Database Programming with SQL
12-1: INSERT Statements
Practice Activities

Objectives

- Give examples of why it is important to be able to alter the data in a database
- Construct and execute INSERT statements that insert a single row using a VALUES clause
- Construct and execute INSERT statements that use special values, null values, and date values
- Construct and execute INSERT statements that copy rows from one table to another using a subquery

Vocabulary

Identify the vocabulary word for each definition below.

End user	Someone doing “real work” with the computer, using it as a means rather than an end
Transaction	Consists of a collection of DML statements that form a logical unit of work.
Explicit	Fully and clearly expressed; leaving nothing implied
INSERT	Adds a new row to a table

Try It / Solve It

Students should execute DESC tablename before doing INSERT to view the data types for each column. VARCHAR2 data-type entries need single quotation marks in the VALUES statement.

1. Give two examples of why it is important to be able to alter the data in a database.

   Keeping customer records up to date.
   Reflecting changes in inventory for a business.

2. DJs on Demand just purchased four new CDs. Use an explicit INSERT statement to add each CD to the copy_d_cds table. After completing the entries, execute a SELECT * statement to verify your work.

CD_Number	Title	Producer	Year
97	Celebrate the Day	R & B Inc.	2003
98	Holiday Tunes for All Ages	Tunes are Us	2004
99	Party Music	Old Town Records	2004
100	Best of Rock and Roll	Old Town Records	2004

```
INSERT INTO copy_d_cds (CD_Number, Title, Producer, Year)
VALUES (97, 'Celebrate the Day', 'R & B Inc.', 2003);
INSERT INTO copy_d_cds (CD_Number, Title, Producer, Year)
VALUES (98, 'Holiday Tunes for All Ages', 'Tunes are Us', 2004);
INSERT INTO copy_d_cds (CD_Number, Title, Producer, Year)
VALUES (99, 'Party Music', 'Old Town Records', 2004);
INSERT INTO copy_d_cds (CD_Number, Title, Producer, Year)
VALUES (100, 'Best of Rock and Roll', 'Old Town Records', 2004);
```

```
SELECT * FROM copy_d_cds
WHERE cd_number IN ('97', '98', '98', '99' ,'100');
```

3. DJs on Demand has two new events coming up. One event is a fall football party and the other event is a sixties theme party. The DJs on Demand clients requested the songs shown in the table for their events. Add these songs to the copy_d_songs table using an implicit INSERT statement.

ID	Title	Duration	Type_Code
52	Surfing Summer	Not known	12
53	Victory Victory	5 min	12

```
INSERT INTO copy_d_songs
VALUES (52, 'Surfing Summer', NULL, NULL, 12);

INSERT INTO copy_d_songs
VALUES (53, 'Victory Victory', '5 min', NULL, 12);
```

```
SELECT * FROM copy_d_songs WHERE ID IN ('52', '53');
```

4. Add the two new clients to the copy_d_clients table. Use either an implicit or an explicit INSERT.

Client_Number	First_Name	Last_Name	Phone	Email
6655	Ayako	Dahish	3608859030	dahisha@harbor.net
6689	Nick	Neuville	9048953049	nnicky@charter.net

```
INSERT INTO copy_d_clients (Client_Number, First_Name, Last_Name, Phone, Email)
VALUES (6655, 'Ayako', 'Dahish', '3608859030', 'dahisha@harbor.net');

INSERT INTO copy_d_clients (Client_Number, First_Name, Last_Name, Phone, Email)
VALUES (6689, 'Nick', 'Neuville', '9048953049', 'nnicky@charter.net');
```

