Name: Danny Chen
Assignment: Database Programming: Sections 12, 13
Date: 11/08/24

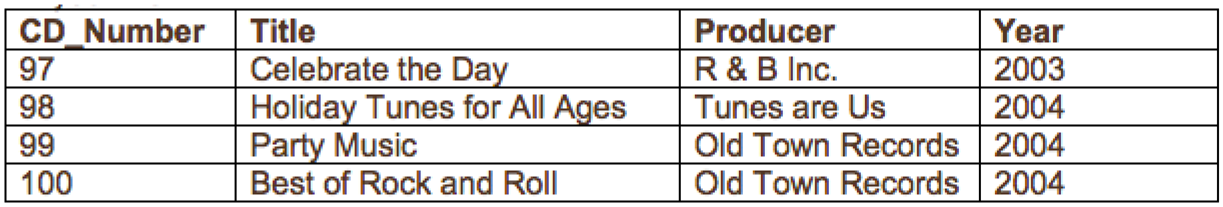
Table of Content

12.1 - INSERT Statements
Vocab

Word	Definition
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 INTO	Adds a new row to a table

1. Give two examples of why it is important to be able to alter the data in a database.
    a. **ANS:** Being able to alter data in a database means that e-commerce companies are able to keep their inventory up to date, which ensures a smooth online shopping experience for customers. Student information is always changing annually with new students attending, existing students advancing to new grades, and course availability changing. Keeping this information up to date is essential to ensure schools operate efficiently such as in the enrollment process, scheduling, and calling emergency contacts.

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

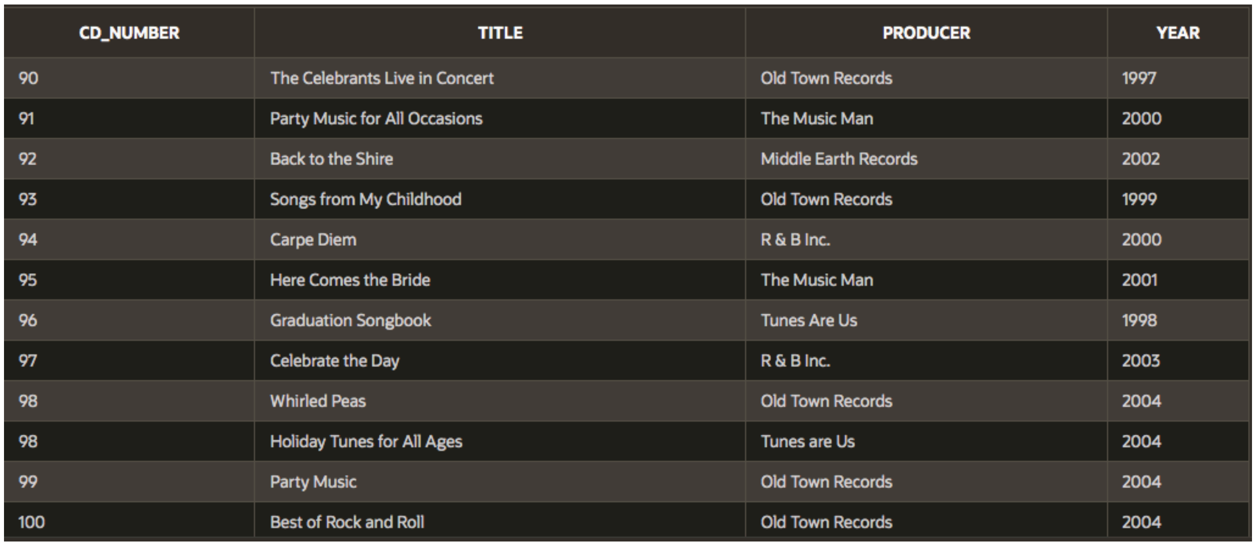
ANS:

```
INSERT INTO copy_d_cds
   (cd_number, title, producer, year)
VALUES
   X (NOTE: each statement below was ran in place of X one at a time)
```

List of X ran one at a time

- (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');

```
SELECT *
FROM copy_d_cds
ORDER BY cd_number;
```



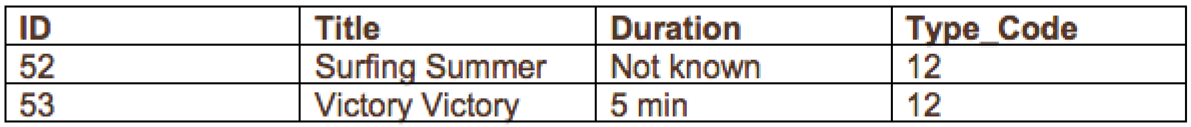
CD_NUMBER	TITLE	PRODUCER	YEAR
90	The Celebrants Live in Concert	Old Town Records	1997
91	Party Music for All Occasions	The Music Man	2000
92	Back to the Shire	Middle Earth Records	2002
93	Songs from My Childhood	Old Town Records	1999
94	Carpe Diem	R & B Inc.	2000
95	Here Comes the Bride	The Music Man	2001
96	Graduation Songbook	Tunes Are Us	1998
97	Celebrate the Day	R & B Inc.	2003
98	Whirled Peas	Old Town Records	2004
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

NOTE: copy_d_cds was made from running the below:

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

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.



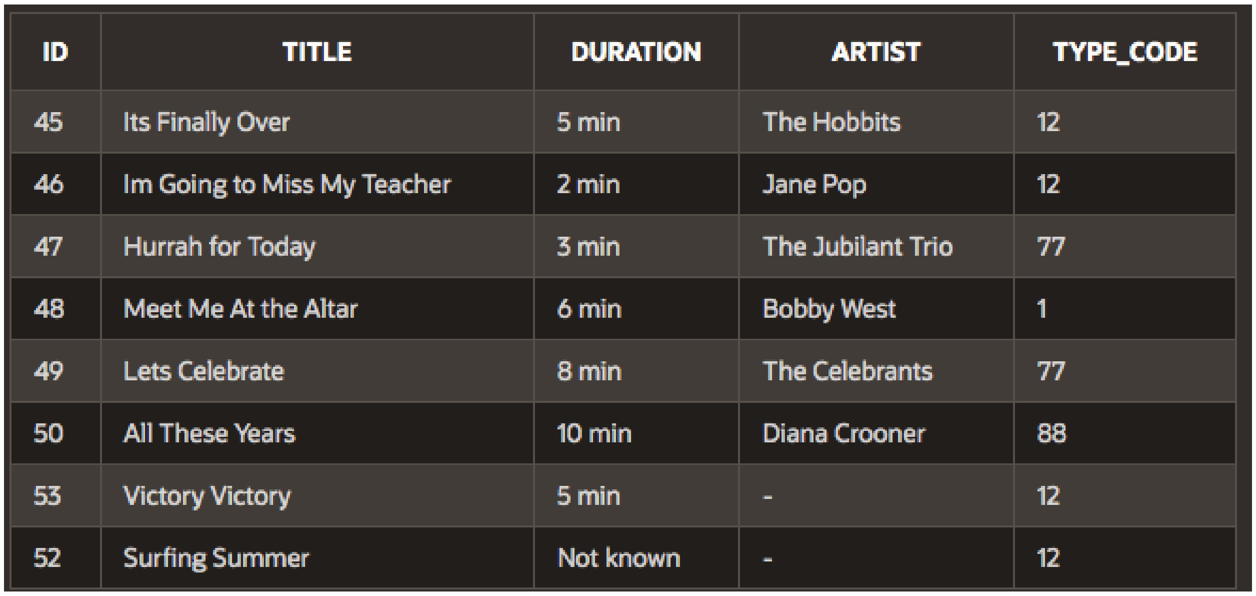
ID	Title	Duration	Type_Code
52	Surfing Summer	Not known	12
53	Victory Victory	5 min	12

ANS:
INSERT INTO copy_d_songs
VALUES
X

List of X ran one at a time

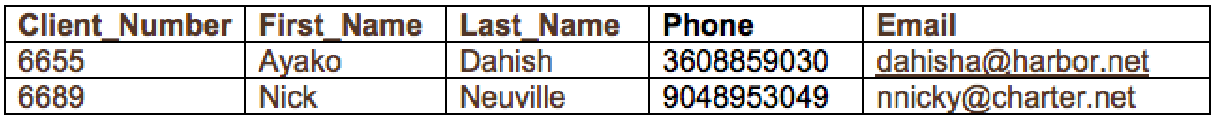
- (52, 'Surfing Summer', 'Not known', '', 12)
- (53, 'Victory Victory', '5 min', '', 12)

SELECT *
FROM copy_d_songs;



ID	TITLE	DURATION	ARTIST	TYPE_CODE
45	Its Finally Over	5 min	The Hobbits	12
46	Im Going to Miss My Teacher	2 min	Jane Pop	12
47	Hurrah for Today	3 min	The Jubilant Trio	77
48	Meet Me At the Altar	6 min	Bobby West	1
49	Lets Celebrate	8 min	The Celebrants	77
50	All These Years	10 min	Diana Crooner	88
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

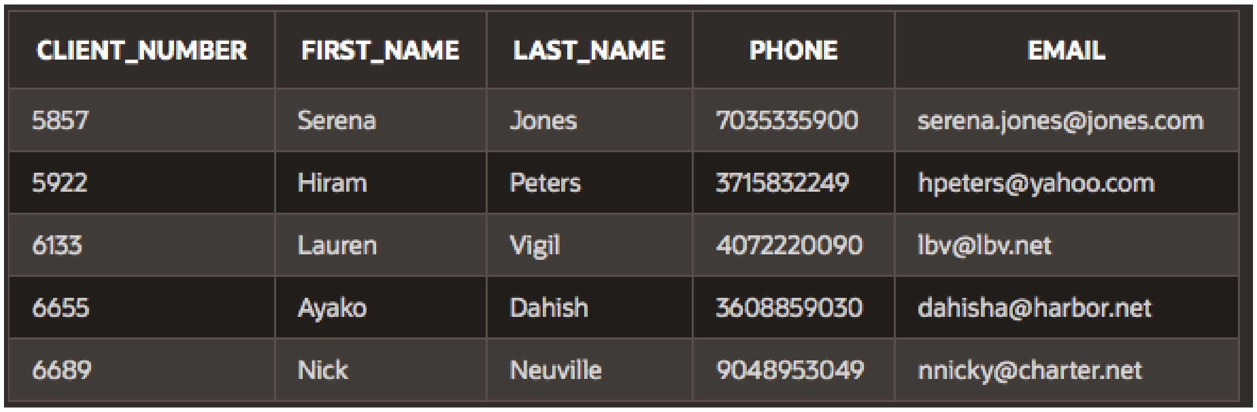
ANS:
INSERT INTO copy_d_clients
(client_number, first_name, last_name, phone, email)

```
VALUES
```

List of X ran one at a time

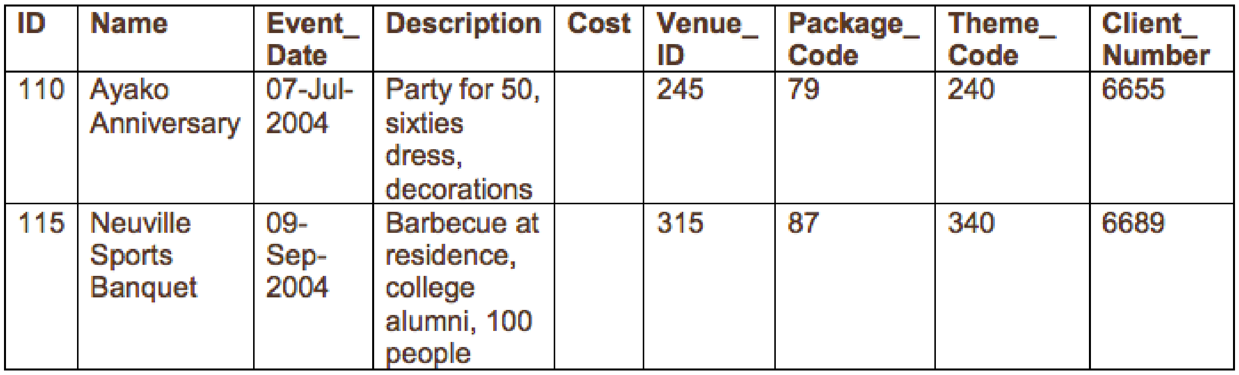
- (6655, 'Ayako', 'Dahish', 3608859030, 'dahisha@harbor.net')
- (6689, 'Nick', 'Neuville', 9048953049, 'nnicky@charter.net')

```
SELECT *
FROM copy_d_clients
ORDER BY client_number;
```



CLIENT_NUMBER	FIRST_NAME	LAST_NAME	PHONE	EMAIL
5857	Serena	Jones	7035335900	serena.jones@jones.com
5922	Hiram	Peters	3715832249	hpeters@yahoo.com
6133	Lauren	Vigil	4072220090	lbv@lbv.net
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

