AT&T 585-350-915
Issue 1
Comcode 407298660

SQL*FORMS® OPERATOR'S GUIDE

VERSION 3.0

ORACLE®

The Relational Database Management System

SQL*Forms Operator's Guide
Version 3.0

Part No. 3301-V3.0 0490

Contributing Authors Susan K. Jackson, Kathleen Gomoll

Contributor Chris Schock

Copyright © Oracle Corporation 1986, 1990

All rights reserved. Printed in the U.S.A.

Restricted Rights Legend

Use, duplication, or disclosure is subject to restrictions stated in your contract with Oracle Corporation.

Use, duplication, or disclosure by the Government is subject to restrictions for commercial computer software and shall be deemed to be Restricted Rights software under Federal Law.

The information in this document is subject to change without notice. If you find any problems in the documentation, please report them to us in writing. Oracle Corporation does not warrant that this document is error-free.

ORACLE and SQL*Forms are registered trademarks of Oracle Corporation.

PREFACE

This guide explains how to operate applications built using SQL*Forms. After reading this guide, you will know how to complete the following tasks:

- run forms
- use your keyboard to make selections and edit entries
- retrieve records from the ORACLE database using search criteria
- enter, modify, and delete records
- record transactions in the ORACLE database

This guide does not teach you how to design or modify forms. To design or modify forms, refer to the *SQL*Forms Designer's Reference* or the *SQL*Forms Designer's Tutorial.*

SQL*Forms Documentation Set

The SQL*Forms documentation set for Version 3.0 consists of the Designer's Help System and the following five documents:

- *SQL*Forms Operator's Guide,* Version 3.0, Oracle Part No. 3301-V3.0
- *SQL*Forms Designer's Tutorial,* Version 3.0, Oracle Part No. 3302-V3.0
- *SQL*Forms Designer's Reference,* Version 3.0, Oracle Part No. 3304-V3.0
- *SQL*Forms Operator's Quick Reference,* Version 3.0, Oracle Part No. 3704-V3.0
- *SQL*Forms Designer's Quick Reference,* Version 3.0, Oracle Part No. 3708-V3.0

Each document has a specific purpose and audience. The following table briefly presents each piece of the set, its intended audience, and any suggested prerequisite reading.

Document	***Primary Audience***	***Pre-Requisite Reading***
Operator's Guide	novice operator	none
Designer's Tutorial	novice designer	*Operator's Guide*
Designer's Help System	novice designer	*Designer's Tutorial*
Designer's Reference	advanced designer	*Designer's Tutorial*
Operator's Quick Reference	advanced operator	*Operator's Guide*
Designer's Quick Reference	advanced designer	*Designer's Tutorial*

How This Guide Is Organized

This guide introduces you to SQL*Forms from an operator's point of view. After an overview of the product, there is a series of annotated exercises that illustrate the general procedures needed to run applications built using SQL*Forms.

The following are the major sections of this book:

Chapter 1	This chapter presents an overview of SQL*Forms.
Chapter 2	This chapter includes the general procedure for running a form, as well as the specific steps for running the sample form used throughout this guide.
Chapter 3	This chapter demonstrates how to use a form to retrieve information from the ORACLE database.
Chapter 4	This chapter shows you how to move the cursor around the form.
Chapter 5	This chapter demonstrates how to execute simple queries that meet specific criteria and complex queries satisfying several conditions.
Chapter 6	This chapter demonstrates how to modify data in the database.
Chapter 7	This chapter contains information about the way SQL*Forms processes transactions.

Three reference appendixes, a glossary, and a comprehensive index are also included.

How to Use This Guide

This guide is designed to teach you how to operate SQL*Forms applications. To make learning easier, exercises are included that are based on a sample form. Reading the guide at your computer allows you to work through the exercises and see the results on your screen.

For the keystroke exercises to work properly, you must complete all of them at one time. Otherwise you will not see the proper results.

Conventions Used in This Guide

The following conventions are observed in the *SQL*Forms Operator's Guide:*

- Function key names appear in square brackets as in [Next Field]. Look at the keypad map to see how function keys correspond to the keys on your terminal.

ž Text that is displayed on the screen and text that you should type at your terminal appears in a special font:

```
Example of special font.
```

Your Comments Are Welcome

Oracle Corporation knows the value of good documentation. One of the ways we continue to improve our products is by asking for your feedback on product documentation.

Please contact us if you have comments on this guide or on any piece of the SQL*Forms documentation set. Use the Reader's Comment Form at the back of this book, or call 415-598-8000 and ask for the SQL*Forms Product Manager.

CONTENTS

CHAPTER

1 OVERVIEW

This chapter provides an important overview of SQL*Forms, and includes the following topics:

• setup requirements
• definition of a form
• forms and the ORACLE database
Ž screen format
Ž function keys
• acknowledging alerts
Ž getting help

Setup Requirements

Before you can run SQL*Forms, the ORACLE Relational Database Management System (RDBMS) must be installed on your computer system. (Refer to the ORACLE Installation and User's guide for your system if the RDBMS or SQL*Forms has not been installed.)

You will need a username and password. A username identifies you as an authorized ORACLE user, and a password proves that you are the legitimate owner of your username. If your computer system is shared by others, get your username and password from the database administrator (DBA).

Setting Up the Sample Database

To setup the sample database, you will need to execute the DEMOBLD command file. To execute this file, type DEMOBLD at your operating system's command prompt.

Note: If typing DEMOBLD does not set up the sample database, your DEMOBLD command file may be stored in a different directory. Ask your DBA where the DEMOBLD command file is stored. Move to the proper directory and then type DEMOBLD.

In addition to setting up the sample database, the DEMOBLD file grants you select, update, and delete privileges for the sample database. Thus, you will be allowed to retrieve, change, or delete information in the sample database.

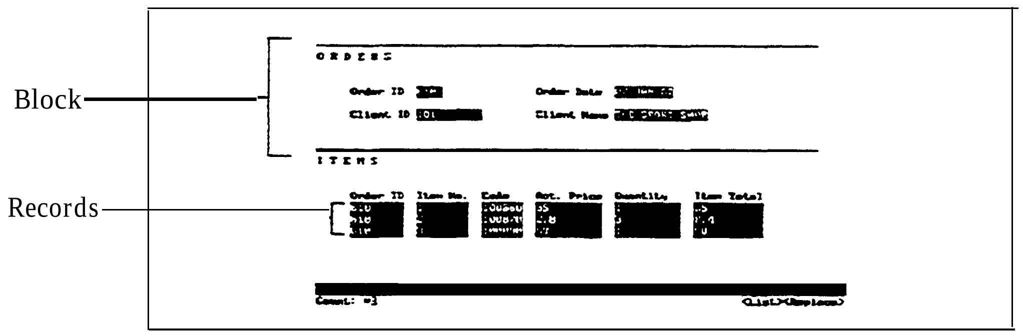
The sample form, SAMPLE, should be located in your ORACLE directory or on your ORACLE disk

Note: Because a form designer builds a form for a specific function, some forms may contain special features that make them behave differently from the sample form in this guide. See your DBA for information about any special features of your form.

What Is a Form?

A form is a fill-in-the-blanks template on your computer screen that allows you to enter, update, and query information in a database. Forms are composed of blocks, records, and fields.

FIGURE 1-1
Sample Form
Page 1



ORDERS

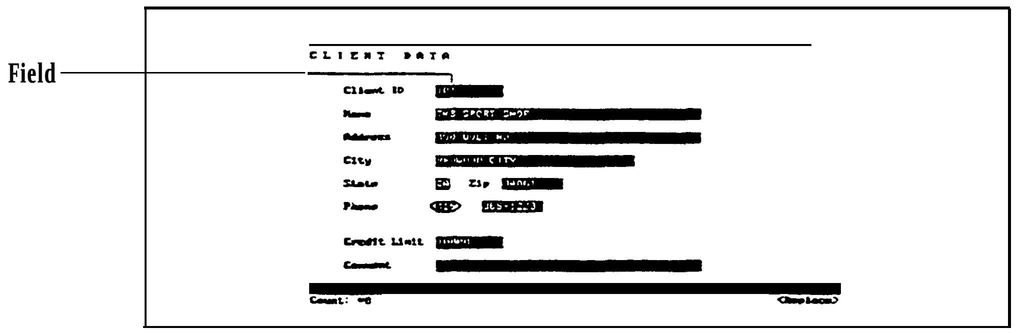
Order ID [illegible] Order Date [illegible]
Client ID 101 Client Name [illegible] SPORT SHOP

ITEMS

Order ID	Item No.	Code	Act. Price	Quantity	Item Total
310	[illegible]	[illegible]	35	[illegible]	[illegible]
310	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
310	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

Count: *3 <List><Replace>

FIGURE 1-2
Sample Form
Page 2



CLIENT DATA

Client ID [illegible]
Name THE SPORT SHOP
Address [illegible]
City [illegible] CITY
State [illegible] Zip [illegible]
Phone [illegible] [illegible]

Credit Limit [illegible]
Comment

Count: *0 <Replace>

Block	A group of related fields on a form.
Record	The data from a row in a database or non-database table.
Field	An area on the screen (usually highlighted) that can display a value or accept an input value. A field normally represents a column from a database table.

Forms and the ORACLE Database

When you run SQL*Forms, you will be entering modifying or deleting data contained in an ORACLE database. The ORACLE database organizes data into tables of related information. These tables are made up of rows and columns. (See the sample table below.)

On a form, a record corresponds to a row in a table. For example, the fourth row of the ITEM Table (Figure 1-3) is the first record in the ITEMS block (Figure 1-4).

FIGURE 1-3
ITEM Table

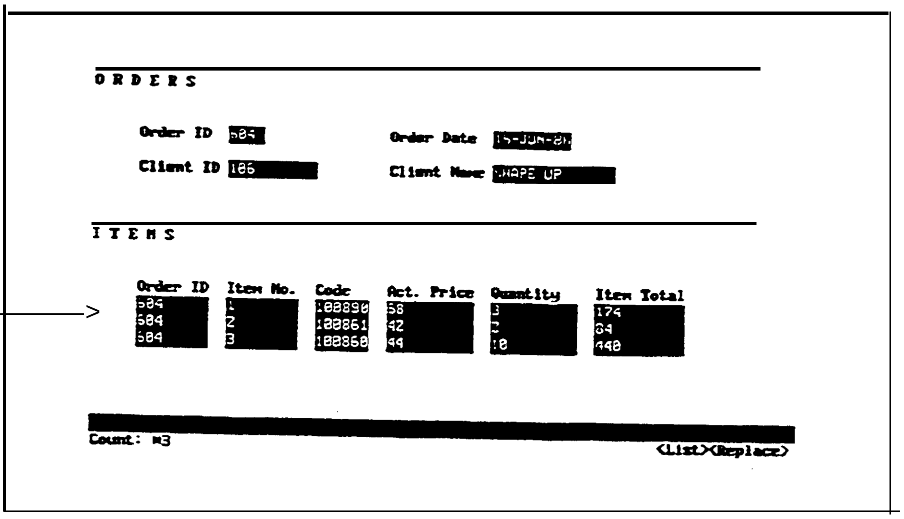
ORDID	ITEMID	PRODID	ACTUALPRICE	QTY	ITEMTOT
601	1	200376	2.4	1	2.4
602	1	100870	2.8	20	56
603	2	100860	56	4	224
604	1	100890	58	3	174
604	2	100861	42	2	84
604	3	100860	44	10	.440
605	1	100861	45	100	4500
605	2	100870	2.8	500	1400
605	3	100890	58	5	290
605	4	101860	24	50	1200
605	5	101863	9	100	900
605	6	102130	3.4	10	34

Row

Column

FIGURE 1-4
ITEMS Block

Record



ORDERS

Order ID 604 Order Date 15-JUN-8[illegible]

Client ID 106 Client Name SHAPE UP

ITEMS

Order ID	Item No.	Code	Act. Price	Quantity	Item Total
604	1	100890	58	3	174
604	2	100861	42	2	84
604	3	100860	44	10	440

Count: *3 <List><Replace>

When you retrieve data into a form (execute a query), each record that is displayed comes from a row in a table. Therefore, if you enter, modify, or delete a record, your action causes a corresponding row in a table to be entered, modified, or deleted. Figure 1-5 lists elements of the database and how they correspond to elements of a form.

Database Element	*Form Element*
Table	Block
Row	Record
column	Field

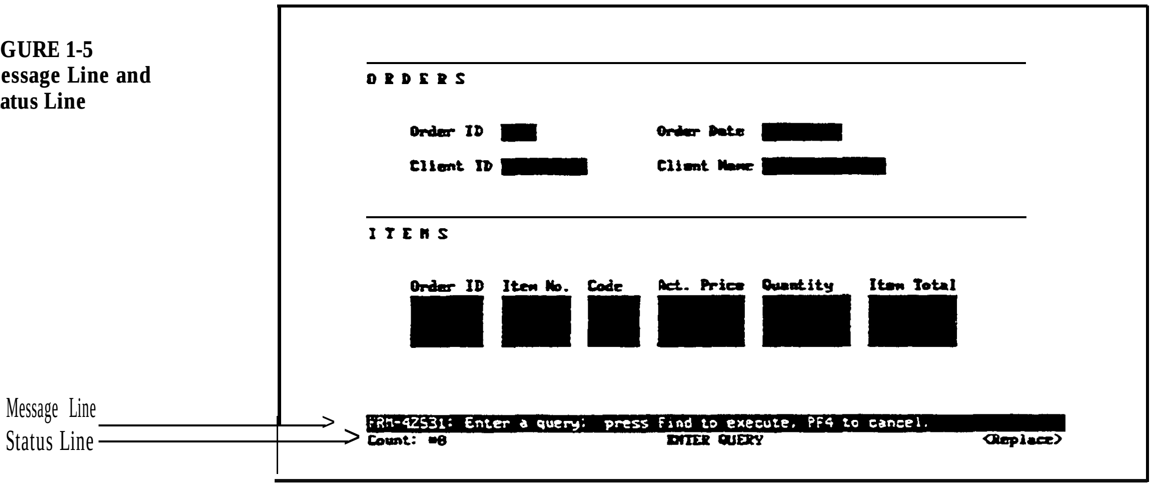
The Work Space

SQL*Forms does not work with database tables directly it works with copies of them that are kept in a work space. This arrangement protects you from making mistakes; you can make a change to the work space and then discard it. If you discard the changes, the table itself will not be affected.

Screen Format

A typical SQL*Forms screen displays explanatory text, graphic elements, and fields that accept data values. At the bottom of the screen are the message line and the status line.

FIGURE 1-5
Message Line and Status Line



```
FRM-42531: Enter a query:  press Find to execute, PF4 to cancel.
Count: *0                        ENTER QUERY                        <Replace>
```

The Message Line

The message line displays SQL*Forms messages.

The Status Line

From left to right, the status line may contain:

Count:	Indicates the number of records retrieved by a query. Each time you display a record fetched by a query, the count is increased. When you have fetched the last record, an asterisk (*) is displayed before the count.
^	Indicates that there are records before the current record in the block.
v	Indicates that there are records after the current record in the block.
ENTER QUERY	Indicates that you have pressed [Enter Query] and have not yet pressed [Execute Query] or [Exit/Cancel].
<List>	Displays when there is a list of values associated with the current field.
<Insert> or <Replace>	Displays the current character mode, either Insert or Replace.

Keyboard Map

When you are running SQL*Forms, you will press certain keys to move the cursor, enter and modify data, or signal your computer to store information. These keys are called function keys because they carry out the functions of SQL*Forms.

Because SQL*Forms runs on many different computers, a function (such as [Next Field]) may not be represented by the sequence of keystrokes on all computers. To solve this problem, SQL*Forms lists function names and their associated keys on a keyboard map.

The following are ways to display the keyboard map for the keyboard you are using:

1. Refer to the standard keyboard maps in your *ORACLE Installation and User's Guide.*
2. Ask your DBA for a copy of the keyboard map for your application (if it is different from the standard keyboard map for your terminal).
3. Display the keyboard map while you are running SQL*Forms by pressing [Show Keys].

Keyboard maps can be customized to fit the specific needs of a business organization. For this reason, you will see function names rather than key names in this Guide.

See Appendix A or the *SQL*Forms Operator's Quick Reference* for a description of the behavior of each function key.

Using a Mouse with SQL*Forms

For those environments that allow the use of a mouse, it is not always necessary to use function or cursor keys. SQL*Forms supports all mouse-driven functions, such as navigation and scrolling. For instance, you can use a mouse to select menu items, click on fields, or scroll an editable field.

Acknowledging Alerts

SQL*Forms alerts appear as one of two types of pop-up windows that partially overlay the current screen. To respond to the first type of alert, use the cursor keys to move to the proper answer (Yes, No, or Cancel) and then press [Select]. For the second type of alert, for which “OK” is the only response, you only have to press [Select] to acknowledge that you read the message.

Getting Help

When you are running a form, you can receive help by pressing [Help], [Show Keys], or [List].

Additional Help

Some forms can provide additional help at your request. Where additional help is available, it may take the form of a box or screen of information that you can display by pressing a function key or by making a menu selection. The documentation accompanying your form may tell you whether additional help is available and how it can be accessed.

CHAPTER

2

Running a Form

This chapter explains the general procedure for running a form, as well as the specific steps for running the sample form used throughout this Guide.

Running a Form

The following general steps are necessary to run a form:

1. Log onto your computer.
2. At the system command prompt:

   a. Type `RUNFORM` and press the Space Bar.

   b. Enter the name o fthe form you wish to run.

   c. Press the Return key.

   You are now at the SQL*Forms logon screen.

3. Enter your ORACLE username in the Name field and press the Return key.
4. Enter your ORACLE password in the Password field and press the Return key. Notice that your password does not appear on the screen as you type. This is a security feature provided by SQL*Forms.

   SQL*Forms now displays the form on your screen

If you want to bypass the logon screen, you can enter your username and password at the system command prompt. The format for the command follows:

```
RUNFORM [options] formname [username/password]
```

For example, to run a form named ORDERS you would enter

```
RUNFORM ORDERS SCOTT/TIGER
```

Note: This procedure works only if the form is in the current directory.

Accessing the Sample Form

To access the form used in the exercises in this guide, work through the exercise below:

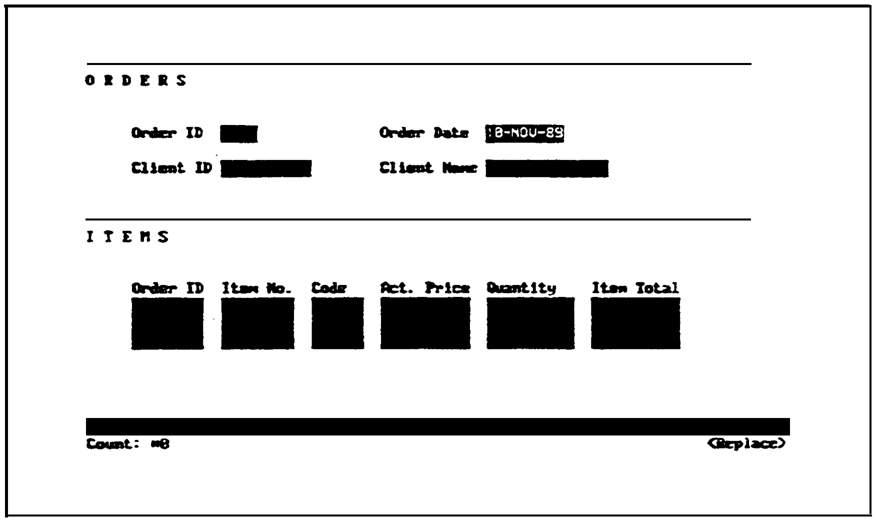
Exercise

1. Logon to your computer's operating system.
2. At the system command prompt, enter:

   ```
   RUNFORM SAMPLE username/password
   ```

   See your DBA if the sample form is not setup.
3. Press the Return key.
4. SQL*Forms displays the sample form on your screen.

