

# CHAPTER 3

## TABLE CREATION AND MANAGEMENT

### LEARNING OBJECTIVES

**After completing this chapter, you should be able to do the following:**

- Identify the table name and structure
- Create a new table using the CREATE TABLE command
- Use a subquery to create a new table
- Add a column to an existing table
- Modify the definition of a column in an existing table
- Delete a column from an existing table
- Mark a column as unused and then delete it at a later time
- Rename a table
- Truncate a table
- Drop a table

### INTRODUCTION

---

Upon joining JustLee Books as a database specialist, you were able to query the existing database; however, new data-tracking features have been requested. The management of JustLee Books has decided to implement a sales commission program for all account managers who have been employed by the company for more than six months. Beginning with orders placed after March 31, 2005, account managers will receive a commission for each order received from the geographical region they supervise. The commission rate will be based on the retail price of books.

To implement this new policy, several changes must be made to the database. It does not contain any information about the account managers, nor does it identify geographical regions in which customers live. Because the account managers represent a new entity, at least one new table must be added to the database.

Chapters 3–5 demonstrate SQL commands to create and modify tables, add data to tables, and edit existing data. This chapter addresses methods for creating tables and modifying existing tables. Commands that are used to create or modify database tables are called **data definition language (DDL)** commands. These commands are basically SQL commands that are used specifically to create or modify database objects. A **database object** is a defined, self-contained structure in Oracle 10g. In this chapter, you will create database tables, which are considered database objects. Later, in Chapter 6, you will learn how to create and modify other types of database objects. Figure 3-1 provides an overview of this chapter's topics.

#### NOTE

This chapter assumes you have created the initial JustLee Books database as instructed in Chapter 2.

OVERVIEW OF CHAPTER CONTENTS	
Commands and Clauses	Description
CREATE TABLE	Creates a new table in the database. The user names the columns and identifies the type of data to be stored. To view a table, use the SQL*PLUS command DESCRIBE.
CREATE TABLE...AS	Creates a table from existing database tables, using the AS clause and subqueries.
Modifying Tables	
ALTER TABLE... ADD	Adds a column to a table.
ALTER TABLE... MODIFY	Changes a column size, datatype, or default value.
ALTER TABLE... DROP COLUMN	Deletes one column from a table.
ALTER TABLE... SET UNUSED <i>or</i> SET UNUSED COLUMN	Marks a column for deletion at a later time.
DROP UNUSED COLUMNS	Completes the deletion of a column previously marked with SET UNUSED.
RENAME...TO	Changes a table name.
TRUNCATE TABLE	Deletes all table rows, but table name and column structure remain.
Deleting Tables	
DROP TABLE	Removes an entire table from the Oracle 10g database.
PURGE TABLE	Permanently deletes a table in the recyclebin.
Recovering Tables	
FLASHBACK TABLE . . . TO BEFORE DROP	Recovers a dropped table if PURGE option not used when table dropped.

**FIGURE 3-1** Overview of chapter contents

## TABLE DESIGN

Before issuing a SQL command to create a table, you must first complete the entity design as discussed in Chapter 1. For each entity, you must choose the table's name and determine its structure—that is, you must determine what columns will be included in the table. In addition, you will need to determine the necessary width of any character or numeric columns.

Let's look at these requirements in more depth. Oracle 10g has the following rules for naming both tables and columns:

- The names of tables and columns can be up to 30 characters in length and must begin with a letter. These limitations apply only to the name of a table or column, not to the data in a column.
- The names of tables and columns cannot contain any blank spaces.
- Numbers, the underscore symbol (`_`), and the number sign (`#`) are allowed in table and column names.
- Each table owned by a user should have a unique table name, and the column names within each table should be unique.
- Oracle 10g “reserved words,” such as `SELECT`, `DISTINCT`, `CHAR`, and `NUMBER`, cannot be used.

Because the new table will contain data about account managers, the name of the table will be `ACCTMANAGER`. The table will need to contain each account manager's name, employment date, and assigned region. In addition, the `ACCTMANAGER` table should contain an ID code to act as the table's primary key and to uniquely identify each account manager. Although it is unlikely that JustLee Books will ever have two account managers with the same name, the ID code can reduce data entry errors because users will need only to type a short code instead of a manager's entire name in SQL commands.

### NOTE

The ID column will be included in the `ACCTMANAGER` table to serve as the primary key for the table. However, the primary key constraint is not defined in this chapter. The next chapter expands the coverage of table creation capabilities by introducing the various constraints that can be defined for a table.

Now that the contents of the table have been determined, the columns can be designed. When you create a table in Oracle 10g, you must define each of the columns. Before you can create the columns, you must do the following:

1. Choose a name for each column.
2. Determine the type of data each column will store.
3. Determine (in some cases) the maximum width of the column.

Before choosing column names, it is useful to look at some datatypes and their default values. Valid datatypes are shown in Figure 3-2.

## ORACLE 10g DATATYPES

Datatype	Description
VARCHAR2( <i>n</i> )	Variable-length character data, where <i>n</i> represents the maximum length of the column. Maximum size is 4000 characters. There is no default size for this datatype; a minimum value must be specified. <i>Example:</i> VARCHAR2(9) can contain up to nine letters, numbers, or symbols.
CHAR( <i>n</i> )	Fixed-length character column, where <i>n</i> represents the length of the column. Default size is 1. Maximum size is 2000 characters. <i>Example:</i> CHAR(9) can contain nine letters, numbers, or symbols. However, if fewer than nine are entered, spaces are added to the right to force the data to reach a length of nine.
NUMBER( <i>p</i> , <i>s</i> )	Numeric column, where <i>p</i> indicates <b>precision</b> , or the total number of digits to the left and right of the decimal position, to a maximum of 38 digits; and <i>s</i> , or <b>scale</b> , indicates the number of positions to the right of the decimal. <i>Example:</i> NUMBER(7, 2) can store a numeric value up to 99999.99. If precision or scale is not specified, the column defaults to a precision of 38 digits.
DATE	Stores date and time between January 1, 4712 B.C. and December 31, 9999 A.D. Seven bytes are allocated to the column to store the century, year, month, day, hour, minute, and second of a date. Oracle 10g displays the date in the format DD-MON-YY. Other aspects of a date can be displayed by using the TO_CHAR format. The width of the field is predefined by Oracle 10g as seven bytes.

57

**FIGURE 3-2** Oracle 10g datatypes

### NOTE

You will discover that Oracle currently has two variable-length character datatypes: VARCHAR and VARCHAR2. Oracle recommends the use of VARCHAR2 rather than VARCHAR, and, therefore, VARCHAR2 is used throughout this text. Oracle's SQL reference states "The VARCHAR datatype is currently synonymous with the VARCHAR2 datatype. Oracle recommends that you use VARCHAR2 rather than VARCHAR. In the future, VARCHAR might be defined as a separate datatype used for variable-length character strings compared with different comparison semantics."

## NOTE

Additional datatypes are available, including BINARY\_FLOAT, BINARY\_DOUBLE, LONG, CLOB, RAW(*n*), LONG RAW, BLOB, BFILE, TIMESTAMP, and INTERVAL. BINARY\_FLOAT is used to support 32-bit, single-precision floating-point numbers. BINARY\_DOUBLE can store a 64-bit, double-precision floating-point number. LONG stores variable-length character data up to 2 gigabytes. CLOB is used for single-byte character data up to 4 gigabytes. RAW(*n*) stores raw binary data up to 2000 bytes. LONG RAW can contain up to 2 gigabytes of unstructured data, whereas BLOB can store up to 4 gigabytes of unstructured binary object data. A BFILE column stores a file locator to a binary file stored by the operating system. TIMESTAMP and INTERVAL are new datatypes supported by Oracle 10g. They are extensions of the DATE datatype, where TIMESTAMP refers to the time value (hour, minute, and second can be referenced without the TO\_CHAR function), and INTERVAL is used to identify a specific interval, or amount, of time.

A **datatype** identifies the type of data Oracle 10g is expected to store in a column. Identifying the type of data not only assists in verifying that you input appropriate data, but it also allows you to manipulate data in ways that are specific to that datatype. For example, imagine you need to calculate the difference in terms of days between two date values such as the Orderdate and Shipdate columns from the ORDERS table. To accomplish this task, the system would need to be able to associate the date values to a calendar. If the columns have a DATE datatype, Oracle 10g automatically associates the values to calendar days.