5. Add the new client's events to the copy_d_events table. The cost of each event has not been determined at this date.

ID	Name	Event_ Date	Description	Cost	Venue_ ID	Package_ Code	Theme_ Code	Client_ Number
110	Ayako Anniversary	07-Jul-2004	Party for 50, sixties dress, decorations		245	79	240	6655
115	Neuville Sports Banquet	09-Sep-2004	Barbecue at residence, college alumni, 100 people		315	87	340	6689

```
INSERT INTO copy_d_events (ID, Name, Event_Date, Description, Cost, Venue_ID, Package_Code, Theme_Code,
Client_Number)
VALUES (110, 'Ayako Anniversary', '07/07/2004', 'Party for 50, sixties dress, decorations', 0, 245, 79, 240,
6655);

INSERT INTO copy_d_events (ID, Name, Event_Date, Description, Cost, Venue_ID, Package_Code, Theme_Code,
Client_Number)
VALUES (115, 'Neuville Sports Banquet', '09/09/2004', 'Barbecue at residence, college alumni, 100 people', 0,
315, 87, 340, 6689);
```

6. Create a table called rep_email using the following statement:

```
CREATE TABLE rep_email (
id NUMBER(3) CONSTRAINT rel_id_pk PRIMARY KEY,
first_name VARCHAR2(10),
last_name VARCHAR2(10),
email_address VARCHAR2(10))
```

```
INSERT INTO rep_email (id, first_name, last_name,
email_address)
SELECT employee_id, first_name, last_name, email
FROM employees
WHERE job_id LIKE '%REP%';
```

Populate this table by running a query on the employees table that includes only those employees who are REP's.

Database Programming with SQL
12-2: Updating Column Values and Deleting Rows
Practice Activities

Objectives

- Construct and execute an UPDATE statement
- Construct and execute a DELETE statement
- Construct and execute a query that uses a subquery to update and delete data from a table
- Construct and execute a query that uses a correlated subquery to update and delete from a table
- Explain how foreign-key and primary-key integrity constraints affect UPDATE and DELETE statements
- Explain the purpose of the FOR UPDATE Clause in a SELECT statement

Vocabulary

Identify the vocabulary word for each definition below.

UPDATE	Modifies existing rows in a table
Subquery UPDATE	retrieves information from one table & uses the information to update another table
Integrity constraint	Ensures that the data adheres to a predefined set of rules
Cascading delete	deletes information on a linked table based on what was deleted on the other table
DELETE	Removes existing rows from a table

Try It / Solve It

NOTE: Copy tables in this section do not exist

If any change is not possible, give an explanation as to why it is not possible.

1. Monique Tuttle, the manager of Global Fast Foods, sent a memo requesting an immediate change in prices. The price for a strawberry shake will be raised from $3.59 to $3.75, and the price for fries will increase to $1.20. Make these changes to the copy_f_food_items table.

```
create table copy_f_food_items as (select * from copy_f_food_items);

UPDATE copy_f_food_items
SET price = 3.75
WHERE DESCRIPTION = 'Strawberry Shake';

UPDATE copy_f_food_items
SET price = 1.20
WHERE DESCRIPTION = 'Fries';
```

2. Bob Miller and Sue Doe have been outstanding employees at Global Fast Foods. Management has decided to reward them by increasing their overtime pay. Bob Miller will receive an additional $0.75 per hour and Sue Doe will receive an additional $0.85 per hour. Update the copy_f_staffs table to show these new values. (Note: Bob Miller currently doesn't get overtime pay. What function do you need to use to convert a null value to 0?)

```
create table copy_f_staffs as (select * from f_staffs);

UPDATE copy_f_staffs
SET overtime_rate = COALESCE(overtime_rate, 0) +
0.75
WHERE first_name = 'Bob' AND last_name = 'Miller';

UPDATE copy_f_staffs
SET overtime_rate = COALESCE(overtime_rate, 0) +
0.85
WHERE first_name = 'Sue' AND last_name = 'Doe';
```