Note: Today's date is automatically displayed in the Order Date field.



```
O R D E R S

    Order ID               Order Date  18-NOV-89
    Client ID              Client Name

I T E M S

    Order ID  Item No.  Code  Act. Price  Quantity  Item Total

Count: *0                                              <Replace>
```

FIGURE 2-1
The Sample Form

About the Sample Form

The sample form used throughout this Guide is based on a fictitious company called The Summit Sporting Goods Company. Assume that The Summit Sporting Goods company uses the sample form to keep track of orders it receives from sales representatives. The company's sales representatives take their orders in hand-written form. At the end of the day, they give the orders to data-entry operators who enter the information into the database using SQL*Forms. You will use the sample form throughout this guide to work through the exercises.

The sample form, called SAMPLE, consists of three blocks: ORDERS, ITEMS, and CLIENT DATA. The ORDERS block, which gives information about the current order, is based on the ORD table from the sample database. ITEMS, based on the ITEM table, gives specific information about items in the order, and CLIENT DATA, based on the CUSTOMER table, includes client information such as address and telephone number. The message line and the status line at the bottom of the screen are not part of the form.

The exercises included in this Guide are based on the ORD, ITEM, and CUSTOMER tables from the ORACLE sample database. If you are new to SQL*Forms, the best way to learn is to complete the exercises in this book. (The initial contents of the ORD, ITEM, and CUSTOMER tables are shown in Appendix B.)

Note: If you or someone else make changes to the data in the sample database tables, the data you see on your screen may differ from the data shown in the screen images of this manual.

Ending an Editing Session

When you are done with a form, you will want to save your changes and exit the form. Complete the following steps to end an editing session:

1. Press [Commit/Accept] to record your modifications in the database.
2. Press [Exit/Cancel]. SQL*Forms returns you to your system's command line.

CHAPTER

3 EXECUTING A QUERY

In this chapter, you will learn how to use a form to retrieve information from the ORACLE database. Retrieving information from the database is called executing a query. Read this chapter to learn when to query the database and how to retrieve all the records available to a particular form.

When to Query the Database

As an operator for Summit Sporting Goods, you not only enter new orders as they come in, you also check on the status of existing orders. To check on an order, however, you first need to retrieve it from the database This process of retrieving information is called querying the database.

You will want to query the database whenever you need to view or verify existing data. For example, if you want to check the date of an order, or if you want to know what a particular client has ordered, you can enter a query to retrieve the relevant records.

You can enter a query in one of two ways.

1. You can retrieve all the records entered in a block, regardless of the data they contain.
2. Or, you can retrieve only those records that fit a certain set of criteria. (For example, you might want to see all the orders taken on 1-MAY-86, or all the orders placed by client 106.)

Retrieving All Records

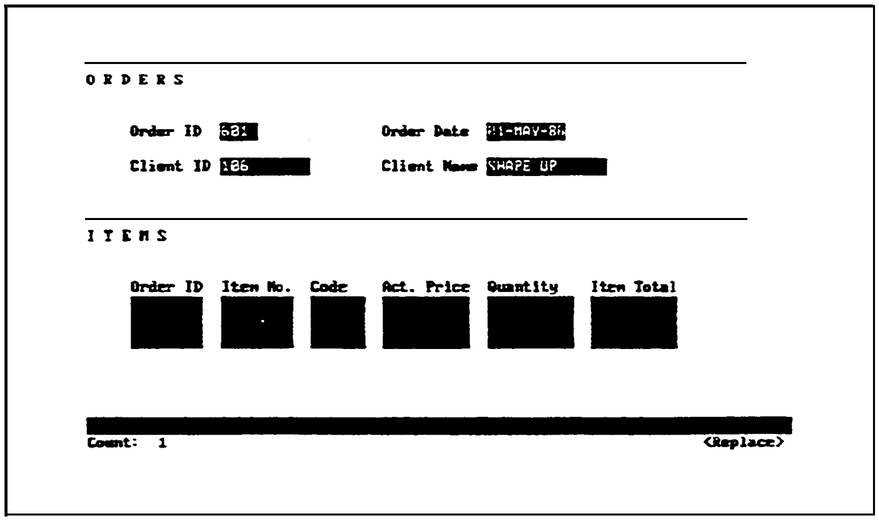
To retrieve all the records stored in a table, position the cursor in the block associated with the table and press [Execute Query].

The SAMPLE form currently has no records displayed. You can retrieve all the records stored in the ORDERS table by doing this exercise:

Exercise

1. Press [Execute Query]. You do not have to be positioned in the first field of the record when you execute a query. To perform a query on a field other then Order ID, you would press [Next Field] to move to any field of the current block and then press [Execute Query] to perform a query on that block.

   Now, your screen should look like Figure 3-1.



```
O R D E R S

     Order ID   681            Order Date   01-MAY-86
     Client ID  106            Client Name  SHAPE UP

I T E M S

     Order ID  Item No.  Code   Act. Price  Quantity   Item Total

Count:   1                                                      <Replace>
```

FIGURE 3-1
Retrieving All Records

Although pressing [Execute Query] retrieves all the records stored in a table, SQL*Forms displays only one record at a time. This is because the ORDERS block is a single-record block; only one record can be displayed at a time. The ITEMS block, on the other hand, is a multi-record block; it can display up to three records at once. You can scroll through all the records retrieved even though only a few are displayed at one time.

CHAPTER

4 MOVING AROUND THE FORM

As you work with a form—either scanning existing data or entering new records—you will need to move the cursor from one area of the form to another. This chapter explains how to make the following movements with the cursor:

- block to block
- record to record
- field to field

Moving from Block to Block

To move the cursor from one block to another, use [Next Block] or [Previous Block].

Exercise

1. With the cursor positioned in the ORDERS block, press [Next Block] once. Now ITEMS is the current block.

   Notice that the value `601` appears in the Order ID field. For convenience, and to ensure that the order is entered accurately, the value for Order ID is read from the ORDERS block and is automatically displayed in the ITEMS block.
2. Press [Previous Block] once to move back to the ORDERS block.
3. Press [Next Block] twice. This positions your cursor in the CLIENT DATA block.
4. Press [Next Block] once more. ORDERS is again the current block.

When you move the cursor from block to block, your movement is cyclical. For example, when you get to the last block of the form and press [Next Block], SQL*Forms takes you back to the first block. Similarly, if you press [Pevious Block] while positioned on the first block of the form, the cursor moves to the form's last block.

Moving from Record to Record

Once you have retrieved records from the database, you can use [Next Record] and [Previous Record] to view them. If the block is a single-record block, only one record is visible at a time. If the block is a multi-record block, more than one record is visible. The record movement keys simply move the cursor from one record to another.

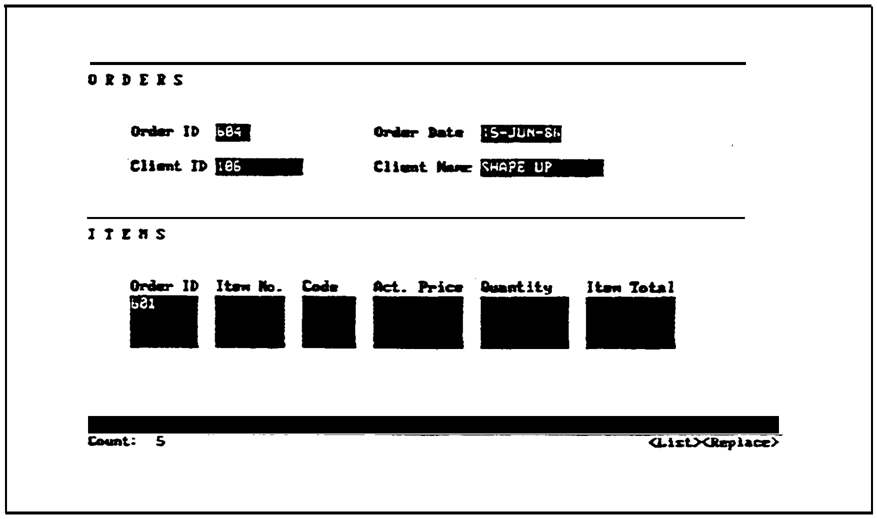
To view the remaining records in the ORDERS block, press the [Next Record] key once for each succeeding record.

Exercise

1. Press [Next Record] four times. SQL*Forms displays Order ID 605 on the screen.
2. Press [Previous Record] once. SQL*Forms displays the information for Order ID 604 in the ORDERS block. (See Figure 4-1.)

   Notice that the value `601` appears in the Order ID field of the ITEMS block. The form's designer specified that when you press [Next Block} the order ID be copied automatically from the Orders ID field in the ORDERS block into the Orders ID field of the ITEMS block.

FIGURE 4-1
Moving from Record to Record



```
O R D E R S

    Order ID  604              Order Date  15-JUN-86
    Client ID 106              Client Name SHAPE UP

I T E M S

    Order ID  Item No.  Code  Act. Price  Quantity  Item Total
    601

Count:  5                                          <List><Replace>
```

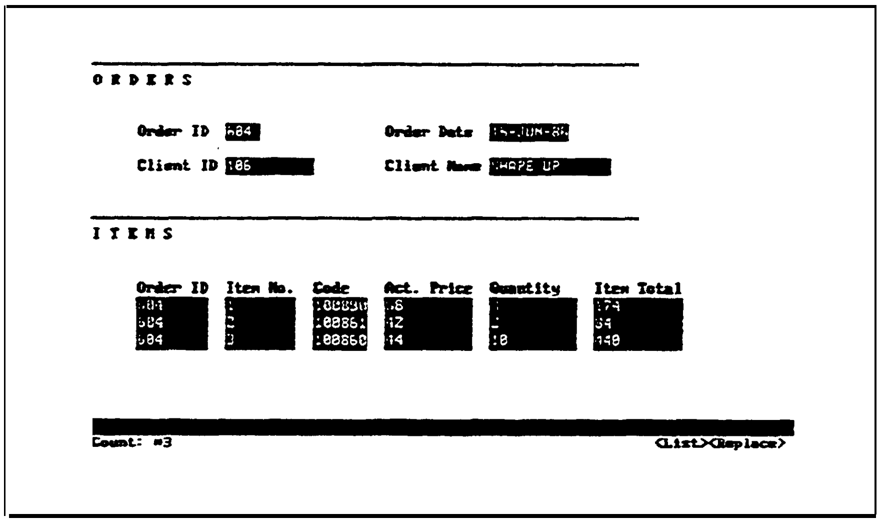
To see how [Next Record], [Previous Record], [Up], [Down] work in a multi-record block work through the next exercise.

Exercise

1. Press [Next Block] to move from the ORDERS block to the ITEMS block.
2. Press [Execute Query].

   This tells SQL*Forms to retrieve all the records in the ITEMS table that have a value of 604 for the Order ID field. SQL*Forms displays three records, as shown in Figure 4-2.

FIGURE 4-2
Multi-Record Block



```
O R D E R S

    Order ID 604              Order Date [illegible]-JUN-88
    Client ID 106             Client Name SHAPE UP

I T E M S

    Order ID  Item No.  Code    Act. Price  Quantity  Item Total
    604       1         100890  58          3         174
    604       2         100861  42          2         84
    604       3         100860  44          10        440

Count: *3                                              <List><Replace>
```

3. Press [Next Record] twice to move to the last record in the ITEMS block.
4. Press [Next Record] again.

   Note: Unlike the block movement keys, the record movement keys are not cyclical. When you reach the last record, you must press [Previous Record] repeatedly to get back to the first record.
5. Now press [Previous Record] four times. You should be positioned on the first record in the ITEMS block. SQL*Forms displays the message:

   ```
   FRM-40100: At first record.
   ```
6. Press [Next Field].
7. Press [Down]. The cursor moves down one record, but stays in the same field.
8. Press [Up]. The cursor moves up one record, but stays in the same field.

   Note: When you are updating the same field in several records, you may want to use [Up] and [Down] instead of [Next Record] and [Previous Record].

Moving from Field to Field

You can move from one field to another by pressing [Next Field] and [Previous Field]

Exercise

1. Press [Pevious Block] to move to the ORDERS block.
2. Press [Next Field] twice. The cursor should be positioned on the Client ID field.
3. Press [Next Field] again.

   Instead of moving to the Client Name field, as you would expect, the cursor moves back to Order ID. This is because the Client Name field has been designated by the form's designer as non-enterable.

   When you enter a value for Client ID in the ORDERS block, SQL*Forms retrieves the name of the client and displays it automatically m the Client Name field. T'his automatic retrievaI helps you to maintain accuracy.
4. Press [Previous Field]. The cursor moves from the Order ID field back to the Client ID field (again skipping the Client Name field).
5. Press [Previous Field] again. You are back at Order Date.

The order in which the cursor moves from one field to another is determined by the form's designer. In the SAMPLE form, you move from the Order ID field to the Order Date field, and then to the CIient lD field. Another form using the same fields could have these fields in a different sequence. Usually, however, you will find that fields are arranged in a sequence that goes from left to right and top to bottom.

If a field's value is invalid, SQL*Forms will not allow the cursor to leave that field until you have corrected the value. For example, if a field requires a numerical value between 100 and 9999, the cursor cannot leave the field when the value is, for example, 99 or 10000. This does not apply to block mode terminals.

If the designer has given a field the autoskip attribute, the cursor leaves the field after you have entered a character in the last field position. The autoskip feature is used to minimize keystrokes for fields that require a standard number of characters, such as a field for a telephone number.

Moving within a Field

When working in a field, use [Left] to move the cursor one position to the left and use [Right] to move the cursor one position to the right.

A field can be shorter than the corresponding table column. Thus, a field can be shorter than the value it contains. When this occurs, you will not see the entire content of the field. However, SQL*Forms allows you to move a value back and forth within a field so you can see the entire value. This is called horizontal scrolling. The next exercise illustrates how to scroll horizontally using [Left] and [Right] as well as [Scroll Left] and [Scroll Right].

Exercise

1. Make sure the cursor is positioned in the ORDERS block and press [Enter Query].
2. Enter 613 in the Order ID field and press [Execute Query].

   Order 613 appears on the screen
3. Press [Next Block] twice.

   Now you are in the CLIENT DATA block with the value of 613 in the Order ID field.
4. Press [Execute Query].

   Notice that the information in the Name field is longer than the display area. The status line displays "<" when the beginning of the current field's value is scrolled off the screen and it displays ">" When the end of the current field's value is scrolled off the screen.
5. Press [Next Field] to position the cursor in the Name field.
6. Press [Right] repeatedly until you reach the end of the client's name.

   The name is scrolled into the field's display area one character at a time.
7. Press [Left].

   The cursor moves one character to the left.
8. Press [Scroll Left] to view the beginning of the name.

   The field's display window shifts to the left, displaying field contents that were outside the window.
9. Press [Scroll Right] to view the end of the name.

   The field's display window shifts to the right, displaying field contents that were outside the window.

CHAPTER

5 RETRIEVING SELECTED RECORDS

This chapter describes several ways to retrieve records that meet specific criteria. After completing the exercises in this chapter, you will not only know how to execute simple queries that meet specific criteria, but also complex queries that satisfy several conditions. The following topics are discussed in this chapter.

- matching exact values
- entering variable conditions
- using the SQL WHERE clause for advanced queries

Matching Exact Values

Suppose a customer wants you to check on the price of an order. You know that the order was placed on January 7,1987, and that the Order ID is 610. SQL*Forms can retrieve the record that contains only these values.

The following are the general steps for retrieving records that match exact values:

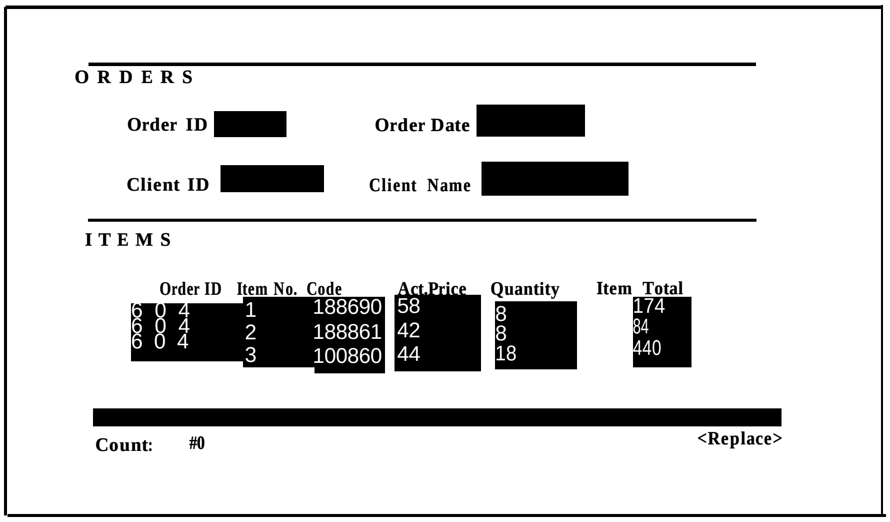
1. Press [Next Block] or [Previous Block] until your cursor is positioned in the correct block.
2. Press [Enter Query].
3. Type the values you want to match into the appropriate fields.
4. Press [Execute Query].
5. Press [Next Record] or [Previous Record] to view the retrieved records.

Exercise

1. Position the cursor in the ORDERS block.
2. Press [Enter Query].

   This clears the displayed record. Your screen should now look like the figure below.

FIGURE 5-1
Matching Exact Values



ORDERS

Order ID

Order Date

Client ID

Client Name

ITEMS

Order ID	Item No.	Code	Act.Price	Quantity	Item Total
6 0 4	1	188690	58	8	174
6 0 4	2	188861	42	8	84
6 0 4	3	100860	44	18	440

Count: #0 <Replace>

3. Type `610` in the Order ID fieId.
4. Press [Execute Query].

   SQL*Forms displays the message

   ```
   Working...
   ```

   at the bottom of the screen, then displays the information for order number 610 in the ORDERS block. (See Figure 5-2, below.)