Let's return to the new ACCTMANAGER table, to which you need to add five columns. The first column will serve to uniquely identify each account manager and will be named AmID. The ID code assigned to each account manager will consist of the first letter of a manager's last name, followed by a three-digit number. Because the column will consist of both letters and numbers, the column will be defined to store the datatype of CHAR and have a width of four. The CHAR datatype should be used only when the length of values to be stored in the column are consistent. In this case, every AmID will have a length of four. Oracle 10g pads a CHAR column to the specified length if an entry does not fill the entire width of the column. This can result in wasted storage space if the majority of the values stored by Oracle 10g must include blank spaces to force the contents of a column to a specific length. If character data to be stored in a column will not be a consistent length, the VARCHAR2 datatype should be used. The VARCHAR2 datatype uses only the physical space required to hold the value input and storage space is conserved because no padding effect takes place.

Why not just always use the VARCHAR2 datatype and avoid the use of CHAR? There is a slight processing overhead in managing the variable-length field. Therefore, there is some advantage to using CHAR when it applies. Many people in the industry use both datatypes; however, this is still a topic of debate.

## NOTE

In this scenario, JustLee Books decided to use the AmID code consisting of characters and digits because this is already being used in the payroll system, and the company hopes to integrate all the systems. It is important to consider codes already in use when trying to determine an appropriate primary key value. If JustLee Books did not have any existing values in use to identify account managers, a system-generated value would most likely be used. Typically, these columns would be numeric to enable sequences to be used to populate the column. Sequences are covered in Chapter 6.

59

The second and third columns of the ACCTMANAGER table will be used to store the first and last name of each account manager. If you store first names and last names in individual columns, you can perform simple searches on each part of a manager's name. Because each account manager's name will consist of characters, these columns will be assigned the datatype of VARCHAR2. Name values vary greatly in length. Therefore, the variable-length datatype is more appropriate than CHAR. Generally, you will define the width so it can hold the largest value that could ever be entered into that column. However, the possibility exists that an account manager will be hired in the future whose first or last name is extremely long. It's relatively easy to increase a column's width at a later time. Therefore, the assumption is that a column width of 12 characters is sufficient to store the names of the current account managers. The columns will be named Amfirst and Amlast.

## NOTE

The actual names of the account managers are provided in Chapter 5.

The fourth column of the table will be used to store the employment date of each account manager. Because the datatype will be DATE, you will not have to worry about the length of the column—the length is predetermined by Oracle 10g. All that remains is to choose a name for the column. In this case, Amedate, for “account manager employment date,” will be appropriate.

The fifth and final column for the ACCTMANAGER table is the region to which the account manager is assigned. The United States will be divided into six geographical regions, each identified by a two-letter code (such as NE for the Northeast region). The name of the column will be Region and will be defined as a CHAR datatype with a width of two characters. The VARCHAR2 datatype could also be used; however, because the values stored in the Region column will always consist of two characters, the CHAR datatype was selected.

## TABLE CREATION

---

The basic syntax of the SQL command to create a table in Oracle 10g is shown in Figure 3-3.

60

```
CREATE TABLE [schema] tablename  
    (columnname datatype [DEFAULT value],  
    [columnname datatype [DEFAULT value], ...]);
```

**FIGURE 3-3** CREATE TABLE syntax

The keywords **CREATE TABLE** instruct Oracle 10g to create a table. The optional **schema** can be included to indicate who will “own” the table. For example, if the person creating the table is also the person who will own the table, the schema can be omitted, and the current user name will be assumed by default. On the other hand, if you were creating the ACCTMANAGER table for someone with the user name of DRAKE, the schema and table name would be entered as DRAKE.ACCTMANAGER to inform Oracle 10g that the table ACCTMANAGER will belong to DRAKE’s schema, not yours. The owner of a database object has the right to perform certain operations on that object. In the case of a table, the only way another database user can query or manipulate the data contained in the table is to be given permission from the table’s owner or the database administrator. The table name, of course, is the name that will be used to identify the table being created.

### NOTE

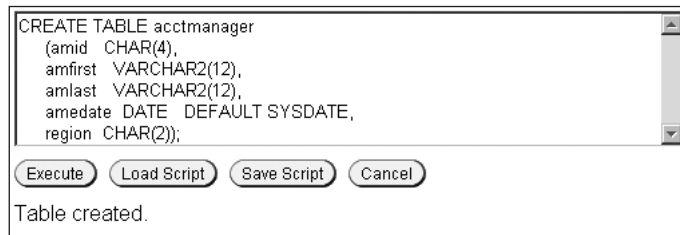
To create a table for someone else’s schema (a table owned by someone else), you must be granted permission, or the privilege, to use the CREATE TABLE command for that user’s schema. The different privileges available in Oracle 10g are presented in Chapter 7.

## Defining Columns

After the table name has been entered, the columns to be included in the table must be defined. A table can contain a maximum of 1000 columns. *The syntax of the CREATE TABLE command requires the column list to be enclosed within parentheses.* If more than one column will exist in the table, then the name, datatype, and appropriate width are listed for the first column before the next column is defined. Commas separate columns in the list. The CREATE TABLE command also allows a default value to be assigned to a column. The default value is the value that will automatically be stored by Oracle 10g if the user makes no entry into that column. For example, imagine users entering order data. Rather than have the user spend time entering the current date as the order date, have the system do it automatically by setting the DEFAULT value to the current date on the system.

Using the syntax presented in Figure 3-3, the SQL command in Figure 3-4 shows the creation of the ACCTMANAGER table.





**FIGURE 3-4** The creation of the ACCTMANAGER table

In the command given in Figure 3-4, the name of the table is ACCTMANAGER. It is entered here in lower case letters to distinguish it from the CREATE TABLE keywords. Oracle 10g SQL commands are not case sensitive; however, for clarity, in this text all keywords are in uppercase and all user-supplied values are in lower case. Because the user who creates the table will also be the owner of the table, the schema has been omitted.

The five columns to be created are listed next, within parentheses. Each column is defined on a separate line simply to improve readability; this is not an Oracle 10g requirement. Notice the definition for the Amedate column; it has been assigned a default value of SYSDATE. This instructs Oracle 10g to insert the current date (according to the Oracle 10g server) if the user enters a new account manager without including the individual's date of employment. Of course, this is only beneficial if the account manager's record is created on the same date the person is hired; otherwise, this makes an incorrect entry if the Amedate column is left blank. In other words, assign defaults with caution!

## NOTE

A user cannot have two tables with the same name. If you attempt to create a table with the same name as another table in your schema, Oracle 10g displays an ORA-00955 error message. Similarly, if you create a table and then want to create another table using the same table name, you must first delete the existing table using the command `DROP TABLE tablename;`. (The DROP TABLE command is shown later in this chapter.)

Notice that after the command has been executed, Oracle 10g returns the message "Table created" to let the user know the table was created successfully. The message does not contain any reference to rows being created. At this point, the table does not contain any data; only the structure of the table has been created (in other words, the table has been defined in terms of a table name and the type of data it will contain). The data, or rows, must be entered into the table as a separate step using the INSERT command. You will enter all the data for the account managers in Chapter 5.

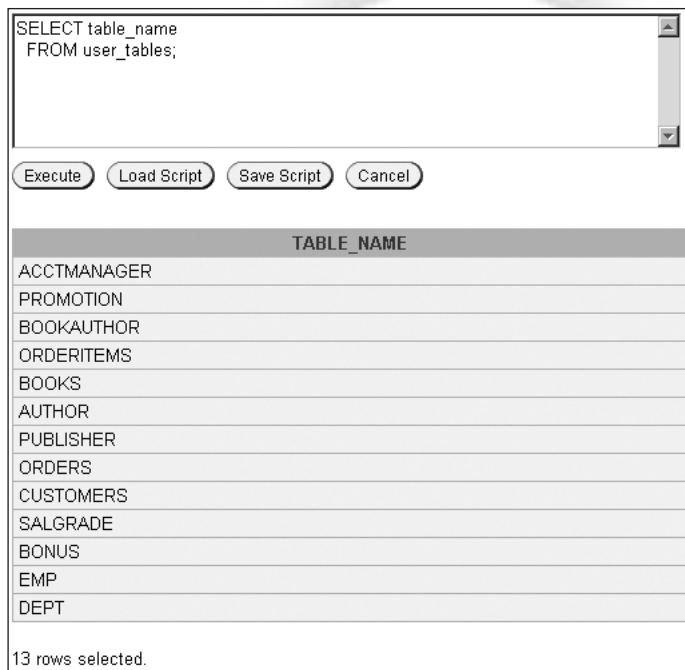
## NOTE

If you receive an error message (such as an ORA-00922 error message) when executing the CREATE TABLE command, it could be a result of either (1) not including a second closing parenthesis at the end of the command to close the column list, or (2) not separating each complete column definition with a comma. If an error message appears stating that you do not have sufficient privileges, ask the database administrator to grant you the CREATE TABLE privilege.

