can reconnect.

If you are running QMF commands in a procedure with logic, the logic of the procedure determines how the procedure ends. When the procedure finishes, and if you are running the procedure interactively, the Lost Connection Prompt panel displays so you can reconnect.

Reconnecting if you are running QMF commands in a batch application

If you are entering QMF commands in a batch application and the connection to the remote location is lost, QMF ends.

To reconnect to the remote location:

1. Start your QMF session.
2. Unless you automatically connect to the remote database when you start QMF, use the CONNECT command to connect to the remote database.

For more information on the CONNECT command, see *DB2 QMF Reference*.

What you can expect when you reconnect

If you are using the Table Editor to update a remote database, and you lose the connection to the database, any updates you have not saved are lost.

If you are running QMF commands from a list of database objects and the connection to that database is lost, the database list is obsolete. You can still display the list, but if you type a command on the list, you will receive an error.

Accessing data at a remote database using distributed unit of work

If you are using a DB2 Version 2 Release 2 (or later) database, you can access data in another DB2 database by using distributed unit of work. With distributed unit of work, you do not need to connect to the remote database. Instead, specify the location name as part of the table name when you select the table.

The following example selects all the rows from a table that is named STAFF that is owned by Q and located in NEW_YORK.

```
SELECT * FROM NEW_YORK.Q.STAFF
```

You can retrieve data from more than one table only when each table is at the same location. For example, you cannot retrieve data from NEW_YORK.Q.STAFF and ATLANTA.Q.ORG in the same query.

You can update tables that are located at remote locations, but you can only create tables at your own location.

Your installation can also assign an alias for the three-part name when querying a remote table. For example, your installation might assign the alias NYSTAFF for NEW_YORK.Q.STAFF. For more information on using remote tables and aliases, see the DB2 publications that are listed in the bibliography at "Bibliography" on page 387. You can also contact your information center.

Using QMF when connected to a database through remote unit of work

This section describes how a remote unit of work environment affects data and QMF objects.

The current location and the system where QMF is running are involved when using remote unit of work. The current location is the database location to which you are connected. *Where QMF is running* is the operating system from which you started QMF.

Data

Commands and queries that access data, such as `DISPLAY TABLE` *`tablename`*, go to the current location. The current location is the location of the application server, unless the current location is DB2 and *tablename* is a three-part name (or an alias for that name) that refers to a DB2 subsystem other than the current one.

QMF objects

QMF objects (queries, procedures, and forms) that are retrieved from the database must reside at the current location. If you start your QMF session connected to location CHICAGO and then connect to location NEWYORK, you can only run your query if it is in NEWYORK.

Tips and techniques

This section offers advice on how to use QMF effectively in a remote unit of work environment.

You can issue a GRANT statement at a remote location if you first connect to the remote location. You can grant privileges on a table that resides at the current server to users at other locations by using the GRANT clause PUBLIC AT ALL LOCATIONS. With remote unit of work, you cannot use a three-part name in GRANT statements if the three-part name refers to an object at the local DB2 database.

Note to CICS users

If you are using QMF at a CICS location and connect to a location in the VM or TSO environment, procedures with logic and report calculations do not run from your CICS session.

CURRENT SQLID

In DB2, your CURRENT SQLID is not active after you connect to a different location. If you need to use the same CURRENT SQLID with multiple DB2 application servers from a single QMF session, you might need to reset the CURRENT SQLID after you connect to each server. For more information, see the discussion of the QMF CONNECT command in *DB2 QMF Reference*.

Function keys and synonyms

After a successful connection, the profile (except for TRACE) resource control table, synonyms, and function keys are re initialized to the values at the current location.

Procedures, forms, and queries

Procedures, forms, and queries must be retrieved from or stored into the database at the current location. However, objects can reside in temporary storage on the system where QMF is running. You cannot refer to objects by using three-part names.

Commands

With remote unit of work support, all programs started by QMF run under the operating system in which QMF is running (the local operating system). Such programs might include CMS, TSO, and CICS commands. If QMF is running in TSO, and you attempt to run a procedure that contains CMS commands from a current DB2 Server for VSE or VM location, the TSO commands work, but the CMS commands do not.

Using the QMF CONNECT command to connect to databases

This section describes the following:

- The impact of the QMF CONNECT command on your user ID in the remote unit of work environment
- Scenarios you might encounter when you connect to a remote location
- Using remote unit of work and distributed unit of work (applies to DB2 only) in tandem

The examples in this section show how QMF returns to the panel from which you issued the CONNECT command after connecting successfully. On that panel, the following message appears immediately above the command line:

```
"OK, CONNECT performed. Please proceed."
```

Example 1: How connecting to a new location affects your user ID

DB2 Server for VM application requester and a DB2 for VM application server: When you connect to a new location, your DB2 Server for VM user ID is not in effect after a connection to a different location. Instead, it is based on the VM logon ID at the previous location.

- Assume that your VM logon ID at MIAMI is DAVID and that you first connect to your local DB2 for VM DBMS (MIAMI):

  ```
  CONNECT DANIEL (Password=PWDAN
  ```

  This command sets your DB2 Server for VM user ID at MIAMI to DANIEL.
- Now, you connect to another DB2 Server for VM DBMS (DETROIT):

  ```
  CONNECT TO DETROIT
  ```
- Your DB2 Server for VM user ID at DETROIT is DAVID, not DANIEL.

DB2 Server for VM application requester and a DB2 UDB for z/OS application server: You can set your user ID to DANIEL at a new location by using the SET CURRENT SQLID SQL statement, if your DBMS at that location is DB2 UDB for z/OS, and you are connected to that location.

- Assume that your VM logon ID at MIAMI is DAVID. Also assume that you first connect to your local DB2 Server for VM DBMS (MIAMI):

  ```
  CONNECT DANIEL (Password=PWDAN
  ```

  This command sets your DB2 Server for VM user ID to DANIEL. Now, you connect to a DB2 UDB for z/OS DBMS (DALLAS):

  ```
  CONNECT TO DALLAS
  ```
- Assuming no name translation, your user ID at DALLAS is DAVID, not DANIEL. However, because you connected to a DB2 UDB for z/OS location, you can use the following SQL statement to change your user ID (your current SQL authorization ID) to DANIEL at this location:

  ```
  SET CURRENT SQLID = 'DANIEL'
  ```

Because SET CURRENT SQLID is an SQL statement, you issue it through an SQL statement. Consequently, the following restrictions apply to the user ID you specify:

- You must enclose it within single quotes.
- It must be your primary ID or one of your secondary authorization IDs.

Example 2: Connecting to like databases

Connecting DB2 UDB for z/OS to DB2 UDB for z/OS: This example assumes that you have DB2 Version 7 Release 1 (or higher) installed at each of the two locations.

If you are working at the local DB2 UDB for z/OS location DALLAS, and you need to issue GRANT statements for tables in the database at the DB2 UDB for z/OS location BOSTON, you must first connect to the BOSTON location.

You can request a connection to the BOSTON location in two ways:

- Enter `CONNECT TO BOSTON` on the command line.
- Enter `CONNECT ?` on the command line, then enter `BOSTON` on the CONNECT Command Prompt panel displayed over the QMF Home panel.

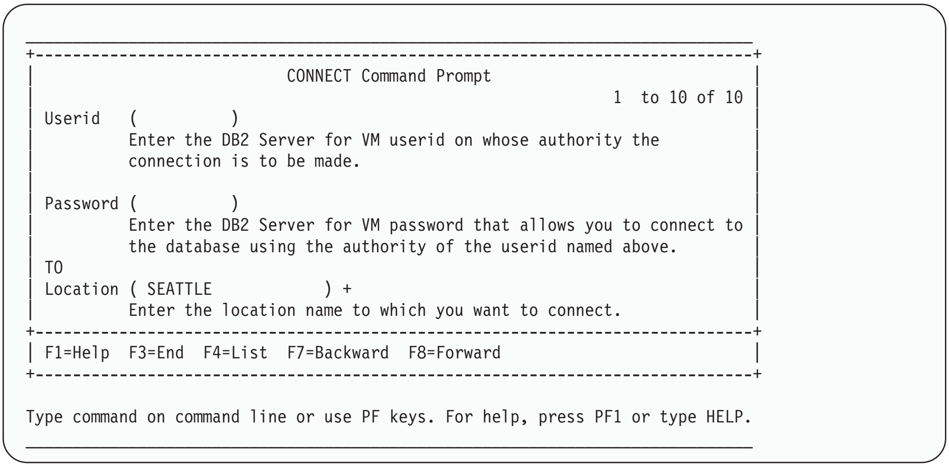
Connecting a DB2 Server for VM Application Requester to a DB2 Server for VM Application Server: If you have QMF running on a DB2 Server for VM database in MIAMI, and you want to access data stored in the DB2 Server for VM database SEATTLE, you must first connect to the SEATTLE location.

This example assumes that you have the following release levels of DB2 Server for VM installed at the two locations:

- MIAMI, DB2 Server for VM Version 7.1
- SEATTLE, DB2 Server for VM Version 7.1

You can request a connection to the SEATTLE location in two ways:

- Enter `CONNECT TO SEATTLE` on the command line
- Enter `CONNECT ?` on the command line, then enter `SEATTLE` on the CONNECT Command Prompt panel displayed over the QMF Home panel, as shown in Figure 179 on page 248



```
 ____________________________________________________________________________
+----------------------------------------------------------------------------+
|                          CONNECT Command Prompt                            |
|                                                            1  to 10 of 10 |
| Userid   (          )                                                      |
|          Enter the DB2 Server for VM userid on whose authority the        |
|          connection is to be made.                                        |
|                                                                            |
| Password (          )                                                      |
|          Enter the DB2 Server for VM password that allows you to connect to |
|          the database using the authority of the userid named above.      |
| TO                                                                          |
| Location ( SEATTLE            ) +                                          |
|          Enter the location name to which you want to connect.           |
+----------------------------------------------------------------------------+
| F1=Help  F3=End  F4=List  F7=Backward  F8=Forward                          |
+----------------------------------------------------------------------------+

 Type command on command line or use PF keys. For help, press PF1 or type HELP.
 ____________________________________________________________________________
```

Figure 179. CMS CONNECT Command Prompt panel - DB2 Server for VM Version 7.1

- You do not need to specify a user ID or password, but if you specify a user ID, you will also need a password. If you do not specify a user ID, DB2 Server for VM uses your VM logon ID.
- The panel you see depends on the release level of DB2 Server for VM that is installed at your location. The panel in Figure 179 is specific for DB2 Server for VM Version 7.1 using the PROTOCOL(AUTO) or PROTOCOL(SQLDS) option.

Example 3: Connecting to unlike databases

If you are working with QMF at the local DB2 UDB for z/OS location DALLAS, and you need to create tables at the DB2 Server for VM database MIAMI, you must first connect to the MIAMI location.

This example assumes that you have the following release levels of DB2 Server for VM and DB2 installed at the various locations:

- DALLAS, DB2 UDB for z/OS Version 7.1
- MIAMI, DB2 Server for VM Version 7.1

You can request a connection to the MIAMI location in two ways:

- Enter `CONNECT TO MIAMI` on the command line.
- Enter `CONNECT ?` on the command line, then enter MIAMI on the CONNECT Command Prompt panel displayed over the QMF Home panel.

The panel you see depends on the DB2 Server for VM release level you have installed at your location and the PROTOCOL option you use for SQLINIT.

Example 4: Connecting to a new location using remote unit of work and distributed unit of work

You want to connect to the DB2 UDB for z/OS subsystem, DALLAS, and QMF is running at the DB2 Server for VM location, CHICAGO. While connected to the DALLAS database location, you also want to access data from the sample inventory table CHARLE.INVENTORY at the DB2 UDB for z/OS subsystem NEWYORK.

1. Use remote unit of work support.

   To connect to location DALLAS:

   - Enter `CONNECT TO DALLAS` on the command line.
   - Or you can enter `CONNECT ?` on the command line, then enter DALLAS on the CONNECT Command Prompt panel displayed over the QMF Home panel.
2. Use distributed unit of work with a three-part name.

   You can access data from the DB2 UDB for z/OS subsystem, NEWYORK, yet remain connected to the DB2 UDB for z/OS subsystem at DALLAS. To do this, you must use a three-part name (or an alias for that name) within the SQL statements.

   For example, use a SELECT query like this:

   ```
   SELECT *
   FROM NEWYORK.CHARLE.INVENTORY
   ```
3. Use distributed unit of work with an alias.

   With the following SQL statement, you can also specify an alias for the three-part name within the query:

   ```
   CREATE ALIAS MONTHLY FOR NEWYORK.CHARLE.INVENTORY
   ```

   Enter the query, and then run it at the location you to which you are connected. In this example, you are connected to the DALLAS location.

   After you create the alias, you can use it in a SELECT query like this:

   ```
   SELECT *
   FROM MONTHLY
   ```

Chapter 14. National Language support in QMF

This chapter discusses bilingual commands, bilingual forms, and double-byte character set (DBCS) data. Ask your QMF administrator whether you have the proper hardware and software necessary to operate with DBCS. National Language Support is available for DB2 QMF for Windows, but not for DB2 QMF HPO features, HPO/Manager and HPO/Compiler. See Appendix C, "The DB2 QMF High Performance Option," on page 367 for more information.

Bilingual command support

A QMF National Language Feature (NLF) is a non-English version of QMF. When running a QMF NLF, you can issue QMF commands in the presiding language or in English by setting a QMF global variable. When you choose English, the QMF panels appear in the presiding language, but only English commands are accepted.

If you select English, any NLF session can run a procedure that is written in English (if all the QMF commands in the procedure are in English). This increases the portability of procedures among the various NLFs by providing a common language for QMF commands.

You can use the following English commands in any QMF NLF session without switching to English:

- INTERACT
- MESSAGE
- GET GLOBAL
- SET GLOBAL

For an example of how to use QMF's bilingual capabilities, see *Developing DB2 QMF Applications*.

Exporting and importing bilingual forms

If you are using a QMF NLF where English is not the presiding language, you can choose to export a form in either your own language or in English. If you export a form in English, you can translate it into the NLF language when you import it. This allows portability of forms among the different NLFs.

For example, if you create a form in a French NLF, you can export it in English, and then import it into a Spanish NLF.

You use the LANGUAGE parameter of both the EXPORT command and the IMPORT command to specify whether a form is exported or imported in English or in the current session (non-English) language. QMF handles all the necessary translations.

For more information on using the LANGUAGE parameter with the IMPORT command and the EXPORT command, see *DB2 QMF Reference*.

Defining DBCS data

In double-byte character sets (DBCS) the internal representation for each character requires two bytes of storage. Writing systems such as Kanji and Chinese require such double-byte representations. In some cases, the Katakana writing system is considered a single-byte character set (SBCS) because it can be represented internally in single bytes. English, German, and French languages fit the category of single-byte character sets.

References made in this chapter to “mixed” data mean that strings of DBCS data and strings of SBCS data appear in one data field. When data is mixed, the DBCS data is preceded by an SO (shift out) delimiter character and followed by an SI (shift in) delimiter character. If you enter DBCS data in a field, you do not need to enter SO and SI; they are automatically generated by the hardware when DBCS data is used. Because SO and SI are delimiters and not characters, the data contained between them is interpreted as double-byte.

How DBCS data looks when displayed

DBCS data differs from SBCS data when it appears on your terminal. It occupies twice as much space on the screen as SBCS data. When double-byte characters are displayed on your terminal screen, the SO and SI characters take up one space each. If you are using a terminal that supports DBCS data, such as an IBM 5560, you can choose to display the SO and SI delimiters in your data or make them appear as spaces.

When QMF displays DBCS data in the Table Editor, it adjusts the length of the input field for a column to allow for the SI and SO characters. This is especially evident in the Show Field window, where QMF inserts an SI or SO character or both on every line of the window. If you type over the SI and the SO characters, you can create an overflow (or error) condition.

You can display any QMF objects that contain DBCS data from the QMF Database Object List with or without a DBCS terminal. However, if the object name contains double-byte characters, and you have a non-DBCS terminal, all double-byte characters are altered. When you enter the DISPLAY command next to the DBCS object you want to display, clear the rest of the row by pressing the Erase EOF key before you press Enter.

Although QMF can display DBCS data in the Table Editor on a non-DBCS terminal, you cannot change the data. If you want to change DBCS data by using the Table Editor, you must use a terminal that supports DBCS data, such as an IBM 5560.

How DBCS data changes the length of names and fields

Generally, when you use double-byte characters in QMF, you enter fewer characters than when you use only single-byte characters. For example, object names in quotes can be 18 single-byte characters long or eight double-byte characters long.

To calculate the length of names and fields containing only double-byte characters:

1. Count the number of single-byte characters possible (for example, 18 for an object name).
2. Subtract 2 characters, one each for the SO and SI delimiters that are automatically generated.
3. Divide the remaining 16 characters by 2 to get the number of double-byte characters the name or field can contain. If the number is odd, drop the remainder after doing the division.

Thus, object names can be eight DBCS characters long.

To determine whether a name or field can contain a particular mix of double-byte and single-byte characters, use a similar process. First, for each string of double-byte characters in the name or field:

1. Count the number of double-byte characters in the string.
2. Multiply the number of double-byte characters by 2.
3. Add 2 (one each for the SO and SI delimiters).

Add the sums from all the individual strings of double-byte characters, then count the number of single-byte characters. Add the number of single-byte characters to the sums of the double-byte characters. The total cannot exceed the maximum length of the name or field that is stated for single-byte characters only.

For descriptions of the types of QMF names and fields in which you can use DBCS data, see “Data types you can use with DBCS data,” “Using DBCS data in Input fields” on page 257, and “Using DBCS in form panels” on page 258.

Data types you can use with DBCS data

You can save DBCS data in your database if you define the columns in which you save the data as character or graphic. Whether you save your DBCS data in character or graphic columns depends on your needs:

- If the column contains DBCS data strings and SBCS data strings, or if it contains a string with both DBCS and SBCS data, define the column as character.
- If the column contains only DBCS data, define the column as character if the SO and SI delimiters must be saved in the database with the double-byte characters. Otherwise, define the column as graphic.

Specifically, QMF can save DBCS data in database columns that are defined as these data types:

Character
DBCS data, when preceded and followed by single-byte single quotation marks, can appear in columns with a character data type. QMF also allows DBCS data strings that are mixed with SBCS data strings. Use this data type if all entries in the column have the same length, up to a maximum of 126 double-byte characters.

Graphic
QMF can put only DBCS data of fixed length into columns that are defined as graphic data type. Use this data type if all entries in the column have the same length, up to a maximum of 127 double-byte characters.

Variable character
Use this for variable-length entries of up to 126 double-byte characters. DBCS data, when preceded and followed by single-byte, single quotation marks, can appear in columns with a variable data type. QMF also allows DBCS data strings that are mixed with SBCS data strings.

In DB2, variable character data can exceed 126 characters. When variable character data exceeds 126 characters, it is handled like the LONG VARCHAR data type.

Variable graphic
QMF can put only DBCS data of variable length up to 127 characters into a column that is defined as VARGRAPHIC data type.

Long variable character
Use this data type with caution. LONG VARCHAR can be up to 16,382 double-byte characters long. QMF has restrictions for how you can use a column with LONG VARCHAR in a query. You cannot use it:

- In search conditions
- In sorting
- With COUNT, GROUP BY, or UNION
- In indexes

- In subqueries
- In inserting or updating queries (the value must be set to NULL)

Long variable graphic

QMF can put only DBCS data of variable length up to 16,383 characters into a column that is defined as LONG VARGRAPHIC data type. Use this data type with caution. The restrictions for how you can use a column with this data type in a query are the same as for the long variable character data type.

LOB data types

In DB2 QMF Version 8.1, LOB data types CLOB, DBCLOB, and BLOB can now be displayed in a QMF report without having to be casted as VARCHAR or VARGRAPHIC data types. The size of a LOB data row ranges from 0 bytes to 2 GB minus one byte.

- CLOB- Character large object string. CLOBs are used generally whenever a character string might exceed the limits of a VARCHAR type.
- DBCLOB- Double-byte character large object string. DBCLOBs always have an associated code page.
- BLOB- Binary large object. BLOBs do not have an associated code page or character set. They can contain image, audio, and video data.

LOB Descriptor Area (LOB DA)

Because of the potential size of LOB data, users may want to limit the viewing of actual data within a report. Instead of displaying the actual LOB data by default, the LOB DA, consisting of the LOB data type name and defined length of the LOB data, will be displayed. The LOB DA is specified through the setting of edit code ’M’ (new in QMF Version 8.1) on every LOB column by default. See “Specifying punctuation for the values in a column” on page 123

Greater than 32 KB support for LOB data type columns

The following EXEC SQL statements are necessary in order to determine the potential number of host variables required to deal with large LOB (greater than 32 KB up to 2 GB).

- Declare cursor:

  ```
  EXEC SQL DECLARE C1 CURSOR FOR 'SELECT LENGTH(CLOB_COL) FROM LOB_TABLE'
  ```

- Open cursor:

  ```
  EXEC SQL OPEN CI
  ```

- Specify End-of-Data condition:

  ```
  EXEC SQL WHENEVER NOT FOUND CONTINUE
  ```

- Execute FETCH:

  ```
  EXEC SQL FETCH CI TO :LOB_ALEN
  ```

The result of these statements will be a one-column result table with each row containing a 4 byte length. Each length will be set into the field LOB_DLEN for the particular LOB record.

A minimimum of five data definition language (DDL) SQL queries must be issued for each request to create a LOB table. For *n* number of LOB columns, *2 + 3n* CREATE statement queries are required. LOB tables will not function if any part of the definition is missing. Here are five required steps to create a complete definition for a LOB table:

1. Create a base LOB table

   ```
   CREATE TABLE LOB (COLCHAR(8), CLOB CLOB(4K), BLOB BLOB(4K),
   ID ROWID NOT NULL GENERATED BY DEFAULT) IN DSQDBDEF.DSQTSDEF
   ```

   Tables with LOB columns must also have a ROWID column. The LOB length can be up to 2 GB.
2. Create an index on the LOB table

   ```
   CREATE TYPE 2 UNIQUE INDEX MCOATES.LOBID ON MCOATES.LOB(ID)
   ```
3. Create tablespace for each LOB column

   ```
   CREATE LOB TABLESPACE LOBTB IN DSQDBDEF LOCKSIZE LOB USING STOGROUP
    DSQSGDEF PRIQTY 1 SECQTY 0 BUFFERPOOL BP0 CLOSE NO
   CREATE LOB TABLESPACE LOBTB2 IN DSQDBDEF LOCKSIZE LOB USING STOGROUP
    DSQSGDEF PRIQTY 1 SECQTY 0 BUFFERPOOL BP0 CLOSE NO
   ```
4. Create an auxiliary table for each LOB column

   ```
   CREATE AUX TABLE MCOATES.AXCLOB IN DSQDBDEF.LOBTB STORES
   MCOATES.LOB
   COLUMN CLOB
   CREATE AUX TABLE MCOATES.AXBLOB IN DSQDBDEF.LOBTB2 STORES
   MCOATES.LOB
   COLUMN BLOB
   ```
5. Create an index for the auxiliary table

   ```
   CREATE INDEX MCOATES.AXCLOBX ON MCOATES.AXCLOB
   CREATE INDEX MCOATES.AXBLOBX ON MCOATES.AXBLOB
   ```

   Table-name, database-name, and table-space-name are required host variable parameters.

Using DBCS data in QMF

The following sections explain how using DBCS data in QMF is different from using SBCS data.

Using DBCS data in commands and procedures

You must issue QMF commands in English (SBCS). However, you can write the following parts of commands and procedures by using double-byte characters:

- Substitution variable names and values

- Comments
- Object names

  Object names are the names you supply with commands such as CONVERT, DRAW, and DISPLAY. If your database manager specifically supports double-byte characters in table names, you can use double-byte characters in object names only if you surround the characters with SO and SI delimiters and do not include any DBCS character that is represented internally with a single-byte double-quote character (EBCDIC code x'7F').
- Table names

  Unless your database specifically supports double-byte characters in table names, table names cannot contain any double-byte characters that are internally represented with single-byte double quotation marks.

Using DBCS data in Input fields

All QMF input fields allow DBCS data if you are using a DBCS display terminal.

Your keyboard can lock while you are keying DBCS data. This indicates that you might not have allowed for the SI character at the end of a field (or line in the Show Field window of the Table Editor). If this happens, press the Reset key on your keyboard and then press Enter to continue. In the Show Field window of the Table Editor and on SQL query and procedure panels, the SI/SO and SI/blank/SO characters are stripped out each time you press Enter. This means that you might have extra space in these input fields after you press Enter.

Using DBCS data in queries

In queries, the following items can be represented in either double-byte characters or mixed single-byte and double-byte characters:

- Column, table, and query names

  Unless your database specifically supports double-byte characters in table names, column names cannot contain any double-byte characters that are internally represented with single-byte double quotation marks.
- Substitution names and values
- Quoted strings in character data type fields
- Comments
- QBE example elements.

  The first character must be a single-byte underscore character. Length limits are the same for SBCS or DBCS data, even though a double-byte character is twice the length of a single-byte character.

In queries, graphic strings that you want to enter or compare to graphic data type fields must be in double-byte characters only. A graphic string consists of either a G or N literal, a single quotation mark, followed by the double-byte character string, and ends with a single quotation mark.

If you are writing a prompted query with a LIKE operator and enter a left-side value with an N literal, when you issue a CONVERT TO SQL command, the query shows a G instead of the N that you entered.

Using DBCS in form panels

You can use DBCS or mixed data in form panels as:

- Column headings
- Break text
- Page text
- Final text
- Form names

Double-byte characters can also appear in the FORM as column labels. For more information about column labels, see *DB2 QMF Reference* .

The following descriptions show ways in which DBCS or mixed data differs from the SBCS data.

Report Width: The report width shown at the top of the FORM.MAIN panel indicates the width of the current report in character positions. Indents, SO and SI delimiters, and characters are counted to calculate this width. A single-byte character counts as one position; a double-byte character counts as two positions. SO and SI delimiters count as one position each.

Column Headings: The underscore (_) character on the form indicates where an SBCS column heading breaks and continues on another line. DBCS data strings can be broken if the underscore used is a single-byte character.

Column headings appearing on the default form are the same as the column names in the database table, unless your installation uses column labels. For columns that have column labels, the labels appear in place of column names on the default form.

USAGE: Form usage codes must be single-byte characters.

INDENT: The leading SO value in a column with graphic data type appears in the indent space. Therefore, the indent value for a graphic data column must be 1 or more when the leading character in the column is SO.

WIDTH: The width of a column is specified in number of characters. Although a double-byte character is twice as wide as a single-byte character, a character of either type is always counted as one character when calculating the column width.

With mixed SBCS and DBCS data (in columns with a data type of character), the SO and SI delimiters are counted as part of the column width. If you are using double-byte characters in a column with a data type of character, the width of that column shown on FORM.MAIN and FORM.COLUMN should be 4 or larger. The minimum column width to display one double-byte character is 4.

With DBCS data (in columns with a data type of graphic), the leading SO is not counted in the column width, but the SI character is part of the width. The minimum column width shown on FORM.MAIN and FORM.COLUMN for columns with a data type of graphic is 1.

EDIT codes: Edit codes must be entered on the form in single-byte characters. Edit codes that begin with G can only be used with DBCS data. Codes that begin with C can be used with either DBCS or mixed data.

Table 15. How DBCS characters display with different edit codes

Edit code	Purpose	Effect on display
C	Columns of data defined as character type	The display of a value is unchanged.
G	Columns of data defined as graphic type	The display of a value is unchanged.
CW	Columns of character data you want wrapped	The display of a value is unchanged, but, if the value cannot fit on one line in the column, CW tells QMF to wrap the text according to the width of the column. Instead of cutting off the data at the end of the column, QMF puts as much data as it can on a line in the column, then wraps the data onto the next line. When you use the CW edit code for a column that contains mixed data, the minimum width for the column is 4.

Table 15. How DBCS characters display with different edit codes (continued)

Edit code	Purpose	Effect on display
GW	Columns of graphic data you want wrapped	The value itself is unchanged, but if the value cannot fit on one line in the column, GW tells QMF to wrap the text according to the width of the column. Instead of cutting off the data at the end of the column, QMF puts as much data as it can on a line in the column, then wraps the remaining data on subsequent lines.
CT	Columns of character data you want wrapped according to the column text	The value itself is unchanged, but if the value cannot fit on one line in the column, CT tells QMF to wrap the column according to the text in the column. Iinstead of cutting off the data at the end of the column, QMF fits as much data as possible on a line, interrupts the line when it finds a single-byte blank, and continues wrapping the data on the next line. If a string of data is too long to fit in the column and does not contain a single-byte blank, QMF wraps the data by width until it finds a single-byte blank and can continue wrapping by text. When you use the CT edit code for a column that contains mixed data, the minimum width for the column is 4.

Table 15. How DBCS characters display with different edit codes (continued)

Edit code	Purpose	Effect on display
CD*x*	Columns of character data you want wrapped according to a delimiter	QMF begins a new line in the column each time it sees a special delimiter in the text. In this edit code, *x* is the special delimiter that can be any single-byte character, including a blank. It does not appear in the output. QMF does not permit column wrapping of graphic data by delimiter. QMF will wrap columns of mixed data by delimiter if the delimiter is outside the DBCS data string. When you use this edit code for a column containing mixed data, the minimum width for the column is 4. If a string of data is too long to fit in the column and does not contain a delimiter, QMF wraps the data by width until it finds a delimiter and can continue wrapping by it. If a string of data contains multiple successive delimiters, QMF inserts a blank line for each one after the first. For example, if the data contains two delimiters, QMF begins a new line when it gets to the first delimiter, skips a line when it gets to the second delimiter, and then continues wrapping the output.
U*xxxx* and **V***xxxx*	Custom edit codes defined at your installation	Format data in ways you define using an edit exit routine you write. Replace *xxxx* in this code with an identifier that names a unique code. See your QMF administrator for a description of your available custom edit codes.

How incorrect DBCS data is handled

When an SO or SI character is missing from a DBCS data string, the existing SO or SI character appears as a question mark. All other double-byte data appears as single-byte characters and is meaningless.

How data truncation Is handled

QMF truncates displayed DBCS data at a field or a screen boundary in a way that avoids splitting double-byte characters. Scrolling is necessary to view the characters on the truncated lines.

SO or SI delimiters are added where truncation occurs. Set the scrolling value to less than the screen width on report panels and QBE query panels to ensure that you do not miss characters that are out of the regular screen view.

Exporting DBCS data

You can export data that is defined as graphic and variable graphic. Developing QMF Applications describes export data file formats in detail.

The data type codes for the header records of exported data are 464 for VARGRAPHIC or 468 for GRAPHIC.

The column width of exported data is the number of double-byte characters in it, which is half the number of bytes that are used to store it. Column data is stored in the data record exactly as it comes from the database, except that SO and SI delimiters are added.

Importing DBCS data

DBCS data can be imported in queries, procedures, and forms. When importing DBCS data in this indirect way, be certain that the record length does not exceed 79 bytes. Also be sure that the data is enclosed in SO and SI delimiters. Data that does not meet these requirements appears as meaningless single-byte characters.

You can also import DBCS data as data by using the IMPORT DATA command. QMF validates it as the data is imported. If the DBCS data is not valid, the import stops. For more information about how to import QMF objects, see *DB2 QMF Reference* and *Developing DB2 QMF Applications*.

Printing DBCS reports

With a DBCS printer, you can print reports that contain DBCS data even if you do not have a terminal that displays DBCS data. See your QMF administrator for information about how to do this.

You can also print any objects that contain DBCS data from the Database Object List panel whether or not you have a DBCS terminal. However, if the object name contains double-byte characters, and you have a non-DBCS terminal, all double-byte characters are altered. When you enter the PRINT command next to a DBCS object, clear the rest of the row for that object before pressing Enter.

If you are using DBCS data and QMF splits the page, printing on the second and subsequent pages of the report resumes at the fourth byte position from the left side of the page.

