12-1: INSERT Statements

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

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

```
1   SELECT * FROM COPY_D_CDS
```

Results | Explain | Describe | Saved SQL | History

CD_NUMBER	TITLE	PRODUCER	YEAR
98	Holiday Tunes for All Ages	Tunes are Us	2003
97	Celebrate the Day	R & B Inc.	2003
99	Party Music	Old Town Records	2004
100	Best of Rock and Roll	Old Town Records	2004

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

```sql
INSERT INTO copy_d_songs (ID, Title, Duration, Type_Code)
VALUES (52, 'Surfing Summer', 'Not known', 12);

INSERT INTO copy_d_songs (ID, Title, Duration, Type_Code)
VALUES (53, 'Victory Victory', '5 min', 12);

Select * from copy_d_songs
```

Results | Explain | Describe | Saved SQL | History

ID	TITLE	DURATION	
53	Victory Victory	5 min	12
52	Surfing Summer	Not known	12

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

```sql
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');

SELECT * FROM COPY_D_CLIENTS
```

Results | Explain | Describe | Saved SQL | History

CLIENT_NUMBER	FIRST_NAME	LAST_NAME	PHONE	EMAIL
6655	Ayako	Dahish	3608859030	dahisha@harbor.net
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', TO_DATE('07-Jul-2004', 'DD-Mon-YYYY'), 'Party for 50, sixties dress, decorations', NULL,

INSERT INTO copy_d_events (ID, Name, Event_Date, Description, Cost, Venue_ID, Package_Code, Theme_Code, Client_Number)
VALUES (115, 'Neuville Sports Banquet', TO_DATE('09-Sep-2004', 'DD-Mon-YYYY'), 'Barbecue at residence, college alumni, 10

SELECT * FROM COPY_D_EVENTS
```

Results | Explain | Describe | Saved SQL | History

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

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))

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

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

SELECT * FROM REP_EMAIL
```

Results | Explain | Describe | Saved SQL | History

no data found

12-2: Updating Column Values and Deleting Rows

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.

```sql
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';

select * from copy_f_food_items
```

Results | Explain | Describe | Saved SQL | History

FOOD_ITEM_NUMBER	DESCRIPTION	PRICE	REGULAR_CODE	PROMO_CODE
90	Fries	1.2	20	-
93	Strawberry Shake	3.75	-	110

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?)

```sql
UPDATE copy_f_staffs
SET OVERTIME_RATE = NVL(OVERTIME_RATE, 0) + 0.75
WHERE FIRST_NAME = 'Bob' AND LAST_NAME = 'Miller';

UPDATE copy_f_staffs
SET OVERTIME_RATE = NVL(OVERTIME_RATE, 0) + 0.85
WHERE FIRST_NAME = 'Sue' AND LAST_NAME = 'Doe';

Select * from copy_f_staffs
```

Results | Explain | Describe | Saved SQL | History

ID	FIRST_NAME	LAST_NAME	BIRTHDATE	SALARY	OVERTIME_RATE	TRAINING	STAFF_TYPE	MANAGER_ID	MANAGER_BUDGET	MANAGER_TARGET
12	Sue	Doe	01-Jul-1980	6.75	11.1	-	Order Taker	19	-	-
9	Bob	Miller	19-Mar-1979	10	.75	-	Grill Cook	19	-	-
19	Monique	Tuttle	30-Mar-1969	60	-	-	Manager	-	50000	70000

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

```
1   select * from copy_f_orders
```

Results | Explain | Describe | Saved SQL | History

ORDER_NUMBER	ORDER_DATE	ORDER_TOTAL	CUST_ID	STAFF_ID
5678	10-Dec-2002	103.02	123	12
5680	12-Jun-2004	159.78	145	9
5691	23-Sep-2004	145.98	225	12
5701	04-Jul-2004	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?