62

### Viewing List of Tables: USER\_TABLES

A data dictionary is a typical component of a DBMS that maintains information about database objects. You can query the data dictionary to verify all the tables that exist in your schema. The USER\_TABLES data dictionary object maintains information regarding all your tables. Figure 3-5 demonstrates querying the data dictionary to generate a list of all your table names. Imagine how important this can be when you start a new job and need to explore an existing database!



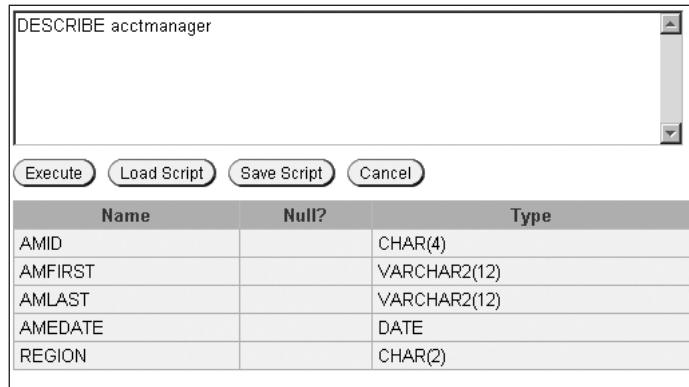
**FIGURE 3-5** Listing names of all tables

## NOTE

Your list should show all the tables previously created for the JustLee Books database. You might or might not see the additional tables named SALGRADE, BONUS, EMP, and DEPT. These tables are created during the installation in the SCOTT user schema.

## Viewing Table Structures: DESCRIBE

To determine whether the structure of the table was created correctly, you can use the SQL\*Plus command `DESCRIBE tablename` to display the structure of the table, as shown in Figure 3-6. Because the DESCRIBE command is a SQL\*Plus command rather than a SQL command, it can be abbreviated and an ending semicolon is not required. The abbreviation for the DESCRIBE command is `DESC`.



Name	Null?	Type
AMID		CHAR(4)
AMFIRST		VARCHAR2(12)
AMLAST		VARCHAR2(12)
AMEDATE		DATE
REGION		CHAR(2)

**FIGURE 3-6** The DESCRIBE command

When you issue the **DESCRIBE** command, all columns defined for the ACCTMANAGER table are listed. For each column name, you can also check whether the column allows NULL values and the column's datatype. Notice that the results do not display a "NOT NULL" requirement for the `AmId` column—it is blank. Because this column will be the primary key for the table, it should not be allowed to contain any blank entries. (This problem is corrected in the next chapter.) Thus, if the four columns have the correct name, datatype, and width—and your `CREATE TABLE` command executed—you now have a table ready to accept account managers' data.

## TABLE CREATION THROUGH SUBQUERIES

In the previous section, a table was created "from scratch." However, sometimes you might need to create a table based on data contained in existing tables. For example, suppose that management wants each account manager to have a table containing her customers' names. In addition, the table should include the book titles ordered by each customer. A word of caution is in order, however. You generally would not want multiple copies of data within the same database—it could lead to data redundancy. However, in this case, account managers would have the flexibility of adding columns for making comments or changes that should not be reflected in the main database tables. Because the account managers' tables would not be relevant to the transaction-processing activities for JustLee Books (such as order processing), allowing account managers to have tables containing data that already exist in other tables would not be a cause for alarm.

A **subquery** is a SELECT statement that is used within another SQL command. Any type of action that you can perform with a SELECT statement (such as filtering rows, filtering columns, and calculating aggregate amounts) can be performed when creating a table with a subquery. At this point, this text has covered only basic queries, so the example is limited to the SELECT statement features that are already familiar. After you have become familiar with all the features associated with the SELECT statement, you will understand the expanded explanation of this topic in Chapter 12.

The JustLee Books Marketing Department needs to analyze customer data, and management does not want the analysis queries to slow down the production server on which orders are entered. A copy of the customer table is required. The name and street address columns are not needed for the analysis. The Marketing Department requested that the table be named CUST\_MKT.

## CREATE TABLE...AS

To create a table that will contain data from existing tables, you can use the CREATE TABLE command with an AS clause that contains a subquery. The syntax is shown in Figure 3-7.

```
CREATE TABLE tablename [(columnname, ...)]  
AS (subquery);
```

**FIGURE 3-7** CREATE TABLE...AS command syntax

The command syntax shown in Figure 3-7 uses the CREATE TABLE keywords to instruct Oracle 10g to create a table. The name of the new table is then provided.

If you need to give the columns in the new table names that are different from the column names in the existing table, the new column names must be listed after the table name, inside parentheses. However, if you do not want to change any of the column names, the column list in the CREATE TABLE clause is omitted. If you do provide a column list in the CREATE TABLE clause, it must contain a name for *every* column to be returned by the query—including those names that will remain the same. In other words, if five columns are going to be returned from the subquery, five columns must be listed in the CREATE TABLE clause, or Oracle 10g returns an error message, and the statement fails. In addition, the column list must be in the *same order* as the columns listed in the SELECT clause of the subquery, so Oracle 10g knows which column from the subquery is assigned to which column in the new table.

The AS keyword instructs Oracle 10g that the columns in the new table will be based on the columns returned by the subquery. The AS keyword must precede the subquery. The columns in the new table will be created based on the same datatype and width the columns had in the existing table(s). To distinguish the subquery from the rest of the CREATE TABLE command, *the subquery must be enclosed within parentheses*.

The creation of the CUST\_MKT table based on a subquery is shown in Figure 3-8. To verify the results in regard to table structure and contents, execute the DESCRIBE and SELECT commands, as shown in Figures 3-9 and 3-10, respectively.

```
CREATE TABLE cust_mkt
AS (SELECT customer#, city, state, zip, referred
    FROM customers);
```

Execute Load Script Save Script Cancel

Table created.

**FIGURE 3-8** Creating a table based on a subquery

```
DESC cust_mkt
```

Execute Load Script Save Script Cancel

Name	Null?	Type
CUSTOMER#		NUMBER(4)
CITY		VARCHAR2(12)
STATE		VARCHAR2(2)
ZIP		VARCHAR2(5)
REFERRED		NUMBER(4)

**FIGURE 3-9** Using DESCRIBE to verify new table structure

SELECT *				
FROM cust_mkt;				
Execute Load Script Save Script Cancel				
CUSTOMER#	CITY	STATE	ZIP	REFERRED
1001	EASTPOINT	FL	32328	
1002	SANTA MONICA	CA	90404	
1003	TALLAHASSEE	FL	32306	
1004	BOISE	ID	83707	
1005	SEATTLE	WA	98115	
1006	ALBANY	NY	12211	
1007	AUSTIN	TX	78710	1003
1008	CHEYENNE	WY	82003	
1009	BURBANK	CA	91510	1003
1010	ATLANTA	GA	30314	
1011	CHICAGO	IL	60606	
1012	BOSTON	MA	02110	
1013	CLERMONT	FL	34711	1006
1014	CODY	WY	82414	
1015	MIAMI	FL	33111	
1016	BURBANK	CA	91508	1010
1017	KALMAZOO	MI	49006	
1018	MACON	GA	31206	
1019	MORRISTOWN	NJ	07962	1003
1020	TRENTON	NJ	08607	

20 rows selected.

**FIGURE 3-10** Using SELECT to verify new table contents

## MODIFYING EXISTING TABLES

There might be times when you need to make structural changes to a table. For example, you might need to add a column, delete a column, or simply change the size of a column. Each of these changes is accomplished through the **ALTER TABLE** command. A useful feature of Oracle 10g is that a table can be modified without having to shut down the database. Even if users are accessing a table, it can still be modified without disruption of service. The syntax for the **ALTER TABLE** command is shown in Figure 3-11.

```
ALTER TABLE tablename
ADD|MODIFY|DROP COLUMN| columnname [definition];
```

**FIGURE 3-11** Syntax for the ALTER TABLE command

Whether you should use an **ADD**, **MODIFY**, or **DROP COLUMN** clause depends on the type of change being made. First, let's look at the **ADD** clause.