Chapter 15. Using QMF with other products

This chapter discusses ways to use QMF with other products to enhance the extraction, manipulation, and reporting of data. Using other products with QMF provides access to a broad range of functions and services. You can use QMF with products such as:

- Data Extract (DXT™) End User Dialogs
- IBM VM/System Product Editor (XEDIT)
- Interactive System Productivity Facility (ISPF)

For example, while using QMF you can access data that is not currently stored in the database you are using. By entering the EXTRACT command, you can gain access to DXT End User Dialogs. Or, you can submit requests to DXT to extract data from various databases and files.

You can access other products from the QMF Home panel or from any other QMF panel. When you do, you might see panels that are not QMF panels. For example, if you use DXT End User Dialogs, you might see the DXT End User Dialogs main menu panel. Or, if you use ISPF, you might see the Interactive System Productivity Facility-Program Development Facility (ISPF-PDF) primary option menu, and so on. However, using other products from QMF has no effect on any other QMF operation and does not disrupt the normal sequence of events. When you exit the product, you return to QMF at the point where you left it. In addition, you can display and manipulate QMF objects, reports, and query results from within nearly any Windows application that is supported by the QMF for Windows feature. See Appendix C, "The DB2 QMF High Performance Option," on page 367 for more information.

This chapter introduces the commands you use to access each of the interface products. For the syntax of the commands, see *DB2 QMF Reference*.

Using DXT end user dialogs

If your installation has DXT, you can access all the functions of DXT End User Dialogs while in QMF. You can send a predefined extract request to DXT for processing, create a new extract request, or update an existing extract request. You can load extract output into physical sequential files, relational tables, or other output targets that are supported by DXT.

To use DXT from QMF, issue the EXTRACT command. Depending upon how you specify the command, it either invokes the DXT End User Dialogs to let you create an extract request or update an existing one. Or, it sends a named data extract to DXT to run.

When you access DXT End User Dialogs from QMF, you remain there until you choose to exit, then you return to the QMF environment.

You can access DXT from QMF in either batch or interactive mode. However, in batch mode, you cannot perform operations that display a panel. In addition, you cannot display a panel through the QMF command interface unless you specify INTERACT.

Your QMF administrator might have already set you up to use DXT. If you do not have all three of the following items or you are not sure whether you do, contact your QMF administrator. You cannot complete a dialog until these items are in place:

- Your authorization information and connection information must be identified to DXT End User Dialogs.
- Your JCL/JCS files and CLISTs or execs must already exist and have all the appropriate routing information.
- Your control profile must be set up and complete.

For information about using DXT, see the *Data Extract: Users Guide*.

To display the DXT main menu: To go directly to the main menu panel in DXT End User Dialogs, enter `EXTRACT` on the command line of any QMF panel.

When the main menu panel displays, you can choose from the options available for building or updating an extract request. You can select any menu option by pressing a function key or by entering a letter on the command line.

When you are ready to return to QMF, exit DXT End User Dialogs.

To send an extract request to DXT from QMF: Issue the EXTRACT command, including the name of the extract request. For example, enter:

```
EXTRACT extract-name  (PASSWORD=
```

QMF sends the named extract request to DXT for processing. However, DXT panels do not appear; it appears as if you never left QMF.

You need a password when you give an extract name for a relational DB2 or DB2 Server for VSE or VM table. The password you enter does not appear on your screen.

If no errors result from the request, QMF returns the message `Extract request successfully sent` on the message line of your screen. You can immediately resume whatever QMF activity you were performing.

If an error related to the request results, QMF displays a message that contains a QMF interpretation of the DXT End Use Dialog's return code.

To display the EXTRACT command prompt panel: On the QMF command line, enter:

```
EXTRACT ?
```

The EXTRACT Command Prompt panel displays. The panel also displays if you enter the EXTRACT command incorrectly twice in succession.

To send the extract request to the DXT End User Dialogs for processing, enter a valid extract name on the panel. You then return to the QMF environment.

Editing objects from outside QMF using ISPF

Note to CICS users

You cannot use an editor from QMF under CICS. You can, however, change a QMF object while viewing it in temporary storage.

You can edit an existing QMF procedure or SQL statement from QMF. The QMF object you edit can be a new, changed, or imported procedure or query. You cannot edit QBE and prompted queries.

QMF supports the ISPF-PDF editor and the XEDIT editor. You can name a user exec (VM) or CLIST (z/OS) that initializes another editor and optionally performs housekeeping functions. The ISPF-PDF editor is the default editor, but if you want to use the ISPF-PDF editor, you must do one of the following:

- Start QMF as an ISPF-PDF dialog.
- Name a user exec or CLIST to set up ISPF and start the PDF editor.

To find out about editors you can use, see your QMF administrator.

To edit an object using ISPF-PDF: To use the ISPF-PDF editor, you must be using ISPF. To display the ISPF-PDF editor and the current query or procedure, enter:

```
EDIT object
```

Where *object* is either PROC or QUERY.

From a PROC or QUERY panel, you can enter the EDIT command without specifying a value for *object*. The procedure or query displayed in the panel is edited. EDIT ? prompts you with the default *object*, either PROC or QUERY, depending on which type of panel you are using when you initiate the command.

When you end the edit session, you return to QMF with the edited object in QMF temporary storage.

You can edit your SQL statements or procedure in a different ISPF application ID by using an exec or CLIST as the editor name of the QMF EDIT command.

To edit an object using XEDIT: To use the XEDIT editor, you must be using CMS. To display the current query or procedure, issue the EDIT command:

```
EDIT object (EDITOR=XEDIT
```

Where *object* is either PROC or QUERY.

When you end the edit session, you return to QMF with the edited object in QMF temporary storage.

To edit an object using a CLIST: To use a CLIST, you must be using TSO. The named editor represents a user's CLIST. For example, enter the following command, where the editor is named MYCLIST:

```
EDIT object (EDITOR=MYCLIST
```

Where *object* is either PROC or QUERY.

Using an editor of your choice, run this CLIST to edit the current query or procedure.

When you end the edit session, you return to QMF with the edited object in QMF temporary storage.

To display the EDIT command prompt panel:

1. On the QMF command line, enter:

   ```
   EDIT ?
   ```

   The EDIT Command Prompt panel displays.
2. To start an edit session, enter either `QUERY` or `PROC`. Another EDIT Command Prompt panel appears.
3. Specify the editor you want to use. PDF is the default editor.

4. Press Enter. QMF displays the appropriate panel for the editor you requested containing the current QUERY or PROC object (the object you last worked on).
5. To return to QMF, exit the editor.

Using ISPF from QMF

To access the ISPF-PDF product from QMF, you must start QMF as an ISPF dialog.

You can access the ISPF-PDF product from QMF in two ways:

- Access the ISPF-PDF primary option menu panel from which you can choose an application.
- Display a specific ISPF-PDF panel.

When you have access to ISPF-PDF, you can use any of the processing options available.

To access the ISPF-PDF primary option menu panel: On the QMF command line, enter:

```
ISPF
```

From the ISPF-PDF primary option menu panel, you can start whatever applications you normally use in ISPF. (While in VM, you can run only those functions that run in CMS subset mode.) All available command options appear on the menu. You can select any of them by entering a letter on the command line or by pressing a function key.

To return to QMF, exit ISPF-PDF.

To display a specific ISPF-PDF panel: Enter the panel identifier as a parameter to the ISPF command. For example:

```
ISPF 3
```

This starts the application identified as **Option 3** on the ISPF-PDF primary option menu panel. The specific panel you see depends on your installation.

To return to QMF, exit ISPF-PDF.

Inserting a QMF report in a document

Note to CICS users

You cannot use the document interface from QMF under CICS.

In an edit session, you can insert a QMF report into the document you are editing without leaving the session. Use the GETQMF macro to insert the report. The GETQMF macro is not a QMF command.

You can insert an existing QMF report into a document or generate a new QMF report by using QMF either interactively or through the command interface. You can also format the QMF report by using SCRIPT/VS control words that are used by the Document Composition Facility (DCF).

Before inserting the QMF report into a document, you must print it from within a QMF session.

The syntax of the GETQMF macro is:

```
GETQMF type option
```

type specifies whether SCRIPT/VS control words are also inserted. Descriptions of the following types appear under “Formatting the report.”

DCF For a SCRIPT/VS document

ASIS For inserting a QMF report “as is”

option specifies whether you are creating a new report or inserting an existing one. Descriptions of the following options appear under “Inserting a report” on page 271.

USEQMF
To create a QMF report dynamically

FILE To insert an existing QMF report (VM only)

DSN To insert an existing QMF report (OS/390 only)

Formatting the report

You can specify whether you want your report formatted for a DCF document, for a PROFS document, or left as it is.

DCF type

The QMF report you identified or produced is inserted into your document with SCRIPT/VS control words. For example, enter from your editor:

```
GETQMF DCF USEQMF
```

DCF places SCRIPT/VS control words before and after the QMF report. Additionally, each printer page eject is replaced by a SCRIPT/VS page eject. SCRIPT/VS control words are placed at the heading and footing of each page.

QMF report length and width must be considered when QMF reports are included within a SCRIPT/VS document. Editor settings always override QMF report characteristics. Use the following specifications on the QMF PRINT command:

- Use a length of 56 lines per page.
- A width of 70 characters is suggested to print on a 6670 information distributor in a non rotating mode. The number of characters per line vary with the DCF print arrangement selected. If the report is too wide to fit in the document, it is inserted anyway. However, a warning message is issued, and the lines that are too long to fit are wrapped (for ISPF-PDF) or truncated (for XEDIT). Wrapping or truncating occurs only when you are inserting an existing QMF report into a document. When you create a new report interactively in QMF, the lines are not too long.

ASIS type

The QMF report you identified or produced is inserted into your document with no alterations, “as is”. For example, enter from your editor:

```
GETQMF ASIS USEQMF
```

ASIS is the default.

Inserting a report

You can insert a new or existing QMF report into another document.

- The USEQMF option inserts a new report
- The FILE option (in VM) inserts an existing report
- The DSN option (in z/OS) inserts an existing report

The USEQMF option

The USEQMF option allows you to insert a QMF report into another document without leaving your QMF session. You might need to initialize the system environments.

When QMF is not active: You are using XEDIT, ISPF-PDF, PS/TSO, or the CMS NOTE facility, and you want to generate a report from QMF and insert it into the document (or note) you are working on. For example, enter from your editor:

```
GETQMF DCF USEQMF
```

This causes the GETQMF macro (with the USEQMF option) to start an interactive QMF session. QMF uses a default initial procedure when it starts. When you are in QMF, you have the full interactive capability available to produce your report. After your report is finished, remember to print it using the PRINT REPORT command. QMF issues ISPF messages and does not allow you to leave QMF with the END command until a QMF report is printed. The ISPF messages and associated help panels tell you how to print a report for the document interface and return to your editor.

If you specified a procedure name after USEQMF, it runs as an initial procedure when you start QMF. You must specify an EXIT command in the procedure to end QMF, or you must manually exit from the QMF session. The END command runs the procedure again.

When QMF is active: You are using QMF, and you want to insert a report into a document outside the QMF environment.

While you are still in QMF, access an ISPF-PDF or XEDIT session through the ISPF bridge or with a CMS XEDIT command. Then, edit your target document outside the QMF environment. After you start the editor, prepare it to receive the new report in the proper place in the document. (This procedure is discussed in “Information about your editor” on page 273.)

With QMF active, you must enter a QMF procedure name after the USEQMF option. For example, enter from your editor:

```
GETQMF DCF USEQMF MYPROC
```

Where MYPROC is the name of a QMF procedure that runs through the QMF command interface and generates a report. If you want to run a shared procedure that you do not own, specify it as *owner.yourproc*. You must specify USEQMF to use the procedure. To call the document interface, enter `GETQMF`. If your procedure printed a report, the report appears in your document. You can save the document and return to QMF.

Your QMF session ends if you use a procedure that issues the EXIT command.

You must use a QMF procedure to produce your QMF report. You cannot produce a query in QMF at the point when you are at your edit session from the QMF document interface.

Using the FILE option

Use FILE if you are using VM and want to insert an existing QMF report. You must follow FILE with the file name, file type, and file mode. For example, enter from your editor:

```
GETQMF DCF FILE fn ft fm
```

Where *fn ft fm* is the name of the file that contains the chart or report to insert. (If file mode is not specified, it defaults to A1.) Lines in the inserted file might be truncated or wrapped.

You can also create a report interactively and direct it to a file (which becomes an existing report) in one step by including USEQMF before the FILE option:

```
GETQMF DCF USEQMF FILE fn ft fm
```

The report is then inserted into your document.

Using the DSN option

Use DSN if you are using OS/390 and want to insert an existing QMF report. You must follow DSN by the fully-qualified data set name. For example, enter from your editor:

```
GETQMF ASIS DSN dataset name
```

Where *dataset name* is the name of the data set that contains the chart or report to be inserted. Lines in the inserted file might be truncated or wrapped.

You can also create a report interactively and export it to a data set (which becomes an “existing” report) in one step by including USEQMF before the DSN option:

```
GETQMF ASIS USEQMF DSN dataset name
```

The report is then inserted into your document.

Information about your editor

You can insert a QMF report into a document while using one of the following products:

- XEDIT
- ISPF-PDF
- PS/TSO
- CMS NOTE facility

XEDIT

When you use XEDIT, the QMF report is inserted in your document after the current line. The new current line is the last line of the inserted report. This is similar to the XEDIT GET command.

You cannot go from XEDIT to interactive QMF through the document interface and then start another XEDIT session by using the CMS XEDIT command. Your original XEDIT environment is lost when you exit QMF.

ISPF-PDF

ISPF-PDF is available in both VM and z/OS. When you use ISPF-PDF, the QMF report is inserted into your document after the line where you enter A, or before the line where you enter B in the prefix area. If you do not choose a line, the report is inserted at the end of the document. The top line displayed after insertion is the line that immediately precedes the inserted report. This is similar to the ISPF-PDF COPY command.

PS/TSO
If you are using the Personal Services for TSO Extensions (PS/TSO), you are using the ISPF-PDF editor. The information that was previously given for ISPF-PDF applies here.

CMS NOTE
If you are using CMS NOTE, you are using XEDIT. See the information for XEDIT.

Restrictions on the document interface

- When printing a report to insert into a document, you cannot use a GDDM printer nickname. The QMF document interface sets a PROFILE value of PRINTER=' ' if you enter QMF through the QMF command interface or interactively using the default initial procedure. When you are running your own initial procedure, make sure that your PROFILE setting contains `PRINTER=' '`. Or you can specify it on the PRINT command.
- You cannot shorten GETQMF, but you can enter its parameters using minimum unique representation. You only need one character for VM and two for z/OS (in English). The exception to this is when you specify USEQMF and FILE or USEQMF and DSN instead of a procedure name. In these cases, anything other than FILE in VM or DSN in z/OS is taken as a procedure name.
- You cannot nest the document interface.
- The ISPF-PDF DEFINE command should not be used to redefine current ISPF-PDF commands.
- No prompt panel or help panel appears with the GETQMF macro because GETQMF is not a QMF command. If QMF uses the default initial procedure, there are help panels for the document interface messages in QMF.

After you install QMF, and it is running successfully, you need to tailor the document interface.

Using the QMF document interface

Although you might not use all the products and environments, you should look at each one to see the different ways the document interface can be used. This section shows examples of inserting QMF reports into documents under four conditions:

- Accessing QMF from a VM editor
- Accessing a VM editor from QMF
- Accessing QMF from a z/OS editor

- Accessing a z/OS editor from QMF

Accessing QMF from a VM editor

The following examples issue the GETQMF macro from:

- XEDIT, or the CMS NOTE facility
- XEDIT, or ISPF-PDF
- XEDIT
- ISPF-PDF

Example 1—From XEDIT or the CMS NOTE facility: The existing QMF report file XX MYREPORT A1 is inserted as is. Use the FILE option to specify the name of the CMS file that contains the QMF report. The insertion occurs without a QMF session.

1. In XEDIT, position your document to insert the QMF report in the proper place (see “Information about your editor” on page 273.)
2. Enter the GETQMF macro on the command line.

   ```
   GETQMF ASIS FILE XX MYREPORT
   ```

   The report named XX MYREPORT A1 is inserted directly into the document you are working on, just after the current line. You get a message that indicates that the report is inserted.

Example 2—From XEDIT or ISPF-PDF: The existing QMF report named XX MYREPORT A1 is inserted into your document as is. (ASIS is the default.) The procedure is the same as in Example 1:

```
GETQMF FILE XX MYREPORT
```

Example 3—From XEDIT: This example inserts a new report into your document and shows the minimum abbreviation of the USEQMF option.

1. In XEDIT, position your document to insert the QMF report after the current line (see “XEDIT” on page 273).
2. Enter the GETQMF macro:

   ```
   GETQMF U
   ```

   (U is the minimum abbreviation for the USEQMF option.)

   Your screen is blank for a few minutes while the macro runs.
3. When the QMF Home panel appears with a document interface message, produce a report as you normally would in QMF.
4. Alter the form of the report if you want.
5. Display the report to check it.
6. Enter `PRINT REPORT`.

7. Enter `END` or `EXIT` to exit QMF.

   The report is inserted into your document, and you return to XEDIT where you were before you issued the GETQMF macro.

If you type EXIT on the QMF command line without printing a report, you return to XEDIT. An error message appears, and no report is inserted.

Example 4—From ISPF-PDF: The USEQMF option specifies that QMF is used to produce a report during the edit session. The named procedure MYPROC runs to produce the report.

1. From a document in ISPF-PDF, insert a prefix command A (after) or B (before) to receive the report at the proper place (see "ISPF-PDF" on page 273).
2. Enter the GETQMF macro:
   ```
   GETQMF DCF USEQMF MYPROC
   ```
   Your screen is blank for a few minutes while MYPROC creates and prints a report.
3. When the QMF object panel appears, enter `EXIT` to exit QMF.

   The report is inserted into your document (with SCRIPT/VS control words) when QMF ends.

   Use EXIT to leave QMF; the END command runs the initial procedure again.

Accessing a VM editor from QMF

In QMF, this example creates and inserts a report into a document with XEDIT.

1. On the QMF command line, issue the CMS XEDIT *fn ft fm* command, where *fn ft fm* is the CMS file name of the target document.
2. Position your document to insert the report after the current line.
3. On the command line, enter the GETQMF macro:
   ```
   GETQMF DCF USEQMF MYPROC1
   ```
   The GETQMF macro runs the MYPROC1 routine in QMF. MYPROC1 creates and prints the report. The report is inserted into your document.
4. Save the document and return to QMF.

   Your QMF session ends if you use a procedure that issues an EXIT command.

Accessing QMF from a z/OS editor

The following examples issue the GETQMF macro from ISPF-PDF and PS/TSO.

Example 1—From ISPF-PDF: The USEQMF option specifies that QMF is used to produce the report during the edit session.

1. Choose where you want the document inserted by using the A (after) or B (before) prefix commands.
2. From your ISPF-PDF edit session, enter the GETQMF macro:

   ```
   GETQMF USEQMF MYPROC
   ```

   Your screen is blank for a few minutes while MYPROC creates and prints a report.
3. When the QMF object panel appears, enter an EXIT command to exit QMF.

   The report is inserted into your document as is when QMF ends.

   Use EXIT to leave QMF; the END command runs the initial procedure again.

Example 2—From ISPF-PDF or PS/TSO: The QMF report data set *userid*.MYREPORT is inserted into the user's document as is.

1. From a document in ISPF-PDF or PS/TSO, insert a prefix command A (after) or B (before) to receive the new report at the proper place.
2. On the command line, enter the GETQMF macro:

   ```
   GETQMF ASIS DSN userid.MYREPORT
   ```

   The macro gets the existing report, *userid*.MYREPORT. Then, it inserts it into your document, and returns you to the ISPF-PDF or PS/TSO editor you were using before you issued the GETQMF macro.

Example 3—From ISPF-PDF: The QMF report is produced interactively in QMF.

1. Enter the GETQMF macro:

   ```
   GETQMF ASIS USEQMF
   ```

   Your screen is blank for a few minutes while the macro is running.
2. When the QMF Home panel appears with a document interface message, produce a report as you normally would in QMF.
3. Alter the form of the report if necessary.
4. Display the report to check it.
5. Enter `PRINT REPORT`.
6. Enter an END or EXIT command to exit QMF.

   The report is inserted into your document, and you return to ISPF where you were before issuing the GETQMF macro.

If you type `EXIT` on the QMF command line without printing a report, you are returned to ISPF. An error message displays, and no report is inserted.

Accessing a z/OS editor from QMF

When you are using QMF, you can create a report and insert it into a document with PS/TSO. To run this example, use the ISPF command to bridge to ISPF-PDF and define the data set where your target document is located.

From your PS/TSO session:

1. Prepare your document to insert the new report at the correct location in the document. ("ISPF-PDF" on page 273 discusses this procedure.).
2. Enter the GETQMF macro:

   ```
   GETQMF ASIS USEQMF MYPROC4
   ```

   The GETQMF macro runs the MYPROC4 routine in QMF, and the report is inserted into your document.
3. Save your document.

   You return to QMF at the point where you issued the ISPF BRIDGE command.

   You lose your QMF session if you use a procedure that issues an EXIT command.

Chapter 16. Using DB2 stored procedures with QMF

This chapter tells you how to develop stored procedures to run from QMF, how to format the CALL statement, how to run a CALL statement from the SQL QUERY panel, how to use QMF forms with result sets returned from stored procedures, and shows you examples of running the stored procedure.

QMF allows you to issue a DB2 CALL statement, store your output parameters as QMF global variables, and display a single result set. To run a stored procedure from QMF, enter a CALL statement from the SQL QUERY panel.

Developing the stored procedures to run from QMF

QMF provides a stored procedure development environment through the DB2 QMF High Performance Option (HPO) feature. For more information, see Appendix C, "The DB2 QMF High Performance Option," on page 367.

The IBM Redbook, *Cross-Platform DB2 Stored Procedures: Building and Debugging*, is another good source to learn how to build stored procedures. This Redbook covers three methods that can be used to develop stored procedures including using the Stored Procedure Builder, which is recommended if you do not use the DB2 QMF High Performance Option (HPO).

Running a CALL statement from the SQL QUERY panel

When you run a stored procedure from QMF, you enter your CALL statement on the SQL QUERY panel and then use the RUN command as you would when running any QMF query. For more information on using the SQL QUERY panel, see Chapter 5, "Viewing the data in the database using SQL statements," on page 69.

Format of the CALL statement

Most simple CALL statements use the following format:

```
CALL spname(parm1, parm2, parmx)
```

Example: This basic CALL statement is for a stored procedure that has two input parameters that are defined as decimal data with a precision of five and a scale of two.:

```
CALL USERAPROC1(123.45,1.22)
```

For more information about the CALL statement, see *DB2 QMF Reference*. The QMF supported syntax is slightly different from the DB2 CALL statement.

To enter the CALL statement on the SQL QUERY panel:

- You must be authorized to run the stored procedure and know the name of the stored procedure that you want to run.
- Parameters must be entered in the order that the stored procedure is expecting them. For example, if a stored procedure has two parameters (the first being a decimal and the second being a character string), you must enter the parameters in that same order on your CALL statement. You must know the number of expected parameters and the data type of each parameter. Output parameters must be initialized to blank character for character strings, or zero for numeric data. QMF global variables can be used to enter input, inout, and to receive output parameters. You must use QMF global variable output or inout parameters if you want to view the returned data. Character strings, all numeric data types, date, time, and datestamp data types are supported for input, output, or inout parameters. Thirty-two parameters is the maximum that can be entered on the CALL statement. The maximum amount of data that can be entered for a parameter is 32 KB.
- DATE, TIME, and TIMESTAMP data types must be enclosed in single quotes.
- When entering decimal data, the scale must match exactly what the stored procedure expects, or you will receive an error message.
- When you run a stored procedure that returns result sets, QMF supports the return of the first 32 result sets. Select the one that you want to use by setting the global variable DSQEC_SP_RS_NUM. The default setting is one. To ignore the result sets, set the global variable to zero.

Examples of running a stored procedure

QMF does not provide the source code for the stored procedures used in the following examples. You can develop the stored procedures on your system using DB2 QMF HPO or the Stored Procedure Builder. Details of the stored procedures input and output parameters and returned result sets is provided so that you can develop similar stored procedures to follow along with the examples.

Running a stored procedure that returns no output parameters

To run a stored procedure with no parameters, enter the CALL statement on the SQL QUERY panel. The stored procedure used in this example performs some DB2 maintenance functions. You can set up your stored procedure to do maintenance functions required at your site. Issue the CALL statement on the SQL QUERY panel:

```
CALL USERA.MYPROC1
```

Upon completion, you will see this message:

```
OK, Your Stored Procedure has successfully completed.
```

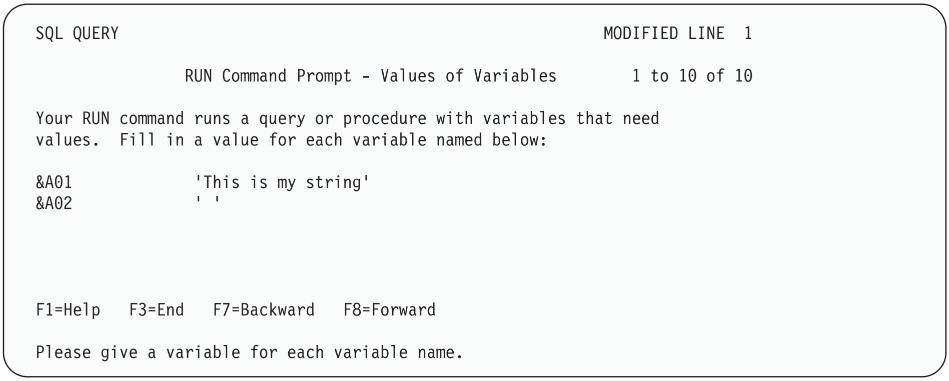
Verify that the maintenance functions provided by your stored procedure were performed.

Running a stored procedure that contains input and output parameters

Stored procedure USERA.IOVCHAR has two parameters, an input and an output parameter which are both defined as VARCHAR 32KB character strings. The stored procedure copies the input string to the output string. You can enter your CALL statement using QMF global variables to handle the input and output parameters. The global variables can be declared before you run the CALL statement using the SET GLOBAL command. If you do not set the value prior to running the CALL statement, you will be prompted to enter a value. Issue the CALL statement on the SQL QUERY panel. Prompting will be used in the following example:

```
CALL USREA.IOVCHAR(&A01, &A02)
```

You enter your data on the lines provided in the prompt panel. You must enter a blank for the output parameter regardless of whether you use prompting or if you set the global variables before running the CALL statement.
Upon completion, you will see this message:

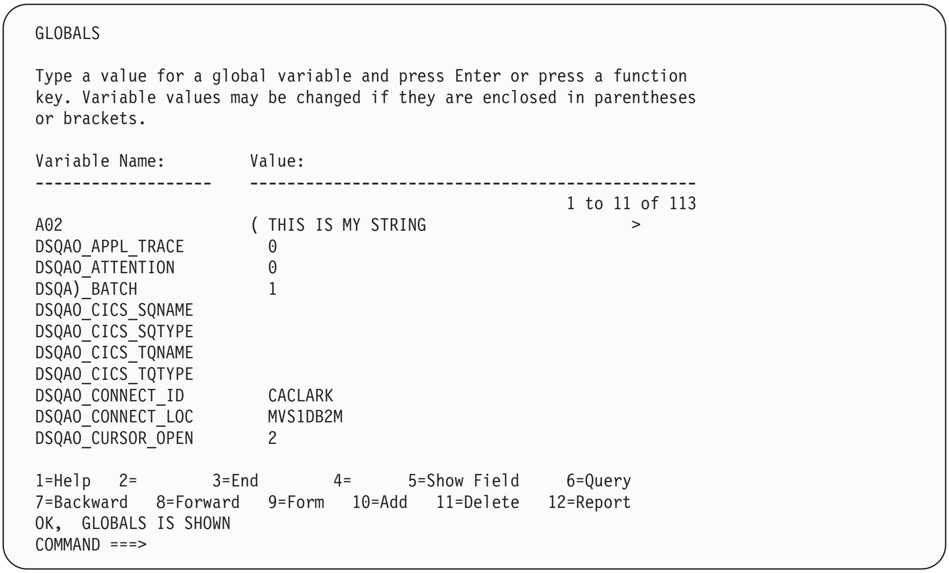


```
SQL QUERY                                                MODIFIED LINE  1

                RUN Command Prompt - Values of Variables        1 to 10 of 10

Your RUN command runs a query or procedure with variables that need
values.  Fill in a value for each variable named below:

&A01             'This is my string'
&A02             ' '




F1=Help   F3=End   F7=Backward   F8=Forward

Please give a variable for each variable name.
```

Figure 180. Prompt panel- input and output parameters

```
OK, Your Stored Procedure has successfully completed.
```

You can view the output parameter in the global variable pool by using the SHOW GLOBALS command:



```
GLOBALS

Type a value for a global variable and press Enter or press a function
key. Variable values may be changed if they are enclosed in parentheses
or brackets.

Variable Name:         Value:
-------------------    ------------------------------------------------
                                                          1 to 11 of 113
A02                    ( THIS IS MY STRING                      >
DSQAO_APPL_TRACE         0
DSQAO_ATTENTION          0
DSQA)_BATCH              1
DSQAO_CICS_SQNAME
DSQAO_CICS_SQTYPE
DSQAO_CICS_TQNAME
DSQAO_CICS_TQTYPE
DSQAO_CONNECT_ID         CACLARK
DSQAO_CONNECT_LOC        MVS1DB2M
DSQAO_CURSOR_OPEN        2

1=Help   2=        3=End        4=      5=Show Field     6=Query
7=Backward   8=Forward   9=Form   10=Add   11=Delete   12=Report
OK,  GLOBALS IS SHOWN
COMMAND ===>
```

Figure 181. Viewing the output parameter by using SHOW GLOBALS

Running a stored procedure that returns a result set

Stored procedure USERA.RET1RS has two output parameters. The first, defined as an integer, returns the SQLCODE. The second, defined as a character string, returns the SQLSTATE. This stored procedure also returns a result set based on the following SELECT statement from the QMF supplied table Q.STAFF:

```
SELECT ID, NAME, DEPT, JOB, YEARS, SALARY, COMM FROM Q.STAFF
```

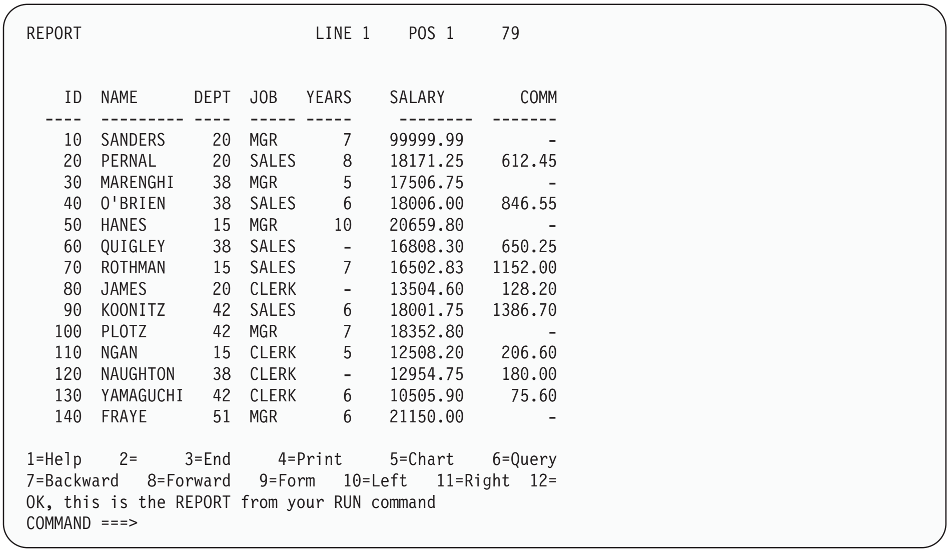
Verify that QMF global variable DSQEC_SP_RS_NUM is set to one. Issue the CALL statement on the SQL QUERY panel:

```
CALL USERA.RET1RS(&A01, &A02)
```

Upon completion, you will see this message:

```
OK, Your Stored Procedure has successfully completed.
```

Your result set is displayed; you can page forward or back to view the entire report.



```
REPORT                       LINE 1    POS 1     79


    ID  NAME      DEPT  JOB   YEARS    SALARY        COMM
  ----  --------- ----  ----- -----    --------   -------
    10  SANDERS     20  MGR       7    99999.99         -
    20  PERNAL      20  SALES     8    18171.25    612.45
    30  MARENGHI    38  MGR       5    17506.75         -
    40  O'BRIEN     38  SALES     6    18006.00    846.55
    50  HANES       15  MGR      10    20659.80         -
    60  QUIGLEY     38  SALES     -    16808.30    650.25
    70  ROTHMAN     15  SALES     7    16502.83   1152.00
    80  JAMES       20  CLERK     -    13504.60    128.20
    90  KOONITZ     42  SALES     6    18001.75   1386.70
   100  PLOTZ       42  MGR       7    18352.80         -
   110  NGAN        15  CLERK     5    12508.20    206.60
   120  NAUGHTON    38  CLERK     -    12954.75    180.00
   130  YAMAGUCHI   42  CLERK     6    10505.90     75.60
   140  FRAYE       51  MGR       6    21150.00         -

1=Help    2=     3=End     4=Print     5=Chart    6=Query
7=Backward   8=Forward   9=Form   10=Left   11=Right  12=
OK, this is the REPORT from your RUN command
COMMAND ===>
```

Figure 182. Result set from USERA.RET1RS

You can view the output parameters in the global variable pool by using the SHOW GLOBALS command.