DP_12_2_Practice.pdf

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

```
1   select * from copy_f_customers
```

Results | Explain | Describe | Saved SQL | History

ID	FIRST_NAME	LAST_NAME	ADDRESS	CITY	STATE	ZIP	PHONE_NUMBER
123	Cole	Bee	123 Main Street	Orlando	FL	32838	4075558234
456	Zoe	Twee	1009 Oliver Avenue	Boston	MA	12889	7098675309
225	Daniel	Spode	1923 Silverado	Denver	CO	80219	7193343523
145	Katie	Hernandez	92 Chico Way	Los Angeles	CA	90001	8586667641
230	Adam	Zurn	5 Admiral Way	Seattle	WA	98008	4258879009

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.

```
1   Select * from copy_f_staffs
```

Results | Explain | Describe | Saved SQL | History

ID	FIRST_NAME	LAST_NAME	BIRTHDATE	SALARY	OVERTIME_RATE	TRAINING	STAFF_TYPE	MANAGER_ID	MANAGER_BUDGET	MANAGER_TARGET
12	Sue	Doe	01-Jul-1980	10	11.1	-	Order Taker	19	-	-
9	Bob	Miller	19-Mar-1979	10	.75	-	Grill Cook	19	-	-
19	Monique	Tuttle	30-Mar-1969	60	-	-	Manager	-	50000	70000

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

```
select * from copy_f_staffs
```

Results | Explain | Describe | Saved SQL | History

ID	FIRST_NAME	LAST_NAME	BIRTHDATE	SALARY	OVERTIME_RATE	TRAINING	STAFF_TYPE	MANAGER_ID	MANAGER_BUDGET	MANAGER_TARGET
25	Kai	Kim	03-Nov-1988	6.75	-	-	Order Taker	-	-	-
12	Sue	Doe	01-Jul-1980	10	11.1	-	Order Taker	19	-	-
9	Bob	Miller	19-Mar-1979	10	.75	-	Grill Cook	19	-	-
19	Monique	Tuttle	30-Mar-1969	60	-	-	Manager	-	50000	70000

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 = 'D
    OVERTIME_RATE = NULL,
    TRAINING = NULL,
    MANAGER_BUDGET = NULL,
    MANAGER_TARGET = NULL
WHERE
    ID = 25;
```

Results | Explain | Describe | Saved SQL | History

1 row(s) updated.

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.

```
SELECT * FROM copy_f_staffs
WHERE ID = 25;
```

Results | Explain | Describe | Saved SQL | History

no data found

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.

```
1  SELECT * FROM lesson7_emp;
```

Results | Explain | Describe | Saved SQL | History

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	COMMISSION_PCT	MANAGER_ID	DEPARTMENT_ID	BONUS	DATE_OF_BIRTH	OVE
100	Steven	King	SKING	515.123.4567	17-Jun-2002	AD_PRES	24000	-	-	90	-	15-Jan-1990	-
205	Shelley	Higgins	SHIGGINS	515.123.8080	07-Jun-2009	AC_MGR	12000	-	101	110	-	-	-
206	William	Gietz	WGIETZ	515.123.8181	07-Jun-2009	AC_ACCOUNT	8300	-	205	110	-	-	-
149	Eleni	Zlotkey	EZLOTKEY	011.44.1344.429018	29-Jan-2015	SA_MAN	10500	.2	100	80	1500	-	-
174	Ellen	Abel	EABEL	011.44.1644.429267	11-May-2011	SA_REP	11000	.3	149	80	1700	-	-
178	Kimberely	Grant	KGRANT	011.44.1644.429263	24-May-2014	SA_REP	7000	.15	149	-	-	-	-
124	Kevin	Mourgos	KMOURGOS	650.123.5234	16-Nov-2014	ST_MAN	5800	-	100	50	-	-	-
141	Trenna	Rajs	TRAJS	650.121.8009	17-Oct-2010	ST_CLERK	3500	-	124	50	-	-	-

12-3: DEFAULT Values, MERGE, and Multi-Table Inserts

1. When would you want a DEFAULT value?

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)

```
1  CREATE TABLE copy_f_promotional_menus AS
2  SELECT * FROM f_promotional_menus;
```

Results | Explain | Describe | Saved SQL | History

```
Table created.
```

b. Alter the current START_DATE column attributes using:

ALTER TABLE copy_f_promotional_menus

MODIFY(start_date DATE DEFAULT SYSDATE)

```
1   ALTER TABLE copy_f_promotional_menus
2   MODIFY (start_date DATE DEFAULT SYSDATE);
```

Results Explain Describe Saved SQL History

```
Table altered.
```

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?

