

# Creating and Managing Tables

EX\_NO:1

DATE:

1. Create the DEPT table based on the DEPARTMENT following the table instance chart below. Confirm that the table is created.

Column name	ID	NAME
Key Type		
Nulls/Unique		
FK table		
FK column		
Data Type	Number	Varchar2
Length	7	25

QUERY:

```
CREATE TABLE DEPT (ID NUMBER(7), NAME VARCHAR2(25));
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. A search bar and user profile are also present. The main area is titled 'SQL Commands' and contains a code editor with the following content:

```
1 CREATE TABLE DEPT (ID NUMBER(7), NAME VARCHAR2(25));
```

Below the code editor, there are tabs for Results, Explain, Describe, Saved SQL, and History. The Results tab is selected, displaying the output: "Table created." and "0.03 seconds". At the bottom, footer information includes email addresses (220701065@rajalakshmi.edu.in, dharshini\_123), a language setting (en), copyright notice (Copyright © 1999, 2023, Oracle and/or its affiliates.), and the version (Oracle APEX 23.2.4).

2.Create the EMP table based on the following instance chart. Confirm that the table is created.

Column name	ID	LAST_NAME	FIRST_NAME	DEPT_ID
<b>Key Type</b>				
<b>Nulls/Unique</b>				
<b>FK table</b>				
<b>FK column</b>				
<b>Data Type</b>	Number	Varchar2	Varchar2	Number
<b>Length</b>	7	25	25	7

### QUERY:

```
CREATE TABLE EMP(ID NUMBER(7),LAST_NAME VARCHAR2(25),FIRST_NAME  
VARCHAR2(25),DEPT_ID NUMBER(7));
```

### OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. The user is logged in as Dharshini P (dharshini\_123). The main workspace is titled "SQL Commands". The command entered is:

```
1 CREATE TABLE EMP(ID NUMBER(7),LAST_NAME VARCHAR2(25),FIRST_NAME  
VARCHAR2(25),DEPT_ID NUMBER(7));
```

The results section shows the message "Table created." and a execution time of "0.02 seconds". The bottom footer includes copyright information for Oracle and the APEX version.

3.Modify the EMP table to allow for longer employee last names. Confirm the modification.(Hint: Increase the size to 50)

### QUERY:

```
ALTER TABLE EMP MODIFY(LAST_NAME VARCHAR2(50));
```

### OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. A search bar and user profile 'Dharshini P dharshini\_123' are also present. The main workspace is titled 'SQL Commands' and contains a single command: '1 ALTER TABLE EMP MODIFY(LAST\_NAME VARCHAR2(50));'. Below the command, the 'Results' tab is selected, showing the output: 'Table altered.' and '0.05 seconds'. The bottom footer includes copyright information for Oracle and the APEX version.

```
1 ALTER TABLE EMP MODIFY(LAST_NAME VARCHAR2(50));
```

Table altered.  
0.05 seconds

Copyright © 1999, 2023, Oracle and/or its affiliates. Oracle APEX 23.2.4

4.Create the EMPLOYEES2 table based on the structure of EMPLOYEES table. Include Only the Employee\_id, First\_name, Last\_name, Salary and Dept\_id coloumns. Name the columns Id, First\_name, Last\_name, salary and Dept\_id respectively.

### QUERY:

```
CREATE TABLE EMPLOYEES2(ID NUMBER(7), FIRST_NAME VARCHAR(25), LAST_NAME  
VARCHAR(25), SALARY INT, DEPT_ID NUMBER(7));
```

### OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. A search bar and user profile information for 'Dharshini P' are also present. The main workspace is titled 'SQL Commands' and contains the SQL code for creating the 'EMPLOYEES2' table. The code is highlighted in red. Below the code, the results tab is selected, showing the message 'Table created.' and a execution time of '0.03 seconds'. The bottom footer displays copyright information and the version 'Oracle APEX 23.2.4'.

```
1 CREATE TABLE EMPLOYEES2(ID NUMBER(7), FIRST_NAME VARCHAR(25), LAST_NAME VARCHAR(25), SALARY INT, DEPT_ID NUMBER(7));
```

Results Explain Describe Saved SQL History

Table created.  
0.03 seconds

Copyright © 1999, 2023, Oracle and/or its affiliates. Oracle APEX 23.2.4

5.Drop the EMP table.

### QUERY:

DROP TABLE EMP;

### OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. A search bar and user profile for 'Dharshini P dharshini\_123' are also present. The main workspace is titled 'SQL Commands' and contains a command line with the following text:

```
1  DROP TABLE EMP;
```

Below the command line, the results tab is selected, showing the output:

Table dropped.  
0.07 seconds

At the bottom, footer information includes email (220701065@rajalakshmi.edu.in), a session ID (dharshini\_123), and language (en). Copyright information for Oracle APEX 23.2.4 is also displayed.

6.Rename the EMPLOYEES2 table as EMP.

**QUERY:**

```
ALTER TABLE EMPLOYEES2 RENAME TO EMP;
```

**OUTPUT:**

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. A search bar and user profile for 'Dharshini P dharshini\_123' are also present. The main workspace is titled 'SQL Commands' and contains the following content:

```
1 ALTER TABLE EMPLOYEES2 RENAME TO EMP;
```

Below the command, the output shows:

```
Table altered.
```

Execution details indicate:

```
0.05 seconds
```

At the bottom, footer information includes:

```
220701065@rajalakshmi.edu.in dharshini_123 en
```

Copyright © 1999, 2023, Oracle and/or its affiliates.

Oracle APEX 23.2.4

7.Add a comment on DEPT and EMP tables. Confirm the modification by describing the table.

### QUERY:

```
COMMENT ON TABLE EMP IS 'Department Info';
```

### OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. A search bar and user profile 'Dharshini P dharshini\_123' are also present. The main workspace is titled 'SQL Commands' and contains the following content:

```
↑ SQL Commands
Language SQL Rows 10 Clear Command Find Tables Save Run
1 COMMENT ON TABLE EMP IS 'Department Info';
```

Below the workspace, a toolbar provides icons for Refresh, Undo, Redo, Search, and Paste. The bottom navigation bar includes tabs for Results (selected), Explain, Describe, Saved SQL, and History. The results section displays the output of the executed command:

```
Statement processed.
0.02 seconds
```

At the bottom of the page, footer information includes the email '220701065@rajalakshmi.edu.in', session ID 'dharshini\_123', language 'en', copyright notice 'Copyright © 1999, 2023, Oracle and/or its affiliates.', and the version 'Oracle APEX 23.2.4'.

8.Drop the First\_name column from the EMP table and confirm it.

### QUERY:

```
ALTER TABLE EMP DROP COLUMN FIRST_NAME;
```

### OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. A search bar and user profile information for 'Dharshini P' are also present. The main workspace is titled 'SQL Commands' and contains the following content:

```
1  ALTER TABLE EMP DROP COLUMN FIRST_NAME;
```

Below the command, the output pane displays the results of the execution:

```
Table altered.
```

Execution time: 0.06 seconds

At the bottom, footer information includes the email 220701065@rajalakshmi.edu.in, the schema name dharshini\_123, and the copyright notice Copyright © 1999, 2023, Oracle and/or its affiliates. The version Oracle APEX 23.2.4 is also mentioned.

Evaluation Procedure	Marks awarded
Query(5)	
Execution (5)	
Viva(5)	
Total (15)	
Faculty Signature	

### RESULT:

# MANIPULATING DATA

EX\_NO:2

DATE:

1.Create MY\_EMPLOYEE table with the following structure

NAME	NULL?	TYPE
ID	Not null	Number(4)
Last_name		Varchar(25)
First_name		Varchar(25)
Userid		Varchar(25)
Salary		Number(9,2)

QUERY:

```
CREATE TABLE MY_EMPLOYEE(Id Number (4) NOT NULL, Last_name Varchar (25), First_name Varchar (25), Userid Varchar (25), Salary Number (9,2));
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. A search bar and user profile 'Dharshini P dharshini\_123' are also present. The main workspace is titled 'SQL Commands' and contains a command line with the SQL code for creating the table. The results tab at the bottom shows the output: 'Table created.' and a execution time of '0.04 seconds'. The bottom footer includes copyright information for Oracle and the APEX version.

```
1 CREATE TABLE MY_EMPLOYEE(Id Number (4) NOT NULL, Last_name Varchar (25), First_name Varchar (25), Userid Varchar (25), Salary Number (9,2));
```

Results Explain Describe Saved SQL History

Table created.

0.04 seconds

Copyright © 1999, 2023, Oracle and/or its affiliates.

Oracle APEX 23.2.4

2.Add the first and second rows data to MY\_EMPLOYEE table from the following sample data.

ID	Last_name	First_name	Userid	salary
1	Patel	Ralph	rpatel	895
2	Dancs	Betty	bdancs	860
3	Biri	Ben	bbiri	1100
4	Newman	Chad	Cnewman	750
5	Ropebur	Audrey	aropebur	1550

### QUERY:

```
INSERT INTO MY_EMPLOYEE VALUES(1, 'Patel', 'Ralph', 'rpatel' ,895);
```

### OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. The user is logged in as Dharshini P (dharsinh\_123). The SQL Commands tab is selected, showing the following command in the editor:

```
1  INSERT INTO MY_EMPLOYEE VALUES(1, 'Patel', 'Ralph', 'rpatel' ,895);
```

The results section below shows the output of the query:

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

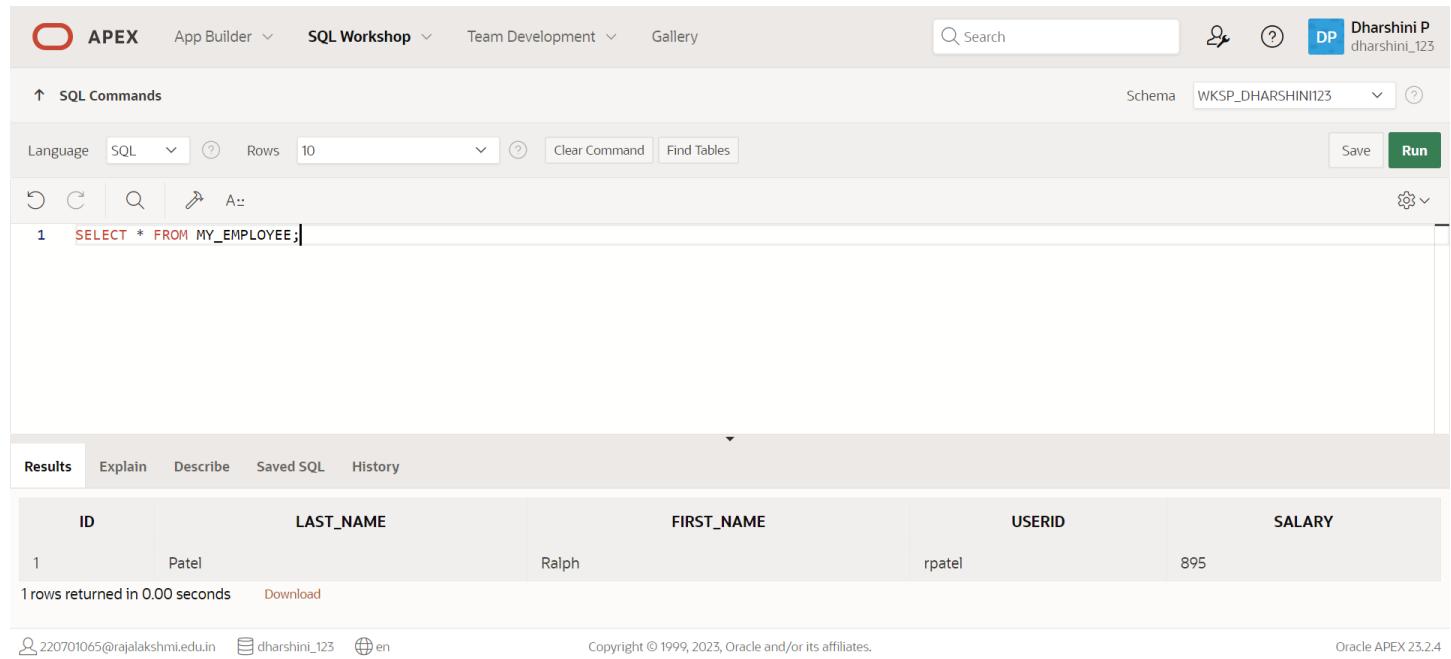
At the bottom, footer information includes email (220701065@rajalakshmi.edu.in), session ID (dharsinh\_123), and page number (1 of 1).