Running a stored procedure that returns multiple result sets

Stored procedure USERA.RET3RS has two output parameters. The first, defined as an integer, returns the SQLCODE. The second, defined as a character string, returns the SQLSTATE. This stored procedure will also return three result sets based on the following SELECT statements from the QMF supplied tables Q.STAFF, Q.ORG, and Q.INTERVIEW:

```
SELECT NAME, DEPT, SALARY, COMM FROM Q.STAFF
SELECT DEPTNUMB, DEPTNAME, MANAGER, DIVISION LOCATION FROM Q.ORG
SELECT TEMPIDL, INTDATE, STARTTIME, ENDTIME, MANAGER, DISP, LASTNAME,
FIRSTNAM FROM Q.INTERVIEW
```

You can display a result set each time that you run the stored procedure. Indicate the result set to be displayed by setting the QMF global variable DSQEC_SP_RS_NUM. Setting the value to 1 will display the first result set returned, setting to 2 will display the second result set returned, and so on. In the following example, DSQEC_SP_RS_NUM is set to 3 to display the third result set returned:

```
SET GLOBAL (DSQEC_SP_RS_NUM=3
```

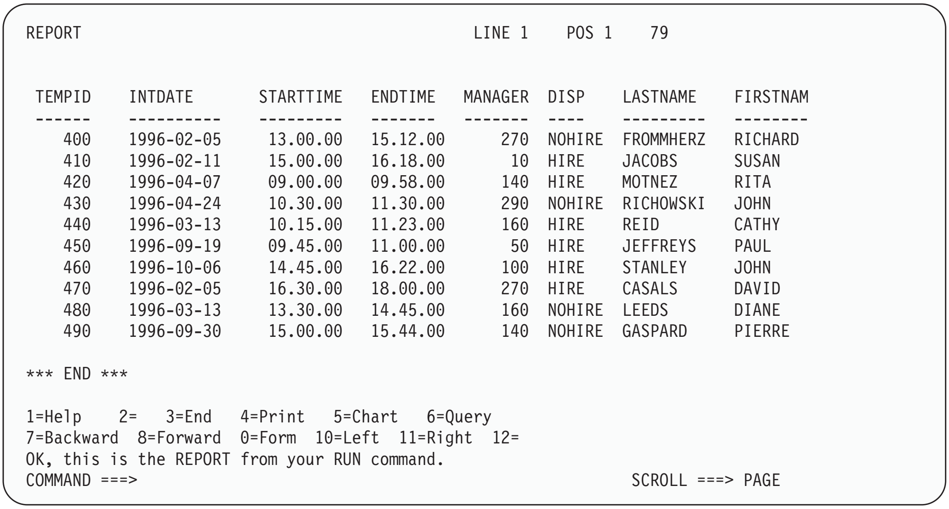
Issue the CALL statement on the SQL QUERY panel:

```
CALL USERA.RET1RS(&A01, &A02)
```

Upon completion you will see this message:

```
OK, Your Stored Procedure has successfully completed.
```

Figure 183 shows an example of a displayed result set:



```
REPORT                                        LINE 1    POS 1    79


 TEMPID    INTDATE       STARTTIME   ENDTIME   MANAGER  DISP    LASTNAME    FIRSTNAM
 ------    ----------    ---------   -------   -------  ----    ---------   --------
    400    1996-02-05     13.00.00   15.12.00      270  NOHIRE  FROMMHERZ   RICHARD
    410    1996-02-11     15.00.00   16.18.00       10  HIRE    JACOBS      SUSAN
    420    1996-04-07     09.00.00   09.58.00      140  HIRE    MOTNEZ      RITA
    430    1996-04-24     10.30.00   11.30.00      290  NOHIRE  RICHOWSKI   JOHN
    440    1996-03-13     10.15.00   11.23.00      160  HIRE    REID        CATHY
    450    1996-09-19     09.45.00   11.00.00       50  HIRE    JEFFREYS    PAUL
    460    1996-10-06     14.45.00   16.22.00      100  HIRE    STANLEY     JOHN
    470    1996-02-05     16.30.00   18.00.00      270  HIRE    CASALS      DAVID
    480    1996-03-13     13.30.00   14.45.00      160  NOHIRE  LEEDS       DIANE
    490    1996-09-30     15.00.00   15.44.00      140  NOHIRE  GASPARD     PIERRE

*** END ***

1=Help    2=   3=End   4=Print   5=Chart   6=Query
7=Backward  8=Forward  0=Form  10=Left  11=Right  12=
OK, this is the REPORT from your RUN command.
COMMAND ===>                                                    SCROLL ===> PAGE
```

Figure 183. The result set from USERA.RET3RS

The output parameters can be viewed in the global variable pool by using the SHOW GLOBALS command.

Using QMF forms with result sets returned from stored procedures

You can change the appearance of the reports that you create from running your stored procedures by changing the report information on the QMF forms panels. The chapter, Chapter 6, “Customizing your reports,” on page 113 will provide you with information about using QMF forms. This information can also be applied to result sets returned from stored procedures.

Chapter 17. QMF sample tables

The following tables in this index contain data about fictional applicants, interviews, parts, products, employees, and suppliers of a fictional company:

- Q.APPLICANT
- Q.INTERVIEW
- Q.ORG
- Q.PARTS
- Q.PRODUCTS
- Q.PROJECT
- Q.SALES
- Q.STAFF
- Q.SUPPLIER

Q.APPLICANT

This table provides information about people who have applied for jobs with the company. Each row represents an applicant. The columns are:

TEMPID
Temporary identification of the applicant

NAME
Last name of the applicant

ADDRESS
City and state in which the applicant lives

EDLEVEL
Education level of the applicant

COMMENTS
Notes made by the interviewer

TEMPID	NAME	ADDRESS	EDLEVEL	COMMENTS
------	---------	-----------------	-------	-----------------------------
400	FROMMHERZ	SAN JOSE,CA	12	NO SALES EXPERIENCE
410	JACOBS	POUGHKEEPSIE,NY	16	GOOD CANDIDATE FOR WASHINGTON
420	MONTEZ	DALLAS,TX	13	OFFER SALES POSITION
430	RICHOWSKI	TUCSON,AZ	14	CAN'T START WORK UNTIL 12/92

TEMPID	NAME	ADDRESS	EDLEVEL	COMMENTS
440	REID	ENDICOTT,NY	14	1 YEAR SALES EXPERIENCE
450	JEFFREYS	PHILADELPHIA,PA	12	GOOD CLERICAL BACKGROUND
460	STANLEY	CHICAGO,IL	11	WANTS PART-TIME JOB
470	CASALS	PALO ALTO,CA	14	EXPERIENCED SALESMAN
480	LEEDS	EAST FISHKILL,NY	12	NEEDS INTERVIEW WITH BROWN
490	GASPARD	PARIS,TX	16	WORKED HERE FROM 1/90 TO 6/90

Q.INTERVIEW

This table is for installations that support date/time data. It shows dates and times in ISO format. The format of DATE, TIME, and TIMESTAMP data in your reports depends on the format chosen as your installation's default. It can be modified with the DATE, TIME, and TIMESTAMP edit codes. The columns are:

TEMPID
Temporary identification of the applicant

INTDATE
Date of interview

STARTTIME
Time the interview started

ENDTIME
Time the interview ended

MANAGER
Employee number of the manager who interviewed the applicant

DISP Whether or not the applicant will be hired

LASTNAME
Last name of the applicant

FIRSTNAME
First name of the applicant

TEMPID	INTDATE	STARTTIME	ENDTIME	MANAGER	DISP	LASTNAME	FIRSTNAME
------	----------	---------	--------	-------	------	---------	---------
400	1990-02-05	13.30.00	15.12.00	270	NOHIRE	FROMMHERZ	RICHARD
410	1990-02-11	15.00.00	16.18.00	10	HIRE	JACOBS	SUSAN
420	1990-04-07	09.00.00	09.58.00	140	HIRE	MONTEZ	RITA
430	1990-04-24	10.30.00	11.30.00	290	NOHIRE	RICHOWSKI	JOHN

TEMPID	INTDATE	STARTTIME	ENDTIME	MANAGER	DISP	LASTNAME	FIRSTNAME
440	1990-03-13	10.15.00	11.23.00	160	HIRE	REID	CATHY
450	1990-09-19	09.45.00	11.00.00	50	HIRE	JEFFREYS	PAUL
460	1990-10-06	14.45.00	16.22.00	100	HIRE	STANLEY	JOHN
470	1990-02-05	16.30.00	18.00.00	270	HIRE	CASALS	DAVID
480	1990-03-13	13.30.00	14.45.00	160	NOHIRE	LEEDS	DIANE
490	1990-09-30	15.00.00	15.44.00	140	NOHIRE	GASPARD	PIERRE

Q.ORG

This table provides information on the organization of the company. Each row represents a department. The columns are:

DEPTNUMB
Number of the department (must be unique)

DEPTNAME
Descriptive name of the department

MANAGER
Employee number of the manager of the department

DIVISION
Division to which the department belongs

LOCATION
Name of the city in which the department is located

DEPTNUMB	DEPTNAME	MANAGER	DIVISION	LOCATION
--------	--------------	-------	----------	-------------
10	HEAD OFFICE	160	CORPORATE	NEW YORK
15	NEW ENGLAND	50	EASTERN	BOSTON
20	MID ATLANTIC	10	EASTERN	WASHINGTON
38	SOUTH ATLANTIC	30	EASTERN	ATLANTA
42	GREAT LAKES	100	MIDWEST	CHICAGO
51	PLAINS	140	MIDWEST	DALLAS
66	PACIFIC	270	WESTERN	SAN FRANCISCO
84	MOUNTAIN	290	WESTERN	DENVER

Q.PARTS

This table provides information about parts. The columns are:

SUPPNO
Number of the supplier

PARTNAME
Name of the part

PRODUCT
Product for which the part is needed

PRODNO
Number of the product

PROJNO
Number of the project

SUPPNO	PARTNAME	PRODUCT	PRODNO	PROJNO
1100P	PLASTIC	RELAY	30	1501
1100P	STEEL	WRENCHSET	509	1520
1200S	WIRE	GENERATOR	10	1401
1200S	BEARINGS	MOTOR	50	1402
1300S	COPPER	RELAY	30	1501
1300S	BLADES	SAW	205	1510
1400P	MAGNETS	GENERATOR	10	1409
1400P	VALVES	MOTOR	50	1407
1400P	OIL	GEAR	160	1405

Q.PRODUCTS

This table provides information about a few products and their prices. The columns are:

PRODNUM
Number of the product

PRODNAME
Descriptive name of the product

PRODGRP
General type of product

PRODPRICE
Price of the product

```
PRODNUM         PRODNAME                    PRODGRP                     PRODPRICE
-------         -----------                 ----------                  ---------
10              GENERATOR                   ELECTRICAL                  45.75
505             SCREWDRIVER                 TOOL                        3.70
101             SHAFT                       MECHANICAL                  8.65
20              SWITCH                      ELECTRICAL                  2.60
30              RELAY                       ELECTRICAL                  7.55
40              SOCKET                      ELECTRICAL                  1.40
50              MOTOR                       ELECTRICAL                  35.80
150             CAM                         MECHANICAL                  1.15
160             GEAR                        MECHANICAL                  9.65
190             BUSHING                     MECHANICAL                  5.90
205             SAW                         TOOL                        18.90
330             HAMMER                      TOOL                        9.35
450             CHISEL                      TOOL                        7.75
509             WRENCHSET                   TOOL                        25.90
```

Q.PROJECT

This table provides information about project schedules. The columns are:

PROJNO
Number of the project (must be unique)

PRODNUM
Number of the product

DEPT Number of the department responsible for the project

STARTD
Date the project is to start

ENDD
Date the project is to end

TIMESTAMP
Year, month, day, and time of the report

This table is for installations that support date/time data. It shows dates and times in ISO format. This format is an arbitrary choice. The table you see depends on the choice made by your installation.

PROJNO	PRODNUM	DEPT	STARTD	ENDD	TIMESTAMP
1401	10	20	1996-01-01	1998-03-31	1994-12-18-10.14.44.000001
1402	50	66	1996-01-30	1997-06-30	1994-12-18-10.15.01.999998
1403	150	51	1996-02-02	1999-05-29	1994-12-18-10.22.23.000001
1404	190	38	1997-01-04	1999-06-30	1994-12-18-10.25.43.999999
1405	160	15	1997-04-29	1999-10-30	1995-12-31-14.23.00.999999
1406	20	20	1997-07-11	1998-12-31	1996-01-05-13.31.18.009999
1407	50	42	1997-12-12	2000-06-15	1996-01-05-13.42.27.000000
1408	30	42	1999-03-13	2000-09-30	1996-01-05-13.44.16.999999
1409	10	66	1998-06-15	1999-12-31	1996-03-13-09.12.57.149572
1410	190	10	1998-09-29	2000-03-31	1996-03-13-12.18.23.402917
1501	30	51	1999-01-04	1999-12-31	1996-03-13-12.22.14.201966
1502	150	38	1999-03-01	2000-07-17	1996-03-13-13.17.48.948276

Q.STAFF

This table provides data on the employees. The columns are:

ID Employee serial number (must be unique)

NAME
Name of the employee

DEPT Department number of the employee

JOB Classification of the employee’s job

YEARS
Number of years the employee has worked for the company

SALARY
Employee’s annual salary in dollars and cents

COMM
Employee’s commission in dollars and cents

ID	NAME	DEPT	JOB	YEARS	SALARY	COMM
10	SANDERS	20	MGR	7	18357.50	-
20	PERNAL	20	SALES	8	18171.25	612.45
30	MARENGHI	38	MGR	5	17506.75	-
40	O'BRIEN	38	SALES	6	18006.00	846.55
50	HANES	15	MGR	10	20659.80	-
60	QUIGLEY	38	SALES	-	16808.30	650.25
70	ROTHMAN	15	SALES	7	16502.83	1152.00
80	JAMES	20	CLERK	-	13504.60	128.20
90	KOONITZ	42	SALES	6	18001.75	1386.70
100	PLOTZ	42	MGR	7	18352.80	-

ID	NAME	DEPT	JOB	YEARS	SALARY	COMM
110	NGAN	15	CLERK	5	12508.20	206.60
120	NAUGHTON	38	CLERK	-	12954.75	180.00
130	YAMAGUCHI	42	CLERK	6	10505.90	75.60
140	FRAYE	51	MGR	6	21150.00	-
150	WILLIAMS	51	SALES	6	19456.50	637.65
160	MOLINARE	10	MGR	7	22959.20	-
170	KERMISCH	15	CLERK	4	12258.50	110.10
180	ABRAHAMS	38	CLERK	3	12009.75	236.50
190	SNEIDER	20	CLERK	8	14252.75	126.50
200	SCOUTTEN	42	CLERK	-	11508.60	84.20
210	LU	10	MGR	10	20010.00	-
220	SMITH	51	SALES	7	17654.50	992.80
230	LUNDQUIST	51	CLERK	3	13369.80	189.65
240	DANIELS	10	MGR	5	19260.25	-
250	WHEELER	51	CLERK	6	14460.00	513.30
260	JONES	10	MGR	12	21234.00	-
270	LEA	66	MGR	9	18555.50	-
280	WILSON	66	SALES	9	18674.50	811.50
290	QUILL	84	MGR	10	19818.00	-
300	DAVIS	84	SALES	5	15454.50	806.10
310	GRAHAM	66	SALES	13	21000.00	200.30
320	GONZALES	66	SALES	4	16858.20	844.00
330	BURKE	66	CLERK	1	10988.00	55.50
340	EDWARDS	84	SALES	7	17844.00	1285.00
350	GAFNEY	84	CLERK	5	13030.50	188.00

Q.SUPPLIER

This table provides data on the suppliers of a company. The columns are:

ACCTNO
The account number of the company

COMPANY
The name of the company

STREET
The street address of the company

CITY The city in which the company is located

STATE
The state in which the company is located

ZIP The company's zip code

NOTES
Information about the company

The form for this table specifies a width of 30 and an edit code of CT for the NOTES column.

```
ACCTNO  COMPANY         STREET          CITY            STATE   ZIP     NOTES
------  --------------  --------------  -----------     -----   -----   --------------------
1100P   WESTCO,         1900 115TH      EMERYVILLE      CA      16600   THIS COMPANY HAS A
        INC.            ST.                                             STRONG HISTORY OF
                                                                        ON-TIME DELIVERY.
                                                                        WESTCO IS GROWING
                                                                        QUICKLY.
1200S   MAJOR           4250 BENSON     DALLAS          TX      87050   MAJOR ELECTRICS
        ELECTRICS       ST.                                             DECLARED BANKRUPTCY
                                                                        IN 1987, BUT HAS
                                                                        RECOVERED. FORESEE
                                                                        NO FURTHER
                                                                        PROBLEMS.
1300S   FRANKLIN,       40025           DOVER           DE      99000   DUE TO ITS LOCATION
        INC.            EASTLAND                                        ON EASTERN SEABOARD,
                                                                        FRANKLIN HAS
                                                                        EXCELLENT
                                                                        TRANSPORTATION
                                                                        FACILITIES.
1400P   MOTORWORKS,     19503           JOLIET          IL      12000   PROXIMITY TO CHICAGO
        INC.            BESWICK                                         ENSURES GOOD
                                                                        TRANSPORTATION, BOTH
                                                                        BY RAIL AND TRUCK. A
                                                                        RELIABLE SUPPLIER.
```

Appendix A. Query-by-example

QBE is a language for querying relational data with a graphic representation of the data. You use QBE keywords to retrieve, update, delete, and insert data. You also use them to control the presentation of report data. To learn more about Query-by-Example, follow the exercises in this appendix.

Displaying the QBE query panel

Before you can write a query in QBE, you need to display the QBE Query panel. There are two ways to do this from the command line on the QMF Home panel. The method you choose depends on whether you plan to use QBE most of the time or switch back and forth between query languages.

1. If you plan to write queries in QBE most of the time, enter:

   ```
   SET PROFILE (LANGUAGE=QBE
   RESET QUERY
   SAVE PROFILE
   ```

2. If you prefer to have another language set in your profile, you can specify QBE for your current session with the command:

   ```
   RESET QUERY (LANGUAGE=QBE
   ```

Running and saving queries

When you finish composing your query, you will want to run it and possibly save it.

To run a query, press the Run function key or enter the command:

```
RUN QUERY
```

To save a query, choose a name (for example, MYQUERY), and enter the command:

```
SAVE QUERY AS MYQUERY
```

When you enter a RUN or SAVE command from a query panel, you do not need to specify the object type as QUERY. The object type defaults to QUERY when these commands are entered from a query panel.

Listing queries

You can also list all your saved queries:

```
LIST QUERIES (OWNER userid
```

If you want additional information about any command, enter the command name and follow it with a question mark. For example:

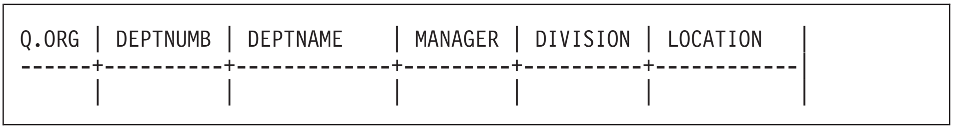
```
LIST ?
```

A few other QMF commands are described in “QMF commands specific to QBE” on page 312.

Drawing example tables

In QBE, you create queries in an example table. An example table is a framework in which you enter instructions about how you want data presented in your report. (If you have the appropriate authorization, you can also use keywords (D. I. and U.) in an example table to make changes to a database.) For example, from a RESET QBE QUERY panel, the Q.ORG example table displays when you issue the command:

```
DRAW Q.ORG
```



```
Q.ORG | DEPTNUMB | DEPTNAME    | MANAGER | DIVISION | LOCATION   |
------+----------+-------------+---------+----------+------------|
      |          |             |         |          |            |
```

Within this framework, you can select the columns you want presented with `P.`, and use other QBE keywords to control the presentation of the report data and make changes to the database.

The program function keys shown at the bottom of the screen make it easier to perform certain functions. Your installation may have changed the setting of the function keys. This book uses the initial settings:

1 Shows help information about your last action.

2 Runs your query.

3 Returns to the QMF Home panel.

4 Makes the object larger. See 316.

5 Makes the object smaller. See 318.

6 Draws an empty example table.

7 Moves the display backward.

8 Moves the display forward.

9 Shows the form panel last used.

10 Scrolls the display to the left.

11 Scrolls the display to the right.

12 Displays your report.

Presenting all the columns of a table

To retrieve data from a table in the database and display it in a report, use the `P.` keyword. You can use the `D.`, `I`, and `U.` keywords in a similar way to delete, insert, and update data in the database.

To display the data in all the columns of a table, put `P.` under the table name and do not remove any of the column headings, as in this example table:

```
Q.ORG | DEPTNUMB | DEPTNAME | MANAGER | DIVISION | LOCATION |
------+----------+----------+---------+----------+----------|
P.    |          |          |         |          |          |
```

With this query on your screen, enter RUN QUERY on the command line (or press the Run function key) to produce the following report.

```
DEPTNUMB  DEPTNAME        MANAGER  DIVISION    LOCATION
--------  --------------  -------  ----------  -------------
      10  HEAD OFFICE         160  CORPORATE   NEW YORK
      15  NEW ENGLAND          50  EASTERN     BOSTON
      20  MID ATLANTIC         10  EASTERN     WASHINGTON
      38  SOUTH ATLANTIC       30  EASTERN     ATLANTA
      42  GREAT LAKES         100  MIDWEST     CHICAGO
      51  PLAINS              140  MIDWEST     DALLAS
      66  PACIFIC             270  WESTERN     SAN FRANCISCO
      84  MOUNTAIN            290  WESTERN     DENVER
```

Presenting certain columns of a table

To see data from only selected columns of an example table, put P. under the names of the columns you want to see.

When you run this query:

```
Q.ORG | DEPTNUMB | DEPTNAME | MANAGER | DIVISION | LOCATION |
------+----------+----------+---------+----------+----------|
      | P.       | P.       |         |          |          |
```

QMF produces this report:

```
DEPTNUMB  DEPTNAME
--------  --------------
      84  MOUNTAIN
      66  PACIFIC
```

```
10  HEAD OFFICE
15  NEW ENGLAND
20  MID ATLANTIC
38  SOUTH ATLANTIC
42  GREAT LAKES
51  PLAINS
```

Changing the order of columns

The columns are, by default, displayed in the same order they are in the sample table. (See "Q.ORG" on page 287.) To change the order of the columns displayed, type over the names of the columns in the example table.

The following example reverses the names DIVISION and LOCATION. You reverse the names by typing LOCATION over DIVISION and vice versa.

```
Q.ORG | DEPTNUMB | DEPTNAME | MANAGER | LOCATION | DIVISION |
------+----------+----------+---------+----------+----------|
      | P.       |          |         | P.       | P.       |
```

To display a column more than once, type the name of the column a second time over the name of some unused column. Or, use the Enlarge function key to add a column to the example table. Then, type the name of the column you want to display in the new column. Put P. under the column name. (See "ENLARGE command" on page 316.)

Presenting certain rows of a table

There are many ways to choose which rows of a table you want to present.

Presenting rows that contain a certain value

To display only those rows of a table that have a certain value in some column, put the value under the column in the example table. That value is then a **condition**. The query selects just those rows of the table that contain that value in the indicated column.

You can, for example, display all the column names that are shown in the example table, and select only the rows with 5 in the YEARS column.

When you run this query:

```
Q.STAFF| DEPT | NAME |  JOB  |  YEARS  |
-------+------+------+-------+---------|
P.     |      |      |       | 5       |
```

QMF produces this report:

```
DEPT  NAME        JOB     YEARS
----  ----------  -----  ------
  38  MARENGHI    MGR         5
  15  NGAN        CLERK       5
  10  DANIELS     MGR         5
  84  DAVIS       SALES       5
  84  GAFNEY      CLERK       5
```

You can display only the columns DEPT, NAME, and JOB and select only the rows with 20 in the DEPT column. (You could get the report without the DEPT column by not including the P. in that column of the example table.)

When you run this query:

```
Q.STAFF | ID | DEPT    | NAME | JOB | YEARS | SALARY | COMM |
--------+----+---------+------+-----+-------+--------+------|
        |    | P. 20   | P.   | P.  |       |        |      |
```

QMF produces this report:

```
DEPT  NAME        JOB
----  ----------  -----
  20  SANDERS     MGR
  20  PERNAL      SALES
  20  JAMES       CLERK
  20  SNEIDER     CLERK
```

Defining example elements

An example element is a symbol that is used to represent data in a column. It must appear in a named column before it can be used with a column function (AVG., COUNT., MAX., MIN., SUM.) in an unnamed column.

In this book, an example element is usually similar to the name of the column to which it refers. For instance, an example element in the SALARY column might be _S, _SAL, or _SALARY. Similarity is not necessary, however. Someone accustomed to writing algebraic expressions might want _X and _Y as example elements.

For example, the following query defines _S as "any salary". Then, in the unnamed column, it calculates and selects the average of all the salaries in the Q.STAFF table. ("ENLARGE command" on page 316 shows how to add an unnamed column to your example table.)

```
Q.STAFF | ID | NAME | DEPT | JOB | YEARS | SALARY |             |
--------+----+------+------+-----+-------+--------+------------|
        |    |      |      |     |       | _S     | P. AVG. _S |
```

If you use an example element, put it in your query at least twice. Put it in once to define it in the example table, and one or more times in writing conditions or calculations, either in the example table or in a conditions box.

Rules for example elements

An example element must start with an underscore character (_). Any string of letters and digits up to 17 characters may follow.

Writing expressions

You can write expressions in conditions by using any of the following symbols:

Condition	**QBE Keyword**
Equal	=
Not equal	¬=
Greater than	>
Greater than or equal	>=
Less than	<
Less than or equal	<=
Multiple conditions	AND, OR
Values within a range	BETWEEN
Values from a list	IN (x, y, z)
A certain string of characters	LIKE '%abc%'
Ignore certain characters	LIKE '_abc_'

Negative conditions
NOT

See Appendix A, “Query-by-example,” on page 293 for descriptions of these keywords.

Note: The QBE language does not recognize the following:

- Concatenation (||) operator
- Not-greater-than (¬>)
- Not-less-than (¬<)
- Not-equal-to (<>

If you use one of these operators, QMF displays an error message.

Order of evaluation:

1. Built-in column functions
2. A plus sign or minus sign before a single value
3. Multiplication or division of two values
4. Addition or subtraction of two values

QMF evaluates operations at the same level of precedence from left to right.

You can change the order of evaluation with parentheses just as you would use them in a mathematical formula. For example, the following two expressions are equivalent:

```
A * - B / C + D / E       ((A*(-B))/C) + (D/E)
```

When you create a table, each column in it holds a certain type of data. QMF performs arithmetic operations only on numeric data types.

Rules for quotation marks

Do not enclose **numeric data** in quotation marks.

You need to enclose **character data** that is used in conditions in quotation marks only when:

- The data contains blanks (as in `'ROOM 27'`), or any characters besides digits, letters, #, $, or @, (as in `'P.D.Q.'`, `'BOW-WOW'`).
- The data contains a single quotation mark or apostrophe. (In this case you have to double the quotation mark inside the data, as in `'O''BRIEN'`).
- To distinguish the constants 'NULL' and 'USER' from the keywords NULL and USER.
- The data contains all double-byte characters.
- Character data that is entirely digits, as in '849276552'.

- The data type is DATE, TIME, or TIMESTAMP.

Do NOT enclose values to compare with columns of numeric data in quotation marks.

Arithmetic overflow

When an operation in a query would produce a result outside an allowable range, the situation is called “arithmetic overflow”. It is possible for the result of an arithmetic operation to be outside the range allowed for the data type of the result. For example, 1000000 is an allowable value in a column with data type INTEGER, but 1000000 * 1000000 cannot have the data type INTEGER. Also, division of any number by 0 produces an overflow.

Using unnamed columns in an example table

In previous examples, the named columns in the example table adequately represented the report you wanted to create. But for more complex queries, you need to add new “unnamed” columns or use target tables (see “Adding a target table” on page 304).

To add an empty column to your query, put your cursor beside the column name to the left of where you want to add a column. Then, press the Enlarge function key. You can also blank out an unwanted column name to create a new (unnamed) column.

You can add a column of descriptive information to your report by putting a constant in an example table in an added (unnamed) column. The following example lists the names and addresses of the people listed in the Q.APPLICANT table with 14 years of education, and identifies each with the **character constant** APPLICANT.

When you run this query:

```
Q.APPLICANT | NAME   | ADDRESS | EDLEVEL |              |
------------+-------+---------+---------+--------------|
            | P.AO. | P.      | 14      | P. APPLICANT |
```

QMF produces this report:

```
NAME        ADDRESS           EXPRESSION 1
----------  ----------------  ------------
CASALS      PALO ALTO,CA      APPLICANT
REID        ENDICOTT,NY       APPLICANT
RICHOWSKI   TUCSON,AZ         APPLICANT
```

You could also use a **numeric constant**. A constant can be up to 254 characters in length and, in addition to alphabetic and numeric characters, it can contain #, $, and @.

Use example elements to refer to the columns in an example table that are the source of the data for the expression in an unnamed column. For example, this query uses _S to refer to the values in the SALARY column and _C to refer to the values in the COMM column.

When you run this query:

```
Q.STAFF | ID | DEPT |           | SALARY | COMM |
--------+----+------+-----------+--------+------|
        | P. |  20  | P._S + _C | _S     | _C   |
```

QMF produces this report:

```
     ID   EXPRESSION 1
-------   ------------
     10              -
     20       18783.70
     80       13632.80
    190       14379.25
```

By using _S and _C, you can then create an expression from the values in two columns and put the sum of the two into the report, via the unnamed column.

There is no restriction on the location of the unnamed column. Like other data columns, however, the unnamed column must be to the right of the table-name column.

Example 1:

List yearly, monthly, and weekly salaries.

```
Q.STAFF | ID | NAME | SALARY |       |       |
--------+----+------+--------+-------+-------|
P.      |    |      | _S     | _S/12 | _S/52 |
```

Example 2:

List the IDs, the commission, and the sum of the salary and commission. Show the percentage of total earnings the commission represents, and list in descending order (DO.) by the percentage.

```
Q.STAFF | ID | SALARY | COMM |           |                       |
--------+----+--------+------+-----------+-----------------------|
        | P. | _S     | P._C | P._S + _C | P.100*_C/(_S+_C) DO.  |
```

Adding conditions to the example table

You can write expressions in your example table that set conditions on which rows are selected. The query below selects only the rows where the commission is greater than or equal to 1000.

When you run this query:

```
Q.STAFF | ID | NAME | DEPT | JOB | YEARS | SALARY | COMM         |
--------+----+------+------+-----+-------+--------+--------------|
        | P. |      |      |     |       |        | >= 1000  P.  |
```

QMF produces this report:

```
   ID     COMM
-----  -------
   70  1152.00
   90  1386.70
  340  1285.00
```

Data types in conditions

If a column contains letters or special characters, it must have a character data type. (If it contains double-byte characters, it can have a graphic data type.)