## ALTER TABLE...ADD Command

Using an **ADD** clause with the **ALTER TABLE** command allows a user to add a new column to a table. The same rules that apply to creating a column in a new table apply to adding a column to a table that already exists. The new column must be defined by a column name and datatype (and width, if applicable). A default value can also be assigned. The difference is that the new column will be added at the end of the existing table—it will be the last column. The syntax for the **ALTER TABLE** command with the **ADD** clause is shown in Figure 3-12.

```
ALTER TABLE tablename
ADD (columnname datatype, [DEFAULT] ...);
```

**FIGURE 3-12** Syntax for the ALTER TABLE...ADD command

Suppose that after the **PUBLISHER** table was created, management requests that the table also contain a column for a telephone number extension. The column name should be **Ext**. The column can consist of a maximum of four numeric digits so the column is defined as a **NUMBER** datatype with a precision of four. To make this change to the **PUBLISHER** table, issue the command shown in Figure 3-13.

The screenshot shows a text input field containing the SQL command: `ALTER TABLE publisher ADD (ext NUMBER(4));`. Below the input field are four buttons: **Execute**, **Load Script**, **Save Script**, and **Cancel**. Below the buttons, the text "Table altered." is displayed. Below this text is a table showing the structure of the **PUBLISHER** table after the command was executed.

Name	Null?	Type
PUBID	NOT NULL	NUMBER(2)
NAME		VARCHAR2(23)
CONTACT		VARCHAR2(15)
PHONE		VARCHAR2(12)
EXT		NUMBER(4)

**FIGURE 3-13** The ALTER TABLE...ADD command

Oracle 10g returns the message "Table altered" to indicate that the command was completed successfully. To double-check that the column was added with the correct datatype, issue the **DESCRIBE** command, as shown in Figure 3-13.

### NOTE

When you add a column to a table that contains rows of data, the new column is empty for all the existing rows. Perform a **SELECT** command on the **PUBLISHER** table to confirm that the new **Ext** column is empty.

If you need to add more than one column to the ACCTMANAGER table, list the additional column(s) in a column list, and separate each column and its datatype from the other columns with a comma, using the same format as the CREATE TABLE command.

## ALTER TABLE...MODIFY Command

A **MODIFY** clause can be used with the ALTER TABLE command to change the definition of an existing column. The changes that can be made to a column include the following:

- Changing the size of a column (increase or decrease)
- Changing the datatype (such as VARCHAR2 to CHAR)
- Changing or adding the default value of a column (such as DEFAULT SYS-DATE)

The syntax for the ALTER TABLE...MODIFY command is shown in Figure 3-14.

```
ALTER TABLE tablename
MODIFY (columnname datatype [DEFAULT],...);
```

**FIGURE 3-14** Syntax of the ALTER TABLE...MODIFY command

You should be aware of three rules when modifying existing columns.

- A column must be as wide as the data fields it already contains.
- If a NUMBER column already contains data, you can't decrease the precision or scale of the column.
- Changing the default value of a column does not change the values of data already in the table.

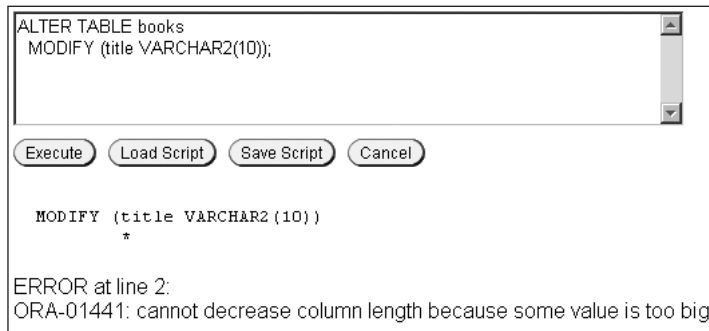
Next, let's look at each of these rules.

The first rule applies when you want to decrease the size of a column that already contains data. You can only decrease the size of a column to a size that is no less than the largest width of the existing data. For example, suppose that a column has been defined as a VARCHAR2 datatype with a width of 15 characters. However, the largest entry in that particular column contains only 12 characters. Thus, you can decrease the size of the column only to a width of 12 characters. If you try to decrease the size to a width less than 12, Oracle 10g returns an error message, and the SQL statement fails. As shown in Figure 3-15, when a user attempts to decrease the width of the column to a size that will not accommodate the data it already contains, Oracle 10g returns an ORA-01441 error message, and the table is not altered.

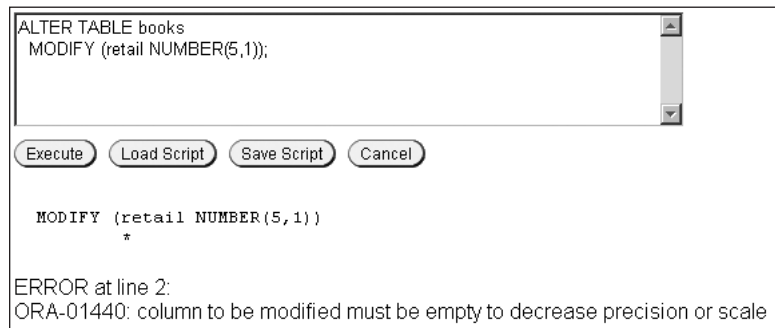
Second, Oracle 10g does not allow you to decrease the precision or scale of a NUMBER column if the column contains any data. Regardless of whether the current values stored in a NUMBER column will be affected, Oracle 10g returns an ORA-01440 error message, and the statement fails unless the column is empty. As shown in Figure 3-16, if you attempt to change the size of the Retail column in the BOOKS table, an error message is displayed, and the table is not altered.

The third rule applies when you modify existing columns and decide to change the default value assigned to a column. When the default value of a column is changed, it changes only the value assigned to *future rows* inserted into the table. The default value





**FIGURE 3-15** Modify column size error



**FIGURE 3-16** Modify numeric column error

assigned to rows that already exist in the table remain the same. Thus, if the default value contained in existing rows must be changed, those changes must be performed manually. (Chapter 5 discusses how to change existing values in a row.) Let's assume JustLee Books has decided to assign a service rating code to every publisher that reflects the publisher's service promptness. The code should initially be set to 'N' when a new publisher is added. To demonstrate the effect of adding a DEFAULT option to an existing column, the new column is added first and then it is modified to set the default value. Figure 3-17 displays the ALTER TABLE command to accomplish this column addition task. Notice that the SELECT statement shows that the new Rating column is empty for the existing rows. Second, the DEFAULT option is added on the new column, as shown in Figure 3-18. Notice that the SELECT statement shows that the new Rating column is still empty for the existing rows—the DEFAULT option takes effect only on new rows added.

Now return your attention to the ACCTMANAGER table created earlier in this chapter. A common issue with tables is the need to widen columns to accommodate data values of greater length. After creating the ACCTMANAGER table, suppose that you find out that one of the account managers has a long name that requires more than the 12 spaces you assigned to the Amlast column. To accommodate the name, the Amlast column must be increased to a width of 18 characters. What command should you use to achieve this?

```
ALTER TABLE publisher
ADD (rating CHAR(1));

SELECT *
FROM publisher;
```

Execute Load Script Save Script Cancel

Table altered.

PUBID	NAME	CONTACT	PHONE	EXT	RAT
1	PRINTING IS US	TOMMIE SEYMOUR	000-714-8321		
2	PUBLISH OUR WAY	JANE TOMLIN	010-410-0010		
3	AMERICAN PUBLISHING	DAVID DAVIDSON	800-555-1211		
4	READING MATERIALS INC.	RENEE SMITH	800-555-9743		
5	REED-N-RITE	SEBASTIAN JONES	800-555-8284		

**FIGURE 3-17** Adding the Rating column to the PUBLISHER table

## NOTE

Column headings may be truncated based on available linespace. Your column headers may vary from those shown in the figures in this textbook. Formatting column widths in output will be covered later in this book.

```
ALTER TABLE publisher
MODIFY (rating DEFAULT 'N');

SELECT *
FROM publisher;
```

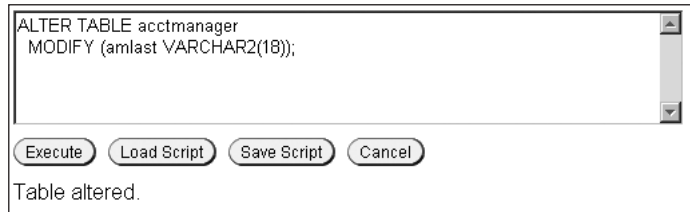
Execute Load Script Save Script Cancel

Table altered.

