Create Table Statements

Introduction

By now you should be pretty familiar with the SELECT statement in SQL. We have covered the basic SELECT statement, restricting rows with the WHERE clause, sorting rows using the ORDER BY clause, joining tables using various methods, single-row functions, group functions and grouping data using the GROUP BY clause, and subqueries.

We will now turn our attention to creating tables and using statements that modify the structure of a table. As an application programmer, you most likely will not be involved in these tasks as they are typically performed by a database administrator. However, being familiar with them will give you a basic background in database administration, and perhaps at some point in your career you may decide to follow this path.

Table Design

Before creating a table, the information-level (logical) design should be viewed in detail. This design will serve as the template for creating the table. Generally, a design document called a data dictionary contains the specifics that can be used on the CREATE TABLE statement, such as table and column names, column data type and size, column constraints, primary key, and foreign key references. Constraints and key definitions will be covered in the next unit. Refer to your textbook for table and column naming rules and conventions specific to Oracle.

Considerable thought should be given to the data type and size of each column in the table. Once a table has been created and loaded with data, it can be quite cumbersome to change a column's data type or size. Some of the more frequently used data types are CHAR, VARCHAR, NUMBER and DATE. Each has a specific use and should be matched accordingly to a column's intended purpose. Refer to the Oracle SQL Reference Manual for information on all the data types available.

The basic form of the CREATE TABLE statement is:

```
CREATE TABLE tablename
(columnname datatype,
columnname datatype);
```

To create a table for students, we might code:

```
CREATE TABLE Student
(StudentNum NUMBER(4),
StudentName CHAR(20));
```

Default values can be specified for a column that allows the system to "plug-in" a value if the user does not enter one for that column. Once a CREATE TABLE statement has executed successfully, you can examine its structure using the object browser in Oracle SQL Developer, or use the DESCRIBE command:

```
DESC tablename;
```

Tables can also be created from an existing table by using a subquery on the AS clause of the CREATE TABLE command. You can use all the columns from the existing table, or pick and choose just the columns necessary for the new table. The subquery returns the structure and data from the existing query and maps it into columns and rows in the new table.

Modifying Columns on Tables

Once a table has been created with the CREATE TABLE command, it may become necessary to modify a column on a table. This might be due to an error or omission when the table was created, or business requirements have changed. The ALTER TABLE command is used to modify the underlying structure of the table, not the actual data in the table. To modify the table's data, the INSERT, UPDATE and DELETE statements are used, which will be covered in a later unit.

The basic syntax of the ALTER TABLE command is:
ALTER TABLE tablename action columnname;

To add grade point average to the above Student table, we would code:
ALTER TABLE Student
ADD (Gpa NUMBER(3,2));

To change the student's name column from 20 to 30 characters, we would code:
ALTER TABLE Student
MODIFY (StudentName CHAR(30));

To remove the Gpa column from the Student table, we would code:
ALTER TABLE Student
DROP (Gpa);

Once a column has been dropped, you can't retrieve it. The column and of course the data for that column is gone. Proceed with caution! The SET/UNUSED and DROP/UNUSED feature of Oracle can be used to "postpone" the actual deletion of a column until the database server is not as busy, such as after normal business hours.

Modifying Tables

In addition to modifying a column, we can make changes that affect the entire table, by renaming the table, deleting all the rows on a table, or deleting the table. To rename the Student table to Students, we would code: RENAME Student TO Students;

To delete all the rows from the Students table and free up the space that the rows occupied, we would code: TRUNCATE TABLE Students;

As you will see in a later unit, the DELETE command will also remove rows from a table, but the space is still allocated to the database.

To completely remove the Students table from the database, we would code:
DROP TABLE Students;

This will delete the table and its data from the active database, but Oracle places a copy of it in its Recycle Bin, which can then be "brought back" by coding: FLASHBACK TABLE flashbackname TO BEFORE DROP; where flashbackname is an Oracle assigned name placed in the Recycle bin. To delete the table permanently, with no copy of it placed in the Recycle Bin, add the keyword PURGE: DROP TABLE Students PURGE;

Recall the use of script files that you have been loading and executing for each unit. These script files contain statements to drop your tables and re-create them with changes necessary to complete the unit lab assignments, and exercises found in your textbook. Now that you are familiar with table creation and modification statements, take a closer look at the script files to see how they are coded and how they work.