If a column contains only numbers or mostly numbers, it might still have a character data type. For example, a column of part numbers might contain mostly digits. However, if a part number like “1390X” is in the column, the column must have a character data type.

Adding a CONDITIONS box

You can express simple conditions in an example table. However, more complicated conditions require the use of example elements and a CONDITIONS box. You can also specify expressions in an example table, as is shown in “Adding conditions to the example table.” However, it is generally more convenient to define example elements in the example table and specify the expressions in a CONDITIONS box.

To add a CONDITIONS box to your query, enter this command:

```
COMMAND===> DRAW COND
```

Note: If you enter DRAW CONDITION or DRAW CONDITIONS instead, an example table of that name rather than a CONDITIONS box displays.

Use a CONDITIONS box to do any of the following:

- Refer to two or more columns in the condition. For example:
```
_S + _C > 20000
```
- Use a column function in the condition. For example:
```
AVG. _S > 20000
```
- Refer to a column in the example table more than once. For example:
```
_SAL > 10000 AND _SAL > _COMM
```
- Use the AND or OR operator in a condition requiring example elements. For example:
```
_Y=10 OR _S>2000
```
- Use parentheses in a complex condition to change the order of precedence. For example:
```
(_SAL > 20000 OR _COMM < 2000) AND DEPT = 84
```
- Avoid widening a column of the example table to hold a long condition.

The CONDITIONS box in the following query uses the example elements (_S and _C) defined in the example table to select rows where salary plus commission (_S + _C) is greater than $20,000.

When you run this query:

```
Q.STAFF | NAME | SALARY | COMM |
--------+------+--------+------|
P.      |AO.   | _S     | _C   |

|        CONDITIONS            |
|------------------------------|
| _S + _C > 20000              |
```

QMF produces this report:

```
NAME          SALARY      COMM
----------  --------  --------
GRAHAM      21000.00    200.30
WILLIAMS    19456.50    637.65
```

QMF presents the names in ascending order (AO.). (The result does not include anyone whose salary alone is greater than $20,000 if the commission is null.)

This query selects anyone whose weekly salary is less than $300.

```
Q.STAFF | ID | NAME | DEPT | JOB | YEARS | SALARY | COMM |
--------+----+------+------+-----+-------+--------+------|
P.      |    |      |      |     |       | _SAL   |      |

|       CONDITIONS            |
|-----------------------------|
| _SAL/52 < 300               |
```

This query selects anyone whose commission is 5% or more of total earnings.

```
Q.STAFF | ID | NAME | DEPT | JOB | YEARS | SALARY | COMM |
--------+----+------+------+-----+-------+--------+------|
P.      |    |      |      |     |       | _S     | _C   |

|       CONDITIONS            |
|-----------------------------|
| _C >= .05 * (_S+_C)         |
```

You can use more than one CONDITIONS box or more than one condition in any box. However, you must place each condition on a single row in a box.

Multiple conditions in a query are implicitly connected by "and". That is, in the example below, the AND keyword is assumed between the two conditions `_Y = 10 OR _S > 20000` and `_C >= 1000`. QMF evaluates the OR condition (`_Y = 10 OR _S > 20000`) before it connects and evaluates the two conditions. (See "Order of evaluation:" on page 299 for information on determining processing order.)

```
Q.STAFF | ID | NAME | DEPT | JOB | YEARS | SALARY   | COMM |
--------+----+------+------+-----+-------+----------+------|
P.      |    |      |      |     | _Y    | _S       | _C   |

|       CONDITIONS            |
|-----------------------------|
| _Y = 10 OR _S > 20000       |

|       CONDITIONS            |
|-----------------------------|
| _C <= 1000                  |
```

Adding a target table

An alternative to adding a new unnamed column to your example table is to use a **target table**. A target table is an empty example table that uses example elements to refer to other example tables. Anything that you can use in an unnamed column added to an example table, you can use in a target table.

To use a target table to combine information from two columns, display your table and issue the DRAW command:

```
COMMAND===> DRAW
```

```
Q.STAFF | ID | DEPT | SALARY | COMM |
--------+----+------+--------+------|
        | _I |  20  | _S     | _C   |


         |         |         |         |         |
---------+---------+---------+---------+---------|
 P.      | _I      | _S + _C |         |         |
```

Restrictions

You cannot use target tables (or unnamed columns in example tables) to do the following:

- Name the column for your report. QMF names columns that are created by expressions (like the one in the example above). You can change the name of a column in a report by using a form. For more information on using forms, see Chapter 6, “Customizing your reports,” on page 113.
- Write a condition. (If you need to write a condition, write it in a named column or a CONDITIONS box. See “Adding a CONDITIONS box” on page 302.
- Define an example element. You must define example elements in a named column of the example table.

Eliminating duplicate rows

QMF displays all rows, including duplicate rows, by default if you have just one P. row in your query. To eliminate duplicate rows, specify UNQ. (unique) under the table name in the row with the P. operator.

Both of the following examples have P. in the DIVISION column. The report in Example 1 presents all the rows, including duplicates.

Example 1:

Without UNQ.

```
Q.ORG | DEPTNUMB | DEPTNAME | MANAGER | DIVISION | LOCATION |
------+----------+----------+---------+----------+----------|
      |          |          |         | P.       |          |
```

QMF produces this report:

```
DIVISION
--------
CORPORATE
EASTERN
EASTERN
EASTERN
MIDWEST
MIDWEST
WESTERN
WESTERN
```

Example 2 specifies `UNQ.` under the table name. Therefore, QMF eliminates all columns that contain duplicate data in the presented column.

Example 2:

With `UNQ.`

```
Q.ORG | DEPTNUMB | DEPTNAME | MANAGER | DIVISION | LOCATION |
------+----------+----------+---------+----------+----------|
UNQ.  |          |          |         | P.       |          |
```

QMF produces this report:

```
DIVISION
--------
CORPORATE
EASTERN
MIDWEST
WESTERN
```

If your example table has two or more `P.` rows, QMF **does not** display duplicate rows. (See "ALL. — Display duplicate rows" on page 321 and "UNQ. — Eliminate duplicate rows" on page 344.)

Presenting data from more than one table

Sometimes you need information from two different tables. You can accomplish this only if there is a link between the two tables. That is, a column in each table contains identical information. For example, both Q.STAFF and Q.ORG have a column that contains employee numbers. In

Q.STAFF this column is ID; in Q.ORG it is MANAGER. With this link, you can combine information from both tables into one report by using the following process:

1. In QMF, enter `RESET QUERY` to display an empty QBE Query panel.
2. Enter `DRAW Q.STAFF`.

```
Q.STAFF | ID | NAME   | DEPT   | JOB    | YEARS   | SALARY   | COMM   |
--------+----+-------+-------+-------+--------+---------+-------|
        |    |       |       |       |        |         |       |
```

3. This query uses only the table name and the first two columns, so we can delete the other columns. (See "REDUCE command" on page 318.)

```
Q.STAFF | ID     | NAME       |
--------+--------+-----------|
        |        |           |
```

4. Place the cursor on the command line and enter `DRAW Q.ORG`.

```
Q.STAFF | ID     | NAME       |
--------+--------+-----------|
        |        |           |

Q.ORG | DEPTNUMB | DEPTNAME   | MANAGER | DIVISION    | LOCATION |
------+----------+-----------+---------+------------+----------|
      |          |           |         |            |          |
```

5. Delete the `DIVISION` and `LOCATION` columns from the Q.ORG table.

```
Q.STAFF | ID     | NAME       |
--------+--------+-----------|
        |        |           |

Q.ORG | DEPTNUMB | DEPTNAME   | MANAGER |
------+----------+-----------+---------|
      |          |           |         |
```

6. Add an unnamed column to the Q.ORG example table and increase its size. (See "ENLARGE command" on page 316.)

```
Q.STAFF | ID     | NAME       |
--------+--------+-----------|
        |        |           |

Q.ORG | DEPTNUMB | DEPTNAME   | MANAGER |            |
------+----------+-----------+---------+------------|
      |          |           |         |            |
```

7. Now add some example elements.

```
Q.STAFF | ID     | NAME       |
--------+--------+-----------|
        | _ID    | _NM        |

Q.ORG | DEPTNUMB | DEPTNAME   | MANAGER |            |
------+----------+-----------+---------+------------|
P.    |          |           | _ID     | _NM        |
```

The same example element (in this case _ID) must be used in two example tables to select only the rows where MANAGER (the manager ID) in Q.ORG is equal to ID in Q.STAFF.

P. can appear in only one table. The example element _NM is added to the unnamed column of the Q.ORG example table so that it is presented from the Q.STAFF table even though no P. appears in the Q.STAFF example table.

This query says the following: Show columns DEPTNUMB, DEPTNAME, and MANAGER from Q.ORG and the NAME column from Q.STAFF. Display the rows where the data in the MANAGER column in Q.ORG is the same as the data in the ID column in Q.STAFF.

Press the Run function key to get this report:

```
DEPTNUMB  DEPTNAME        MANAGER  NAME
--------  --------------  -------  ---------
      20  MID ATLANTIC         10  SANDERS
      38  SOUTH ATLANTIC       30  MARENGHI
      15  NEW ENGLAND          50  HANES
      42  GREAT LAKES         100  PLOTZ
      51  PLAINS              140  FRAYE
      10  HEAD OFFICE         160  MOLINARE
      66  PACIFIC             270  LEA
      84  MOUNTAIN            290  QUILL
```

See also “P. — Present data in a table” on page 339.

Writing queries to be shared

To make it possible to share a query with another user, use any or all of the following methods:

- Model query
- Substitution variables
- The USER variable

Model query

A model is a copy of a query that allows you or other users to produce different reports by specifying different conditions in a copy of the model.

Suppose, for example, that you are the sales manager of Department 38, and you have written a query that lists the name, job, and commission for everyone in your department.

```
Q.STAFF | NAME      | DEPT   | JOB   | COMM       |
--------+-----------+--------+-------+------------|
        | P. AO.    | 38     | P.    | P.         |
```

Other sales managers can get a report for their departments by using your model query. They can display, change, and run the query, or change it and run it later.

Substitution variables

Another way to use a model is to set substitution variables for the values that you want to change.

A substitution variable can represent anything that you can write into a query, such as a column name, a search condition, or a specific value. You supply the value for a substitution variable in the “&variable” option on the RUN command or the RUN Command Prompt panel. You can specify the substitution variable on a SET GLOBAL command (instead of RUN) before you run the query.

For example, if you want a list of the employee IDs, names, and jobs of everyone in each of several different departments, you could construct the query like this:

```
Q.STAFF | ID | NAME   | DEPT        | JOB  |
--------+----+--------+-------------+------|
        | P. | P. AO. | &DEPARTMENT | P.   |
```

If you run this query without a value on your RUN command, a Prompt panel displays. On the Prompt panel, you fill in a value to substitute for the variable in the query.

When the value to substitute for the variable is one of the following:

A single valid numeric value
Specify the desired value.

Text without embedded quotes, parentheses, blanks, equal signs, or commas
Specify exactly as desired.

Text with embedded quotes
Enclose the entire value in quotes. (Quotation marks are not removed when QMF makes the substitution.)

Text with embedded parentheses, blanks, equal signs, or commas
Enclose the entire value in parentheses. (The outer parentheses are removed when QMF makes the substitution.)

You could write the following query, for example:

```
Q.STAFF | NAME      | DEPT   | JOB   | COMM       |
--------+-----------+--------+-------+------------|
        | P. AO.    | &DEPT  | P.    | P.         |
```

When you run this query, you can specify the variable value:

```
RUN QUERY (&DEPT = 38
```

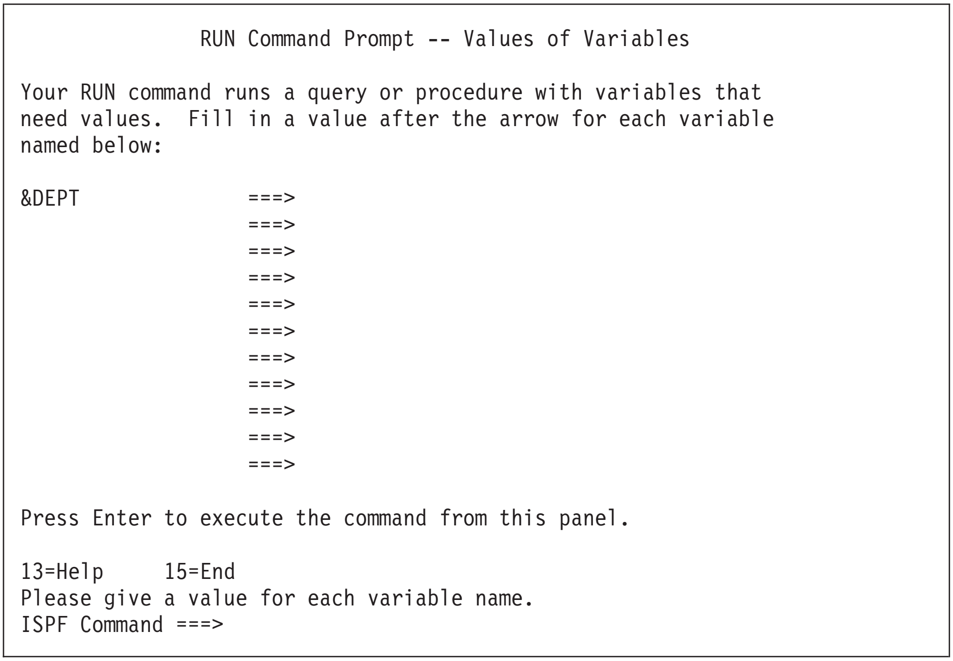
QMF then reads the query like this:

```
Q.STAFF | NAME      | DEPT   | JOB   | COMM       |
--------+-----------+--------+-------+------------|
        | P. AO.    | 38     | P.    | P.         |
```

Substitution variables make it possible for other people to use your query. Other users can substitute any value in place of the variable and produce a report specific to their needs. For example, if your RUN command does not supply values for the variable as is shown in the following command:

```
COMMAND===> RUN REPT4QRY
```

QMF displays a prompt panel:



```
               RUN Command Prompt -- Values of Variables

Your RUN command runs a query or procedure with variables that
need values.  Fill in a value after the arrow for each variable
named below:

&DEPT              ===>
                   ===>
                   ===>
                   ===>
                   ===>
                   ===>
                   ===>
                   ===>
                   ===>
                   ===>
                   ===>

Press Enter to execute the command from this panel.

13=Help     15=End
Please give a value for each variable name.
ISPF Command ===>
```

Enter the desired department number after the arrow on the panel. For example:

```
&DEPT    ===> 84
```

A substitution variable can be a whole name or be included in part of a name.

Substitution variable names:

- Can be no longer than 18 characters, and the first character must be an ampersand (&).
- Can contain only these characters:
  – Letters of the alphabet
  – National characters: @ # $
  – Special characters: ¦ % ? ~ ` { } \ | &cent !
  – Numbers
  – Underscores (_)
- Can be separated from another variable or command word by any of the characters that are not mentioned above, such as commas, blanks, or parentheses.

The USER variable

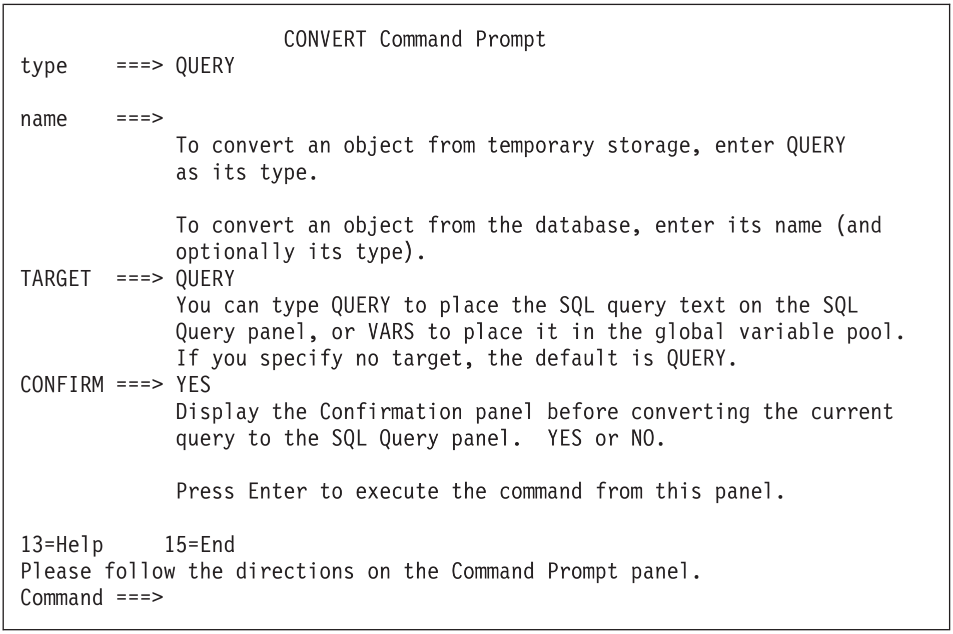
Another method of sharing a query is to create a query with USER under the NAME column (or any column that contains user identification (*user ID*) numbers. You can then share the query with other users, who can run it without change, because their *user ID* is substituted for the word USER as a condition in the query. (See “USER — Present rows with a value equal to a user identification” on page 345.)

QMF commands specific to QBE

The following QMF commands are either unique to QBE or function differently with QBE queries than with SQL queries.

CONVERT command

The CONVERT command converts a QBE query to an SQL query. If you specify `CONVERT ?`, the following Prompt panel displays. You can complete the command on the Prompt panel.



```
                              CONVERT Command Prompt
type    ===> QUERY

name    ===>
                To convert an object from temporary storage, enter QUERY
                as its type.

                To convert an object from the database, enter its name (and
                optionally its type).
TARGET  ===> QUERY
                You can type QUERY to place the SQL query text on the SQL
                Query panel, or VARS to place it in the global variable pool.
                If you specify no target, the default is QUERY.
CONFIRM ===> YES
                Display the Confirmation panel before converting the current
                query to the SQL Query panel.  YES or NO.

                Press Enter to execute the command from this panel.

13=Help     15=End
Please follow the directions on the Command Prompt panel.
Command ===>
```

If your query contains substitution variables, and you do not provide values for them on your CONVERT command, a Prompt panel displays. You can use this panel to fill in the values for the variables. For example, assume that you write the following query, and save it as THISONE.

```
Q.STAFF | NAME      | DEPT   | JOB   | COMM       |
--------+-----------+--------+-------+------------|
        | P. AO.    | &DEPT  | P.    | P.        |
```

Now, suppose that you want to convert it to SQL, but specify only:

```
CONVERT THISONE
```

The following Prompt panel displays:

```
             CONVERT Command Prompt -- Values of Variables

Your CONVERT command converts a query with variables that need values.
Fill in a value after the arrow for each variable named below:


&DEPT              ===>
                   ===>
                   ===>
                   ===>
                   ===>
                   ===>
                   ===>
                   ===>
                   ===>
                   ===>

Then press Enter to execute the command from this panel.



13=Help     15=End
Please give a value for each variable name.
Command ===>
```

When you fill in the Department number (84), the following SQL query displays:

```
SELECT "NAME", "JOB", "COMM"
FROM "Q"."STAFF"
WHERE ("DEPT" = 84)
ORDER BY 0000001
```

The CONVERT command does not operate on a query at a remote location.

DELETE command

The DELETE command removes the following:

- An example table from a QBE query
- A COMMENTS box from a QBE query
- A CONDITIONS box from a QBE query
- Error messages on Query panel

To delete one of the above items, follow these steps:

1. Type DELETE on the command line. Do not press Enter yet.
2. Move the cursor to a position anywhere within any of the items that are listed above.
3. Press Enter. QMF deletes the item.

Note: The keyword D. is different from the DELETE command. See "D. — Delete rows from a table" on page 326 for more information.

DRAW command

The DRAW command creates an example table or adds a `COMMENTS` box, `CONDITIONS` box, or target table to a QBE query.

If you specify the DRAW command by itself (or press the Draw function key), an empty table appears. DRAW has the following forms:

- `COMMAND ===> DRAW`
- `COMMAND ===> DRAW name`
- `COMMAND ==> DRAW COMM`
- `COMMAND ==> DRAW COND`

DRAW

Draws an empty target table.

```
           |           |           |           |           |
-----------+-----------+-----------+-----------+-----------|
           |           |           |           |           |
```

DRAW name

Draws an example table with the name of the table or view in the first column.

If name specifies an existing table or view, QMF draws an example of that table or view. The example table has the same number of columns with the same column names as the table or view name has. The width of each column in the example table depends on the data type for each column.

For example, DRAW Q.STAFF produces this example table:

```
Q.STAFF | ID | NAME | DEPT | JOB | YEARS | SALARY | COMM |
--------+----+------+------+-----+-------+--------+------|
        |    |      |      |     |       |        |      |
```

If name is qualified with an owner and a location, and if your database supports 3-part names, QMF draws an example table with the fully-qualified name in the table-name column. For example:

```
 VENICE.Q.STAFF  |           |           |           |           |
-----------------+-----------+-----------+-----------+-----------|
                 |           |           |           |           |
```

If name specifies a nonexistent table, QMF draws an example table with the name specified appearing in the table-name column. For example, if there is no table in the database that is named EMPTYBOX, DRAW EMPTYBOX produces this example table:

```
EMPTYBOX    |           |           |           |           |
------------+-----------+-----------+-----------+-----------|
            |           |           |           |           |
```

DRAW COMM

Adds an empty COMMENTS box:

```
|         COMMENTS         |
|--------------------------|
|                          |
```

DRAW COND

Adds an empty CONDITIONS box:

```
|        CONDITIONS        |
|--------------------------|
|                          |
```

ENLARGE command

The ENLARGE command increases the size of an example table, COMMENTS box, or CONDITIONS box. The maximum allowable table width depends on the number of columns that are selected and the size of the column names. Longer names use more space. You can select up to 300 columns.

To enlarge without a function key:

1. Type ENLARGE on the command line.
2. Position the cursor as is shown by one of the diagrams on this page.
3. Press Enter.

To enlarge with a function key, position the cursor in the area you want to change and press the Enlarge function key. The following diagrams show this method. An asterisk indicates the placement of the cursor. (*).

Example 1: Add a column to the right of the **table-name column**. Put the cursor above the line, on the vertical bar, and press the Enlarge function key.

```
Before:                          After:

TNAME * COL1 | COL2 |            TNAME |      | COL1 | COL2 |
------+------+------|            ------+------+------+------|
      |      |      |                  |      |      |      |
```

Example 2: Add a column to the right of any other column. Put the cursor above the line, within the left-hand adjacent column, and press the Enlarge function key.

```
Before:                          After:

TNAME | COL1*| COL2 |            TNAME | COL1 |      | COL2 |
------+------+------|            ------+------+------+------|
      |      |      |                  |      |      |      |
```

Example 3: Widen the table-name column. Put the cursor in that column, above the line, and press the Enlarge function key.

```
Before:                          After:

TNAME*| COL1 | COL2 | COL3 |     TNAME      | COL1 | COL2 | COL3 |
------+------+------+------|     -----------+------+------+------|
      |      |      |      |                |      |      |      |
```

Example 4: Widen any other column. Put the cursor in that column, on or below the line, and press the Enlarge function key.

```
Before:                             After:

TNAME | COL1 | COL2 |               TNAME | COL1 | COL2        |
------+------+------|               ------+------+------------|
      |      |  *   |                     |      |            |
```

Example 5: Add a row below any row. Put the cursor below the line, under the table name, and press the Enlarge function key.

```
Before:                             After:

TNAME | COL1 | COL2 | COL3 |        TNAME | COL1 | COL2 | COL3 |
------+------+------+------|        ------+------+------+------|
P. *  | 10   |      |      |        P.    | 10   |      |      |
P.    |      | J48  |      |              |      |      |      |
                                    P.    |      | J48  |      |
```

Example 6: Add a new first row. Put the cursor on the line, under the table name, and press the Enlarge function key.

```
Before:                             After:

TNAME | COL1 | COL2 | COL3 |        TNAME | COL1 | COL2 | COL3 |
-----* ------+------+------|        ------+------+------+------|
P.    | 10   |      |      |              |      |      |      |
P.    |      | J48  |      |        P.    | 10   |      |      |
                                    P.    |      | J48  |      |
```

Example 7: Widen a `CONDITIONS` or `COMMENTS` box. Put the cursor below the line, inside the box, and press the Enlarge function key.

```
Before:                              After:

|     CONDITIONS     |               |     CONDITIONS             |
|--------------------|               |----------------------------|
|          *         |               |                            |
```

Example 8: Add a new row to a CONDITIONS or COMMENTS box. Put the cursor below the line, on the vertical bar at the left, and press the Enlarge function key.

```
Before:                              After:

|     CONDITIONS     |               |     CONDITIONS     |
|--------------------|               |--------------------|
* _COL1   100        |               | _COL1   100        |
| _COL3/12   90      |               |                    |
                                     | _COL3/12   90      |
```

REDUCE command

The REDUCE command reduces the size of an example table, COMMENTS box, or CONDITIONS box.

To reduce without a function key:

1. Type REDUCE on the command line.
2. Position the cursor as is shown by one of the diagrams on this page.
3. Press Enter.

To reduce with a function key, position the cursor in the area you want to change and press the Reduce function key. The following diagrams show this method. An asterisk indicates the position of the cursor (*).

Example 1: Remove a column. Put the cursor above the line, within the column, and press the Reduce function key.

```
Before:                               After:

TNAME | COL1 *| COL2 | COL3 |         TNAME | COL2 | COL3 |
------+-------+------+------|         ------+------+------|
      |       |      |      |               |      |      |
```

Example 2: Narrow the table-name column. Put the cursor above the line, in that column, and press the Reduce function key.

```
Before:                             After:

TNAME  * | COL1 | COL2 | COL3 |     TNAME | COL1 | COL2 | COL3 |
---------+------+------+------|     ------+------+------+------|
         |      |      |      |           |      |      |      |
```

Example 3: Narrow any other column. Put the cursor on or below the line, in that column, and press the Reduce function key.

```
Before:                              After:

TNAME | COL1 | COL2       | COL3 |   TNAME | COL1 | COL2 | COL3 |
------+------+-----------+------|    ------+------+------+------|
      |      |        *  |      |          |      |      |      |
```

Example 4: Remove a row. Put the cursor under the table name, in the row to be removed, and press the Reduce function key.

```
Before:                              After:

TNAME | COL1 | COL2 | COL3 |         TNAME | COL1 | COL2 | COL3 |
------+------+------+------|         ------+------+------+------|
P.    |   10 |      |      |         P.    |   10 |      |      |
P.  * |      | J48  |      |
```

Example 5: Narrow a CONDITIONS or COMMENTS box. Put the cursor below the line, inside the box, and press the Reduce function key.

```
Before:                              After:

|             CONDITIONS           |  |    CONDITIONS     |
|----------------------------------|  |-------------------|
|                  *               |  |                   |
```

Example 6: Remove a row from a CONDITIONS or COMMENTS box. Put the cursor below the line, on the vertical bar at the left, and press the Reduce function key.

```
Before:                                      After:

|      CONDITIONS      |                     |     CONDITIONS      |
|----------------------|                     |---------------------|
*  _COL1 > 100         |                     | _COL3/12 < 90       |
|  _COL3/12 < 90       |
```

Keyword reference

Keyword	Action	Page
ALL.	Display duplicate rows	321
AND	Present on two conditions	321
AO., AO(n).	Sort rows in ascending order	322
AVG.	Calculate the average value	323
BETWEEN x AND y	Present values within a range	324
COUNT.	Count the number of values in a column	326
D.	Delete a row from a table	326
DO., DO(n).	Sort rows in descending order	327
G.	Grouping	329
I.	Insert a row into a table	330
IN (x, y, z)	Present certain values in a list	331
LIKE	Present on part of a value	332
MAX.	Calculate the maximum value	333
MIN.	Calculate the minimum value	334
NOT	Present the opposite condition	335
NULL	Present rows with missing entries	337
OR	Present either of two conditions	338
P.	Present information in a table	339
SUM.	Calculate the sum	342
U.	Update a row in a table	343
UNQ.	Eliminate duplicate rows	344
USER	Present rows with a value of *user ID*	345
+ – * /	Calculate arithmetic expressions	345
= ¬= > <	Present on equality and inequality	347

ALL. — Display duplicate rows

ALL. ensures that all rows, including duplicate rows display. Specify ALL. under the table name in the row with the P. operator. You can *only* use ALL. in rows with P..

ALL. is the default operator if a sample table has only one P. row. For this query, you would *not* have to specify ALL. to display all the rows in the report.

```
Q.ORG | DEPTNUMB | DEPTNAME | MANAGER | DIVISION | LOCATION |
------+----------+----------+---------+----------+----------|
ALL.  |          |          |         | P.       |          |
```

However, if a table has two or more P. rows, QMF excludes duplicates. See also “UNQ. — Eliminate duplicate rows” on page 344.

AND — Present on two conditions

Two conditions connected by AND cause the query to select only rows that satisfy both conditions. The query below selects rows where the YEARS column is equal to 10, and the SALARY column is greater than 20000. The query selects only the two rows that satisfy both of these conditions.

When you run this query:

```
Q.STAFF | ID | NAME | DEPT | JOB | YEARS | SALARY | COMM |
--------+----+------+------+-----+-------+--------+------|
        | P. | P.   |      |     | P. _Y | P. _S  |      |

|        CONDITIONS           |
|-----------------------------|
| _Y = 10 AND _S > 20000      |
```

QMF produces this report:

```
   ID  NAME       YEARS     SALARY
-----  ---------  -----  ---------
   50  HANES         10   20659.80
  210  LU            10   20010.00
```

Two conditions on one row

You can specify two conditions on the same row of an example table. For example, to display every clerk in Department 20, the following query is the same as connecting the two conditions by AND.

```
Q.STAFF | ID | NAME | DEPT | JOB   |
--------+----+------+------+-------|
P.      |    |      | 20   | CLERK |
```

AO., AO(n). — Sort rows in ascending order

To put rows in a report in ascending order by the values in some column, put `AO.` in that column. (Make sure that you use the letter O.)

The sorting sequence for character data, in ascending order, is as follows:

1. Special characters, including blank
2. Lowercase letters, in alphabetic order
3. Uppercase letters, in alphabetic order
4. Numbers, in ascending order
5. NULL

The sorting sequence for DATE, TIME, and TIMESTAMP values is chronological.

The internal value of the data determines the sorting sequence for double-byte character set (DBCS) data. It is generally not meaningful.

The following query produces a report listing the name, job, and years of employment for each employee in department 84 in ascending alphabetic order by job.

When you run this query:

```
Q.STAFF | ID | NAME | DEPT | JOB         | YEARS | SALARY | COMM |
--------+----+------+------+-------------+-------+--------+------|
        |    | P.   | 84   | P. AO.      | P.    |        |      |
```

QMF produces this report:

```
NAME       JOB    YEARS
---------  -----  -----
GAFNEY     CLERK      5
QUILL      MGR       10
DAVIS      SALES      5
EDWARDS    SALES      7
```

Order by more than one column

To order by more than one column, put `AO(1).` under the column to be ordered first. Then, put `AO(2).` under the next most significant column, and so on.

The number following AO. indicates the sort priority. The sequence of sort priorities you use need not be complete. For example, you can use 1, 2, and 4 without using 3, but no two columns can have the same priority.

The following query sorts by job first (in ascending order). Then, within each job classification, it sorts the rows by years of employment (in ascending order).

When you run this query:

```
Q.STAFF | NAME | DEPT | JOB       | YEARS     |
--------+------+------+-----------+-----------|
        | P.   | 84   | P. AO(1). | P. AO(2). |
```

QMF produces this report:

```
NAME       JOB    YEARS
---------  -----  -----
GAFNEY     CLERK      5
QUILL      MGR       10
DAVIS      SALES      5
EDWARDS    SALES      7
```

You can only sort on columns that the query selects. If you use AO. in some row and column of an example table, you must use P. either in the same row and column, or in the same row under the table name (which displays every column).