3.Display the table with values.

### QUERY:

```
SELECT * FROM MY_EMPLOYEE;
```

### OUTPUT:



The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. A search bar and user profile are also present. The main area is titled "SQL Commands" and contains a code editor with the following content:

```
1  SELECT * FROM MY_EMPLOYEE;
```

Below the code editor, there are tabs for Results, Explain, Describe, Saved SQL, and History. The Results tab is selected, displaying the following table:

ID	LAST_NAME	FIRST_NAME	USERID	SALARY
1	Patel	Ralph	rpatel	895

Below the table, it says "1 rows returned in 0.00 seconds" and provides a "Download" link. At the bottom of the page, there are footer links for email, user profile, and help, along with copyright information: "Copyright © 1999, 2023, Oracle and/or its affiliates." and "Oracle APEX 23.2.4".

4.Populate the next two rows of data from the sample data. Concatenate the first letter of the first\_name with the first seven characters of the last\_name to produce Userid.

### QUERY:

```
INSERT INTO MY_EMPLOYEE VALUES (2, 'Dancs', 'Betty', 'bdancs' ,860);
```

### OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. A search bar and user profile for 'Dharshini P dharshini\_123' are also present. The main workspace displays the SQL command:

```
1  INSERT INTO MY_EMPLOYEE VALUES (2, 'Dancs', 'Betty', 'bdancs' ,860);|
```

Below the command, the results section shows the output:

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

Execution time is listed as 0.01 seconds. The bottom footer includes copyright information for Oracle and the APEX version.

5. Make the data additions permanent.

### QUERY:

```
SELECT * FROM MY_EMPLOYEE;
```

### OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. At the top, there are tabs for App Builder, SQL Workshop (selected), Team Development, and Gallery. A search bar and user profile are also at the top right. Below the tabs, there's a toolbar with icons for Undo, Redo, Find, Replace, and others. The main area contains a SQL command line and a results grid.

**SQL Commands:**

```
1  SELECT * FROM MY_EMPLOYEE;
```

**Results Grid:**

ID	LAST_NAME	FIRST_NAME	USERID	SALARY
1	Patel	Ralph	rpatel	895
2	Dancs	Betty	bdancs	860
2	Dancs	Betty	bDANCS	860

3 rows returned in 0.01 seconds    [Download](#)

Copyright © 1999, 2023, Oracle and/or its affiliates.    Oracle APEX 23.2.4

6.Change the last name of employee 3 to Drexler.

### QUERY:

```
UPDATE MY_EMPLOYEE SET Last_Name ='Drexler' WHERE Id=3;
```

### OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. A search bar and user profile 'Dharsini P dharshini\_123' are also present. The main workspace is titled 'SQL Commands' and contains a command line with the following SQL statement:

```
1 | UPDATE MY_EMPLOYEE SET Last_Name ='Drexler' WHERE Id=3;
```

Below the command line, the results tab is selected, showing the output of the query:

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

Execution time is listed as 0.00 seconds. The bottom of the page displays copyright information and the version 'Oracle APEX 23.2.4'.

7.Change the salary to 1000 for all the employees with a salary less than 900.

### QUERY:

```
UPDATE MY_EMPLOYEE SET Salary =1000 WHERE Salary<900;
```

### OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. A search bar and user profile 'Dharshini P dharshini\_123' are also present. The main workspace displays the SQL command:

```
1 UPDATE MY_EMPLOYEE SET Salary =1000 WHERE Salary<900;
```

Below the command, the results section shows the output:

```
2 row(s) updated.
```

Execution time is listed as 0.00 seconds. The bottom of the page includes footer information: email (220701065@rajalakshmi.edu.in), session ID (dharshini\_123), language (en), copyright notice (Copyright © 1999, 2023, Oracle and/or its affiliates.), and version (Oracle APEX 23.2.4).

8.Delete Betty dancs from MY\_EMPLOYEE table.

### QUERY:

```
DELETE FROM MY_EMPLOYEE WHERE First_name= 'Betty';
```

### OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. A search bar and user profile are also present. The main workspace displays the SQL command:

```
1  DELETE FROM MY_EMPLOYEE WHERE First_name= 'Betty';
```

Below the command, the results section shows the output:

```
1 row(s) deleted.  
0.01 seconds
```

At the bottom, footer information includes email addresses, a copyright notice, and the version Oracle APEX 23.2.4.

9.Empty the fourth row of the emp table.

### QUERY:

DELETE FROM MY\_EMPLOYEE WHERE Id=4;

### OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. The user is logged in as Dharshini P (dharshini\_123). The SQL Commands tab is selected, showing the following command in the editor:

```
1  DELETE FROM MY_EMPLOYEE WHERE Id=4;
```

Below the editor, the Results tab is active, displaying the output of the query:

1 row(s) deleted.  
0.01 seconds

At the bottom of the page, there are footer links for email (220701065@rajalakshmi.edu.in), user profile (dharshini\_123), and language (en). The copyright notice reads "Copyright © 1999, 2023, Oracle and/or its affiliates." and the version is "Oracle APEX 23.2.4".

