VAX® INTOUCH®

GUIDED QUERY LANGUAGE

This document contains PROPRIETARY and CONFIDENTIAL information and such information may not be disclosed to others for any purpose without written permission from Touch Technologies, Inc.

Touch Technologies, Inc.
9988 Hibert Street, Suite 310
San Diego, California 92131
Phone (800) 525-2527
FAX (619) 566-3663

® VAX is a registered trademark of Digital Equipment Corporation.
® INTOUCH is a registered trademark of Touch Technologies, Inc.

NOTICE

Touch Technologies, Inc. (TTI) has prepared this publication for use by TTI personnel, licensees, and customers. This information is protected by copyright. No part of this document may be photocopied, reproduced or translated to another language without prior written consent of Touch Technologies Incorporated.

TTI believes the information described in this publication is accurate and reliable; much care has been taken in its preparation. However, no responsibility, financial or otherwise, is accepted for any consequences arising out of the use of this material.

The information contained herein is subject to change without notice and should not be construed as a commitment by Touch Technologies, Inc.

The following are trademarks of Touch Technologies, Inc., and may be used only to describe products of Touch Technologies, Inc.:

DYNAMIC TAPE ACCELERATOR INTOUCH REMOTE TAPE FACILITY

DYNAMIC LOAD BALANCER PLUS FRAGMENTATION CONTROLLER REMOTE DEVICE FACILITY

The following are trademarks of System Management Software, Inc., and may be used only to describe products of System Management Software, Inc.:

RMS Expert

The following are trademarks of Digital Equipment Corporation, and may be used only to describe products of Digital Equipment Corporation:

DBMS DCL DECNET RDB RMS VAX VMS

Last revision date 04-Mar-1994

Copyright ©1992, 1994 Touch Technologies, Inc.

Contents

PREFACE xi

CHAPTER 1 USING THE GUIDED QUERY LANGUAGE 1–1

1.1 GETTING STARTED 1–1

1.2 MENU INTERFACE 1–1
1.2.1 Selecting GQL Menu Options 1–2

CHAPTER 2 CREATING A SAMPLE REPORT 2–1

2.1 REPORT TITLE 2–1

2.2 OPEN THE DATA STRUCTURE 2–2

2.3 SELECTING RECORDS 2–3

2.4 SORTING FIELDS 2–6

2.5 FIELDS TO PRINT ON THE REPORT 2–8

2.6 GENERATE THE REPORT 2–11
2.6.1 Report Output Options 2–12

2.7 EXITING GQL 2–14

CHAPTER 3 CHANGING AND SAVING THE SAMPLE REPORT 3–1

3.1 EDITING THE REPORT FILE 3–1

Contents

3.2 SAVING THE REPORT FILE 3–4
3.2.1 Getting into VMS from within GQL 3–5

3.3 RUNNING A SAVED REPORT 3–6

CHAPTER 4 GUIDED QUERY LANGUAGE TRAINING 4–1

4.1 INTOUCH TRAINING SYSTEM 4–1

CHAPTER 5 GENERAL INFORMATION ABOUT THE GUIDED QUERY LANGUAGE 5–1

5.1 THE HELP AND EXIT COMMANDS 5–1

5.2 WORKING IN VMS 5–2

5.3 SOME COMMONLY USED GQL COMMANDS 5–2

5.4 STARTING A REPORT 5–3

5.5 DISPLAYING RECORD FIELDS 5–4

5.6 SELECTING AND SORTING RECORDS 5–4

5.7 PRINTING FIELDS 5–5

5.8 TO GENERATE THE REPORT 5–5

5.9 SAVING THE REPORT FILE 5–6

5.10 EXITING GQL 5–6

CHAPTER 6 GUIDED QUERY LANGUAGE COMMANDS 6–1

6.1 %INCLUDE 6–1

6.2 APPEND 6–2

6.3 BREAK [ON] 6–2

6.4 EDIT 6–6

6.5 EXCLUDE 6–7

6.6 EXCLUDE FOUND 6–8

6.7 GO 6–9

6.8 HELP 6–10

6.9 IF THEN 6–10

6.10 INCLUDE 6–12

6.11 INCLUDE FOUND 6–13

6.12 INPUT 6–14

6.13 LET 6–16

6.14 NEW 6–18

6.15 OLD 6–18

6.16 OPEN 6–19

Contents

6.17 OUTPUT 6–20

6.18 PRINT 6–20
6.18.1 PRINT Options 6–23
6.18.1.1 HEADING • 6–24
6.18.1.2 LENGTH • 6–24
6.18.1.3 MASK • 6–24
6.18.1.4 POSITION • 6–25
6.18.1.5 TOTALS • 6–26
6.18.1.6 GROUP nn • 6–26

6.19 PROGRAM 6–30

6.20 REFRESH 6–30

6.21 RELATE 6–31
6.21.1 RELATE Options—VAX/DBMS Structures 6–33

6.22 SAVE 6–33

6.23 SET COMMANDS 6–34
6.23.1 SET COMMANDS [SHORT | LONG] 6–34
6.23.2 SET CUTOFF 6–35
6.23.3 SET EDITOR 6–35
6.23.4 SET GRANDTOTAL 6–35
6.23.5 SET INPUT MESSAGE 6–37
6.23.6 SET INPUT ROUTINE 6–38
6.23.7 SET INTOUCH [ON | OFF] 6–38
6.23.8 SET LOOKUPERR 6–39
6.23.9 SET MENU [ON | OFF] 6–40
6.23.10 SET PROCEED 6–40
6.23.11 SET PROMPT 6–41
6.23.12 SET REPROMPT 6–41
6.23.13 SET SCREEN 6–42
6.23.14 SET SPACING 6–42
6.23.15 SET TSUPPRESS [ON | OFF] 6–42

6.24 SHOW COMMANDS 6–43

6.25 SORT 6–44

6.26 SUMMARY 6–45

6.27 TITLE 6–49

CHAPTER 7 THE GUIDED QUERY LANGUAGE VOCABULARY FILE 7–1

APPENDIX A THE GUIDED QUERY LANGUAGE SUMMARY OF COMMANDS A–1

A.1 COMMANDS AFFECTING OUTPUT A–1

A.2 PRINT EXPRESSIONS A–3

A.3 COMMANDS AFFECTING SCREEN DISPLAY A–3

A.4 ADDING SYNONYMS A–4

APPENDIX B GUIDED QUERY LANGUAGE SYSTEM OPTIONS B–1

B.1 GQL SYSTEM OPTIONS B–1

B.2 NOSYSTEM B–1

B.2.1 INTERNAL EDITOR B–2

B.3 MENU ON B–2

APPENDIX C SAMPLE GQL PROCEDURE TO ACCESS DBMS DATA C–1

INDEX

EXAMPLES

1–1	Guide Screen	1–1
1–2	Guide Options Menu	1–2
2–1	Title Options Menu	2–1
2–2	Structures Menu	2–3
2–3	Record Selection Submenu	2–4
2–4	Include Options and Fields Menu	2–4
2–5	Include Options Menu	2–5
2–6	Sort Options and Fields Menu	2–7
2–7	Sort Options Menu	2–7
2–8	Print Menu	2–9
2–9	Report Generation Statistics	2–12
2–10	Report Output Options Menu	2–12
3–1	GQL Command File Example	3–2
3–2	MORE submenu	3–4
3–3	Pass to VMS from within GQL	3–6
5–1	Starting a New Report	5–3
5–2	Opening a Structure	5–3
5–3	Displaying Record Fields	5–4
5–4	Mock Report	5–5
5–5	Report Print Options	5–6
6–1	Example of BREAK Command	6–3
6–2	Example of BREAK Command - Output	6–4
6–3	Example of BREAK Command with DESCRIPTION Qualifier	6–4
6–4	Example of BREAK Command with DESCRIPTION Qualifier - Output	6–5
6–5	Example of BREAK Command with POSITION Qualifier	6–5
6–6	Example of BREAK Command with POSITION Qualifier - Output	6–6
6–7	Example of EXCLUDE Command	6–8
6–8	Example of EXCLUDE Command - Output	6–8
6–9	Example of IF THEN Command	6–11
6–10	Example of IF THEN Command - Output	6–11
6–11	Example of INCLUDE Command	6–12
6–12	Example of INCLUDE Command - Output	6–13
6–13	Example of INPUT Command	6–15
6–14	Example of INPUT Command - Output Prompts	6–15
6–15	Example of INPUT Command - Output Report	6–16
6–16	Example of LET Command	6–17
6–17	Example of LET Command - Output	6–17
6–18	Example of PRINT Command and Mock Report	6–22
6–19	Example of PRINT Command - Commands	6–22

6–20	Example of PRINT Command - Output	6–23
6–21	Example of PRINT Command with GROUP Option	6–26
6–22	Example of PRINT Command with GROUP Option - Output	6–27
6–23	Example of RELATE Command	6–32
6–24	Example of RELATE Command - Output	6–32
6–25	Example of SET GRANDTOTAL Command	6–36
6–26	Example of SET GRANDTOTAL Command - Output	6–37
6–27	Example of SORT Command	6–45
6–28	Example of SORT Command - Output	6–45
6–29	Example of SUMMARY Command, Part 1 - Full Detail	6–46
6–30	Example of SUMMARY Command, Part 1 - Full Detail - Output	6–47
6–31	Example of SUMMARY Command, Part 2 - Summary	6–48
6–32	Example of SUMMARY Command, Part 2 - Summary - Output	6–48
7–1	Example of Some Synonyms	7–1
B–1	VMS Access Error Message	B–2

TABLES

5–1	Guided Query Language HELP and EXIT Commands	5–1
5–2	Commonly Used GQL Commands	5–2

Preface

The Guided Query Language (GQL) is a natural language INTOUCH interface. GQL is intended for end-users who want to access complex databases for report design and generation. GQL helps you to easily and quickly write these reports without spending costly hours learning a traditional procedural language.

The GUIDED QUERY LANGUAGE:

- is concise and powerful.
- promotes rapid data access.
- is easy to learn—even for non-programmers.

GQL provides the user with a simple command repertoire powerful enough to allow:

- access to multiple data structures (tables of information).
- interactive report generation.
- a custom view of existing data.
- calculation of totals.
- calculation of percents.
- level breaks with summarization.
- saving and retrieving completed reports for subsequent use.

You will find the Guided Query Language the perfect tool for report design and generation. A simple set of commands entered by the end-user causes GQL to build an INTOUCH program. This program rapidly and efficiently builds the necessary report.

About this Manual

This manual describes the Guided Query Language interface to the INTOUCH system. For more information on INTOUCH, refer to the following manuals:

- A Guide to the Language
- Specifications

A Guide to the Language describes the INTOUCH system and how to develop programs with it. The manual explains the development cycle and INTOUCH's features for speeding up this cycle. It describes the INTOUCH environment, program elements and the INTOUCH language and debugging system.

The **Specifications** manual contains a brief description of the INTOUCH system, statements, commands, etc. and some background on how the system operates.

Preface

Purpose

The purpose of this document is to present information you will need to start using the Guided Query Language interface. The intent is to provide you with simple and concise explanations of the Guided Query Language system, features, and language. This manual is designed to provide a basic and thorough explanation of each element. It may also be used as a reference guide by advanced users.

Intended Audience

This manual is written for anyone interested in using The Guided Query Language. Since no programming experience is necessary to successfully create reports with GQL, this document contains no programming jargon. The manual can be easily referenced by computer users of all levels.

1 Using the Guided Query Language

The Guided Query Language (GQL) is very easy to use. As an aid to new users, a menu interface is available to get you acquainted with the language.

The menu interface will provide you with the commands you need to create a report. After you get familiar with the GQL commands, you can easily expand your report writing skills.

1.1 Getting Started

You can start the Guided Query Language by typing the word **GUIDE** at the VAX prompt ($) in upper or lower case letters.

```
$ GUIDE
```

The Guided Query Language will be initialized and you will see the GQL screen:

Example 1–1 Guide Screen

```
Guide>

                              Type  HELP  for assistance
EXIT = Exit              INTOUCH Guided Query Language                HELP = Help
```

1.2 Menu Interface

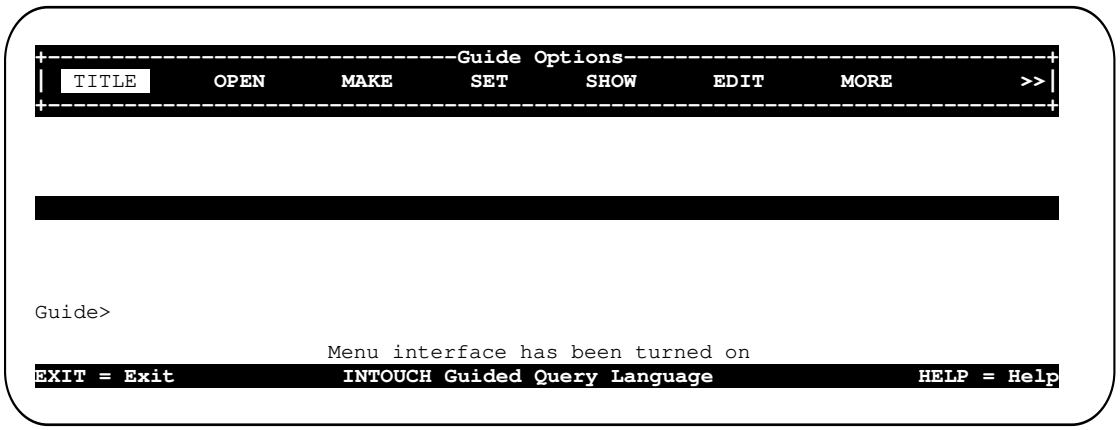
The easiest way to get familiar with the Guided Query Language commands is to use the GQL menus. The menu interface displays the Guide options and commands that you will use to create your report.

To get into the GQL menu interface, type the words **SET MENU ON** at the Guide prompt.

```
Guide> SET MENU ON
```

When the GQL menu interface is set on, a list of Guide options is displayed.

Example 1–2 Guide Options Menu



1.2.1 Selecting GQL Menu Options

Use the arrow keys to move back and forth across the top of the screen to select an option. After an option is selected, press the RETURN key to execute that option.

Some options have submenus. To select an item from a submenu, use the arrow keys to move up and down (and sometimes across). After you have selected an item from the submenu, press the RETURN key to execute that item.

If you select the wrong menu option or item, press the "\" key and you will be returned to the menu or submenu.

The symbols << and >> mean that there are more menu options or items than can be displayed on the screen. To see what the options or items are, move the arrow key in that direction.

2 Creating a Sample Report

To learn how to use the Guided Query Language, we will create a sample report using the menu options and items.

The sample report will list client information from the CLIENT file. The report will include clients which have an ID number greater than 80500 and the clients will be listed in alphabetical order by last name.

The following sections will explain exactly how to create the sample report.

2.1 Report Title

We will start the report by giving the report a name or title. To give the report a title, select the **TITLE** option from the Guide Options menu. The Title Options menu will be displayed.

Example 2–1 Title Options Menu

Select the **enter text** item. This will allow you to enter the report title.

You will be prompted for a title.

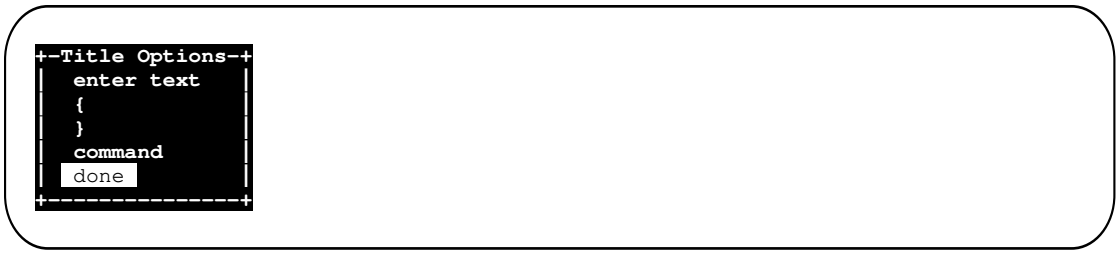
```
Guide_title>  TITLE
```

Type the words **CLIENT LIST** at the Guide_title prompt and press RETURN. The input title will be displayed.

```
Guide_title>  TITLE  "CLIENT LIST"
```

One to nine title lines can be entered. For our sample report, we will just use one title line.