AVG. — Calculate the average value

The AVG. column function calculates the average of all values in a column for rows selected. It applies to numeric data and returns a single value for that data. You can use the UNQ. operator with AVG. to use only unique values when calculating the average. A column function does not include null values in its calculation.

To select only the SALARY column, define an example element for SALARY in the SALARY column. Then, add an unnamed column, request the average, and put the example element in the unnamed column.

When you run this query:

```
Q.STAFF | SALARY |         |
--------+--------+---------|
        | _S     |P.AVG._S |
```

QMF produces this report:

```
      AVG(SALARY)
-----------------
  16675.6422857142
```

To find the average of the values in the SALARY column for clerks only, add a condition to your query:

When you run this query:

```
Q.STAFF | SALARY | JOB   |          |
--------+--------+-------+----------|
        | _S     | CLERK | P.AVG._S |
```

QMF produces this report:

```
      AVG(SALARY)
-----------------
  12612.6125000000
```

Rules for AVG.

- You can only use AVG. on columns of numeric data.
- In an unnamed column, you must specify AVG. along with the example element that identifies the column to be averaged.
- AVG. can be followed by an example element, an arithmetic expression that contains at least one example element, or the UNQ. operator that is followed by an example element. If the data in a column you want to average is defined by an arithmetic expression, enclose the expression in parentheses.
- When you apply AVG. to one column named in an example table, you must apply a column function (AVG., MIN., MAX., COUNT., OR SUM.) or the G. (group) operator to every other column selected.

BETWEEN x AND y — Present values within a range

You can select all the rows that have a value between two limits. The limits are inclusive. You can abbreviate BETWEEN as BT. Comparisons using BETWEEN do not work unless the smaller value comes before the larger value. In the following example, notice that the smaller value, 20000, appears immediately after BT.

When you run this query:

```
Q.STAFF | ID | NAME | SALARY             |
--------+----+------+--------------------|
P.      |    |      | BT 20000 AND 21000 |
```

QMF produces this report:

```
   ID  NAME         SALARY
-----  ---------  ---------
   50  HANES       20659.80
  210  LU          20010.00
  310  GRAHAM      21000.00
```

You can select all rows that have YEARS equal to 8, 9, or 10.

When you run this query:

```
Q.STAFF | ID | NAME   | YEARS            | SALARY     |
--------+----+--------+------------------+------------|
P.      |    |        | BETWEEN 8 AND 10 |            |
```

QMF produces this report:

```
   ID  NAME        YEARS     SALARY
-----  ---------  ------ ----------
   20  PERNAL          8   18171.25
   50  HANES          10   20659.80
  190  SNEIDER         8   14252.75
  210  LU             10   20010.00
  270  LEA             9   18555.50
  280  WILSON          9   18674.50
  290  QUILL          10   19818.00
```

Use BETWEEN either in an example table or in a CONDITIONS box. You can enter _Y in the YEARS column and _Y BETWEEN 8 AND 10 in a CONDITIONS box to produce the same report as above.

```
Q.STAFF | ID | NAME   | YEARS         | SALARY     |
--------+----+--------+---------------+------------|
P.      |    |        | _Y            |            |

 | CONDITIONS               |
 |--------------------------|
 | _Y BETWEEN 8 AND 10      |
```

Note: _Y BETWEEN 8 and 10 produces the same results as _Y >= 8 AND _Y <= 10, but is easier to write.

COUNT. — Count the number of values in a column

The COUNT. column function finds the number of unique values in a column. Specify COUNT. either in an unnamed column or in a target table. You can abbreviate COUNT. as CNT.

The following query finds the average salary of each department for those departments with more than four members.

When you run this query:

```
Q.STAFF | DEPT | ID  | SALARY |            |
--------+------+-----+--------+------------|
        | G.P. | _ID |  _S    | P. AVG._S  |


| CONDITIONS     |
|----------------|
| COUNT._ID > 4  |
```

QMF produces this report:

```
  DEPT       AVG SALARY
------  ---------------
    38  15457.110000000
    51  17218.160000000
    66  17215.240000000
```

COUNT. can count values in columns of any data type. For example, by adding a search condition, you can determine the number of employees with a salary in a given range or the number of employees at a given location.

Rules for COUNT.

- COUNT. counts only unique values.
- An example element must follow COUNT.
- You cannot follow COUNT. with an expression or an example element within an expression.
- You can only use COUNT. in reference to a specific column. Follow COUNT. by an example element alone.

D. — Delete rows from a table

To delete one or more rows from a table, put the operator D. under the table name in the row you want deleted.

You can delete rows from a table that you created or from a copy of a table that someone else created. (You need authorization to create or copy a table.) To copy the Q.STAFF sample table, for example, enter DISPLAY Q.STAFF. When

Q.STAFF displays, enter SAVE DATA AS PERS. The examples that use D. assume that you created (or copied) a table and called it PERS.

This query deletes the row that contains the ID number 140 from the PERS table:

```
PERS | ID           | NAME | DEPT | JOB | YEARS | SALARY | COMM |
-----+--------------+------+------+-----+-------+--------+------|
D.   | 140          |      |      |     |       |        |      |
```

You can delete more than one row with one DELETE statement.

This query deletes everyone in Department 10:

```
PERS | ID | NAME | DEPT | JOB | YEARS | SALARY | COMM |
-----+----+------+------+-----+-------+--------+------|
D.   |    |      | 10   |     |       |        |      |
```

The example table with D. can have multiple rows, but you cannot mix the operators D., I., P., or U. in a single example table.

Attention:
If D. appears under the table name with no conditions in other columns, QMF deletes the entire contents of the table.

Rules for D.

- An example table can have multiple D. rows.
- You cannot delete rows if the deletion is dependent on values in other rows of the same table.

DO., DO(n). — Sort rows in descending order

To put rows in a report in descending order by the values in some column, put DO. in that column. Use the letter “O” (not the digit zero “0”).

The sorting sequence for character data, in descending order, is as follows:

1. NULL
2. Numbers, in descending order
3. Uppercase letters, in descending alphabetic order
4. Lowercase letters, in descending alphabetic order
5. Special characters, including blank

With DO., the sorting sequence for DATE, TIME, and TIMESTAMP values is reverse chronological.

The internal value of the data determines the sorting sequence for DBCS data. The sorting sequence generally is not meaningful.

The following query produces a report that lists the name, job, and years of employment for each employee in department 84 in descending order by job.

When you run this query:

```
Q.STAFF | ID | NAME | DEPT | JOB         | YEARS | SALARY | COMM |
--------+----+------+------+-------------+-------+--------+------|
        |    | P.   | 84   | P. DO.      | P.    |        |      |
```

QMF produces this report:

```
NAME       JOB    YEARS
---------  -----  -----
EDWARDS    SALES      7
DAVIS      SALES      5
QUILL      MGR       10
GAFNEY     CLERK      5
```

Order by more than one column

To order by more than one column, put `DO(1).` under the column to be ordered first. Then, put `DO(2).` under the next most significant column, and so on.

The number following `DO.` is called the sort priority. The sequence of sort priorities you use need not be complete. For example, you can use 1, 2, and 4 without using 3, but no two columns can have the same priority.

The following query sorts by job first (in descending order). Then, within each job classification, it orders the rows by years of employment, beginning with the greatest number of years (descending order).

When you run this query:

```
Q.STAFF | NAME | DEPT | JOB        | YEARS      |
--------+------+------+-----------+-----------|
        | P.   | 84   | P. DO(1). | P. DO(2). |
```

QMF produces this report:

```
NAME       JOB    YEARS
---------  -----  -----
EDWARDS    SALES      7
DAVIS      SALES      5
QUILL      MGR       10
GAFNEY     CLERK      5
```

You can sort on only those columns that the query selects. If you use DO. in some row and column of an example table, you must use P. either in the same row and column, or in the same row under the table name (which displays every column).

G. — Grouping data

The keyword G. groups selected rows by a specified column for the purpose of performing operations on each group. G. accumulates the results by group, but it does not order the groups. (Use AO. or DO. to ensure the desired order.)

For example, you can group by department to determine each department's average salary with the following steps:

1. Group the rows by department number (G. under DEPT).
2. Specify one average for each department (_S under SALARY and AVG._S in an unnamed column to link _S to the SALARY column).
3. Add P. in the columns you want the results selected.
4. Add AO. to put them in ascending order by department.

When you run this query:

```
Q.STAFF | DEPT    | SALARY |          |
--------+---------+--------+----------|
        | G.P.AO. | _S     | P.AVG._S |
```

QMF produces this report:

```
  DEPT     AVG(SALARY)
------  ---------------
    10  20865.862500000
    15  15482.332500000
    20  16071.525000000
    38  15457.110000000
    42  14592.262500000
    51  17218.160000000
    66  17215.240000000
    84  16536.750000000
```

Generally, G. produces one group for each set of identical values in a column. If there are null values in the column, they form a single group.

Rules for G.

- Any example element that does not refer to a G. column must have an associated column function.
- When you use grouping, you can only select data that refers to the groups. Only columns that contain G. or an aggregating function can contain P.
- A row of an example table that uses G.cannot use I., U., or D.
- If more than one column contains G., QMF groups the selected rows by each unique value of the combined columns. For example, if G. appears in both the DEPT column and the LOCATION column, each row of a group has the same DEPT value and LOCATION value.

I. — Insert rows into a table

To insert one or more rows into a table, put the operator I. under the table name and the values you want to insert under their respective columns. Each row you want to insert must contain the I. operator.

If you leave a blank under a column, or omit a column from the example table, a null value is inserted in that column in the database. You must specify values for all columns that are defined as NOT NULL.

You can insert rows into a table you created or into a copy of a table someone else created.rows that are created by someone else (You need authorization to create or copy a table.) To copy the Q.STAFF sample table, for example, enter DISPLAY Q.STAFF. When Q.STAFF displays, enter SAVE DATA AS PERS. The examples using I. assume that you created (or copied) a table and called it PERS.

This query inserts two rows into the PERS table:

```
PERS | ID  | NAME     | DEPT | JOB   | YEARS | SALARY   | COMM   |
-----+-----+----------+------+-------+-------+----------+--------|
I.   | 400 | HARRISON | 20   | SALES |       | 18000.66 | 0      |
I.   | 455 | STONER   | 17   |       |       | 19000.00 | 540.00 |
```

This query inserts date and time values in a table called TEST.DATETIME:

```
TEST.DATETIME | SMALLINTEGER |     DATE     |    TIME    |
--------------+--------------+--------------+------------|
I.            |              | '1987-11-11' | '14.22.00' |
```

If authorized, you can copy rows from one table to another using I.. In the query below, the example elements show which columns the query copies from Q.STAFF into PERS. The DEPT column in Q.STAFF is duplicated; one

DEPT column contains a condition that limits one set of rows to those from Department 38. The YEARS column is also duplicated; one YEARS column contains a condition that limits the second set of rows to those with YEARS > 10. If employees in Department 38 have more than ten years of experience, they are in the report twice.

```
Q.STAFF | ID  | NAME | DEPT | DEPT | JOB | YEARS | YEARS |
--------+-----+------+------+------+-----+-------+-------|
        | _I1 | _N1  | _D1  | 38   | _J1 | _Y1   |       |
        | _I2 | _N2  | _D2  |      | _J2 | _Y2   | >10   |


PERS | ID  | NAME | DEPT | JOB | YEARS |
-----+-----+------+------+-----+-------|
I.   | _I1 | _N1  | _D1  | _J1 | _Y1   |
I.   | _I2 | _N2  | _D2  | _J2 | _Y2   |
```

A CONDITIONS box containing the conditions _D1 = 38 and _Y2 > 10 could be used instead of duplicating the DEPT and YEARS columns.

Rules for I.

- You cannot use a column name more than once in a table that receives inserted rows.
- You cannot insert a row in a table into the same table.

IN (x,y,z) — Present certain values in a set

You can select all the rows that contain any value in a set of values. Enclose the values in parentheses and separate one value from the next with a comma. A blank between values is optional. (You can not specify NULL in a set of values.)

In the following query, the condition IN (20, 38, 42) in the DEPTNUMB column means "select every row with a department number of 20, 38, or 42." It is equivalent to, but simpler than, writing the condition _D=20 OR _D=38 OR _D=42.

When you run this query:

```
Q.ORG | DEPTNUMB        | DEPTNAME |
------+-----------------+----------|
P.    | IN (20, 38, 42) |          |
```

QMF produces this report:

```
DEPTNUMB  DEPTNAME
--------  ---------------
      20  MID ATLANTIC
      38  SOUTH ATLANTIC
      42  GREAT LAKES
```

LIKE — Present on part of a value

To select character or graphic data when you only know part of a value, use LIKE plus a symbol for the unknown data.

- An underscore (_) is the symbol for any single character. Use more than one underscore in succession to represent an exact number of missing characters in their specific location.
- A percent sign (%) is the symbol for any number of characters, or none.

You can use both symbols in the same value.

You can only use LIKE with character or graphic data.

- For character data, you must always enclose the value after LIKE in single quotation marks. (z/OS and OS/390 requires single quotation marks around an all-digit value of character data.)
- For graphic data, you must always precede the value after LIKE by the single-byte character "G."

Like any single character (underscore)

You can specify a search value that ignores a given number of characters. The underscore (_) in the following condition means to ignore the character between LE and DS. In other words, search for LE, followed by any one character, followed by DS.

```
LIKE 'LE_DS'
```

This singles out the name LEEDS from the NAMES column.

Enclose values containing underscore characters in single quotation marks to prevent search values from being mistaken for example elements.

Use a specific number of underscores to indicate that you want that number of characters ignored. For example, for an 8-character column of part numbers, you can use the following condition to search it for the combination G2044 in positions 2 through 6. The first character and the last two can be any characters.

```
LIKE '_G2044_ _'
```

Like any number of characters (percent sign)

You can select rows that contain a string of characters that are part of a word or number that you know exists in the data. In the following query, LIKE %NY in the ADDRESS column means "where the address ends with NY, with anything at all before that.” The *percent sign* (%) stands for “anything at all” (any number of preceding characters or none).

When you run this query:

```
Q.APPLICANT | NAME | ADDRESS   |
------------+------+----------|
P.          | AO.  | LIKE %NY |
```

QMF produces this report:

```
NAME       ADDRESS
---------  -----------------
JACOBS     POUGHKEEPSIE, NY
LEEDS      EAST FISHKILL, NY
REID       ENDICOTT, NY
```

Data-type dependencies

When the data type of a column is VARCHAR, you do not need to know how many blanks to specify with LIKE. With VARCHAR, there are no blanks in the column. The size of the column varies with the size of its data.

When the data type of a column is CHAR, however, the size of the column is fixed. The column contains blanks, so use the appropriate number of blanks when you specify LIKE.

If the data type of a column is LONG VARCHAR or LONG VARGRAPHIC, you cannot use it with `LIKE` (or any other search conditions).

MAX. — Calculate the maximum value

The `MAX.` column function returns the greatest value in the group of numbers or characters in a specified column. You can apply `MAX.` to columns of any type.

If `MAX.` is applied to a CHAR or VARCHAR type column, alphanumeric ordering is used.

- The number 9 is greater than 8, and so on, to 0 (zero).
- Zero is greater than the uppercase Z, which is greater than Y, and so on, to A.
- A is greater than the lowercase z, which is greater than y, and so on, to a.
- Lowercase a is greater than the special characters.

QMF ignores null values when searching for the maximum. If all values specified in a column are null, QMF returns no value.

You can use an example element with MAX. to select the maximum years of employment and maximum salary from the Q.STAFF table.

When you run this query:

```
Q.STAFF | YEARS | SALARY     |            |            |
--------+-------+------------|------------+------------|
        | _Y    | _S         | P. MAX. _Y | P. MAX. _S |
```

QMF produces this report:

```
MAX(YEARS)   MAX(SALARY)
----------   -----------
        13      22959.20
```

Rules for MAX.

- You can follow MAX. with an example element or an arithmetic expression that contains at least one example element.
- In an unnamed column, you must specify MAX. with the example element that identifies the column from which the largest value is to be retrieved. This example element also appears in the column that contains the value to retrieve.
- When you apply MAX. to a column named in an example table, you must apply a column function (AVG., SUM., MIN., MAX., COUNT.), or the G. (group) operator to every other column to select.

MIN. — Calculate the minimum value

The MIN. column function returns the smallest value in the group of numbers or characters in a specified column. You can apply MIN. to columns of any type.

If MIN. is applied to a CHAR or VARCHAR type column, alphanumeric ordering is used.

- The number 9 is greater than 8, and so on, to 0 (zero).
- Zero is greater than the uppercase Z, which is greater than Y, and so on, to A.
- A is greater than the lowercase z, which is greater than y, and so on, to a.
- Lowercase a is greater than the special characters.

QMF ignores null values when searching for the minimum. If all values specified in a column are null, QMF returns no value.

You can use an example element with MIN. to select the minimum years of employment for employees in the Q.STAFF table.

When you run this query:

```
Q.STAFF | NAME     | DEPT | YEARS |            |
--------+---------+------+-------+------------|
        |          |      | _Y    | P. MIN. _Y |
```

QMF produces this report:

```
MIN(YEARS)
----------
         1
```

Rules for MIN.

- You can follow MIN. with an example element or an arithmetic expression that contains at least one example element.
- In an unnamed column, specify MIN. with the example element that identifies the column from which the smallest value is to be retrieved. This example element also appears in the column that contains the value to retrieve.
- When you apply MIN. to a column named in an example table, you must apply a column function (AVG., SUM., MIN., MAX., COUNT.), or the G. (group) operator to every other column to select.

NOT — Present on the opposite of the condition

You can use the opposite of any condition by putting NOT before it. NOT takes precedence over AND and OR. For example, in this query, rows are selected that do not contain 38 in the DEPTNUMB column but do contain EASTERN in the DIVISION column. The other row in the Q.ORG table containing EASTERN in the DIVISION column contains 38 in the DEPTNUMB column, so it is not presented.

When you run this query:

```
Q.ORG | DEPTNUMB | DIVISION   | LOCATION       |
------+----------+------------+----------------|
P.    | _DEP     | _DIV       |                |

|        CONDITIONS            |
|------------------------------|
| NOT _DEP=38 AND _DIV=EASTERN |
```

QMF produces this report:

```
DEPTNUMB  DIVISION       LOCATION
--------  -------------  ---------
      15  EASTERN        BOSTON
      20  EASTERN        WASHINGTON
```

To illustrate how parentheses can change the results of a query, the first query that follows contains no parentheses. The second adds some parentheses. The third moves them a little.

When you run this query:

```
Q.ORG | DEPTNUMB | DIVISION   | LOCATION       |
------+----------+------------+---------------|
P.    | _DEP     | _DIV       | _LOC           |

|        CONDITIONS                           |
|---------------------------------------------|
| NOT _DEP=51 AND _DIV=MIDWEST OR _LOC=BOSTON |
```

QMF produces this report:

```
DEPTNUMB  DIVISION       LOCATION
--------  -------------  ---------
      15  EASTERN        BOSTON
      42  MIDWEST        CHICAGO
```

When you place parentheses as follows, your report is exactly the same as in the previous example.

```
(NOT _DEP=51 AND _DIV=MIDWEST) OR _LOC=BOSTON
```

However, if you move the left parenthesis after NOT, as in the following query, you get different results.

When you run this query:

```
Q.ORG | DEPTNUMB | DIVISION   | LOCATION       |
------+----------+------------+---------------|
P.    | _DEP     | _DIV       | _LOC           |

|        CONDITIONS                             |
|-----------------------------------------------|
| NOT (_DEP=51 AND _DIV=MIDWEST) OR _LOC=BOSTON |
```

QMF produces this report:

```
DEPTNUMB  DIVISION       LOCATION
--------  -------------  ---------
      10  CORPORATE      NEW YORK
```

```
    15  EASTERN        BOSTON
    20  EASTERN        WASHINGTON
    38  EASTERN        ATLANTA
    42  MIDWEST        CHICAGO
    66  WESTERN        SAN FRANCISCO
    84  WESTERN        DENVER
```

Rules for NOT

- You can write NOT =, NOT NULL, NOT LIKE, NOT IN, or NOT BETWEEN.
- With greater than or less than, NOT must precede the entire condition, for example, NOT _YEARS > 10.

NULL — Present rows with missing entries

If you create a table that is partially filled with data, QMF places the code word NULL, which means “value unknown”, in the locations that contain no data. Do not confuse NULL with any of these values:

- A numeric value of zero
- A character string of all blanks
- A character string of length zero
- The character string NULL (of length 4)

Each of the above is a value that you can enter in some row and column of a table. NULL occurs when no value is entered, or where the value is specifically set to NULL. It prints and displays as a single hyphen (-).

To select rows that have no entry in a column, put NULL in that column. For example, you can display the IDs and names of employees in Department 38 for whom YEARS is null.

When you run this query:

```
Q.STAFF | ID | NAME | DEPT | JOB | YEARS | SALARY | COMM |
--------+----+------+------+-----+-------+--------+------|
        | P. | P.   | 38   |     | NULL  |        |      |
```

QMF produces this report:

```
    ID   NAME
-------  ---------
     60  QUIGLEY
    120  NAUGHTON
```

You can not use NULL with an example element in an example table. Use a CONDITIONS box instead. For example:

This query is INCORRECT:

```
Q.STAFF | NAME | COMM        | SALARY |          |
--------+------+------------+--------+----------|
P.      |      | _C ¬=NULL  | _S     | _C + _S  |
```

This query is CORRECT:

```
Q.STAFF | NAME | COMM        | SALARY |          |
--------+------+------------+--------+----------|
P.      |      | _C         | _S     | _C + _S  |

|       CONDITIONS       |
|------------------------|
| _C ¬=NULL              |
```

Unknown values

QMF interprets the NULL keyword as “unknown”. The result of an operation on an unknown value is also unknown, so the result of any operation on NULL is NULL.

Remember that NULL is not zero. NULL is the absence of a value. In the sample table Q.STAFF, there is no value for COMM for managers because managers do not make commissions. In some examples, we calculate earnings as SALARY + COMM. If we did this calculation for managers, the result would always be NULL.

Rules for NULL:

- You can use NULL alone or with =, ¬=, or NOT.
- In a CONDITIONS box, you can use NULL only with a column name or an example element.

OR — Present on either of two conditions

Two conditions connected by OR allow the query to select every row that satisfies one or the other of the conditions. The following query selects rows where either the YEARS column is equal to 10 or the SALARY column is greater than 20000.

When you run this query:

```
Q.STAFF | ID | NAME | DEPT | JOB | YEARS | SALARY | COMM |
--------+----+------+------+-----+-------+--------+------|
        | P. | P.   |      |     | P. _Y | P. _S  |      |

|        CONDITIONS          |
|----------------------------|
| _Y = 10 OR _S > 20000      |
```

QMF produces this report:

```
   ID  NAME       YEARS     SALARY
-----  ---------  -----  ---------
   50  HANES         10   20659.80
  140  FRAYE          6   21150.00
  160  MOLINARE       7   22959.20
  210  LU            10   20010.00
  260  JONES         12   21234.00
  290  QUILL         10   19818.00
  310  GRAHAM        13   21000.00
```

P. — Present data in a table

You can use P. to present all the columns or some of the columns in a table. You can not use the D. (delete), I. (insert), and U. (update) keywords in the same query with P.

Present all the columns in a table

To see all the columns in a table, put P. under the table name in the example table. All the columns shown in the example table appear.

When you run this query:

```
Q.ORG | DEPTNUMB | DEPTNAME | MANAGER | DIVISION | LOCATION |
------+----------+----------+---------+----------+----------|
P.    |          |          |         |          |          |
```

QMF produces this report:

```
DEPTNUMB  DEPTNAME        MANAGER  DIVISION    LOCATION
--------  --------------  -------  ----------  -------------
      84  MOUNTAIN            290  WESTERN     DENVER
      66  PACIFIC             270  WESTERN     SAN FRANCISCO
      10  HEAD OFFICE         160  CORPORATE   NEW YORK
      15  NEW ENGLAND          50  EASTERN     BOSTON
      20  MID ATLANTIC         10  EASTERN     WASHINGTON
      38  SOUTH ATLANTIC       30  EASTERN     ATLANTA
      42  GREAT LAKES         100  MIDWEST     CHICAGO
      51  PLAINS              140  MIDWEST     DALLAS
```

Present some of the columns in a table

To see selected columns, put P. under the names of the columns you want to see. You can put the P. before or after other things you put under the column heading.

When you run this query:

```
Q.ORG | DEPTNUMB | DEPTNAME | MANAGER | DIVISION | LOCATION |
------+----------+----------+---------+----------+----------|
      | P.       |          |         | P.       | P.       |
```

QMF produces this report:

```
DEPTNUMB  DIVISION    LOCATION
--------  ----------  -------------
      84  WESTERN     DENVER
      66  WESTERN     SAN FRANCISCO
      10  CORPORATE   NEW YORK
      15  EASTERN     BOSTON
      20  EASTERN     WASHINGTON
      38  EASTERN     ATLANTA
      42  MIDWEST     CHICAGO
      51  MIDWEST     DALLAS
```

Present some of the rows in a table

To see only certain rows of a table, add conditions to your query. For example, present all the Q.STAFF table columns, but only the rows that contain SALES in the JOB column.

When you run this query:

```
Q.STAFF | ID | NAME | DEPT | JOB   | YEARS | SALARY | COMM |
--------+----+------+------+-------+-------+--------+------|
P.      |    |      |      | SALES |       |        |      |
```

QMF produces this report:

```
    ID  NAME         DEPT  JOB     YEARS      SALARY        COMM
------  ---------  ------  -----  ------  ----------  ----------
    20  PERNAL         20  SALES       8    18171.25      612.45
    40  O'BRIEN        38  SALES       6    18006.00      846.55
    60  QUIGLEY        38  SALES       -    16808.30      650.25
    70  ROTHMAN        15  SALES       7    16502.83     1152.00
    90  KOONITZ        42  SALES       6    18001.75     1386.70
   150  WILLIAMS       51  SALES       6    19456.50      637.65
   220  SMITH          51  SALES       7    17654.50      992.80
   280  WILSON         66  SALES       9    18674.50      811.50
   300  DAVIS          84  SALES       5    15454.50      806.10
```

```
 310  GRAHAM         66  SALES      13    21000.00      200.30
 320  GONZALES       66  SALES       4    16858.20      844.00
 340  EDWARDS        84  SALES       7    17844.00     1285.00
```

Present data from multiple tables

To present data from two tables, draw two example tables that have at least one column that contains the same data (in the example, ID and MANAGER). Add one or more unnamed columns to one of the tables. Enter the same example element in each table in the columns that contain the same data. Then, enter another example element in an unnamed column of the first table, and enter that same example element in a named column of the second table. (P. can appear only in the table with the unnamed column.)

When you run this query:

```
Q.STAFF | ID | NAME |      |
--------+----+------+------|
P.      | _I |      | _D   |

Q.ORG | DEPTNUMB | DEPTNAME | MANAGER |
------+----------+----------+---------|
      | _D       |          | _I      |
```

QMF produces this report:

```
    ID  NAME       DEPTNUMB
------  ---------  --------
    10  SANDERS          20
    30  MARENGHI         38
    50  HANES            15
   100  PLOTZ            42
   140  FRAYE            51
   160  MOLINARE         10
   270  LEA              66
   290  QUILL            84
```

Present data dependent on non-presented data

A query using multiple tables can present data from one table that is dependent on data in another table. For example, using the example element _D in the DEPT column of Q.STAFF and the DEPTNUMB column of Q.ORG, you can present the ID, name, and department of only those employees who are located in Dallas.

When you run this query:

```
Q.STAFF | ID | NAME | DEPT  |
--------+----+------+-------|
P.      |    |      | _D    |


Q.ORG | DEPTNUMB | LOCATION |
------+----------+----------|
      | _D       | DALLAS   |
```

QMF produces this report:

```
    ID  NAME       DEPT
------  ---------  ----
   140  FRAYE        51
   150  WILLIAMS     51
   220  SMITH        51
   230  LUNDQUIST    51
   250  WHEELER      51
```

SUM. — Calculate the total

The SUM. column function calculates the total of all values in a column for selected rows. It applies to a group of numbers and returns a single value for each group of numbers to which it is applied. You can use the UNQ. operator with SUM. to request that QMF use only unique values when calculating the sum. QMF ignores nulls. If all values in the specified column are null, the total is null.

You can use an arithmetic expression with SUM.. The following example calculates the total earnings (salaries plus commissions) for every selected row in Q.STAFF:

```
Q.STAFF | NAME | SALARY | COMM   |                  |
--------+------+--------+--------+------------------|
        |      | _S     | _C     | P. SUM. (_S+_C)  |
```

All the columns referred to in an unnamed column are either grouped or have a column function specified. For example, you can select the total, average, and maximum salaries by department.

When you run this query:

```
Q.STAFF | DEPT | SALARY |            |            |            |
--------+------+--------+------------+------------|------------|
        | P. G.| _S     | P. SUM._S  | P. AVG. _S | P. MAX. _S |
```

QMF produces this report:

```
  DEPT          SUM(SALARY)          AVG(SALARY)  MAX(SALARY)
------  -------------------  ------------------  -----------
    10             83463.45    20865.8625000000     22959.20
    15             61929.33    15482.3325000000     20659.80
    20             64286.10    16071.5250000000     18357.50
    38             77285.55    15457.1100000000     18006.00
    42             58369.05    14592.2625000000     18352.80
    51             86090.80    17218.1600000000     21150.00
    66             86076.20    17215.2400000000     21000.00
    84             66147.00    16536.7500000000     19818.00
```

Rules for SUM.

- You can use SUM. only on columns of numeric data type.
- In an unnamed column, specify SUM. with the example element that identifies the column to be totaled.
- SUM. can be followed by an example element, an arithmetic expression that contains at least one example element, or the UNQ. operator that is followed by an example element. If the data in a column to be totaled is defined by an arithmetic expression, enclose the expression in parentheses.
- When you apply SUM. to a column named in an example table, you must apply a column function (AVG., MIN., MAX., COUNT. or SUM.), or the G. (group) operator to every other column to select. See “G. — Grouping data” on page 329.

U. — Update a row in a table

To update one or more values in an existing row of a table, put the operator U., with the new value, in each column you want to change. An example table that uses U. can have more than one row. However, all rows must contain the U. operator. Values in other columns identify the row or rows to change.

You can update rows in a table that you created or in a copy of a table someone else created. (You need authorization to create or copy a table.) For example, to copy the Q.STAFF sample table, enter DISPLAY Q.STAFF. When Q.STAFF displays, enter SAVE DATA AS PERS. The examples that use U. assume that you created (or copied) a table and called it PERS.

This query updates the PERS table for employees 250 and 330. It changes data in the JOB column to SALES and increases salary by 15%.

```
PERS | ID   | JOB       | SALARY | SALARY       |
-----+------+-----------+--------+--------------|
     | 250  | U. SALES  | _S1    | U. _S1*1.15  |
     | 330  | U. SALES  | _S2    | U. _S2*1.15  |
```

To see the changed rows in your PERS table, enter `DISPLAY PERS`. The updated PERS table should look like this:

```
    ID  NAME       JOB        SALARY
------  ---------  -----  ----------
   250  WHEELER    SALES    16629.00
   330  BURKE      SALES    12636.00
```

To update date and time values in a QBE update query, enclose them in single quotation marks. For example:

```
MY.INTERVIEW |     INTDATE     |   STARTTIME  | MANAGER  |
-------------+-----------------+--------------+----------|
             | U.'1987-04-04'  | U.'14.22.00' | 270      |
```

Rules for U.

- You can only update a column with a constant or with values from other columns in the same row.
- You cannot update a column in a row from columns in other rows in the same table.
- You cannot update a row if it is dependent on other rows in the same table.

UNQ. — Eliminate duplicate rows

`UNQ.` eliminates duplicate rows from query results. If your example table contains two or more `P.` rows, QMF deletes duplicate rows from the query result by default. However, if your table has only one `P.` row, and you want to prevent duplicate rows, use `UNQ.` under the table name in the row with the `P.` operator.