3. Add the orders shown to the Global Fast Foods copy_f_orders table:

ORDER_NUMBER	ORDER_DATE	ORDER_TOTAL	CUST_ID	STAFF_ID
5680	June 12, 2004	159.78	145	9
5691	09-23-2004	145.98	225	12
5701	July 4, 2004	229.31	230	12

```
INSERT INTO copy_f_orders (ORDER_NUMBER, ORDER_DATE, ORDER_TOTAL, CUST_ID, STAFF_ID)
VALUES (5680, TO_DATE('June 12, 2004', 'Month DD, YYYY'), 159.78, 145, 9);

INSERT INTO copy_f_orders (ORDER_NUMBER, ORDER_DATE, ORDER_TOTAL, CUST_ID, STAFF_ID)
VALUES (5691, TO_DATE('09-23-2004', 'MM-DD-YYYY'), 145.98, 225, 12);

INSERT INTO copy_f_orders (ORDER_NUMBER, ORDER_DATE, ORDER_TOTAL, CUST_ID, STAFF_ID)
VALUES (5701, TO_DATE('July 4, 2004', 'Month DD, YYYY'), 229.31, 230, 12);
```

4. Add the new customers shown below to the copy_f_customers table. You may already have added Katie Hernandez. Will you be able to add all these records successfully?

ID	FIRST_ NAME	LAST_ NAME	ADDRESS	CITY	STATE	ZIP	PHONE_NUMBER
145	Katie	Hernandez	92 Chico Way	Los Angeles	CA	98008	8586667641
225	Daniel	Spode	1923 Silverado	Denver	CO	80219	7193343523
230	Adam	Zurn	5 Admiral Way	Seattle	WA		4258879009

```
INSERT INTO copy_f_customers (ID, FIRST_NAME, LAST_NAME, ADDRESS, CITY, STATE, ZIP, PHONE_NUMBER)
VALUES (145, 'Katie', 'Hernandez', '92 Chico Way', 'Los Angeles', 'CA', 98008, '8586667641');
```

Cannot add ID 230, ZIP cannot be null.

```
INSERT INTO copy_f_customers (ID, FIRST_NAME, LAST_NAME, ADDRESS, CITY, STATE, ZIP, PHONE_NUMBER)
VALUES (225, 'Daniel', 'Spode', '1923 Silverado', 'Denver', 'CO', 80219, '7193343523');
```

5. Sue Doe has been an outstanding Global Foods staff member and has been given a salary raise. She will now be paid the same as Bob Miller. Update her record in copy_f_staffs.

```
UPDATE copy_f_staffs
SET salary = (SELECT salary FROM copy_f_staffs WHERE first_name = 'Bob' AND last_name = 'Miller')
WHERE first_name = 'Sue' AND last_name = 'Doe';
```

6. Global Fast Foods is expanding their staff. The manager, Monique Tuttle, has hired Kai Kim. Not all information is available at this time, but add the information shown here.

ID	FIRST_NAME	LAST_NAME	BIRTHDATE	SALARY	STAFF_TYPE
25	Kai	Kim	3-Nov-1988	6.75	Order Taker

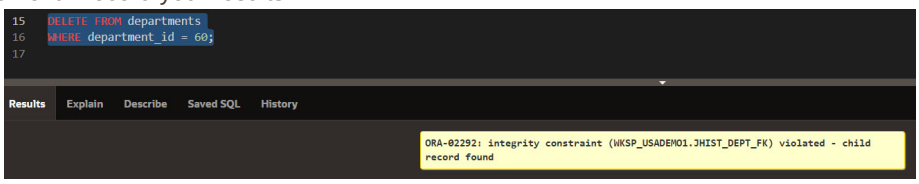
```
INSERT INTO copy_f_staffs (ID, FIRST_NAME, LAST_NAME, BIRTHDATE, SALARY, STAFF_TYPE)
VALUES (25, 'Kai', 'Kim', TO_DATE('03-Nov-1988', 'DD-Mon-YYYY'), 6.75, 'Order Taker');
```

7. Now that all the information is available for Kai Kim, update his Global Fast Foods record to include the following: Kai will have the same manager as Sue Doe. He does not qualify for overtime. Leave the values for training, manager budget, and manager target as null.

```
UPDATE copy_f_staffs
SET manager_id = (SELECT manager_id FROM copy_f_staffs WHERE first_name = 'Sue' AND last_name = 'Doe'),
    overtime_rate = 0
WHERE first_name = 'Kai' AND last_name = 'Kim';
```

8. Execute the following SQL statement. Record your results.

DELETE from departments
WHERE department_id = 60;