Evaluation Procedure	Marks awarded
Query(5)	
Execution (5)	
Viva(5)	
Total (15)	
Faculty Signature	

### RESULT:

# INCLUDING CONSTRAINTS

EX\_NO:3

DATE:

1. Add a table-level PRIMARY KEY constraint to the EMP table on the ID column. The constraint should be named at creation. Name the constraint my\_emp\_id\_pk.

QUERY:

```
CREATE TABLE EMP2(ID number (6), Last_Name varchar2(25) NOT NULL, Salary number (8,2), constraint my_emp_id_pk PRIMARY KEY(ID));
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. A search bar and user profile information for 'Dharshini P' are also present. The main workspace displays the SQL command for creating the 'EMP2' table with a primary key constraint:

```
1 CREATE TABLE EMP2(ID number (6), Last_Name varchar2(25) NOT NULL, Salary number (8,2), constraint my_emp_id_pk PRIMARY KEY(ID));
```

The results section below shows the output of the query:

```
Table created.  
0.06 seconds
```

At the bottom, footer information includes email addresses (220701065@rajalakshmi.edu.in, dharshini\_123), a language setting (en), copyright notice (Copyright © 1999, 2023, Oracle and/or its affiliates.), and the version (Oracle APEX 23.2.4).

2.Create a PRIMARY KEY constraint to the DEPT table using the ID column. The constraint should be named at creation. Name the constraint my\_dept\_id\_pk.

### QUERY:

**CREATE TABLE DEPT2(ID number (6), Last\_Name varchar2(25) NOT NULL, Email varchar2(25), constraint my\_dept\_id\_pk PRIMARY KEY(ID));**

### OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. A search bar and user profile 'Dharshini P dharshini\_123' are also present. The main workspace displays the SQL command for creating the 'DEPT2' table, which includes a PRIMARY KEY constraint named 'my\_dept\_id\_pk'. The command is highlighted in red. Below the command, the output shows 'Table created.' and a execution time of '0.06 seconds'. The bottom footer includes copyright information for Oracle and the APEX version 'Oracle APEX 23.2.4'.

```
1 CREATE TABLE DEPT2(ID number (6), Last_Name varchar2(25) NOT NULL, Email varchar2(25), constraint my_dept_id_pk PRIMARY KEY(ID));
```

Table created.  
0.06 seconds

Copyright © 1999, 2023, Oracle and/or its affiliates. Oracle APEX 23.2.4

3.Add a column DEPT\_ID to the EMP table. Add a foreign key reference on the EMP table that ensures that the employee is not assigned to nonexistent department. Name the constraint my\_emp\_dept\_id\_fk.

### QUERY:

```
ALTER TABLE EMP2 ADD CONSTRAINT my_emp_dept_id_fk FOREIGN KEY(DEPT_ID)
REFERENCES EMP2 (ID);
```

### OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. A search bar and user profile 'Dharshini P dharshini\_123' are also present. The main workspace is titled 'SQL Commands' and contains the following SQL code:

```
1  ALTER TABLE EMP2 ADD CONSTRAINT my_emp_dept_id_fk FOREIGN KEY(DEPT_ID) REFERENCES EMP2 (ID);
```

Below the code, the 'Results' tab is selected, showing the output: "Table altered." and "5.09 seconds". The bottom footer displays user information (220701065@rajalakshmi.edu.in, dharshini\_123, en) and copyright notice (Copyright © 1999, 2023, Oracle and/or its affiliates). The version 'Oracle APEX 23.2.4' is also mentioned.

4.Modify the EMP table. Add a COMMISSION column of NUMBER data type, precision 2, scale 2. Add a constraint to the commission column that ensures that a commission value is greater than zero.

### QUERY:

```
ALTER TABLE EMP2 ADD COMMISSION NUMBER (2,2) CHECK (COMMISSION>0);
```

### OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. A search bar and user profile information for 'Dharshini P dharshini\_123' are also present. The main workspace is titled 'SQL Commands' and contains the following content:

```
1  ALTER TABLE EMP2 ADD COMMISSION NUMBER (2,2) CHECK (COMMISSION>0);
```

Below the command, the results section displays the output:

```
Table altered.
```

Execution time is listed as 5.15 seconds. The bottom of the page includes footer information:

220701065@rajalakshmi.edu.in dharshini\_123 en Copyright © 1999, 2023, Oracle and/or its affiliates. Oracle APEX 23.2.4

Evaluation Procedure	Marks awarded
Query(5)	
Execution (5)	
Viva(5)	
Total (15)	
Faculty Signature	

**RESULT:**

# Writing Basic SQL SELECT Statements

EX\_NO:4

DATE:

- 1.The following statement executes successfully.

## Identify the Errors

```
SELECT employee_id, last_name  
sal*12 ANNUAL SALARY  
FROM employees;
```

## QUERY:

```
SELECT EMPLOYEE_ID, LAST_NAME, SALARY*12 AS "ANNUAL SALARY" FROM  
MY_EMPLOYEE;
```

## OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. A search bar and user profile information for 'Dharshini P' are also present. The main workspace is titled 'SQL Commands'. It features a toolbar with icons for Language (SQL), Rows (10), Clear Command, Find Tables, Save, and Run. Below the toolbar, the SQL command is entered:

```
1  SELECT EMPLOYEE_ID, LAST_NAME, SALARY*12 AS "ANNUAL SALARY" FROM MY_EMPLOYEE;
```

The results section displays the output of the query:

EMPLOYEE_ID	LAST_NAME	ANNUAL SALARY
1	Patel	12000
3	Drexler	13200

Below the table, it says '2 rows returned in 0.01 seconds' and provides a 'Download' link. The bottom footer includes copyright information for Oracle and the APEX version.

2. Show the structure of departments the table. Select all the data from it.

**QUERY:**

DESC DEPT2;

**OUTPUT:**

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. A search bar and user profile information (Dharshini P dharshini\_123) are also present. The main workspace displays the SQL command 'DESC DEPT2;' entered in the command line. Below the command, the 'Describe' tab is selected, showing the structure of the DEPT2 table. The table has three columns: ID (NUMBER), LAST\_NAME (VARCHAR2(25)), and EMAIL (VARCHAR2(25)). The EMAIL column is marked as nullable (indicated by a checkmark in the 'Nullable' column). The bottom of the screen shows user session details (220701065@rajalakshmi.edu.in, dharshini\_123, en) and a copyright notice (Copyright © 1999, 2023, Oracle and/or its affiliates).

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
DEPT2	ID	NUMBER	-	6	0	1	-	-	-
	LAST_NAME	VARCHAR2	25	-	-	-	-	-	-
	EMAIL	VARCHAR2	25	-	-	-	✓	-	-

3.Create a query to display the last name, job code, hire date, and employee number for each employee, with employee number appearing first.

### QUERY:

```
SELECT EMPLOYEE_ID, LAST_NAME, JOB_CODE, HIRE_DATE FROM MY_EMPLOYEE;
```

### OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. At the top, there are tabs for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there is a search bar, user profile for Dharshini P (dharshini\_123), and a schema dropdown set to WKSP\_DHARSHINI123. Below the tabs, there are buttons for Language (SQL selected), Rows (10), Clear Command, Find Tables, Save, and Run. The main area contains the SQL command: `SELECT EMPLOYEE_ID, LAST_NAME, JOB_CODE, HIRE_DATE FROM MY_EMPLOYEE;`. The Results tab is selected, displaying the following data:

EMPLOYEE_ID	LAST_NAME	JOB_CODE	HIRE_DATE
1	Patel	20	21
3	Drexler	30	15

Below the table, it says "2 rows returned in 0.01 seconds" and has a "Download" link. At the bottom, there are footer links for 220701065@rajalakshmi.edu.in, dharshini\_123, and en, along with copyright information: Copyright © 1999, 2023, Oracle and/or its affiliates. and Oracle APEX 23.2.4.

4. Provide an alias STARTDATE for the hire date.

### QUERY:

```
SELECT HIRE_DATE AS "STARTDATE" FROM MY_EMPLOYEE;
```

### OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. A search bar and user profile are also present. The main workspace is titled 'SQL Commands' and contains a SQL editor with the following content:

```
1 SELECT HIRE_DATE AS "STARTDATE" FROM MY_EMPLOYEE;
```

Below the editor, the results tab is selected, displaying the output:

STARTDATE
21
15

Text at the bottom of the results pane indicates "2 rows returned in 0.01 seconds" and provides download options.

At the bottom of the page, footer information includes email addresses (220701065@rajalakshmi.edu.in, dharshini\_123), language (en), and copyright notice (Copyright © 1999, 2023, Oracle and/or its affiliates). The page is identified as Oracle APEX 23.2.4.

5.Create a query to display unique job codes from the employee table.

### QUERY:

```
SELECT DISTINCT JOB_CODE FROM MY_EMPLOYEE;
```

### OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. A search bar and user profile for 'Dharshini P' are also present. The main workspace displays the SQL command:

```
1 SELECT DISTINCT JOB_CODE FROM MY_EMPLOYEE;
```

The results tab is selected, showing the output:

JOB_CODE
30
20

Below the results, it says '2 rows returned in 0.01 seconds' and provides a 'Download' link. The bottom footer includes copyright information for Oracle and the APEX version.

6. Display the last name concatenated with the job ID , separated by a comma and space, and name the column EMPLOYEE and TITLE.

### QUERY:

```
SELECT LAST_NAME ||', '|| JOB_CODE AS "EMPLOYEE AND TITLE" FROM MY_EMPLOYEE;
```

### OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. A search bar and user profile are also present. The main area is titled 'SQL Commands' and contains the following SQL code:

```
1 SELECT LAST_NAME ||', '|| JOB_CODE AS "EMPLOYEE AND TITLE" FROM MY_EMPLOYEE;
2
```

The 'Results' tab is selected, displaying the output:

EMPLOYEE AND TITLE	
Patel, 20	
Drexler, 30	

Below the results, it says '2 rows returned in 0.02 seconds' and provides a 'Download' link. The bottom footer includes copyright information for Oracle and the APEX version.

7.Create a query to display all the data from the employees table. Separate each column by a comma. Name the column THE\_OUTPUT.

### QUERY:

```
SELECT LAST_NAME ||','|| JOB_CODE ||','|| EMPLOYEE_ID ||','|| HIRE_DATE AS "THE OUTPUT" FROM  
MY_EMPLOYEE;
```

### OUTPUT:

The screenshot shows the Oracle APEX interface with the SQL Workshop module selected. The query entered is:

```
1 SELECT LAST_NAME ||','|| JOB_CODE ||','|| EMPLOYEE_ID ||','|| HIRE_DATE AS "THE OUTPUT" FROM MY_EMPLOYEE;
```

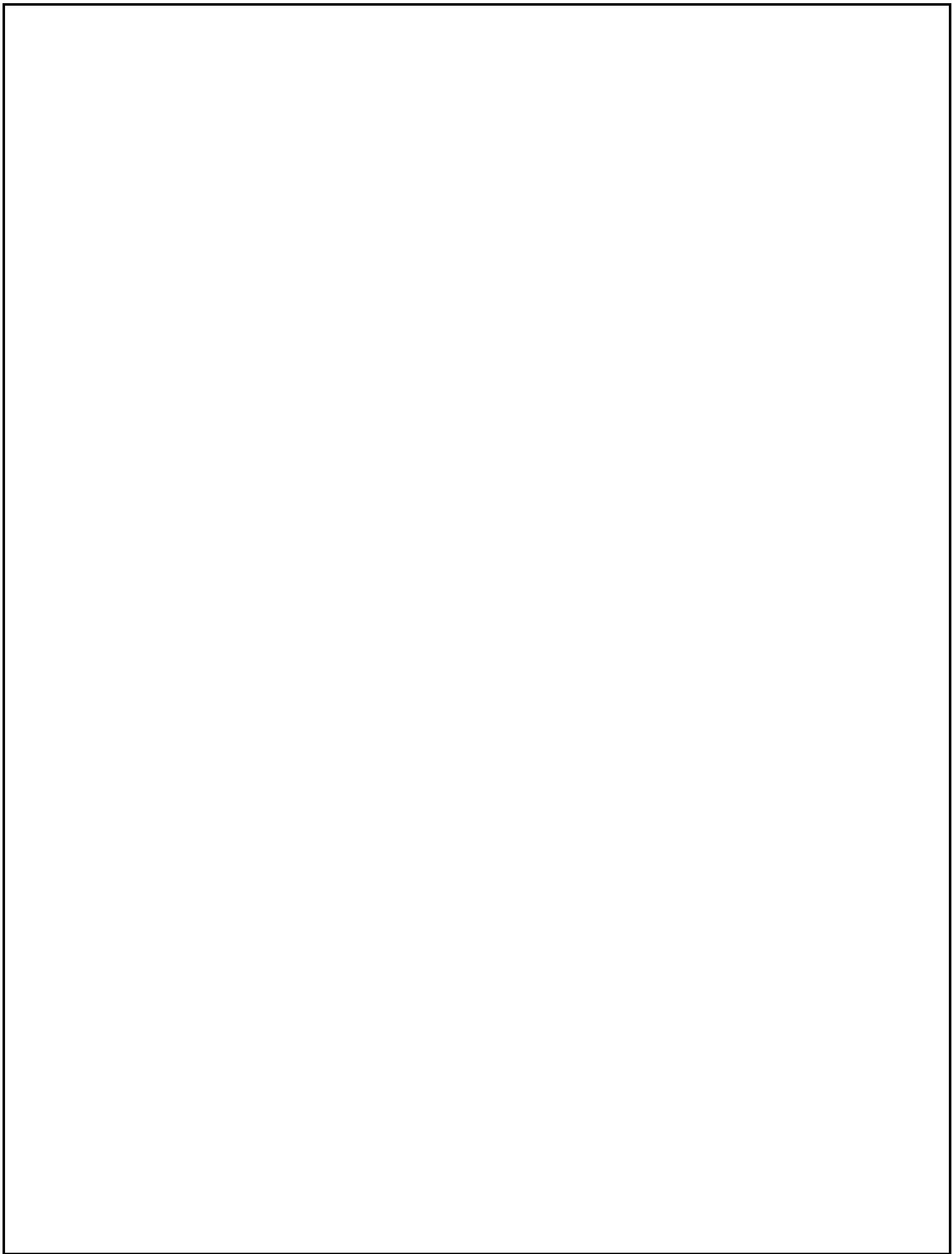
The results section displays the output:

THE OUTPUT
Patel,20,1,21
Drexler,30,3,15

Below the results, it says "2 rows returned in 0.01 seconds".

Evaluation Procedure	Marks awarded
Query(5)	
Execution (5)	
Viva(5)	
Total (15)	
Faculty Signature	

**RESULT:**



# **RESTRICTING AND SORTING DATA**

**EX.NO.5**

**DATE:**

Find the Solution for the following:

1. Create a query to display the last name and salary of employees earning more than 12000.

**QUERY:**

```
SELECT LAST_NAME, SALARY FROM EMPLOYEES WHERE SALARY>12000;
```

**OUTPUT:**

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for App Builder, SQL Workshop (selected), Team Development, and Gallery. A search bar and user profile 'Dharshini P dharshini\_123' are also present. The main workspace is titled 'SQL Commands' and contains the following content:

```
1 SELECT LAST_NAME, SALARY FROM EMPLOYEES WHERE SALARY>12000;
```

The 'Results' tab is selected, displaying the query's output:

LAST_NAME	SALARY
Arora	13000
Patel	20000

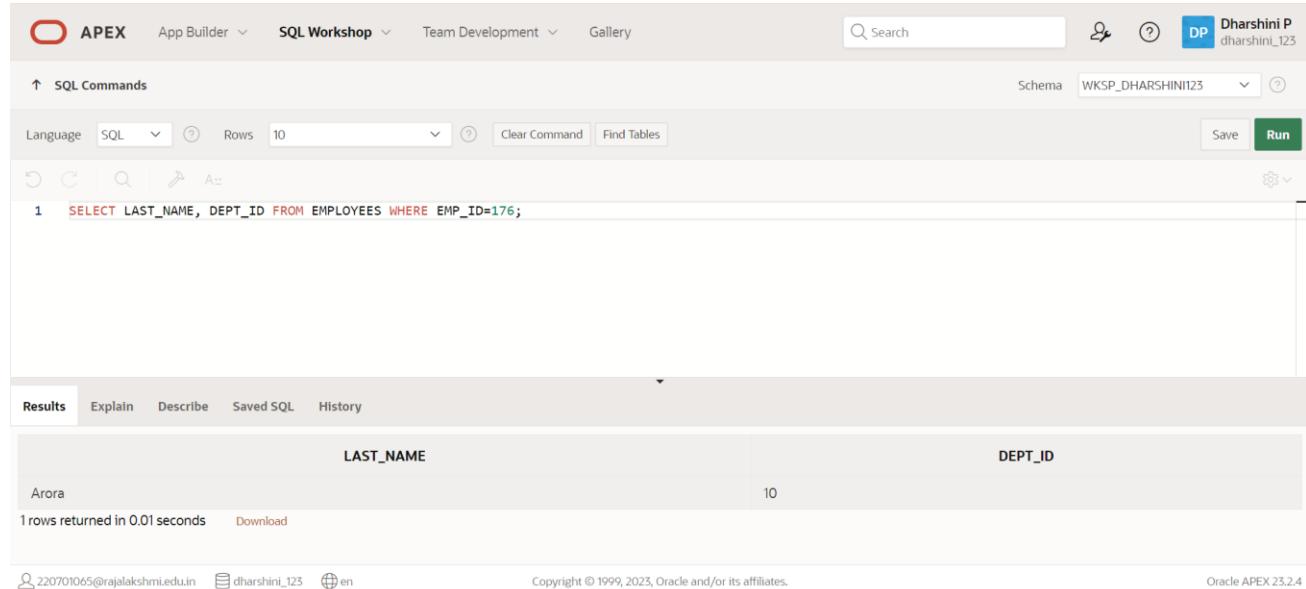
Below the table, it says '2 rows returned in 0.01 seconds'. The bottom of the page includes footer links for 'Download', 'Copyright © 1999, 2023, Oracle and/or its affiliates.', and 'Oracle APEX 23.2.4'.

2. Create a query to display the employee last name and department number for employee number 176.

**QUERY:**