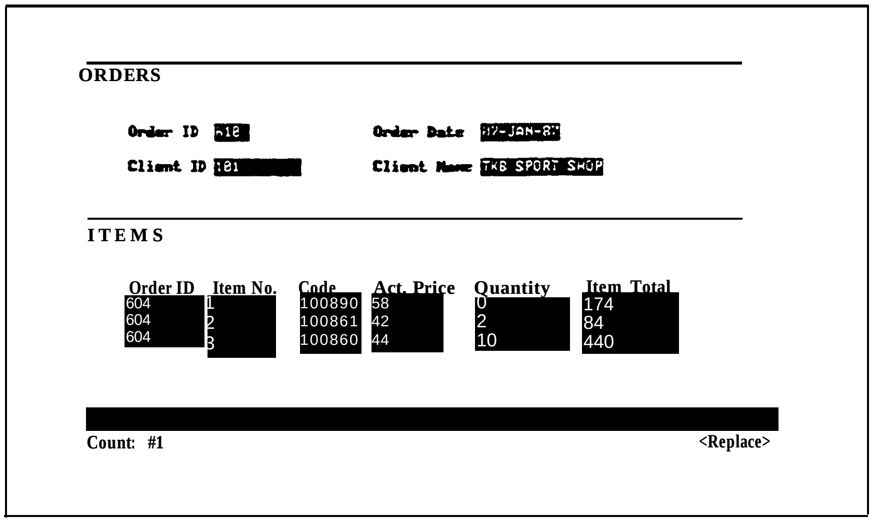


FIGURE 5-2
Order Number 610

ORDERS

Order ID	610	Order Date	07-JAN-87
Client ID	101	Client Name	TKB SPORT SHOP

ITEMS

Order ID	Item No.	Code	Act. Price	Quantity	Item Total
604	1	100890	58	0	174
604	2	100861	42	2	84
604	3	100860	44	10	440

Count: #1 <Replace>

You now have the Order ID, Client ID, Order Date, and Client Name for order 610. To find out the item total for each line item, you will need to retrieve the record for order 610 in the ITEMS block as well.

Exercise

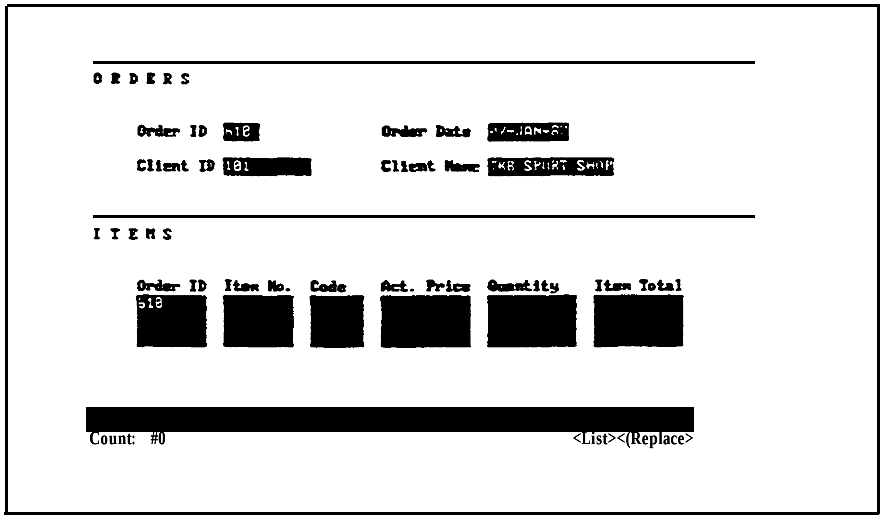
1. Press [Next Block].

   Notice that the data from a previous query is still in the ITEMS block.

2. Press [Clear Block].

   The previous records are cleared, and the current order ID is copied from the ORDERS block. (See Figure 5-3.)

FIGURE 5-3
Cleared Data

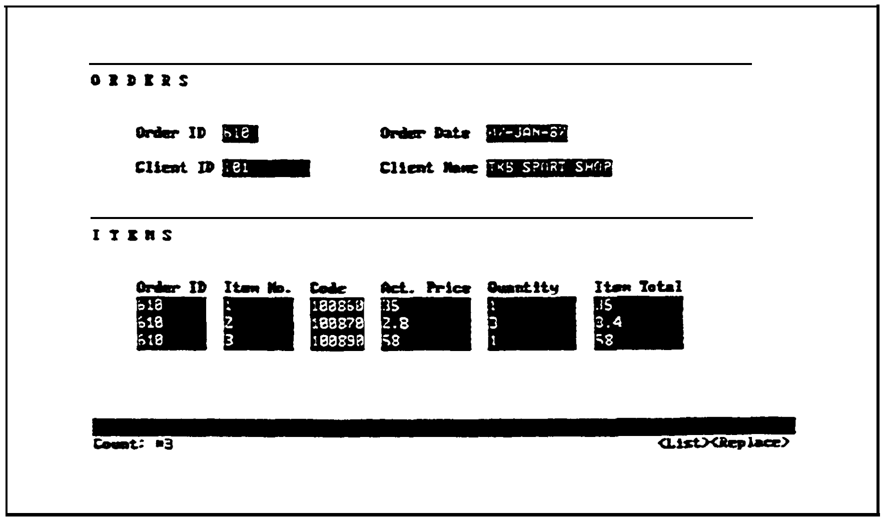


```
ORDERS

    Order ID 610              Order Date 07-JAN-87
    Client ID 101             Client Name TKB SPORT SHOP

ITEMS

    Order ID  Item No.  Code   Act. Price  Quantity   Item Total
    610

Count:  #0                                        <List><(Replace>
```

3. Press [Execute Query].

   SQL*Forms finds the records that have an Order ID of 610 and displays them in the ITEMS block.

FIGURE 5-4
Total Price



```
ORDERS

    Order ID 610              Order Date 07-JAN-87
    Client ID 101             Client Name TKB SPORT SHOP

ITEMS

    Order ID  Item No.  Code    Act. Price  Quantity   Item Total
    610       1         100860  35          1          35
    610       2         100870  2.8         3          3.4
    610       3         100890  58          1          58

Count: #3                                         <List><Replace>
```

Using Pattern Matching

You can retrieve records using values in two or more fields by moving the cursor to those fields and entering the values. SQL*Forms will only fetch those records that meet the search criteria specified in all fields. For example, if you enter `SALESMAN` in a field called "Job" and `0` in a field called "Commission," the query selects records in which the "Job" is SALESMAN and the "Commission" is 0. You can also select records where a value fits a certain pattern. To do this, enter a value into a field where"_" represents any character and "%" represents any combination of characters (including no characters).

The following list contains examples of pattern matching:

Pattern	*Possible Matches*
JON_S	JONES, JONAS, JONOS, JONQS, JON-S
S_AR_	SMART, SNARE, SHARE, SHARD, SHARP, SHARK
ENTER%	ENTER, ENTERS, ENTERED, ENTERTAIN
_IN%S	BINS, FINES, WINNERS, WINEMAKERS

Entering Variable Conditions

Sometimes it is not practical to enter the exact values that you want retrieved records to match. For example, you might want to retrieve the following:

- all the records with an item total of more than $100
- all the records with an Order ID of 10 or more
- all the orders place in the month of June

Rather than entering an exact data value, you can enter a relational operator before the data value in one or more fields. Table 5-1 shows some relational operators and how you can use them. (For a complete discussion of relational operators, see the *SQL Language Reference Manual.)*

Table 5-1
Meanings and Example of Relational Operators

Operator	***Meaning***	***Examples***
=	equal to	`='SMITH'`
!=	not equal to	`!=19.5`
>	greater than	`>100.00`
>=	greater than or equal to	`>=2000`
<	less than	`<'DAVIS'`
<=	Iess than or equal to	`<=100.00`
BETWEEN	between two values	`#BETWEEN 100 AND 110`

Note: The operators in Table 5-1 do not work in time fields.

For example, to select records that have an item total of more than $100, you would press [Enter Query] and enter `>100` in the Item Total field. (The character “>” is a relational operator meaning greater than, and 100 is the value to be tested.) If SQL*Forms is testing a value that is a character string (such as 'SMITH'), the character “>” would retrieve records that come after the character string in the alphabet (e.g., THOMAS and WILLIAMS).

Note: Fields containing character or date values must be enclosed by single quotes Also, whenever you use a relational operator that is a word (such as BETWEEN) in a field, you must precede the operator with a "#".

Note: Do not use placeholders in non-database fields.

Using the SQL WHERE Clause for Advanced Queries

Although you can execute fairly complex queries by entering data values or relational operators in fields, you may want to execute more advanced queries. You can execute advanced queries through a SQL WHERE clause. The WHERE clause allows you to express queries based on conditions other than an exact match.

To use the WHERE clause, you must place a variable in one or more fields. The variable, which serves as a placeholder, is preceded by a colon (:). By placing a variable in a field, you signal that you want to enter a WHERE clause using the value in that field. The first character of the variable name itself is alphabetic, and any following characters are alphabetic, numeric, or the special characters "_", "$" or "#".

Specifying a Range of Values

The exercise below shows you how to use a WHERE clause to retrieve all the orders that were placed between the 5th and the 25th of January.

Exercise

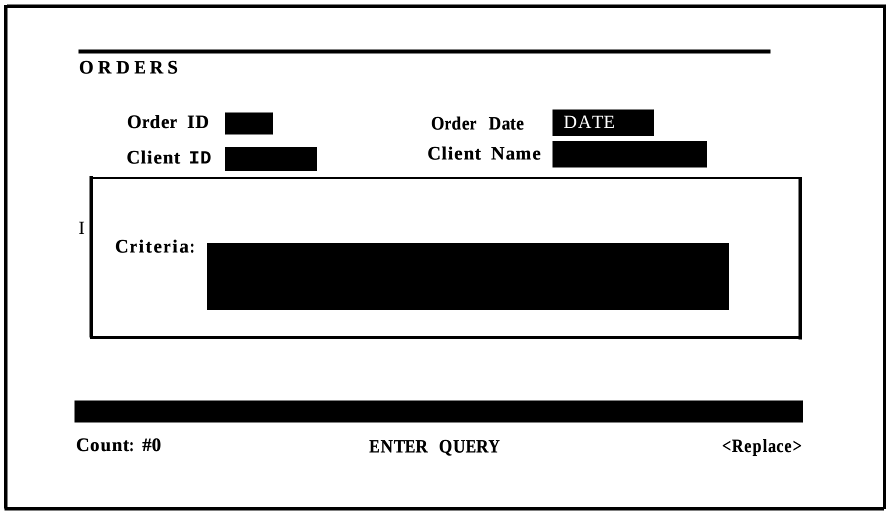
1. Press [Previous Block] once to move to the ORDERS block.
2. Press [Enter Query].

   As you learned previously, this clears the block of the previously displayed record.
3. Press [Next Field] to move to the Order Date field.
4. Type :DATE in the Order Date field.

   Instead of typing an exact value (like 05-Jan-87), you have entered a variable. The colon (:) indicates a variable name rather than an exact value. The variable name identifies the field, and can be referenced in a WHERE clause.
5. Press [Execute Query].

   SQL*Forms displays the Query Where alert, with the cursor positioned in an enterable field. You can now enter the conditions you want retrieved records to meet. At this point, your screen should look like Figure 5-5.

FIGURE 5-5
Query Where Dialog Box

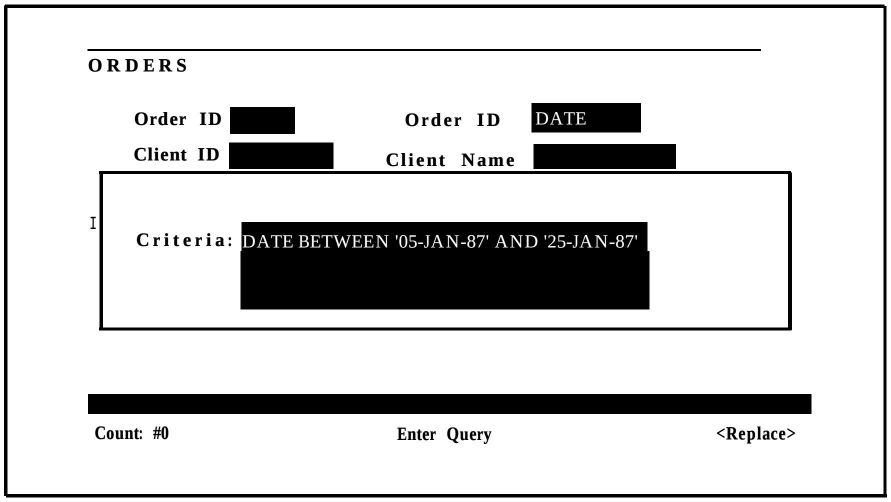


```
ORDERS

   Order ID              Order Date  DATE
   Client ID             Client Name
I
  Criteria:



Count: #0              ENTER QUERY              <Replace>
```

6. Type: `DATE BETWEEN  '05-JAN-87' AND '25-JAN-87'`

   This tells SQL*Forms that you want to see all the records that have an Order Date between the 5th and the 25th of January. (See Figure 5-6.) If you make a mistake when entering your query, you can use [Delete Character] to delete the character under the cursor. Or, you can use [Right] or [Left] to move the cursor in either direction horizontally without changing any characters. Then type the desired character over the incorrect one.

FIGURE 5-6
Entering the Condition



```
ORDERS

   Order ID              Order ID  DATE
   Client ID             Client Name
I
  Criteria: DATE BETWEEN '05-JAN-87' AND '25-JAN-87'



Count: #0              Enter Query              <Replace>
```

7. Press [Commit/Accept].

   SQL*Forms retrieves the three orders placed between the 5th and the 25th of January. To view the records one by one, press [Next Record] or [Previous Record].

Summary for Entering Variable Conditions

In summary, to enter a query that uses a WHERE clause, follow these steps:

1. Press [Next Block] until your cursor is positioned in the correct block.
2. Press [Enter Query].
3. Instead of placing a value into a field, enter a variable name (such as `:ID` or `:DATE`).
4. Press [Execute Query].
5. In the Query Where dialog box, enter the condition you want the retrieved records to meet.
6. Press [Commit/Accept].
7. Press [Next Record] or [Previous Record] to view the retrieved records.

Using the WHERE Clause in Additional Ways

In the previous example, you entered a WHERE clause that specfied a range for retrieved records to fall within. (The value for Order Date had to be between 05-JAN-87 and 25-JAN-87.) Table 5-2 presents additional ways to use WHERE clauses in queries.

TABLE 5-2
Using WHERE Clauses in Queries

Purpose	*Examples*
To retrieve records that have a value:	
greater than (>)	`:CLIENTID > 106`
greater than or equal to (>=)	`:ITEMTOT >= 100`
less than (<)	`:ORDERID < 305`
less than or equal to (<=)	`:ITEMTOT <= 100`
equal to (=)	`:CLIENTID = 106`
not equal to (!=)	`:NAME != 'SHAPE UP'`
To express a query that can be satisfied by either of two conditions	`(:CODE = 100860 OR :CODE = 100861)`
To express a query with two conditions	`(:ORDERDATE = '14-JUL-86') AND (:CLIENTID = 106)`

Using Complex Search Criteria

You can enter queries with search criteria more complex than those described above. For example, you can select records in which Client ID has the following values:

- one of several values
- 107 *or* Order ID is greater than 615 (Entering values into the Client ID and Order ID fields would select records in Which Client ID is 107 *and* Order ID is greater than 615.)
- 107 *and* Order ID is greater than 615 *or* Order Date is January 7

Following is the general procedure for using complex search criteria:

1. Press [Enter Query].
2. For each field involved, enter a variable (a letter or short word works well). For example, enter `:ID` in the Order ID field and `:N` in the Client Name field.
3. Press [Execute Query] to display the Query Where dialog box.
4. Enter an expression that describes the search criteria, using placeholders similar to the ones in Step 2.

   For example:

Search Criteria	*Values Entered*
Order ID greater than 615 or Client Name is JUST TENNIS	`:ID > 615 OR :N = 'JUST TENNIS'`
Order ID less than or equal to 615 or Client Name ends with 'Tennis'	`:ID <= 615 OR :N like '%TENNIS'`

5. Press [Commit/Accept]. SQL*Forms analyzes the logical expression entered and executes the query.

Reusing Search Criteria

To reuse the search criteria of a previous query in the current block, press [Enter Query] twice press it once to initiate anew query and a second time to redisplay the previous search criteria. You can use the criteria displayed or modify it before pressing [Execute Query].

If you leave placeholders in any field, and then press [Enter Query] twice, SQL*Forms redisplays those placeholder in the field in which you entered them. Then press [Execute Query] to display the last response entered in the Query Where dialog box. You may use the response as is by pressing [Next Field], or you may edit, replace, or remove it.

Counting Query Records

If you want to know how many records a search will return before actually executing a query, you need to use [Count Query Hits]. Following is the general procedure for counting the number of records that meet a set of search criteria:

1. Press [Enter Query].
2. Enter the search criteria.

3. Press [Count Query Hits].

   SQL*Forms counts the records that meet the search criteria and displays the number on the message line.

   Note that you pressed [Count Query Hits] instead of [Execute Query]. You can use both keys, one after the other, to count and fetch records.

After you press [Count Query Hits], you have the following options

- Press [Execute Query] to perform the query.
- Enter other search criteria.
- Press [Exit/Cancel] to exit Enter Query mode without executing a query.

Using the WHERE Clause with Multiple Conditions

For situations where you want to enter more sophisticated queries, you can use the SQL language. (Refer to the *SQL Language Reference Manual.)* SQL*Forms supports most SELECT clauses, with the exception of the GROUP BY and CONNECT BY clauses.

Using SQL, you can enter a WHERE clause that has several conditions. For example, you could:

1. Press [Enter Query] then place the variable `:ORDER` in the Order ID field and the variable `:CLIENT` in the Client ID field.
2. Press [Execute Query], and enter the following condition in the Query Where alert:

   ```
   :ORDER > 500 AND :CLIENT < 110 ORDER BY :CLIENT
   ```

   This SQL statement will retrieve all the records that have both an Order ID greater than 500 and a Client ID less than 110. The records will be ordered by Client ID.

CHAPTER

6 MODIFYING DATA IN THE DATABASE

This chapter explains how to use SQL*Forms to modify data in the database. This is a very important chapter to understand because you will be using SQL*Forms frequently to add new records to the database or to modify existing records. The following topics are covered in this chapter:

- replacing, inserting, and deleting characters
- updating a record
- deleting information from the database
- creating records

Inserting, Replacing and Deleting Characters

SQL*Forms allows you to enter characters using one of two modes:

1. Insert mode

   When Insert mode is active, each character you enter appears at the cursor. The character currently at the cursor and all following characters are moved to the right.

2. Replace mode

   When Replace mode is active, each character you enter replaces the character at the cursor. The characters after the cursor do not move.

You can switch between Insert mode and Replace mode by pressing [Insert/Replace]. Once set, the mode remains active until you change it by pressing this key again. (The status line displays the current mode.)

To delete a character, place the cursor on the character and press [Delete Character]. [Delete Character] function in the same manner whether insert or replace mode is active. (When the cursor is after the last character in a field, [Delete Character] has no effect.)