When you run this query:

```
Q.ORG | DEPTNUMB | DEPTNAME | MANAGER | DIVISION | LOCATION |
------+----------+----------+---------+----------+----------|
UNQ.  |          |          |         | P.       |          |
```

QMF produces this result:

```
DIVISION
----------
CORPORATE
EASTERN
MIDWEST
WESTERN
```

USER — Present rows with a value equal to a user identification

When you run a query with USER in the NAME column (or in any column containing user identification (user ID) numbers), your own user ID is substituted for the word USER as a condition in the query. You can then share the query with other users, who can run it without change. QMF automatically substitutes their user IDs for the USER keyword. (USER is *not* preceded by &.)

Suppose, for example, that you usually run a query once a month (using the Q.STAFF table) to find out what your commission is to the current day. You discover that your friends also want to check the same information. You can write the following query and share it with them.

```
Q.STAFF | NAME | COMM |
--------+------+------|
        | USER | P.   |
```

+, −, *, / — Calculated values

A QBE query can present not only data already in a table, but also results that can be calculated using that data.

_S/12 is an example of an expression. It means the result of dividing SALARY by 12. You can form expressions by using symbols for operations:

Symbol	**Operation**
+	Add
−	Subtract
*	Multiply
/	Divide

Within expressions you can use column headings (RATE*HOURS), constants (RATE*1.07), and column functions (AVG.(_S)/2).

In the report, column names for calculated values are different depending on whether you are using DB2 Server for VSE or VM or DB2. You may see, for example:

- 1, 2, or 3
- COL1, COL2, or :COL3
- EXPRESSION 1, EXPRESSION 2, or EXPRESSION 3
- AVG(EXPRESSION 2)

The examples in this book were created using DB2 Server for VSE or VM. You see the term EXPRESSION in column headings for calculated values.

Expression columns

You can produce reports with columns that contain the values of expressions. To do so, put the expression in an unnamed column as in the following query.

To show the total earnings of employees in Department 20, include _S + _C in the unnamed column.

When you run this query:

```
Q.STAFF | ID    | NAME    | DEPT | SALARY | COMM |           |
--------+-------+--------+------+--------+------+-----------|
        | P.    | P.      | 20   | _S     | _C   | P._S + _C |
```

QMF produces this report:

```
   ID  NAME       EXPRESSION 1
-----  ---------  ------------
   10  SANDERS               -
   20  PERNAL     18783.70
   80  JAMES      13632.50
  190  SNEIDER    14379.25
```

The value of SALARY+COMM for employee 10 is NULL, because the value of COMM is NULL, and the result of any calculation with NULL is NULL.

You can get a report for everybody in Department 38 and their monthly salaries.

When you run this query:

```
Q.STAFF | DEPT | NAME | SALARY |         |
--------+------+------+--------+---------|
        | P.38 | P.   | _S     | P._S/12 |
```

QMF produces this report:

```
DEPT  NAME              EXPRESSION 1
----  ----------  ------------------
  38  MARENGHI        1458.895833333
  38  O'BRIEN         1500.500000000
  38  QUIGLEY         1400.691666666
  38  NAUGHTON        1079.562500000
  38  ABRAHAMS        1000.812500000
```

=, ¬=, >, < — Equality and inequality

To select rows that satisfy a condition that is based on equality or inequality, put the condition under the appropriate column.

You can display a report with everyone who has 10 or more years of service.

When you run this query:

```
Q.STAFF | ID | NAME | DEPT | JOB | YEARS          | SALARY | COMM |
--------+----+------+------+-----+----------------+--------+------|
        | P. | P.   |      |     | P. >=10        |        |      |
```

QMF produces this report:

```
    ID  NAME         YEARS
------  ----------  ------
    50  HANES           10
   210  LU              10
   260  JONES           12
   290  QUILL           10
   310  GRAHAM          13
```

If you do not specify an operator this way, the operator defaults to equality. You can write the following query to produce a report that contains all the managers.

When you run this query:

```
Q.STAFF | ID | NAME | DEPT | JOB | YEARS | SALARY | COMM |
--------+----+------+------+-----+-------+--------+------|
        | P. | P.   |      | MGR |       |        |      |
```

QMF produces this report:

```
    ID  NAME
------  ----------
    10  SANDERS
    30  MARENGHI
    50  HANES
   100  PLOTZ
   140  FRAYE
   160  MOLINARE
   210  LU
   240  DANIELS
   260  JONES
   270  LEA
   290  QUILL
```

Equality and inequality

You can display a report that contains everyone in the alphabet after SMITH.

When you run this query:

```
Q.STAFF | ID | NAME         | DEPT | JOB | YEARS | SALARY | COMM |
--------+----+--------------+------+-----+-------+--------+------|
        | P. | P. >SMITH AO.|      |     |       |        |      |
```

QMF produces this report:

```
    ID  NAME
------  ----------
   190  SNEIDER
   250  WHEELER
   150  WILLIAMS
   280  WILSON
   130  YAMAGUCHI
```

QBE exercises with solutions

For the solutions to the following exercises, see “Solutions to exercises” on page 351.

Exercises

Exercises 1 through 4 use the Q.STAFF table.

Exercise 1

1. Write a query to produce a list of employee names and jobs for every employee in Department 84.
2. After you have successfully run the query in step 1, use the Query function key to bring it back to your display panel. Change it to produce a list of employee numbers, employee names, years of service, and salary for every employee in Department 51.
3. Change the preceding query to show all of the columns for employees in Department 51.
4. Produce a report that contains the employee identification number, name, department, and years of service for each person who has no data in the YEARS column.
5. Write a query to produce a list that shows employee identification number, name, job, and years of service for everyone with 10 or more years of service. Would someone with exactly 10 years of service appear on your list?
6. Produce a report that contains the name and commission for any manager whose row contains a commission amount.

Exercise 2

1. Produce a report that contains each clerk's name, department, and years of service. Arrange the report in ascending alphabetic order by employee name.
2. Produce a report that contains the name, department, and years of service for every clerk. Put the department numbers in ascending order and, within each department, put the years of service in ascending order.
3. Write a query to produce a list that shows employee number, employee name, and years of service for all the clerks. Arrange the report by years of service with the most senior clerk first.
4. Change step 3 to again arrange the report in descending order by years of service but, within each year, in ascending order by department number. Include the department numbers in your report.
5. Produce a report that contains each employee whose name contains the letter Z.
6. Produce a report that contains each employee whose name begins with S.
7. Produce a report that contains each employee whose name has an A as its third character.

Exercise 3

1. Produce a report that contains the name, salary, and commission of all persons whose salary is greater than $18,000 or whose commission exceeds $1,000.
2. Produce a report that lists all employees who have no data in their years of service column or no data in their commission column. Show the employee's name, years of service, and commission. (Hint: Remember that you must use the equal (=) or not equal (¬=) symbols when comparing for NULL values in QBE.)
3. Write a query to produce a list that shows employee number, name, and salary for everyone with a salary between $20,000 and $21,000. Did people with a salary of exactly $20,000 or $21,000 appear on your list?
   - If they did appear, how could you exclude them?
   - If they did not appear, how could you include them?
4. Produce a report that lists all managers who have been with the company less than 10 years, but whose salary is at least $20,000. Show the name, job title, years of service, and salary.
5. Show the name, years of service, salary, and commission of those employees with less than 10 years of service and either a salary over $20,000 or a commission over $1,000.

Exercise 4

1. Write a query to produce the name, employee number, salary, commission, and total earnings (salary plus commission) of everyone with a job of sales.
2. Write a query to produce the name, number, salary, commission, and total earnings of everyone with a job of sales whose total earnings are less than $17,500.
3. Produce a report that lists each sales person's name and commission as a percentage of salary. (For example, if a person's salary is $20,000 and commission is $2,000, the commission percentage is 10.) Arrange the report in descending order by the commission percentage.
4. Change step 3 so that commission percentage is based on total earnings (salary plus commission = 100%).

Exercise 5

1. Write a query that will access both the Q.STAFF and Q.ORG tables (DRAW Q.STAFF and DRAW Q.ORG). Produce a report that contains each department name, location, and manager's name.
2. Change step 1 to list only those departments in the Eastern Division.
3. Change step 2 to list any managers in the Eastern Division who have 10 or more years of service. For each manager, list the department name, location, and manager's name.

Exercise 6

1. Make a copy of the Q.STAFF table and call it MYTABLE.
2. Write a query to update MYTABLE. Change the name of the manager for Department 66 to RAMOTH, years of service to 7, and salary to $18,238.50. Write a query to retrieve the row after you have updated it.
3. Write a query that increases the salaries of MYTABLE by 10%. Retrieve all the rows for clerks. Because MYTABLE began with data identical to Q.STAFF, you can randomly check YEARS and SALARY against the Q.STAFF table in Chapter 17, "QMF sample tables," on page 285, to ensure that the correct persons received the salary increases.
4. Insert a new row into MYTABLE. The new employee's information is as follows:

```
ID      =  275
NAME    =  ROGERS
DEPT    =  66
JOB     =  SALES
YEARS   =  NULL
SALARY  =  $14,000.00
COMM    =  NULL
```

After you have inserted the row, write and run a query that will display it.

5. Delete from MYTABLE the rows for salespersons in Department 66.

Solutions to exercises

Note: Solutions show minimum columns. Your answer may include unused columns that have been deleted here.

Solutions to exercise 1

1.

```
Q.STAFF | NAME      | DEPT    | JOB   |
--------+-----------+---------+-------|
        | P.        | 84      | P.    |
```

```
NAME       JOB
---------  -----
QUILL      MGR
DAVIS      SALES
EDWARDS    SALES
GAFNEY     CLERK
```

2.

```
Q.STAFF | ID      | NAME      | DEPT    | YEARS   | SALARY     |
--------+---------+-----------+---------+---------+------------|
        | P.      | P.        | 51      | P.      | P.         |
```

```
    ID  NAME        YEARS      SALARY
------  ---------  ------  ----------
   140  FRAYE           6    21150.00
   150  WILLIAMS        6    19456.50
   220  SMITH           7    17654.50
   230  LUNDQUIST       3    13369.80
   250  WHEELER         6    14460.00
```

3.

```
Q.STAFF | ID | NAME | DEPT | JOB | YEARS | SALARY | COMM |
--------+----+------+------+-----+-------+--------+------|
P.      |    |      | 51   |     |       |        |      |
```

```
    ID  NAME         DEPT  JOB     YEARS      SALARY        COMM
------  ---------  ------  -----  ------  ----------  ----------
   140  FRAYE          51  MGR         6    21150.00           -
   150  WILLIAMS       51  SALES       6    19456.50      637.65
   220  SMITH          51  SALES       7    17654.50      992.80
   230  LUNDQUIST      51  CLERK       3    13369.80      189.65
   250  WHEELER        51  CLERK       6    14460.00      513.30
```

4.

```
Q.STAFF | ID      | NAME       | DEPT    | YEARS  |
--------+---------+------------+---------+--------|
        | P.      | P.         | P.      | P.NULL |
```

```
    ID  NAME         DEPT   YEARS
------  ---------  ------  ------
    60  QUIGLEY        38       -
    80  JAMES          20       -
   120  NAUGHTON       38       -
   200  SCOUTTEN       42       -
```

5.

```
Q.STAFF | ID      | NAME       | JOB    | YEARS  |
--------+---------+------------+--------+--------|
        | P.      | P.         | P.     | P.>=10 |
```

```
    ID  NAME       JOB     YEARS
------  ---------  -----  ------
    50  HANES      MGR        10
   210  LU         MGR        10
   260  JONES      MGR        12
   290  QUILL      MGR        10
   310  GRAHAM     SALES      13
```

6.

```
Q.STAFF | NAME       | JOB    | COMM        |
--------+------------+--------+-------------|
        | P.         | MGR    | P. ¬NULL    |
```

```
NAME              COMM
---------  ----------
```

Note: This is a correct result; it is an empty set. None of the managers is on commission, so no data has been entered in their COMM column.

Solutions to exercise 2

1.

```
Q.STAFF | NAME       | DEPT     | JOB    | YEARS   |
--------+------------+----------+--------+---------|
        | P. AO.     | P.       | CLERK  | P.      |
```

```
NAME          DEPT   YEARS
---------   ------  ------
ABRAHAMS        38       3
BURKE           66       1
GAFNEY          84       5
JAMES           20       -
KERMISCH        15       4
LUNDQUIST       51       3
NAUGHTON        38       -
NGAN            15       5
SCOUTTEN        42       -
SNEIDER         20       8
WHEELER         51       6
YAMAGUCHI       42       6
```

2.

```
Q.STAFF | NAME       | DEPT              | JOB    | YEARS           |
--------+------------+-------------------+--------+-----------------|
        | P.         | P. AO(1).         | CLERK  | P. AO(2).       |
```

```
NAME          DEPT   YEARS
---------   ------  ------
KERMISCH        15       4
NGAN            15       5
SNEIDER         20       8
JAMES           20       -
ABRAHAMS        38       3
NAUGHTON        38       -
YAMAGUCHI       42       6
SCOUTTEN        42       -
LUNDQUIST       51       3
WHEELER         51       6
BURKE           66       1
GAFNEY          84       5
```

3.

```
Q.STAFF | ID      | NAME       | JOB    | YEARS  |
--------+---------+------------+--------+--------|
        | P.      | P.         | CLERK  | P. DO. |
```

```
    ID  NAME        YEARS
------  ---------  ------
    80  JAMES           -
   200  SCOUTTEN        -
   120  NAUGHTON        -
   190  SNEIDER         8
   130  YAMAGUCHI       6
   250  WHEELER         6
   350  GAFNEY          5
   110  NGAN            5
   170  KERMISCH        4
   230  LUNDQUIST       3
   180  ABRAHAMS        3
   330  BURKE           1
```

Notes:

1. Null years sort to the top when you specify descending order.
2. Order of names may vary within order by fields. For example, Scoutten may be listed before James, because both have null years.

4.

```
Q.STAFF | ID    | NAME     | DEPT           | JOB    | YEARS         |
--------+-------+----------+----------------+--------+---------------|
        | P.    | P.       | P. AO(2).      | CLERK  | P. DO(1).     |
```

```
    ID  NAME         DEPT   YEARS
------  ---------  ------  ------
    80  JAMES          20       -
   120  NAUGHTON       38       -
   200  SCOUTTEN       42       -
   190  SNEIDER        20       8
   130  YAMAGUCHI      42       6
   250  WHEELER        51       6
   110  NGAN           15       5
   350  GAFNEY         84       5
   170  KERMISCH       15       4
   180  ABRAHAMS       38       3
   230  LUNDQUIST      51       3
   330  BURKE          66       1
```

5.

```
Q.STAFF | NAME                 |
--------+----------------------|
        | P. LIKE '@(#)'        |
```

```
NAME
---------
KOONITZ
PLOTZ
GONZALES
```

6.

```
Q.STAFF | NAME                 |
--------+----------------------|
        | P. LIKE 'S%'         |
```

```
NAME
---------
SANDERS
SNEIDER
SCOUTTEN
SMITH
```

7.

```
Q.STAFF | NAME                 |
--------+----------------------|
        | P. LIKE '_ _A%'      |
```

```
NAME
---------
NGAN
FRAYE
LEA
GRAHAM
```

Solutions to exercise 3

1.

```
Q.STAFF | NAME       | SALARY      | COMM        |
--------+-----------+------------+------------|
        | P.         | P. _S       | P. _C       |

|        CONDITIONS         |
|--------------------------|
| _S > 18000 OR _C > 1000  |
```

```
NAME           SALARY        COMM
---------  ----------  ----------
SANDERS      18357.50           -
PERNAL       18171.25      612.45
O'BRIEN      18006.00      846.55
HANES        20659.80           -
ROTHMAN      16502.83     1152.00
KOONITZ      18001.75     1386.70
PLOTZ        18352.80           -
FRAYE        21150.00           -
WILLIAMS     19456.50      637.65
MOLINARE     22959.20           -
LU           20010.00           -
DANIELS      19260.25           -
JONES        21234.00           -
LEA          18555.50           -
WILSON       18674.50      811.50
QUILL        19818.00           -
GRAHAM       21000.00      200.30
EDWARDS      17844.00     1285.00
```

2.

```
Q.STAFF | NAME       | YEARS  | COMM        |
--------+-----------+---------+------------|
        | P.         | P. _Y  | P. _C       |

|        CONDITIONS         |
|--------------------------|
| _Y = NULL OR _C = NULL   |
```

```
NAME        YEARS        COMM
---------  ------  ----------
SANDERS         7           -
MARENGHI        5           -
HANES          10           -
QUIGLEY         -      650.25
JAMES           -      128.20
PLOTZ           7           -
NAUGHTON        -      180.00
FRAYE           6           -
MOLINARE        7           -
SCOUTTEN        -       84.20
```

```
LU             10           -
DANIELS         5           -
JONES          12           -
LEA             9           -
QUILL          10           -
```

3.

```
          Q.STAFF | ID      | NAME      | SALARY      |
          --------+--------+-----------+------------|
                  | P.      | P.        | P. _S       |

      Exclusive BETWEEN:                  Inclusive BETWEEN:

|        CONDITIONS          |     |         CONDITIONS          |
|---------------------------|     |-----------------------------|
| _S > 20000 AND _S < 21000 |     | _S >= 20000 AND _S <= 21000 |

             OR                                OR

|        CONDITIONS          |     |         CONDITIONS          |
|---------------------------|     |-----------------------------|
| _S BT 20001 AND 20999      |     | _S BETWEEN 20000 AND 21000  |
```

```
   ID  NAME           SALARY         ID  NAME           SALARY
-----  ---------  ----------      -----  ---------  ----------
   50  HANES        20659.80         50  HANES        20659.80
  210  LU           20010.00        210  LU           20010.00
```

4.

```
Q.STAFF | NAME      | JOB              | YEARS  | SALARY     |
--------+-----------+-----------------+--------+------------|
        | P.        | P. =MGR          | P. <10 | P. >=20000 |
```

```
NAME       JOB     YEARS      SALARY
---------  -----  ------  ----------
FRAYE      MGR         6    21150.00
MOLINARE   MGR         7    22959.20
```

5.

```
Q.STAFF | NAME       | YEARS  | SALARY     | COMM       |
--------+------------+--------+------------+------------|
        | P.         | P. <10 | P. _S      | P. _C      |

|        CONDITIONS        |
|--------------------------|
| _S > 20000 OR _C > 1000  |
```

```
NAME        YEARS      SALARY        COMM
---------  ------  ----------  ----------
ROTHMAN         7    16502.83     1152.00
KOONITZ         6    18001.75     1386.70
FRAYE           6    21150.00           -
MOLINARE        7    22959.20           -
EDWARDS         7    17844.00     1285.00
```

Solutions to exercise 4

1.

```
Q.STAFF | NAME | ID  | JOB    | SALARY  | COMM   |          |
--------+------+-----+--------+---------+--------+----------|
        | P.   | P.  | SALES  | P. _S   | P. _C  | P. _S+_C |
```

```
NAME           ID      SALARY        COMM  EXPRESSION 1
---------  ------  ----------  ----------  ------------
PERNAL         20    18171.25      612.45      18783.70
O'BRIEN        40    18006.00      846.55      18852.55
QUIGLEY        60    16808.30      650.25      17458.55
ROTHMAN        70    16502.83     1152.00      17654.83
KOONITZ        90    18001.75     1386.70      19388.45
WILLIAMS      150    19456.50      637.65      20094.15
SMITH         220    17654.50      992.80      18647.30
WILSON        280    18674.50      811.50      19486.00
DAVIS         300    15454.50      806.10      16260.60
GRAHAM        310    21000.00      200.30      21200.30
GONZALES      320    16858.20      844.00      17702.20
EDWARDS       340    17844.00     1285.00      19129.00
```

2.

```
Q.STAFF | NAME     | ID     | JOB    | SALARY     | COMM      |           |
--------+----------+-------+--------+-----------+----------+-----------|
        | P.       | P.     | SALES  | P. _S      | P. _C     | P. _S+_C  |

|        CONDITIONS        |
|--------------------------|
| (_S + _C) < 17500        |
NAME           ID      SALARY        COMM  EXPRESSION 1
---------  ------  ----------  ----------  ------------
QUIGLEY        60    16808.30      650.25      17458.55
DAVIS         300    15454.50      806.10      16260.60
```

3.

```
Q.STAFF | NAME   | JOB    | SALARY   | COMM   |                      |
--------+--------+-------+---------+--------+----------------------|
        | P.     | SALES | _S       | _C     | P. DO. 100*(_C/_S)  |
```

```
NAME             EXPRESSION 1
---------  ------------------
KOONITZ            7.70313900
EDWARDS            7.20130000
ROTHMAN            6.98062000
SMITH              5.62349500
DAVIS              5.21595600
GONZALES           5.00646500
O'BRIEN            4.70148800
WILSON             4.34549700
QUIGLEY            3.86862400
PERNAL             3.37043400
WILLIAMS           3.27731000
GRAHAM             0.95380900
```

You might try using this instead:

```
Q.STAFF | NAME | JOB    | SALARY    | COMM    |                      |
--------+------+-------+----------+---------+----------------------|
        | P.   | SALES | _S        | _C      | P. DO. (100*_C)/_S  |
```

```
NAME             EXPRESSION 1
---------  ------------------
EDWARDS                     7
KOONITZ                     7
ROTHMAN                     6
GONZALES                    5
DAVIS                       5
SMITH                       5
O'BRIEN                     4
WILSON                      4
PERNAL                      3
QUIGLEY                     3
WILLIAMS                    3
GRAHAM                      0
```

Note what happens to the precision of the ratio that you calculated. This truncation can also affect the order into which your rows are sorted.

4.

```
Q.STAFF | NAME | JOB   | SALARY | COMM |                         |
--------+------+-------+--------+------+-------------------------|
        | P.   | SALES | _S     | _C   | P. DO. 100*(_C/(_S+_C)) |
```

```
NAME              EXPRESSION 1
---------  ------------------
KOONITZ            7.15219600
EDWARDS            6.71754900
ROTHMAN            6.52512600
SMITH              5.32409500
DAVIS              4.95738100
GONZALES           4.76776800
O'BRIEN            4.49037300
WILSON             4.16452800
QUIGLEY            3.72453600
PERNAL             3.26053900
WILLIAMS           3.17331100
GRAHAM             0.94479700
```

Solutions to exercise 5

1.

```
Q.STAFF | ID      | NAME       |
--------+---------+-----------|
        | _MID    | _MNM       |

Q.ORG | DEPTNAME       | MANAGER | LOCATION      |               |
------+----------------+---------+---------------+--------------|
      | P.             | _MID    | P.            | P. _MNM       |
```

```
DEPTNAME        LOCATION       NAME
--------------  -------------  ---------
MID ATLANTIC    WASHINGTON     SANDERS
SOUTH ATLANTIC  ATLANTA        MARENGHI
NEW ENGLAND     BOSTON         HANES
GREAT LAKES     CHICAGO        PLOTZ
PLAINS          DALLAS         FRAYE
HEAD OFFICE     NEW YORK       MOLINARE
PACIFIC         SAN FRANCISCO  LEA
MOUNTAIN        DENVER         QUILL
```

2.

```
Q.STAFF | ID      | NAME       |
--------+---------+-----------|
        | _MID    | _MNM       |

Q.ORG | DEPTNAME    | MANAGER | DIVISION | LOCATION  |          |
------+------------+---------+----------+-----------+----------|
      | P.          | _MID    | EASTERN  | P.        | P. _MNM  |
```

```
DEPTNAME        LOCATION       NAME
--------------  -------------  ---------
MID ATLANTIC    WASHINGTON     SANDERS
SOUTH ATLANTIC  ATLANTA        MARENGHI
NEW ENGLAND     BOSTON         HANES
```

3.

```
Q.STAFF | ID      | NAME       | YEARS  |
--------+---------+-----------+--------|
        | _MID    | _MNM       | >=10   |


Q.ORG | DEPTNAME   | MANAGER | DIVISION | LOCATION  |           |
------+-----------+---------+----------+-----------+-----------|
      | P.         | _MID    | EASTERN  | P.        | P. _MNM   |
```

```
DEPTNAME        LOCATION       NAME
--------------  -------------  ---------
NEW ENGLAND     BOSTON         HANES
```

Solutions to exercise 6

1. To copy Q.STAFF as MYTABLE, enter:

   ```
   DISPLAY Q.STAFF
   SAVE DATA AS MYTABLE
   ```

2.

```
MYTABLE | NAME       | DEPT     | JOB    | YEARS   | SALARY      |
--------+-----------+---------+-------+--------+------------|
        | U.RAMOTH   | 66       | MGR    | U. 7    | U.18238.50  |
```

After you have run the previous step:

```
MYTABLE | ID   | NAME     | DEPT | JOB | YEARS | SALARY | COMM |
--------+-----+---------+------+-----+-------+--------+------|
P.      |      | RAMOTH   |      |     |       |        |      |
```

```
    ID  NAME         DEPT  JOB     YEARS      SALARY        COMM
------  ---------  ------  -----  ------  ----------  ----------
   270  RAMOTH         66  MGR         7    18238.50           -
```

3.

```
MYTABLE | JOB    | YEARS   | SALARY      | SALARY           |
--------+-------+--------+------------+-----------------|
        | CLERK | > 5     | _S          | U. _S * 1.1      |
```

To retrieve rows to audit your result:

```
MYTABLE | ID | NAME | DEPT | JOB    | YEARS | SALARY | COMM |
--------+----+------+------+-------+-------+--------+------|
P.      |    |      |      | CLERK |       |        |      |
```

```
    ID  NAME         DEPT  JOB     YEARS      SALARY        COMM
------  ---------  ------  -----  ------  ----------  ----------
    80  JAMES          20  CLERK       -    13504.60      128.20
   110  NGAN           15  CLERK       5    12508.20      206.60
   120  NAUGHTON       38  CLERK       -    12954.75      180.00
   130  YAMAGUCHI      42  CLERK       6    11556.49       75.60
   170  KERMISCH       15  CLERK       4    12258.50      110.10
   180  ABRAHAMS       38  CLERK       3    12009.75      236.50
   190  SNEIDER        20  CLERK       8    15678.02      126.50
   200  SCOUTTEN       42  CLERK       -    11508.60       84.20
   230  LUNDQUIST      51  CLERK       3    13369.80      189.65
   250  WHEELER        51  CLERK       6    15906.00      513.30
   330  BURKE          66  CLERK       1    10988.00       55.50
   350  GAFNEY         84  CLERK       5    13030.50      188.00
```

4.

```
MYTABLE | ID   | NAME   | DEPT | JOB    | YEARS | SALARY | COMM |
--------+-----+--------+------+-------+-------+--------+------|
I.      | 275 | ROGERS | 66   | SALES | NULL  | 14000  | NULL |
```

Instead of 14000 you can use 14000.00, but not 14,000 or 14,000.00 (commas are not valid numeric input characters).

You can retrieve the row with this query:

```
MYTABLE | ID   | NAME | DEPT | JOB   | YEARS | SALARY | COMM |
--------+------+------+------+-------+-------+--------+------|
P.      | 275 |       |      |       |       |        |      |
```

```
    ID  NAME         DEPT  JOB     YEARS      SALARY        COMM
------  ---------  ------  -----  ------  ----------  ----------
   275  ROGERS         66  SALES       -    14000.00           -
```

5.

```
MYTABLE | DEPT   | JOB   |
--------+--------+-------|
D.      | 66     | SALES |
```

Appendix B. QMF functions that require specific support

Table 16. These functions require the support of specific database management systems.

Function Supported	DB2 UDB for z/OS	workstation database servers	DB2 Server for VSE or VM
Length of query statement	32,765	32,765	8,192
Number of columns in SELECT statement	750	255	255
Import single-precision floating point numbers	X		X
Long fields with LIKE statement	X		X
Database synonyms	X		X
Database aliases for tables or views	X	X	
SAVE=IMMEDIATE option available in Table Editor (Supports CURSOR HOLD	X	X	
Distributed Unit of Work (three-part names)	X		
Remote Unit of Work	X	X	on VSE, requires Version 3 Release 4

QMF functions not available in CICS

The following QMF and QMF-related functions are not available in the CICS-VSE or CICS- z/OS environment.

- Command interface
- EDIT PROC
- EDIT QUERY
- Document interface
- BATCH application
- Canceling transactions
- EXTRACT
- ISPF
- DPRE
- Report calculations
- External variables
- LAYOUT application

- Conditional formatting
- Column definition
- Procedures with logic

Appendix C. The DB2 QMF High Performance Option

The DB2 QMF High Performance Option (HPO) is a separately orderable feature of QMF that includes two components:

- QMF HPO/Manager
- QMF HPO/Compiler

This appendix provides a brief overview of the QMF HPO components.

For more detailed information about DB2 QMF HPO, see *DB2 QMF High Performance Option User's Guide*. You can also contact your IBM representative or, in the U.S., call 1-800-879–2755.

QMF HPO/Manager

The QMF HPO/Manager consists of a group of functions that improves governing and object management capabilities, including a preemptive governor to analyze QMF queries. The governing capabilities allow you to establish controls that protect production applications, while delivering "on demand" information. Many governing parameters are included, such as time of day, day of the week, maximum number of rows to fetch, allowing and disallowing SQL verbs and QMF commands, and controlling resource consumption based on the use of QMF commands and SQL statements.

QMF HPO/Compiler

The QMF HPO/Compiler lets you convert queries and reports into efficient programs or stored procedures in OS/VS COBOL or COBOL II. This reduces:

- CPU consumption
- DB2 catalog contention
- DB2 Optimizer overhead
- Security concerns because converted programs use static Structured Query Language (SQL) in place of dynamic SQL

Appendix D. Notices

This information was developed for products and services offered in the U.S.A. IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10594-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106-0032, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
J46A/G4
555 Bailey Avenue
San Jose, CA 95141-1003
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurement may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

Trademarks

The following terms are trademarks of the IBM Corporation in the United States or other countries or both:

AIX
C/370
CICS
COBOL/370
DataJoiner
DB2
DB2 Information Integrator
DB2 Universal Database
Distributed Relational Database Architecture
DRDA
GDDM
IBM
IBMLink
IMS
iSeries
MVS
OS/390
Parallel Sysplex
PL/I
QMF
RACF
S/390
SQL/DS
VM/ESA
VSE/ESA
VTAM
WebSphere
z/OS
zSeries

Java or all Java-based trademarks and logos, and Solaris are trademarks of Sun Microsystems, Inc. in the United States, other counrtries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, and service names may be trademarks or service marks of others.

Glossary of Terms and Acronyms

This glossary defines terms as they are used throughout the QMF library.

abend. The abnormal termination of a task.

ABENDx. The keyword for an abend problem.

Advanced Peer-to-Peer Networking®. A distributed network and session control architecture that allows networked computers to communicate dynamically as equals. Compare with Advanced Program-to-Program Communication (APPC). An implementation of the SNA synchronous data link control LU 6.2 protocol that allows interconnected systems to communicate and share the processing of programs.

Advanced Program-to-Program Communication (APPC). An implementation of the SNA synchronous data link control LU 6.2 protocol that allows interconnected systems to communicate and share the processing of programs.

aggregation function. Any of a group of functions that summarizes data in a column. They are requested with these usage codes on the form panels: AVERAGE, CALC, COUNT, FIRST, LAST, MAXIMUM, MINIMUM, STDEV, SUM, CSUM, PCT, CPCT, TPCT, TCPCT.

aggregation variable. An aggregation function that is placed in a report using either the FORM.BREAK, FORM.CALC, FORM.DETAIL, or FORM.FINAL panels. Its value appears as part of the break footing, detail block text, or final text when the report is produced.

alias. In DB2 UDB for OS/390®, an alternate name that can be used in SQL statements to refer to a table or view in the same or a remote DB2 UDB for OS/390 subsystem. In OS/2®, an alternate name used to identify a object, a database, or a network resource such as an LU. In QMF, a locally defined name used to access a QMF table or view stored on a local or remote DB2 UDB for OS/390 subsystem.

APAR. Authorized Program Analysis Report.

APPC. Advanced Program-to-Program Communication

application. A program written by QMF users that extends the capabilities of QMF without modifying the QMF licensed program. Started from a QMF session by issuing a RUN command for a QMF procedure, an installation-defined command, or a CMS or TSO command that invokes an EXEC or CLIST, respectively.

application requester. (1) A facility that accepts a database request from an application process and passes it to an application server. (2) In DRDA®, the source of a request to a remote relational database management system.