The Title Options are displayed again.



Select the **done** item because we are done with the title information.

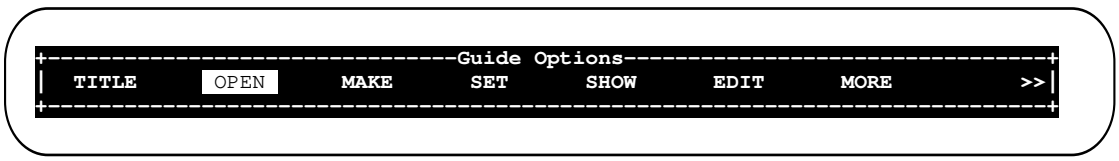
The report title is displayed in the center of the screen.

2.2 Open the Data Structure

We need to open the data structure to get the data we want to print on the report.

A data *STRUCTURE* is a tiny file which contains the name of the data file (e.g. CLIENT.DAT), the name of the definition file (e.g. CLIENT.DEF) and some information about the two files. The structure name (e.g. CLIENT.STR) is normally the same name as the data file name but it can be different.

GQL prompts us to select the **OPEN** option.



Select the **OPEN** option.

The data Structures menu is displayed. The Structures menu includes a list of the data structures that you have access to.

Example 2–2 Structures Menu

```
+--------Structures---------+
|  CLIENT     |   VENDOR     |
|  CUSTOMER   | ------------ |
|  DETAIL     |   command    |
|  INVOICE    |   done       |
|  PRODUCT    |              |
|  VEND       |              |
+-----------------------------+
```

Select the **CLIENT** structure.

The command to open the CLIENT structure is displayed at the bottom of the screen.

```
Guide_open>  OPEN CLIENT
```

GQL displays the Open Structure Options menu and prompts for the next option.

```
+Open Structure Options+
|  name                |
|  datafile            |
|  SYSTEMSET           |
|  command             |
|  done                |
+----------------------+
```

Select the **done** item because we are done selecting structures to open.

The CLIENT structure is opened and a message displays the number of fields.

```
CLIENT opened...fields: 18
```

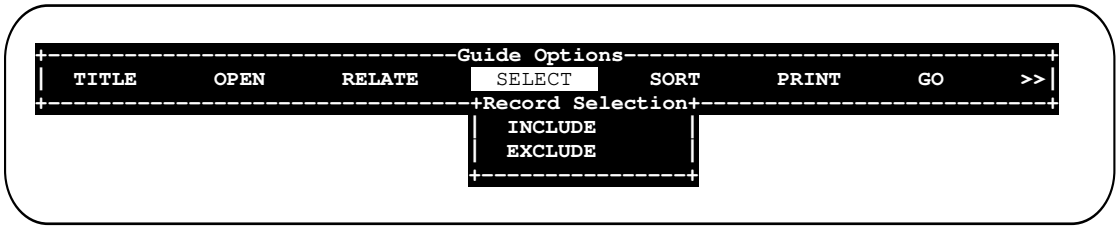
2.3 Selecting Records

We need to select what records we want to include on the report.

For the sample report, we will include records which have an ID number greater than 80500.

GQL prompts with the **SELECT** option and displays the Record Selection submenu.

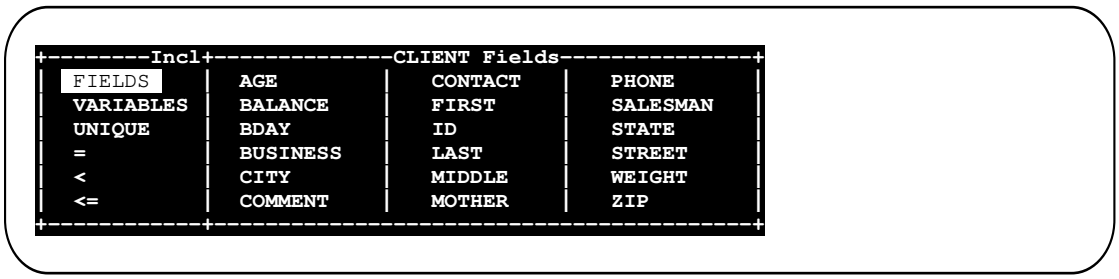
Example 2–3 Record Selection Submenu



First select the **SELECT** option. Then, select the **INCLUDE** option because we want to include records with the ID greater than 80500.

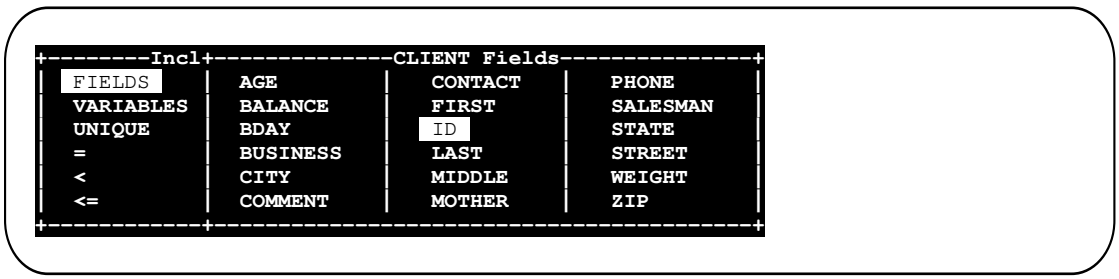
GQL displays the Include options and the fields in the CLIENT record.

Example 2–4 Include Options and Fields Menu



Select the **FIELDS** item.

Then, use the arrow keys to select the **ID** field.



The **ID** field that has been included is displayed at the bottom of the screen.

```
Guide_include>  INCLUDE CLIENT(ID)
```

The Include Options menu is displayed.

Example 2–5 Include Options Menu

```
+--------Include Options--------+
|  FIELDS    [>|   >             |
|  VARIABLES [>|   >=            |
|  UNIQUE      |   <>            |
|  =           |   enter text    |
|  <           |   command       |
|  <=          |   done          |
+--------------------------------+
```

We want to include IDs greater than 80500.

Use the arrow keys to select the **>** option.

```
+--------Include Options--------+
|  FIELDS    [>|   >             |
|  VARIABLES [>|   >=            |
|  UNIQUE      |   <>            |
|  =           |   enter text    |
|  <           |   command       |
|  <=          |   done          |
+--------------------------------+
```

GQL prompts to select the **enter text** option.

```
+--------Include Options--------+
|  FIELDS    [>|   >             |
|  VARIABLES [>|   >=            |
|  UNIQUE      |   <>            |
|  =           |   enter text    |
|  <           |   command       |
|  <=          |   done          |
+--------------------------------+
```

Select the **enter text** option.

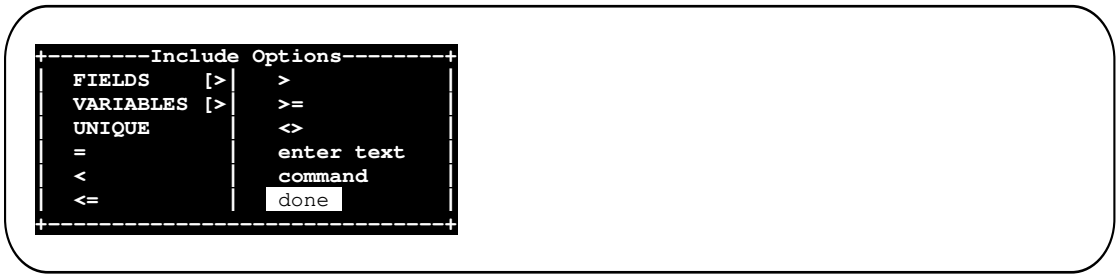
Type **80500** at the Guide_include prompt.

```
Guide_include>  INCLUDE CLIENT(ID) > 80500
```

We have completed creating the GQL command which will: include IDs that are greater than 80500. The actual command is displayed at the bottom of the screen.

```
Guide_include>  INCLUDE CLIENT(ID) > "80500"
```

GQL displays the next prompt.

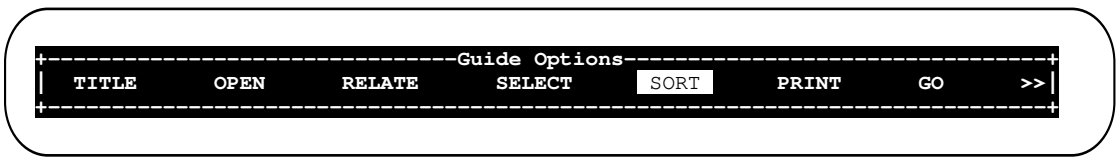


Select **done** because we are done telling GQL what records to include on the report.

2.4 Sorting Fields

We want the report to list the client information in alphabetical order by last name. The next step is to sort the records.

GQL prompts us to select the **SORT** option.



Select the **SORT** option.

The Sort Options and the CLIENT record fields are displayed.

Example 2–6 Sort Options and Fields Menu

```
+-Sort Options+-------------CLIENT Fields--------------+
| FIELDS      | AGE         |  CONTACT   |  PHONE      |
| ASCENDING   | BALANCE     |  FIRST     |  SALESMAN   |
| DESCENDING  | BDAY        |  ID        |  STATE      |
| command     | BUSINESS    |  LAST      |  STREET     |
| done        | CITY        |  MIDDLE    |  WEIGHT     |
+-------------| COMMENT     |  MOTHER    |  ZIP        |
              +-----------------------------------------+
```

GQL prompts us to select the **FIELDS** option.

Select the **FIELDS** option.

Use the arrow keys to select the **LAST** name field.

```
+-Sort Options+-------------CLIENT Fields--------------+
| FIELDS      | AGE         |  CONTACT   |  PHONE      |
| ASCENDING   | BALANCE     |  FIRST     |  SALESMAN   |
| DESCENDING  | BDAY        |  ID        |  STATE      |
| command     | BUSINESS    |  LAST      |  STREET     |
| done        | CITY        |  MIDDLE    |  WEIGHT     |
+-------------| COMMENT     |  MOTHER    |  ZIP        |
              +-----------------------------------------+
```

The Sort Options menu is displayed.

We want to list the last names in ascending order.

Use the arrow keys to select the **ASCENDING** sort option.

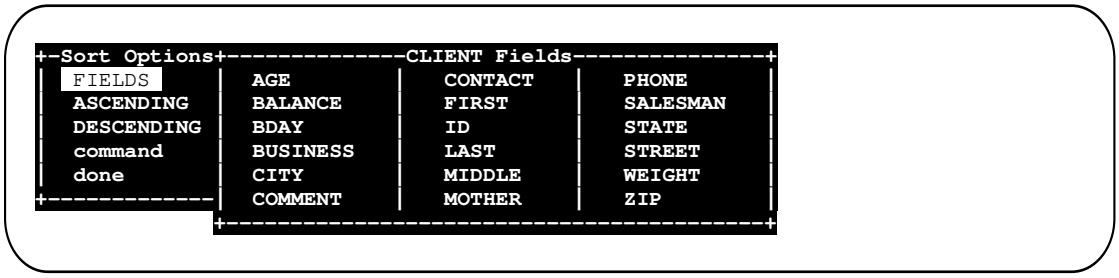
Example 2–7 Sort Options Menu

```
+-Sort Options--+
| FIELDS     [>|
| ASCENDING    |
| DESCENDING   |
| command      |
| done         |
+--------------+
```

We have completed creating the GQL command which will: sort the last names in ascending order. The actual command is displayed at the bottom of the screen.

```
Guide_sort>  SORT CLIENT(LAST) ASCENDING
```

GQL prompts with the **FIELDS** option. If there were other fields to be sorted, the preceding sort steps would be repeated for each field to be sorted.



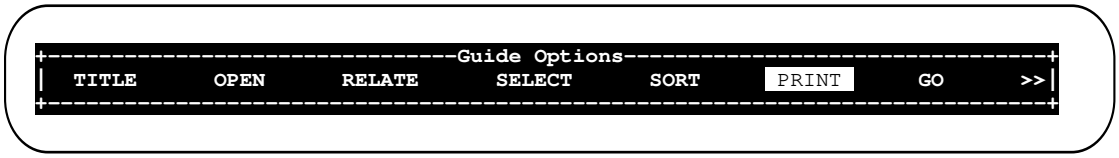
Use the arrow keys to select the **done** option because we are done telling GQL what fields to sort.

2.5 Fields to Print on the Report

We have selected the records to be included on the report and how they are to be sorted. Now we have to select the fields we want to have printed on the report.

For this sample report, we will print the ID, last name, first name and city.

GQL prompts us to select the **PRINT** option.



Select the **PRINT** option.

The Print menu and the CLIENT fields are displayed.

Example 2–8 Print Menu

```
+---------------------------------Print Menu----------------------------------+
| FIELDS      VARIABLES      item_done      done      OPTIONS      command    |
++--------------CLIENT Fields---------------+----------------------------------+
 |  AGE        |   CONTACT   |   PHONE      |
 |  BALANCE    |   FIRST     |   SALESMAN   |
 |  BDAY       |   ID        |   STATE      |
 |  BUSINESS   |   LAST      |   STREET     |
 |  CITY       |   MIDDLE    |   WEIGHT     |
 |  COMMENT    |   MOTHER    |   ZIP        |
 +-----------------------------------------+
```

GQL prompts us to select the **FIELDS** option.

Select the **FIELDS** option.

The first field we want to print on the report is the ID.

Use the arrow keys to select the **ID** field.

```
+---------------------------------Print Menu----------------------------------+
| FIELDS      VARIABLES      item_done      done      OPTIONS      command    |
++--------------CLIENT Fields---------------+----------------------------------+
 |  AGE        |   CONTACT   |   PHONE      |
 |  BALANCE    |   FIRST     |   SALESMAN   |
 |  BDAY       |   ID        |   STATE      |
 |  BUSINESS   |   LAST      |   STREET     |
 |  CITY       |   MIDDLE    |   WEIGHT     |
 |  COMMENT    |   MOTHER    |   ZIP        |
 +-----------------------------------------+
```

We have created the GQL command which will: print the ID on the report. The actual command is displayed at the bottom of the screen.

```
Guide_list>  PRINT CLIENT(ID)
```

GQL prompts us to select the **item_done** option.

```
+---------------------------------Print Menu----------------------------------+
| FIELDS      VARIABLES      item_done      done      OPTIONS      command    |
+-----------------------------------------------------------------------------+
```

Select the **item_done** option because we are done with the ID field.

GQL shows us where on the report the ID will be printed. The ID column heading and last position used are also displayed.

```
                                CLIENT LIST
                                                                    Pos:   5
CLNT
ID
-----
XXXXX
```

The next field to print on the report will be the client's last name. Perform the following steps to create the GQL command to print the LAST name field on the report.

1. Use the arrow keys to select the **LAST** field.
2. Select the **item_done** option.

GQL shows where the last name will be printed.

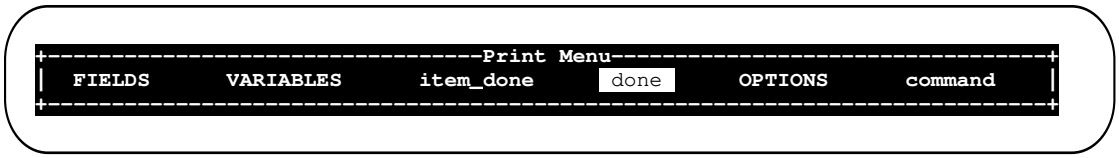
Perform the following steps to create the GQL commands to print the FIRST name field and the CITY on the report.