If you want to delete the character before the cursor, press [Delete Backward].

Inserting and Deleting a Record

Sometimes you want to retrieve records simply to view the information they contain; other times, you want to change information.

Suppose that the client Shape Up wants to change order number 615. They want to add an order three SP Junior Rackets (Code 101863).

Exercise

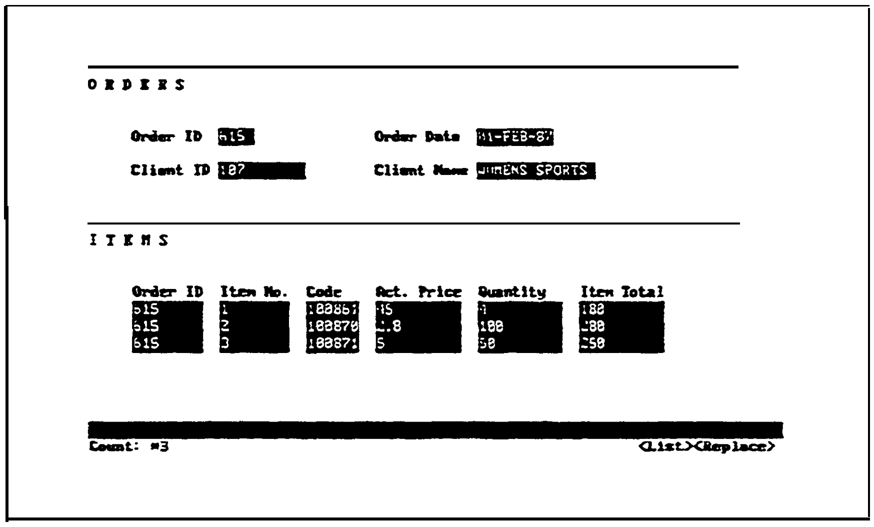
1. Make sure your cursor is positioned in the ORDERS block.
2. Press [Enter Query].
3. Type `615` in the Order ID field.
4. Press [Execute Query].

   SQL*Forms retrieves the record for order 615.

5. Press [Next Block] to move to the ITEMS block.
6. Press [Clear Block] to clear the data remaining from the last query.

   The new value displayed in the Order ID field is the current order number, 615. This is an automatic default value placed thereby the form's designer.
7. Press [Execute QueryJ.

   SQL*Forms retrieves the three items that make up order 615. Your screen should now look like Figure 6-1.



```
ORDERS

    Order ID 615            Order Date 01-FEB-87
    Client ID 107           Client Name WOMENS SPORTS

ITEMS

    Order ID  Item No.  Code    Act. Price  Quantity  Item Total
    615       1         100861  45          4         180
    615       2         100870  2.8         100       280
    615       3         100871  5           50        250

Count: *3                                          <List><Replace>
```

FIGURE 6-1
Order Number 615

Now that you have the correct order displayed on your screen, you can make the changes requested by the client. Remember that Shape Up wants to add another item to its order.

Exercise

1. Press [Next Record] to move to the record that has a value of 2 in the Item No. field.
2. Press [Clear Record].

   This clears item number 2 from the screen (but does not delete it from the database) and makes room for you to add the new item 4 to the order.
3. Press [Next Record] to move to the next line.

   SQL*Forms displays 615 as the default value for Order ID.

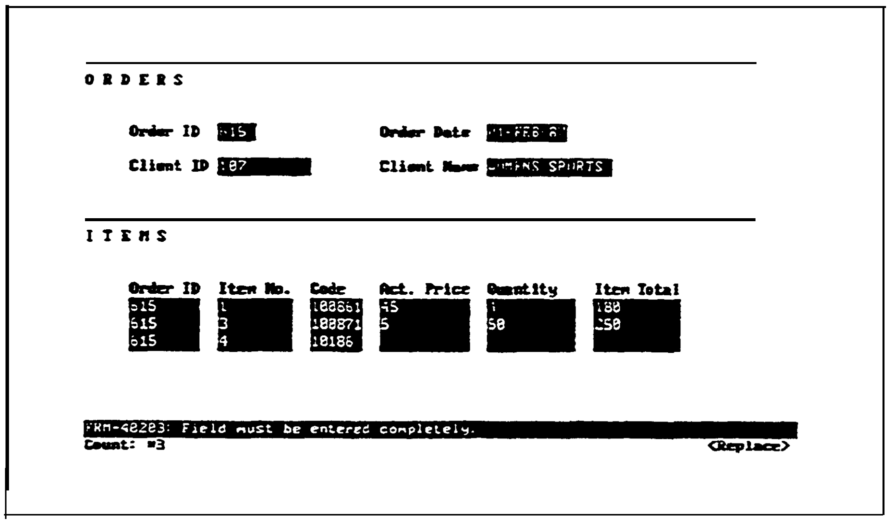
4. Press [Next Field] to move to the Item No. field.
5. Type `4` in the Item No. field and Press [Next Field].
6. Type `10186` in the Code field and press [Next Field].

   SQL*Forms displays the message:

   ```
   FRM-40203: Field must be entered completely.
   ```

   The Code field is an example of a fixed-length field. Because all product identification numbers contain six digits, the designer of the form has specified that the cursor cannot leave the code field unless a six-digit number has been entered. (See Figure 6-2, below.)

FIGURE 6-2
Help Message

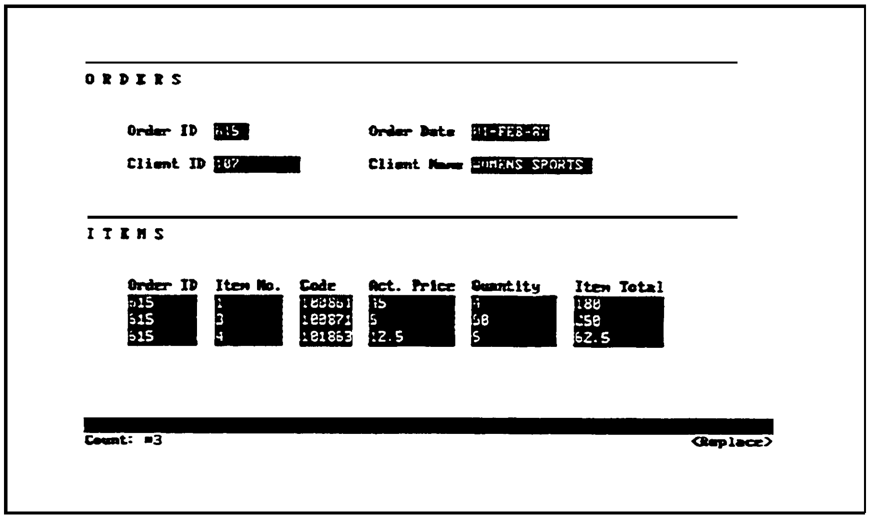


```
O R D E R S

    Order ID   615              Order Date  [illegible]
    Client ID  107              Client Name [illegible]NS SPORTS

I T E M S

    Order ID  Item No.  Code     Act. Price  Quantity   Item Total
    615       1         100861   45          1          100
    615       3         100871   5           50         250
    615       4         10186

FRM-40203: Field must be entered completely.
Count: *3                                                 <Replace>
```

7. Retype the code as `101863` and press [Next Field].
8. Type `12.50` as the Act. Price and press [Next Field].
9. Type `5` as the Quantity and press [Next Field].

   As soon as your cursor moves into the Item Total field, SQL*Forms calculates and displays the total price for the item. The designer of the form has SQL*Forms multiply the value in the Act. Price field by the value in the Quantity field. The resulting number is displayed in Item Total when your cursor enters that field. (See Figure 6-3.)

Figure 6-3
Item Total



ORDERS

Order ID	615	Order Date	[illegible]-FEB-[illegible]
Client ID	107	Client Name	WOMENS SPORTS

ITEMS

Order ID	Item No.	Code	Act. Price	Quantity	Item Total
615	1	100861	45	4	180
615	3	100871	5	50	250
615	4	101863	12.5	5	62.5

Count: *3 <Replace>

10. Press [Commit/Accept] to save the record in the database.

Not all fields of a form can be updated. The designer may have designated some fields as non-updatable. The following exercise shows what happens when you try to update a non-updatable field.

Exercise

1. Press [Previous Block] to move to the ORDERS block.
2. Try typing any number in the Order ID field.

   SQL*Forms displays the message:

   FRM-40200: Field is protected against update.

   The designer made this field non-updatable to prevent you from accidentally changing an order identification number. The Order ID is a unique value, and is the only way of linking the information in the ORD table to the information in the ITEM table. The error message goes away when you press another key.

Deleting Information from the Database

Another way to modify the contents of the database is to delete entire records. For example, you may want to do one of the following:

- delete an item from an order
- cancel an entire order
- delete a client from the database

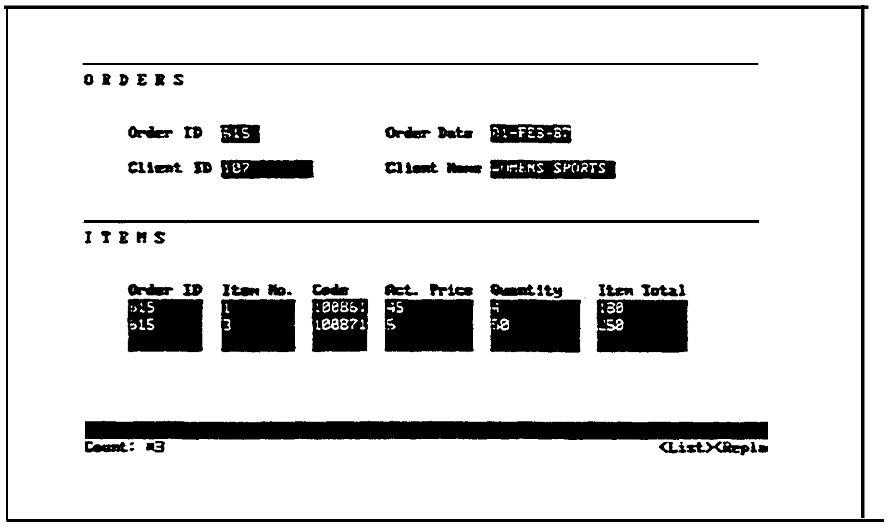
Suppose that Shape Up has decided to cancel their order for the SP Junior Rackets (Code 101863). The following exercise shows you how to delete the newly added record. (This also restores the sample database to its original form.)

Exercise

1. Make sure the cursor is in the ITEMS block.
2. Make sure the cursor is positioned on the record with an Item number of 4.
3. Press [Delete Record] to delete record number 4.

   Your screen should look like Figure 6-4, below.

FIGURE 6-4
Deleting a Record



```
ORDERS

   Order ID    515              Order Date   01-FEB-87
   Client ID   107              Client Name  [illegible]MENS SPORTS

ITEMS

   Order ID   Item No.   Code     Act. Price   Quantity   Item Total
   515        1          100861   45           4          180
   515        3          100871   5            50         250

Count: *3                                                  <List><Repla
```

4. Press [Commit/Accept] to have the record deleted.

   SQL*Forms displays the message:

   ```
   Transaction completed -- 1 records processed.
   ```

Although the record with Code 101863 has been deleted, the original record (with Code 100861—the Ace Tennis Racket IIs) is still in the database. Remember that you did not delete it—you simply cleared it from the form with [Clear Record]. To verify that this original record is still in the database and that the record you just deleted is gone, do the following exercise.

Exercise

1. Make sure your cursor is positioned in the Order ID field of the ITEMS block.
2. Press [Execute Query].

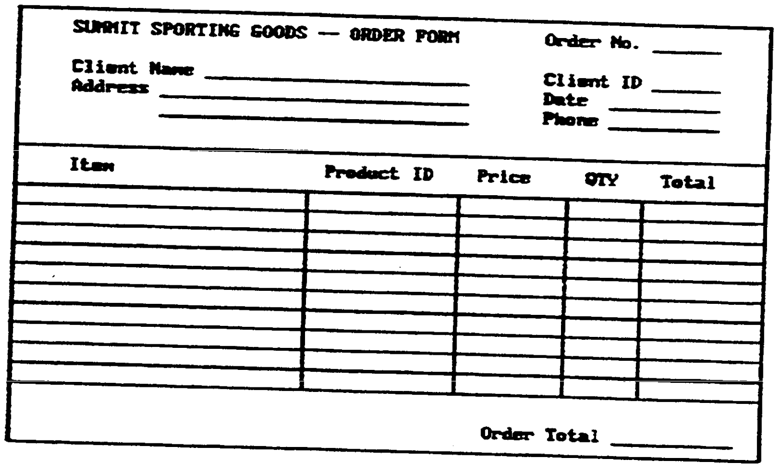
   This tells SQL*Forms to retrieve all the records in ITEMS where the Order ID is 615.

   SQL*Forms retrieves three records from the database. The original record that you cleared, but did not delete, is still in the database.
3. Press [Clear Form/Rollback] to clear the form for the next exercise.

In summary, remember that [Clear Record] removes the record from your workspace, but leaves it in the database. [Delete Record] deletes the record from the database once you press [Commit/Accept].

Creating New Records

Whenever Summit Sporting Goods receives another order, you need to add a new record to the database. Suppose you have just received a three-item order from the Just Tennis shop in the Northside Mall. (See Figure 6-5.) You will need to add this order to the database. The next exercise shows you how to enter the new records into the database.



SUMMIT SPORTING GOODS — ORDER FORM

Order No. ______

Client Name ______________________

Address ______________________

Client ID ______

Date ______

Phone ______

Item	Product ID	Price	QTY	Total

Order Total ________

FIGURE 6-5
Order from Just Tennis

Completing the ORDERS Block

The first part of the order contains general information about the client and order. Follow these steps to filll in the ORDERS block:

Exercise

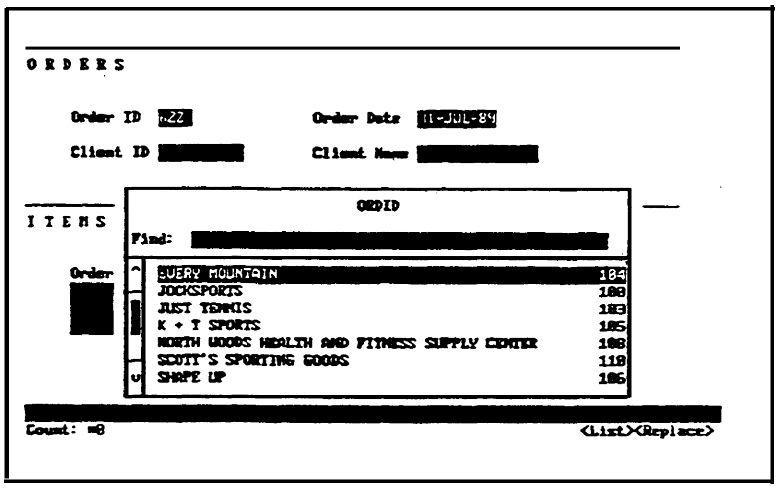
1. Position the cusor in the ORDERS block and press [Clear Block].
2. Type 622 in the Order ID field and press [Next Field].

   Notice that the Order Date field has been filled with today's date. This is a convenience provided by the form designer. Because you are entering the order on the same day that it was received, you do not need to change the date.
3. Press [Next Field] to accept the value in Order Date and to move the cursor to the Client ID field.

When you move into the Client ID field, the List lamp appears on the status line to the immediate left of the Insert/Replace lamp. On any field of the form, the designer can provide a list of field values that you may then search through to find an appropriate value.

4. Press [List]. The list of values pop-up window appears (see Figure 6-6). To scroll up and down through the list of client IDs, use the [Up] and [Down] keys.

FIGURE 6-6
List of Values Pop-Up Window



```
ORDERS

   Order ID 622             Order Date 11-JUL-89
   Client ID                Client Name

ITEMS               ORDID
        Find:
   Order  EVERY MOUNTAIN                                   104
          JOCKSPORTS                                       100
          JUST TENNIS                                      103
          K + T SPORTS                                     105
          NORTH WOODS HEALTH AND FITNESS SUPPLY CENTER     108
          SCOTT'S SPORTING GOODS                           110
          SHAPE UP                                         106

Count: *8                                       <List><Replace>
```

5. Type JUST in the entry field of the list of values and press [List] or [Next Field].

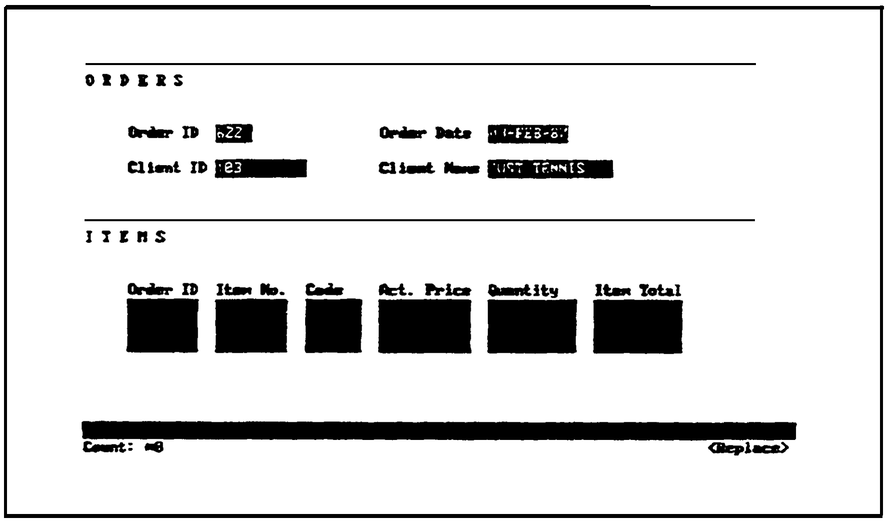
   Notice that "JUST TENNIS" is the only name that appears.

6. Press [Select].

   "Just Tennis" appears in the Client Name field and the number “103” appears in the Client ID field. See Figure 6-7.

Note: You can search on all the columns selected in a list of values.

FIGURE 6-7
Completed ORDERS Block



```
O R D E R S

    Order ID  622            Order Date  [illegible]-FEB-8[illegible]
    Client ID 123            Client Name JUST TENNIS

I T E M S

    Order ID  Item No.  Code   Act. Price  Quantity   Item Total

Count: *8                                                  <Replace>
```

Other Methods for Adding Records

You can also create record by moving the cursor to the empty "record" following the last record displayed and entering field values there. To add several records in this manner, press [Insert Record] or [Next Record] after you have completed data entry for each record.

If you begin adding a record and want to start over, press [Clear Record].

Use [Duplicate Record] and [Duplicate Field] to add records that are similar. [Duplicate Record] copies every field of the preceeding record into a new record; [Duplicate Field] copies only the current field. After pressing either key, you can modify the field values that have been copied.

Completing the ITEMS Block

The ITEMS block is where you enter specific information about the items to be ordered. The following exercise tells you how to complete it.

Exercise

1. Press [Next Block] to move to the ITEMS block.

   The new order number, 622, appears as the default in the Order ID field.