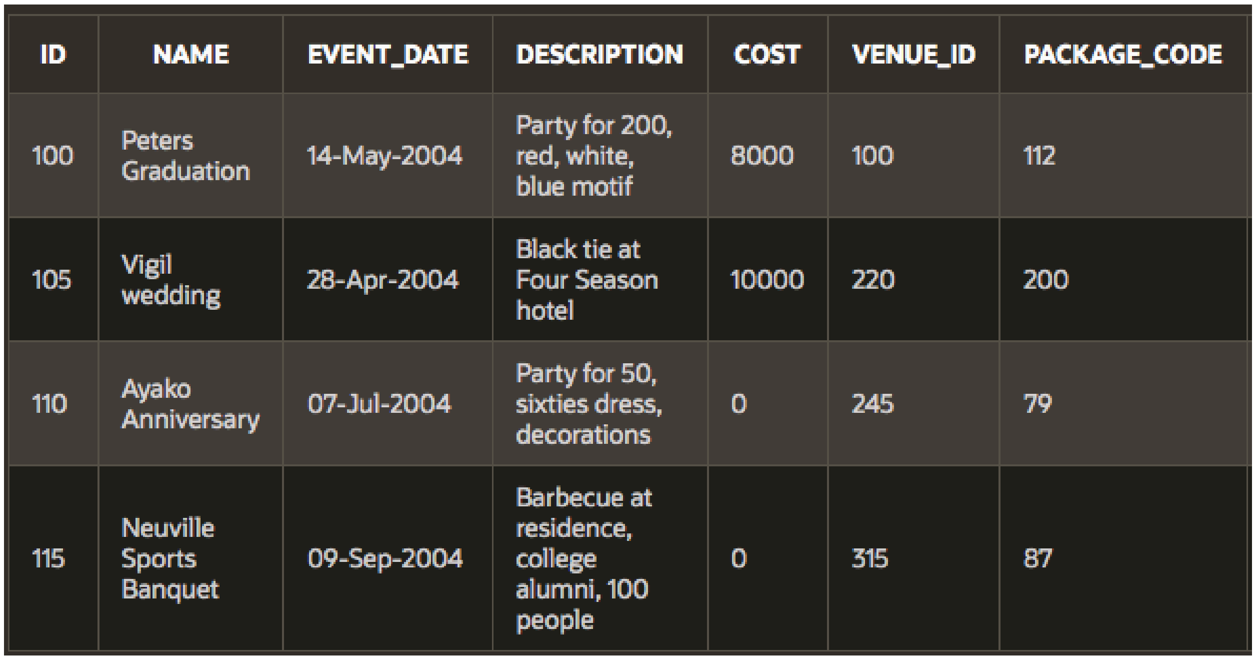
ANS:

```
INSERT INTO copy_d_events (
   id,
   name,
   event_date,
   description,
   cost,
   venue_id,
   package_code,
   theme_code,
   client_number
)
VALUES
   X
```

List of X ran one at a time (NOTE: since the cost column is not nullable, 0 was put as a placeholder)

- (110, 'Ayako Anniversary', '07-Jul-2004', 'Party for 50, sixties dress, decorations', 0, 245, 79, 240, 6655)
- (115, 'Neuville Sports Banquet', '09-Sep-2004', 'Barbecue at residence, college alumni, 100 people', 0, 315, 7, 340, 6689)

```
SELECT *
FROM copy_d_events
ORDER BY id;
```



ID	NAME	EVENT_DATE	DESCRIPTION	COST	VENUE_ID	PACKAGE_CODE
100	Peters Graduation	14-May-2004	Party for 200, red, white, blue motif	8000	100	112
105	Vigil wedding	28-Apr-2004	Black tie at Four Season hotel	10000	220	200
110	Ayako Anniversary	07-Jul-2004	Party for 50, sixties dress, decorations	0	245	79
115	Neuville Sports Banquet	09-Sep-2004	Barbecue at residence, college alumni, 100 people	0	315	87

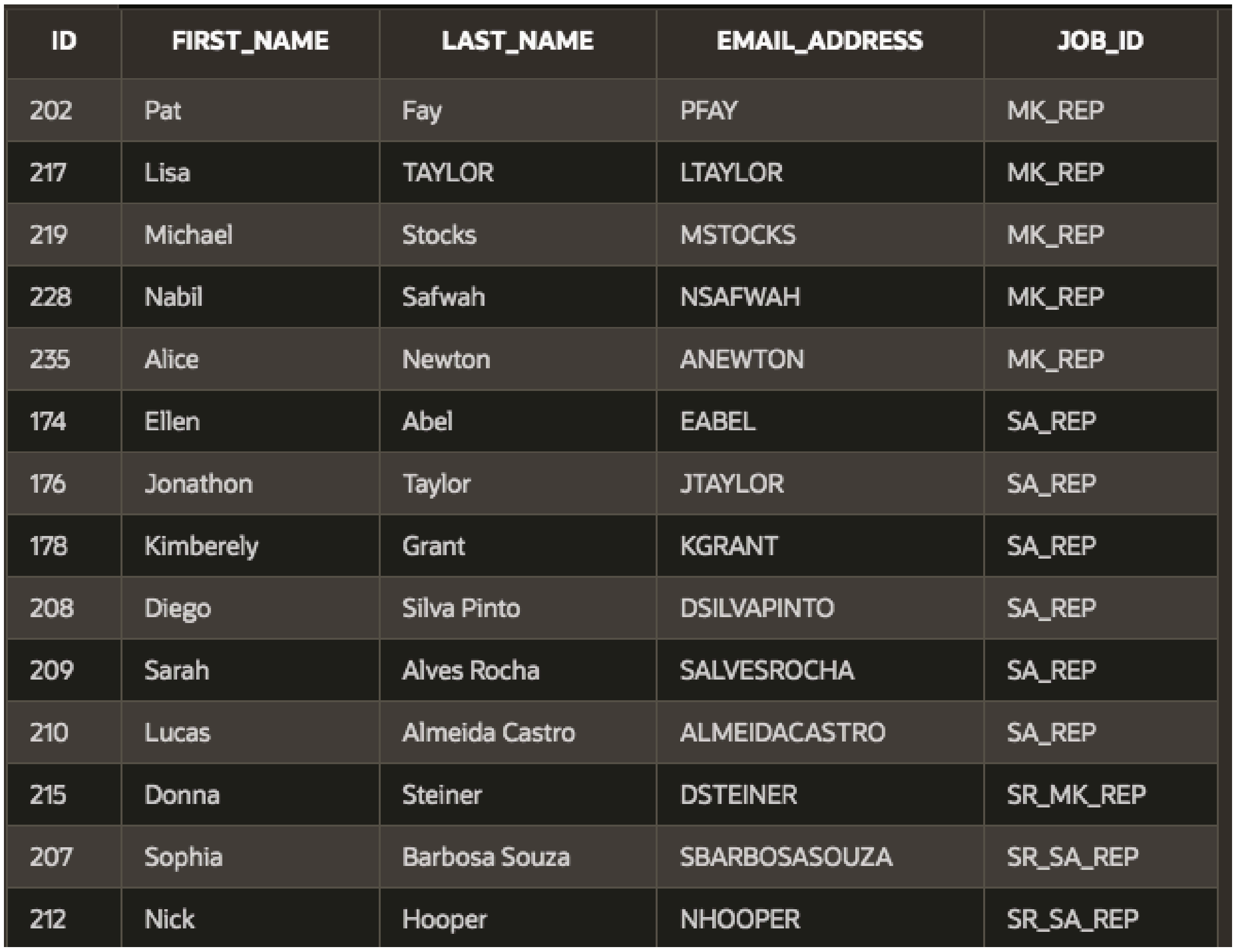
6. Create a table called rep_email using the following statement:

```
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.

ANS:

```
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%';

SELECT r.id, r.first_name, r.last_name, r.email_address, e.job_id
FROM rep_email r
INNER JOIN employees e
   ON r.id = e.employee_id;
```



ID	FIRST_NAME	LAST_NAME	EMAIL_ADDRESS	JOB_ID
202	Pat	Fay	PFAY	MK_REP
217	Lisa	TAYLOR	LTAYLOR	MK_REP
219	Michael	Stocks	MSTOCKS	MK_REP
228	Nabil	Safwah	NSAFWAH	MK_REP
235	Alice	Newton	ANEWTON	MK_REP
174	Ellen	Abel	EABEL	SA_REP
176	Jonathon	Taylor	JTAYLOR	SA_REP
178	Kimberely	Grant	KGRANT	SA_REP
208	Diego	Silva Pinto	DSILVAPINTO	SA_REP
209	Sarah	Alves Rocha	SALVESROCHA	SA_REP
210	Lucas	Almeida Castro	ALMEIDACASTRO	SA_REP
215	Donna	Steiner	DSTEINER	SR_MK_REP
207	Sophia	Barbosa Souza	SBARBOSASOUZA	SR_SA_REP
212	Nick	Hooper	NHOOPER	SR_SA_REP

NOTE: The max length specified for the field names in the CREATE TABLE query above is shorter than the corresponding field names in the employees table, leading to the error that the values are too large for the column. To resolve this, the following query was executed to modify the VARCHAR2 lengths of the rep_email table's columns to match those in the employees table:

```
ALTER TABLE rep_email
MODIFY (
   id NUMBER(6),
   first_name VARCHAR2(20),
   last_name VARCHAR2(25),
   email_address VARCHAR2(25)
)
```

12.2 - Updating Column Values and Deleting Rows
Vocab

Word	Definition
UPDATE	Modifies existing rows in a table
Correlated 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
Correlated subquery DELETE	Deletes information on a linked table based on what was deleted on the other table
DELETE	Removes existing rows from a table

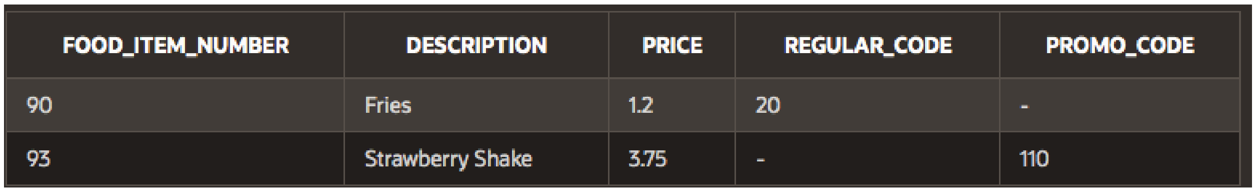
If any change is not possible, give an explanation as to why it is not possible.

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.

ANS:

```
UPDATE copy_f_food_items
SET price = 3.75
WHERE LOWER(description) = 'strawberry shake';