1. Use the arrow keys to select the **FIRST** field.
2. Select the **item_done** option.
3. Use the arrow keys to select the **CITY** field.
4. Select the **item_done** option.

We are now done creating the GQL commands to list the four fields on the report and we can see what our report will look like.

```
                                CLIENT LIST
                                                                    Pos:  65
CLNT  Last              First
ID    Name              Name             City
----- ----------------- ---------------- -------------------------
XXXXX XXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXXXXXXX
```

Use the arrow keys to select the **done** option on the Print Menu. This will tell GQL that we are done using the Print Menu.

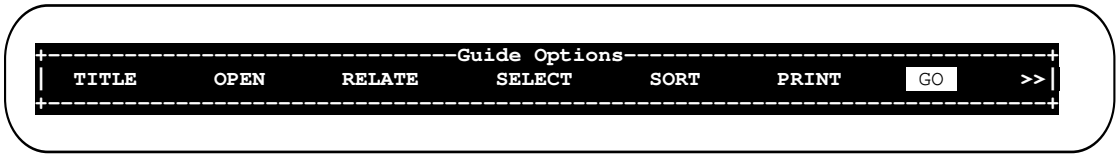


The next step will be to generate the report.

2.6 Generate the Report

We are now ready to actually generate the report.

Select the **GO** option from the Guide Options menu and report generation will begin.



As GQL starts processing your report commands, information messages are displayed.

```
Building the procedure...
Processing your request...
```

During GQL processing, report generation statistics are displayed.

Example 2–9 Report Generation Statistics

```
+------------------------------+
|             GQL              |
| Report Generation Statistics |
|          CLIENT LIST         |
|                              |
|                              |
|                              |
| Records searched:        12  |
| Records selected:         6  |
| Records read    :         6  |
| Pages printed   :         1  |
| EST time to go  :            |
|                              |
+------------------------------+
```

After the report is completed, you are given a choice of where to print the report.

2.6.1 Report Output Options

GQL displays the Output Options menu from which you can select where you want to print the report. **SCREEN** is the default option.

Example 2–10 Report Output Options Menu

```
+---------------------- Output Options ----------------------+
| Screen          |   System Printer   |   Laser Printer    |
| Printer Port    |   Save to Disk     |   EXIT             |
+-----------------------------------------------------------+
```

Select the **Screen** option to print the report on the screen.

The completed sample report is displayed.

```
14-Feb-1994                CLIENT LIST                      Page 1

CLNT  Last              First
ID    Name              Name            City
----- ----------------- --------------- -------------------------
80542 Brock             Bud             Duluth
80543 Cass              Cathy
80561 Derringer         Dale
80522 Errant            Earl
80573 Farmer            Fred            Miami
80544 Porter


                        Press the RETURN key to continue
```

After reviewing the report, press the RETURN key to return to the Output Options menu.

Use the arrow keys to select the **EXIT** option and press RETURN.

You will be returned to the Guide Options menu.

```
+-------------------------------Guide Options--------------------------------+
|<<  OPEN      RELATE      SELECT      SORT      PRINT      GO      EDIT     >>|
+----------------------------------------------------------------------------+
```

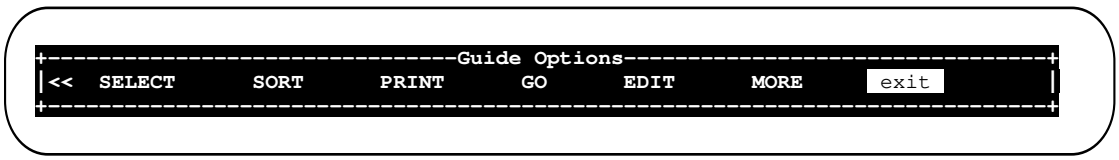
You have now completed your first Guided Query Language report.

Note

At this point, you can EITHER exit out of the Guided Query Language OR proceed to the next chapter for further instruction.

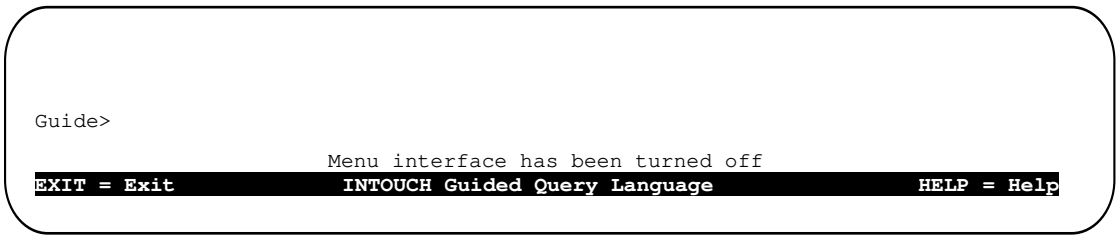
2.7 Exiting GQL

If you wish to exit the GQL menu interface, move the arrow key to the right until the **exit** option is displayed.



Select the **exit** option.

You are now out of the GQL menu interface and at the Guide prompt.



Type the word **EXIT**.

```
Guide> EXIT
```

You will be out of the Guided Query Language and at the VAX prompt ($).

3 Changing and Saving the Sample Report

In the previous chapter, you created a sample report using GQL commands.

Note

If you created the sample report as instructed in the previous chapter and then EXITed out of GQL, you will need to re-create the sample report before proceeding unless you SAVEd the sample report.

As each of the report commands was created, GQL stored the command in a temporary file.

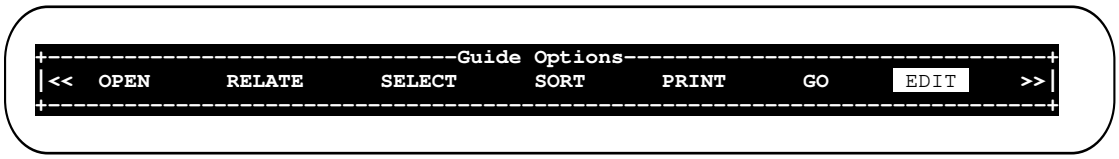
The temporary file now contains all of the GQL commands which create the sample report, "CLIENT LIST".

Let us say that we want to change the sample report to print first name, last name instead of last name, first name and also add the state.

We can make report changes by editing the temporary file which contains the GQL commands.

3.1 Editing the Report File

To see what the GQL commands look like and to make changes, GQL prompts us to select the **EDIT** option from the Guide Options menu.



Select the **EDIT** option.

When you select the **EDIT** option, you are put into edit mode. This allows you to make changes to the GQL commands.

The GQL commands for the "CLIENT LIST" report are displayed.

Changing and Saving the Sample Report

Example 3–1 GQL Command File Example

```
SET MENU ON
TITLE "CLIENT LIST"
OPEN CLIENT
INCLUDE CLIENT(ID) > "80500"
SORT CLIENT(LAST) ASCENDING
PRINT CLIENT(ID)
PRINT  CLIENT(LAST)
PRINT  CLIENT(FIRST)
PRINT  CLIENT(CITY)
[End of file]


 Buffer: GUIDE_EDIT_050002.TMP                    | Write | Insert | Forward

9 lines read from file USER:[JEANNIE]GUIDE_EDIT_050002.TMP;3
```

You can see that the GQL commands consist of key words such as TITLE, OPEN, INCLUDE, SORT, PRINT, etc. As you become familiar with the GQL commands, you will be able to write your own commands at the Guide prompt. For example, you could have typed in the following commands at the Guide prompt and created the exact same "CLIENT LIST" report that we created using the GQL menus.

```
Guide> TITLE CLIENT LIST
Guide> OPEN CLIENT
Guide> INCLUDE CLIENT(ID) > "80500"
Guide> SORT CLIENT(LAST)
Guide> PRINT CLIENT(ID)
Guide> PRINT CLIENT(LAST)
Guide> PRINT CLIENT(FIRST)
Guide> PRINT CLIENT(CITY)
```

We will now edit the temporary GQL report command file and make a change so the first name prints before the last name (e.g FIRST, LAST) on the "CLIENT LIST" report.

Using the editor, move the line which contains the **FIRST** field above the line which contains the **LAST** field.

The file will then appear as follows:

```
SET MENU ON
TITLE "CLIENT LIST"
OPEN CLIENT
INCLUDE CLIENT(ID) > "80500"
SORT CLIENT(LAST) ASCENDING
PRINT CLIENT(ID)
PRINT  CLIENT(FIRST)            <--- changed line
PRINT  CLIENT(LAST)             <--- changed line
PRINT  CLIENT(CITY)
[End of file]
```

After you have changed the line, exit out of edit mode. This will take you back to the Guide Options menu.

The report format is displayed and shows that the first and last names have been switched.

```
                                CLIENT LIST
                                                                        Pos:  65
CLNT  First           Last
ID    Name            Name               City
----- --------------- ------------------ -------------------------
XXXXX XXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXXXXXXX
```

Now, we want to include the state on the report.

To add the state field to our report, perform the following steps:

1. Select the **PRINT** option from the Guide Options menu.
2. Select the **FIELDS** item from the Print Menu.
3. Use the arrow keys to select the **STATE** field.
4. Select the **item_done** Print menu item.
5. Select the **done** Print menu item.

The report display shows that the state has been added to the report.

Select the **GO** option to run the report.

The changed report will appear as follows:

```
14-Feb-1994                  CLIENT LIST                        Page 1

CLNT  First           Last
ID    Name            Name              City                      ST
----- --------------- ----------------- ------------------------- --
80542 Bud             Brock             Duluth                    MN
80543 Cathy           Cass                                        CA
80561 Dale            Derringer                                   CA
80522 Earl            Errant                                      CA
80573 Fred            Farmer            Miami                     FL
80544                 Porter


                        Press the RETURN key to continue
```

The "CLIENT LIST" report has been changed. Now, we want to save the report so that it can be run at another time.

3.2 Saving the Report File

To save the temporary GQL command file which creates the "CLIENT LIST" report, first, get back to the Guide Options menu.

Move the arrow keys to the right and select the **MORE** option.

Example 3–2 MORE submenu

```
+-------------------------------Guide Options--------------------------------+
|<<  RELATE      SELECT      SORT      PRINT      GO      EDIT      MORE     >>|
+------------------+-----MORE-----+-----------------------------------------+
                   |  SAVE        |
                   |  HELP        |
                   |  BREAK       |
                   |  SUMMARY     |
                   |  INPUT       |
                   |  ...         |
                   +--------------+
```

You will see that **SAVE** is one of the items on the **MORE** submenu.

Use the arrow keys to select the **SAVE** option.

You are now at the Guide prompt and GQL is asking you to enter a command file name. This is the name you want to give to the temporary file that creates the "CLIENT LIST" report. GQL provides the default name of **sys$login:guide.guide**.

```
Guide_save>  SAVE sys$login:guide.guide
```

We will use the name **CLIENT_LIST** for our command file name.

Type **CLIENT_LIST** at the Guide prompt.

```
Guide_save>  SAVE CLIENT_LIST
```

Using the command file name we provided, GQL saves the temporary command file on disk. The extension *.GUIDE* is appended to the name we provided.

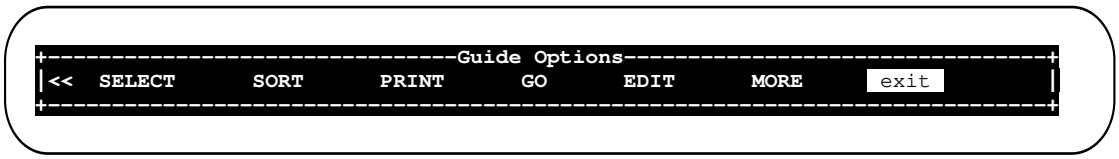
```
Commands saved in CLIENT_LIST.guide
```

You can now check your directory to see if the report command file was saved.

3.2.1 Getting into VMS from within GQL

To run a **DIRECTORY**, you need to be at the VMS prompt. You can pass to the VMS command level from within the Guided Query Language.

First, use the arrow key to select the **exit** option on the Guide Options menu and exit out of the GQL menu interface.



Note

The PF4 key can also be used at anytime to ENTER or EXIT the GQL menu interface.

You will now be at the Guide prompt.

To get to the VMS prompt, type **$$** (two "$" signs) and press the RETURN key.

```
Guide> $$
```

This clears the screen and GQL passes you to the VMS command level.

The screen appears as follows:

Example 3–3 Pass to VMS from within GQL

```
Passed to VMS...  Press  RETURN  when done.  USER:[JEANNIE]
$$
```

Type the words **DIRECTORY CLIENT_LIST** at the $$ prompt.

```
$$ DIRECTORY CLIENT_LIST
```

You will see that the GQL command file has been saved.

```
Directory USER:[JEANNIE]

CLIENT_LIST.GUIDE;1

Total of 1 file.
$$
```

Press the RETURN key *twice* to get out of VMS and back to the Guide prompt.

The commands for the "CLIENT LIST" report have been saved on disk in the **CLIENT_LIST.GUIDE** file and the report can be run at any time. The following section will explain how to do this.

3.3 Running a Saved Report

Once a GQL report has been created and the command file has been saved, it is very easy to retrieve the report file and run the report.

As an example, we will retrieve the **CLIENT_LIST.GUIDE** file and run the "CLIENT LIST" report.

To begin, we must get to the Guide prompt.

```
Guide>

EXIT = Exit              INTOUCH Guided Query Language              HELP = Help
```

Type the words **OLD CLIENT_LIST** at the Guide prompt and press the RETURN key.

```
Guide> OLD CLIENT_LIST
```

GQL displays the report information.

```
+-------------------------------Guide Options-----------------------------------+
|  TITLE      OPEN      RELATE      SELECT      SORT      PRINT      GO      >>|
+-------------------------------------------------------------------------------+


                                CLIENT LIST
                                                                        Pos:  68
CLNT  First           Last
ID    Name            Name              City                        ST
----- --------------- ----------------- --------------------------- --
XXXXX XXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXXXXXXX XX

Guide>

                           CLIENT opened...fields: 18
EXIT = Exit              INTOUCH Guided Query Language              HELP = Help
```

Use the arrow keys to select the **GO** option from the Guide Options menu.

GQL processes the **CLIENT_LIST.GUIDE** command file and generates the "CLIENT LIST" report.

Note

If you want to make changes to the report, you can select the PRINT option to add fields or the EDIT option to edit the command file. If you do make changes and you want to save the changes, you must use the SAVE option to save the changed command file.

4 Guided Query Language Training

There are many commands in the Guided Query Language. Creating the sample "CLIENT LIST" report required only a few of the commands.

To get familiar with a variety of the GQL commands and their uses, you can run the **INTOUCH TRAINING SYSTEM**.

The TRAINING SYSTEM is a series of Tutorials designed to guide you, step by step, through the creation of various GQL reports. For example, you learn how to:

- Sort information in a particular order
- Perform calculations
- Group information
- Relate a master file to detail information
- Summarize reports

4.1 INTOUCH TRAINING SYSTEM

The INTOUCH TRAINING SYSTEM is very simple to use. The *INTOUCH TRAINING SYSTEM* manual explains how to start up the TRAINING SYSTEM and select items from the TTI TRAINING SYSTEM menus.

It would be very beneficial to run the INTOUCH TRAINING SYSTEM before attempting to write more complex GQL reports.

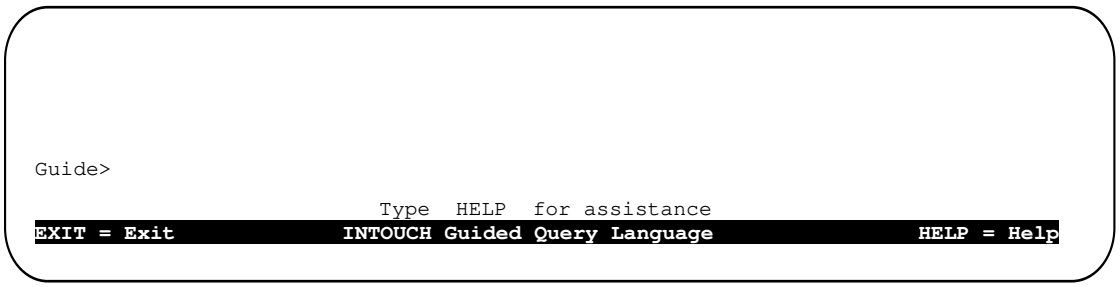
5 General Information about the Guided Query Language

This chapter recaps information discussed in previous chapters and describes some of the commonly used GQL commands.

