Database Programming with SQL
13-3: Modifying a Table
Practice Solutions

Try It / Solve It

Before beginning the practice exercises, execute a DESCRIBE for each of the following tables: o_employees and o_jobs. These tables will be used in the exercises. You will need to know which columns do not allow null values.

NOTE: If students have not already created the o_employees, o_departments, and o_jobs tables they should create them using the four steps outlined in the practice.

1. **Create the three o_tables – jobs, employees, and departments – using the syntax:**
   CREATE TABLE o_jobs AS (SELECT * FROM jobs);
   CREATE TABLE o_employees AS (SELECT * FROM employees);
   CREATE TABLE o_departments AS (SELECT * FROM departments);

2. **Add the Human Resources job to the jobs table:**
   INSERT INTO o_jobs (job_id, job_title, min_salary, max_salary)
   VALUES('HR_MAN', 'Human Resources Manager', 4500, 5500);

3. **Add the three new employees to the employees table:**
   INSERT INTO o_employees (employee_id, first_name, last_name, email, hire_date, job_id)
   VALUES(210, 'Ramon', 'Sanchez', 'RSANCHEZ', SYSDATE, 'HR_MAN');

4. **Add Human Resources to the departments table:**
   INSERT INTO o_departments(department_id, department_name)
   VALUES (210,'Human Resources');

1. Why is it important to be able to modify a table?

   Solution:
   It is important to be able to modify a table because the requirements of a business may change or to remedy mistakes that may have been made.

2. CREATE a table called Artists.
   a. Add the following to the table:
      - artist ID
      - first name
      - last name
      - band name
      - email
      - hourly rate
      - song ID from d_songs table

   b. INSERT one artist from the d_songs table.
   c. INSERT one artist of your own choosing; leave song_id blank.
   d. Give an example how each of the following may be used on the table that you have created:
      1) ALTER TABLE
      2) DROP TABLE
      3) RENAME TABLE
      4) TRUNCATE
      5) COMMENT ON TABLE

Solution:

a. Explain to students how you want the DJs on Demand artist's table assignment to be completed. Students should be able to list the term followed by the SQL statement they used. For example:

```
CREATE TABLE artists
(artist_id NUMBER(4),
first_name VARCHAR2 (15),
last_name VARCHAR2 (15),
band_name VARCHAR2 (15),
email VARCHAR2(15),
hourly_rate NUMBER (8),
song_id NUMBER(5));
```

b, c, and d answers may vary.

3. In your o_employees table, enter a new column called “Termination.” The datatype for the new column should be VARCHAR2. Set the DEFAULT for this column as SYSDATE to appear as character data in the format: February 20th, 2003.

   Solution:
   ALTER TABLE o_employees
   ADD (termination VARCHAR2(20) DEFAULT TO_CHAR(SYSDATE, 'Month ddth, YYYY'));

4. Create a new column in the o_employees table called start_date. Use the TIMESTAMP WITH LOCAL TIME ZONE as the datatype.

   Solution:
   ALTER TABLE o_employees
   ADD start_date TIMESTAMP WITH LOCAL TIME ZONE;

5. Truncate the o_job_description table. Then do a SELECT * statement.
   Are the columns still there? Is the data still there?

   Solution:
   TRUNCATE TABLE o_job_description;
   The TRUNCATE TABLE command removes all rows from the table and releases storage space used by the table.

6. What is the distinction between TRUNCATE, DELETE, and DROP for tables?

   Solution:
   TRUNCATE removes the data from a table and releases storage space, but it does not create any rollback information.
   DELETE removes the data from a table but *does* generate rollback information
   DROP removes the table.

7. List the changes that can and cannot be made to a column.

   Solution:

   - You can increase the width or precision of a numeric column.
   - You can increase the width of a character column.
   - You can decrease the width of a column only if the column contains only null values or if the table has no rows.
   - You can change the datatype only if the column contains null values.
   - You can convert a CHAR column to VARCHAR2 or convert a VARCHAR2 column to CHAR only if the column contains null values or if you do not change the size.
   - A change to the DEFAULT value of a column affects only later insertions to the table.

8. Add the following comment to the o_jobs table:
   "New job description added"
   View the data dictionary to view your comments.

   Solution:

   ```
   COMMENT ON TABLE o_jobs
   IS 'New job description added';

   SELECT *
   FROM USER_TAB_COMMENTS;
   ```

9. Rename the o_jobs table to o_job_description.

   Solution:

   ```
   RENAME o_jobs to o_job_description;
   ```

10. F_staffs table exercises:

   a. Create a copy of the f_staffs table called copy_f_staffs and use this copy table for the remaining labs in this lesson.
   Solution:
   CREATE TABLE copy_f_staffs
   AS SELECT * FROM f_staffs

   b. Describe the new table to make sure it exists.
   Solution:
   DESC copy_f_staffs;

   c. Drop the table.
   Solution:
   DROP TABLE copy_f_staffs;

   d. Try to select from the table.
   Solution:
   SELECT *
   FROM copy_f_staffs; – this will error, as table no longer exists

   e. Investigate your recyclebin to see where the table went.
   Solution:
   SELECT *
   FROM user_recyclebin

   f. Try to select from the dropped table by using the value stored in the OBJECT_NAME column. You will need to copy and paste the name as it is exactly, and enclose the new name in “ “ (double quotes). So if the dropped name returned to you is BIN$Q+x1nJdcUnngQESYELVIdQ==$0, you need to write a query that refers to “BIN$Q+x1nJdcUnngQESYELVIdQ==$0”.

   Solution:
   SELECT *
   FROM "BIN$Q+x1nJdcUnngQESYELVIdQ==$0"
   Answers will vary, as each dropped name will be different.

   g. Undrop the table.

   Solution:
   FLASHBACK table copy_f_staffs TO BEFORE DROP

   h. Describe the table.

   Solution:
   DESC copy_f_staffs;

11. Still working with the copy_f_staffs table, perform an update on the table.

    a. Issue a select statement to see all rows and all columns from the copy_f_staffs table;
    Solution:
    ```
    SELECT *
    FROM copy_f_staffs
    ```

    b. Change the salary for Sue Doe to 12 and commit the change.
    Solution:
    ```
    UPDATE copy_f_staffs
    SET salary = 12
    WHERE id = 12
    ```

    c. Issue a select statement to see all rows and all columns from the copy_f_staffs table;
    Solution:
    ```
    SELECT *
    FROM copy_f_staffs
    ```

    d. For Sue Doe, update the salary to 2 and commit the change.
    Solution:
    ```
    UPDATE copy_f_staffs
    SET salary = 2
    WHERE id = 12
    ```

    e. Issue a select statement to see all rows and all columns from the copy_f_staffs table;
    Solution:
    ```
    SELECT *
    FROM copy_f_staffs
    ```

    f. Now, issue a FLASHBACK QUERY statement against the copy_f_staffs table, so you can see all the changes made.
    Solution:
    ```
    SELECT versions_starttime "start", versions_endtime "end", salary
    FROM   copy_f_staffs
    VERSIONS BETWEEN SCN MINVALUE AND MAXVALUE
    ```

    g. Investigate the result of f), and find the original salary and update the copy_f_staffs table salary column for Sue Doe back to her original salary.
    Solution:
    ```
    UPDATE copy_f_staffs
    SET salary = 6.75
    WHERE id = 12
    ```