2. Accept the order number by pressing [Next Field].

3. Enter the items to be ordered as shown in the tables below. After each entry, press [Next Field]. To begin entering data in a new record, press [Next Record].

   Record 1

Field	*Value*
Item No.	`1`
Code	`100860`
Act. Price	`35`
Quantity	`7`

   Record 2

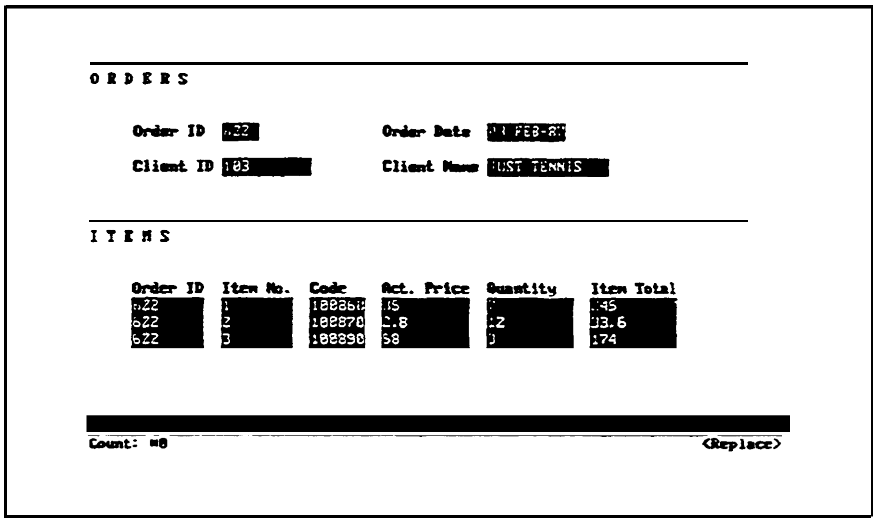
Field	*Value*
Item No.	`2`
Code	`100870`
Act. Price	`2.80`
Quantity	`12`

   Record 3

Field	*Value*
Item No.	`3`
Code	`100890`
Act. Price	`58`
Quantity	`3`

Your screen should now look like Figure 6-8. You may have noticed that when you were in the Quantity field and pressed [Next Field], the value for Item Total was calculated for you. Comparing this number against the figure on the Salesman's order check the accuracy of the order.

FIGURE 6-8
Completed ITEMS Block



```
O R D E R S

    Order ID  622              Order Date  [illegible]-FEB-8[illegible]
    Client ID 103              Client Name UST TENNIS

I T E M S

    Order ID  Item No.  Code     Act. Price  Quantity  Item Total
    622       1         100860   35          [illegible]  [illegible]45
    622       2         100870   2.8         12        33.6
    622       3         100890   58          3         174

Count: *8                                                   <Replace>
```

Completing the CLIENT DATA Block

The CLIENT DATA block contains information regarding the client placing the order, such as name, address, and so on. Perform the next exercise to complete the CLIENT DATA block.

Exercise

1. Press [Next Block] to move to the CLIENT DATA block and then press [Clear Block].

   Note that the client's ID number, 103, is automatically entered in the Client ID field.

2. Press [Next Field].

3. Type `JUST TENNIS` in the Client Name field and press [Next Field].
4. Type `NORTHSIDE MALL` for the Address, then press [Next Field].
5. Type `BURLINGAME` for City, and press [Next Field].
6. Type `ca` — in lowercase letters — in the State field.

   Notice that, although you typed the state in lowercase letters, SQL*Forms converted it to uppercase letters. This is because the form's designer designated this field as an uppercase field.

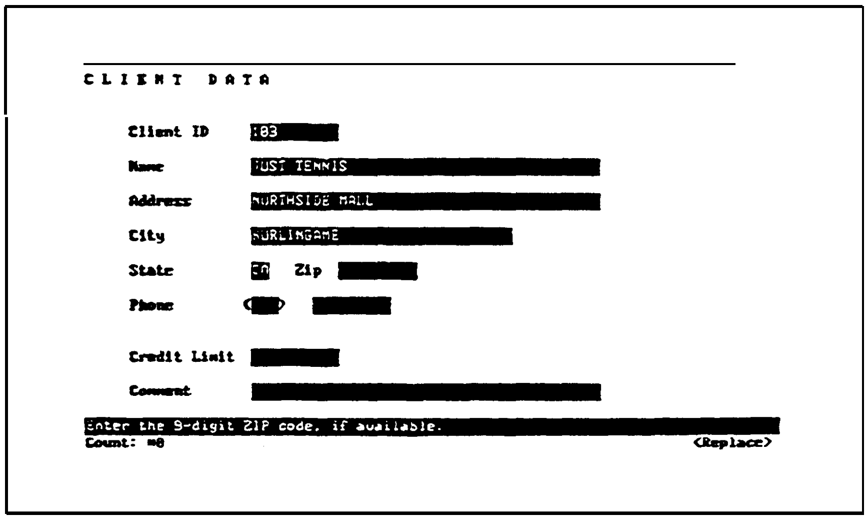
   In addition, once you typed the state abbreviation, your cursor automatically skipped to the next field—even though you did not press [Next Field]. In this case the designer has designated State as an autoskip field. Once you enter the correct number of characters in the field, the cursor automatically moves to the next field. The autoskip feature only works if entries in the field are all the same length.

   Now that you are in the Zip field, notice that SQL*Forms has automatically displayed the help message:

   ```
   Enter the 9-digit ZIP code, if available.
   ```

   The form's designer can choose to have help messages displayed automatically, whenever you enter certain fields. (See Figure 6-9.)

FiGURE 6-9
Automatic Help for the ZIP Field

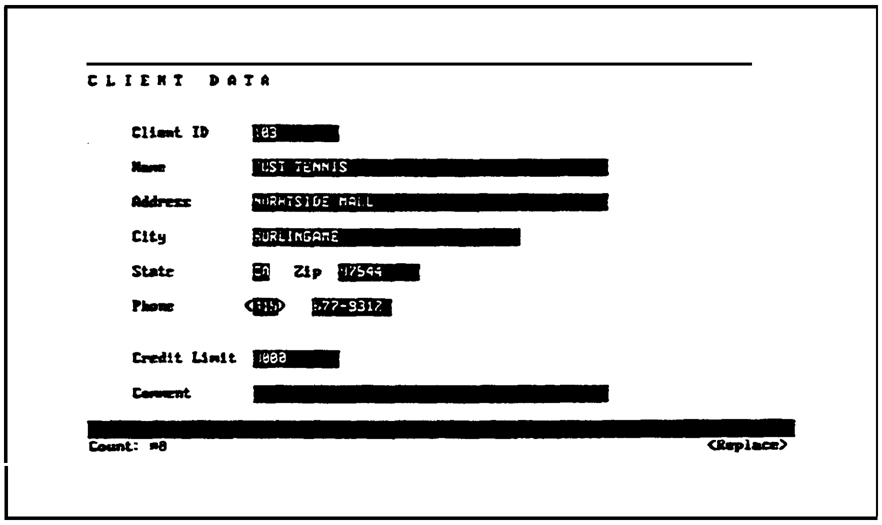


```
CLIENT DATA

     Client ID      103
     Name           JUST TENNIS
     Address        NORTHSIDE MALL
     City           BURLINGAME
     State          CA   Zip
     Phone          (  )

     Credit Limit
     Comment
Enter the 9-digit ZIP code, if available.
Count: *0                                                  <Replace>
```

7. Type 97544 in the Zip field and press [Next Field].
8. Type 415 in the area code section of the Phone field.

   Note that the area code section is actually a separate field. This field also has the autoskip feature, so you are now in the phone number section of the Phone field.
9. Type 677-9312 in the phone number field and press [Next Field].
10. Type 3000 in the Credit Limit field. At this point, your screen should look like Figure 6-10.

FIGURE 6-10
Completed Entry in CLIENT DATA Block



```
CLIENT DATA

    Client ID      103
    Name           JUST TENNIS
    Address        NORTHSIDE MALL
    City           BURLINGAME
    State          CA  Zip  97544
    Phone         (415)  677-9312

    Credit Limit   3000
    Comment

Count: *8                                         <Replace>
```

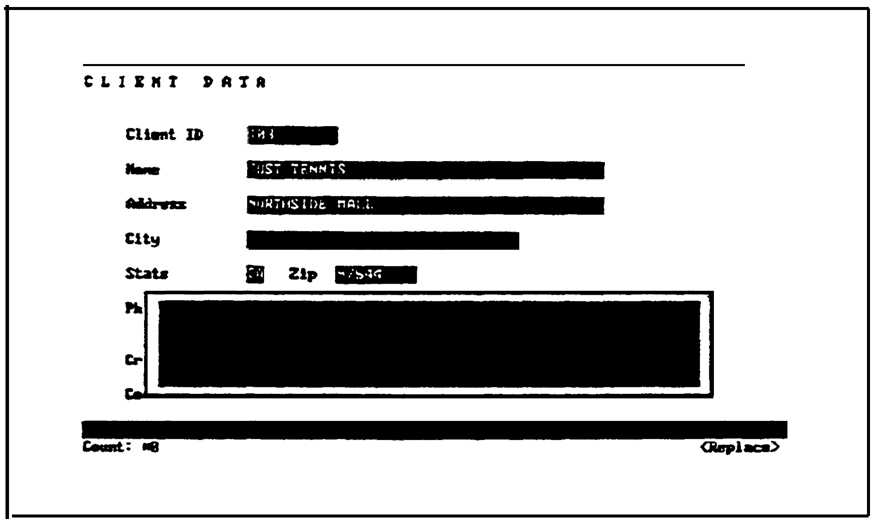
In the following exercise you add comments to your entry of the new order.

Adding Comments to the CLIENT DATA Block

Complete the following steps to add comments to the CLIENT DATA block:

1. Make sure the cursor is in the Comments field.
2. Press [Edit] to invoke the field editor. The Edit pop-up window appears on the screen (see Figure 6-11).

FIGURE 6-11
Edit Pop-Up Window



```
CLIENT DATA

Client ID     [illegible]
Name          JUST TENNIS
Address       NORTHSIDE MALL
City
State  [illegible]  Zip  [illegible]
Ph
Cr
Co

Count: #8                                   <Replace>
```

3. Type in the following statement:

   `This account is overdue.`

   Because this comment will appear in the Comment field whenever this record is queried, you want to give more specific information about the account. Use the keys available only in the editor for pop-up windows to revise and expand the statement you just typed. (See Appendix A for key description.)

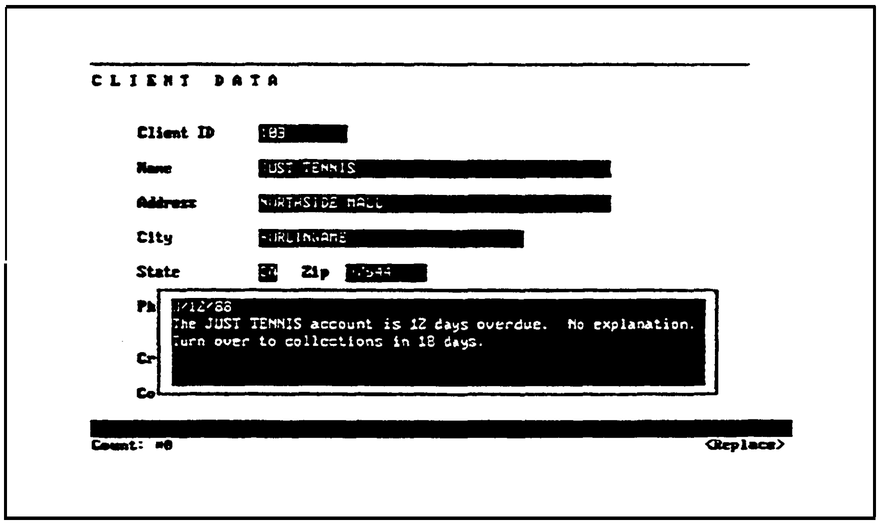
4. Press [Beginning of Line] to move the cursor to the beginning of the line.
5. Press [Delete Line] to clear the pop-up window.
6. Enter the following text in the window:

   ```
   The JUST TENNIS account is 12 days overdue.  No explanation.
   Turn over to collections in 18 days.  3/12/88
   ```

   The date, a highly recommended component of every comment, really belongs at the beginning of a comment. So you need to make room for it at the top of the text, and then cut and paste the date there.

7. Press [First Line] to move the cursor to the top of the text.
8. Insert an extra line at the top of the text by pressing the Return key. This will create a blank line at the beginning of the text.

9. Press [Last Line] to move to the bottom of the text in the pop-up window.
10. Move the cursor to the 3 in the date. [Press Select].
11. Place the cursor on the last 8 in the date; press [Cut].
12. Press [Fit Line] again and then press [Paste]. The date is now on the first line of the pop-up window, shown in Figure 6-12.



```
CLIENT DATA

Client ID   103
Name        JUST TENNIS
Address     NORTHSIDE MALL
City        BURLINGAME
State       [illegible]   Zip   [illegible]
Ph  3/12/88
    The JUST TENNIS account is 12 days overdue.  No explanation.
    Turn over to collections in 18 days.
Cr
Co

Count: *0                                             <Replace>
```

FIGURE 6-12
Completed Edit Pop-Up Window

13. Press [Edit] again or [Commit/Accept] to remove the editing window.
14. Press [Exit/Cancel] and answer “No” to the commit alert. You have no need to commit these changes because you have just completed all the exercises in the Guide.

    SQL*Forms return you to your system's command line.

CHAPTER

7 SQL*FORMS PROCESSING

This chapter contains information about the way SQL*Forms processes, especially while transactions are being committed to the database. In this chapter, you will find descriptions of the following processes:

- committing a transaction
- rolling back a transaction
- using automatic record locking

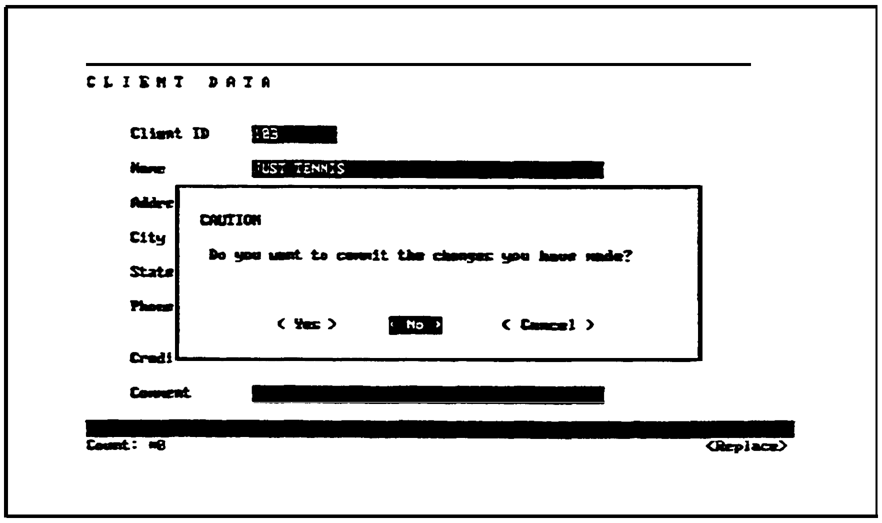
Committing a Transaction

When you use a form, the modifications you make to database tables are not recorded directly in the database; rather, they are recorded in the work space. To make your modifications permanent, you must commit the contents of the workspace to the database by pressing [Commit/Accept]. After you have pressed this function key, data in the work space is recorded in a table. Note that [Commit/Accept] affects your work in every block of the current form, not just in the current block.

Note that while modifications are held in your workspace, other ORACLE users cannot see them. Therefore, if another user fetches a row after you have modified it—but before you have committed it—that user will see the unmodified version of the row.

The Commit Alert

Many operation in SQL*Forms help you protect your modifications before you can proceed. For example, if you have modified the current block but have not committed the modifications, SQL*Forms displays an alert (see Figure 7-1) when you press [Enter Query].



```
CLIENT  DATA

Client ID   123
Name        JUST TENNIS
Addre
City    CAUTION
State   Do you want to commit the changes you have made?
Phone
               < Yes >     < No >     < Cancel >
Credi
Comment

Count: *8                                              <Replace>
```

FIGURE 7-1
The Commit Alert

Press [Next Field] to reach the response you want and press [Select] to choose it. If you respond “Yes,” SQL*Forms commits the pending modifications. If you respond “No,” SQL*Forms discards the modifications for the current block. In either case, SQL*Forms then proceeds with the operation you requested. If, however, you respond "Cancel," SQL*Forms returns you to where you were in the form without committing any changes.

The following operations will prompt you with the above alert if there are changes to commit:

- [Enter Query]
- [Execute Query]
- [Exit/Cancel]
- [Clear Block]
- [Clear Form/Rollback]
- [Count Query Hits]

Note that a form can be designed to execute a commit at any time, and it may perform the commit without notifing you.

Validity Checking during a Commit

During a commit, a form can check the validity of data in numerous ways. However, these validity checks are concerned only with relationships among fields and records, not blocks.

Validity checks performed during a commit may include:

- checks for uniqueness. These checks prevent any two rows of a table from having the same value in given field. For example, this check prevents two records from having the same order number.
- checks for consistency. For example, a consistency check might ensure that the sum of the detail lines in an order matches the total order value.

If an error is detected, SQL*Form displays a message informing you of the error. This message maybe a standard SQL*Forms message, or it may be a message created by the form’s designer. Next, SQL*Forms moves the cursor to the record and field where the error was detected.

Note: If any record in the work space fails to pass a validity check, the entire commit operation fails; nothing in the workspace is committed. You must correct the error and commit again.

Rolling Back a Transaction

To discard the contents of the work space, you can perform a rollback by pressing [Clear Form / Rollback]. Once you have committed a transaction to the database, [Clear Form / Rollback] will not discard it.

Note that you can use [Clear Block] to clear the part of the work space that holds records for the current block.

Using Automatic Record Locking

SQL*Forms provides automatic record locking to prevent two or more users from updating the same record at the same time. When you attempt to update a record, SQL*Forms determines whether the record has been up dated or deleted by another user since you executed the query that retrieved the record. If the record has been updated or deleted since you retrieved it, you will have to re-execute the query in order to see and work with the revised record. If, however, the record has not been changed, SQL*Forms will lock the record so that other users cannot make modifications while you are updating it.

Under certain circumstances,you may want to lock a record before automatic locking takes place. (For example, you may want to lock an order while you modify its items.) If you need to lock a word before automatic locking takes place, press [Lock Record].