All the GQL commands that are mentioned in this chapter and elsewhere in the manual can be accessed either at the Guide prompt or from the GQL menu interface.

5.1 The HELP and EXIT Commands

When you type **GUIDE** at the VAX prompt ($), the GQL screen is displayed.



At the bottom of the frame, a message appears. The commands described in the lower frame can be used at any time while you are in the Guided Query Language.

Table 5–1 Guided Query Language HELP and EXIT Commands

Command	Description
HELP	You can get help by typing **HELP** at the Guide prompt. This will cause GQL to provide you with helpful information.
EXIT	You may exit the GQL environment by typing **EXIT** at the Guide prompt. You will be returned to the VAX prompt ($).

Note

HELP is also available in the GQL menu interface. Press the HELP key or select the HELP menu item if it is displayed. Then, enter the GQL command that you want to get information on.

5.2 Working in VMS

You can work temporarily in VMS. For example, to send a mail message, type **$$** at the Guide prompt:

```
Guide> $$
```

This clears the screen and GQL passes you to the VMS command level. When you have finished working in VMS, press the RETURN key *twice*. The Guide prompt will reappear:

```
Guide>
```

5.3 Some Commonly Used GQL Commands

You will find it easy to write a report using just a few simple Guided Query Language commands. Some of the commonly used commands are:

Table 5–2 Commonly Used GQL Commands

Command	Description
NEW	Start a new report.
OPEN	Open a data structure.
SHOW FIELDS	Show the fields in the record.
INCLUDE	Include specific fields on the report.
SORT	Sort fields for the report.
PRINT	Select the fields to be printed on the report.
GO	Generate the report.
SAVE	Save the report command file.

5.4 Starting a Report

To start a new report, enter **NEW** at the Guide prompt. This instructs GQL to start a new report from scratch.

Example 5–1 Starting a New Report

```
Guide> NEW

                              Type  HELP  for assistance
EXIT = Exit              INTOUCH Guided Query Language                HELP = Help
```

The message **New program** is displayed.

To tell GQL which structure or structures you want to work with, enter **OPEN** and the name of the data structure at the Guide prompt.

For example, you might want to use a structure called PRODUCT to make a list of products. At the Guide prompt, enter **OPEN PRODUCT**.

Example 5–2 Opening a Structure

```
Guide> OPEN PRODUCT

                           PRODUCT opened...fields: 7
EXIT = Exit              INTOUCH Guided Query Language                HELP = Help
```

The number of fields in the PRODUCT record is displayed.

5.5 Displaying Record Fields

To get a list of fields in the record, enter **SHOW FIELDS** at the Guide prompt.

Example 5–3 Displaying Record Fields

```
+---------------------------------- PRODUCT -------------------------------------+
|COST                              REORDRQTY                                      |
|DESC                                                                             |
|MINSTOCK                                                                         |
|PRICE                                                                            |
|PRODNBR        <--- key field                                                    |
|QTYONHAND                                                                        |
+---------------------------------------------------------------------------------+

Guide> SHOW FIELDS

EXIT = Exit              INTOUCH Guided Query Language                HELP = Help
```

The key field(s) are high-lighted.

Enter **SHOW COMMANDS** at the Guide prompt to display a list of the commands which have been entered.

5.6 Selecting and Sorting Records

Use the **INCLUDE** and **EXCLUDE** commands to select structure records.

Use the **SORT** options to sort the structure records.

5.7 Printing Fields

To print record fields, use the **PRINT** command. For example, if you want a list of products that includes product number, description and cost, enter the following at the Guide prompt:

```
Guide> PRINT PRODNBR, DESC, COST
```

GQL displays a mock-up of the PRODUCT report:

Example 5–4 Mock Report

```
DD-MON-YYYY                    Report on PRODUCT                    Page: NNN
                                                                     Pos:  43

                                     Purchase
Prod# Product Description               Cost
----- ------------------------- -----------
XXXXX XXXXXXXXXXXXXXXXXXXXXXXX $XXX,XXX.XX



Guide>

EXIT = Exit              INTOUCH Guided Query Language              HELP = Help
```

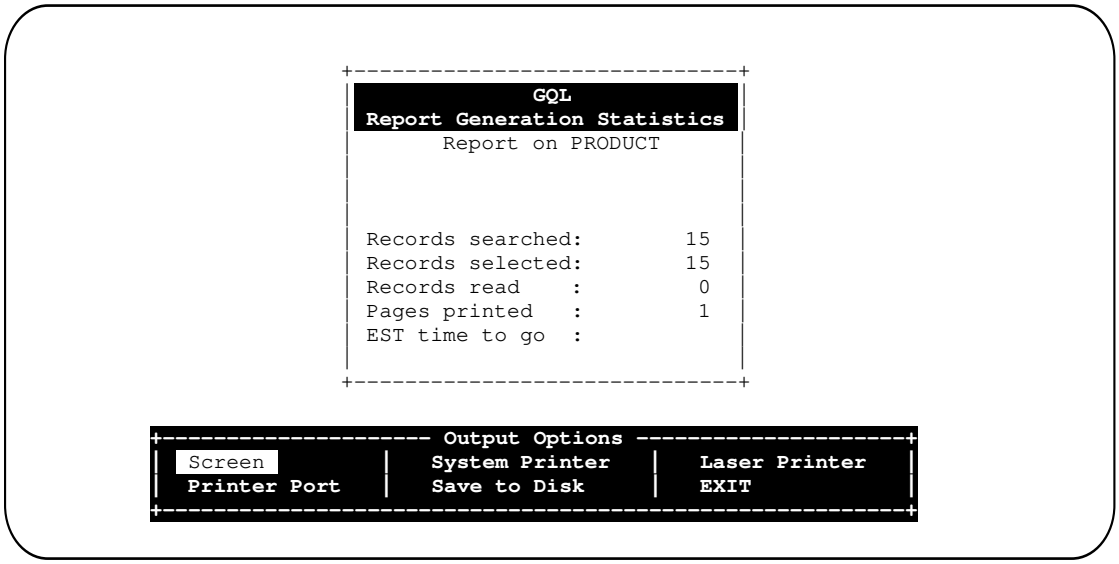
5.8 To Generate the Report

To generate the product report, enter **GO** at the Guide prompt.

Report generation messages and statistics are displayed.

After the report is generated, you are given a choice of where to print the the report. Press the RETURN key to select the default **SCREEN** option or use the arrow keys to select another option.

Example 5–5 Report Print Options



When you have finished looking at the report, use the arrow keys to select the **EXIT** option and then press the RETURN key. This will return you to the Guide prompt.

5.9 Saving the Report File

To save the report commands, enter **SAVE** and a command file name at the Guide prompt.

This will save the report commands in a disk file under the name you selected.

5.10 Exiting GQL

To exit GQL, enter **EXIT** at the Guide prompt.

6 Guided Query Language Commands

The Guided Query Language commands tell GQL what to do. GQL processes the commands and generates a report. This chapter provides a detailed explanation of each of the Guided Query Language commands. Some report examples are also included. The report examples show what commands to use to create specific report information.

Synonyms

A command can have one or more synonyms. The synonym performs the same function as the command. In the following example, **SORT** and its synonym, **ARRANGE**, organize data in sequence:

```
SORT BY customer(custnbr)

ARRANGE BY customer(custnbr)
```

Please see the section, "To Add Synonyms", in the "The Guided Query Language Summary of Commands" appendix.

Line Continuation

If the command is too long to fit on one line, the command can be continued on the following line(s). To continue a command line on the next line(s), end the line with an **ampersand** (&).

6.1 %INCLUDE

FORMAT:

```
%INCLUDE file_spec
```

EXAMPLE:

```
%INCLUDE check_account.inc
```

PURPOSE:

%INCLUDE includes a user-written routine.

The **%INCLUDE** command is used to load a user-written routine into the program that GQL creates to execute the requested query.

GQL will check that the file to be included does exist. GQL does NOT do any syntax checking of the code contained within the included file.

The **%INCLUDE** command is used to include the user-written routines specified by a SET INPUT ROUTINE command. Please refer to SET INPUT ROUTINE for further information.

The default extension for files to be included is ".INC"

6.2 APPEND

FORMAT:

```
APPEND filename.ext
```

EXAMPLE:

```
APPEND myfile.tmp
```

PURPOSE:

APPEND adds a file containing commands to the current list of commands, without your having to use the EDIT command. If no filename is given, you will be asked for a filename.

6.3 BREAK [ON]

FORMAT:

```
BREAK [ON] structure(field) [DESCRIPTION 'text' [POSITION nn]] [PAGE] [NOTOTALS]
```

EXAMPLE:

```
BREAK ON invoice(custnbr)
BREAK ON prodnbr DESCRIPTION 'Extended price sub-total:' POSITION 2
```

PURPOSE:

Use **BREAK** to indicate where level breaks (breaks in categories of information) will occur in the report. For example, you may want a break to occur between product names. **BREAK** is used to subtotal data. **BREAK** is always used after SORT arranges the data in a particular order.

ADDITIONAL:

At a level break, the **BREAK** command prints subtotals for any column keeping totals. An extra blank line is printed. When more than one level break is given, the first level break is the highest, or most significant level, and the last level break is the lowest, or least significant level. In most cases, the break level sequence should be on the same fields and in the same order as the sort sequence.

The **BREAK** command instructs GQL to perform level breaks on the field specified.

The **DESCRIPTION** qualifier attaches text (a description) to subtotals.

The **POSITION** qualifier allows users to position the text string for subtotals.

The **NOTOTALS** qualifier suppresses the printing of totals when the level break occurs.

The **PAGE** qualifier starts a new page when the break occurs.

The following commands create a report which breaks on the product number.

Example 6–1 Example of BREAK Command

```
OPEN DETAIL
SORT BY PRODNBR
SORT BY LINEID
INCLUDE INVNBR < 10330
EXCLUDE INVNBR = 10320
BREAK ON PRODNBR
PRINT PRODNBR
PRINT LINEID
PRINT QTY
PRINT PRICE
PRINT EXTPRICE WITH TOTALS
GO
```

Example 6–2 Example of BREAK Command - Output

```
01-Mar-1994   Report on DETAIL        Page 1

           Line
 Prod      Item  Order    Selling   Extended
  Nbr       ID#    Qty      Price      Price
----- --------- ------ ---------- ----------
20120 10304-002      2   1,495.00   2,990.00
20120 10323-004      1   1,495.00   1,495.00
                                   ==========
                                     4,485.00

22600 10314-002      1     325.00     325.00
22600 10323-005      1     325.00     325.00
22600 10327-002      1     325.00     325.00
22600 10328-002      1     325.00     325.00
22600 10328-003      4     325.00   1,300.00
22600 10329-003      2     325.00     650.00
                                   ==========
                                     3,250.00

22800 10301-002      2     375.00     750.00
22800 10301-004      1     375.00     375.00
22800 10301-006      2     375.00     750.00
22800 10304-003      1     375.00     375.00
22800 10313-001      2     375.00     750.00
22800 10321-002      1     375.00     375.00
22800 10324-001      4     375.00   1,500.00
22800 10325-003     10     375.00   3,750.00
                                   ==========
                                     8,625.00
```

Example 6–3 Example of BREAK Command with DESCRIPTION Qualifier

```
OPEN DETAIL
SORT BY PRODNBR
SORT BY LINEID
INCLUDE INVNBR < 10330
EXCLUDE INVNBR = 10320
BREAK ON PRODNBR DESCRIPTION 'Extended price sub-total:'
PRINT PRODNBR
PRINT LINEID
PRINT QTY
PRINT PRICE
PRINT EXTPRICE WITH TOTALS
GO
```

Example 6–4 Example of BREAK Command with DESCRIPTION Qualifier - Output

```
01-Mar-1994   Report on DETAIL        Page 1

           Line
 Prod      Item  Order    Selling   Extended
  Nbr       ID#    Qty      Price      Price
----- --------- ------ ---------- ----------
20120 10304-002      2   1,495.00   2,990.00
20120 10323-004      1   1,495.00   1,495.00
                                   ==========
       Extended price sub-total:    4,485.00

22600 10314-002      1     325.00     325.00
22600 10323-005      1     325.00     325.00
22600 10327-002      1     325.00     325.00
22600 10328-002      1     325.00     325.00
22600 10328-003      4     325.00   1,300.00
22600 10329-003      2     325.00     650.00
                                   ==========
       Extended price sub-total:    3,250.00

22800 10301-002      2     375.00     750.00
22800 10301-004      1     375.00     375.00
22800 10301-006      2     375.00     750.00
22800 10304-003      1     375.00     375.00
22800 10313-001      2     375.00     750.00
22800 10321-002      1     375.00     375.00
22800 10324-001      4     375.00   1,500.00
22800 10325-003     10     375.00   3,750.00
                                   ==========
       Extended price sub-total:    8,625.00
```

Example 6–5 Example of BREAK Command with POSITION Qualifier

```
OPEN DETAIL
SORT BY PRODNBR
SORT BY LINEID
INCLUDE INVNBR < 10330
EXCLUDE INVNBR = 10320
BREAK ON PRODNBR DESCRIPTION 'Extended price sub-total:' POSITION 2
PRINT PRODNBR
PRINT LINEID
PRINT QTY
PRINT PRICE
PRINT EXTPRICE WITH TOTALS
GO
```

Example 6–6 Example of BREAK Command with POSITION Qualifier - Output

```
01-Mar-1994   Report on DETAIL        Page 1

           Line
 Prod      Item  Order    Selling   Extended
  Nbr       ID#    Qty      Price      Price
----- --------- ------ ---------- ----------
20120 10304-002      2   1,495.00   2,990.00
20120 10323-004      1   1,495.00   1,495.00
                                   ==========
 Extended price sub-total:           4,485.00

22600 10314-002      1     325.00     325.00
22600 10323-005      1     325.00     325.00
22600 10327-002      1     325.00     325.00
22600 10328-002      1     325.00     325.00
22600 10328-003      4     325.00   1,300.00
22600 10329-003      2     325.00     650.00
                                   ==========
 Extended price sub-total:           3,250.00

22800 10301-002      2     375.00     750.00
22800 10301-004      1     375.00     375.00
22800 10301-006      2     375.00     750.00
22800 10304-003      1     375.00     375.00
22800 10313-001      2     375.00     750.00
22800 10321-002      1     375.00     375.00
22800 10324-001      4     375.00   1,500.00
22800 10325-003     10     375.00   3,750.00
                                   ==========
 Extended price sub-total:           8,625.00
```

6.4 EDIT

FORMAT:

```
EDIT
```

EXAMPLE:

```
EDIT
```

PURPOSE:

The **EDIT** command allows you to make changes to a GQL command file. When the **EDIT** command is executed, you are put into edit mode.

ADDITIONAL:

You can edit your commands to correct errors, insert comments, and/or resequence the commands. To EXIT the editor, press the PF1 key and then the F key. GQL will close files, reset the command list and reload the commands from the edited command file.

If you want to save the changes, you must use the SAVE command or the changes will be lost.

6.5 EXCLUDE

FORMAT:

```
EXCLUDE cond_expr
```

EXAMPLE:

```
EXCLUDE customer(days_past_due) < 15
```

PURPOSE:

EXCLUDE tells GQL which records to exclude from the report.

ADDITIONAL:

When GQL reads the **EXCLUDE** command, it evaluates the expression cond_expr. If the expression is TRUE, GQL will exclude the current record from the report.

When more than one **EXCLUDE** command is given, ANY command expression that is TRUE will cause the record to be excluded from the selected subset.