PUBID	NAME	CONTACT	PHONE	EXT	RAT
1	PRINTING IS US	TOMMIE SEYMOUR	000-714-8321		
2	PUBLISH OUR WAY	JANE TOMLIN	010-410-0010		
3	AMERICAN PUBLISHING	DAVID DAVIDSON	800-555-1211		
4	READING MATERIALS INC.	RENEE SMITH	800-555-9743		
5	REED-N-RITE	SEBASTIAN JONES	800-555-8284		

**FIGURE 3-18** Adding the DEFAULT option to the new column

Figure 3-19 displays the ALTER TABLE command to widen the Amlast column. Notice that the MODIFY clause of the command does not explicitly state that the Amlast column should be increased by six characters. Instead, the datatype and new width are stated with the width increased to the total desired width for the column. After the command is processed, Oracle 10g returns the message “Table altered,” as shown in Figure 3-19. To make certain that change is made, you might also use the DESCRIBE command to view the new table structure.



**FIGURE 3-19** The ALTER TABLE...MODIFY command used to increase the width of the Amlast column

## ALTER TABLE...DROP COLUMN Command

The **DROP COLUMN** clause can be used with the ALTER TABLE command to delete an existing column from a table. The clause deletes both the column and its contents, so it should be used with extreme caution. The syntax for the ALTER TABLE...DROP COLUMN command is shown in Figure 3-20.

```
ALTER TABLE tablename
DROP COLUMN columnname;
```

**FIGURE 3-20** Syntax for the ALTER TABLE...DROP COLUMN command

You should keep the following in mind when using the DROP COLUMN clause:

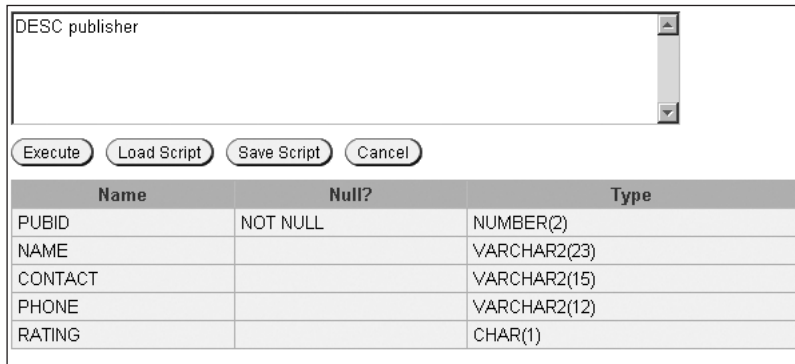
- Unlike using an ALTER TABLE command with the ADD or MODIFY clauses, a DROP COLUMN clause can reference only *one* column.
- If you drop a column from a table, the deletion is permanent. You cannot “undo” the damage if you accidentally delete the wrong column from a table. The only option is to add the column back to the table and then manually reenter all the data that it previously contained.
- You cannot delete the last remaining column in a table. If a table contains only one column, and you try to delete the column, the command fails, and Oracle 10g returns an error message.
- A primary key column cannot be dropped from a table.

Previously, you added the Ext column to store the telephone extension of each publisher. However, management has now decided that the extension is not necessary and does not want to waste the storage space that would be taken up by that column. Therefore, the Ext column needs to be deleted from the PUBLISHER table. The command to delete the Ext column is shown in Figure 3-21.

After the command is processed, the Ext column of the PUBLISHER table is removed. To verify that the column no longer exists, use the DESCRIBE command to list the structure of the PUBLISHER table, as shown in Figure 3-22.



**FIGURE 3-21** The ALTER TABLE...DROP COLUMN command



**FIGURE 3-22** Verifying the column is dropped

### ALTER TABLE...SET UNUSED/DROP UNUSED COLUMNS Command

When the Oracle 10g server drops a column from a very large table, this can slow down the processing of queries or other SQL commands from users. To avoid this problem, a **SET UNUSED** clause can be included in the ALTER TABLE command to mark the column for deletion at a later time. If a column is marked for deletion, it is unavailable and will not be displayed in the table structure. Because the column is unavailable, it will not appear in the results of any queries, nor can any other operation be performed on the column except the ALTER TABLE...DROP UNUSED command. In other words, after a column is set as “unused,” the column and all its contents are no longer available and cannot be recovered at a future time. It simply postpones the physical erasing of the data from the storage device until a later time—usually when the server is processing fewer queries, such as after business hours. A **DROP UNUSED** clause is used with the ALTER TABLE command to complete the deletion process for any column that has been marked as unused.

The syntax for the ALTER TABLE...SET UNUSED command is shown in Figure 3-23.

```
ALTER TABLE tablename
SET UNUSED (columnname);
OR
ALTER TABLE tablename
SET UNUSED COLUMN columnname;
```

**FIGURE 3-23** Syntax for the ALTER TABLE...SET UNUSED command

As shown in Figure 3-23, the syntax for the ALTER TABLE...SET UNUSED command has two options for the SET UNUSED clause. Regardless of the syntax used, only one column per command can be marked for deletion. The syntax to drop a column previously identified as unused is shown in Figure 3-24.

```
ALTER TABLE tablename
DROP UNUSED COLUMNS;
```

**FIGURE 3-24** Syntax for the ALTER TABLE...DROP UNUSED COLUMNS command

When the DROP UNUSED COLUMNS clause is used, any column that has previously been set as “unused” is deleted, and any storage previously occupied by data contained in the column(s) becomes available.

Suppose that the JustLee Books Marketing Department has decided that they do not need to see the customer state data in the CUST\_MKT table. To delete that column from the CUST\_MKT table, you could use the DROP COLUMN option of the ALTER TABLE command. However, suppose that the table contains thousands of records and deleting a column would slow down operations for other Oracle 10g users. In this case, you can mark the State column of the CUST\_MKT table as unused with the command shown in Figure 3-25.



**FIGURE 3-25** The ALTER TABLE...SET UNUSED command

To make certain that the State column was correctly marked for deletion, the DESCRIBE command can be used to check that the column is no longer available, as shown in Figure 3-26.

You can also reference the data dictionary to determine if any columns are marked as unused. The data dictionary object named USER\_UNUSED\_COL\_TABS contains information on which tables have unused columns and how many columns are set to unused. Figure 3-27 displays a query on this object.

After the column has been set as unused, the storage space taken up by data contained within the State column can be reclaimed, using the command shown in Figure 3-28.

## Renaming a Table

Oracle 10g allows you to change the name of any table you own, using the RENAME...TO command. The syntax for the RENAME...TO command is shown in Figure 3-29.

DESC cust\_mkt

Execute Load Script Save Script Cancel

Name	Null?	Type
CUSTOMER#		NUMBER(4)
CITY		VARCHAR2(12)
ZIP		VARCHAR2(5)
REFERRED		NUMBER(4)

**FIGURE 3-26** Verifying the column can no longer be used

SELECT \*  
FROM user\_unused\_col\_tabs;

Execute Load Script Save Script Cancel

TABLE_NAME	COUNT
CUST_MKT	1

**FIGURE 3-27** Listing tables with columns marked as unused

ALTER TABLE cust\_mkt  
DROP UNUSED COLUMNS;

Execute Load Script Save Script Cancel

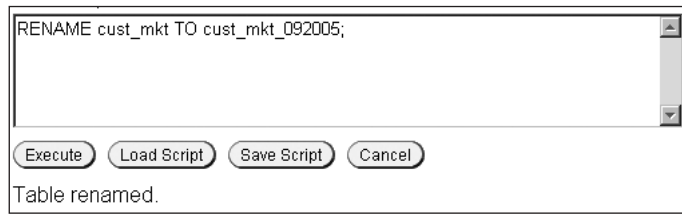
Table altered.

**FIGURE 3-28** The ALTER TABLE...DROP UNUSED COLUMNS command

```
RENAME oldtablename TO newtablename;
```

**FIGURE 3-29** Syntax for the RENAME...TO command

In a previous section, a table named CUST\_MKT was created. However, the Marketing Department wants to maintain a series of snapshots for customer data and, therefore, wants the table name to reflect the month and year the table was created. Assume the current CUST\_MKT table was created in September 2005. The command to make the name change is shown in Figure 3-30.



**FIGURE 3-30** The RENAME...TO command

After the RENAME...TO command is executed, any queries directed to the original table named CUST\_MKT result in an error message. The table can now only be referenced as the CUST\_MKT\_092005 table. The SELECT statements shown in Figure 3-31 demonstrate this point.

## TIP

When working in an organization, don't change the name of a table that is accessed by other users unless you first inform them of the new table name. Failure to inform users of the change will prevent them from finishing their work and will create havoc until the problem is identified. Of course, this is assuming that you didn't change the name of the table to stop someone from accessing it in the first place!