```
1   Select * from copy_f_promotional_menus
```

Results Explain Describe Saved SQL History

CODE	NAME	START_DATE	END_DATE	GIVE_AWAY
100	Back to School	01-Sep-2004	30-Sep-2004	ballpen and highlighter
110	Valentines Special	10-Feb-2004	15-Feb-2004	small box of chocolates
120	New Customer	12-Nov-2024	01-Jun-2005	10% discount coupon

13-1: Creating Tables

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					
Nulls/Unique					
FK Column					
Datatype	NUMBER	VARCHAR2	VARCHAR2	NUMBER	DATE
Length	6			3	

Column Name	student_id	last_name	first_name	credits	graduation _date
Key type	Primary Key				
Nulls/Unique	NOT NULL	NOT NULL	NOT NULL	NOT NULL	
FK Column				credits	
Datatype	NUMBER	VARCHAR2	VARCHAR2	NUMBER	DATE
Length	6	50	50	3	

2. Write the syntax to create the grad_candidates table.

```
CREATE TABLE grad_candidates (
    student_id NUMBER(6) PRIMARY KEY,
    last_name VARCHAR2(50) NOT NULL,
    first_name VARCHAR2(50) NOT NULL,
    credits NUMBER(3) NOT NULL,
    graduation_date DATE,
    FOREIGN KEY (credits) REFERENCES requirements(credit_column)
);
```

Results | Explain | Describe | Saved SQL | History

```
Table created.
```

3. Confirm creation of the table using DESCRIBE.

Results | Explain | **Describe** | Saved SQL | History

Object Type **TABLE** Object **GRAD_CANDIDATES**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comme
GRAD_CANDIDATES	STUDENT_ID	NUMBER	-	6	0	1	-	-	-
	LAST_NAME	VARCHAR2	50	-	-	-	-	-	-
	FIRST_NAME	VARCHAR2	50	-	-	-	-	-	-
	CREDITS	NUMBER	-	3	0	-	-	-	-
	GRADUATION_DATE	DATE	7	-	-	-	✓	-	-

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 laura_table AS
SELECT *
FROM grad_candidates;
```

Results | Explain | Describe | Saved SQL | History

```
Table created.
```

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

```sql
INSERT INTO laura_table (student_id, last_name, first_name, credits, graduation_date)
VALUES (1234, 'salinas', 'laura', 120, TO_DATE('2025-05-15', 'YYYY-MM-DD'));
```

Results | Explain | Describe | Saved SQL | History

```
1 row(s) inserted.
```

6. Query the data dictionary for each of the following:In separate sentences, summarize what each query will return.

- USER_TABLES: This query returns information about the tables owned by the current user, including table names, creation dates, etc.
- USER_OBJECTS: This query returns all objects (like tables, views, indexes) owned by the current user.
- USER_CATALOG or USER_CAT: The USER_CATALOG or USER_CAT views contain information about all database objects the user has access to.

13-2: Using Data Types

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

```sql
CREATE TABLE example_timestamp_local (
    id NUMBER PRIMARY KEY,
    event_name VARCHAR2(50),
    event_time TIMESTAMP WITH LOCAL TIME ZONE
);

INSERT INTO example_timestamp_local (id, event_name, event_time)
VALUES (1, 'Sample Event', TO_TIMESTAMP_TZ('2024-12-01 10:00:00 UTC', 'YYYY-MM-DD HH24:MI:SS TZD'));
```

Results | Explain | Describe | Saved SQL | History

```
1 row(s) inserted.
```

b. INTERVAL YEAR TO MONTH

```sql
CREATE TABLE example_interval_year_month (
    id NUMBER PRIMARY KEY,
    event_name VARCHAR2(50),
    duration INTERVAL YEAR TO MONTH
);

INSERT INTO example_interval_year_month (id, event_name, duration)
VALUES (1, 'Annual Maintenance', INTERVAL '1' YEAR);
```