Example 6–7 Example of EXCLUDE Command

```
OPEN DETAIL
INCLUDE INVNBR = 10301
EXCLUDE PRODNBR = 24200
SORT BY PRODNBR
SORT BY LINEID
PRINT INVNBR, PRODNBR, LINEID, QTY, PRICE, EXTPRICE
GO
```

The report will not include the product number "24200" because it has been excluded.

Example 6–8 Example of EXCLUDE Command - Output

```
01-Mar-1994      Report on DETAIL           Page 1

                 Line
  Inv  Prod      Item  Order    Selling   Extended
  Nbr   Nbr       ID#    Qty      Price      Price
----- ----- --------- ------ ---------- ----------
10301 22800 10301-002      2     375.00     750.00
10301 22800 10301-004      1     375.00     375.00
10301 22800 10301-006      2     375.00     750.00
10301 28800 10301-009      2     249.00     498.00
10301 31020 10301-005      1     595.00     595.00
10301 31150 10301-001      1   2,495.00   2,495.00
10301 31150 10301-008      8   2,495.00  19,960.00
10301 33090 10301-007      2   1,495.00   2,990.00
```

6.6 EXCLUDE FOUND

FORMAT:

```
EXCLUDE FOUND structure_name
```

EXAMPLE:

```
EXCLUDE FOUND customer
```

PURPOSE:

EXCLUDE FOUND tells GQL to exclude a record if it is found in the given structure.

Note

EXCLUDE FOUND must be used AFTER the RELATE command.

6.7 GO

FORMAT:

```
GO
```

EXAMPLE:

```
GO
```

PURPOSE:

The **GO** command starts generation of the report. The nature and appearance of the report depends on all of the commands given prior to the **GO** command.

ADDITIONAL:

When you enter the **GO** command, GQL calls a template program and inserts INTOUCH code at various places to customize the program according to the commands given. If an error is detected at this point, a message is displayed. The modified template program runs, giving progress statistics as it selects data and formats the report. After the report is formatted, you can select where to print the report. Selecting EXIT from the report output options menu will take you back to the Guide prompt.

After returning from **GO**, your screen will be refreshed with the report title and column headings, and your commands will be as you left them.

6.8 HELP

FORMAT:

```
HELP
```

EXAMPLE:

```
HELP
```

PURPOSE:

The **HELP** command displays information about GQL and the commands.

6.9 IF THEN

FORMAT:

```
IF cond_expr THEN
  command(s)
ELSE
  command(s)
ENDIF
```

EXAMPLE:

```
IF age > 50 THEN
   let x$ = 'yes'
ELSE
   let x$ = 'no'
ENDIF
```

PURPOSE:

Use the **IF THEN** command when you want to execute a command or block of commands only under specific conditions.

Example 6–9 Example of IF THEN Command

```
OPEN VEND
EXCLUDE CITY = "FALLBROOK"
SORT BY NAME
PRINT ID
PRINT NAME
PRINT BALANCE
IF BALANCE > 7500  THEN
  LET A$ = 'Send notification'
ELSE
  LET A$ = ''
ENDIF
PRINT A$ LENGTH 17 HEADING 'Necessary Action'
GO
```

Example 6–10 Example of IF THEN Command - Output

```
01-Mar-1994                 Report on VEND                      Page 1

                                                  YTD
Student ID Full name                           balance Necessary Action
---------- ------------------------------ ----------- -----------------
0100137324 ARNETT, CHRISTINA M.             $8,025.50 Send notification
0300137324 BECKWITH, JENNIFER               $8,217.50 Send notification
AK00000401 COX, SHEILA S.                   $7,607.00 Send notification
0400136501 EDGMON, JENNIFER L.              $8,060.00 Send notification
0500000484 HAMPTON, AMY L.                  $7,418.00
0200000459 HAUSER, SHANE R.                 $7,890.00 Send notification
0400000546 HODSDON, ERIC                    $7,915.00 Send notification
0600000415 HOLLADAY, SCOTT T.               $7,546.00 Send notification
0300137118 JENSEN, ALFRED G.                $7,698.50 Send notification
0200137324 KLEMPLE, RANDAL M.               $7,900.50 Send notification
0400137323 REYES, MARIO                     $7,421.50
0100137155 THORN, MELISSA R.                $7,856.50 Send notification
```

6.10 INCLUDE

FORMAT:

```
INCLUDE cond_expr
```

EXAMPLE:

```
INCLUDE customer(balance) > customer(min_balance)
```

PURPOSE:

INCLUDE tells GQL which records to include on the report.

ADDITIONAL:

When GQL reads the **INCLUDE** command, it evaluates the expression cond_expr. If the expression is TRUE, GQL will include the current record on the report.

When more than one **INCLUDE** expression is given, ALL expressions must be TRUE in order for the data to be included.

INCLUDE also allows you to provide a range of values (e.g., low value to high value):

```
INCLUDE custnbr = 10000 TO 11000
```

Example 6–11 Example of INCLUDE Command

```
OPEN DETAIL
INCLUDE PRODNBR = 31020
SORT BY PRODNBR
PRINT PRODNBR, LINEID, QTY, PRICE, EXTPRICE
GO
```

Example 6–12 Example of INCLUDE Command - Output

```
02-Mar-1994   Report on DETAIL        Page 1

           Line
 Prod      Item  Order    Selling   Extended
  Nbr       ID#    Qty      Price      Price
----- --------- ------ ---------- ----------
31020 10301-005      1     595.00     595.00
31020 10322-002      1     595.00     595.00
31020 10323-002      1     595.00     595.00
31020 10328-005      9     595.00   5,355.00
31020 10335-001      1     595.00     595.00
31020 10341-002      1     595.00     595.00
31020 10341-003      2     595.00   1,190.00
31020 10341-005      1     595.00     595.00
31020 10345-010      2     595.00   1,190.00
31020 10348-006      8     595.00   4,760.00
31020 10356-001      5     595.00   2,975.00
31020 10363-001      2     595.00   1,190.00
31020 10364-003      3     595.00   1,785.00
31020 10372-003      1     595.00     595.00
31020 10378-002      1     595.00     595.00
31020 10379-003      2     595.00   1,190.00
31020 10384-005      2     595.00   1,190.00
31020 10390-001      4     595.00   2,380.00
```

6.11 INCLUDE FOUND

FORMAT:

```
INCLUDE FOUND structure_name
```

EXAMPLE:

```
INCLUDE FOUND invoice
```

PURPOSE:

INCLUDE FOUND tells GQL that you want to work ONLY with records where the relationship between two structures exists.

For example: If CUSTOMER is related to INVOICE and only CUSTOMER records with INVOICE records are to be included, "**INCLUDE FOUND** INVOICE" must be used or the CUSTOMER records without INVOICE records will also be included.

6.12 INPUT

FORMAT:

```
[LINE] INPUT ['Prompt_text'] [, PROMPT str_expr]
             [, AT row, col] [, LENGTH num_expr] [, DEFAULT str_expr]
             [, VALID str_expr] [, TIMEOUT time_limit] : var
```

EXAMPLE:

```
INPUT 'Enter the account to print' : acct$
INPUT at 3, 10, valid 'number', prompt 'Account number: ' : acct$
```

PURPOSE:

INPUT allows data to be entered by a user.

The **INPUT** command allows the developer of the report to prompt the user for information necessary to process the report.

Any valid INTOUCH **INPUT** commands can be used. When the report is executed, the user will be prompted for data for all the **INPUT** commands. The data is available for use in the INCLUDE, EXCLUDE and PRINT commands.

In addition, a user-written validation routine can be specified. Please refer to SET INPUT ROUTINE for further information about validation routines.

The **INPUT** commands are processed in the order that they are entered to GQL. The user is given BACK and EXIT capabilities from all **INPUT** commands. The developer can enter a message to be displayed during each **INPUT** command to assist the user in entering the data. The SET INPUT MESSAGE command is used to enter input messages.

If the SET PROCEED option is set to TRUE, the user will be asked a "Proceed (Y/N)" question before the report is produced.

If the SET REPROMPT option is set to TRUE, the user will be reprompted for all inputs when finished printing the report.

For additional information on INPUT, please refer to the chapter, "Inputting Data," in the *INTOUCH, A Guide to the Language* manual.

Example 6–13 Example of INPUT Command

```
OPEN DETAIL
INPUT 'Enter the product to be included' : PRODUCT$
INCLUDE PRODNBR = PRODUCT$
PRINT PRODNBR, LINEID, QTY, PRICE, EXTPRICE
GO
```

Example 6–14 Example of INPUT Command - Output Prompts

```
                                Report on DETAIL
Enter the product to be included? 22600



Proceed (Y/N): Y___
```

Example 6–15 Example of INPUT Command - Output Report

```
02-Mar-1994   Report on DETAIL        Page 1

           Line
 Prod      Item  Order    Selling   Extended
  Nbr       ID#    Qty      Price      Price
----- --------- ------ ---------- ----------
22600 10314-002      1     325.00     325.00
22600 10323-005      1     325.00     325.00
22600 10327-002      1     325.00     325.00
22600 10328-002      1     325.00     325.00
22600 10328-003      4     325.00   1,300.00
22600 10329-003      2     325.00     650.00
22600 10342-002      1     325.00     325.00
22600 10345-005      1     325.00     325.00
22600 10345-007      3     325.00     975.00
22600 10348-003      3     325.00     975.00
22600 10362-003      1     325.00     325.00
22600 10363-006      2     325.00     650.00
22600 10368-002      2     325.00     650.00
22600 10379-005      2     325.00     650.00
22600 10393-005      1     325.00     325.00
22600 10400-003      1     325.00     325.00
```

6.13 LET

FORMAT:

```
LET variable_name = expression
```

EXAMPLE:

```
LET extamt = qty * price
PRINT extamt heading 'Extended,Amount' MASK '$##,###.##'
```

PURPOSE:

The **LET** command allows you to assign a value to a user variable. **LET** sets up a variable, defined by the given expression. The variable can then be used in PRINT commands and in other expressions.

Variable names must start with a letter.

Example 6–16 Example of LET Command

```
OPEN DETAIL
SORT BY LINEID
PRINT LINEID, QTY, PRICE
LET EXTAMT = QTY * PRICE
PRINT EXTAMT HEADING 'Extended,Amount' MASK '$##,###.##'
GO
```

Example 6–17 Example of LET Command - Output

```
02-Mar-1994  Report on DETAIL   Page 1

     Line
     Item  Order    Selling   Extended
      ID#    Qty      Price     Amount
--------- ------ ---------- ----------
10301-001      1   2,495.00  $2,495.00
10301-002      2     375.00    $750.00
10301-003      1     299.00    $299.00
10301-004      1     375.00    $375.00
10301-005      1     595.00    $595.00
10301-006      2     375.00    $750.00
10301-007      2   1,495.00  $2,990.00
10301-008      8   2,495.00 $19,960.00
10301-009      2     249.00    $498.00
10302-001      3     149.00    $447.00
10303-001      1   1,495.00  $1,495.00
10303-002      3     299.00    $897.00
10303-003      2     795.00  $1,590.00
10304-001      1   2,095.00  $2,095.00
10304-002      2   1,495.00  $2,990.00
```

6.14 NEW

FORMAT:

```
NEW
```

EXAMPLE:

```
NEW
```

PURPOSE:

Use **NEW** to instruct GQL to start a new report from scratch.

ADDITIONAL:

NEW clears memory of SOME commands entered in the current session.

Note

SAVE your file before using the NEW command if the file is one you wish to keep. (See SAVE.)

6.15 OLD

FORMAT:

```
OLD filename
```

EXAMPLE:

```
OLD client_list
```

PURPOSE:

The **OLD** command gets a previously SAVE'd GQL program and executes the commands as if they were being entered by the user.

To delete previous commands, enter the NEW command before entering the **OLD** command. If you want to save the previous commands, use the SAVE command before using the NEW command.

ADDITIONAL:

GQL searches for the file specified and loads it into current memory. Filename is the specification for the file you are loading.

6.16 OPEN

FORMAT:

```
OPEN [str-syn NAME] filespec [DATAFILE data_file_spec] [SYSTEMSET 'set-name']
OPEN structure-synonym-in-GUIDE_NAMES
```

EXAMPLE:

```
OPEN product

OPEN tti_run:vendor

OPEN cust NAME tti_run:customer
```

PURPOSE:

OPEN accesses a structure that you will use in the report. The structure is opened immediately so the field names and attributes are available for reference and error checking.

ADDITIONAL:

str-syn	Synonym or nickname to use when referring to the structure within GQL (e.g., CUST for CUSTOMER). If you do not indicate a synonym, you will use the structure's filename to refer to the structure.
filespec	Structure to open. The structure name can be any VMS file specification.
datafile	The name of an alternate datafile to be used when opening the structure.
systemset	*For DBMS structures only.* This is the name of a system owned set to be used when accessing the records of the primary (first) structure.

On receiving the **OPEN** command, GQL first checks for a specific structure file name based on the NAME keyword. If NAME is not found, the GUIDE_NAMES database is checked for the structure name. If found, the structure specified in GUIDE_NAMES is opened. If not found, GQL looks for a structure filename.

The first structure opened is considered the *primary structure*. Other opened structures can only be used as a cross-reference lookup. Currently you can have up to *nine* structures open.

In other commands, when you refer to fields, you can specify a field name without using the structure name, unless the name is ambiguous. The first structure that has a field with that name will be used.

When GQL executes the **OPEN** command, GQL searches for the file specified. If the file is not found, GQL will display an appropriate warning message. If the file is found, GQL opens the structure and reports on the number of fields in the record. GQL now has all pertinent information related to the file such as:

- number of fields
- field names
- field print masks
- default column headers

6.17 OUTPUT

FORMAT:

```
OUTPUT report_filename [UNIQUE]
```

EXAMPLE:

```
OUTPUT invoice_report
```

PURPOSE:

GQL normally creates a temporary program which in turn creates a temporary report file with the name "GUIDE_LIST_xxxxxx.TMP". If the report is to be saved for later printing, it can be useful to give it a more appropriate name. The default output file extension is ".LIS".

The **UNIQUE** qualifier adds a unique alpha-numeric string to the filename extension to guarantee that the filename will be unique. For example, the command **OUTPUT INVOICE UNIQUE** causes GQL to save the report in a filename similar to:

INVOICE.LIS_850001

6.18 PRINT

FORMAT:

```
PRINT [print_keyword] item1 [options] [, item2 [options . . .]
```

EXAMPLE:

```
PRINT name
PRINT prodnbr, desc, price
PRINT CUME price
PRINT addr HEADING 'Address'
```

PURPOSE:

The **PRINT** command tells GQL which field(s) or item(s) to print on the report and establishes a new column to print. You can specify more than one column by separating field names or items with commas.

print_keyword *Optional word CUME or PERCENT. See below.*

ITEM is the item to print. Each item can be:

A field	PRINT price
	PRINT customer(name)
A user variable	PRINT custname$
An expression	PRINT price * quantity

ADDITIONAL:

The **PRINT** command causes GQL to paste a mock-up of the field or item on the screen. If more than 80 columns are required to display all the requested fields, GQL will automatically switch to the 132-column mode. Use **PRINT** to access and display any item or field you want to use in your report.

Printing Elements of Arrays

To print an element of an array, specify the element number using a **#** character. For example:

```
PRINT customer(address#1)
PRINT customer (address#index%)
```

PRINT Keywords

After the word **PRINT**, you can use one of the following optional keywords: **CUME** or **PERCENT**

PRINT CUME - Print a Cumulative Running Total.

PRINT PERCENT - Print a percentage of an amount.

PRINT PERCENT expression OF [TOTAL] expression - Print percentage of total amount.