## Truncating a Table

When a table is truncated, all the rows contained in the table are removed, but the table itself remains. In other words, the columns still exist even though no values are stored in them. This is basically the same as deleting all the rows in a table. However, if you simply delete all rows in a table, the storage space occupied by those rows will still be allocated to the table. To delete the rows stored within a table *and* free up the storage space that was occupied by those rows, use the **TRUNCATE TABLE** command. The syntax for the command is shown in Figure 3-32.

Assume that the CUST\_MKT\_092005 table was originally created a day early and it did not include the customers added on the last day of September. The database administrator is going to repopulate the table; however, all the current rows need to be removed before the repopulation can occur. Keep in mind that the TRUNCATE command will keep the table structure intact, so that it can be reused in the future. TRUNCATE will delete the rows currently contained in the CUST\_MKT\_092005 table and release the storage space they occupy. The command to perform the truncation is shown in Figure 3-33.

To verify that all rows of the CUST\_MKT\_092005 table have been removed, perform a query to see all the rows in the table. If the table still exists but contains no rows, Oracle 10g displays the message “no rows selected.”

## NOTE

The TRUNCATE command is quite useful when creating and using database tables to support testing new applications. As you test an application, the tables typically are populated with sample data. The TRUNCATE command is an easy way to maintain the table structures while eliminating all the test data either to start another round of testing or to move the tables into production.

SELECT \*  
FROM cust\_mkt;

SELECT \*  
FROM cust\_mkt\_092005;

Execute Load Script Save Script Cancel

FROM cust\_mkt  
\*

ERROR at line 2:  
ORA-00942: table or view does not exist

CUSTOMER#	CITY	ZIP	REFERRED
1001	EASTPOINT	32328	
1002	SANTA MONICA	90404	
1003	TALLAHASSEE	32306	
1004	BOISE	83707	
1005	SEATTLE	98115	
1006	ALBANY	12211	
1007	AUSTIN	78710	1003
1008	CHEYENNE	82003	
1009	BURBANK	91510	1003
1010	ATLANTA	30314	
1011	CHICAGO	60606	
1012	BOSTON	02110	
1013	CLERMONT	34711	1006
1014	CODY	82414	
1015	MIAMI	33111	
1016	BURBANK	91508	1010
1017	KALMAZOO	49006	
1018	MACON	31206	
1019	MORRISTOWN	07962	1003
1020	TRENTON	08607	

20 rows selected.

**FIGURE 3-31** Verifying the RENAME operation

TRUNCATE TABLE *tablename*;

**FIGURE 3-32** Syntax for the TRUNCATE TABLE command

TRUNCATE TABLE cust\_mkt\_092005;

Execute Load Script Save Script Cancel

Table truncated.

**FIGURE 3-33** The TRUNCATE TABLE command



## DELETING A TABLE

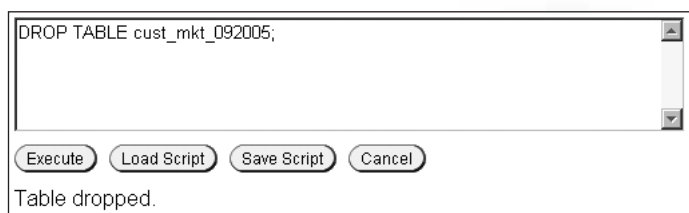
A table can be removed from an Oracle 10g database by issuing the **DROP TABLE** command. The syntax for the DROP TABLE command is shown in Figure 3-34.

77

```
DROP TABLE tablename [PURGE];
```

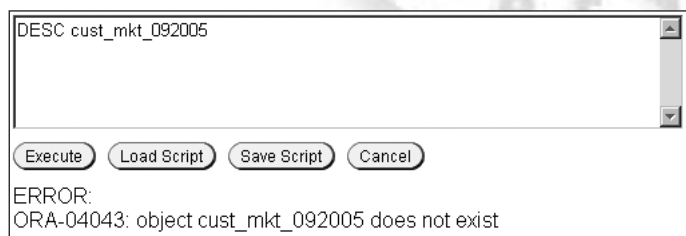
**FIGURE 3-34** Syntax for the DROP TABLE command

Suppose that after truncating the CUST\_MKT\_092005 table, you realize that you will no longer need the table (or that numerous modifications will have to be made to the table structure so that it is not worth the trouble to make the changes). The CUST\_MKT\_092005 table can be deleted using the DROP TABLE command, shown in Figure 3-35.



**FIGURE 3-35** Using the DROP TABLE command to remove the CUST\_MKT\_092005 table

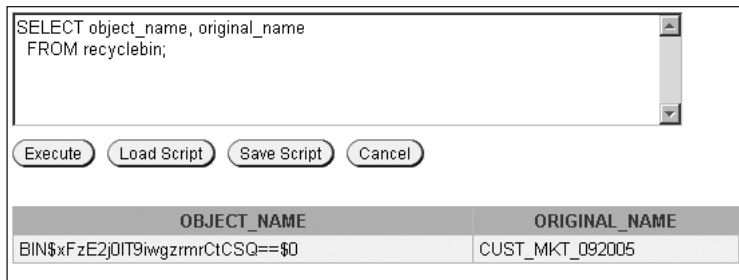
After the table has been dropped, the table name will no longer be valid, and the table cannot be accessed by any commands. To verify that the correct table was deleted, you can use the DESCRIBE command to see the structure of the CUST\_MKT\_092005 table. If the table no longer exists, Oracle 10g returns an error message, as shown in Figure 3-36.



**FIGURE 3-36** Using DESCRIBE to verify dropped table

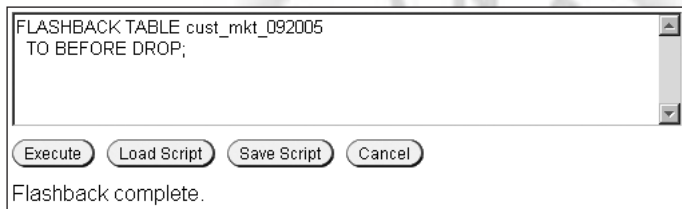
Oracle 10g introduced a new feature related to the DROP TABLE command. In previous Oracle versions, the DROP TABLE command was permanent, and the only method to recover a dropped table was to restore the data from backups. In Oracle 10g, a dropped table is placed into a recycle bin and can be restored—both table structure and data! Now that the CUST\_MKT\_092005 table has been dropped, you should check the recycle bin with the command shown in Figure 3-37. Notice a new name beginning with BIN has been assigned to the table. This new name is assigned by the system, so results may vary.

Table Creation and Management



**FIGURE 3-37** Checking the recycle bin

Now that you have a table in the recycle bin, how do you recover the table to use it again? You need to use the FLASHBACK command, as shown in Figure 3-38. Keep in mind that the entire table structure and all the data that was in the table at the time of the drop will be restored. After performing the FLASHBACK command, execute a DESCRIBE and SELECT on the table to verify that it is restored.



**FIGURE 3-38** Using FLASHBACK to restore a dropped table

Obviously, if the dropped table is actually just moved to a recycle bin, the storage space required for the table is still being used. If you have limited storage space, this might not be desirable. And, in other cases, you know that you want to permanently delete a table. If the table has already been dropped and is now in the recycle bin, you need to remove the table from the recycle bin to permanently delete the table and clear the storage space being used. Drop the CUST\_MKT\_092005 table again and verify its existence in the recycle bin with the commands shown in Figure 3-39.

Now you can remove the table from the recycle bin using the PURGE TABLE command and referencing the name the table has been assigned in the recycle bin. Figure 3-40 demonstrates removing the table from the recycle bin.

## NOTE

Keep in mind that the system assigns the table name in the recycle bin, so you will most likely have a different table name than what is shown in Figure 3-40. Also, if you want to remove all the tables in the recycle bin, you can do so by issuing a PURGE RECYCLE BIN command.

```
DROP TABLE cust_mkt_092005;
```

```
SELECT object_name, original_name
FROM recyclebin;
```

Execute Load Script Save Script Cancel

Table dropped.

OBJECT_NAME	ORIGINAL_NAME
BIN\$IDMdpyuMT4OR6PszkWX4gg==\$0	CUST_MKT_092005

**FIGURE 3-39** Dropping a table and check the recycle bin

```
PURGE TABLE "BIN$IDMdpyuMT4OR6PszkWX4gg==$0";
```

Execute Load Script Save Script Cancel

Table purged.

**FIGURE 3-40** Removing a table from the recycle bin

If you are sure you want to permanently delete a table, you can bypass moving the table to the recycle bin by using the PURGE option in the DROP TABLE statement. For example, you could have initially dropped the CUST\_MKT\_092005 table using the command shown in Figure 3-41 and then it would not have been moved to the recycle bin. This means the table cannot be recovered.

```
DROP TABLE cust_mkt_092005 PURGE;
```

**FIGURE 3-41** Dropping a table with the PURGE option

## CAUTION

Always use caution when deleting with the PURGE option. After a table is deleted with PURGE, the table and all the data it contains are gone. In addition, any index that has been created based on that table is also dropped. (Indexes are presented in Chapter 6.)