```
SELECT LAST_NAME, DEPT_ID FROM EMPLOYEES WHERE EMP_ID=176;
```

**OUTPUT:**



The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. The right side shows a user profile for 'Dharshini P dharshini\_123'. The main area has tabs for 'SQL Commands' and 'Results'. Under 'SQL Commands', the query 'SELECT LAST\_NAME, DEPT\_ID FROM EMPLOYEES WHERE EMP\_ID=176;' is entered. Under 'Results', the output is displayed in a table:

LAST_NAME	DEPT_ID
Arora	10

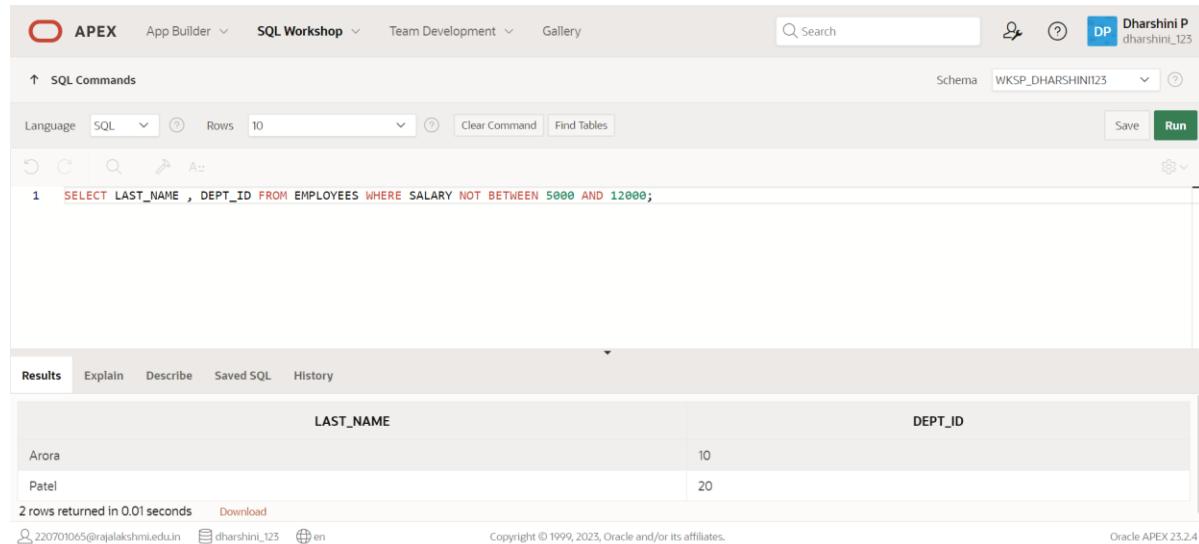
Below the table, it says '1 rows returned in 0.01 seconds' and provides a 'Download' link. The bottom of the page includes copyright information for Oracle and the APEX version 'Oracle APEX 23.2.4'.

3. Create a query to display the last name and salary of employees whose salary is not in the range of 5000 and 12000. (hints: not between )

**QUERY:**