9. Kim Kai has decided to go back to college and does not have the time to work and go to school. Delete him from the Global Fast Foods staff. Verify that the change was made.

```
DELETE FROM copy_f_staffs
WHERE first_name = 'Kai' AND last_name = 'Kim';

select * from copy_f_staffs;
```

10. Create a copy of the employees table and call it lesson7_emp;
Once this table exists, write a correlated delete statement that will delete any employees from the lesson7_employees table that also exist in the job_history table.

```
CREATE TABLE lesson7_emp AS
SELECT * FROM employees;
```

```
DELETE FROM lesson7_emp e
WHERE EXISTS (
    SELECT 1 FROM job_history jh
    WHERE jh.employee_id = e.employee_id
);
```

Database Programming with SQL
12-3: DEFAULT Values, MERGE, and Multi-Table Inserts
Practice Activities

Objectives

- Understand when to specify a DEFAULT value
- Construct and execute a MERGE statement
- Construct and execute DML statements using SUBQUERIES
- Construct and execute multi-table inserts

Try It / Solve It

1. When would you want a DEFAULT value?

```
The column should have a predefined value if no specific value is
provided during data insertion.
You need to enforce consistent values for specific columns.
To reduce errors when inserting new rows, especially in scenarios
where data is optional.
```

2. Currently, the Global Foods F_PROMOTIONAL_MENUS table START_DATE column does not have SYSDATE set as DEFAULT. Your manager has decided she would like to be able to set the starting date of promotions to the current day for some entries. This will require three steps:

   a. In your schema, Make a copy of the Global Foods F_PROMOTIONAL_MENUS table using the following SQL statement:

      CREATE TABLE copy_f_promotional_menus
      AS (SELECT * FROM f_promotional_menus)

   b. Alter the current START_DATE column attributes using:

      ALTER TABLE copy_f_promotional_menus
      MODIFY(start_date DATE DEFAULT SYSDATE)

   c. INSERT the new information and check to verify the results.
      INSERT a new row into the copy_f_promotional_menus table for the manager's new promotion. The promotion code is 120. The name of the promotion is 'New Customer.' Enter DEFAULT for the start date and '01-Jun-2005' for the ending date. The giveaway is a 10%discount coupon. What was the correct syntax used?

```
INSERT INTO copy_f_promotional_menus (code, name, start_date, end_date, give_away)
VALUES (120, 'New Customer', DEFAULT, TO_DATE('01-Jun-2005', 'DD-Mon-YYYY'), '10%
discount coupon');
```

CODE	NAME	START_DATE	END_DATE	GIVE_AWAY
100	Back to School	09/01/2004	09/30/2004	ballpen and highlighter
110	Valentines Special	02/10/2004	02/15/2004	small box of chocolates
120	New Customer	11/17/2024	06/01/2005	10% discount coupon

3. Allison Plumb, the event planning manager for DJs on Demand, has just given you the following list of CDs she acquired from a company going out of business. She wants a new updated list of CDs in inventory in an hour, but she doesn't want the original D_CDS table changed. Prepare an updated inventory list just for her.

   a. Assign new cd_numbers to each new CD acquired.

   b. Create a copy of the D_CDS table called manager_copy_d_cds. What was the correct syntax used?

```
CREATE TABLE manager_copy_d_cds AS (SELECT * FROM d_cds);
```

   c. INSERT into the manager_copy_d_cds table each new CD title using an INSERT statement. Make up one example or use this data:
   20, 'Hello World Here I Am', 'Middle Earth Records', '1998'
   What was the correct syntax used?

```
INSERT INTO manager_copy_d_cds (cd_number, title,
producer, year)
VALUES (20, 'Hello World Here I Am', 'Middle Earth
Records', '1998');
```

   d. Use a merge statement to add to the manager_copy_d_cds table, the CDs from the original table. If there is a match, update the title and year. If not, insert the data from the original table. What was the correct syntax used?

```
MERGE INTO manager_copy_d_cds target
USING d_cds source
ON (target.cd_number = source.cd_number)
WHEN MATCHED THEN
  UPDATE SET target.title = source.title, target.year = source.year
WHEN NOT MATCHED THEN
  INSERT (cd_number, title, producer, year)
  VALUES (source.cd_number, source.title, source.producer, source.year);
```