Results | Explain | Describe | Saved SQL | History

```
1 row(s) inserted.
```

c. INTERVAL DAY TO SECOND

```sql
CREATE TABLE example_interval_day_second (
    id NUMBER PRIMARY KEY,
    event_name VARCHAR2(50),
    duration INTERVAL DAY TO SECOND
);

INSERT INTO example_interval_day_second (id, event_name, duration)
VALUES (1, 'Meeting Duration', INTERVAL '2 04:30:00' DAY TO SECOND);
```

Results | Explain | Describe | Saved SQL | History

```
1 row(s) inserted.
```

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

```sql
SELECT * FROM example_timestamp_local;
```

Results | Explain | Describe | Saved SQL | History

ID	EVENT_NAME	EVENT_TIME
1	Sample Event	01-DEC-24 10.00.00.000000 AM

1 rows returned in 0.00 seconds Download

```sql
SELECT * FROM example_interval_year_month;
```

Results | Explain | Describe | Saved SQL | History

ID	EVENT_NAME	DURATION
1	Annual Maintenance	+01-00

1 rows returned in 0.01 seconds Download

```
SELECT * FROM example_interval_day_second;
```

Results | Explain | Describe | Saved SQL | History

ID	EVENT_NAME	DURATION
1	Meeting Duration	+02 04:30:00.000000

1 rows returned in 0.00 seconds Download

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

1. International Airlines.
2. Global E-commerce.
3. Financial Institutions.

Personal Situations:

1. Scheduling a Meeting with International Colleagues
2. Travel Plans.
3. Calling International Loved Ones.

13-3: Modifying a Table

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);

```sql
CREATE TABLE o_jobs AS
SELECT *
FROM jobs;

CREATE TABLE o_employees AS
SELECT *
FROM employees;

CREATE TABLE o_departments AS
SELECT *
FROM departments;
```

Results | Explain | Describe | Saved SQL | History

```
Table created.
```

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);

```sql
INSERT INTO o_jobs (job_id, job_title, min_salary, max_salary)
VALUES ('HR_MAN', 'Human Resources Manager', 4500, 5500);
```

Results | Explain | Describe | Saved SQL | History

```
1 row(s) inserted.
```

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');

```sql
INSERT INTO o_employees (employee_id, first_name, last_name, email, hire_date, job_id)
VALUES (210, 'Ramon', 'Sanchez', 'RSANCHEZ', SYSDATE, 'HR_MAN');

INSERT INTO o_employees (employee_id, first_name, last_name, email, hire_date, job_id)
VALUES (211, 'Maria', 'Lopez', 'MLOPEZ', SYSDATE, 'HR_MAN');

INSERT INTO o_employees (employee_id, first_name, last_name, email, hire_date, job_id)
VALUES (212, 'Carlos', 'Martinez', 'CMARTINEZ', SYSDATE, 'HR_MAN');
```

Results | Explain | Describe | Saved SQL | History

```
1 row(s) inserted.
```

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

```sql
INSERT INTO o_departments (department_id, department_name)
VALUES (210, 'Human Resources');
```

Results | Explain | Describe | Saved SQL | History

```
1 row(s) inserted.
```

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

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

Modifying a table is essential because it allows you to adjust the structure of your database schema without losing existing data.

2. CREATE a table called Artists.

a. Add the following to the table:

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

```sql
CREATE TABLE Artists (
    artist_id NUMBER PRIMARY KEY,
    first_name VARCHAR2(50),
    last_name VARCHAR2(50),
    band_name VARCHAR2(100),
    email VARCHAR2(100),
    hourly_rate NUMBER
);
```

Results | Explain | Describe | Saved SQL | History

```
Table created.
```

b. INSERT one artist from the d_songs table.

```sql
INSERT INTO Artists (artist_id, first_name, last_name, band_name, email, hourly_rate)
SELECT artist_id, first_name, last_name, band_name, email, hourly_rate
FROM d_songs
WHERE Bobby West = 1;
```

c. INSERT one artist of your own choosing.

```sql
INSERT INTO Artists (artist_id, first_name, last_name, band_name, email, hourly_rate)
VALUES (101, 'Ben', 'Barlow', 'NECKDEEP', 'BB@example.com', 100);
```