UPDATE copy_f_food_items
SET price = 1.20
```

WHERE LOWER(description) = 'fries';



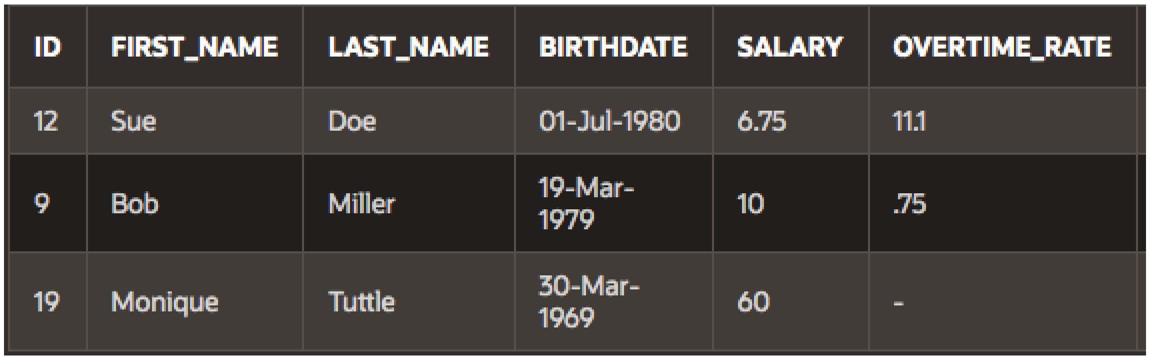
FOOD_ITEM_NUMBER	DESCRIPTION	PRICE	REGULAR_CODE	PROMO_CODE
90	Fries	1.2	20	-
93	Strawberry Shake	3.75	-	110

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?)

ANS:
For Bob Miller:
UPDATE copy_f_staffs
SET overtime_rate = NVL(overtime_rate, 0) + 0.75
WHERE LOWER(first_name) = 'bob' AND LOWER(last_name) = 'miller';

For Sue Doe:
UPDATE copy_f_staffs
SET overtime_rate = NVL(overtime_rate, 0) + 0.85
WHERE LOWER(first_name) = 'sue' AND LOWER(last_name) = 'doe';

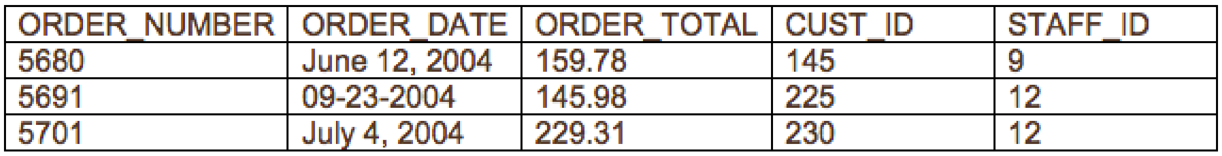
SELECT * FROM copy_f_staffs



ID	FIRST_NAME	LAST_NAME	BIRTHDATE	SALARY	OVERTIME_RATE
12	Sue	Doe	01-Jul-1980	6.75	11.1
9	Bob	Miller	19-Mar-1979	10	.75
19	Monique	Tuttle	30-Mar-1969	60	-

While I can just do overtime_rate = 0.75 for Bob Miller since his overtime rate is NULL, that only works if I am aware of that fact. Using the NVL function to change any NULL to 0 is more effective because I do not have to find out if anyone's rate is NULL since the function will just take care of that for me.

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



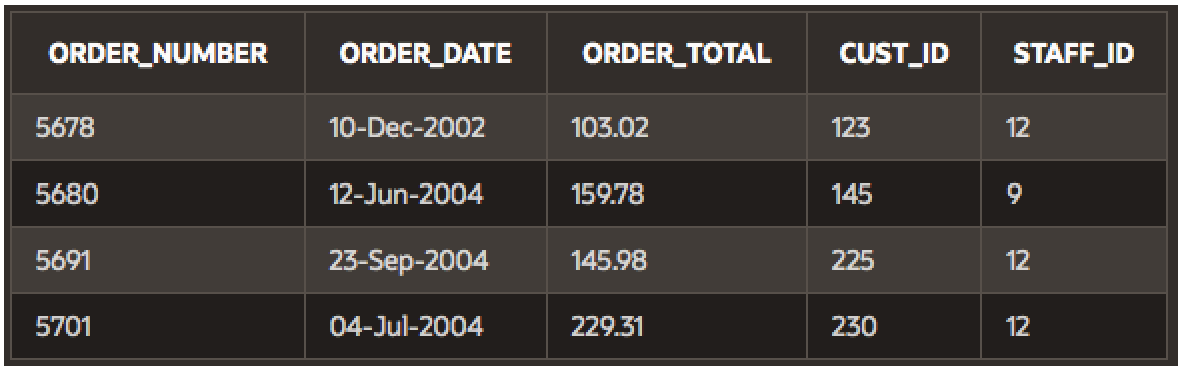
ORDER_NUMBER	ORDER_DATE	ORDER_TOTAL	CUST_ID	STAFF_ID
5680	June 12, 2004	159.78	145	9
5691	09-23-2004	145.98	225	12
5701	July 4, 2004	229.31	230	12

ANS:
INSERT INTO copy_f_orders
(order_number, order_date, order_total, cust_id, staff_id)
VALUES
X

List of X ran one at a time

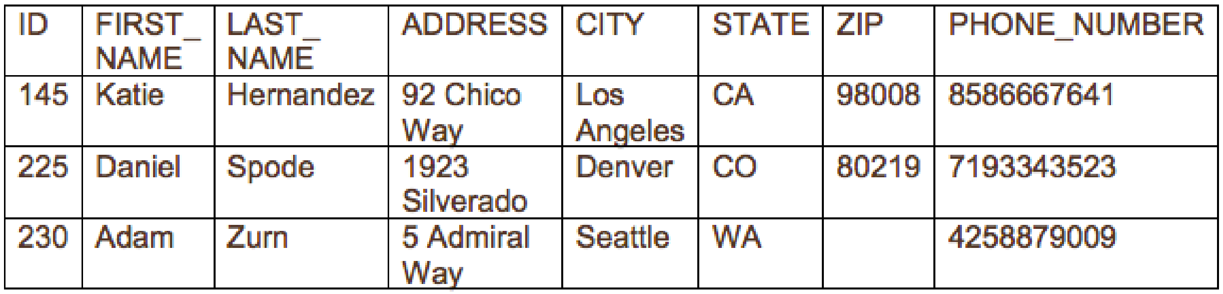
- (5680, TO_DATE('June 12, 2004', 'MONTH DD, YYYY'), 159.78, 145, 9)
- (5691, TO_DATE('09-23-2004', 'MM-DD-YYYY'), 145.98, 225, 12)
- (5701, TO_DATE('July 4, 2004', 'fmMONTH DD, YYYY'), 229.31, 230, 12)

SELECT *
FROM copy_f_orders
ORDER BY order_number



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?



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

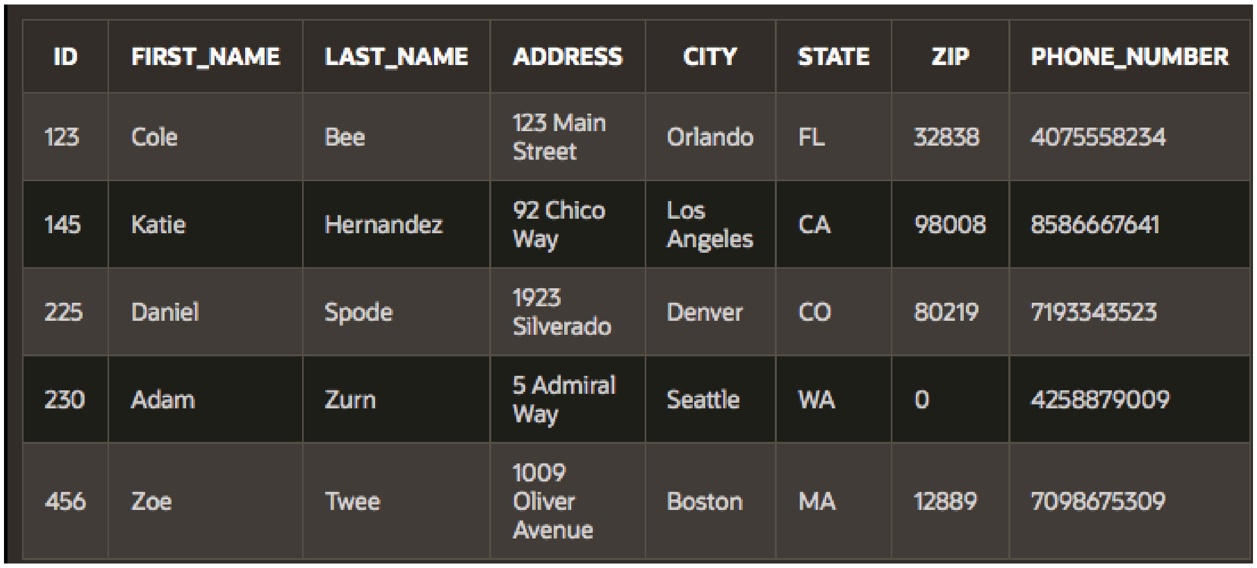
ANS:

```
INSERT INTO copy_f_customers
   (id, first_name, last_name, address, city, state, zip, phone_number)
VALUES
   X
```

List of X ran one at a time

- (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', 0, '4258879009')

```
SELECT *
FROM copy_f_customers
ORDER BY id
```



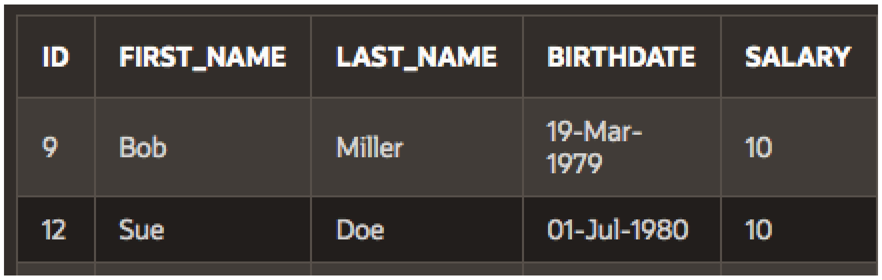
ID	FIRST_NAME	LAST_NAME	ADDRESS	CITY	STATE	ZIP	PHONE_NUMBER
123	Cole	Bee	123 Main Street	Orlando	FL	32838	4075558234
145	Katie	Hernandez	92 Chico Way	Los Angeles	CA	98008	8586667641
225	Daniel	Spode	1923 Silverado	Denver	CO	80219	7193343523
230	Adam	Zurn	5 Admiral Way	Seattle	WA	0	4258879009
456	Zoe	Twee	1009 Oliver Avenue	Boston	MA	12889	7098675309

Katie Hernandez has not been added before. Even if she was added, the duplicate record would have been added without issue because copying a table does not copy the primary key – foreign key constraints.

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.

ANS:

```
UPDATE copy_f_staffs
SET salary = (
   SELECT salary
   FROM copy_f_staffs
   WHERE LOWER(first_name) = 'bob' AND LOWER(last_name) = 'miller'
)
WHERE LOWER(first_name) = 'sue' AND LOWER(last_name) = 'doe'

SELECT *
FROM copy_f_staffs
ORDER BY id
```



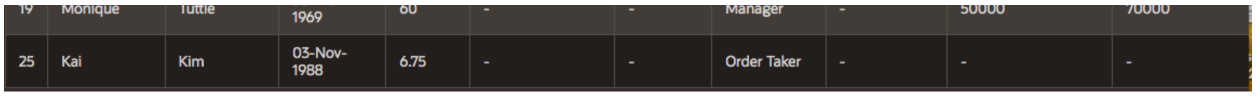
ID	FIRST_NAME	LAST_NAME	BIRTHDATE	SALARY
9	Bob	Miller	19-Mar-1979	10
12	Sue	Doe	01-Jul-1980	10

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

ANS:

```
INSERT INTO copy_f_staffs
   (id, first_name, last_name, birthdate, salary, staff_type)
VALUES
   (25, 'Kai', 'Kim', '3-Nov-1988', 6.75, 'Order Taker')
```



19	Monique	Tuttle	1969	60	-	-	Manager	-	50000	70000
25	Kai	Kim	03-Nov-1988	6.75	-	-	Order Taker	-	-	-

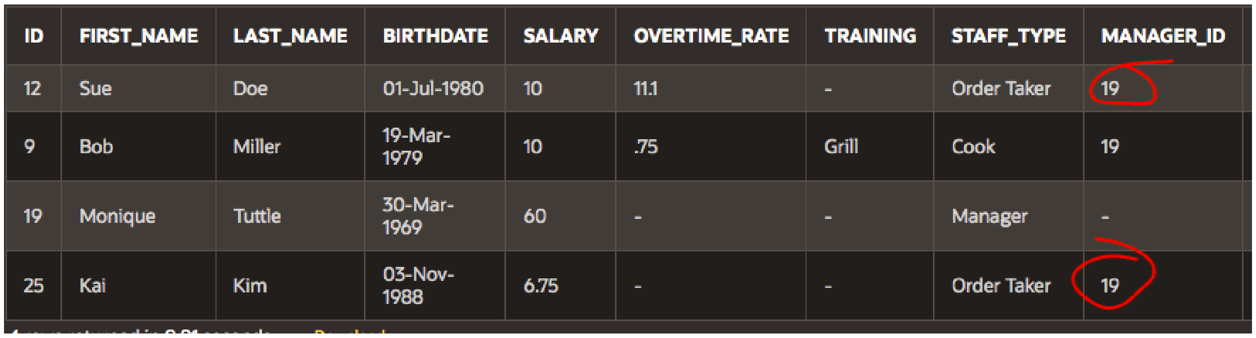
This query above works because the missing information's columns are nullable

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.

ANS:

```
UPDATE copy_f_staffs
SET manager_id = (
   SELECT manager_id
   FROM copy_f_staffs
   WHERE LOWER(first_name || ' ' || last_name) = 'sue doe'
)
```

WHERE LOWER(first_name || ' ' || last_name) = 'kai kim'



ID	FIRST_NAME	LAST_NAME	BIRTHDATE	SALARY	OVERTIME_RATE	TRAINING	STAFF_TYPE	MANAGER_ID
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	-
25	Kai	Kim	03-Nov-1988	6.75	-	-	Order Taker	19

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