APPENDIX

A FUNCTION KEYS

This appendix lists the entire range of function keys and then groups them into the six categories for the operator's interface. The function key groupings include the following categories:

- cursor movement keys
- editing keys
- query processing keys
- database maintenance keys
- block mode function keys
- user assistance keys

Function Key Definitions

[Beginning of Line]	Moves the cursor to the first character in the line. Available only in editing mode.
[Block Menu]	Displays a list of all the blocks in the current form. From this list, you can select the block to which you would like to move. When you choose a block from the block menu, the cursor moves to that block. If a form has many blocks, [Block Menu] will usually move the cursor to a block more quickly than [Next Block] or [Previous Block]
[Clear Block]	Clears all records from the current block and creates an new record. [Clear Block] might prompt you to commit your changes. [Clear Block] does not delete records from the database; it only removes records from the work space.
[Clear Field]	Clears the contents of the current field beginning at the current cursor position. If the cursor is to the right of all the characters in the field, [Clear Field] clears the field. In the pop-up editor, [Clear Field] clears to the end of the line.
[Clear Form/Rollback]	Clears all the blocks of the current form, deletes all data in all blocks of the form, and does a rollback. [Clear Form/RollbackJ might prompt you to commit your changes. [Clear Form/Rollback] undoes all inserts, updates, and deletes posted to the database. [Clear Form/Rollback] does not delete records from the database; it only removes records from the work space.

[Clear Record]	Removes the current reocord from the current block, reversing any uncommitted changes made to that record. A cleared record is NOT deleted from the database.
[Commit / Accept]	Closes the dialog box and acts upon your entry. Enters into the database all changes made since the last [Commit / Accept] or [Clear Form / Rollback].
[Copy]	Copies area of text after it has been selected with [Select] and stores it in the paste buffer.
[Count Query Hits]	Clears the current block and displays on the message line the number of rows that a query would retrieve if executed. If you are in Enter Query mode, the current block does not clear. When used in Enter Query mode, [Count Query Hits] counts the number of records matching the specified search criteria. Note: If you press [Count Query Hits] after Pressing [Execute Query], it terminates the query clears all the records from the work space, and counts all of the records in the table that can be retrieved by the block.
[Cut]	Cuts an area of text after it has been selected with [Select] and stores it in the paste buffer.
[Delete Backward]	Deletes the character to the left of the current cursor position.
[Delete Character]	Deletes the character at the current cursor position
[Delete Line]	Deletes the current line. Available only in the pop-up editor.
[Delete Record]	Deletes a retrieved record from the screen and from the database. Records are not permanently deleted until you commit your changes to the database.

[Display Error]	Displays error information and/or advanced help information, if available, for the field where the last error occurred.
[Down]	Moves the cursor to the field in the next record. If then next record is a new record, [Down] moves the cursor to the first field of the new record.
[Duplicate Field]	Copies the field value from the same field of the previous record into the current field.
[Duplicate Record]	Copies all field values from the previous record into a new record.
[Edit]	Displays a pop-up window in which the operator can edit a field. [Edit] is a toggle switch—pressing it twice dissolves the window. Press [Exit/Cancel] to dissolve the window without accepting its contents.
[End of Line]	Moves cursor to the right of the last character in the line. Available only in the pop-up editor.
[Enter]	Transmits and validates data but does not commit the data to the database.
[Enter Query]	Clears the current block and allows you to enter query criteria. In Enter Query mode, the following keys have these functions: • [Enter Query] displays the query criteria last used. • [Execute Query] performs the query. If records are retrieved, returns to normal operation; however, if no records are found, remains in Enter Query mode. • [Exit/Cancel] returns to normal operation without performing the query. • [Count Query Hits] displays the number of rows that satisfy the current query Criteria.

[Execute Query]	Clears the current block and retrieves all the records from the database table referenced by the block (Only those records that can fit on the screen are displayed.) When used after [Enter Query], [Execute Query] executes a query with the criteria you have specified.
[Exit/Cancel]	Exits the current form and returns to the system command prompt. [Exit/Cancel] also terminates query processing or interrupts the [List] function. In editor, functions like [Undo] by undoing all changes in that session. [Exit/Cancel] also dissolves the editing window.
[First Line]	Moves the cursor to the top of the text in the window. Available only in the pop-up editor.
[Help]	Displays brief help message for the current field. Pressing [Help] twice may display advanced help information if available for that field.
[Insert Line]	Inserts line break at any point in the editor and creates a blank Iine after the current line. Available only in the pop-up editor.
[Insert Record]	Inserts a new record after the current record.
[Insert/Replace]	Toggle between Insert character mode and Replace character mode.
[Last Line]	Moves the cursor to the bottom of the text in the window. Available only in the pop-up editor.
[Left]	Moves the cursor one character to the left (within a field or a line).

[List]	Activates a list of values, if there is one available for this field. Following are the two types of lists of values: 1. If a pop-up window appears, the window will display an enterable field and a list of possible values for the current field. Press [Next Field] to move the cursor to the enterable field, enter search criteria, and press [List]. SQL*Forms executes the selection and returns the cursor to the list. Use the cursor keys or [Scroll Up] and [Scroll Down] to navigate through the list. Press [Select] to choose the selection and dissolve the list of values pop-up. Press [Exit/Cancel] to leave the list without a selection. 2. If nothing pops up when you press [List], that field will display possible values for the current field. Press [Next Field] to see subsequent values. Press [Exit/Cancel] to select a value.
[Lock Record]	Locks a record so that another user cannot change the record while you are updating it. [Lock Record] does not allow you to enter or change any data in a field that is protected against entry or update.
[Menu]	Activates the main menu in SQL*Forms if it is available.
[Next Block]	Moves the cursor to the next block in the form that contains at least one enterable field. (The order is established by the form designer.)
[Next Field]	Moves the cursor to the next enterable field in the current record. (The order is established by the form designer.)
[Next Primary Key Fld]	Moves the cursor to the next enterable field. in the current record that has been designated as part of the "primary key"—those fields that uniquely identify a particular row of a database table.

[Next Record]	Moves the cursor to the next record in the current block. If no more records are found, [Next Record] creates a new blank record (unless the current record is blank).
[Next Set of Records]	Retrieves the next set of records (a number specified by the designer) into the current block from records that satisfy an active query.
[Paste]	Pastes text in paste buffer at current cursor location
[Previous Block]	Moves the cursor to the previous block in the form that contains at least one enterable field. (The order is established by the form designer.)
[Previous Field]	Moves the cursor to the previous enterable field in the current record. (The order is established by the form designer.)
[Previous Record]	Moves the cursor to the previous record in the current block.
[Print]	Writes the current screen to a file and asks if you want to print it.
[Refresh]	Redraws the screen image.
[Right]	Moves the cursor one character to the right (within a field or a line.).
[Scroll Down]	Shifts the window of the current block or list down by approximately 80 percent, displaying records that are outside of the window.
[Scroll Right]	Shifts the field's window to the left by approximately 80 percent of the field's entire display width, displaying field contents that are outside of the window.
[Scroll Right]	Shifts the field's display window to the right by approximately 80 percent of the field's entire display width, displaying field contents that are outside of the window.

[Scroll Up]	Shifts the window of the current block or list up by approximately 80 percent, displaying records that are outside of the window.
[Search]	Displays a dialog box for entering search and replace criteria. Searches forward or backward from the current cursor location. Available only in the pop-up editor.
[Select]	Selects a choice in a list or in a dialog box. In the pop-up editor, marks a point on the screen that SQL*Forms uses for text cutting copying, and pasting.
[Show Keys]	Displays, on the keyboard map, the function key assignments currently in effect. Return to the forms you were working on by pressing [Select].
[Transmit]	Sends data displayed on the current screen to the host computer without validating or committing the data.
[Up]	Moves the cursor to the same field in the previous record.

Function Key Groupings

The function keys in the operator's interface fall into the following five categories:

- cursor movement keys
- editing keys
- query processing keys
- database maintenance keys
- block mode function keys
- user assistance keys

Use these categories when you know what you want to do, but are not sure which key is the proper one.

Cursor Movement Keys

[Left]	[Next Primary Key Fld]
[Right]	[Block Menu]
[Next Field]	[Up]
[Previous Field]	[Down]
[Next Block]	[Scroll Up]
[Previous Block]	[Scroll Down]
[Next Record]	[Scroll Left]
[Previous Record]	[Scroll Right]
[Next Set of Records]	

Editing Keys

[Delete Character]	[Insert Line]
[Delete Backward]	[Beginning of Line]
[Delete Line]	[End of Line]
[Insert/Replace]	[First Line]
[CIear Field]	[Last Line]
[Clear Record]	[Cut]
[Clear Block]	[Copy]
[ClearForm/RollbackJ	[Paste]
[Edit]	[Search]
	[select]

Query Processing Keys

[Count Query Hits]
[Enter Query]
[Execute Query]

Database Maintenance Keys

[Commit/Accept]
[Insert Record]
[Delete Record]

Block Mode Function Keys

[Transmit]
[Enter]
[Lock Record]

User Assistance Keys

[Block Menu]
[Display Error]
[Duplicate Field]
[Duplicate Record]
[Exit/Cancel]
[Help]
[List]
[Menu]
[Print]
[Refresh]
[Show Keys]

APPENDIX

B SAMPLE TABLES

This appendix lists the sample tables that are used throughout this Guide. The following sample tables are included:

- the ORD table
- the ITEM table
- the CUSTOMER table

ORD Table

ORDID	*ORDERDATE*	*C*	*CUSTID*	*SHIPDATE*	*TOTAL*
601	01-MAY-86	A	106	30-MAY-86	2.4
602	05-JUN-86	B	10 2	20-JUN-86	5 6
603	05-JUN-86		102	05-JUN-86	224
604	15-JUN-86	A	106	30-JUN-86	698
605	14-JUL-86	A	106	30-JUL-86	8324
606	14-JUL-86	A	100	30-JUL-86	3.4
607	18-JUL-86	C	10 4	18-JUL-86	5.6
608	25-JUL-86	C	104	25-JUL-86	35.2
609	01-AUG-86	B	10 0	15-AUG-86	97.5
610	07-JAN-87	A	101	08-JAN-87	101.4
611	11-JAN-87	B	102	11-JAN-87	45
612	l5-JAN-87	C	104	20-JAN-87	5860
613	01-FEB-87		108	01-FEB-87	6400
614	01-FEB-87		102	05-FEB-87	23940
615	01-FEB-87		107	06-FEB-87	710
616	03-FEB-87		103	10-FEB-87	764
617	05-FEB-87		105	03-MAR-87	46370
618	15-FEB-87	A	102	06-MAR-87	3510.5
619	22-FEB-87		104	04-FEB-87	1260
620	12-MAR-87		100	12-MAR-87	4450
621	15-MAR-87	A	100	01-JAN-87	730

ITEM Table

ORDID	***ITEMID***	***PRODID***	***ACTUALPRICE***	***QTY***	***ITEMTOT***
601	1	200376	2.4	1	2.4
602	1	100870	2.8	20	56
603	2	100860	56	4	224
604	1	100890	58	3	174
604	2	l00861	42	2	84
604	3	100860	44	10	440
605	1	100861	45	100	4500
605	2	100870	2.8	500	1400
605	3	100890	58	5	290
605	4	101860	24	50	1200
605	5	101863	9	100	900
605	6	102130	3.4	10	34
606	1	102130	3.4	1	3.4
607	1	100871	5.6	1	5.6
608	1	101860	24	1	24
608	2	100871	5.6	2	11.2
609	1	100861	35	1	35
609	2	100870	2.5	5	12.5
609	3	100890	50	1	50
610	1	100860	35	1	35
610	2	100870	2.8	3	8.4
610	3	100890	58	1	58
611	1	100861	45	1	45
612	1	100860	30	100	3000
612	2	100861	40.5	20	810
612	3	101863	10	150	1500
612	4	100871	5.5	100	550
613	1	100871	5.6	100	560
613	2	101860	24	200	4800
613	3	200380	4	150	600
613	4	200376	2.2	200	440

Continued on next page.

ITEM Table continued.

ORDID	***ITEMID***	***PRODID***	***ACTUALPRICE***	***QTY***	***ITEMTOT***
614	1	100860	35	444	15540
614	2	100870	2.8	1000	2800
614	3	100871	5.6	1000	5600
615	1	100861	45	4	180
615	2	100870	2.8	100	280
615	3	100871	5	50	250
616	1	100861	45	10	450
616	2	100870	2.8	50	140
616	3	100890	58	2	116
616	4	102130	3.4	10	34
616	5	200376	2.4	10	24
617	1	100860	35	50	1750
617	2	100861	45	100	4500
617	3	100870	2.8	500	1400
617	4	100871	5.6	500	2800
617	5	100890	58	500	29000
617	6	101860	24	100	2400
617	7	101863	12.5	200	2500
617	8	102130	3.4	100	340
617	9	200376	2.4	200	480
617	10	200380	4	300	1200
618	1	100860	35	23	805
618	2	100861	45.11	50	2255.5
618	3	100870	45	10	450
619	1	200380	4	100	400
619	2	200376	2.4	100	240
619	3	102130	3.4	100	340
619	4	100871	5.6	50	280
620	1	100860	35	10	350
620	2	200376	2.4	1000	2400
620	3	102130	3.4	500	1700
621	1	100861	45	10	450
621	2	100870	2.8	100	280

CUSTOMER Table

CUST ID	*Name*	*ADDRESS CITY*	*ST ZIP*	*AREA/PHONE*	*REPID*	*CREDIT LIMIT*	*COMMENT*
101	TKB SPORT SHOP	490 BOLI RD. REDWOOD CITY	CA 94061	415 / 368-1223	7521	10000	
102	VOLLYRITE	9722 HAMILTON BURLINGAME	CA 95133	415 / 644-3341	7654	7000	
103	JUST TENNIS	NORTHSIDE MALL BURLINGAME	CA 97544	415 / 677-9312	7521	3000	
104	EVERY MOUNTAIN	574 SURRY RD. CUPERTINO	CA 93301	408 996-2323	7499	10000	
105	K+T SPORTS	3476 EL PASEO SANTA CLARA	CA 91003	408 376-9966	7844	5000	
106	SHAPE UP	908 SEQUOIA PALO ALTO	CA 94301	415 364-9777	7521	6000	
107	WOMENS SPORTS	VALCO VILLAGE SUNNYVALE	CA 93301	408 967-4398	7499	10000	
108	NORTHWOODS HEALTH AND FITNESS SUPPLY CENTER	98 LONE PINE WAY HIBBING	MN 55649	612 566-9123	7844	8000	

APPENDIX

C MESSAGES

This appendix contains a complete numerical listing of error messages in SQL*Forms (Run Form). Also included are a description of the cause of each message and a recommended action.

Attempting to reserve record for update or delete (^C to cancel)...

Cause: At the moment, another user has this row reserved. When that user is done, SQL*Forms will process your new update. In the meantime, you must wait.

Action: Another reserve for update message will eventually appear. If you don't want to wait, press CTRL-C (the key sequence may be different in some environments) to cancel.

Working...

Cause: SQL*Forms has started a potentially time-consuming process or action.

Action: No action necessary.

FRM-40001: Missing arguments on command line.

Cause: You did not enter all the arguments on the command line necessary to execute the command.

Action: Type `RUNFORM -?` to see the actual syntax and retype the command properly.

FRM-40002: Undefined switch <switch name> on command line.

Cause: You mistyped the command by using an undefined command line switch.

Action: Type `RUNFORM -?` to see the actual syntax and retype the command properly.

FRM-40003: Too many arguments on command line.

Cause: You entered too many arguments on the command line or you mistyped the command.

Action: Type `RUNFORM-?` to see the actual syntax and retype the command properly.

FRM-40007: Invalid user identifier or password. Reenter.

Cause: You entered an incorrect ORACLE username or password.

Action: Retype your username and password properly.

FRM-40008: Name/Password combination too large. Re-enter.

Cause: You typed more than the 61-character limit in a use/name/password combination. Usernames and passwords are at most 30 characters each.

Action: Retype your username and password properly.

FRM-40010: Can't read form by that name.

Cause: You entered a nonexistent form name or typed an incomplete path, you do not have the proper privileges to run the form, or the form has not been generated.

Action: Retype the form name correctly, provide the proper path name, generate the form or contact your DBA.

FRM-40011: Form was created by an old version of SQL*Forms (Generate).

Cause: The FRM file was created with an old and incompatible version of SQL*Forms (Generate).

Action: Regenerate the form or relink SQL*Forms (Generate).

FRM-40012: Form was created by a new version of SQL*Forms (Generate).

Cause: The FRM file was created by a new and incompatible version of SQL*Forms (Generate).

Action: Relink SQL*Forms (Run Form) and regenerate the form.

FRM-40013: Program Error. Error occurred while reading form

Cause: An internal error occurred while SQL Forms was trying to read the FRM file.

Action: Contact your DBA to regenerate the form.

FRM-40014: Not enough memory to load the form.

Cause: Your computer does not have enough memory to run your form. This message refers to the computer's main memory, not to free disk space. It may indicate an error in the way the operating system is configured, an error that prevents SQL*Forms from gaining access to the memory it needs.

Action: Notify the designer. The designer maybe able to modify the form so that it will run. If that is not feasible, your installation must make more memory available, either by modifying the operating system parameters or by adding more memory to the computer.

FRM-40015: Unexpected end of file reading form.

Cause: The form was fragmented or incomplete.

Action: Regenerate the FRM file.

FRM-40019: Unknown screen number to display.

Cause: Internal error.

Action: Contact your DBA.

FRM-40020: Page <page number> too small for this form.

Cause: Designer error. A field is positioned off the page.

Action: Notify the designer.

FRM-40021: Item in form file is too large.

Cause: An internal error occurred while SQL*Forms was reading the form.

Action: Contact your DBA.

FRM-40023: Couldn't create record buffer file.

Cause: SQL*Forms could not create a buffer file because there is not enough space on the disk.

Action: Contact your DBA.

FRM-40024: Out of memory.

Cause: Internal error. Your computer does not have enough memory to run your form. This message refers to the computer's main memory, not to free disk space. It may indicate an error in the way the operating system is configured, which prevents SQL*Forms from gaining access to the memory it needs.

Action: Notify the designer. The designer may be able to modify the form so that it will run. If that is not feasible, your installation must make more memory available, either by modifying the operating system parameters or by adding more memory to the computer.

FRM-40025: Can't suppress screen output without field input.

Cause: You tried to run a form on the command line using incompatible switches. The -w switch works only in conjunction with the -r switch.

Action: Retype the command to include both the -w and -r switches.

FRM-40026: Error opening key script file.

Cause: SQL*Forms can't open the file you specified with the -r command line switch.

Action: Make sure the file exists and the protection is proper. Or create a file with the -r command line switch.

FRM-40027: Error opening display spool file.

Cause: Operating system error. SQL*Forms can't open a file specified with the -w switch because there is insufficient disk space or you have specified an incorrect file name.

Action: Contact your DBA or system administrator.

FRM-40100: At first record.

Cause: You have pressed [Previous Record] while the cursor was at the first record.

Action: No action is necessary.

FRM-40101: Can’t position to a key field. None are directly enterable.

Cause: You pressed [Next Primary Key Fld], but there are no enterable primary key fields in this block.

Action: Use [Next Field] for navigation rather than [Next Primary Key Fld].

FRM-40102: Record must be entered or deleted first.

Cause: You have pressed [Next Record] or [Down] in a context where it is meaningless. The last record in a block is the current record, or the block is empty, or you are in a “new” record in the middle of a block that was created using [Insert Record].

Action: No action is necessary.