The application requester is the DBMS code that handles the QMF end of the distributed connection. The local DB2 UDB for OS/390 subsystem to which QMF attaches is known as the application requester for QMF, because DB2 UDB for OS/390's application requester is installed within the local database manager. Therefore, an entire DB2 UDB for OS/390 subsystem (including data) is associated with the application requester, but the SQL statements are processed at the current location. This subsystem is called the “local DB2 UDB for OS/390”.

With DB2 for VM and VSE the application requester runs in the same virtual machine as QMF; that is, no database is inherently associated with the DB2 for VM and VSE application requester.

application server. The target of a request from an application requester. (1) The local or remote database manager to which the application process is connected. The application server executes at the system containing the desired data. (2) In DRDA, the target of a request from an application requester. With DB2 UDB for OS/390, the application server is part of a full DB2 UDB for OS/390 subsystem.

With DB2 for VM and VSE, the application server is part of a DB2 for VM and VSE database machine.

application-support command. A QMF command that can be used within an application program to exchange information between the application program and QMF. These commands include INTERACT, MESSAGE, STATE, and QMF.

area separator. The barrier that separates the fixed area of a displayed report from the remainder of the report.

argument. An independent variable.

base QMF environment. The English-language environment of QMF, established when QMF is installed. Any other language environment is established after installation.

batch QMF session. A QMF session running in the background. Begins when a specified QMF procedure is invoked and ends when the procedure ends. During a background QMF session, no user interaction and panel display interaction are allowed.

bind. In DRDA, the process by which the SQL statements in an application program are made known to a database management system over application support protocol (and database support protocol) flows. During a bind, output from a precompiler or preprocessor is converted to a control structure called a package. In addition, access paths to the referenced data are selected and some authorization checking is performed. (Optionally in DB2 UDB for OS/390, the output may be an application plan.)

built-in function. Generic term for scalar function or column function. Can also be “function.”

calculation variable. CALCid is a special variable for forms that contains a user-defined calculated value. CALCid is defined on the FORM.CALC panel.

callable interface. A programming interface that provides access to QMF services. An application can access these services even when the application is running outside of a QMF session. Contrast with command interface.

chart. A graphic display of information in a report.

CICS®. Customer Information Control System.

client. A functional unit that receives shared services from a server.

CMS. Conversational Monitor System.

column. A vertical set of tabular data. It has a particular data type (for example, character or numeric) and a name. The values in a column all have the same data characteristics.

column function. An operation that is applied once to all values in a column, returns a single value as a result, and is expressed in the form of a function name followed by one or more arguments enclosed in parentheses.

column heading. An alternative to the column name that a user can specify on a form. Not saved in the database, as are the column name and label.

column label. An alternative descriptor for a column of data that is saved in the database. When used, column labels appear by default on the form, but they can be changed by users.

column wrapping. Formatting values in a report so that they occupy several lines within a column. Often used when a column contains values whose length exceeds the column width.

command interface. An interface for running QMF commands. The QMF commands can only be issued from within an active QMF session. Contrast with callable interface.

command synonym. The verb or verb/object part of an installation-defined command. Users enter this for the command, followed by whatever other information is needed.

command synonym table. A table each of whose rows describes an installation-defined command. Each user can be assigned one of these tables.

commit. The process that makes a data change permanent. When a commit occurs, data locks are freed enabling other applications to reference the just-committed data. See also “rollback”.

concatenation. The combination of two strings into a single string by appending the second to the first.

connectivity. The enabling of different systems to communicate with each other. For example, connectivity between a DB2 UDB for OS/390 application requester and a DB2 for VM and VSE application server enables a DB2 UDB for OS/390 user to request data from a DB2 for VM and VSE database.

conversation. A logical connection between two programs over an LU 6.2 session that allows them to communicate with each other while processing a transaction.

correlation name. An alias for a table name, specified in the FROM clause of a SELECT query. When concatenated with a column name, it identifies the table to which the column belongs.

CP. The Control Program for VM.

CSECT. Control section.

current location. The application server to which the QMF session is currently connected. Except for connection-type statements, such as CONNECT (which are handled by the application requester), this server processes all the SQL statements. When initializing QMF, the current location is indicated by the DSQSDBNM startup program parameter. (If that parameter is not specified, the local DB2 UDB for OS/390 subsystem

current object. An object in temporary storage currently displayed. Contrast with saved object.

Customer Information Control System (CICS). An IBM® licensed program that enables transactions entered at remote terminals to be processed concurrently by user-written application programs. It includes facilities for building, using, and maintaining databases.

DATA. An object in temporary storage that contains the information returned by a retrieval query. Information represented by alphanumeric characters contained in tables and formatted in reports.

database. A collection of data with a given structure for accepting, storing, and providing on demand data for multiple users. In DB2 UDB for OS/390, a created object that contains table spaces and index spaces. In DB2 for VM and VSE, a collection of tables, indexes, and supporting information (such as control information and data recovery information) maintained by the system. In OS/2, a collection of information, such as tables, views, and indexes.

database administrator. The person who controls the content of and access to a database.

database management system. A computer-based system for defining, creating, manipulating, controlling, managing, and using databases. The database management system also has transaction management and data recovery facilities to protect data integrity.

database manager. A program used to create and maintain a database and to communicate with programs requiring access to the database.

database server. (1) In DRDA, the target of a request received from an application server (2) In OS/2, a workstations that provides database services for its local database to database clients.

date. Designates a day, month, and year (a three-part value).

date/time default formats. Date and time formats specified by a database manager installation option. They can be the EUR, ISO, JIS, USA, or LOC (LOCAL) formats.

date/time data. The data in a table column with a DATE, TIME, or TIMESTAMP data type.

DB2 UDB for OS/390. DB2 Universal Database for OS/390™ (an IBM relational database management system).

DB2 for AIX®. DATABASE2 for AIX. The database manager for QMF's relational data.

DBCS. Double-byte character set.

DBMS. Database management system.

default form. The form created by QMF when a query is run. The default form is not created if a saved form is run with the query.

destination control table (DCT). In CICS, a table containing a definition for each transient data queue.

detail block text. The text in the body of the report associated with a particular row of data.

detail heading text. The text in the heading of a report. Whether or not headings will be printed is specified in FORM.DETAIL.

dialog panel. A panel that overlays part of a Prompted Query primary panel and extends the dialog that helps build a query.

distributed data. Data that is stored in more than one system in a network, and is available to remote users and application programs.

distributed database. A database that appears to users as a logical whole, locally accessible, but is comprised of databases in multiple locations.

distributed relational database. A distributed database where all data is stored according to the relational model.

Distributed Relational Database Architecture™. A connection protocol for distributed relational database processing that is used by IBM and vendor relational database products.

distributed unit of work. A method of accessing distributed relational data in which users or applications can, within a single unit of work, submit SQL statements to multiple relational database management systems, but no more than one RDBMS per SQL statement.

DB2 UDB for OS/390 introduced a limited form of distributed unit of work support in its V2R2 called system-directed access, which QMF supports.

DOC. The keyword for a document problem.

double-byte character. An entity that requires two character bytes.

double-byte character set (DBCS). A set of characters in which each character is represented by two bytes. Languages such as Japanese, Chinese, and Korean, which contain more symbols that can be represented by 256 code points, require double-byte character sets. Because each character requires two bytes, the typing, display, and printing of DBCS characters requires hardware and programs that support DBCS. Contrast with single-byte character set.

DRDA. Distributed Relational Database Architecture.

duration. An amount of time expressed as a number followed by one of seven keywords: YEARS, MONTHS, DAYS, HOURS, MINUTES, SECONDS, MICROSECONDS.

EBCDIC. Extended Binary-Coded Decimal Interchange Code.

echo area. The part of the Prompted Query primary panel in which a prompted query is built.

EUR (European) format. A format that represents date and time values as follows:

- Date: dd.mm.yyyy
- Time: hh.mm.ss

extended syntax. QMF command syntax that is used by the QMF callable interface; this syntax defines variables that are stored in the storage acquired by the callable interface application and shared with QMF

example element. A symbol for a value to be used in a calculation or a condition in a QBE query.

example table. The framework of a QBE query.

fixed area. That part of a report that contains fixed columns.

fixed columns. The columns of a report that remain in place when the user scrolls horizontally. On multiple-page, printed reports, these columns are repeated on the left side of each page.

form. An object that contains the specifications for printing or displaying a report or chart. A form in temporary storage has the name of FORM.

function key table. A table containing function key definitions for one or more QMF panels, along with text describing the keys. Each user can be assigned one of these tables.

gateway. A functional unit that connects two computer networks of different network architectures. A gateway connects networks or systems of different architectures, as opposed to a bridge, which connects networks or systems with the same or similar architectures.

GDDM®. Graphical Data Display Manager.

global variable. A variable that, once set, can be used for an entire QMF session. A global variable can be used in a procedure, query, or form. Contrast with run-time variable.

Graphical Data Display Manager. A group of routines that allows pictures to be defined and displayed procedurally through function routines that correspond to graphic primitives.

grouped row. A row of data in a QBE target or example table that is summarized either by a G. or a built-in function.

HELP. Additional information about an error message, a QMF panel, or a QMF command and its options.

host. A mainframe or mid-size processor that provides services in a network to a workstation.

HTML. Hypertext Markup Language. A standardized markup language for documents displayed on the World Wide Web.

ICU. Interactive Chart Utility.

INCORROUT. The keyword for incorrect output.

index. A collection of data about the locations of records in a table, allowing rapid access to a record with a given key.

initial procedure. A QMF procedure specified by the DSQSRUN parameter on the QMF start command which is executed immediately after QMF is invoked.

initialization program. A program that sets QMF program parameters. This program is specified by DSQSCMD in the callable interface. The default program for interactive QMF is DSQSCMD*n*, where *n* is the qualifier for the presiding language ('E' for English).

installation-defined command. A command created by an installation. QMF will process it as one of its own commands or as a combination of its commands.

installation-defined format. Date and time formats, also referred to as LOCAL formats, that are defined (or built) by the installation.

interactive execution. Execution of a QMF command in which any dialog that should take place between the user and QMF during the command's execution actually does take place.

interactive session. Any QMF session in which the user and QMF can interact. Could be started by another interactive session by using the QMF INTERACT command.

interactive switch. A conceptual switch which, when on, enables an application program to run QMF commands interactively.

invocation CLIST or EXEC. A program that invokes (starts) QMF.

ISO (International Standards Organization) format. A format that represents date and time values as follows:

- Date: yyyy-mm-dd
- Time: hh.mm.ss

ISPF. Interactive System Productivity Facility.

IXF. Integration Exchange Format: A protocol for transferring tabular data among various software products.

JCL. Job control language for OS/390®.

job control. In VSE, a program called into storage to prepare each job or job step to be run. Some of its functions are to assign I/O devices to symbolic names, set switches for program use, log (or print) job control statements, and fetch the first phase of each job step.

JIS (Japanese Industrial Standard) format. A format that represents date and time values as follows:

- Date: yyyy-mm-dd
- Time: hh:mm:ss

join. A relational operation that allows retrieval of data from two or more tables based on matching columns that contain values of the same data type.

keyword parameter. An element of a QMF command consisting of a keyword and an assigned value.

like. Pertaining to two or more similar or identical IBM operating environments. For example, like distribution is distribution between two DB2 UDB for OS/390's with compatible server attribute levels. Contrast with "unlike".

literal. In programming languages, a lexical unit that directly represents a value. A character string whose value is given by the characters themselves.

linear procedure. Any procedure *not* beginning with a REXX comment. A linear procedure can contain QMF commands, comments, blank lines, RUN commands, and substitution variables. See also "procedure with logic."

linear syntax. QMF command syntax that is entered in one statement of a program or procedure, or that can be entered on the QMF command line.

line wrapping. Formatting table rows in a report so they occupy several lines. The row of column names and each row of column values are split into as many lines as are required by the line length of the report.

local. Pertaining to the relational database, data, or file that resides in the user's processor. See also "local DB2 UDB for OS/390", and contrast with *remote*.

local area network (LAN). (1) Two or more processors connected for local resource sharing (2) A network within a limited geographic area, such as a single office building, warehouse, or campus.

local data. Data that is maintained by the subsystem that is attempting to access the data. Contrast with remote data.

local DB2 UDB for OS/390. With DB2 UDB for OS/390, the application requester is part of a DB2 UDB for OS/390 subsystem that is running in the same MVS™ system as QMF. Therefore, an entire DB2 UDB for OS/390 subsystem (including data) is associated with the application requester, but the SQL statements are processed at the current location. This subsystem is where the QMF plan is bound.

When QMF runs in TSO, this subsystem is specified using DSQSSUBS startup program parameter. When QMF runs in CICS, this subsystem is identified in the Resource Control Table (RCT). The local DB2 UDB for OS/390 is the subsystem ID of the DB2 UDB for OS/390 that was started in the CICS region.

location. A specific relational database management system in a distributed relational database system. Each DB2 UDB for OS/390 subsystem is considered to be a location.

logical unit (LU). A port through which an end user accesses the SNA network to communicate with another end user and through which the end user accesses the functions provided by system services control points.

Logical Unit type 6.2 (LU 6.2). The SNA logical unit type that supports general communication between programs in a distributed processing environment.

LU. Logical unit.

LU 6.2. Logical Unit type 6.2.

LOOP. The keyword for an endless-loop problem.

MSGx. The keyword for a message problem.

Multiple Virtual Storage. Implies the MVS/ESA™ product

MVS/ESA. Multiple Virtual Storage/Enterprise System Architecture (IBM operating system).

NCP. Network Control Program.

Network Control Program (NCP). An IBM licensed program that provides communication controller support for single-domain, multiple-domain, and interconnected network capability.

NLF. National Language Feature. Any of several optional features available with QMF that lets the user select a language other than US English.

NLS. National Language Support.

node. In SNA, an end point of a link or a junction common to two or more links in a network. Nodes can be distributed to host processors, communication controllers, cluster controllers, or terminals. Nodes can vary in routing and other functional capabilities.

null. A special value used when there is no value for a given column in a row. *Null* is not the same as zero.

null value. See *null*.

object. A QMF query, form, procedure, profile, report, chart, data, or table. The report, chart, and data objects exist only in temporary storage; they cannot be saved in a database. The table object exists only in a database.

object name. A character string that identifies an object owned by a QMF user. The character string can be a maximum of 18 bytes long and must begin with an alphabetic character. The term "object name" does not include the "owner name" prefix. Users can access other user's objects only if authorized.

object panel. A QMF panel that can appear online after the execution of one QMF command and before the execution of another. Such panels include the home, report, and chart panels, and all the panels that display a QMF object. They do not include the list, help, prompt, and status panels.

online execution. The execution of a command from an object panel or by pressing a function key.

owner name. The authorization id of the user who creates a given object.

package. The control structure produced when the SQL statements in an application program are bound to a relational database management system. The database management system uses the control structure to process SQL statements encountered during statement execution.

panel. A particular arrangement of information, grouped together for presentation in a window. A panel can contain informational text, entry fields, options the user can choose from, or a mixture of these.

parameter. An element of a QMF command. This term is used generically in QMF documentation to reference a *keyword parameter* or a *positional parameter*.

partner logical unit. In SNA, the remote system in a session.

PERFM. The keyword for a performance problem.

permanent storage. The database where all tables and QMF objects are stored.

plan. A form of package where the SQL statements of several programs are collected together during bind to create a plan.

positional parameter. An element of a QMF command that must be placed in a certain position within the command.

primary panel. The main Prompted Query panel containing your query.

primary QMF session. An interactive session begun from outside QMF Within this session, other sessions can be started by using the INTERACT command.

procedure. An object that contains QMF commands. It can be run with a single RUN command. A procedure in temporary storage has the name of PROC. See also “linear procedure” and “procedure with logic.”

procedure termination switch. A conceptual switch that a QMF MESSAGE command can turn on. While on, every QMF procedure to which control returns terminates immediately.

procedure with logic. Any QMF procedure beginning with a REXX comment. In a procedure with logic, you can perform conditional logic, make calculations, build strings, and pass commands back to the host environment. See also “linear procedure.”

profile. An object that contains information about the characteristics of the user’s session. A stored profile is a profile that has been saved in permanent storage. A profile in temporary storage has the name PROFILE. There can be only one profile for each user.

prompt panel. A panel that is displayed after an incomplete or incorrect QMF command has been issued.

Prompted Query. A query built in accordance with the user’s responses to a set of dialog panels.

protocol. The rules governing the functions of a communication system that must be followed if communication is to be achieved.

PSW. Program status word.

PTF. Program temporary fix.

QBE (Query-By-Example). A language used to write queries graphically. For more information see *Using QMF*

QMF administrative authority. At minimum, insert or delete priviledge for the Q.PROFILES control table.

QMF administrator. A QMF user with QMF administrative authority.

QMF command. Refers to any command that is part of the QMF language. Does **not** include installation-defined commands.

QMF session. All interactions between the user and QMF from the time the user invokes QMF until the EXIT command is issued.

qualifier. When referring to a QMF object, the part of the name that identifies the owner. When referring to a TSO data set, any part of the name that is separated from the rest of the name by periods. For example, ‘TCK’, ‘XYZ’, and ‘QUERY’ are all qualifiers in the data set name ‘TCK.XYZ.QUERY’.

query. An SQL or QBE statement, or a statement built from prompting, that performs data inquiries or manipulations. A saved query is an SQL query, QBE query, or Prompted Query that has been saved in a database. A query in temporary storage, has the name QUERY.

RDBMS. Relational database management system

relational database. A database perceived by its users as a collection of tables.

relational database management system (RDBMS). A computer-based system for defining, creating, manipulating, controlling, managing, and using relational databases.

remote. Pertaining to a relational DBMS other than the local relational DBMS.

remote data. Data that is maintained by a subsystem other than the subsystem that is attempting to access the data. Contrast with local data.

remote data access. Methods of retrieving data from remote locations. The two remote data access functions used by QMF are *remote unit of work* and DB2 UDB for OS/390-only distributed unit of work, which is called *system-directed access*.

remote unit of work. (1) The form of SQL distributed processing where the application is on a system different from the relational database and a single application server services all remote unit of work requests within a single logical unit of work. (2) A unit of work that allows for the remote preparation and execution of SQL statements.

report. The formatted data produced when a query is issued to retrieve data or a DISPLAY command is entered for a table or view.

REXX. Restructured extended executor.

rollback. The process that removes uncommitted database changes made by one application or user. When a rollback occurs, locks are freed and the state of the resource being changed is returned to its state at the last commit, rollback, or initiation. See also *commit*.

row. A horizontal set of tabular data.

row operator area. The leftmost column of a QBE target or example table.

run-time variable. A variable in a procedure or query whose value is specified by the user when the procedure or query is run. The value of a run-time variable is only available in the current procedure or query. Contrast with global variable.

sample tables. The tables that are shipped with QMF. Data in the sample tables is used to help new QMF users learn the product.

saved object. An object that has been saved in the database. Contrast with current object.

SBCS. Single-byte character set.

scalar. A value in a column or the value of a literal or an expression involving other scalars.

scalar function. An operation that produces a single value from another value and is expressed in the form of a function name followed by a list of arguments enclosed in parentheses.

screen. The physical surface of a display device upon which information is presented to the user.

scrollable area. The view of a displayed object that can be moved up, down, left, and right.

server. A functional unit that provides shared services to workstations over a network.

session. All interactions between the user and QMF from the time the user logs on until the user logs off.

single-byte character. A character whose internal representation consists of one byte. The letters of the Latin alphabet are examples of single-byte characters.

SNA. Systems Network Architecture.

SNAP dump. A dynamic dump of the contents of one or more storage areas that QMF generates during an abend.

sort priority. A specification in a retrieval query that causes the sorted values in one retrieved column to determine the sorting of values in another retrieved column.

SQL. Structured Query Language.

SQLCA. Structured Query Language Communication Area.

SSF. Software Support Facility. An IBM online database that allows for storage and retrieval of information about all current APARs and PTFs.

stored object. An object that has been saved in permanent storage. Contrast with current object.

string. A set of consecutive items of a similar type; for example, a character string.

Structured Query Language. A language used to communicate with DB2 UDB for OS/390 and DB2 for VSE or VM. Used to write queries in descriptive phrases.

subquery. A complete SQL query that appears in a WHERE or HAVING clause of another query (the main query or a higher-level subquery).

substitution variable. (1) A variable in a procedure or query whose value is specified either by a global variable or by a run-time variable. (2) A variable in a form whose value is specified by a global variable.

substring. The part of a string whose beginning and length are specified in the SUBSTR function.

System Log (SYSLOG). A data set or file in which job-related information, operational data, descriptions of unusual occurrences, commands, and messages to and from the operator may be stored.

Systems Network Architecture. The description of the logical structure, formats, protocols, and operational sequences for transmitting information units through and controlling the configuration and operation of networks.

table. A named collection of data under the control of the relational database manager. A table consists of a fixed number of rows and columns.

Table Editor. The QMF interactive editor that lets authorized users make changes to a database without having to write a query.

table name area. The leftmost column of a QBE example table.

tabular data. The data in columns. The content and the form of the data is specified on FORM.MAIN and FORM.COLUMNS.

target table. An empty table in which example elements are used to combine columns, combine rows, or include constant values in a report.

temporary storage. An area where the query, form, procedure, profile, report, chart, and data objects in current use are stored. All but the data object can be displayed.

temporary storage queue. In CICS, a temporary storage area used for transfer of objects between QMF and an application or a system service.

time. Designates a time of day in hours and minutes and possibly seconds (a two- or three-part value).

thread. The DB2 UDB for OS/390 structure that describes an application's connection, traces its progress, provides resource function processing capability, and delimits its accessibility to DB2 UDB for OS/390 resources and services. Most DB2 UDB for OS/390 functions execute under a thread structure.

three-part name. A fully-qualified name of a table or view, consisting of a location name, owner ID, and object name. When supported by the application server (that is, DB2 UDB for OS/390), a three-part name can be used in an SQL statement to retrieve or update the specified table or view at the specified location.

timestamp. A date and a time, and possibly a number of microseconds (a six- or seven-part value).

TP. Transaction Program

TPN. Transaction program name

transaction. The work that occurs between 'Begin Unit of Work' and 'Commit' or 'Rollback'.

transaction program. A program that processes transactions in an SNA network. There are two kinds of transactions programs: application transaction programs and service transaction programs.

transaction program name. The name by which each program participating in an LU 6.2 conversation is known. Normally, the initiator of a connection identifies the name of the program it wants to connect to at the other LU. When used in conjunction with an LU name, it identifies a specific transaction program in the network.

transient data queue. In CICS, a storage area, whose name is defined in the Destination Control Table (DCT), where objects are stored for subsequent internal or external processing.

TSO. Time Sharing Option.

two-phase commit. A protocol used in distributed unit of work to ensure that participating relational database management systems commit or roll back a unit of work consistently.

unit of work. (1) A recoverable sequence of operations within an application process. At any time, an application process is a single unit of work, but the life of an application process may involve many units of work as a result of commit or rollback operations. (2) In DRDA, a sequence of SQL commands that the database manager treats as a single entity. The database manager ensures the consistency of data by verifying that either all the data changes made during a unit of work are performed or none of them are performed.

unlike. Refers to two or more different IBM operating environments. For example, unlike distribution is distribution between DB2 for VM and VSE and DB2 UDB for OS/390. Contrast with *like*.

unnamed column. An empty column added to an example table. Like a target table, it is used to combine columns, combine rows, or include constant values in a report.

USA (United States of America) format. A format that represents date and time values as follows:

- Date: mm/dd/yyyy
- Time: hh:mm xM

value. A data element with an assigned row and column in a table.

variation. A data formatting definition specified on a FORM.DETAIL panel that conditionally can be used to format a report or part of a report.

view. An alternative representation of data from one or more tables. It can include all or some of the columns contained in the table or tables on which it is defined. (2) The entity or entities that define the scope of the data to be searched for a query.

Virtual Storage Extended. An operating system that is an extension of Disk Operating System/ Virtual Storage. A VSE consists of (1) VSE/Advanced Functions support and (2) any IBM-supplied and user-written programs that are required to meet the data processing needs of a user. VSE and the hardware it controls form a complete computing system.

VM. Virtual Machine (IBM operating system). The generic term for the VM/ESA® environment.

VSE. Virtual Storage Extended (IBM operating system). The generic term for the VSE/ESA™ environment.

WAIT. The keyword for an endless-wait-state problem.

window. A rectangular portion of the screen in which all or a portion of a panel is displayed. A window can be smaller than or equal to the size of the screen.

Workstation Database Server. The IBM family of DRDA database products on the UNIX® and Intel platforms (such as DB2 Universal Database (UDB), DB2 Common Server, DB2 Parallel Edition, and DataJoiner®.)

wrapping. See “column wrapping” and “line wrapping”.

Bibliography

The following lists do not include all the books for a particular library. To get copies of any of these books, or to get more information about a particular library, contact your IBM representative.

CICS publications

CICS Transaction Server for OS390
CICS User's Handbook
CICS Application Programming Reference
CICS Application Programming Guide
CICS DB2 Guide
CICS Resource Definition Guide
CICS Problem Determination Guide
CICS System Definition Guide
CICS Intercommunication Guide
CICS Performance Guide

CICS Transaction Server for VSE/ESA
User's Handbook
Application Programming Reference
Application Programming Guide
Resource Definition Guide
Problem Determination Guide
System Definition Guide
Intercommunication Guide
Performance Guide

COBOL publications

COBOL for VSE/ESA Language Reference
COBOL for VSE/ESA Programming Guide

DB2 Universal Database for z/OS publications

DB2 Universal Database for z/OS
Installation Guide
Administration Guide
SQL Reference
Command Reference
Application Programming and SQL Guide
Messages and Codes

Utility Guide and Reference
Reference for Remote DRDA Requesters and Servers

IBM DB2 Server for VSE & VM
Diagnosis Guide and Reference
DB2 Server for VSE Messages and Codes
DB2 Server for VM Messages and Codes
DB2 Server for VSE System Administration
DB2 Server for VM System Administration
DB2 Server for VSE & VM Operation
DB2 Server for VSE & VM SQL Reference
DB2 Server for VSE & VM Application Programming
DB2 Server for VSE & VM Interactive SQL Guide and Reference
DB2 Server for VSE & VM Database Services Utility
DB2 Server for VSE & VM Performance Tuning Handbook

DB2 Universal Database for iSeries
SQL Reference
SQL Programming with Host Languages

DB2 Universal Database
Command Reference
SQL Reference
Message Reference

DB2 DataJoiner
DataJoiner Application Programming and SQL Reference Supplement

Document Composition Facility (DCF) publications

DCF and DLF General Information

Distributed Relational Database Architecture (DRDA) publications

Every Manager's Guide
Connectivity Guide

Graphical Data Display Manager (GDDM) publications

GDDM General Information
GDDM Base Application Programming Reference
GDDM User's Guide
GDDM/VSE Program Directory
GDDM Messages
GDDM System Customization and Administration

HIgh Level Assembler (HLASM) publications

High-Level Assembler for MVS, VM and VSE Programming Guide
High-Level Assembler for MVS, VM and VSE Language Reference

Interactive System Productivity Facility (ISPF) publications

OS/390

ISPF Planning and Customizing
ISPF Dialog Developer's Guide and Reference

VM

ISPF for VM Dialog Management Guide and Reference

OS/390 publications

JCL

OS/390 MVS JCL Reference
OS/390 MVS JCL User's Guide

Pageable Link Pack Area (PLPA)

OS/390 Extended Architecture Initialization and Tuning
OS/390 SPL: Initialization and Tuning

VSAM

OS/390 VSAM Administration Guide
OS/390 VSAM Catalog Administration Access Method Services

TSO/E

TSO/E Primer
TSO/E User's Guide

SMP/E

OS/390 System Modification Program Extended Messages and Codes
OS/390 System Modification Program Extended Reference
OS/390 System Modification Program Extended User's Guide

OS PL/I publications

OS PL/I Programming Language Reference
OS PL/I Programming Guide

REXX publications

OS/390 environment
TSO/E REXX/MVS User's Guide
TSO/E REXX/MVS Reference

VM environment
System Product Interpreter Reference
REXX/VM User's Guide

VM/ESA publications

VM/ESA Planning and Administration
VM/ESA Command Reference

VSE/ESA publications

Planning
System Utilities
Guide for Solving Problems

Index

Special characters

&variable parameter 310

A

access
- current location name 242
- OS/390 editor from QMF 278
- QMF from a VM editor 275
- QMF with document interface 275, 278
- remote data 243

accessing data, methods of 3

add
- break heading/footing 140
- break segments and text 139
- columns 228, 300
- conditions
  - in a CONDITIONS box 302
  - to an example table 302
  - using AND 321
- data to long fields 221
- date, time, page number to report 138
- dates and times 100
- dates/times 100
- expressions 302
- information to queries 65
- lines to a query 79
- new column to report 117
- page heading/footing to report 131
- rows 225, 330
  - I (insert) 330
- subtotals to report 128
- target tables 304

addition of dates and times 100

alias
- deleting 214
- for table 214
- for view 214

alias, in place of three-part name 243

ALL keyword 321
- QBE 321

AND keyword 321
- QBE 321

AO keyword 322

application requester 241, 246

application server 245, 246

arithmetic 300, 345
- expressions 345
- overflow 300

ascending order, sorting sequence 48, 322

ASIS formatting type 271

asterisk (*), placement of cursor 316, 318

authorization
- granting on table 229
- revoking on table 230
- to use objects 4

automatic interrupt 64

AVG column function 323

AVG column function in SQL 89

AVG keyword 323

B

BAR chart format 167

batch mode
- procedures
  - errors 205
  - example for MVS 203
  - example for VM 203
  - restrictions 202
  - termination 205
  - using IMPORT/EXPORT commands 204
  - using the QMF EXIT command 204
  - writing 202

BETWEEN keyword 325
- QBE 325

bilingual command 251

BLOB 229

break segments, adding to reports 139

break text, adding to reports 140

C

calculated values
- average 323
- columns of expressions 300, 346
- count 326
- displaying on reports 150
- for groups 329
- maximum 333
- minimum 334
- of expressions 346
- total 342

CANCEL command 63

canceling a command or query 63

case operand for Prompted Query 38

changing
- appearance of a report 23
- column names in queries 65
- date/time values using scalar functions 92
- default chart format 178
- information in queries 65
- row conditions in queries 65
- rows 343
  - U operator 343
- saved query 64
- sort order in queries 65
- table names in queries 65

CHAR
- scalar function 93

CHAR scalar function 93

character
- constants 300
- data
  - in conditions 299
  - in descriptive columns 300
  - with LIKE 332

CHART database object 4

charts 178
- changing
  - data, using QMF forms 173
  - formats, using the ICU 177
  - properties, using the ICU 176
- creating 167, 179
- default type 171
- fixing problems 178
- format
  - changing 177
  - creating 167
  - saving 177
- formats, QMF 167
- location of data
  - on pie charts 169
  - on the X-axis 169
  - rules for specifying 168
- printing 179
- size limits for data 170
- specifying type 171
- ways to display data 167