```
DELETE from departments
WHERE department_id = 60;
```

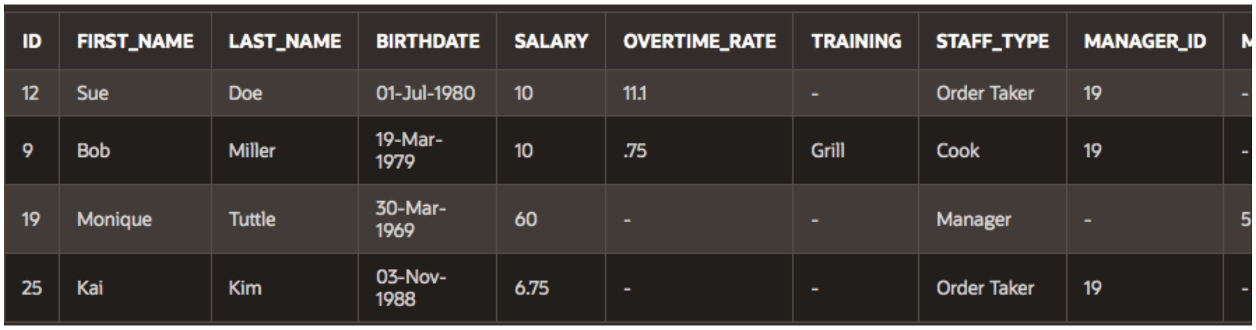
ANS:
The query above leads to an errors – "integrity constraint (US_A296_SQL_S06.JHIST_DEPT_FK) violated - child record found"

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.

ANS:

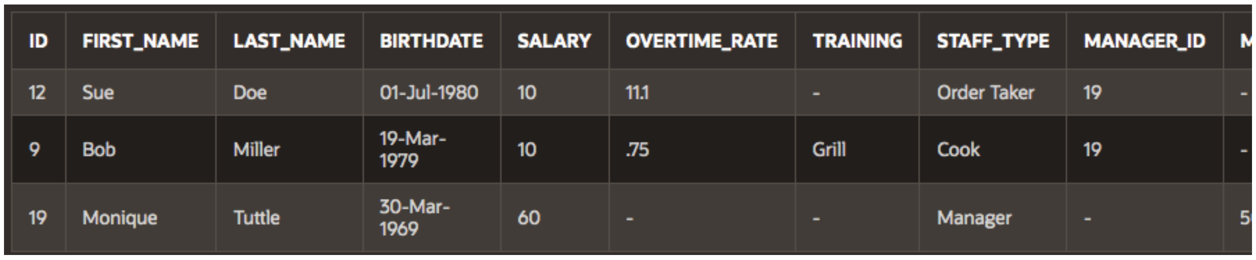
```
DELETE from copy_f_staffs
WHERE LOWER(first_name || ' ' || last_name) = 'kai kim';
```

Before:



ID	FIRST_NAME	LAST_NAME	BIRTHDATE	SALARY	OVERTIME_RATE	TRAINING	STAFF_TYPE	MANAGER_ID	M
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	-	5
25	Kai	Kim	03-Nov-1988	6.75	-	-	Order Taker	19	-

After:



ID	FIRST_NAME	LAST_NAME	BIRTHDATE	SALARY	OVERTIME_RATE	TRAINING	STAFF_TYPE	MANAGER_ID	M
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	-	5

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.

ANS:

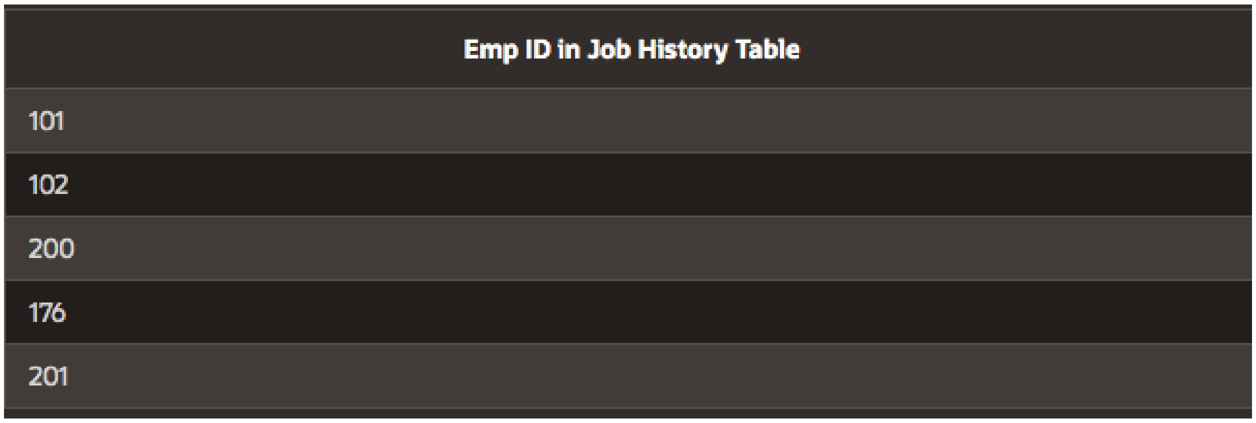
```
CREATE TABLE lesson7_emp
   AS (SELECT * FROM employees)
```

<u>Check the employee id from lesson7_emp that are in job_history table:</u>

```
SELECT e.employee_id "Emp ID in Job History Table"
FROM lesson7_emp e
WHERE e.employee_id IN (
   SELECT h.employee_id
   FROM job_history h
   WHERE h.employee_id = e.employee_id
);
```

OR

```
SELECT employee_id "Emp ID in Job History Table"
FROM lesson7_emp
WHERE employee_id IN (
   SELECT employee_id
   FROM job_history
);
```



Emp ID in Job History Table
101
102
200
176
201

Delete Query:

Correlated statement:

```
DELETE FROM lesson7_emp e
WHERE e.employee_id IN (
   SELECT h.employee_id
   FROM job_history h
   WHERE h.employee_id = e.employee_id
);
```

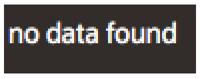
OR

Non-correlated statement:

```
DELETE FROM lesson7_emp
WHERE employee_id IN (
   SELECT DISTINCT employee_id
   FROM job_history
);
```

Verification:

```
SELECT *
FROM lesson7_emp
WHERE employee_id IN (101, 102, 200, 176, 201)
```



```
no data found
```

12.3 - DEFAULT Values, MERGE, and Multi-Table Inserts

1. When would you want a DEFAULT value?

a. DEFAULT values are useful when there is a column that is not nullable or if it is preferred there be a default value instead of a blank cell.

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)
      ```

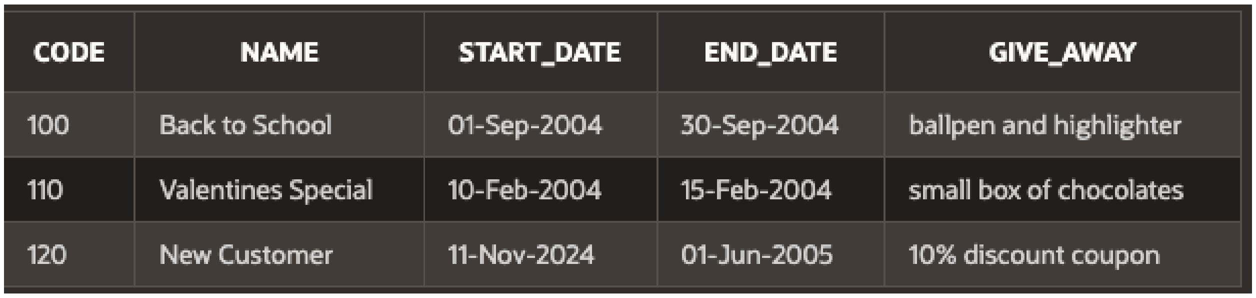
   b. Alter the current START_DATE column attributes using:

      ```
      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?

ANS:

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



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	11-Nov-2024	01-Jun-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?

ANS:

```
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?

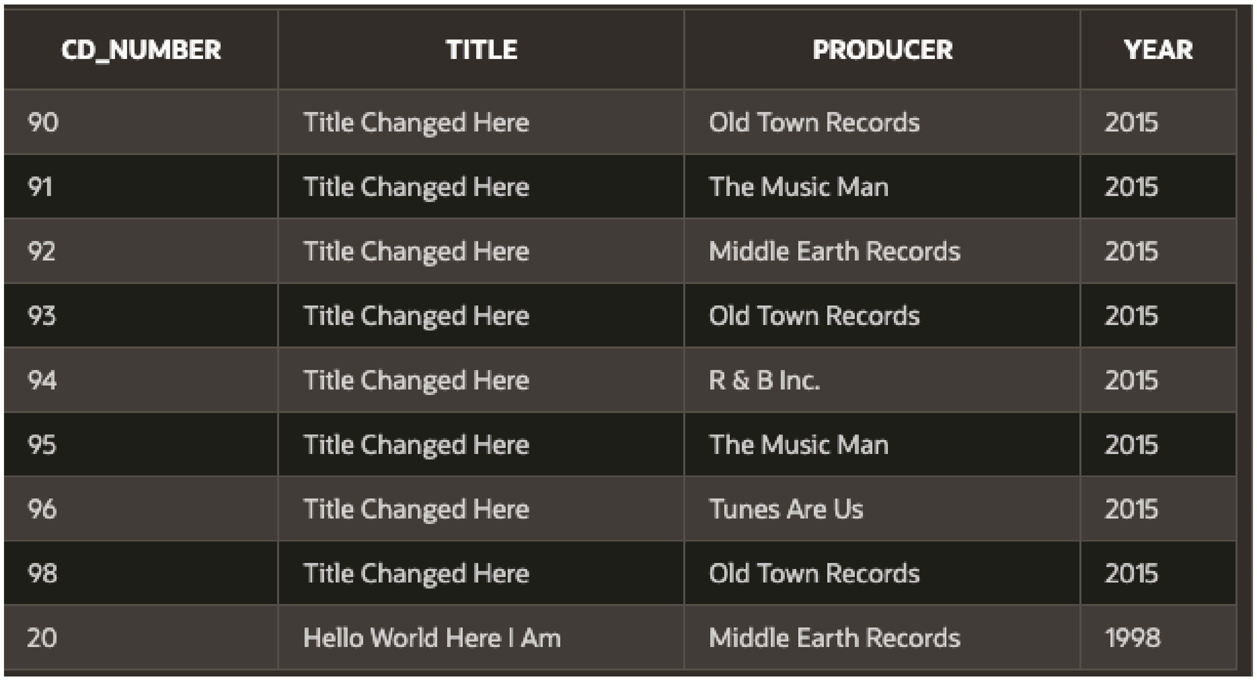
ANS:

```
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?

ANS:

```
MERGE INTO manager_copy_d_cds cp USING d_cds o
   ON (cp.cd_number = o.cd_number)
WHEN MATCHED THEN UPDATE
   SET
      cp.title = 'Title Changed Here',
      cp.year = '2015'
WHEN NOT MATCHED THEN INSERT
   VALUES (o.cd_number, o.title, o.producer, o.year)
```



CD_NUMBER	TITLE	PRODUCER	YEAR
90	Title Changed Here	Old Town Records	2015
91	Title Changed Here	The Music Man	2015
92	Title Changed Here	Middle Earth Records	2015
93	Title Changed Here	Old Town Records	2015
94	Title Changed Here	R & B Inc.	2015
95	Title Changed Here	The Music Man	2015
96	Title Changed Here	Tunes Are Us	2015
98	Title Changed Here	Old Town Records	2015
20	Hello World Here I Am	Middle Earth Records	1998

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

ANS:

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

13.1 - Creating Tables

Word	Definition
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

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	Primary key			Foreign key	
Nulls/Unique	Unique	Not nullable	Not nullable	Not nullable	Nullable
FK Column				credits	
Datatype	NUMBER	VARCHAR2	VARCHAR2	NUMBER	DATE
Length	6	30	30	3	

2. Write the syntax to create the grad_candidates table.

ANS:

```
CREATE TABLE grad_candidates (
   student_id NUMBER(6) NOT NULL,
   last_name VARCHAR2(30) NOT NULL,
   first_name VARCHAR2(30) NOT NULL,
   credits NUMBER(3) NOT NULL,
   grad_date DATE,
   CONSTRAINT student_pk PRIMARY KEY (student_id),
   CONSTRAINT credits_fk FOREIGN KEY (credits) REFERENCES requirements(credits)
);
```

NOTE: I created a table called requirements with "credits" as a primary key. Assigning a foreign key seems to require that referenced column to be uniquely indexed (primary key or unique constraint).