Example 6–18 Example of PRINT Command and Mock Report

```
DD-MON-YYYY                    Report on DETAIL                        Page: NNN
                                                                        Pos:  44
     Line
     Item  Prod  Order    Selling   Extended
      ID#   Nbr    Qty      Price      Price
--------- ----- ------ ---------- ----------
XXXXX-XXX XXXXX XX,XXX XXX,XXX.XX XXX,XXX.XX



+--------------------------------- Commands -----------------------------------+
|OPEN DETAIL                                                                   |
|SORT BY LINEID                                                                |
|PRINT LINEID, PRODNBR, QTY, PRICE, EXTPRICE                                   |
|                                                                              |
|                                                                              |
|                                                                              |
+------------------------------------------------------------------------------+
Guide>

EXIT = Exit              INTOUCH Guided Query Language              HELP = Help
```

Example 6–19 Example of PRINT Command - Commands

```
OPEN DETAIL
SORT BY LINEID
PRINT LINEID, PRODNBR, QTY, PRICE, EXTPRICE
GO
```

Example 6–20 Example of PRINT Command - Output

```
02-Mar-1994   Report on DETAIL        Page 1

     Line
     Item  Prod  Order    Selling   Extended
      ID#   Nbr    Qty      Price      Price
--------- ----- ------ ---------- ----------
10301-001 31150      1   2,495.00   2,495.00
10301-002 22800      2     375.00     750.00
10301-003 24200      1     299.00     299.00
10301-004 22800      1     375.00     375.00
10301-005 31020      1     595.00     595.00
10301-006 22800      2     375.00     750.00
10301-007 33090      2   1,495.00   2,990.00
10301-008 31150      8   2,495.00  19,960.00
10301-009 28800      2     249.00     498.00
10302-001 28400      3     149.00     447.00
10303-001 33090      1   1,495.00   1,495.00
10303-002 24200      3     299.00     897.00
10303-003 31030      2     795.00   1,590.00
10304-001 33180      1   2,095.00   2,095.00
10304-002 20120      2   1,495.00   2,990.00

                        Press the RETURN key to continue
```

6.18.1 PRINT Options

For each item, you can use one or more of the following print options:

HEADING ’column_heading’

LENGTH column_length

MASK ’edit_mask’

POSITION column_position

[WITH] TOTALS [= total_expression]

GROUP break_level

6.18.1.1 HEADING

One to three **HEADING** lines are permitted. A comma causes the heading to be printed on the next line. If no heading is specified, GQL uses the heading that is stored in the field definition file.

HEADING 'hdg1, hdg2, hdg3'

Example:

```
PRINT employee(last) HEADING 'Employee,Last,Name'
```

This example will produce a column heading that looks like this:

```
Employee
Last
Name
--------------------
XXXXXXXXXXXXXXXXXXX
```

6.18.1.2 LENGTH

Specifying the column **LENGTH** (number of characters in the column) overrides the default column length. This option can be used to shorten (or lengthen) the column.

LENGTH nn

Example:

```
PRINT customer(name) LENGTH 12
```

6.18.1.3 MASK

MASK permits you to override the default edit mask. The default mask is the one stored in the field definition file. You might want to specify your own mask in columns where calculations are performed or totals are specified.

For further description of edit masks, refer to the "PRINT USING" section of the *INTOUCH, A Guide to the Language* manual.

MASK '###,###'

Example:

```
PRINT invoice(qty) * invoice(unitprice) MASK '$##,###.##'
```

6.18.1.4 POSITION

The **POSITION** option specifies the absolute printing position to start the column. If the column specified is less than or equal to the last position of the last specified column, the column will be placed at that position on the NEXT line down. If any part of the column underlies another column, the heading for the new column will be suppressed.

The following example could be used for printing a mailing-label format. Position 1 is the far-left side of the screen; position 80 (normal screen) or 132 (wide screen) is the far-right side.

POSITION nn

Example:

```
PRINT customer(name)
PRINT customer(address) POSITION 1
PRINT customer(city) POSITION 1
PRINT customer(state)
PRINT customer(zip)
```

POSITION +nn specifies a relative printing position from the first available space after the last specified column. For example, POSITION +3 will leave 3 blank spaces before this column, and POSITION +0 will leave no blank spaces between this column and the last one.

Example:

```
PRINT customer(name)
PRINT customer(address) POSITION +3
PRINT customer(city) POSITION +3
```

POSITION = is used to print this column directly beneath the last column. In the following example, customer(address) and customer(city) are printed underneath customer(name).

Example:

```
PRINT customer(custnbr)
PRINT customer(name)
PRINT customer(address) POSITION =
PRINT customer(city) POSITION =
PRINT customer(state)
PRINT customer(zip)
```

POSITION = nn will print this column beneath column number nn. Columns are counted in the order they are listed. In the following example, customer(address) will print underneath customer(name).

Example:

```
PRINT customer(custnbr)
PRINT customer(name)
PRINT customer(balance)
PRINT customer(days_past_due)
PRINT customer(address) POSITION = 2
```

6.18.1.5 TOTALS

TOTALS specifies that this column will be totaled. Subtotals will be printed at level breaks and grand totals will be printed at the end of the report. The word **WITH** is optional.

PRINT [WITH] TOTALS [= total_expression]

Examples:

```
PRINT detail(qty) with totals

PRINT price / qty with totals = total price / total qty &
      heading 'Average,Unit,Price' mask '###,###.##'
```

6.18.1.6 GROUP nn

The **GROUP** option groups records together. Where the data in a column is identical within a given break level, only the first occurrence is printed. You can specify the break level, **GROUP nn**, that the grouping is associated with. If the group number is not specified, the grouping is associated with break level 1.

Example 6–21 Example of PRINT Command with GROUP Option

```
OPEN DETAIL
SORT BY PRODNBR
SORT BY LINEID
INCLUDE INVNBR = 10301 TO 10302
INCLUDE PRODNBR = 22800 TO 31150
BREAK ON PRODNBR
PRINT PRODNBR GROUP 1
PRINT LINEID
PRINT QTY
PRINT PRICE
PRINT EXTPRICE WITH TOTALS
GO
```

Example 6–22 Example of PRINT Command with GROUP Option - Output

```
02-Mar-1994   Report on DETAIL        Page 1

           Line
 Prod      Item  Order    Selling   Extended
  Nbr       ID#    Qty      Price      Price
----- --------- ------ ---------- ----------
22800 10301-002      2     375.00     750.00
      10301-004      1     375.00     375.00
      10301-006      2     375.00     750.00
                                   ==========
                                     1,875.00

24200 10301-003      1     299.00     299.00
                                   ==========
                                        299.00

28400 10302-001      3     149.00     447.00
                                   ==========
                                        447.00

28800 10301-009      2     249.00     498.00
                                   ==========
                                        498.00

31020 10301-005      1     595.00     595.00
                                   ==========
                                        595.00

31150 10301-001      1   2,495.00   2,495.00
      10301-008      8   2,495.00  19,960.00
                                   ==========
                                     22,455.00
                                   ==========
                                     26,169.00
```

The following is another example using the **PRINT GROUP** option.

```
OPEN DETAIL
OPEN INVOICE
OPEN CUSTOMER
OPEN PRODUCT
RELATE DETAIL(INVNBR) TO INVOICE(INVNBR)
RELATE CUSTNBR TO CUSTOMER(CUSTNBR)
RELATE PRODNBR TO PRODUCT(PRODNBR)
TITLE 'INVOICES'
SORT BY LINEID
BREAK ON INVNBR
PRINT INVNBR GROUP 1
PRINT DATE GROUP 1
PRINT CUSTOMER(NAME) GROUP 1
PRINT INVAMT POSITION 48 GROUP 1
PRINT ' ' POSITION 5 GROUP 1
PRINT ITEMNBR POSITION 5
PRINT DETAIL(QTY)
PRINT PRODNBR
PRINT PRODUCT(DESC)
PRINT PRICE
PRINT EXTPRICE
GO
```

```
02-Mar-1994                    INVOICES                          Page 1

  Inv Invoice                                              Invoice   Extended
  Nbr Date     Customer name                                Amount      Price
----- -------- -------------------------       ---------- ----------
10301 88010100 Anywhere Travel Service          28,712.00

      1      1 31150 150MB disk drive             2,495.00   2,495.00
      2      2 22800 EZVue-800 CRT terminal         375.00     750.00
      3      1 24200 ZX-LQ printer                  299.00     299.00
      4      1 22800 EZVue-800 CRT terminal         375.00     375.00
      5      1 31020 20MB disk drive                595.00     595.00
      6      2 22800 EZVue-800 CRT terminal         375.00     750.00
      7      2 33090 90MB streaming tape drive    1,495.00   2,990.00
      8      8 31150 150MB disk drive             2,495.00  19,960.00
      9      2 28800 800KB floppy disk drive        249.00     498.00

10302 88010100 PacWest Corp                           447.00

      1      3 28400 400KB floppy disk drive        149.00     447.00

10303 88010100 Anywhere Travel Service            3,982.00

      1      1 33090 90MB streaming tape drive    1,495.00   1,495.00
      2      3 24200 ZX-LQ printer                  299.00     897.00
      3      2 31030 30MB disk drive                795.00   1,590.00
```

The following example shows 2 level breaks.

```
OPEN DETAIL
OPEN INVOICE
OPEN CUSTOMER
OPEN PRODUCT
RELATE DETAIL(INVNBR) TO INVOICE(INVNBR)
RELATE CUSTNBR TO CUSTOMER(CUSTNBR)
RELATE PRODNBR TO PRODUCT(PRODNBR)
SORT ON PRODNBR
SORT ON CUSTOMER(NAME)
SORT ON DATE
BREAK ON PRODNBR
BREAK ON NAME
PRINT PRODNBR GROUP 1
PRINT DESC GROUP 1 LENGTH 18
PRINT NAME GROUP 2 LENGTH 18
PRINT LINEID
PRINT QTY
PRINT PRICE
GO
```

```
02-Mar-1994                Report on DETAIL                    Page 1

                                                    Line
 Prod                                               Item  Order    Selling
  Nbr Product Descriptio Customer name               ID#    Qty      Price
----- ------------------ ------------------ --------- ------ ----------
20120 Bananna PC Jr      Alpha Products Inc 10376-003      7   1,495.00
                                            10394-002      3   1,495.00

                         Anywhere Travel Se 10340-004      1   1,495.00
                                            10340-008      2   1,495.00
                                            10400-004      1   1,495.00
                                .
                                .
                                .
                         Tumbleweeds Real E 10393-006      2   1,495.00

22600 EZVue-600 CRT term Alpha Products Inc 10314-002      1     325.00

                         Anywhere Travel Se 10348-003      3     325.00
                                            10400-003      1     325.00
```

6.19 PROGRAM

FORMAT:

```
PROGRAM program_name [UNIQUE]
```

EXAMPLE:

```
PROGRAM make_invoice_report
```

PURPOSE:

GQL normally creates a program with the name "GUIDE_RUN_xxxxxx.TMP" in your SYS$SCRATCH directory. If the report is to be used often, you can give this program a name so that it can be run from outside GQL. The default file extension is ".INT".

The **UNIQUE** qualifier adds a unique alpha-numeric string to the program extension to guarantee that program names will be unique. For example, the command **PROGRAM MAKE_INVOICE_REPORT UNIQUE** causes GQL to save the INTOUCH program in a filename similar to:

MAKE_INVOICE_REPORT.INT_3E0004

6.20 REFRESH

FORMAT:

```
REFRESH
```

EXAMPLE:

```
REFRESH
```

PURPOSE:

The **REFRESH** command erases the screen and redisplays the current titles and column headings.

6.21 RELATE

FORMAT:

```
RELATE structure_1(field) TO [rel_type] structure_2(keyfield)
RELATE [rel_type] structure_1(field) TO structure_2(key-field)
```

EXAMPLE:

```
RELATE line(invoice_no) TO invoice(invoice_no)
RELATE invoice(custnbr) TO customer(custno)
RELATE partial invoice(invoice_no) TO line(line_key)
```

PURPOSE:

The **RELATE** command sets up a lookup relationship between the primary structure (structure_1) and another structure (structure_2). Structure_2 is any structure that has not yet been related and contains information you need to extract. You must use the **RELATE** command so that GQL knows how to access the information in structure_2.

Here is an example where the **RELATE** command would be used.

You have opened the INVOICE structure which gives you the customer number (CUSTNBR). You now need the customer name (NAME) that corresponds to this customer number. The customer name is in the CUSTOMER file. To get the customer name, use the following command to relate the two structures:

RELATE invoice(custnbr) TO customer(custnbr)

For each INVOICE record, GQL uses the data in its CUSTNBR field to get the matching CUSTOMER record.

Field names do not need to be the same for the **RELATE** command to work. They need only contain the same type of data.

Structures MUST be related in the same order in which they were opened. For Example:

```
OPEN A
OPEN C
OPEN B
RELATE A(FIELD_1) TO C(FIELD_1)
RELATE C(FIELD_1) TO B(FIELD_1)
```

Example 6–23 Example of RELATE Command

```
OPEN DETAIL
OPEN INVOICE
OPEN CUSTOMER
RELATE DETAIL(INVNBR) TO INVOICE(INVNBR)
RELATE INVOICE(CUSTNBR) TO CUSTOMER(CUSTNBR)
PRINT CUSTOMER(CUSTNBR), NAME, LINEID, PRODNBR, QTY, EXTPRICE
GO
```

Example 6–24 Example of RELATE Command - Output

```
02-Mar-1994              Report on DETAIL                 Page 1

                                    Line
                                    Item  Prod  Order   Extended
Cust# Customer name                  ID#   Nbr    Qty      Price
----- ------------------------- --------- ----- ------ ----------
21670 Anywhere Travel Service   10301-001 31150      1   2,495.00
21670 Anywhere Travel Service   10301-002 22800      2     750.00
21670 Anywhere Travel Service   10301-003 24200      1     299.00
21670 Anywhere Travel Service   10301-004 22800      1     375.00
21670 Anywhere Travel Service   10301-005 31020      1     595.00
21670 Anywhere Travel Service   10301-006 22800      2     750.00
21670 Anywhere Travel Service   10301-007 33090      2   2,990.00
21670 Anywhere Travel Service   10301-008 31150      8  19,960.00
21670 Anywhere Travel Service   10301-009 28800      2     498.00
10546 PacWest Corp              10302-001 28400      3     447.00
21670 Anywhere Travel Service   10303-001 33090      1   1,495.00
21670 Anywhere Travel Service   10303-002 24200      3     897.00
21670 Anywhere Travel Service   10303-003 31030      2   1,590.00
21138 Peabody Plumbing          10304-001 33180      1   2,095.00
21138 Peabody Plumbing          10304-002 20120      2   2,990.00
```

6.21.1 RELATE Options—VAX/DBMS Structures

There are two additional **RELATE** options that are for use only with VAX/DBMS structures. These options allow you to specify a set to walk when relating structures.

FORMAT:

```
RELATE [rel_type] struct(field) TO lookup_str SET set_name [: EXPRESSION expr_text]
```

EXAMPLE:

```
RELATE component TO part SET 'part_uses' : EXPRESSION 'owner'
RELATE part TO many component SET 'part_used_on'
```

PURPOSE:

The two structures must each have a field with common information, such as an invoice number. The two fields do not have to have the same name. The field in the second structure (master) must be a key field. Each time you get to a new record, GQL uses the value in the first field to look up the master record in the second structure by means of the key field.

rel_type	Type of relationship to establish
	ONE - one to one relationship
	MANY - one to many relationship using duplicate keys
	PARTIAL - one to partial key
struct	Detail structure or synonym
lookup_str	Master structure or synonym
set_name	Set name to walk
expr_text	Expression to be passed to the database engine

Please refer to the "INTOUCH, INTERFACES TO VAX DATABASE ENGINES" manual for further information about accessing VAX/DBMS data.

6.22 SAVE

FORMAT:

```
SAVE filename
```

EXAMPLE:

```
SAVE client_list
```

PURPOSE:

The **SAVE** command writes the preceding commands to the filename. The default file extension is **.GUIDE**.

After you have completed creating a GQL report or changing a previously SAVE'd report, you can save the commands for later execution by using the **SAVE** command.