FRM-40103: Can’t position to a key field. None are query-able.

Cause: You have tried to use [Next Primary Key Fld] but none of the primary key fields in the block allow you to enter query criteria.

Action: No action is necessary.

FRM-40104: No such block.

Cause: The designer referenced a non-existent block in a GO_BLOCK statement.

Action: Notify the designer.

FRM-40105: No such field.

Cause: The designer referenced non-existent field in a GO_FIELD statement.

Action: Notify the designer.

FRM-40106: No enterable fields in destination block.

Cause: The designer referenced a block with no enterable fields in a GO_BLOCK statement.

Action: Notify the designer.

FRM-40107: Can't position cursor to a non-displayable field.

Cause: The designer referenced a non-displayed field in a GO_FIELD statement.

Action: Notify the designer.

FRM-40200: Field is protected against update.

Cause: You have tried to update a field that does not allow updates.

Action: No action is necessary. You cannot update this field in this form.

FRM 40201: Field is full. Can't insert character.

Cause: SQL*Forms is in insert mode, and the current field is full.

Action: Delete a character to make room for the new character or press [Insert/Replace] to activate replace mode.

FRM-40202: Field must be entered.

Cause: You have not entered a value (or you have deleted a value) in a field that requires data input.

Action: You must enter a value in this field.

FRM-40203: Field must be entered completely.

Cause: You have not entered a complete value (or you have deleted part of a value) in a field that has a fixed length requirement.

Action: Enter a complete value (one that extends to the end of the field).

FRM-40204: Cursor is at beginning of field value.

Cause: You tried to delete a character before the first character position of the field.

Action: Use [Delete Character] to delete the character the cursor is on.

FRM-40205: Cursor is beyond the current field value.

Cause: On a block mode terminal, you positioned the cursor out of a field.

Action: Move the cursor into the field and try the entry again.

FRM-40206: Previous character is currently hidden.

Cause: You tried to delete a character that is off the screen.

Action: Scroll the character you want to delete into view using the arrow keys or [Scroll Left] and Scroll Right].

FRM-40207: Must be in range <low value> to <high value>.

Cause: You entered a value not in the valid field range.

Action: Enter a value in the range shown.

FRM-40208: Form running in query-only mode. Can’t change database fields.

Cause: You have entered a value on a query-only form.

Action: Do not enter values on this form. You may execute queries and view data, but you may not alter existing data or enter new data.

FRM-40209: Field must be of form <format mask>.

Cause: The field value did not satisfy the format mask on the field.

Action: Make sure the field value matches the format mask.

FRM-40300: Field is not a column of the table being queried.

Cause: You entered query criteria into a non-queryable field.

Action: Do not attempt a query on this field.

FRM-40301: Query caused no records to be retrieved. Re-enter.

Cause: You entered a query while in query mode, but no records matched the query criteria. SQL*Forms remains in enter query mode.

Action: Either adjust the query criteria or press [Exit/Cancel] to leave query mode.

FRM-40302: Cannot enter a query. No fields are queryable.

Cause: You pressed [Enter Query] with the cursor in a block that does not have any queryable fields.

Action: No action necessary.

FRM-40350: Query caused no records to be retrieved.

Cause: The current query has fetched no records from the table. Either the table is empty or it contains no records that meet the query's search criteria.

Action: No action necessary.

FRM-40352: Last row of query retrieved.

Cause: You pressed [Down], [Next Record], [Next Set of Records] or [Scroll Down] after all records had been retrieved.

Action: No action necessary.

FRM-40353: Query cancelled.

Cause: Either the form executed an ABORT_QUERY function code or you pressed [Exit/Cancel] in Enter Query mode.

Action: No action necessary.

FRM-40355: Query will retrieve <number> records.

Cause: You pressed [Count Query Hits]. If you now press [Execute Query], this number of records will be retrieved.

Action: No action necessary.

FRM-40356: Invalid number in example record. Query not issued.

Cause: In Enter Query mode, you entered an invalid number in the example record.

Action: Correct the entry and retry the query.

FRM-40375: Invalid string in example record. Query not issued.

Cause: In Enter Query mode, you entered an invalid ALPHA or CHAR value in the example record.

Action: Correct the entry and retry the query.

FRM-40358: Invalid date in example record. Query not issued.

Cause: In Enter Query mode, you entered an invalid DATE in the example record.

Action: Correct the entry and retry the query.

FRM-40359: Invalid date or time in example record. Query not issued.

Cause: In Enter Query mode, you entered an invalid JDATE, EDATE, or TIME value in the example record.

Action: Correct the entry and retry the query.

FRM-40400: Transaction complete—<number> records posted and committed.

Cause: You finished a commit.

Action: No action necessary.

FRM-40401: No changes to commit.

Cause: You have tried to perform a commit, but no records have been added or modified in the workspace since the last post or commit.

Action: No action necessary.

FRM-40402: Commit cancelled.

Cause: You pressed CTRL-C or its equivalent while waiting for a lock.

Action: No action necessary.

FRM-40403: A calling form has unposted changes. Commit not allowed.

Cause: A calling form has unposted changes.

Action: Post the changes or return to the calling form and retry the commit.

FRM-40404: Database posting complete— <number> records posted.

Cause: You finished a post.

Action: No action necessary.

FRM-40405: No changes to post.

Cause: You have tried to perform a post, but no records have been added or modified in the workspace since the last post or commit.

Action: No action necessaary.

FRM-40406: Transaction complete—<number> records posted; all records committed.

Cause: You finished a commit that recorded your changes to the workspace and committed previously posted changes.

Action: No action necessary.

FRM-40407: Transaction complete—posted records committed.

Cause: You finished a commit that committed previously posted changes.

Action: No action necessary.

FRM-40501: ORACLE error—unable to reserve record for update or delete.

Cause: A fatal error occurred while SQL*Forms was trying to select the record for update

Action: Pressing [Display Error] provides more information, if it is available. You can also try to update or delete this record later.

FRM-40502: ORACLE error—unable to read list of values.

Cause: A fatal error occurred while SQL*Forms was trying to read a list of values.

Action: Pressing [Display Error] provides more information, if it is available. You can also try to update or delete this record later. Notify the designer of the error and/or contact your DBA.

FRM-40504: ORACLE error—unable to execute a <trigger type> trigger.

Cause: A fatal error occurred while SQL*Forms was trying to execute a trigger.

Action: Pressing [Display Error] provides more information, if it is available. Notify the form designer.

FRM-40505: ORACLE error—unable to perform query.

Cause: SQL*Forms has encountered a processing error. The table associated with the current block of the form may not exist, or your user name may not have authority to perform the specified action on the table.

Action: Pressing [Display Error] provides more information, if it is available. Notify the designer of the error and/or contact your DBA.

FRM-40506: ORACLE error—unable to check for record uniqueness.

Cause: SQL*Forms has encounter a processing error while checking a records primary key fields for uniqueness. The table associated with the current block of the form does not exist, or you do not have authority to access the table.

Action: Notify the designer of the error and/or contact your DBA.

FRM-40507: ORACLE error—unable to fetch next query record.

Cause: A fatal error occurred while SQL*Forms was trying to fetch the next query record.

Action: Notify the designer of the error and/or contact your DBA.

FRM-40508: ORACLE error—unable to INSERT record.

Cause: A fatal error occurred while SQL*Forms was trying to insert a record.

Action: Notify the designer of the error and/or contact your DBA.

FRM-40509: ORACLE error—unable to UPDATE record.

Cause: A fatal error occurred while SQL*Forms was trying to update a record.

Action: Notify the designer of the error and/or contact your DBA.

FRM-40510: ORACLE error—unable to DELETE record.

Cause: A fatal error occurred while SQL*Forms was trying to delete a record. The table associated with the current block of the form may not exist, or your user name may not have authority to perform the specified action on the table, etc.

Action: Notify the designer of the error and/or contact your DBA.

FRM-40600: Row has already been inserted.

Cause: You have attempted to insert or update a record, but SQL*Forms has enforced uniqueness on the block's primary key field. The record, as inserted or updated, is not unique.

Action: Change the values in one or more primary key fields of the current record, making them unique. If the requirment of unique primary key fields creates difficulties for you, ask the designer if that requirement can be eliminated.

FRM-40602: Can't insert into or update data in a view.

Cause: You have modified the contents of a view in a manner that is not permitted. (A view is a set of rows extracted from one or more tables. It functions like a table; a form may refer to both views and tables. During a query, there is no difference between a view and a table. There are, however, certain restrictions on a view's insert, update, and delete operations)

Action: No action is necessary; you cannot perform the operation you have attempted.

FRM-40603: Records no longer reserved for update. Re-query to make changes.

Cause: You committed your modifications in a block where you had previously entered an ENTER_QUERY or EXECUTE_QUERY packaged procedure with the FOR_UPDATE parameter. This released all locks on the records in this block.

Action: If you want to modify the block, you will need to re-query.

FRM-40651: Record is now reserved for update or delete.

Cause: The lock you were waiting for is now in effect

Action: No action necessary.

FRM-40652: Can't lock table in shared update mode.

Cause: You have received this message for one of two reasons:

1. You do not have access to this table.
2. SQL*Forms cannot lock the table in shared update mode.

Action: Contact your DBA.

FRM-40653: Record not reserved for update or delete. Try again later.

Cause: You pressed CTRL-C (or the equivalent) to cancel. The operation that was attempting to update or delete the record terminates with an error.

Action: No action is necessary.

FRM-40654: Record changed by another user. Re-query to see change.

Cause: Another user has updated this record since you performed a query and has changed at least one field in the record. Your actions have not changed the record in memory.

Action: You can update or delete this record now only if another user has restored the field values back to the way they were when you performed the query. Otherwise, you must re-query to fetch and display the new record into the form before you can update or delete it.

FRM-40655: SQL error forced rollback: Clear form and re-enter transaction.

Cause: A deadlock or some other error has caused the current transaction to fail. SQL*Forms has rolled back the changes you have made in the transaction.

Action: Clear the form (or exit and re-enter the form) and re-enter the transaction. Notify the designer. It may be necessary to modify the form's design to prevent the error from recurring.

FRM-40656: Update can't be made due to prior rollback. Clear the record.

Cause: This record was already updated, but when you attempted to commit your changes, a serious error prevented this update or any further update or delete from being performed. The error might have occurred due to one of the following reasons:

1. A deadlock formal the loss of row locks, or
2. The record had a database cluster key, and the previous attempt to update the record m the database was rolled back due to an error somewhere else in the form.

Action: You must clear this record before you can commit any other transactions in the form.

FRM-40657: Record changed or deleted by another user.

Cause: Another user has deleted the record since the query was executed.

Action: You can clear this record from your screen, but you cannot update or delete it since it no longer exists in the database.

FRM-40700: No such triggers <trigger name>.

Cause: Designer error. The form attempted to execute a trigger that doesn't exist, causing a fatal error.

Action: Notify the designer.

FRM-40704: Illegal SQL statement in query-only mode.

Cause: Designer error. The form tried to execute a function that is illegal in a query-only form.

Action: Notify the designer.

FRM-40705: Illegal SQL statement in non-commit-time trigger.

Cause: Designer error. The designer wrote a trigger that contains a SQL statement that is illegal for the trigger type.

Action: Notify the designer.

FRM-40714: Function illegal in this context.

Cause: Designer error. The designer specified an illegal function code.

Action: Notify the designer.

FRM-40720: No statement found in list of statements.

Cause: Designer error. A step in a V2 trigger uses incorrect syntax.

Action: Notify the designer.

FRM-40721: Missing function code in trigger.

Cause: Designer error. A step in a V2 trigger uses incorrect syntax.

Action: Notify the designer.

FRM-40722: Unrecognized function name.

Cause: Designer error. The designer specified a nonexistent function code.

Action: Notify the designer.

FRM-40723: Missing argument for function code.

Cause: Designer error. The designer omitted an argument in an ERASE, EXECUTE_TRIGGER, GO_BLOCK, GO_FIELD, GO_TO, GO_STEP, or NEW_FORM statement.

Action: Notify the designer.

FRM-40724: Missing selector in CASE statement.

Cause: Designer error. The designer omitted the selector portion of the CASE statement.

Action: Notify the designer.

FRM-40725: Missing keyword in CASE Statement.

Cause: Designer error. The designer omitted a keyword in the CASE statement.

Action: Notify the designer.

FRM-40726: Missing choice in CASE statement.

Cause: Designer error. The designer forgot to specify a choice in the CASE statement.

Action: Notify the designer.

FRM-40727: Illegal keyword in CASE statement.

Cause: Designer error. The designer mistyped a keyword in the CASE statement.

Action: Notify the designer.

FRM-40728: Unrecognized or extra arguments to function.

Cause: Designer error. The designer specified extra arguments to a function.

Action: Notify the designer.

FRM-40729: Missing semicolon at end of statement.

Cause: Designer error. The designer omitted the semicolon at the end of the function code.

Action: Notify the designer.

FRM-40730: Invalid message suppress level—unchanged from <current message suppression level>.

Cause: Designer error. The designer set the system message level to an invalid number.

Action: Notify the designer.

FRM-40731: INTO keyword missing.

Cause: Designer error. The designer omitted the INTO clause in the COPY statement.

Action: Notify the designer.

FRM-40733: PL/SQL error-internal SQL*Forms error occurred.

Cause: A fatal error occurred in PL/SQL during trigger execution.

Action: Notify the DBA.

FRM-40734: PL/SQL error—internal PL/SQL error occurred.

Cause: A fatal error occurred in PL/SQL during trigger execution.

Action: Notify the DBA.

FRM-40735: Unhandled exception <exception name> occurred while executing <trigger name> trigger.

Cause: The current trigger raised an exception (other than FORM_TRIGGER_FAILURE), but it did not handle the exception.

Action: Notify the designer.

FRM-40736: Cannot initialize PL/SQL.

Cause: An internal error occurred while initializing PL/SQL.

Action: Notify you DBA.

FRM-40737: Illegal restricte procedure <procedure> in <trigger> trigger.

Cause: Designer error. A trigger tried to execute a restricted packaged procedure.

Action: Notify the DBA.

FRM-40739: CLEAR_FORM with FULL_ROLLBACK not allowed in post-only form.

Cause: Designer error. A trigger tried to issue a CLEAR_FORM packaged procedure with the FULL_ROLLBACK paramenter in a post-only form.

Action: Notify the designer.

FRM-40800: User exit <user exit name> does not exist.

Cause: Designer error. The form has tried to invoke a user exit that does not exist. You may be using the wrong version of SQL*Forms (Run Form), or there maybe an error in the form.

Action: If you are using the wrong version of SQL*Forms (Run Form), determine which version contains the user exit, and use that version to run the form. If there is an error in the form, notify the designer.

FRM-40801: Out of memory processing user exit.

Cause: A fatal error occurred.

Action: Contact your DBA.

FRM-40804: Not enough arguments to COPY command.

Cause: Designer error. The designer used the COPY statement without specifying a source or destination.

Action: Notify the designer.

FRM-40805: Not enough arguments to ERASE command.

Cause: Designer error. The designer used the ERASE statement without specifying a variable.

Action: Notify the designer.

FRM-40807: No command string specified for HOST command.

Cause: Designer error. The designer used the HOST statement without specifying a command string.

Action: Notify the designer.

FRM-40808: Can't execute HOST command. Error code=<hex address>.

Cause: SQL*Forms cannot execute the HOST command because of an operating system error.

Action: Contact your DBA.

FRM-40809: HOST command had error code= <hex address>.

Cause: The operating system command contained the above error code.

Action: Verify that you have the right command.

FRM-40811: Shell command had error code= <hex address>.

Cause: The operating system command contained the above error code.

Action: Verify that you have the right command.

FRM-40814: Illegal function code in this context.

Cause: Designer error. The designer specified the NOFAIL keyword with a function code that does not accept that keyword.

Action: Notify the designer.

FRM-40815: Variable <global variable name> does not exist.

Cause: Designer error. The designer referenced a global variable that does not exist.

Action: Notify the designer.

FRM-40816: Could not allocate memory for new symbol.

Cause: Internal error. SQL*Forms ran out of memory while accessing a new global variable.

Action: Contact your DBA.

FRM-40817: Could not allocate memory for new value.

Cause: Internal error. SQL*Forms ran out of memory while accessing anew global variable.

Action: Contact your DBA.

FRM-40818: System variable name not defined.

Cause: Designer error. The designer referenced a global variable that does not exist.

Action: Notify the designer.

FRM-40819: System variable isn't modifiable.

Cause: Designer error. The designer tried to put a value into a non-modifiable system variable.

Action: Notify the designer.

FRM-40820: Not enough memory to evaluate system variable.

Cause: Internal error. SQL*Forms ran out of memory while accessing a system variable.

Action: Contact your DBA.

FRM-40822: Not enough arguments to SET function.

Cause: Designer error. The designer did not specify enough arguments in the SET function code.

Action: Notify the designer.

FRM-40823: Extra argument to SET function.

Cause: Designer error. The designer specified too many arguments in the SET function code.

Action: Notify the designer.

FRM-40824: Not enough arguments to PAGE function

Cause: Designer error. The designer did not specify enough arguments in the HIDEPAGE or the SHOWPAGE function code.

Action: Notify the designer.

FRM-40825: Not enough arguments to VIEW function.

Cause: Designer error. The designer did not specify enough arguments in the MOVEVIEW, ANCHORVIEW, or RESIZEVIEW function codes.

Action: Notify the designer.

FRM-40826: Invalid number of arguments to CALL function.

Cause: Designer error. The designer did not specify enough arguments in the CALL or the CALLQRY function code.

Action: Notify the designer.

FRM-40827: Not enough arguments to GOREC function

Cause: Designer error. The designer did not specify enough arguments in the GOREC function code.

Action: Notify the designer.

FRM-40831: Truncation occurred: Value too long for field <field name>.

Cause: Designer error. A trigger, query, or user exit read a value into a field that is not long enough to hold the entire value. The field truncated the value.

Action: Notify the designer.

FRM-40901: Note: Not enough memory to remember all or part of this query.

Cause: Internal error.

Action: Contact your DBA.

FRM-40902: SQL statement too large.

Cause: The form's design includes a SQL command that is more than 2048 characters long.

Action: Notify the designer. The designer must shorten the SQL command in question.

FRM-40903: Couldn't create output file.

Cause: You have pressed [Print Screen], but SQL*Forms could not write the contents of the screen to a file. Possible reasons include: you have entered an illegal filename, the operating system does not give you authority to create files, or the necessary disk or directory space is not available.

Action: Check the file name you have entered and correct it if necessary. If you need help, ask your DBA.

FRM-40904: Program error: Don't know operation to be performed on record.

Cause: Internal error.

Action: Contact your DBA.

FRM-40906: FATAL ERROR: Can't write a buffered record to disk.

Cause: SQL*Forms failed while trying to write a buffered record to the disk.

Action: Contact your DBA.

FRM-40907: FATAL ERROR: Can't read a buffered record from disk.

Cause: SQL*Forms failed while trying to read a buffered record from the disk.

Action: Contact your DBA.