```
CREATE TABLE requirements (
   credits NUMBER(5) PRIMARY KEY
);
```

3. Confirm creation of the table using DESCRIBE.

ANS:

```
DESCRIBE grad_candidates
```

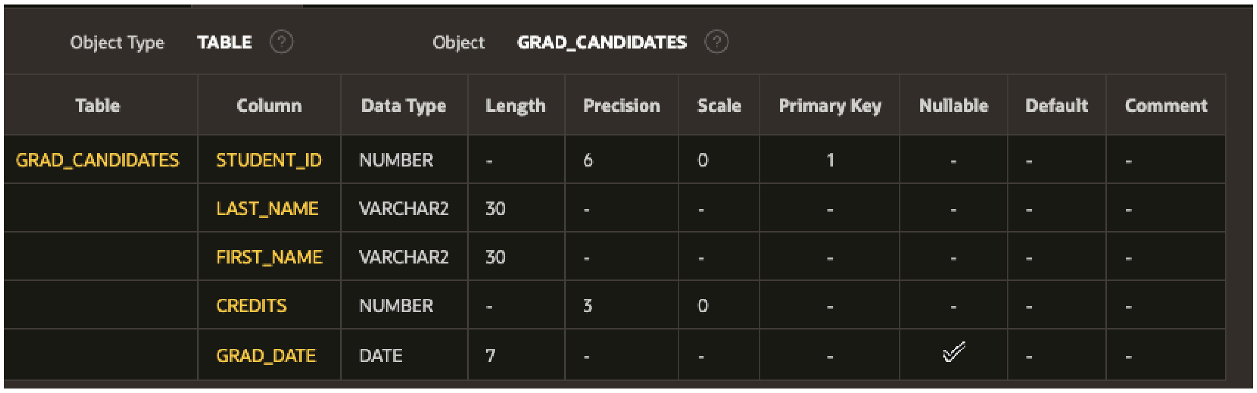

Object Type **TABLE** Object **GRAD_CANDIDATES**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
GRAD_CANDIDATES	STUDENT_ID	NUMBER	-	6	0	1	-	-	-
	LAST_NAME	VARCHAR2	30	-	-	-	-	-	-
	FIRST_NAME	VARCHAR2	30	-	-	-	-	-	-
	CREDITS	NUMBER	-	3	0	-	-	-	-
	GRAD_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.

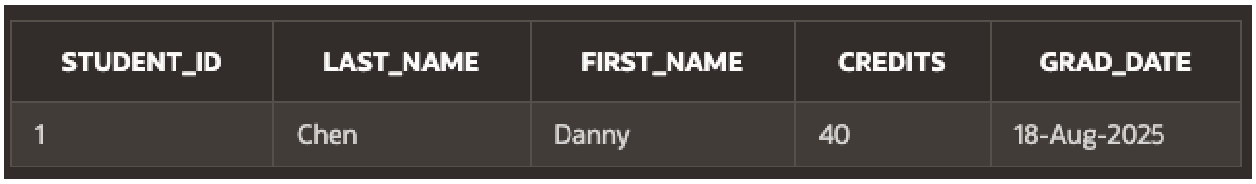
ANS:

```
CREATE TABLE chen_table
   AS (SELECT * FROM grad_candidate)
```

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

ANS:

```
INSERT INTO chen_table
   (student_id, last_name, first_name, credits, grad_date)
VALUES
   (1, 'Chen', 'Danny', 40, TO_DATE('18-Aug-2025', 'DD-MM-YYYY'));
```

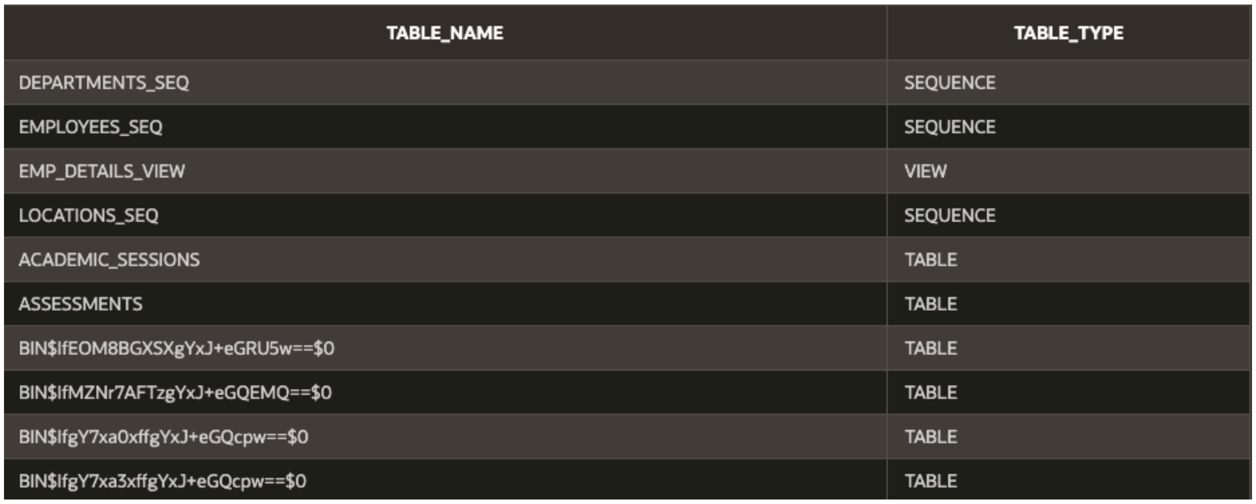


STUDENT_ID	LAST_NAME	FIRST_NAME	CREDITS	GRAD_DATE
1	Chen	Danny	40	18-Aug-2025

6. Query the data dictionary for each of the following below and in separate sentences, summarize what each query will return:
   a. USER_TABLES
      i. SELECT * FROM USER_TABLES
      ii. This query returns the table names I have and many information with columns relating to memory, cache, activity tracking, extents, freelist, and PCT
   b. USER_OBJECTS
      i. SELECT * FROM USER_OBJECTS

        ii. This query also returns the table names I have and information pertaining to objects such as their id, type, creation date, last DDL time, sharing status, app id and vsn id

    c. USER_CATALOG or USER_CAT
        i. SELECT * FROM USER_CATALOG
            1. NOTE: Running USER_CAT leads to a “table or view does not exist” error
        ii. This query returns a few of the table names that I have, but mostly tables that I have never made. It also shows the table type.



TABLE_NAME	TABLE_TYPE
DEPARTMENTS_SEQ	SEQUENCE
EMPLOYEES_SEQ	SEQUENCE
EMP_DETAILS_VIEW	VIEW
LOCATIONS_SEQ	SEQUENCE
ACADEMIC_SESSIONS	TABLE
ASSESSMENTS	TABLE
BIN$IfEOM8BGXSXgYxJ+eGRU5w==$0	TABLE
BIN$IfMZNr7AFTzgYxJ+eGQEMQ==$0	TABLE
BIN$IfgY7xa0xffgYxJ+eGQcpw==$0	TABLE
BIN$IfgY7xa3xffgYxJ+eGQcpw==$0	TABLE

13.2 - Using Data Types

Word	Definition
INTERVAL YEAR TO MONTH	Allows time to be stored as an interval of years and months
TIMESTAMP WITH LOCAL TIMEZONE	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 TIMEZONE	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

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

ANS:
```
CREATE TABLE time_zones (
   tz TIMESTAMP WITH TIME ZONE,
   local_tz TIMESTAMP WITH LOCAL TIME ZONE
)

INSERT INTO time_zones
   (tz, local_tz)
VALUES
('12-Nov-2024 11:00:00 AM -07:00', '12-Nov-2024 11:00:00');
```

    b. INTERVAL YEAR TO MONTH

ANS:
```
CREATE TABLE durations (
   restock_duration INTERVAL YEAR(2) TO MONTH,
   loan_duration INTERVAL YEAR(2) TO MONTH
);

INSERT INTO durations
   (restock_duration, loan_duration)
VALUES (
   INTERVAL '6' MONTH(2),
   INTERVAL '40-8' YEAR TO MONTH
);
```

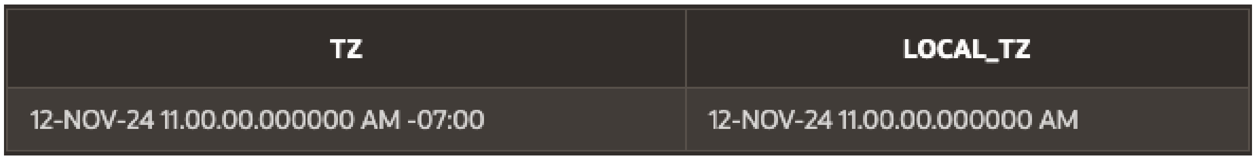
    c. INTERVAL DAY TO SECOND

ANS:
```
CREATE TABLE day_durations (
   day_duration_1 INTERVAL DAY(2) TO SECOND,
   day_duration_2 INTERVAL DAY(2) TO SECOND
);

INSERT INTO day_durations
   (day_duration_1, day_duration_2)
VALUES (
   INTERVAL '90' DAY(2),
   INTERVAL '8 05:45:20' DAY TO SECOND
);
```

2. Execute a SELECT * from each table to verify your input.
   a. TIMESTAMP WITH LOCAL TIME ZONE

ANS:

SELECT * FROM time_zones

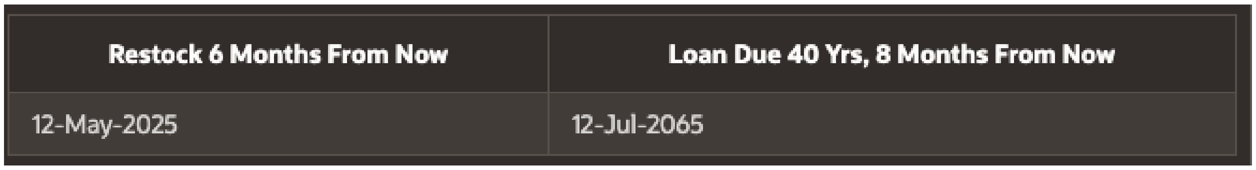


TZ	LOCAL_TZ
12-NOV-24 11.00.00.000000 AM -07:00	12-NOV-24 11.00.00.000000 AM

   b. INTERVAL YEAR TO MONTH

ANS:

```
SELECT
   SYSDATE + restock_duration "Restock 6 Months From Now",
   SYSDATE + loan_duration "Loan Due 40 Yrs, 8 Months From Now"
FROM durations
```

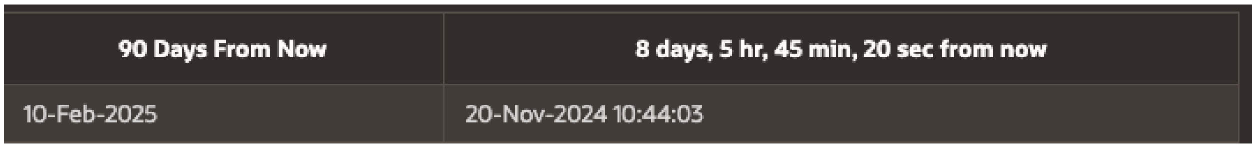


Restock 6 Months From Now	Loan Due 40 Yrs, 8 Months From Now
12-May-2025	12-Jul-2065

   c. INTERVAL DAY TO SECOND

ANS:

```
SELECT
   SYSDATE + day_duration_1 "90 Days From Now",
   TO_CHAR(SYSDATE + day_duration_2, 'dd-Mon-yyyy hh:mi:ss') "8 days, 5 hr, 45 min, 20
sec from now"
FROM day_durations
```



90 Days From Now	8 days, 5 hr, 45 min, 20 sec from now
10-Feb-2025	20-Nov-2024 10:44:03

3. Give 3 examples of organizations and personal situations where it is important to know to which time zone a date-time value refers.
   ANS:
   a. A company needs to contact another company from overseas and needs to see if they're open yet in their time.
   b. I want to call a friend from a country or state in another time zone, but it might be too early or late to call them although that's not the case in my own time zone.
   c. Friends figuring out what time they should pick for their airplane ticket because the concert they're going to is in another time zone and they need to make sure they arrive on time.

13.3 - Modifying a Table

1. Why is it important to be able to modify a table?
   a. ANS: Modifying tables is important because nothing is ever truly permanent. Things may change, so the ability to modify tables will be essential to ensure operations flow smoothly.
2. CREATE a table called Artists
   a. Add the following to the table:
      i. artist ID
      ii. first name
      iii. last name
      iv. band name
      v. email
      vi. hourly rate

ANS:

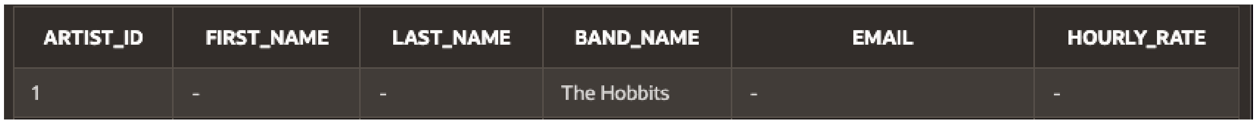
```
CREATE TABLE artists (
   artist_id NUMBER PRIMARY KEY,
   first_name VARCHAR2(30),
   last_name VARCHAR2(30),
   band_name VARCHAR2(60),
   email VARCHAR2(120),
   hourly_rate NUMBER
)
```

   b. INSERT one artist from the d_songs table.

ANS:

```
INSERT INTO artists (artist_id, band_name)
   SELECT 1, artist
   FROM d_songs
```

WHERE ROWNUM = 1

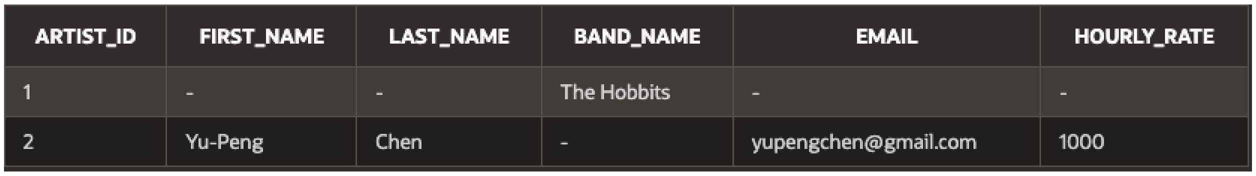


ARTIST_ID	FIRST_NAME	LAST_NAME	BAND_NAME	EMAIL	HOURLY_RATE
1	-	-	The Hobbits	-	-

c. INSERT one artist of your own choosing.

ANS:

INSERT INTO artists
  (artist_id, first_name, last_name, email, hourly_rate)
VALUES
  (2, 'Yu-Peng', 'Chen', 'yupengchen@gmail.com', '1000')



ARTIST_ID	FIRST_NAME	LAST_NAME	BAND_NAME	EMAIL	HOURLY_RATE
1	-	-	The Hobbits	-	-
2	Yu-Peng	Chen	-	yupengchen@gmail.com	1000

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

i. ALTER TABLE

ANS:

ALTER TABLE artists
ADD (phone_number VARCHAR(30))

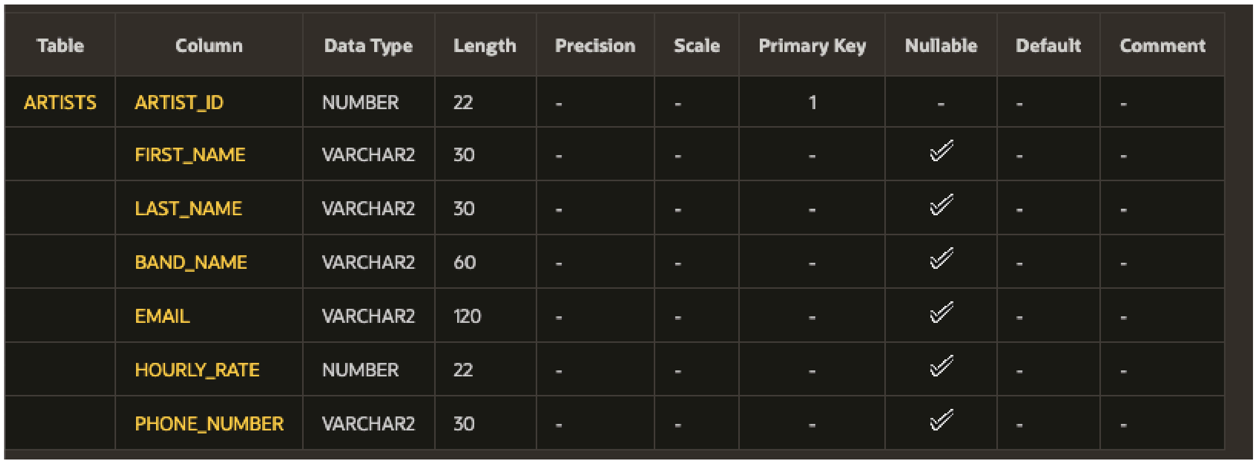
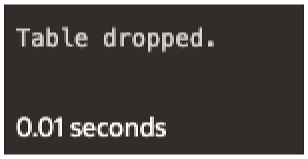


Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
ARTISTS	ARTIST_ID	NUMBER	22	-	-	1	-	-	-
	FIRST_NAME	VARCHAR2	30	-	-	-	✓	-	-
	LAST_NAME	VARCHAR2	30	-	-	-	✓	-	-
	BAND_NAME	VARCHAR2	60	-	-	-	✓	-	-
	EMAIL	VARCHAR2	120	-	-	-	✓	-	-
	HOURLY_RATE	NUMBER	22	-	-	-	✓	-	-
	PHONE_NUMBER	VARCHAR2	30	-	-	-	✓	-	-

ii. DROP TABLE

ANS:

DROP TABLE copy_artists

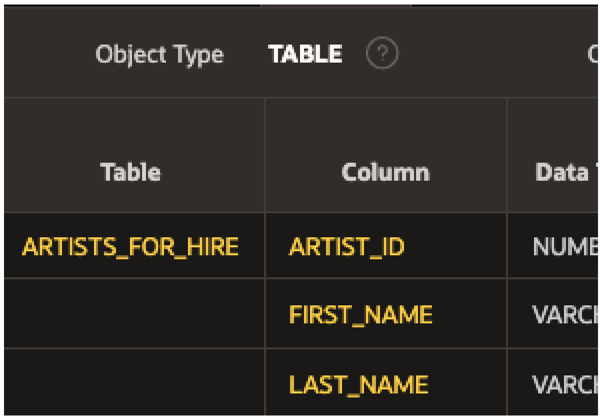


```
Table dropped.

0.01 seconds
```

NOTE: A copy of the artists table was made since dropping the tables deletes it
CREATE TABLE copy_artists
  AS (SELECT * FROM artists)

iii. RENAME TABLE

ANS:
RENAME artists TO artists_for_hire



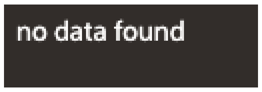
Object Type **TABLE**

Table	Column	Data
ARTISTS_FOR_HIRE	ARTIST_ID	NUME
	FIRST_NAME	VARC
	LAST_NAME	VARC

iv. TRUNCATE

ANS:
TRUNCATE TABLE artists

Running the query, SELECT * FROM artists, results in no data found

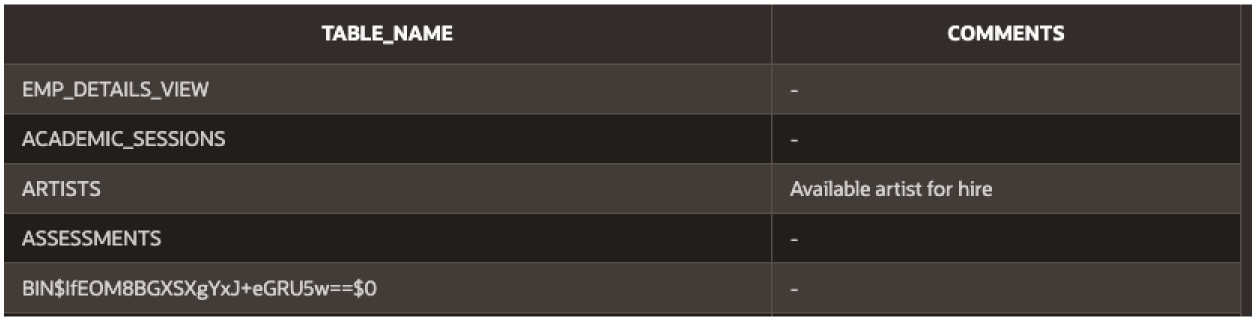


```
no data found
```

v. COMMENT ON TABLE

ANS:
COMMENT ON TABLE artists
  IS 'Available artist for hire'

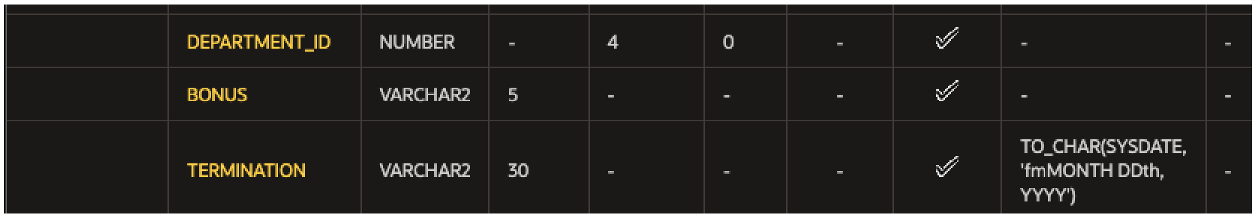
SELECT table_name, comments
FROM user_tab_comments;



TABLE_NAME	COMMENTS
EMP_DETAILS_VIEW	-
ACADEMIC_SESSIONS	-
ARTISTS	Available artist for hire
ASSESSMENTS	-
BIN$lfEOM8BGXSXgYxJ+eGRU5w==$0	-

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

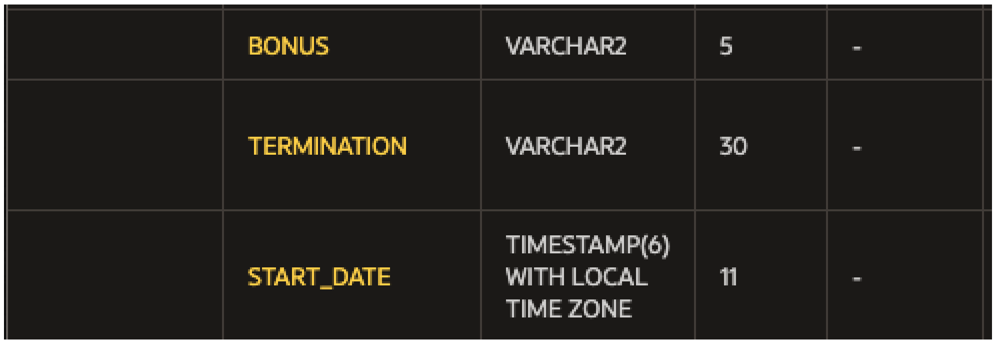
ANS:
ALTER TABLE o_employees
ADD (termination VARCHAR2(30) DEFAULT TO_CHAR(SYSDATE, 'fmMONTH DDth, YYYY'))



	DEPARTMENT_ID	NUMBER	-	4	0	-	✓	-	-
	BONUS	VARCHAR2	5	-	-	-	✓	-	-
	TERMINATION	VARCHAR2	30	-	-	-	✓	TO_CHAR(SYSDATE, 'fmMONTH 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

ANS:
ALTER TABLE o_employees
ADD (start_date TIMESTAMP WITH LOCAL TIME ZONE)

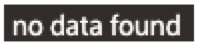


	BONUS	VARCHAR2	5	-
	TERMINATION	VARCHAR2	30	-
	START_DATE	TIMESTAMP(6) WITH LOCAL TIME ZONE	11	-

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

ANS:

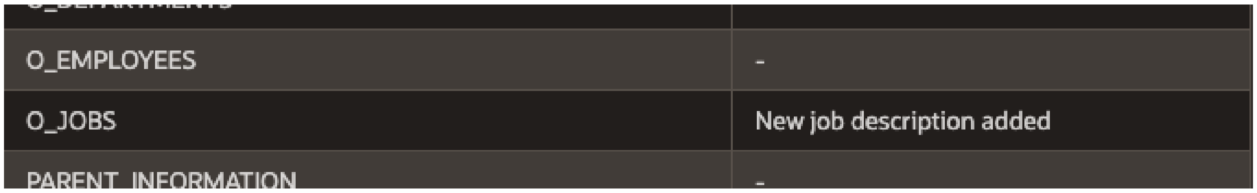
```
TRUNCATE TABLE o_jobs;
SELECT * FROM o_jobs;
```