6.23 SET COMMANDS

The following sections describe the **SET** commands.

6.23.1 SET COMMANDS [SHORT | LONG]

FORMAT:

```
SET COMMANDS SHORT | LONG
```

EXAMPLE:

```
SET COMMANDS SHORT
SET COMMANDS LONG
```

PURPOSE:

By default, commands are truncated to fit when displayed with SHOW COMMANDS. **SET COMMANDS LONG** allows a longer command to be displayed in the "show" box.

6.23.2 SET CUTOFF

FORMAT:

```
SET CUTOFF max_records
```

EXAMPLE:

```
SET CUTOFF 50
```

PURPOSE:

The **SET CUTOFF** command limits the number of records to be selected. This feature is useful if you want to test the format of a report when using a large data file.

6.23.3 SET EDITOR

FORMAT:

```
SET EDITOR [INTERNAL | EXTERNAL]
```

PURPOSE:

You can choose the editor you want to use. **SET EDITOR INTERNAL** allows you to use GQL's editor. **SET EDITOR EXTERNAL** (the default) allows you to use the VMS editor.

6.23.4 SET GRANDTOTAL

FORMAT:

```
SET GRANDTOTAL 'desc_text' [POSITION nn] [NOTOTALS]
```

EXAMPLE:

```
SET GRANDTOTAL 'Extended price grand-total:' POSITION 5
```

PURPOSE:

SET GRANDTOTAL allows the user to define description text to be displayed on the grand total line.

The **POSITION nn** qualifier allows users to position the grand total description text.

The **NOTOTALS** qualifier suppresses the printing of grand totals.

Example 6–25 Example of SET GRANDTOTAL Command

```
OPEN DETAIL
SORT BY PRODNBR
SORT BY LINEID
INCLUDE INVNBR < 10330
EXCLUDE INVNBR = 10320
BREAK ON PRODNBR
SET GRANDTOTAL 'Extended price grand-total:' POSITION 2
PRINT PRODNBR GROUP 1
PRINT LINEID
PRINT QTY
PRINT PRICE
PRINT EXTPRICE WITH TOTALS
GO
```

Example 6–26 Example of SET GRANDTOTAL Command - Output

```
02-Mar-1994   Report on DETAIL        Page 1

           Line
 Prod      Item  Order    Selling   Extended
  Nbr       ID#    Qty      Price      Price
----- --------- ------ ---------- ----------
20120 10304-002      2   1,495.00   2,990.00
      10323-004      1   1,495.00   1,495.00
                                   ==========
                                     4,485.00

22600 10314-002      1     325.00     325.00
      10323-005      1     325.00     325.00
      10327-002      1     325.00     325.00
      10328-002      1     325.00     325.00
      10328-003      4     325.00   1,300.00
      10329-003      2     325.00     650.00
                                   ==========
                                     3,250.00
                         .
                         .
                         .
33180 10304-001      1   2,095.00   2,095.00
      10308-002      1   2,095.00   2,095.00
      10315-001      2   2,095.00   4,190.00
                                   ==========
                                     8,380.00
                                   ==========
 Extended price grand-total:       135,900.00
```

6.23.5 SET INPUT MESSAGE

FORMAT:

```
SET INPUT MESSAGE text_str
```

EXAMPLE:

```
SET INPUT MESSAGE Enter the date in MMDDYYYY format.
```

PURPOSE:

The **SET INPUT MESSAGE** command sets up a message to be used for the next input command entered into GQL. This message will be displayed by the report program generated by GQL. This message will be reset after the next input command.

6.23.6 SET INPUT ROUTINE

FORMAT:

```
SET INPUT ROUTINE routine_name
```

EXAMPLE:

```
SET INPUT ROUTINE check_account
```

PURPOSE:

The **SET INPUT ROUTINE** command allows the developer of a GQL procedure to specify that a user-written validation routine should be executed after the user responds to an input command.

The routine can perform sophisticated validations of the data that has been input.

The routine name specified in the **SET INPUT ROUTINE** command will be executed by the next input command entered.

The value that the user entered is contained in the variable U_REPLY$. The specified routine is called prior to loading the user's response into the target variable.

Upon completion of the routine, _ERROR is checked. If an error has been detected (_ERROR is TRUE), the input will be repeated. _ERROR can be set by either issuing a MESSAGE ERROR command or by SET ERROR ON.

If the variable U_REPLY$ is altered by the routine, the changed value will be stored into the target variable of the input command.

In the above example, the routine CHECK_ACCOUNT will be executed after the user inputs a value.

You can include user-written routines into your GQL report using the %INCLUDE command. Please refer to %INCLUDE for further information.

6.23.7 SET INTOUCH [ON | OFF]

FORMAT:

```
SET INTOUCH ON | OFF
```

EXAMPLE:

```
SET INTOUCH ON
SET INTOUCH OFF
```

PURPOSE:

Allows INTOUCH-like statements to be expressed in LET and PRINT commands. Words that GQL does not recognize as structure names, field names or keywords are used "as is" without generating error messages.

Note

This command should be used with caution.

6.23.8 SET LOOKUPERR

FORMAT:

```
SET LOOKUPERR character
```

PURPOSE:

When a file lookup is established with the RELATE command, there can be cases where the lookup is unsuccessful. In that case, a series of periods ('.') is used to fill the field to show that no matching record was found. You can use a different character with the **SET LOOKUPERR** option.

For example, to use an underline:

```
SET LOOKUPERR "_"
```

Or, to leave the column blank:

```
SET LOOKUPERR " "
```

6.23.9 SET MENU [ON | OFF]

FORMAT:

```
SET MENU ON | OFF
```

EXAMPLE:

```
SET MENU ON
SET MENU OFF
```

PURPOSE:

The **SET MENU** command sets the GQL menu interface ON or OFF. GQL can be run with either a menu interface or a command line interface. The default setting is determined by translating the logical "GUIDE_MENU_INTERFACE".

During execution of GQL, the user can change interfaces by using the **SET MENU** command. The menu interface can either be ON or OFF.

Note

The PF4 key can be used at anytime to ENTER or EXIT the GQL menu interface.

6.23.10 SET PROCEED

FORMAT:

```
SET PROCEED [TRUE | FALSE]
```

EXAMPLE:

```
SET PROCEED TRUE
SET PROCEED FALSE
```

PURPOSE:

Determines whether or not a GQL program with input commands will prompt the user with the "Proceed (Y/N)?" question.

If you want the report program to "ask proceed" after all other inputs are asked, then **SET PROCEED TRUE**. This is the default state for GQL.

If **SET PROCEED FALSE** is entered, GQL will produce the report after the last input has been processed.

6.23.11 SET PROMPT

FORMAT:

```
SET PROMPT 'prompt_text'
```

EXAMPLE:

```
SET PROMPT 'Command'
```

PURPOSE:

Use the **SET PROMPT** command to change the "Guide>" prompt to whatever you want to use.

6.23.12 SET REPROMPT

FORMAT:

```
SET REPROMPT [TRUE | FALSE]
```

EXAMPLE:

```
SET REPROMPT TRUE
SET REPROMPT FALSE
```

PURPOSE:

Determines whether GQL will reprompt the input commands or return to the Guide prompt upon completion of a report.

Entering **SET REPROMPT TRUE** causes the report program to reprompt the user for all inputs upon completion of the report. This reprompting will continue until the user responds to a question with "EXIT". This is the default state for GQL.

Entering **SET REPROMPT FALSE** causes GQL to return to its command mode after the user has finished displaying the report.

6.23.13 SET SCREEN

FORMAT:

```
SET SCREEN [80 | 132 | NARROW | WIDE]
```

EXAMPLE:

```
SET SCREEN 132
```

PURPOSE:

The **SET SCREEN** command changes the number of columns displayed to either 80 (narrow) or 132 (wide).

6.23.14 SET SPACING

FORMAT:

```
SET SPACING [SINGLE | DOUBLE | TRIPLE | nn]
```

EXAMPLE:

```
SET SPACING double
SET SPACING 5
```

PURPOSE:

Normally, the output report is single spaced. For double spacing, type **SET SPACING DOUBLE**, etc. For 10 blank lines between printed lines, type **SET SPACING 11**.

6.23.15 SET TSUPPRESS [ON | OFF]

FORMAT:

```
SET TSUPPRESS [ON | OFF]
```

EXAMPLE:

```
SET TSUPPRESS ON
SET TSUPPRESS OFF
```

PURPOSE:

If **TSUPPRESS** is set on, minor level break totals will be suppressed if there is only 1 item (e.g. line) printed on the report. Only the printing of the subtotals is suppressed, not the processing of the total amounts.

6.24 SHOW Commands

FORMAT:

```
SHOW COMMANDS
SHOW FIELDS [IN structure_name]
SHOW OPEN
SHOW STRUCTURES [IN device_directory_spec]
SHOW COLUMN nn
```

SHOW COMMANDS

Displays the commands currently entered.

The **SHOW COMMANDS** command instructs GQL to display all commands previously input by the user in a current session.

SHOW FIELDS [IN structure_name]

Displays the fields for the named structure. If no structure is named, the fields in the primary structure (the first one opened) are displayed. Key fields are in **BOLD** print.

SHOW FIELDS
SHOW FIELDS IN CUSTOMER

SHOW OPEN

Displays the structures currently open.

SHOW STRUCTURES

Displays the structures found in the GUIDE_NAMES structure.

SHOW STRUCTURES IN device_directory_spec

Displays a directory listing of structure files in the specified directory.

```
SHOW STRUCTURES IN tti_run:
SHOW STRUCTURES IN USER:[FRED]
```

SHOW COLUMN nn

Displays the numerical starting position (e.g., 33) and other information for a column.

6.25 SORT

FORMAT:

```
SORT [ASCENDING | UP | DESCENDING | DOWN]
     [ON | BY] structure(field) [, structure(field), ...]
```

EXAMPLE:

```
SORT DESCENDING BY customer(days_past_due)

SORT BY customer(custnbr)
```

PURPOSE:

The **SORT** command causes the records in the primary structure to be sorted in the order given.

The **SORT** command should precede the BREAK command if both are used.

ADDITIONAL:

SORT can order the records in either **ASCENDING** or **DESCENDING** order. ASCENDING creates a list in ascending order (lowest to highest) and is the default order. DESCENDING creates a list in descending order (highest to lowest).

GQL does sorts in order. Therefore, you can use multiple sorts to order the print on multiple levels. For example, if GQL is directed to first sort by product number and then by lineid, the products are sorted first by product number and then by line id.

Example 6–27 Example of SORT Command

```
OPEN CLIENT
INCLUDE CLIENT(ID) > '80500'
SORT CLIENT(LAST) ASCENDING
PRINT CLIENT(ID)
PRINT CLIENT(LAST)
PRINT CLIENT(FIRST)
PRINT CLIENT(CITY)
PRINT CLIENT(STATE)
GO
```

Example 6–28 Example of SORT Command - Output

```
03-Mar-1994               Report on CLIENT                    Page 1

CLNT  Last              First
ID    Name              Name            City                       ST
----- ----------------- --------------- -------------------------- --
80542 Brock             Bud             Duluth                     MN
80543 Cass              Cathy                                      CA
80561 Derringer         Dale                                       CA
80522 Errant            Earl                                       CA
80573 Farmer            Fred            Miami                      FL
80544 Porter
```

6.26 SUMMARY

FORMAT:

```
SUMMARY
```

EXAMPLE:

```
SUMMARY
```

PURPOSE:

The **SUMMARY** command causes GQL to generate a summary report. The summary report is a shorter form of the full (detailed) report. In a summary report, a line is printed every time the lowest level break occurs, instead of printing every record.

Example 6–29 Example of SUMMARY Command, Part 1 - Full Detail

```
OPEN DETAIL
SORT BY PRODNBR
SORT BY LINEID
INCLUDE INVNBR < 10330
EXCLUDE INVNBR = 10320
BREAK ON PRODNBR
SET GRANDTOTAL 'Extended price grand-total:' POSITION 2
PRINT PRODNBR GROUP 1
PRINT LINEID
PRINT QTY
PRINT PRICE
PRINT EXTPRICE WITH TOTALS
GO
```

Example 6–30 Example of SUMMARY Command, Part 1 - Full Detail - Output

```
03-Mar-1994   Report on DETAIL        Page 1

           Line
 Prod      Item  Order    Selling   Extended
  Nbr       ID#    Qty      Price      Price
----- --------- ------ ---------- ----------
20120 10304-002      2   1,495.00   2,990.00
      10323-004      1   1,495.00   1,495.00
                                   ==========
                                     4,485.00

22600 10314-002      1     325.00     325.00
      10323-005      1     325.00     325.00
      10327-002      1     325.00     325.00
      10328-002      1     325.00     325.00
      10328-003      4     325.00   1,300.00
      10329-003      2     325.00     650.00
                                   ==========
                                     3,250.00
                         .
                         .
                         .
33090 10301-007      2   1,495.00   2,990.00
      10303-001      1   1,495.00   1,495.00
      10311-007      1   1,495.00   1,495.00
      10322-004      4   1,495.00   5,980.00
      10325-002      2   1,495.00   2,990.00
                                   ==========
                                    14,950.00

33180 10304-001      1   2,095.00   2,095.00
      10308-002      1   2,095.00   2,095.00
      10315-001      2   2,095.00   4,190.00
                                   ==========
                                     8,380.00
                                   ==========
 Extended price grand-total:      135,900.00
```

Example 6–31 Example of SUMMARY Command, Part 2 - Summary

```
OPEN DETAIL
SUMMARY
SORT BY PRODNBR
SORT BY LINEID
INCLUDE INVNBR < 10330
EXCLUDE INVNBR = 10320
BREAK ON PRODNBR
SET GRANDTOTAL 'Extended price grand-total:' POSITION 2
PRINT PRODNBR GROUP 1
PRINT LINEID
PRINT QTY
PRINT PRICE
PRINT EXTPRICE WITH TOTALS
GO
```

Example 6–32 Example of SUMMARY Command, Part 2 - Summary - Output

```
03-Mar-1994   Report on DETAIL        Page 1

           Line
 Prod      Item  Order    Selling   Extended
  Nbr       ID#    Qty      Price      Price
----- --------- ------ ---------- ----------
20120 10323-004      1   1,495.00   4,485.00
22600 10329-003      2     325.00   3,250.00
22800 10325-003     10     375.00   8,625.00
24100 10329-002      1     199.00   2,786.00
24200 10329-004      3     299.00   3,887.00
28120 10329-005      2     379.00   6,443.00
28400 10327-001      3     149.00   1,639.00
28800 10329-007      1     249.00   8,715.00
30170 10329-001      2   1,595.00  15,950.00
31020 10328-005      9     595.00   7,140.00
31030 10321-001      2     795.00   4,770.00
31040 10326-002      2     995.00   9,950.00
31150 10329-006      1   2,495.00  34,930.00
33090 10325-002      2   1,495.00  14,950.00
33180 10315-001      2   2,095.00   8,380.00
                                  ==========
 Extended price grand-total:      135,900.00
```

6.27 TITLE

FORMAT:

```
TITLE 'title_text'
```

EXAMPLE:

```
TITLE 'Daily Invoice Report'
TITLE "Fred's Customer List"
```

PURPOSE:

The **TITLE** command sets up a report title which will be printed at the top of each page of the report.

ADDITIONAL:

Each **TITLE** command sets up a successive title line, up to a maximum of nine (9) title lines.

Until a title is entered for a report, a default title is assigned. The default title is "Report On <structure_name>". One to no more than nine title lines are permitted.

7 The Guided Query Language Vocabulary File

The Guided Query Language includes a natural language interface. This interface acts to translate everyday words and phrases into computer terminology. The user is freed from having to learn computer jargon.