## Chapter Summary

---

- You create a table with the `CREATE TABLE` command. Each column to be contained in the table must be defined in terms of the column name, datatype, and for certain datatypes, the width.
- A table can contain up to 1000 columns.
- Each column name within a table must be unique.
- Table and column names can contain as many as 30 characters. The names must begin with a letter and cannot contain any blank spaces.
- To create a table based on the structure and data contained in existing tables, use the `CREATE TABLE...AS` command to use a subquery to extract the necessary data from the existing table.
- You can change the structure of a table with the `ALTER TABLE` command. Columns can be added, resized, and even deleted with the `ALTER TABLE` command.
- When using the `ALTER TABLE` command with the `DROP COLUMN` clause, only one column can be specified for deletion.
- You can use the `SET UNUSED` clause to mark a column so its storage space can be freed up at a later time.
- Tables can be renamed with the `RENAME...TO` command.
- To delete all the rows in a table, use the `TRUNCATE TABLE` command.
- To remove both the structure of a table and all its contents, use the `DROP TABLE` command.
- A dropped table is moved to the recycle bin and can be recovered using the `FLASHBACK TABLE` command.
- Using the `PURGE` option in a `DROP TABLE` command permanently removes the table, meaning you cannot recover it from the recycle bin.

## Chapter 3 Syntax Summary

81

The following table presents a summary of the syntax that you have learned in this chapter. You can use the table as a study guide and reference.

SYNTAX GUIDE		
Creating Tables		
Commands and Clauses	Description	Example
CREATE TABLE	Creates a new table in the database-user names columns; defaults and datatypes define/ limit columns. To view the table, use the SQL*PLUS command DESCRIBE.	<pre>CREATE TABLE acctmanager (amid    VARCHAR2(4), amname  VARCHAR2(20), amedate DATE DEFAULT SYSDATE, region  CHAR(2));</pre>
CREATE TABLE ... AS (...)	Creates a table from existing database tables, using the AS clause and subqueries.	<pre>CREATE TABLE secustomerorder AS (SELECT customer#, state, ISBN, category, quantity, cost, retail FROM customers NATURAL JOIN orders NATURAL JOIN orderitems NATURAL JOIN books WHERE state IN ('FL', 'GA', 'AL'));</pre>
Modifying Tables		
ALTER TABLE... ADD	Adds a column to a table.	<pre>ALTER TABLE acctmanager ADD (ext NUMBER(4));</pre>
ALTER TABLE... MODIFY	Changes a column size, datatype, or default value.	<pre>ALTER TABLE acctmanager MODIFY (amname VARCHAR2(25));</pre>
ALTER TABLE... DROP COLUMN	Deletes one column from a table.	<pre>ALTER TABLE acctmanager DROP COLUMN ext;</pre>
ALTER TABLE... SET UNUSED or SET UNUSED COLUMN	Marks a column for deletion at a later time.	<pre>ALTER TABLE cust_mkt SET UNUSED (state);</pre>
DROP UNUSED COLUMNS	Completes the deletion of a column marked with SET UNUSED.	<pre>ALTER TABLE cust_mkt DROP UNUSED COLUMNS;</pre>
RENAME...TO	Changes a table name.	<pre>RENAME cust_mkt TO cust_mkt_ 092005;</pre>

## SYNTAX GUIDE

TRUNCATE TABLE	Deletes table rows, but table name and column structure remain.	TRUNCATE TABLE cust_mkt_092005;
Deleting Tables		
DROP TABLE . . . PURGE	Removes an entire table permanently from the Oracle 10g database with the PURGE option.	DROP TABLE cust_mkt_092005 PURGE;
PURGE TABLE	Permanently deletes a table in the recyclebin.	PURGE TABLE "BIN\$IDMdosJceWxgg041==\$0";
Recovering Tables		
FLASHBACK TABLE . . . TO BEFORE DROP	Recovers a dropped table if option not used when table dropped.	PURGE FLASHBACK TABLE cust_mkt_ 092005 TO BEFORE DROP;

## Review Questions

*To answer the following questions, refer to the tables in Appendix A.*

1. Which command is used to create a table based upon data already contained in an existing table?
2. List four datatypes supported by Oracle 10g, and provide an example of data that could be stored by each datatype.
3. What are the guidelines you should follow when naming tables and columns in Oracle 10g?
4. What is the difference between dropping a column and setting a column as unused?
5. How many columns can be dropped in one ALTER TABLE command?
6. What happens to the existing rows of a table if the DEFAULT value of a column is changed?
7. Explain the difference between truncating a table and deleting a table.
8. If you add a new column to an existing table, where will the column appear relative to the existing columns?
9. What happens if you try to decrease the scale or precision of a NUMBER column to a value less than the data already stored in the field?
10. Is a table and the data contained in the table permanently erased from the system if a DROP TABLE command is issued on the table?

## Multiple Choice

To answer the following questions, refer to the tables in Appendix A.

1. Which of the following is a correct statement?
  - a. You can restore the data deleted with the DROP COLUMN clause, but not the data deleted with the SET UNUSED clause.
  - b. You cannot create empty tables—all tables must contain at least three rows of data.
  - c. A table can contain a maximum of 1000 columns.
  - d. The maximum length of a table name is 265 characters.
2. Which of the following is a valid SQL statement?
  - a. ALTER TABLE secustomersspent ADD DATE lastorder;
  - b. ALTER TABLE secustomerorders DROP retail;
  - c. CREATE TABLE newtable AS (SELECT \* FROM customers);
  - d. ALTER TABLE drop column \*;
3. Which of the following is not a correct statement?
  - a. A table can be modified only if it does not contain any rows of data.
  - b. The maximum number of characters in a table name is 30.
  - c. You can add more than one column at a time to a table.
  - d. You cannot recover data contained in a table that has been truncated.
4. Which of the following is not a valid SQL statement?
  - a. CREATE TABLE anothernewtable (newtableid VARCHAR2(2));
  - b. CREATE TABLE anothernewtable (date, anotherdate) AS (SELECT orderdate, shipdate FROM orders);
  - c. CREATE TABLE anothernewtable (firstdate, seconddate) AS (SELECT orderdate, shipdate FROM orders);
  - d. All of the above are valid statements.
5. Which of the following is true?
  - a. If you truncate a table, you cannot add new data to the table.
  - b. If you change the default value of an existing table, any previously stored default value will be changed to a null value.
  - c. If you delete a column from a table, you cannot add a column to the table using the same name as the previously deleted column.
  - d. If you add a column to an existing table, it will always be added as the last column of that table.

6. Which of the following will create a new table containing the order number, book title, quantity ordered, and retail price of every book that has been sold?
  - a. `CREATE TABLE newtable AS(SELECT order#, title, quantity, retail FROM orders);`
  - b. `CREATE TABLE newtable AS(SELECT * FROM orders);`
  - c. `CREATE TABLE newtable AS(SELECT order#, title, quantity, retail FROM orders NATURAL JOIN orderitems);`
  - d. `CREATE TABLE newtable AS(SELECT order#, title, quantity, retail FROM orders NATURAL JOIN orderitems NATURAL JOIN books);`
7. Which of the following commands will drop any columns marked as unused from the SECUSTOMERORDERS table?
  - a. `DROP COLUMN FROM SECUSTOMERORDERS WHERE column_status = UNUSED;`
  - b. `ALTER TABLE SECUSTOMERORDERS DROP UNUSED COLUMNS;`
  - c. `ALTER TABLE SECUSTOMERORDERS DROP (unused);`
  - d. `DROP UNUSED COLUMNS;`
8. Which of the following statements is correct?
  - a. A table can only contain a maximum of one column that is marked as unused.
  - b. You can delete a table by removing all the columns within the table.
  - c. Using the SET UNUSED clause allows you to free up all the storage space used by a column.
  - d. None of the above statements are correct.
9. Which of the following commands will remove all the data from a table but leave the table's structure intact?
  - a. `ALTER TABLE secustomerorders DROP UNUSED COLUMNS;`
  - b. `TRUNCATE TABLE secustomerorders;`
  - c. `DELETE TABLE secustomerorders;`
  - d. `DROP TABLE secustomerorders;`
10. Which of the following commands would change the name of a table from OLDNAME to NEWNAME?
  - a. `RENAME oldname TO newname;`
  - b. `RENAME table FROM oldname TO newname;`
  - c. `ALTER TABLE oldname MODIFY TO newname;`
  - d. `CREATE TABLE newname (SELECT * FROM oldname);`
11. The default width of a VARCHAR2 field is:
  - a. 1
  - b. 30
  - c. 255
  - d. None—there is no default width for a VARCHAR2 field.