```
no data found
```

6. What is the distinction between TRUNCATE, DELETE, and DROP for tables?
   ANS:
   a. TRUNCATE removes all the rows in a table and releases the storage space used by that table. It is faster than DELETE because it does not generate rollback information.
   b. DELETE removes all the rows in a table, but does NOT release storage space
   c. DROP deletes the entire table, not just the rows. All data is deleted from the table and the table's description is removed from the Data Dictionary.
7. List the changes that can and cannot be made to a column
   ANS:
   a. Can be altered without restrictions
      i. Increase width and precision of numeric column
      ii. Increase width of character column
   b. Can be altered, but with restrictions
      i. NUMBER column → can only decrease width if the entire column contains NULL or if the table has no rows
      ii. VARCHAR columns → can decrease width down to the largest existing value in the column
      iii. Changing the data type itself → can only be done if the column has ONLY NULL values
      iv. CHAR can be converted to VARCHAR2 or vice versa if the column only has NULL values or if the size is not changed to something smaller than any existing column value
      v. DEFAULT values can be changed, but only affects later insertions to the table
8. Add the following comment to the o_jobs table:
   a. "New job description added"
   ANS:
```
COMMENT ON TABLE o_jobs
   IS 'New job description added'
```

   b. View the data dictionary to view your comments
   ANS:
```
SELECT table_name, comments
FROM user_tab_comments
```



O_EMPLOYEES	-
O_JOBS	New job description added
PARENT_INFORMATION	-

9. Rename the o_jobs table to o_job_description

ANS:

RENAME o_jobs TO o_job_description;

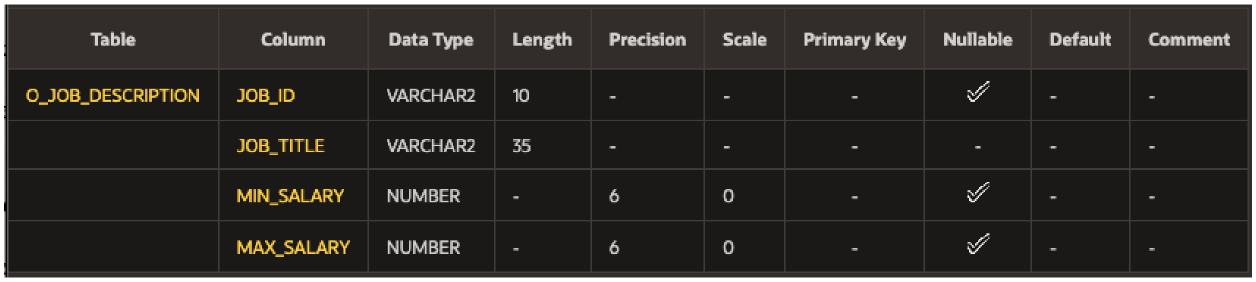


Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
O_JOB_DESCRIPTION	JOB_ID	VARCHAR2	10	-	-	-	✓	-	-
	JOB_TITLE	VARCHAR2	35	-	-	-	-	-	-
	MIN_SALARY	NUMBER	-	6	0	-	✓	-	-
	MAX_SALARY	NUMBER	-	6	0	-	✓	-	-

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
        i. **ANS:** CREATE TABLE copy_f_staffs AS (SELECT * FROM f_staffs)
    b. Describe the new table to make sure it exists
        i. **ANS:** DESCRIBE copy_f_staffs

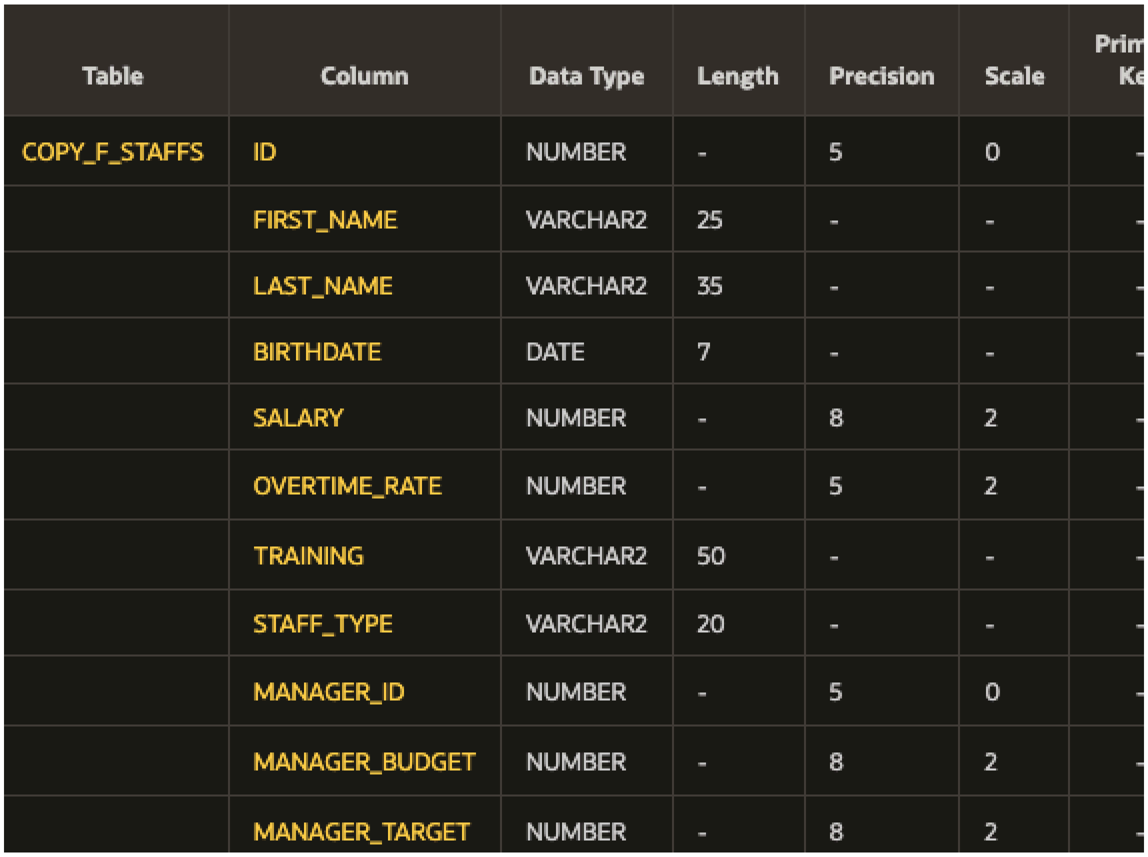
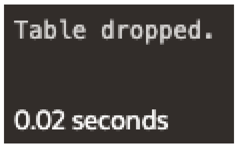


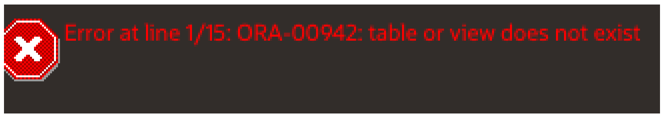
Table	Column	Data Type	Length	Precision	Scale	Prim Ke
COPY_F_STAFFS	ID	NUMBER	-	5	0	-
	FIRST_NAME	VARCHAR2	25	-	-	-
	LAST_NAME	VARCHAR2	35	-	-	-
	BIRTHDATE	DATE	7	-	-	-
	SALARY	NUMBER	-	8	2	-
	OVERTIME_RATE	NUMBER	-	5	2	-
	TRAINING	VARCHAR2	50	-	-	-
	STAFF_TYPE	VARCHAR2	20	-	-	-
	MANAGER_ID	NUMBER	-	5	0	-
	MANAGER_BUDGET	NUMBER	-	8	2	-
	MANAGER_TARGET	NUMBER	-	8	2	-

c. Drop the table
   i. **ANS:** DROP TABLE copy_f_staffs



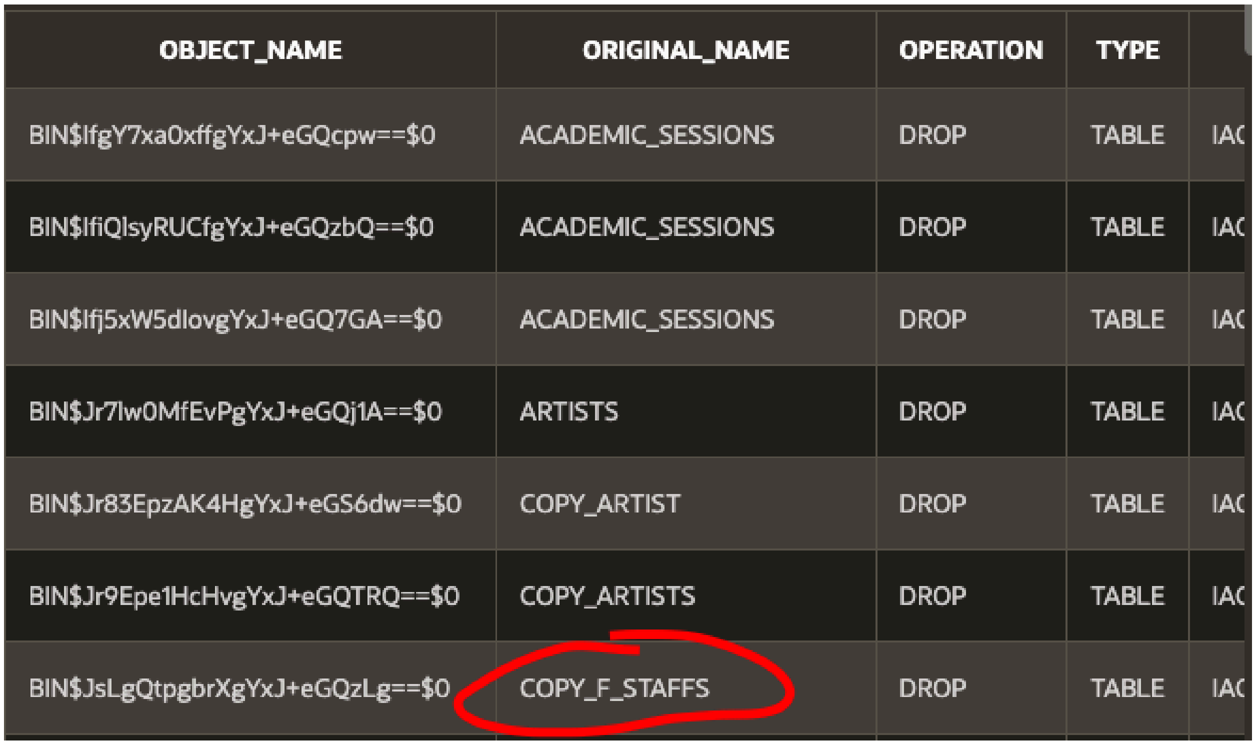
```
Table dropped.