To facilitate this, GQL includes a vocabulary file. This vocabulary file contains a list of valid phrases and throwaway words (such as "the" and "a"). The name of the vocabulary file is:

TTI_RUN:GUIDE_VOCAB.TAB

The vocabulary file can be modified in order to add new phrases or to ignore certain words. For example, if your users are comfortable with the word ROLLUP instead of SUMMARY, you can modify the vocabulary file to indicate that ROLLUP = SUMMARY. In other words, your vocabulary file can be customized for a particular use (e.g., department) at any time.

The vocabulary file can be modified in order to add new phrases or to ignore certain words.

Note

You must have sufficient privileges in order to modify the vocabulary file.

Some examples of synonyms are listed below:

Example 7–1 Example of Some Synonyms

```
Word                Synonym             Contexts
_______________________________________________________________

open,               $open,              all
ouvrez,             $open,              all

sort,               $sort,              all
arrange,            $sort,              all

list,               $list,              all
print,              $list               all

omit,               $exclude,           all
skip,               $exclude,           all
```

In open (the file), "the" and "file" are examples of throwaway words.

A summary of The Guided Query Commands can be found in Appendix A. Appendix A also contains an example of the vocabulary file.

GQL comes packaged with a built-in vocabulary.

A The Guided Query Language Summary of Commands

A.1 Commands Affecting Output

APPEND *filename.ext*

BREAK [ON] *expression [, ...] [DESCRIPTION 'text' [POSITION nn]] [PAGE] [NOTOTALS]*

CONDITION *varname IS expression*

EXCLUDE *expression [, expression[, ...]]*

EXCLUDE FOUND *structure_name*

IF THEN *if expression then ..... else ..... endif*

INCLUDE *expression [, expression [, ...]]*
INCLUDE FOUND *structure_name*
INCLUDE *keyfield = value1 TO value2*

KEY *keyfield = expression*

LET *varname = expression*

NEW

OLD *GQL_filename*

OPEN *structure_synonym_in_GUIDE_NAMES*
OPEN *structure_name*
OPEN *structure_synonym NAME filename*

OUTPUT *output_filename*

PRINT *expression [, ...]*
PRINT *[GROUP break_level]*
PRINT *[HEADING "heading_expression1" ["hdg_expr2" ["hdg_expr3"]]]*
PRINT *[MASK "edit mask"]*
PRINT *[LENGTH field_length]*
PRINT *[POS[ITION] [column_position | +preceding_spaces | =column_nbr]]*
PRINT *[[WITH] TOTALS [= totals_expression]]*

PROGRAM *program_filename* [UNIQUE]

RELATE *str_name(field) TO lookup_str_name(field)*

SAVE *GQL_filename*

SET COMMANDS *[SHORT | LONG]*
SET CUTOFF *max_records*
SET EDITOR *[INTERNAL | EXTERNAL]*
SET GRANDTOTAL *"text" [POSITION nn] [NOTOTALS]*
SET INPUT MESSAGE *text_str*
SET INPUT ROUTINE *routine_name*
SET INTOUCH *[ON|OFF]*
SET LOOKUPERR *"substitute_character"*
SET MENU *[ON|OFF]*
SET PROCEED *[TRUE|FALSE]*
SET PROMPT *"new_prompt"*
SET REPROMPT *[TRUE|FALSE]*
SET SCREEN *[80|132|WIDE|NARROW]*
SET SPACING *[SINGLE|DOUBLE|TRIPLE|1|2| 3|blanks-1]*
SET TSUPPRESS *[ON|OFF]*

SORT *[ON|BY] [UP|ASCENDING|DOWN|DESCENDING] expression [, ...]*

TITLE *"page_heading1" ["page_heading2" ["page_heading3"]]*

A.2 Print Expressions

PRINT *field*
PRINT *field1 + field2 (etc.)*
PRINT CUME *expression*
PRINT PERCENT *expression OF [TOTAL] expression*

A.3 Commands Affecting Screen Display

SHOW COMMANDS
SHOW OPEN
SHOW FIELDS *[[IN] structure_name]*
SHOW COLUMN *column_number*
SHOW STRUCTURES
SHOW STRUCTURES *IN device_directory_spec*
SHOW *command_name*

RESTART

REMOVE *command_name*

REFRESH

EDIT

HELP

SET SCREEN *[80|132|WIDE|NARROW]*
SET PROMPT *new_gql_prompt*
SET COMMANDS *[SHORT|LONG]*
SET DISPLAY *[ON|OFF]*

A.4 Adding Synonyms

To create your own synonyms, do the following:

- At the VAX prompt (the "$"), invoke the editor by entering EDIT and the vocabulary file name:

```
$ EDIT TTI_RUN:GUIDE_VOCAB.TAB
```

- Using the INSERT mode, insert your synonym directly below the last synonym given for the word.

Example:

Suppose you want to add the synonym SKIP for the command EXCLUDE:

- Arrow to the line where the EXCLUDE command is found, e.g.:

```
except,     $exclude,     all
```

- Directly beneath the line, add your new synonym. Your text will look as follows:

```
except,     $exclude,     all
skip,       $exclude,     all
```

Now, if you enter your new synonym (command) SKIP at the command prompt, you will find that it does exactly the same thing as the EXCLUDE command.

B Guided Query Language System Options

B.1 GQL System Options

The following are GQL System Options:

- **NOSYSTEM**

  Prevents access to VMS from GQL.

- **INTERNAL EDITOR**

  Causes GQL to use its internal editor rather than using the VMS editor.

- **MENU ON**

  Causes the GQL menu interface to be the default.

FORMAT:

```
$ DEFINE GUIDE_OPTIONS "Option1, Option2, Option3"
```

EXAMPLES:

```
$ DEFINE GUIDE_OPTIONS "NOSYSTEM"
$ DEFINE GUIDE_OPTIONS "INTERNAL EDITOR"
$ DEFINE GUIDE_OPTIONS "NOSYSTEM, INTERNAL EDITOR"
$ DEFINE GUIDE_OPTIONS "MENU ON"
```

B.2 NOSYSTEM

The **NOSYSTEM** option prevents access to VMS from GQL.

When users try to access VMS by entering "$$" at the Guide prompt, they will see the error message, "System access is not allowed":

Example B–1 VMS Access Error Message

```
Guide> $$

                          System access is not allowed
EXIT = Exit              INTOUCH Guided Query Language                HELP = Help
```

B.2.1 INTERNAL EDITOR

The **INTERNAL EDITOR** option causes GQL to use its internal editor rather than using the VMS editor. Use this option if you do not want users to access a VMS editor.

B.3 MENU ON

GQL has two user interfaces available:

- a pop-up menu interface
- a command line interface

The command line interface is the default when GQL is run. If you want the menu interface to be the default, use the MENU ON option.

C Sample GQL Procedure to Access DBMS Data

The following GQL procedure produces a report using several DBMS record types to illustrate the different methods of accessing DBMS data:

```
open devdisk:[devint.dbms]employee systemset 'all_employees'
open devdisk:[devint.dbms]part
open devdisk:[devint.dbms]component
open devdisk:[devint.dbms]class
relate employee to many part set 'responsible_for'
relate part to many component set 'part_uses'
relate part to class set 'class_part' expression 'owner'
break emp_id nototals
break part_id
print emp_id group 1, emp_first_name group 1, emp_last_name group 1
print part_id group 2
print class(class_code)
print comp_sub_part
```

Index

$$ • 3–5

A

Ampersand • 6–1
APPEND • 6–2
Arrays
  Printing Elements of • 6–21
Arrow keys • 1–2
ASCENDING order • 6–44

B

BREAK example • 6–3
BREAK [ON] • 6–2
  DESCRIPTION • 6–2
  NOTOTALS • 6–2
  PAGE • 6–2
  POSITION • 6–2

C

Changing the report file • 3–1
Command file name • 3–4
Commands • 6–1
COMMANDS
  APPEND • 6–2
  BREAK [ON] • 6–2
  EDIT • 6–6
  EXCLUDE • 6–7
  EXCLUDE FOUND • 6–8
  GO • 6–9
  HELP • 6–10
  IF THEN • 6–10
  INCLUDE • 6–12
  %INCLUDE • 6–1
  INCLUDE FOUND • 6–13
  INPUT • 6–14
  LET • 6–16
  NEW • 6–18
  OLD • 6–18
  OPEN • 6–19
  OUTPUT • 6–20
COMMANDS (cont'd)
  PRINT • 6–20
  PROGRAM • 6–30
  REFRESH • 6–30
  RELATE • 6–31
  SAVE • 6–33
  SET • 6–34
  SET COMMANDS [SHORT | LONG] • 6–34
  SET CUTOFF • 6–35
  SET EDITOR • 6–35
  SET GRANDTOTAL • 6–35
  SET INPUT MESSAGE • 6–37
  SET INPUT ROUTINE • 6–38
  SET INTOUCH [ON | OFF] • 6–38
  SET LOOKUPERR • 6–39
  SET MENU [ON | OFF] • 6–40
  SET PROCEED [TRUE | FALSE] • 6–40
  SET PROMPT • 6–41
  SET REPROMPT [TRUE | FALSE] • 6–41
  SET SCREEN • 6–42
  SET SPACING • 6–42
  SET TSUPPRESS [ON | OFF • 6–42
  SHOW • 6–43
  SORT • 6–44
  SUMMARY • 6–45
  TITLE • 6–49
Commands affecting output • A–1
Commands affecting screen display • A–3
Commonly used commands • 5–2
CUME • 6–21

D

Data structure • 2–2
DBMS structures • 6–19, 6–33, C–1
Define INTERNAL EDITOR • B–2
DESCENDING order • 6–44
DESCRIPTION qualifier • 6–4
DESCRIPTION qualifier for BREAK • 6–3

E

EDIT • 3–1, 6–6

Editing files • 3–1
Editor • 6–7
Enter/Exit GQL • 3–5
_ERROR • 6–38
EXCLUDE • 2–3, 5–4, 6–7
EXCLUDE example • 6–8
EXCLUDE FOUND • 6–8
EXIT • 2–14, 5–1, 5–6
Exiting GQL • 2–14, 5–6

F

Field display • 5–4
Fields to print • 2–8

G

General information • 5–1
Generate the report • 2–11, 5–5
GO • 2–11, 5–2, 5–5, 6–9
GQL Command File example • 3–1
GQL commands • 6–1
GQL training • 4–1
GROUP print option • 6–26
.GUIDE extension • 3–5, 6–34
Guide Options menu • 1–2
GUIDE_LIST_xxxxxx.TMP • 6–20
GUIDE_MENU_INTERFACE logical • 6–40
GUIDE_NAMES file • 6–19
GUIDE_NAMES structure • 6–43
GUIDE_RUN_xxxxxx.TMP • 6–30

H

HEADING print option • 6–24
HELP • 5–1, 6–10

I

IF THEN • 6–10
IF THEN example • 6–11
INCLUDE • 2–3, 5–2, 5–4, 6–12
%INCLUDE • 6–1, 6–38
INCLUDE example • 6–12
INCLUDE FOUND • 6–13
Include Options and Fields menu • 2–4
Include Options menu • 2–5
INCLUDE range of values • 6–12
INPUT • 6–14
INPUT example • 6–14
INTERNAL EDITOR • B–2
.INT extension • 6–30
INTOUCH TRAINING SYSTEM • 4–1

L

LENGTH print option • 6–24
LET • 6–16
LET example • 6–16
Level breaks • 6–2
Line continuation • 6–1
.LIS extension • 6–20

M

MASK print option • 6–24
Menu interface • 1–1
MENU ON • B–2
Menu options • 1–2
Mock report • 5–5
MORE submenu • 3–4

N

NEW • 5–2, 5–3, 6–18
NOSYSTEM • B–1
NOTOTALS qualifier for BREAK • 6–3
NOTOTALS qualifier for SET GRANDTOTAL • 6–36

O

OLD • 6–18
OPEN • 2–2, 5–2, 5–3, 6–19
Opening structures • 2–2, 5–3
OUTPUT • 6–20
Output options • 5–5

Output Options menu • 2–12

P

PAGE qualifier for BREAK • 6–3
Passing to VMS • 3–5
PERCENT • 6–21
PF4 key • 3–5
POSITION print option • 6–25
POSITION qualifier • 6–5
POSITION qualifier for BREAK • 6–3
POSITION qualifier for SET GRANDTOTAL • 6–36
Primary structure • 6–19, 6–31
Print
  Elements of Arrays • 6–21
PRINT • 2–8, 5–2, 5–5, 6–20
  GROUP option • 6–23, 6–26
  HEADING option • 6–23, 6–24
  keyword CUME • 6–21
  keyword PERCENT • 6–21
  LENGTH option • 6–23, 6–24
  MASK option • 6–23, 6–24
  POSITION option • 6–23, 6–25
  TOTALS option • 6–23, 6–26
PRINT CUME • 6–21
PRINT example • 6–22
Print expressions • A–3
PRINT GROUP examples • 6–26
PRINT keyword • 6–20
PRINT keywords • 6–21
Print menu • 2–8
PRINT options • 6–20, 6–23
PRINT PERCENT • 6–21
PRINT PERCENT OF [TOTAL] • 6–21
PRIVILEGES • 7–1
PROGRAM • 6–30

R

Record selection • 2–3
Record Selection submenu • 2–3
REFRESH • 6–30
RELATE • 6–31
RELATE example • 6–32
RELATE options for DBMS structures • 6–33
Report generation • 2–11
Report generation statistics • 2–11
Report output options • 2–12
Report title • 2–1
Running saved report files • 3–6

S

Sample report • 2–1, 3–1
SAVE • 3–4, 5–2, 5–6, 6–33
Saving files • 3–4, 5–6
SELECT • 2–3, 2–4
Selecting menu items • 1–2
Selecting records • 5–4
SET COMMANDS • 6–34
SET COMMANDS [SHORT | LONG] • 6–34
SET CUTOFF • 6–35
SET EDITOR • 6–35
SET GRANDTOTAL • 6–35
  NOTOTALS • 6–36
  POSITION • 6–36
SET GRANDTOTAL example • 6–36
SET INPUT MESSAGE • 6–37
SET INPUT ROUTINE • 6–38
SET INTOUCH [ON | OFF] • 6–38
SET LOOKUPERR • 6–39
SET MENU OFF • 6–40
SET MENU ON • 1–1, 6–40
SET PROCEED [TRUE | FALSE] • 6–40
SET PROMPT • 6–41
SET REPROMPT [TRUE | FALSE] • 6–41
SET SCREEN • 6–42
SET SPACING • 6–42
SET TSUPPRESS [ON | OFF] • 6–42
SHOW COLUMN nn • 6–43, 6–44
SHOW COMMANDS • 6–43
SHOW FIELDS • 5–2, 5–4
SHOW FIELDS [IN structure_name] • 6–43
SHOW OPEN • 6–43
SHOW STRUCTURES • 6–43
SHOW STRUCTURES IN device_directory_spec • 6–44
SHOW STRUCTURES [IN device_directory_spec] • 6–43
SORT • 2–6, 5–2, 5–4, 6–44
SORT example • 6–44
Sorting fields • 2–6
Sorting records • 5–4
Sort Options and Fields menu • 2–6
Sort Options menu • 2–7

Starting a new report • 5–3
Starting GQL • 1–1
Structures • 2–2
Structures menu • 2–2
Subtotals • 6–3
SUMMARY • 6–45
SUMMARY example • 6–46
Summary of commands • A–1
Synonyms • 6–1
SYNONYMS
  adding • A–4
  Examples of • 7–1
SYS$SCRATCH directory • 6–30
System options • B–1

T

Throwaway words • 7–1
TITLE • 2–1, 6–49
Title Options menu • 2–1
TOTALS print option • 6–26
TTI_RUN:GUIDE_VOCAB.TAB • 7–1, A–4
Tutorials • 4–1

U

UNIQUE qualifier
  OUTPUT • 6–20
  PROGRAM • 6–30
Using the TRAINING SYSTEM • 4–1
U_REPLY$ • 6–38

V

VMS • 5–2
VMS Access Error Message • B–2
VMS from GQL • 3–5
Vocabulary file • 7–1
  Modification of • 7–1