4. Run the following 3 statements to create 3 new tables for use in a Multi-table insert statement. All 3 tables should be empty on creation, hence the WHERE 1=2 condition in the WHERE clause.

```
CREATE TABLE sal_history (employee_id, hire_date, salary)
AS SELECT employee_id, hire_date, salary

  FROM   employees
WHERE  1=2;


CREATE TABLE mgr_history (employee_id, manager_id, salary) AS SELECT
employee_id, manager_id, salary

  FROM   employees
WHERE  1=2;


CREATE TABLE special_sal (employee_id, salary) AS SELECT
employee_id, salary

  FROM   employees
WHERE  1=2;
```

Once the tables exist in your account, write a Multi-Table insert statement to first select the employee_id, hire_date, salary, and manager_id of all employees. If the salary is more than 20000 insert the employee_id and salary into the special_sal table. Insert the details of employee_id, hire_date, and salary into the sal_history table. Insert the employee_id, manager_id, and salary into the mgr_history table.

You should get a message back saying 39 rows were inserted. Verify you get this message and verify you have the following number of rows in each table:

Sal_history: 19 rows
Mgr_history: 19 rows
Special_sal: 1

```
INSERT ALL
  WHEN salary > 20000 THEN
    INTO special_sal (employee_id, salary)
    VALUES (employee_id, salary)
  INTO sal_history (employee_id, hire_date, salary)
    VALUES (employee_id, hire_date, salary)
  INTO mgr_history (employee_id, manager_id, salary)
    VALUES (employee_id, manager_id, salary)
SELECT employee_id, hire_date, salary, manager_id
FROM employees;
```

Database Programming with SQL
13-1: Creating Tables
Practice Activities

Objectives

- List and categorize the main database objects
- Review a table structure
- Describe how database schema objects are used by the Oracle database

Vocabulary

Identify the vocabulary word for each definition below.

Data Dictionary	Created and maintained by the Oracle Server and contains information about the database
Schema	A collection of objects that are the logical structures that directly refer to the data in the database
DEFAULT	Specifies a preset value if a value is omitted in the INSERT statement
Table	Stores data; basic unit of storage composed of rows and columns
CREATE TABLE	Command use to make a new table

Try It / Solve It

1. Complete the GRADUATE CANDIDATE table instance chart. Credits is a foreign-key column referencing the requirements table.

Column Name	student_id	last_name	first_name	credits	graduation_date
Key Type	PK			FK	
Nulls/Unique	NOT NULL	NOT NULL	NOT NULL	NOT NULL	
FK Column				YES	
Datatype	NUMBER	VARCHAR2	VARCHAR2	NUMBER	DATE
Length	6			3	

2. Write the syntax to create the grad_candidates table.

```
CREATE TABLE requirements (
    credits NUMBER(3) PRIMARY KEY
);
```

```
CREATE TABLE grad_candidates (
    student_id NUMBER(6) PRIMARY KEY,
    last_name VARCHAR2(20) NOT NULL,
    first_name VARCHAR2(20) NOT NULL,
    credits NUMBER(3) NOT NULL,
    graduation_date DATE,
    CONSTRAINT fk_credits FOREIGN KEY (credits) REFERENCES
requirements(credits)
);
```

3. Confirm creation of the table using DESCRIBE.

```
describe grad_candidates;
```

4. Create a new table using a subquery. Name the new table your last name -- e.g., smith_table. Using a subquery, copy grad_candidates into smith_table.

```
CREATE TABLE butt_table AS SELECT * FROM grad_candidates;
```

5. Insert your personal data into the table created in question 4.

```
INSERT INTO butt_table (student_id, last_name, first_name, credits, graduation_date)
      VALUES (393, 'Butt', 'Kamran', 12, TO_DATE('15-May-2025', 'DD-Mon-YYYY'));
```

6. Query the data dictionary for each of the following:
   - USER_TABLES
   - USER_OBJECTS
   - USER_CATALOG or USER_CAT

   In separate sentences, summarize what each query will return.