12. Which of the following is NOT a valid statement?
  - a. You can change the name of a table only if it does not contain any data.
  - b. You can change the length of a column that does not contain any data.
  - c. You can delete a column that does not contain any data.
  - d. You can add a column to a table.
13. Which of the following can be used in a table name?
  - a. \_
  - b. (
  - c. %
  - d. !
14. Which of the following is true?
  - a. When you execute an ALTER TABLE command, other users cannot use the database.
  - b. You can add a column to a table as long as the table is not being queried by another user.
  - c. All data contained in a table will be lost if the table is dropped.
  - d. All of the above statements are true.
15. Which of the following commands is valid?
  - a. RENAME customer# TO customernumber FROM customers;
  - b. ALTER TABLE customers RENAME customer# TO customernum;
  - c. DELETE TABLE customers;
  - d. ALTER TABLE customers DROP UNUSED COLUMNS;
16. Which of the following commands will create a new table containing two columns?
  - a. CREATE TABLE newname (col1 DATE, col2 VARCHAR2);
  - b. CREATE TABLE newname AS(SELECT title, retail, cost FROM books);
  - c. CREATE TABLE newname (col1, col2);
  - d. CREATE TABLE newname (col1 DATE DEFAULT SYSDATE, col2 VARCHAR2(1));
17. Which of the following is a valid table name?
  - a. 9NEWTABLE
  - b. DATE9
  - c. NEW"TABLE
  - d. None of the above are valid table names.
18. Which of the following is a valid datatype?
  - a. CHAR3
  - b. VARCHAR4(3)
  - c. NUMERIC
  - d. NUMBER

19. Which of the following will create a table listing books that have a retail price of at least \$30.00?
  - a. `SELECT * FROM books WHERE retail >= 30;`
  - b. `CREATE newtable AS (SELECT * FROM books WHERE retail >= 30);`
  - c. `CREATE newtable AS (SELECT * FROM books WHERE retail > 30);`
  - d. `CREATE newtable AS (SELECT * FROM books) WHERE retail >= 30;`
  - e. none of the above
20. Which of the following SQL statements will change the size of the Title column in the BOOKS table from the current length of 30 characters to the needed length of 35 characters?
  - a. `ALTER TABLE books CHANGE title VARCHAR(35);`
  - b. `ALTER TABLE books MODIFY (title VARCHAR2(35));`
  - c. `ALTER TABLE books MODIFY title (VARCHAR2(35));`
  - d. `ALTER TABLE books MODIFY (title VARCHAR2(+5));`

## Hands-On Assignments

---

*To perform the following activities, refer to the tables in Appendix A.*

1. Create a new table that contains the category code and description for the categories of books sold by JustLee Books. The table should be called Category. The columns should be called CatCode and CatDesc. The CatCode column should store a maximum of two characters, and the CatDesc column should store a maximum of 10 characters.
2. Create a new table that contains the following four columns: Emp#, Lastname, Firstname, Job\_class. The name of the table should be EMPLOYEES. The Job\_class column should be able to store character strings up to a maximum length of four. The column values should not be padded if the value has less than four characters. The Emp# column will contain a numeric ID and should allow a five-digit number.
3. Add two columns to the EMPLOYEES table. One will contain the date of employment for each employee. The default value of the column should be the system date. The new column should be named EmpDate. The second column will be used to contain the date of termination and should be named EndDate.
4. Modify the Job\_class column of the EMPLOYEES table so that it will allow a maximum width of two characters to be stored in the column.
5. Delete the EndDate column from the EMPLOYEES table.
6. Rename the EMPLOYEES table to JL\_EMPS.
7. Create a new table that contains the following four columns from the existing BOOKS table: ISBN, Cost, Retail, and Category. The name of the ISBN column should be ID, and the other columns should keep their original names. Name the new table BOOK\_PRICING.
8. Mark the Category column of the BOOK\_PRICING table as unused. Verify that the column is no longer available.

9. Truncate the BOOK\_PRICING table and then verify that the table still exists but no longer contains any data.
10. Delete the BOOK\_PRICING table permanently so it is not moved to the recycle bin. Delete the JL\_EMPS table so it can be restored. Restore the JL\_EMPS table and verify that it is now available again.

## Advanced Challenge

To perform this activity, refer to the tables in Appendix A.

1. The management of JustLee Books has approved the implementation of the new commission policy for the account managers. The following changes need to be made to the existing database:
  - A new column must be added to the CUSTOMERS table to indicate the region in which each customer lives. The column should be able to store variable-length data to a maximum length of two characters. It should be named Region.
  - A new table, COMMRATE, must be created to store the commission rate schedule. The following columns must exist in the table:
    1. Rate: a numeric field that can store two decimal digits (such as .01 or .03)
    2. Minprice: a numeric field that can store the lowest retail price for a book in that price range of the commission rate
    3. Maxprice: a numeric field that can store the highest retail price for a book in that price range of the commission rate

**Required:** Make the necessary changes to the JustLee Books database to support the implementation of the new commission policy.

## Case Study: City Jail

In the Chapter 1 case study, you designed the new database for City Jail. Now you need to create all the tables needed for the database. First, create all the tables using the information outlined below Section A that follows. Second, make the modifications outlined in Section B. Save all SQL statements used to accomplish these tasks.

### Section A

TABLE	COLUMN	DATA DESCRIPTION	LENGTH	SCALE	DEFAULT VALUE
Aliases	Criminal_id	Numeric	6	0	
	Alias	Variable Character	10		
Criminals	Criminal_id	Numeric	6	0	
	Last	Variable Character	15		
	First	Variable Character	10		
	Street	Variable Character	30		
	City	Variable Character	20		

TABLE	COLUMN	DATA DESCRIPTION	LENGTH	SCALE	DEFAULT VALUE
Crimes	State	Fixed Character	2		
	Zip	Numeric	5	0	
	Phone	Numeric	10	0	
	V_status	Fixed Character	1		N (for No)
	P_status	Fixed Character	1		N (for No)
	Crime_id	Numeric	9	0	
	Criminal_id	Numeric	6	0	
	Classification	Fixed Character	1		
	Data_charged	Date			
	Status	Fixed Character	2		
Probations	Hearing_date	Date			
	Appeal_cut_date	Date			
	Probation_id	Numeric	6		
	Criminal_id	Numeric	6		
	Prob_id	Numeric	5		
	Start_date	Date			
	End_date	Date			
	Violations	Numeric	3		
	Prob_id	Numeric	5		
	Last	Variable Character	15		
Prob_officers	First	Variable Character	10		
	Street	Variable Character	30		
	City	Variable Character	20		
	State	Fixed Character	2		
	Zip	Numeric	5	0	
	Phone	Numeric	10	0	
	Email	Variable Character	30		
	Status	Fixed Character	1		A (for Active)
	Charge_id	Numeric	10	0	
	Crime_id	Numeric	9	0	

TABLE	COLUMN	DATA DESCRIPTION	LENGTH	SCALE	DEFAULT VALUE
	Crime_code	Numeric	3	0	
	Charge_status	Fixed Character	2		
	Fine_amount	Numeric	7	2	
	Court_fee	Numeric	7	2	
	Amount_paid	Numeric	7	2	
	Pay_due_date	Date			
Crime_officers	Crime_id	Numeric	9	0	
	Officer_id	Numeric	8	0	
Officers	Officer_id	Numeric	8	0	
	Last	Variable Character	15		
	First	Variable Character	10		
	Precinct	Fixed Character	4		
	Badge	Variable Character	14		
	Phone	Numeric	10	0	
	Status	Fixed Character	1		A (for Active)
Appeals	Crime_id	Numeric	9	0	
	Filing_date	Date			
	Hearing_date	Date			
	status	Fixed Character	1		P (for Pending)
Crime_codes	Crime_code	Numeric	3	0	
	Code_description	Variable Character	30		

## Section B

- Add a default value of 'U' for the Classification column of the CRIMES table.
- Add a column named Date\_Recorded to the CRIMES table. This column will need to hold date values and should initially be set to the current date by default.
- Add a column to the PROB\_OFFICERS table to contain the pager number for each officer. The column needs to accommodate a phone number including area code. Name the column Pager#.
- Change the Alias column in the ALIASES table to accommodate up to 20 characters.