0.02 seconds
```

d. Try to select from the table
   i. **ANS:** SELECT * FROM copy_f_staff



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

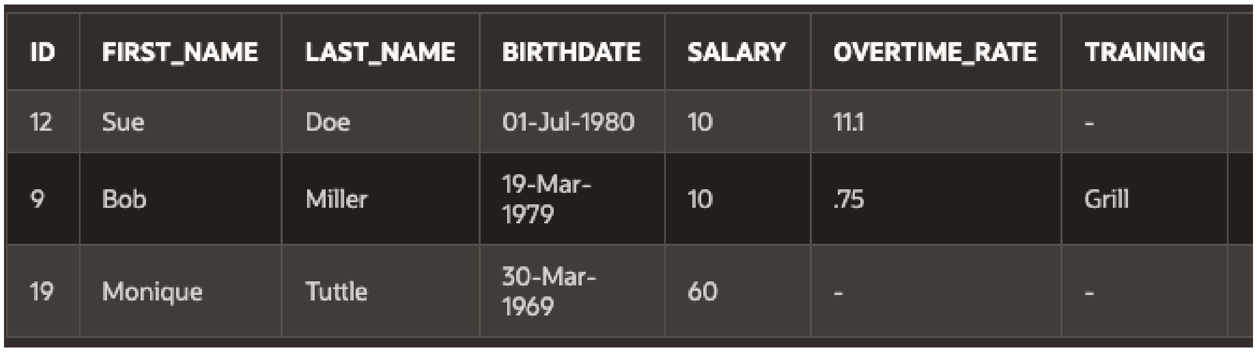
e. Investigate your recycle bin to see where the table went
   i. **ANS:**
      SELECT original_name, operation, droptime
      FROM user_recyclebin
      ORDER BY original_name



OBJECT_NAME	ORIGINAL_NAME	OPERATION	TYPE	
BIN$IfgY7xa0xffgYxJ+eGQcpw==$0	ACADEMIC_SESSIONS	DROP	TABLE	IA(
BIN$IfiQlsyRUCfgYxJ+eGQzbQ==$0	ACADEMIC_SESSIONS	DROP	TABLE	IA(
BIN$Ifj5xW5dlovgYxJ+eGQ7GA==$0	ACADEMIC_SESSIONS	DROP	TABLE	IA(
BIN$Jr7lw0MfEvPgYxJ+eGQj1A==$0	ARTISTS	DROP	TABLE	IA(
BIN$Jr83EpzAK4HgYxJ+eGS6dw==$0	COPY_ARTIST	DROP	TABLE	IA(
BIN$Jr9Epe1HcHvgYxJ+eGQTRQ==$0	COPY_ARTISTS	DROP	TABLE	IA(
BIN$JsLgQtpgbrXgYxJ+eGQzLg==$0	COPY_F_STAFFS	DROP	TABLE	IA(

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"

ANS:
SELECT *
FROM "BIN$JsLgQtpgbrXgYxJ+eGQzLg==$0"



ID	FIRST_NAME	LAST_NAME	BIRTHDATE	SALARY	OVERTIME_RATE	TRAINING
12	Sue	Doe	01-Jul-1980	10	11.1	-
9	Bob	Miller	19-Mar-1979	10	.75	Grill
19	Monique	Tuttle	30-Mar-1969	60	-	-

g. Undrop the table
   i. **ANS:** FLASHBACK TABLE copy_f_staffs TO BEFORE DROP;

h. Describe the table
   i. **ANS:** DESCRIBE copy_f_staffs

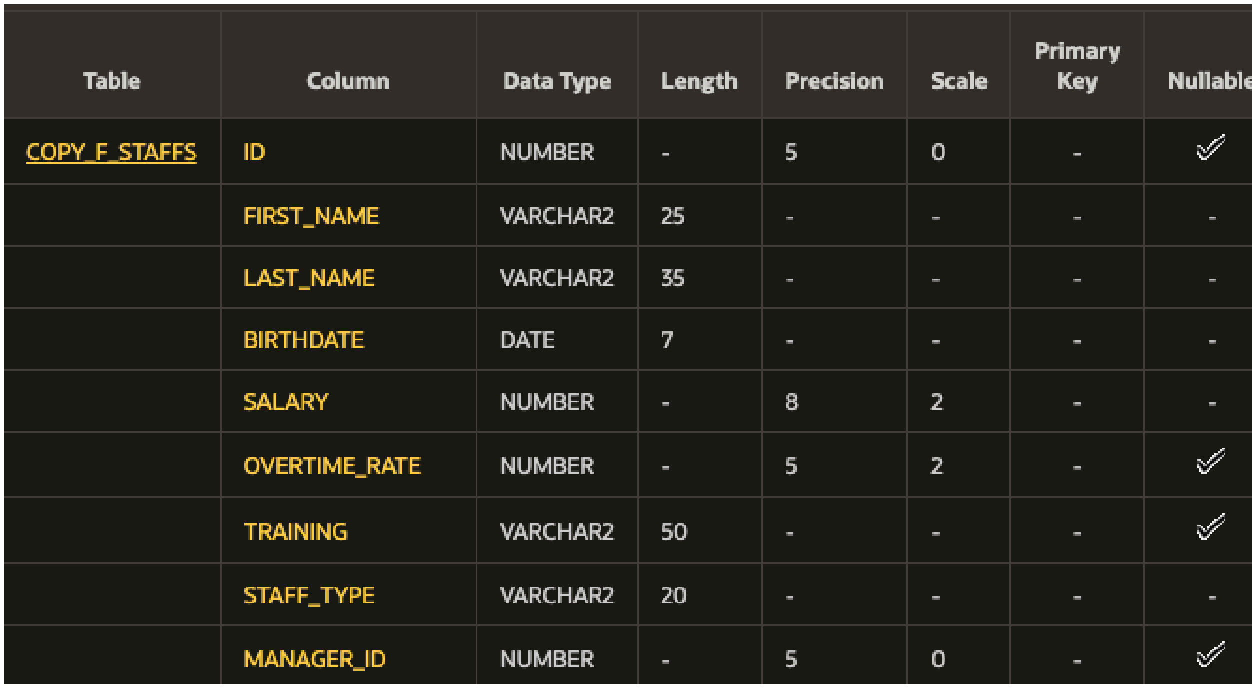
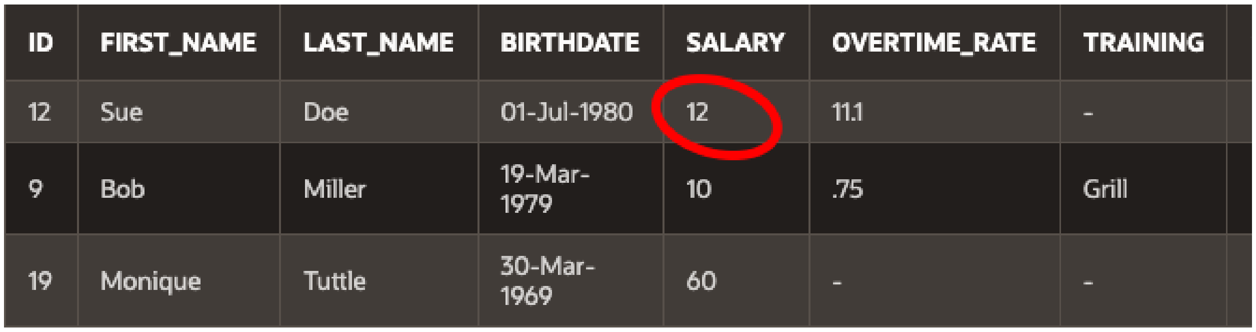


Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable
COPY_F_STAFFS	ID	NUMBER	-	5	0	-	✓
	FIRST_NAME	VARCHAR2	25	-	-	-	-
	LAST_NAME	VARCHAR2	35	-	-	-	-
	BIRTHDATE	DATE	7	-	-	-	-
	SALARY	NUMBER	-	8	2	-	-
	OVERTIME_RATE	NUMBER	-	5	2	-	✓
	TRAINING	VARCHAR2	50	-	-	-	✓
	STAFF_TYPE	VARCHAR2	20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

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
        i. **ANS:** SELECT * FROM copy_f_staffs
    b. Change the salary for Sue Doe to 12 and commit the change
        i. **ANS:**
           UPDATE copy_f_staffs
           SET salary = 12
           WHERE LOWER(first_name || ' ' || last_name) = 'sue doe';
    c. Issue a select statement to see all rows and all columns from the copy_f_staffs table
        i. **ANS:** SELECT * FROM copy_f_staffs



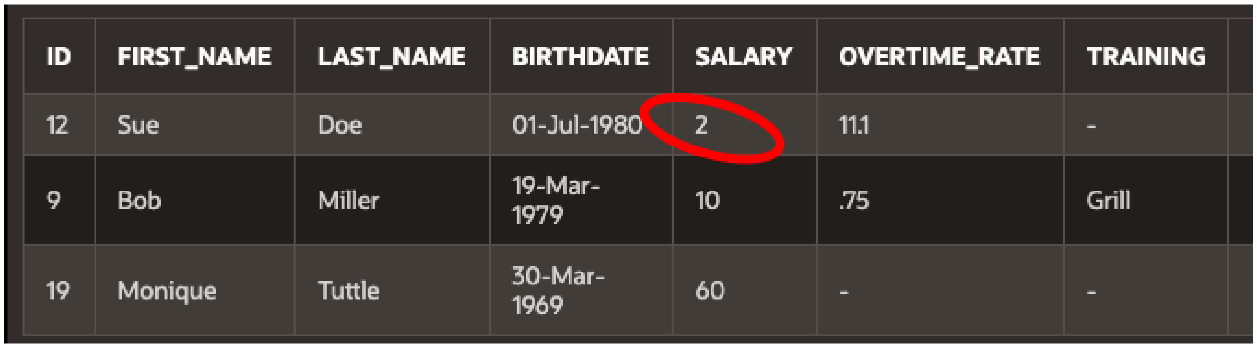
ID	FIRST_NAME	LAST_NAME	BIRTHDATE	SALARY	OVERTIME_RATE	TRAINING
12	Sue	Doe	01-Jul-1980	12	11.1	-
9	Bob	Miller	19-Mar-1979	10	.75	Grill
19	Monique	Tuttle	30-Mar-1969	60	-	-

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

i. **ANS:**
UPDATE copy_f_staffs
SET salary = 2
WHERE LOWER(first_name || ' ' || last_name) = 'sue doe';

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

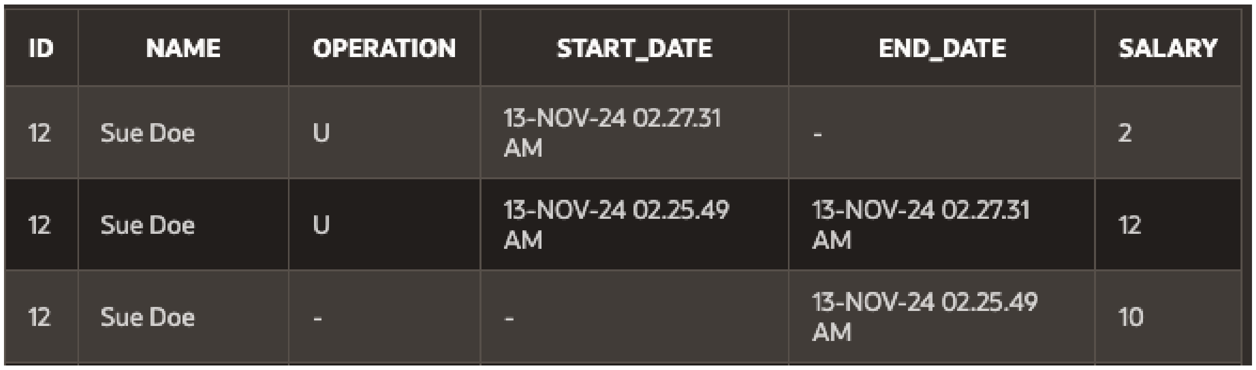
i. **ANS:** SELECT * FROM copy_f_staffs



ID	FIRST_NAME	LAST_NAME	BIRTHDATE	SALARY	OVERTIME_RATE	TRAINING
12	Sue	Doe	01-Jul-1980	2	11.1	-
9	Bob	Miller	19-Mar-1979	10	.75	Grill
19	Monique	Tuttle	30-Mar-1969	60	-	-

f. Now, issue a FLASHBACK QUERY statement against the copy_f_staffs table, so you can see all the changes made

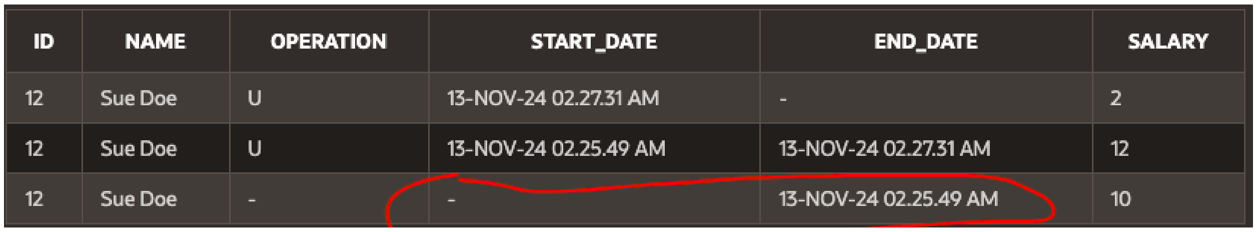
ANS:
SELECT id, first_name ||' '|| last_name AS "NAME",
  versions_operation AS "OPERATION",
  versions_starttime AS "START_DATE",
  versions_endtime AS "END_DATE",
  salary
FROM copy_f_staffs
  VERSIONS BETWEEN SCN MINVALUE AND MAXVALUE
WHERE LOWER(first_name || ' ' || last_name) = 'sue doe';



ID	NAME	OPERATION	START_DATE	END_DATE	SALARY
12	Sue Doe	U	13-NOV-24 02.27.31 AM	-	2
12	Sue Doe	U	13-NOV-24 02.25.49 AM	13-NOV-24 02.27.31 AM	12
12	Sue Doe	-	-	13-NOV-24 02.25.49 AM	10

g. Investigate the result of step 6, and find the original salary and update the copy_f_staffs table salary column for Sue Doe back to her original salary

ANS:
The original salary is the one where the start_date is blank

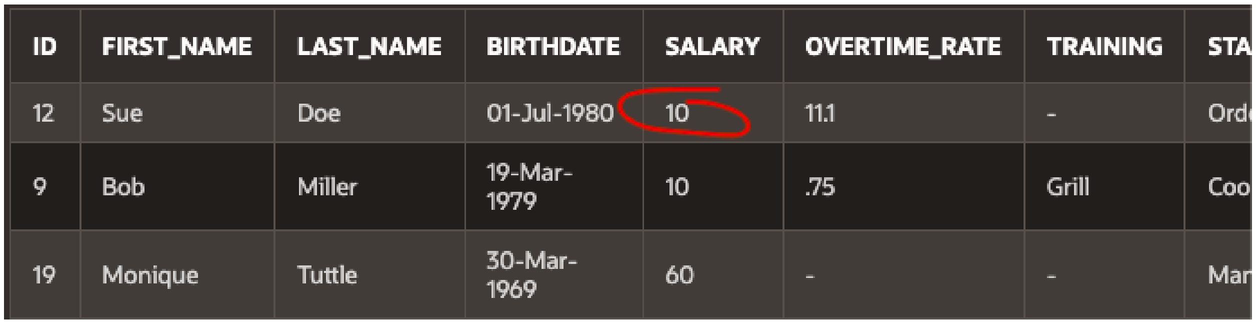


ID	NAME	OPERATION	START_DATE	END_DATE	SALARY
12	Sue Doe	U	13-NOV-24 02.27.31 AM	-	2
12	Sue Doe	U	13-NOV-24 02.25.49 AM	13-NOV-24 02.27.31 AM	12
12	Sue Doe	-	-	13-NOV-24 02.25.49 AM	10

<u>Version 1</u>
UPDATE copy_f_staffs
SET salary = 10
WHERE LOWER(first_name || ' ' || last_name) = 'sue doe';

<u>Version 2</u> (more dynamic approach)

```
UPDATE copy_f_staffs
SET salary = (
    SELECT salary
    FROM copy_f_staffs
        VERSIONS BETWEEN SCN MINVALUE AND MAXVALUE
        WHERE LOWER(first_name || ' ' || last_name) = 'sue doe'
            AND versions_starttime IS NULL
)
WHERE LOWER(first_name || ' ' || last_name) = 'sue doe';
```



ID	FIRST_NAME	LAST_NAME	BIRTHDATE	SALARY	OVERTIME_RATE	TRAINING	STA
12	Sue	Doe	01-Jul-1980	10	11.1	-	Ord
9	Bob	Miller	19-Mar-1979	10	.75	Grill	Coo
19	Monique	Tuttle	30-Mar-1969	60	-	-	Mar