USER_TABLES: Returns the list of tables owned by the user, along with metadata such as row count, storage size, and table status.

USER_OBJECTS: Returns a list of all objects owned by the user, including tables, indexes, views, sequences, and more.

USER_CATALOG: Provides a combined view of all database objects owned by the user, including tables, views, synonyms, and sequences.

Database Programming with SQL
13-2: Using Data Types
Practice Activities

Objectives

- Create a table using TIMESTAMP and TIMESTAMP WITH TIME ZONE column data types
- Create a table using INTERVAL YEAR TO MONTH and INTERVAL DAY TO SECOND column data types
- Give examples of organizations and personal situations where it is important to know to which time zone a date-time value refers
- List and provide an example of each of the number, date, and character data types

Vocabulary

Identify the vocabulary word for each definition below.

INTERVAL YEAR TO MONTH	Allows time to be stored as an interval of years and months
TIMESTAMP WITH LOCAL TIME ZONE	When a column is selected in a SQL statement the time is automatically converted to the user's timezone
BLOB	Binary large object data up to 4 gigabytes
TIMESTAMP WITH TIME ZONE	Stores a time zone value as a displacement from Universal Coordinated Time or UCT
INTERVAL DAY TO SECOND	Allows time to be stored as an interval of days to hours, minutes, and seconds
CLOB	Character data up to 4 gigabytes
TIMESTAMP	Allows the time to be stored as a date with fractional seconds

Try It / Solve It

1. Create tables using each of the listed time-zone data types, use your time-zone and one other in your examples. Answers will vary.

   a. TIMESTAMP WITH LOCAL TIME ZONE
   b. INTERVAL YEAR TO MONTH
   c. INTERVAL DAY TO SECOND

2. Execute a SELECT * from each table to verify your input.

```
SELECT * FROM local_time_example;

SELECT * FROM interval_year_month_example;

SELECT * FROM interval_day_second_example;
```

3. Give 3 examples of organizations and personal situations where it is important to know to which time zone a date-time value refers.

```
a. Global Business Operations:
Coordinating meetings across time zones for multinational teams.

b. Airline Industry:
Managing flight schedules and ensuring accuracy in departure and arrival times.

c. Financial Transactions:
Processing trades or transactions based on stock exchange operating hours in different
time zones.
```

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

Objectives

- Explain why it is important to be able to modify a table
- Explain and provide an example for each of the DDL statements—ALTER, DROP, RENAME, and TRUNCATE—and the effect each has on tables and columns
- Construct a query and execute the ALTER TABLE commands ADD, MODIFY, and DROP
- Explain and perform a FLASHBACK QUERY on a table
- Explain and perform FLASHBACK table operations
- Track the changes to data over a period of time
- Explain the rationale for using TRUNCATE versus DELETE for tables
- Add a comment to a table using the COMMENT ON TABLE command
- Name the changes that can and cannot be made to modify a column
- Explain when and why the SET UNUSED statement is advantageous

Try It / Solve It

Before beginning the practice exercises, execute a DESCRIBE for each of the following tables: o_employees, o_departments and o_jobs. These tables will be used in the exercises. If they do not exist in your account, create them as follows:

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');
```

You will need to know which columns do not allow null values.

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

```
To adapt the database schema to changing requirements without
losing existing data.
To add new functionality or meet evolving business needs
(e.g., new columns, constraints).
To maintain data integrity by updating rules and constraints.
```

2. CREATE a table called Artists.

   a. Add the following to the table:

   - artist ID
   - first name
   - last name
   - band name
   - email
   - hourly rate

```
CREATE TABLE artists (
    artist_id NUMBER PRIMARY KEY,
    first_name VARCHAR2(30),
    last_name VARCHAR2(30),
    band_name VARCHAR2(50),
    email VARCHAR2(50),
    hourly_rate NUMBER(5, 2)
);
```

   b. INSERT one artist from the d_songs table.

   c. INSERT one artist of your own choosing.

   d. Give an example how each of the following may be used on the table that you have created:

   1) ALTER TABLE
```
ALTER TABLE artists ADD genre VARCHAR2(20);
```
   2) DROP TABLE
```
DROP TABLE artists;
```
   3) RENAME TABLE
```
RENAME artists TO music_artists;
```
   4) TRUNCATE
```
TRUNCATE TABLE artists;
```
   5) COMMENT ON TABLE
```
COMMENT ON TABLE artists IS 'Table for storing artist details';
```

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.