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 birth_date DATE;

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 of music artists';

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(20) DEFAULT TO_CHAR(SYSDATE, 'Month DDth, YYYY');
```

Results | Explain | Describe | Saved SQL | History

```
Table altered.
```

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;
```

Results | Explain | Describe | Saved SQL | History

```
Table altered.
```

5. Truncate the o_jobs table. Then do a SELECT * statement. Are the columns still there? Is the data still there?
After truncating the table, the columns of the table will remain, but all the data will be removed.

6. What is the distinction between TRUNCATE, DELETE, and DROP for tables?
TRUNCATE: Removes all rows from a table but leaves the structure intact. It is faster than DELETE and does not log individual row deletions.
DELETE: Removes rows from a table based on a condition and can be rolled back. It logs each row deletion and allows you to specify conditions.
DROP: Completely removes the table (structure and data) from the database. It cannot be rolled back.

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

Changes that can be made:

- Change the data type.
- Add or remove a NOT NULL constraint.
- Change the default value.

Changes that cannot be made:

- You cannot change the size of a column if it's used as a primary key.
- You cannot change the type of a column that has foreign key references unless you drop the foreign key constraint first.

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';
```

Results | Explain | Describe | Saved SQL | History

```
Statement processed.
```

9. Rename the o_jobs table to o_job_description.

```
RENAME o_jobs TO o_job_description;
```

Results | Explain | Describe | Saved SQL | History

```
Statement processed.
```

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;
```

Results | Explain | Describe | Saved SQL | History

```
Table created.
```

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

```
DESCRIBE copy_f_staffs;
```

Results | Explain | Describe | Saved SQL | History

Describe

Object Type **TABLE** Object **COPY_F_STAFFS**

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

c. Drop the table.

```
DROP TABLE copy_f_staffs;
```

Results | Explain | Describe | Saved SQL | His

```
Table dropped.
```

d. Try to select from the table.

```
SELECT * FROM copy_f_staffs;
```

Results | Explain | Describe | Saved SQL | History

```
Error at line 1/15: ORA-00942: table or view does not exist
```

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

```
SELECT * FROM USER_RECYCLEBIN;
```

Results | Explain | Describe | Saved SQL | History

OBJECT_NAME	ORIGINAL_NAME	OPERATION	TYPE	TS_NAME
BIN$I2C7llpqkmLgYxJ+eGQmcA==$0	SYS_C0011917342	DROP	INDEX	IACADEMY_1509
BIN$I2C7llprkmLgYxJ+eGQmcA==$0	AD_COURSES	DROP	TABLE	IACADEMY_1509
BIN$I2C7llpukmLgYxJ+eGQmcA==$0	SYS_C0011917344	DROP	INDEX	IACADEMY_1509
BIN$I2C7llpvkmLgYxJ+eGQmcA==$0	AD_DEPARTMENTS	DROP	TABLE	IACADEMY_1509

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;
```

Results | Explain | Describe | Saved SQL | History

```
Statement processed.
```

h. Describe the table.

Object Type **TABLE** Object **COPY_F_STAFFS**

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

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.

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

Results | Explain | Describe | Saved SQL | History

```
1 row(s) updated.
```

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.

```sql
SELECT *
FROM copy_f_staffs AS OF TIMESTAMP (SYSDATE - INTERVAL '5' MINUTE);
```

Results | Explain | Describe | Saved SQL | History

ID	FIRST_NAME	LAST_NAME	BIRTHDATE	SALARY	OVERTIME_RATE	TRAINING
12	Sue	Doe	01-Jul-1980	6.75	10.25	-
9	Bob	Miller	19-Mar-1979	10	-	Grill
19	Monique	Tuttle	30-Mar-1969	60	-	-

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.

```sql
UPDATE copy_f_staffs
SET salary = 10
WHERE first_name = 'Sue' AND last_name = 'Doe';
```

Results | Explain | Describe | Saved SQL | History

```
1 row(s) updated.
```