FRM-40908: Internal Error: Unknown screen number to display.

Cause: Internal error.

Action: Contact your DBA.

FRM-40911: Record not created due to sequence number generation error.

Cause: Internal error. Either the sequence number does not exist, or you do not have privileges for the number, or another fatal database occurred.

Action: Contact your DBA.

FRM-41000: You pressed an undefined function key.

Cause: You pressed an undefined function key.

Action: Press [Show Keys] to determine which function key you should have pressed.

FRM-41001: Function not allowed on this device.

Cause: You tried to execute a function that is not allowed on a block mode or bit-mapped device.

Action: Press [Show Keys] to determine which function key you should have pressed.

FRM-41002: Invalid selection.

Cause: You entered an invalid selection number on the block menu; that block does not exist in this form.

Action: Select an existing block.

FRM-41003: This block does not correspond to a table. Function ignored.

Cause: You have tried to perform a function that references a table. The current block is a control block (one that does not correspond to any table), so the function you have requested is meaningless.

Action: No action is necessary. You cannot perform the requested function on this block.

FRM-41004: Function key not allowed in this mode.

Cause: You pressed a function key that does not work in this mode.

Action: No action necessary.

FRM-41005: Function key not implemented (internal error).

Cause: You have pressed a disabled function key.

Action: No action is necessary. You cannot use the function key in the context in which it was pressed unless the form's definition is modified.

FRM-41006: Can't find SQL*Forms (Run Form) command.

Cause: You have pressed a function key that has not been defined for your keyboard map.

Action: No action is necessary. You can press [Show Keys] to display the keyboard map.

FRM-41007: Cursor not in a valid field. Function key was ignored.

Cause: You were outside of the field when you pressed the function key.

Action: Position the cursor inside the field and press the function key again.

FRM-41008: Undefined function key. Press [Show Keys] for list of valid keys.

Cause: You pressed an undefined function key.

Action: Press [Show Keys] to determine which function key you should have pressed.

FRM-41009: Function key not allowed. Press [Show Keys] for list of valid keys.

Cause: You pressed a function key that is not allowed in this environment.

Action: Press [Show Keys] to determine which function key you should have pressed.

FRM-41010: Can’t set attribute of the current field.

Cause: A trigger was unable to set an attribute of the current field.

Action: Notify the designer.

FRM-41011: Undefined attribute

Cause: Designer error. A trigger tried to set an undefined attribute.

Action: Notify the designer.

FRM-41013: Undefined attribute specified for field <block name>.<field name>.

Cause: Designer error. A SET_FIELD packaged procedure specifies an undefined parameter.

Action: Notify the designer.

FRM-41014: Cannot set attribute of page 0 field <block name>.<field name>.

Cause: Designer error. A SET_FIELD packaged procedure tried to change some characteristic of a page 0 field.

Action: Notify the designer.

FRM-41015: Cannot set ENTERABLE attribute of the current field <block name>.<field name>.

Cause: Designer error. A SET_FIELD packaged procedure tried to change the Input Allowed characteristic of the current field.

Action: Notify the designer.

FRM-41016: Cannot set DISPLAYED attribute of the current field <block name>.<field name>.

Cause: Designer error. A SET_FIELD packaged procedure tried to change the Displayed characteristic of the current field.

Action: Notify the designer.

FRM-41017: Cannot set UPDATEABLE attribute of the non-enterable <block name>.<field name>.

Cause: Designer error. A SET_FIELD packaged procedure tried to change the Update Allowed characteristic of a non-enterable field.

Action: Notify the designer.

FRM-41018: Cannot set UPDATE_NULL attribute of non-enterable field <block name>.<field name>.

Cause: Designer error. A SET_FIELD packaged procedure tried to change the Update If Null characteristic of a non-enterable field.

Action: Notify the designer.

FRM-41019: Cannot set REQUIRED attribute of non-enterable field <block name>.<field name>.

Cause: Designer error. A SET_FIELD packaged procedure tried to change the Required characteristic of a non-enterable field.

Action: Notify the designer.

FRM-41020: Cannot set ENTERABLE attribute of non-displayed field <block name>. <field name>.

Cause: Designer error. A SET_FIELD packaged procedure tried to change the Input Allowed characteristic of a non-displayed field.

Action: Notify the designer.

FRM-41021: Cannot set QUERYABLE attribute of non-displayed field <block name>.<field name>.

Cause: Designer error. A SET_FIELD packaged procedure tried to change the Query Allowed characteristic of a non-displayed field.

Action: Notify the designer.

FRM-41022: Cannot set REQUIRED attribute of non-updateable field <block name>.<field name>.

Cause: Designer error. A SET_FIELD packaged procedure tried to change the Required characteristic of a non-updateable field.

Action: Notify the designer.

FRM-41800: List of values not available for this field.

Cause: You have pressed [List], but the form does not provide a list of values for this field.

Action: No action is necessary.

FRM-41801: Last value retrieved.

Cause: You have pressed [List] and then pressed [Next Field] after the last value in the list was displayed.

Action: Enter a field value or press [List] again to redisplay the list of possible values.

FRM-41802: Duplicate Record function allowed on new records only.

Cause: You have pressed [Duplicate Record], but the current record is the one that has been fetched from the database.

Action: No action is necessary. You can use [Duplicate Record] only when creating a new record.

FRM-41803: No previous record to copy value from.

Cause: You have pressed [Duplicate Field] or [Duplicate Record], but the current record is the first record in the block.

Action: No action is necessary. [Duplicate Field] and [Duplicate Record] are meaningless in this context.

FRM-41804: Variable wasn't entered <variable name>.

Cause: You have entered a response to the Query Where alert that contained an invalid placeholder (one not used in any of the query fields).

Action: Correct the placeholder in your Query Where response, or define it in one of the query fields. Now re-execute the query.

FRM-41805: Ambiguous field name: <fieId>.

Cause: The current trigger references a field in two places.

Action: Notify the designer.

FRM-41806: Too many variables used.

Cause: You used more than 25 relational operators in your query.

Action: Reduce the number of substitution variables and re-query.

FRM-41807: This key hasn't yet been entered. Try again.

Cause: You didn't enter data in a primary key field.

Action: Enter data in all primary key fields and re-execute.

FRM-41809: Error initializing SQL*Menu.

Cause: An internal SQL*Menu error occurred when you tried to use SQL*Menu from within SQL*Forms.

Action: Contact your DBA.

FRM-41810: Error creating menu.

Cause: An internal SQL*Menu error occurred when you tried to use SQL*Menu from within SQL*Forms.

Action: Contact your DBA.

FRM-41811: Error removing menu.

Cause: An internal SQL*Menu error occurred when you tried to use SQL*Menu from within SQL*Forms.

Action: Contact your DBA.

FRM-41812: Error resetting menu.

Cause: An internal SQL*Menu error occurred when you tried to use SQL*Menu from within SQL*Forms.

Action: Contact your DBA.

FRM-41813: Form exited by debug mode.

Cause: You selected the Exit SQL*Forms (Run Form) option on the Break Processing menu.

Action: No action required.

FRM-41814: Invalid page position.

Cause: Designer error. A trigger tried to move or resize a view to a pop-up page that would cause all or part of the view to display off the screen.

Action: Notify the designer.

FRM-41900: Run aborted by fatal error.

Cause: Fatal error caused form to abort.

Action: Contact your DBA.

FRM-41901: Error: <number> cursors were not closed.

Cause: Internal error.

Action: Contact your DBA.

FRM-41902: Total cursors used <number>.

Cause: This message appears when you specify the -s switch on the command line.

Action: No action necessary.

FRM-42100: No errors encountered recently.

Cause: You have pressed [Display Error], but no error has occurred recently.

Action: No action is necessary.

FRM-42400: Performing event trigger <trigger name>.

Cause: This informative message occurs during a trigger when debug mode is specified.

Action: No action neccessary.

FRM-42401: Performing program trigger <trigger name>.

Cause: This informative message occurs during a trigger when debug mode is specified.

Action: No action necessary.

FRM-42536: Use [Next Field] key to enter answer.

Cause: SQL*Forms is waiting for your response to an alert.

Action: Enter an appropriate response and press [Next Field].

FRM-50001: Acceptable characters are a-z, A-Z, and space.

Cause: You entered an unacceptable character into the field.

Action: Enter a character from a-z, A-Z, or a space.

FRM-50002: Month must be between 1 and 12.

Cause: You have entered an invalid month value in a date.

Action: Enter a month value from 1 (for January) to 12 (for December).

FRM-50003: Year must be 00-99 or 1000-4712.

Cause: You have entered an invalid year number.

Action: Enter a valid year. The year may be a number between 0 and 99 (representing the years 1900 to 1999) or between 1000 and 2100.

FRM-50004: Day must be between 1 and last of month.

Cause: You have entered an invalid day.

Action: Enter a valid day. For April, for example, enter a number between 1 and 30.

FRM-50006: Legal characters are 0-9 + and -.

Cause: You have entered an unacceptable character in a number field.

Action: Enter a valid number. A valid number has digits 0 through 9. A number maybe preceded by a + or - sign. If the message allows it, a number may contain one decimal point at any location, except before the sign.

FRM-50007: Too many decimal digits.

Cause: You have entered a number with 3 or more digits after the decimal point in a money field.

Action: Re-enter a valid number.

FRM-50009: Too many decimal points.

Cause: You have entered a number that contains two or more decimal points, or you have entered a number that contains a decimal point in a field that requires a whole (non-decimal) number.

Action: Enter a number with no more than one decimal point. If you have used only one decimal, remove the decimal and the decimal part of the number.

FRM-50010: Format is: [+-]9999999.99

Cause: You have entered an invalid value in a MONEY or RMONEY field.

Action: Enter a valid value. This value should have zero or dollar digits, followed by a decimal and two cents digits. The entire number may be preceded by a + or a -.

FRM-50011: Not a valid month name.

Cause: You have entered an invalid month name in a date.

Action: Enter a valid month name. SQL*Forms recognizes the first three characters of a month's actual name. For example, JAN stands for January, JUN for June.

FRM-50012: Date format is DD-MON-YY.

Cause: You have entered an invalid date.

Action: Re-enter the date in the requested format. In the format shown above, July 20,1986 would be 20-JUL-86.

FRM-50013: Plus or minus must be in first position.

Cause: You entered the plus or minus sign in the wrong position.

Action: Retype with the plus or minus sign in the first position.

FRM-50014: Bad exponent.

Cause: You entered an exponent in a field that does not accept exponents.

Action: Enter a value without an exponent.

FRM-50015: Too many decimal points.

Cause: You entered too many decimal points.

Action: Retype with only one decimal point.

FRM-50016: Legal characters are 0-9 - + E.

Cause: You have entered an unacceptable character in a number field.

Action: Enter a valid number. A valid number has digits 0 through 9. A number may be preceded by a + or - sign. If the message allows it, a number may contain one decimal point at any location, except before the sign.

FRM-50017: Hour must be between 0 and 23.

Cause: You have entered an invalid hour.

Action: Enter a valid hour. SQL*Forms tells time on a 24-hour basis.

FRM-50018: Minutes must be between 00 and 59.

Cause: You have entered an invalid minute value.

Action: Enter a valid minute value.

FRM-50019: Seconds must be between 00 and 59.

Cause: You have entered an invalid second value.

Action: Enter a value between 00 and 59.

FRM-50020: Missing exponent.

Cause: You have failed to enter an exponent.

Action: Enter a proper exponent.

FRM-50021: Date format is MM/DD/YY.

Cause: You have entered an invalid date.

Action: Re-enter the date in the requested format. In the format shown above, July 20,1986 would be 07/20/86.

FRM-50022: Time format is HH:MM[:SS].

Cause: You have entered an invalid time format.

Action: Re-enter the time in the requested format. For example, 9:30 a.m. is b/9:30:00 (where b/ represents a space); 3:00 p.m. is 15:00:00.

FRM-50023: Date format is DD/MM/YY.

Cause: You have entered an invalid date.

Action: Re-enter the date in the requested format. In the format shown above, July 20,1986 would be 20/07/86.

FRM-50024: Spaces allowed in leading positions only.

Cause: You entered spaces intermixed with data.

Action: Re-enter data with no spaces intermixed.

FRM-50025: Datetime format is DD-MM-YYYY HH:MM[:SS].

Cause: You have entered an invalid date.

Action: Reenter the date and time in the requested format. In the format shown above, July 20, 1986 at 3:00pm would be 20-07-1986 15:00:00.

FRM-50026: Date format is DD-MM-YYYY.

Cause: You have entered an invalid date.

Action: Re-enter the date in the requested format. In the format shown above, July 20, 1986 would be 20-07-1986.

GLOSSARY

alert A window that appears in the middle of the screen, overlaying a portion of the current display. Alerts notify you of a condition that has arisen due to your last action.

autoskip A field characteristic. When a character is entered in the last position of an autoskip field, SQL*Forms skips to the next field of the current block. Autoskip fields are customarily short, fixed-length fields, such as one-character response codes or two-character state name abbreviations.

block A portion of a form containing a group of related fields. An ordinary block displays one or more records whose fields are associated with the columns of a table or view.

characteristic A property of a field in a form that influences the way the field behaves. Examples of characteristics are Input Allowed, Mandatory, and Fixed Length.

column A subdivision of a table with a column name and a specific datatype. For example, in a table of employee information, aIl of the employees' dates of hire would constitute one column. See also *row*.

commit The process of storing new, changed, or deleted records from the work space to a table in the database.

control block A block not associated with any table or view.

control key A special key found on many terminals. When the control key is held down, certain keys perform different functions than they normally do.

count The number of rows selected by the current query. Count is displayed on the SQL*Forms status line.

current block The block the cursor is in. At any time while a form is being run, only one block can be the current block. See *block*.

current field The field in the current block in which data may be entered or edited. At any time while a form is being run, only one field can be the current field.

cursor A marker on the display screen, such as a rectangle or a flashing bar, that indicates your current position.

database (1) The disk space that ORACLE sets aside for storing information in tables. (2) A field characteristic denoting a field that represents a column in the table associated with a block.

DBA Database administrator. A person who authorizes others to use SQL*Forms and who performs other administrative functions concerned with ORACLE operations.

default device type The terminal that SQL*Forms assumes is being used unless told otherwise.

delete To remove a record from the work space or a row from a table.

designer An individual who uses SQL*Forms to create forms. Contrast with operator.

display device A device that a computer program can use to communicate with a person. The display device consists essentially of a display screen and a keyboard.

fetch To retrieve selected rows from a table into the work space during a query.

field (1) In a table, the information stored at the intersection of a row and a column. (2) In a SQL*Forms block, a highlighted or underlined area on the screen that can display an output value or (usually) accept an input value.

fixed length A field characteristic denoting a field whose value must be the same length as the field.

form A set of fields and constant text that SQL*Forms displays on a terminal display screen, so users can see and change data that is stored in database tables.

format mask A designer-defined mask that specifies the display characteristics of information that is entered and queried. For instance, if the designer specified a format mask of $9999.99 for a field, then the entered or retrieved number 55678 will be displayed as "$556.78".

function key A key on your terminal’s keyboard that performs a SQL*Forms function, such as inserting a new record.

help A SQL*Forms function that displays information about the current field. Also, any similar function implemented by a form or by another computer program to give you helpful information.

horizontal scrolling A SQL*Forms feature that enables a field to display a value longer than the field itself. SQL*Forms displays the value a piece at a time by moving it (scrolling it) back and forth within the field.

input allowed A field characteristic that allows a user to enter a value in that field.

insert mode A mode in which each character you enter is inserted at the cursor, pushing the following characters to the right. The opposite of *replace mode.*

keyboard map A chart for your terminal that shows its function keys and their SQL*Forms standard operations.

logical expression A statement about the relationship among two or more field values and/or constants, which may be either true or false. Logical expressions may be used in search criteria through the Query Where alert. An example of a logical expression is :SAL > 1000 AND :COMM = 0; which means the value of the field SAL is greater than 1000, and the value of the field COMM is zero.

mandatory A field characteristic denoting a field whose value may not be null.

message line A line on your screen on which SQL*Forms displays error messages, prompts, and similar information. The message line is customarily the next-to-last line on the screen, immediately above the status line.

multi-record block A block that can display more than one record at a time.

null Empty; describes a field that has no value. Note that a null character field is not the same as a blank field or a field whose value is zero characters long; similarly, a null numerical field is not the same as a numerical field whose value is zero.

operator The end user of a SQL*Forms application. Contrast with designer.

pattern A description of a field's value, which may be used in search criteria to match more than one value. In a pattern, the underscore character (_) will match any single character in a value; the percent character (%) will match any combination and number of characters, including zero characters.

placeholder A word, preceded by a colon, that is entered into a field as part of the search criteria. When a placeholder is used in any field, SQL*Forms prompts for a logical expression, using a Query Where statement. The placeholder may then be used in the logical expression to represent the value of the field in which it was entered.

primary key (1) A group of one or more fields in a block. If the form enforces uniqueness on the primary key, no commit operation can result where two rows in the associated table have the same value in every primary key field. (2) A field characteristic denoting a field that is part of a block's primary key and for which every value must be unique.

primary key field A field that is part of the primary key of a block

query An operation that selects a set of rows from the table associated with the current block and allows you to fetch the records into the work space.

record A unit of information in the SQL*Forms workspace that corresponds to a row in a database table. A record may be fetched from and/or stored into a row.

record locking A feature that protects two users from updating the same row of data at the same time.

relational operator A symbol used in search criteria to indicate a comparison between two values, such as the equal sign in "WHERE DEPTNO = 10". Only those rows are returned (fetched) in which the comparison results in "true."

replace mode A mode in which each character you enter replaces the current character at the cursor. The opposite of *insert mode.*

roll back (v.) The process of clearing all or part of the workspace without committing it to the database. A rollback (n.) discards part or all of the work you have done in the current transaction.

row One set of fields in a table. For example, in a table of employee information, the information about each employee would constitute one row. See also *column.*

RUNFORM The command used to run a form created in SQL*Forms

search criteria Criteria for selecting the rows to be fetched in the course of a query.

shift key A key on your terminal that modifies the effect of other keys while it is held down.

single-record block A block that can display one record at a time.

status line A line on your screen where SQL*Forms displays information about its current status. The status line is usually the last line on the screen.

table A named collection of related information, stored in the ORACLE database. A SQL*Forms block is associated with one base table.

transaction The sum of the modifications made to the workspace between commits or rollbacks.

uniqueness A property of a block; no commit operation in that block may result when two rows of the associated table have the same value in every primary key field.

update To change the value of; particularly, to change the value of a record in the work space or table row.

update allowed A field characteristic denoting a field whose value you may update. Update If Null and Update Allowed are mutually exclusive.

update if null A field characteristic denoting a field whose value you may update only if its value is null. Update If Null and Update Allowed are mutually exclusive.

uppercase A field characteristic denoting a field in which lowercase letters are converted to uppercase as they are entered.

view A database object that may be used like a table, in most respects, allowing a user to retrieve and update selected rows and columns of one or more tables.

work space The place where SQL*Forms keeps records before they are committed to the database.

INDEX