```
ALTER TABLE o_employees
ADD termination VARCHAR2(50) 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.

```
ALTER TABLE o_employees
ADD start_date TIMESTAMP WITH LOCAL TIME ZONE;
```

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

```
TRUNCATE TABLE o_jobs;

SELECT * FROM o_jobs; ---> no data found
```

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

```
TRUNCATE: Removes all rows but keeps the structure; faster, cannot be rolled back.
DELETE: Removes specific rows or all rows; can be rolled back.
DROP: Completely removes the table from the database, including its structure.
```

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

```
Allowed:
Add a column.
Modify a column's data type, size, or default value.
Drop a column.
Rename a column or table.
```

```
Not Allowed:
Reduce the size of a column if data exists.
Change a column to a data type incompatible with
existing data.
```

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

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

SELECT comments FROM user_tab_comments WHERE table_name = 'O_JOBS';
```

9. Rename the o_jobs table to o_job_description.

```
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.

```
CREATE TABLE copy_f_staffs AS SELECT * FROM f_staffs;
```

b. Describe the new table to make sure it exists.

```
DESC copy_f_staffs;
```

c. Drop the table.

```
DROP TABLE copy_f_staffs;
```

d. Try to select from the table.

```
--> no data found
```

e. Investigate your recyclebin to see where the table went.

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".

g. Undrop the table.

```
FLASHBACK TABLE copy_f_staffs TO BEFORE DROP;
```

h. Describe the table.

```
Table   Column  Data Type      Length  Precision      Scale   Primary Key    Nullable
Default Comment
COPY_F_STAFFS  ID      NUMBER  -       5       0      -       nullable       -       -
       FIRST_NAME      VARCHAR2        25      -      -       -      -       -       -
       LAST_NAME       VARCHAR2        35      -      -       -      -       -       -
       BIRTHDATE       DATE    7       -      -       -       -      -       -
       SALARY  NUMBER  -       8       2      -       -       -
       OVERTIME_RATE   NUMBER  -       5       2      -       nullable       -       -
       TRAINING        VARCHAR2        50      -      -       -      nullable       -       -
       STAFF_TYPE      VARCHAR2        20      -      -       -      -       -       -
       MANAGER_ID      NUMBER  -       5       0      -       nullable       -       -
       MANAGER_BUDGET  NUMBER  -       8       2      -       nullable       -       -
       MANAGER_TARGET  NUMBER  -       8       2      -       nullable       -       -
```

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;

```
SELECT * FROM copy_f_staffs;
```

b. Change the salary for Sue Doe to 12 and commit the change.

```
UPDATE copy_f_staffs
SET salary = 12
WHERE first_name = 'Sue' AND last_name = 'Doe';
```

c. Issue a select statement to see all rows and all columns from the copy_f_staffs table;

```
SELECT * FROM copy_f_staffs;
```

d. For Sue Doe, update the salary to 2 and commit the change.

```
UPDATE copy_f_staffs
SET salary = 2
WHERE first_name = 'Sue' AND last_name = 'Doe';
```

e. Issue a select statement to see all rows and all columns from the copy_f_staffs table;

```
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.

```
SELECT * FROM copy_f_staffs
VERSIONS BETWEEN TIMESTAMP 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.

```
SELECT * FROM copy_f_staffs
VERSIONS BETWEEN TIMESTAMP MINVALUE AND MAXVALUE
WHERE ID = 12;

UPDATE copy_f_staffs
SET salary = 6.75
WHERE first_name = 'Sue' AND last_name = 'Doe';
```