- CHECK command 162
- CICS
  - and remote unit of work 245
  - restrictions
    - calculated values in reports 149
    - document interface 270
    - external editors 267
    - procedures with logic 245
    - QMF BATCH command 201
    - viewing data in tables 230
- CLIST used with editor 265
- CLOB 229
- CMS
  - exporting objects into 234
  - importing objects from 236
  - note facility 270
  - NOTE facility 270, 274
  - used with QMF document interface 270
  - XEDIT 274
- codes
  - edit 123
  - usage 127
- column
  - functions
    - AVG 89, 323
    - COUNT 89, 326
    - MAX 89, 333
    - MIN 89, 334
    - SUM 89, 342
- column functions 89, 90, 323, 326, 333, 334, 342
- column names
  - changing
    - in queries 65
  - distinguishing between
    - using correlation names 81
    - using qualifiers 80
  - finding 72
- columns
  - adding to reports 117
  - changing
    - alignment of headings and data, on reports 122
    - column spacing 121
    - headings, on reports 119
    - on reports 115
    - order, on reports 119
    - spacing, on reports 121
    - widths, on reports 121
  - column functions 99
  - creating 72
  - creating empty 300
  - edit codes in 123
- columns *(continued)*
  - expressions 345, 346
  - fixed, on reports 132
  - functions
    - nesting within scalar functions 99
  - joining 79
    - in Prompted Query 50
    - multiple 57, 79
  - names 300
  - punctuating 123
  - select
    - using P. 295, 339
  - selecting 71, 295, 339
  - specifying, on reports 132
  - unnamed 300
  - with calculated values 345
- command 6
  - CANCEL 63
  - CHECK 162
  - CONNECT
    - compared to the DSQSDBNM parameter 240
    - from the command line 240
  - CONVERT 312
    - QBE to SQL 312
  - DBCS data 256
  - DELETE 314
    - using QBE 314
  - DISPLAY 309
    - using QBE 309
  - DRAW 226, 304, 314
    - using QBE 304, 314
  - EDIT 268
  - ENLARGE 316
  - FORM.COLUMNS 117
  - governor interrupt 63
  - how to issue 6
  - interrupt 63
  - LAYOUT 134
  - line 209
    - global variables 209
  - LIST 293
    - under QBE 293
  - QBE-specific 312
  - REDUCE 318
  - RUN 310
    - substitution variables 310
  - SHOW FORM 116
- command line 4
- comment
  - in QBE query 314, 318
- COMMENTS box 314, 318
- concatenation, rules for 107
- conditions
  - character data in 299
  - data types in 302
  - DBCS characters 302
  - grouping 77
  - in a CONDITIONS box 302
  - in an example table 302
  - multiple
    - AND 76, 321
    - BETWEEN 324
    - IN 77
    - in Prompted Query 45
    - OR 76, 338
  - negative 335
  - opposite 74
  - quotation marks in 299
  - row 74
  - selection symbols in 75
  - special characters in 302
  - two on one row 321
  - values in a set 331
  - with equalities 347
  - with example elements 297
  - with expressions 303
  - with inequalities 347
  - writing 296, 299
- CONDITIONS box 302, 314, 318
  - deleting 314
  - drawing 315
  - reducing 318
  - restricting data presentation 302
- CONNECT command
  - compared to the DSQSDBNM parameter 240
  - from the command line 240
  - prompt panels for 241
  - user ID 246
- connecting
  - from DB2 to DB2 247
  - from DB2 to SQL/DS 248
  - from SQL/DS to SQL/DS 247
  - general considerations 240
  - QMF CONNECT command prompt panel 241
  - using QMF CONNECT command 240
  - with remote unit of work 240, 241
- constants 300
- conversion functions
  - scalar 91
- conversion scalar functions 91
- CONVERT command 312
  - QBE to SQL 312

converting
QBE to SQL 312
converting queries to SQL 312
copy tables 213, 326
correcting, saved query 64
correlation names
rules for 87
used in a subquery 87
used to distinguish between columns 81
COUNT
column function 89, 326
COUNT column function 89, 326
counting number of values in a column 326
creating
expressions 41
new column in report 41, 72
reports 23
reusable procedures 188
row conditions 20, 43
currency symbol
changing 124
current location
changing 240
DSQAO_CONNECT_LOC 242
procedures, forms, and queries 245
QMF objects 244
QMF's governor exit 242
CURRENT SQLID 245, 246

D

D operator 326
data
deletion 326
entry 330, 343
inserting rows 330
updating rows 343
retrieval by remote unit of work 244
type 299, 333
DATA database object 4
data type
dependencies with LIKE 333
result of operation 299
data types
conversion using scalar functions 90
GRAPHIC 253
LONG VARGRAPHIC 253
valid 253
VARGRAPHIC 253
with DBCS 253
database 251
database *(continued)*
enhancements 365
erasing queries from 66
objects, types of 4
release support 365
retrieving saved query from 64
Database Status Panel 63
DATE
scalar function 92
DATE scalar function 92
date/time
adding
to page headings/footings 138
arithmetic, date/time 100, 107
data 100
edit codes 93
formats 94
scalar functions 92, 96
DAY scalar function 94
DAYS scalar function 101
DB2 for MVS
requirement for QMF 1
specific QMF function support in 365
DB2 for VM
user ID and application requester 246
DB2/6000 for AIX
requirement for QMF 1
specific QMF function support in 365
DBCLOB 229
DBCS (double-byte character set)
appearance compared to SBCS data 252
changing lengths of names and fields 253
description 252, 262
display from database object list 252
exporting 262
followed by SI delimiter 252
graphic data type 302
how data truncation is handled 262
how incorrect data string is handled 261
importing 262
in forms 258
in input fields 257
in queries 257
on non-DBCS terminals 252
preceded by SO delimiter 252
sorting 322, 328
DBCS (double-byte character set) *(continued)*
with example elements 297
DCF (Document Composition Facility)
formatting type 270
how to insert a QMF report 270
DECIMAL
SQL scalar function 91
DECIMAL scalar function 91
decrementing dates, times, or timestamps 100
default
chart format, changing 178
report format 114
report format, changing 114
report format,changing 23
defining
example elements 297
expressions 41
new column in report 41
DELETE
command
using QBE 314
DELETE command 314
deleting
aliases 214
COMMENTS box 314
CONDITIONS box 314
example table 314
information from queries 66
lines from a query 79
query 66, 314
rows from a table 326
using QBE 326
synonyms 214
tables 214
views 214
descending order, sorting sequence 48, 327
detail blocks
using panel variations to change 147
using to refine report format 145
DIGITS scalar function 91
display
form panels 114
DISPLAY command 309
using QBE 309
displaying 27
a list of database objects using the List key 27
calculated values on reports 150
correcting queries 64

- displaying *(continued)*
  - duplicate rows 321
  - form panels 114
  - QBE Query panel 293, 306
  - reports 61
  - representative reports 134
  - special conditions on reports 152
- distributed unit of work 2, 249
- division sign (/) 299
- DO keyword 328
- Document Composition Facility 270
- document interface 279
  - CMS note facility 270
  - how to insert a QMF report
    - using XEDIT 270
  - restrictions 274
  - using 274
- double byte data 253
- double-byte character set (DBCS) 252
- DRAW command 226, 304, 314
  - using QBE 304, 314, 316
- Draw function key 72
- drawing
  - COMMENTS box 315
  - CONDITIONS box 315
  - example tables 294, 314
  - target tables 304, 314
- DSN option, GETQMF 273
- DSQAO_CONNECT_LOC 242
- DSQDC_COST_EST 208
- DSQSDBNM program parameter 240, 241
- duplicate rows
  - eliminating 305, 344
- duplicate rows in reports, eliminating 53, 305, 344
- durations
  - date/time arithmetic 101
  - incrementing and decrementing
    - dates 102
    - times 105
    - timestamps 107
  - making easier to read 105
- DXT (Data Extract) 265
  - end user dialogs
    - EXTRACT command prompt panel 267
    - extract request 266
    - main menu 266
    - used within QMF 265
  - prerequisites 266

E

- echo area, definition 19
- edit codes
  - changing 124
  - character data 123
  - currency symbol 124
  - date/time 93
  - definition 123
  - numeric data 123
  - percent data 123
  - specifying punctuation for values, in report columns 123
  - suppressing zero values 124
- EDIT command 268
- editing
  - in CMS NOTE 274
  - in ISPF-PDF 273
  - in PS/TSO 274
  - QMF objects 267, 268
  - using CLIST 268
  - using CLIST as editor name 267
  - using editors with QMF 267
  - using EXEC as editor name 267
  - using ISPF/PDF 267
  - using XEDIT 268, 273, 274
  - within QMF
    - ISPF 267
- eliminating duplicate rows in reports 53, 305, 344
- ending QMF session 6
- ENLARGE command 316
- Entry of long names within an SQL statement 69
- equalities 347
- error
  - messages
    - HELP command 14
- error messages, getting help for 14, 15
- European format, date/time edit codes 93
- evaluation of expressions, rules 298
- example
  - elements 297
  - table
    - conditions in 302
    - deleting 314
    - description 294
    - drawing 314
    - enlarging 316
    - expressions in 302
    - reducing 318
    - restrictions on 305
    - with added columns 300
- example *(continued)*
  - table *(continued)*
    - with example elements 301, 305
    - with target table 304
- example elements 297
- example table
  - conditions in 302
  - deleting 314
  - description 294
  - drawing 314
  - enlarging 316
  - expressions in 302
  - reducing 318
  - restrictions on 301, 305
  - with added columns 300
  - with example elements 301, 305
  - with target table 304
  - with unnamed columns 300
- example, database connection 246, 249
- exercises
  - using QBE 348
- exercises for Query-By-Example 348
- EXPORT command
  - CICS 234
  - HTML reports 235
  - language 251
  - language parameter 234
  - TSO 234
- exporting
  - DBCS data 262
  - HTML reports 235
  - objects
    - into CICS/VSE 234
    - into CMS 234
    - into TSO 234
- expressions
  - arithmetic 345
  - definition 41
  - evaluating 298
  - in a CONDITIONS box 302
  - in conditions 303
  - in example tables 302
  - numeric data in 299
  - order of evaluating 299
  - results with nulls 338
  - summary functions 41, 42
  - use of parentheses 299

F

- FILE option, GETQMF 272
- final text, specifying on reports 147

FLOAT
scalar function 91
FLOAT scalar function 91
footings
adding
date, time, page number 138
to reports 131
changing alignment of 138
refining, on reports 136
using global variables in 137
form
panel
changing column names 301
FORM database object 4
FORM.BREAKn panel 129
FORM.CALC panel 150
FORM.COLUMNS panel 117
FORM.CONDITIONS panel 156
FORM.DETAIL panel 145
FORM.FINAL panel 148
FORM.MAIN panel 116
FORM.OPTIONS panel 133
FORM.PAGE panel 131
formats
time and date functions 93
formats for time and date functions 93
formatting reports
for document type
ASIS 271
DCF 270
using QMF forms 113, 164
forms
correcting errors on 162
displaying 114
resetting to default values 163
saving 163
using DBCS data in 258
using to create reports 113, 164
FORMS panels, displaying 115
function keys 4, 294, 316, 318
defined 4
enlarging 316
reducing 318
function keys (and synonyms) and remote unit of work 245

G

G keyword 329
G literal 257
GETQMF editor macro 270
global variable
add 207
change 207
delete 207
global variable *(continued)*
display 207
list 207, 209
and the CASE option 209
to add or remove a variable 209
to change or delete a variable 207
location 242
panel 207
remove 209
reset 209
RESET GLOBAL command 209
set 207, 209
SET GLOBAL command 209
governor interrupt routine 64, 242
GRANT statement 245
graphic data in Prompted Query 38
GRAPHIC data type 253
graphic data with LIKE 332, 333
grouping data 329

H

HEX scalar function 91
High Performance Option 279
HISTOGRAM chart format 167
home panel
starting a session 4
Home panel 241
HOUR scalar function 95

I

I keyword 330
I operator 330
ICU (Interactive Chart Utility)
used with editor 265
IMPORT command 236
importing
DBCS data 262
objects
from CICS 236
from CMS 236
from TSO 236
IN keyword 331
using in QBE 331
with NOT 335
IN with NOT 335
incrementing dates, times, or timestamps 100
inequalities 347
in conditions 347
information, adding or changing in queries 65
input fields 257
inserting
a row into a table 330
CMS NOTE 270
GETQMF macro 271
ISPF 270
PROFS 270
PS/TSO 270
QMF report from an editor 270
reports 271, 273
rows into tables 330
USEQMF option 271
XEDIT 270
INTEGER
SQL scalar function 91
INTEGER SQL scalar function 91
interrupt
a command or query 63
by the governor 64
ISO format, date/time edit codes 93
ISPF-PDF (Interactive System Productivity Facility-Program Development Facility) 265
accessing from QMF 269
primary option menu panel 269
supported environments 273
used while in QMF 274

J

JIS format, date/time edit codes 93
joining
columns
in Prompted Query 50
in QBE 306
in SQL 79
multiple tables 54, 79, 306
strings 107
tables in Prompted Query 50

K

keys, program function (PF), on home panel 4
keywords 342, 343, 345
ALL 321
AND 321
AO 322
AVG 323
BETWEEN 325
COUNT 326
D 326
DISTINCT 78
DO 328
G 329
I 330
IN 331

keywords *(continued)*
  keywords, QBE 344
  LIKE 332
  MAX 333
  MIN 334
  NOT 335
  NULL 337
  OR 338
  P 339
  UNQ 344
  USER 345
keywords, QBE
  ALL 321
  AND 321
  AO 322
  AVG 323
  BETWEEN 325
  COUNT 326
  D 326
  DO 328
  G 329
  I 330
  IN 331
  LIKE 332
  MAX 333
  MIN 334
  NOT 335
  NULL 337
  OR 338
  P 339
  SUM 342
  U 343

L

language operand for Prompted Query 38
LANGUAGE parameter
  SET PROFILE command 293
LANGUAGE parameter on SET PROFILE command 293
LAYOUT command 134
LENGTH
  string function 97
LENGTH string function 97
LIKE keyword
  any number of characters (%) 332
  any single character (_) 332
  data type dependencies 333
  graphic data 333
  selecting on conditions 332
  with NOT 335
limits for size of chart data 170
LINE chart format 167
linear procedures
  example 181
  guidelines 183
LIST command 30, 293
  under QBE 293
List function key 27
  command prompt panel 28
  QMF CONNECT prompt panel 241
list of database objects
  displaying 27
listing tables 38
locating data on charts
  in general 168
  on pie charts 169
  on the X-axis 169
location name
  global variable for 242
  list panel 241
  QMF CONNECT prompt panel 241
  QMF's governor exit 242
  viewing 241
LONG VARGRAPHIC data type 253

M

manipulating character/graphic strings 90
MAX column function 89
  in QBE 333
  in SQL 89
message line 4
methods of accessing data 3
MICROSECOND scalar function 96
MIN column function 89, 334
minimum unique abbreviations, for commands 6
minus sign (–) 299, 345
  in expressions 345
  order of evaluation 299
MINUTE scalar function 96
model queries 309
MONTH scalar function 94
More Help, for error messages 15
more than one table, presenting data from 306
multiple
  columns, joining 57
  conditions 321, 338
    under QBE 321, 338
  table queries 79
  tables, joining 54
multiplication operator (*) 299

N

N literal 257
name
  column 1, 300
  for columns of calculated values 300
  for form panels, displaying list of 114
  for tables 2
  location qualifier 2
  owner qualifier 2
  query, maximum length 38
  tables 1
National Language Feature 251
National Language Feature (NLF) 203
negative conditions 335
NLF (National Language Feature) 203, 251
NOT keyword 335
Notices 369
null
  definition of 337
  in condition with IN keyword 331
  passing in expressions 118
  replacing with data 118
  values
    defined 337
    prints and displays as 337
    replacing using the VALUE function 98
    results in expressions 338
    using the VALUE function 98
    with conditions 303
    with G 329
    with I 330
  with NOT 335
numeric
  constants 300
  data 299
    in expressions 299

O

object owner 4
objects 4
  CHART 4
  current location 244
  DATA 4
  database 4
  displaying a list of 27
  exporting
    HTML reports 235
    into CICS 234

objects *(continued)*
- exporting *(continued)*
  - into CMS 234
  - into TSO 234
- FORM 4
- importing
  - from CICS 236
  - from CMS 236
  - from TSO 236
- listing 38
- PROC 4
- PROFILE 4
- QUERY 4
- REPORT 4
- retrieving from the database 11
- saving in the database 9
- sharing with other users 9

operators
- ALL 321
- AO 322
- D 326
- DO 328
- G 329
- I 330
- P 339
- U 343
- UNQ 344

option
- GETQMF macro 270
- USEQMF 271

OR
- keyword 338

OR keyword 338

order
- evaluating expressions 299
- of columns
  - ascending 322
  - descending 327
  - reversing 296
- rows in a report 48, 78, 322, 327
  - ascending 322
  - descending 327

OS/390 (Multiple Virtual Storage) 276, 278

P

P (present) keyword 295, 339

page heading and footing 23
- adding to a report 23

page headings
- adding
  - date, time, page number 138
  - to reports 131
- changing alignment of 138
- refining, on reports 136

page headings *(continued)*
- using global variables in 137

panel
- CONVERT Command Prompt 312
- database status 63
- display form panels using the SHOW and DISPLAY commands 114
- home 4
- Prompted Query dialog 35
- QBE QUERY 294
- RUN command prompt 310

parentheses
- in expressions 299

parentheses in expressions 299

PASS NULLS field 118

percent sign (%)
- with LIKE 332

percent sign (%) with LIKE 332

performance
- database status panel 63

performance, database status panel 63

PF (program function) keys on home panel 4

PIE chart format 167

placement of cursor 316, 318

plus sign (+) 299, 345
- in expressions 345
- order of evaluation 299

POLAR chart format 167

presenting
- all columns of a table 295, 339
- certain values in a set 331
- data 339
- data from more than one table 306, 341
- maximum number of specific columns 295
- on either of two conditions 338
- on part of a value 332
- on the opposite of a condition 335
- on two conditions 321
- rows with a certain value 296
- rows with missing entries 337
- some columns in a table 340
- some rows in a table 340
- specific columns of a table 295, 340
- specific rows of a table 296
- values within a range 324

printing
- charts 179

printing *(continued)*
- DBCS reports 262
- reports 164

PROC database object 4

procedures
- batch
  - errors 205
  - example for MVS 203
  - example for VM 203
  - termination 205
  - using IMPORT/EXPORT commands 204
  - using the QMF EXIT command 204
  - writing 202
- bilingual command 251
- DBCS data 256
- for QMF
  - in CICS environment 181
  - in CMS environment 181
  - in TSO environment 181
- linear 181
  - connecting from, to a remote location 197
  - creating 181
- reusable
  - creating 188
- running in batch 201
- sharing with other QMF users 187
- to create queries 199
  - using global variables 201
  - using REXX variable values 199
  - using template SQL statements 199
- to run 186
- with logic
  - connecting from, to a remote location 197
  - creating 181
  - example 184
  - using REXX variables in 190

processing
- data and time values 90
- order 299

processing date and time values 90

processing order 299

profile
- saving changes in the database 7
- SET PROFILE command 293
- setting up 7, 17, 293
- viewing 7

PROFILE database object 4

PROFS
- used with QMF document interface 270

Program Function (PF) keys
- on home panel 4
- QBE initial settings 294

Prompt panel
- variable data 309

prompt panel for variable data 309

prompted query
- main panel
  - command line 37
  - echo area 37
  - function key area 37
  - scroll indicator 37

Prompted Query
- and echo areas 35
- dialog panels 35
- displaying a report 61
- eliminating duplicate rows in reports 53
- finding comments about tables 39
- general rules 38
- joining multiple columns in 57
- joining tables in 50
- listing tables 38
- main panel 35
- profile requirements 38
- running a query 61
- selecting tables 38
- SQL equivalent of 67
- starting 38
- substitution variables in 58

PS/TSO, used with QMF document interface 270

PULL statements to specify REXX variables 191

Q

QBE (Query-By-Example)
- calculated values in expressions 326, 345
- commands 312
- keywords 321, 348
- Query panel 294

QMF
- getting acquainted with 1
- Home panel 17
- objects, definition 4
- quick lessons in using 17
- session
  - ending 6

qualifiers, to distinguish between columns 80

query 3
- adding
  - lines to 65, 79
  - specifications to 65
- calculated values
  - columns of expressions 346
  - for groups 329
- calculated values in 329, 346
- changing 65
- changing saved 64
- conditions
  - selecting on 296, 303
- converting 312
  - to SQL 312
- creating using Prompted Query 17, 35, 69
- creating using SQL 69, 113
- data entry 330, 343
  - insert rows 330
  - update rows 343
- definition 3
- deleting
  - DELETE command 314
  - from database 66, 314
  - information from 66
  - lines from 79
  - rows 326
- eliminating duplicate rows 321, 344
- erasing from database 66
- example elements 297
- expressions
  - arithmetic 345
  - definition 345
  - in conditions 303
- format 69
- joining multiple columns in 57
- listing 293
  - LIST command 293
- making reusable 58, 108
- model 309
- multiple table 79
- non-displaying, correcting 64
- ordering rows in a report 322, 327
- Prompted Query 3
- Query-by-Example 3
- resetting 293
- retrieving from database 64
- reusing 309
- row conditions in 43, 74, 296, 303
- rows 303
- running 61, 71, 293
  - under QBE 293

query *(continued)*
- saving 62, 110, 293
- select
  - all columns 339
  - from multiple tables 341
  - specific columns 295
- selecting
  - all columns 39, 71, 339
  - columns for a Prompted Query 19
  - from multiple tables 49, 81, 341
  - rows for a Prompted Query 20
  - specific columns 40, 71, 295
  - specific rows 43, 73, 296, 340
  - table for a Prompted Query 17
- selecting on conditions
  - BETWEEN 324
  - IN keyword 331
  - multiple 321, 338
  - negative 335
  - specific columns 340
  - specific rows 296, 340
  - values within a range 325
  - with a certain string of characters 332
  - with equality and inequality 347
- sharing with other users 63, 110, 345
- sorting rows in 48, 78
- SQL 3
- starting 38, 70
- substitution variables in 58, 108
- with DBCS data
  - graphic strings 257
- with substitution variable data 309

QUERY database object 4

Query-by-Example 293, 365

quotation marks
- when needed 299
- with constants 300
- with LIKE 332

R

range of values 325

REDUCE command 318

reformatted text, mixing with tabular data 159

remote data access, DB2 environment 2

remote location, connecting from a procedure 197
remote unit of work
  accessing the current location name 242
  connect from
    DB2 to DB2 247
    DB2 to SQL/DS 248
    SQL/DS to SQL/DS 247
  granting privileges to other locations 245
  Lost Connection prompt panel 242
  QMF objects 244
  reconnecting to a location 242
  states of QMF when a connection is lost 243
  tables and views 243
  tips and techniques 245
  tips for procedures 245
  using QMF with 244
  using with distributed unit of work 249
replacing null values using the VALUE function 98
REPORT database object 4
reports
  adding
    break segments 139
    break text 140
    new column to 117
    page headings and footings 23
    subtotals to 128
  calculating values on 149
  changing 23
    alignment of headings and data 122
    column heading 119
    column names 23
    column order 119
    column width 23, 121
    columns in 115
    default format 114
  completing before connecting to remote location 240
  correcting errors before display 162
  creating 113, 164
  default report format 23
  definition 23
  displaying
    calculated values in 150
    FORMS panels for 115
    representative 134
reports *(continued)*
  displaying *(continued)*
    special conditions on 152
  edit codes in 123
  eliminating duplicate rows from 53
  final text in 147
  footings in 131, 132, 136, 137, 139
  FORM.COLUMNS command 117
  FORM.MAIN command 116
  formatting with detail blocks 145
  headings in 131, 132, 136, 139
  mixing tabular data with reformatted text 159
  page headings 137
  printing 164
  SHOW FORM command 116
  showing totals across rows 160
  specifying fixed columns 132
  specifying punctuation for values in columns 123
  specifying text for subtotals 129
  specifying usage codes 127
  using default format 114
RESET GLOBAL command 209
RESET QUERY command 293
resetting forms to default values 163
restrictions
  AVG 324
  COUNT 326
  example elements 298
  MAX 334
  MIN 335
  SUM 343
  target tables 305
  unnamed columns 305
  variable names 311
retrieving
  data
    from multiple tables 79, 84
    with QBE 295
  query from database 64
  saved query 64
reusable procedures, creating 188
reusing queries 309
reversing order of columns 296
REXX error-handling instructions
  using messages with the EXIT instruction 194
REXX error-handling statements
  branching to subroutines 194
REXX EXECs
  calling from a procedure with logic 196
    with substitution variables 196
    without substitution variables 196
  writing 118
REXX logic in procedures
  example 184
  guidelines 185
REXX variables
  differences from substitution variables 193
  passing values to procedures with logic 192
  specifying values, using SAY and PULL statements 191
  using in procedures with logic 190
rows
  adding 330
  conditions
    changing in queries 65
    specifying 74
  deleting 326
  duplicates in reports, eliminating 53
  eliminating duplicates 78, 321, 344
  from multiple tables 341
  inserting 330
  ordering 322, 328
  select
    specific 296
  select on conditions
    AND 321
    BETWEEN 324
    OR 338
  selecting
    both conditions true 76
    one of two conditions true 76
    specific 43, 73
    using character values 74
    using conditions 74
    using grouped conditions 77
    using multiple conditions 76
    using multiple OR conditions 77
    using multiple row conditions 45
    using opposite conditions 74
    using selection symbols 75
    with no data 73

- rows *(continued)*
  - selecting on conditions 296
  - selecting some 340
  - sorting 78
  - updating 343
  - with nulls 337
- rules
  - for creating a subquery 84
  - for date/time addition 100
  - for date/time subtraction 100
  - for location of data on charts 168
- RUN command 23, 293, 310
  - description 293
  - substitution variables 310
  - to run a query 23
- running 23
  - a prompted query 23, 61
  - an SQL query 71

S

- sample tables 2
- SAVE command 293
  - under QBE 293
- saved query, changing 64
- saving
  - chart format 177
  - queries in the database 62, 110, 293
  - report forms 163
- SAY statements to specify REXX variables 191
- scalar functions 90, 92, 99
  - nesting 99
  - uses of 90, 92
- SCATTER chart format 167
- SCRIPT/VS
  - how to insert a QMF report 270
- SECOND scalar function 96
- select 17
  - all columns 71, 339
  - columns for a Prompted Query 19
  - maximum number of specific columns 295
  - multiple tables 341
  - on conditions 296
    - introduction 296
    - multiple 321, 338
    - negative 303, 335
    - values in a set 331
    - values within a range 325
    - with a certain string of characters 332
- select *(continued)*
  - on conditions *(continued)*
    - with equality and inequality 347
  - rows 20, 43, 73
  - some columns 71, 340
  - some rows 340
  - specific columns 295
  - specific rows 296
  - table 17
  - tables 38, 72
- selecting 340, 341, 347
- selection symbols 31
- SET GLOBAL command
  - creating global variables 209
  - extended syntax 209
- SET PROFILE command 293
  - LANGUAGE parameter 293
- sharing
  - queries 63, 110, 345
- shift in delimiter (SI) 252, 257
- shift in delimiter character 252
- shift out delimiter (SO) 252, 257
- shift out delimiter character 252
- shortened form panel names 114
- SHOW command
  - globals 207, 209
  - to display forms 114
- SHOW FIELD
  - for long expressions 208
- SHOW FIELD function key 208
- SHOW FORM command 116
- SHOW GLOBAL command 207, 209
- Show Global Variable panel 208
- SHOW GLOBALS command 207
- SI character 252
- size limits for chart data 170
- slash (/) 345
  - in expressions 345
- SO character 252
- solutions to exercises for QBE 351
- sort order
  - changing in queries 65
  - specifying 65, 322, 327
- sorting sequence
  - AO (ascending order) 322
  - DO (descending order) 327
- special characters in Prompted Query 38
- special conditions
  - displaying on reports 152
  - identifying using expressions 155
- special conditions *(continued)*
  - identifying using REXX EXEC 152
- SQL
  - converting queries to 312
  - equivalent of prompted query 67
  - functions, advanced
    - date/time arithmetic 100
    - joining strings 107
    - multiple table queries 79
  - statements 69
    - for adding columnns 228
    - for adding rows 225, 226
    - for authorizing access to tables 229
    - for changing rows 226, 227
    - for copying rows from one table to another 228
    - for deleting rows 227
    - using to work with data 113
  - substitution variables in 108
- SQL (Structured Query Language)
  - functions
    - date/time arithmetic 100
    - joining strings 107
- SQL query
  - converting queries from QBE 312
- SQL/DS
  - requirement for QMF 1
  - specific QMF function support in 365
- starting
  - Prompted Query 38
  - QMF 4
- status, database panel 63
- Stored Procedure Builder 279
- string
  - functions
    - LENGTH 97
    - SUBSTR 97
    - VALUE 98
- string functions 97, 98
- string of characters with LIKE 332
- subquery
  - rules for creating 84
  - to retrieve data from multiple tables 84
  - to retrieve more than one value 85
  - to satisfy a condition 86
  - using a correlation name 87
- substitution
  - values 311

substitution *(continued)*
variable
in a query 309, 311
substitution values 311
substitution variables 309
differences from REXX variables 193
in procedures with logic 188
in QBE 311
making queries reusable with 58, 108
specifying values for
as part of RUN command 59, 109
on RUN Command Prompt panel 60, 109
using global variables 61, 110
supplying values 188
using the RUN command 188
using the RUN command prompt panel 189
using the SET GLOBAL command 188
SUBSTR scalar function 97
SUBSTR string function 97
subtotals
adding to reports 128
specifying text for 129
subtraction of dates and times 100, 107
SUM
column function 89, 342
SUM column function
in QBE 342
in SQL 89
summary
conditions 326
summary conditions 326
summary functions 42
SURFACE chart format 167
synonym
deleting 214
for table 213
for view 213

T

TABLE chart format 167
table editor
search 223
using column defaults 220
using nulls 220
Table Editor
ending a session 225
Table Editor *(continued)*
for adding data to long fields 221
for adding rows 222
for changing rows 222, 224
for deleting rows 225
tables
adding columns using SQL statements 228
adding rows
using SQL statements 225, 226
using the QMF DRAW command 226
using the Table Editor 222
appending to existing 212
authorizing access 229
to add new rows 229
to change rows 229
to delete rows 229
to update columns 230
to view 229
changing rows
using SQL statements 226, 227
using the QMF DRAW command 227
using the Table Editor 222, 224
columns, relationship to 1
copying 213, 326
copying rows from one to another
using SQL statements 228
creating 211
a view from 213
alias for 214
synonym for 213
using SQL statements 211
deleting 214
deleting rows
using QBE 326
using SQL statements 227
using the Table Editor 225
inserting rows
using QBE 330
using SQL statements 225
joining 50, 79
joining multiple 54, 79
listing 38
location qualifier 2
multiple
joining columns from 79
merging data from 81
retreiving data from 84
tables *(continued)*
multiple *(continued)*
used to create a query 79
names
changing in queries 65
specifying 1
naming conventions 2
owner qualifier 2
planning to create 211
presenting data from more than one 306
revoking access 230
rows, relationship to 1
sample 2
saving 212
selecting 38, 72
target 304, 314
updating rows 343
using column defaults with Table Editor 220
with null data 337
tabular data, mixing with reformatted text 159
target table, drawing 304, 314
temporary storage
QMF objects in 9
saving to 9
three-part names
CREATE ALIAS... 249
GRANT statements 245
QMF objects 245
tablename 244
use in an SQL statement 249
use of an alias 243
TIME
scalar function 92
TIME scalar function 92
times sign (*) in expressions 345
timestamp
adding or subtracting duration 107
TIMESTAMP
scalar function 92
TIMESTAMP data 100
TIMESTAMP scalar function 92
timestamps, adding or subtracting duration 107
tips
fixing problems with charts 178
for remote unit of work 245
totals, showing across rows in reports 160
TOWER chart format 167
two conditions on one row 321

two tables, presenting data from 306

U

U keyword and operator 343
underscore (_) 298, 332
  example element 298
  with LIKE 332
UNQ 344
updating
  rows 343
updating rows 222, 343
USA format, date/time edit codes 93
usage codes
  definition 127
  specifying 127
USEQMF option, GETQMF 271
user ID
  obtaining from QMF administrator 4
user ID parameter for CONNECT command 246
USER variable 311, 345
using scalar functions to avoid null values 90

V

VALUE string function 98
values
  calculated 149
    on reports 300, 345
  expressions 345
  on reports
    sources 149
    specifying calculations for 149
values, calculated 345
  with unnamed columns 300
VARGRAPHIC
  data type 253
  scalar function 91
variables 309
  data 311
  delete 209
  reset 209
  substituting user ID for 345
  substitution 311
  USER 311, 345
  value 208, 209
view
  creating alias for 214
  creating from a table 213
  creating synonym for 213
  deleting 214
view *(continued)*
  location name 241, 242
VM
  editor 275, 276
  when QMF is active 276
  when QMF is inactive 275

W

wildcards 31

X

XEDIT editor
  used while in QMF 265
  used with QMF document interface 270

Y

YEAR scalar function 95

Z

zero values
  suppressing 124

Program Number: 5625-DB2

Printed in USA

SC18-7445-00