```
SELECT LAST_NAME , DEPT_ID FROM EMPLOYEES WHERE SALARY NOT BETWEEN  
5000 AND 12000;
```

**OUTPUT:**



The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. The right side shows a user profile for 'Dharshini P dharshini\_123'. The main area has tabs for 'SQL Commands' and 'Results'. Under 'SQL Commands', the query 'SELECT LAST\_NAME , DEPT\_ID FROM EMPLOYEES WHERE SALARY NOT BETWEEN 5000 AND 12000;' is entered. Under 'Results', the output is displayed in a table:

LAST_NAME	DEPT_ID
Arora	10
Patel	20

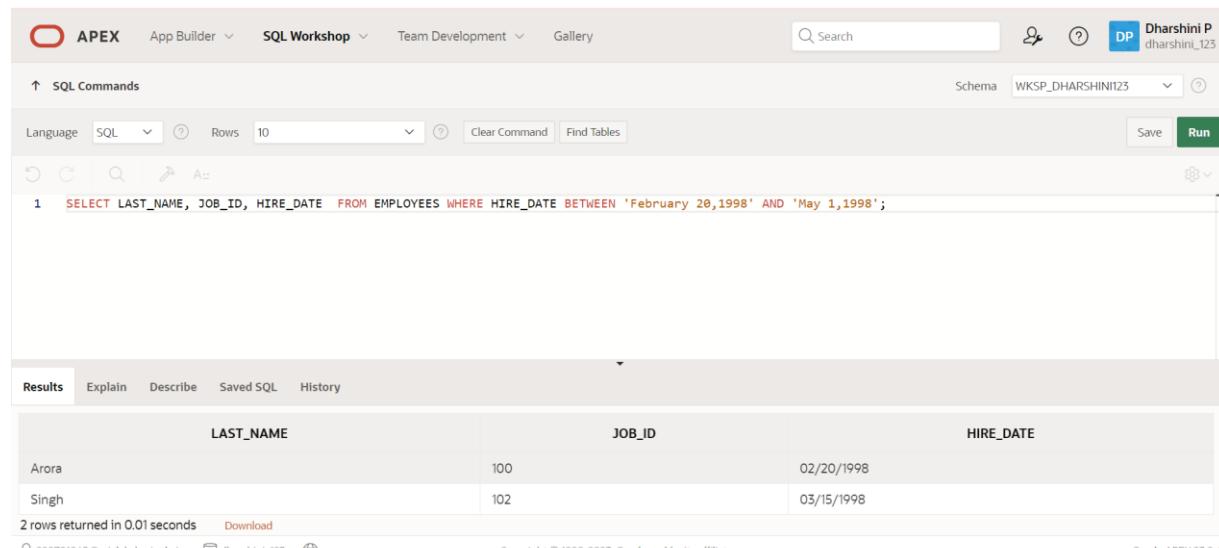
Below the table, it says '2 rows returned in 0.01 seconds' and provides a 'Download' link. The bottom of the page includes copyright information for Oracle and the APEX version 'Oracle APEX 23.2.4'.

4. Display the employee last name, job ID, and start date of employees hired between February 20,1998 and May 1,1998.order the query in ascending order by start date.(hints: between)

#### QUERY:

```
SELECT LAST_NAME, JOB_ID, HIRE_DATE FROM EMPLOYEES WHERE HIRE_DATE BETWEEN 'February 20,1998' AND 'May 1,1998';
```

#### OUTPUT:



The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. The right side shows a user profile for 'Dharshini P dharshini\_123'. The main area has tabs for 'SQL Commands' and 'Results'. The SQL command entered is:

```
1 SELECT LAST_NAME, JOB_ID, HIRE_DATE FROM EMPLOYEES WHERE HIRE_DATE BETWEEN 'February 20,1998' AND 'May 1,1998';
```

The results section displays the following data:

LAST_NAME	JOB_ID	HIRE_DATE
Arora	100	02/20/1998
Singh	102	03/15/1998

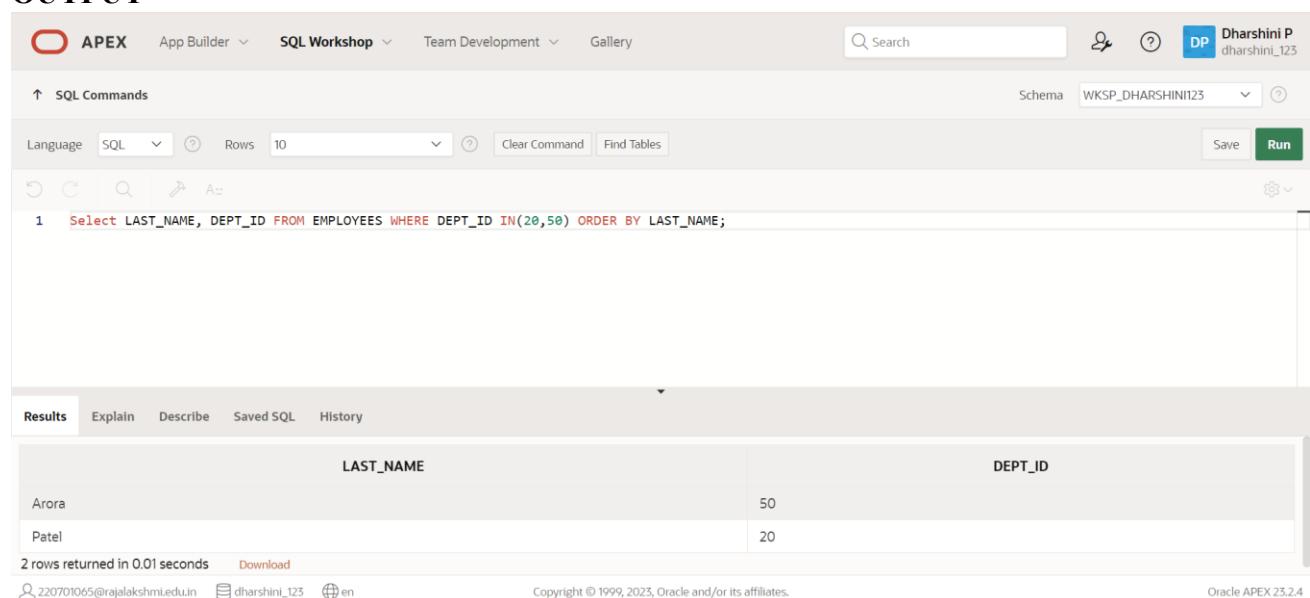
Below the table, it says '2 rows returned in 0.01 seconds'. The bottom of the page includes copyright information for Oracle and the APEX version.

5. Display the last name and department number of all employees in departments 20 and 50 in alphabetical order by name.(hints: in, orderby)

#### QUERY:

```
Select LAST_NAME, DEPT_ID FROM EMPLOYEES WHERE DEPT_ID IN(20,50) ORDER BY LAST_NAME;
```

#### OUTPUT



The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. The right side shows a user profile for 'Dharshini P dharshini\_123'. The main area has tabs for 'SQL Commands' and 'Results'. The SQL command entered is:

```
1 Select LAST_NAME, DEPT_ID FROM EMPLOYEES WHERE DEPT_ID IN(20,50) ORDER BY LAST_NAME;
```

The results section displays the following data:

LAST_NAME	DEPT_ID
Arora	50
Patel	20

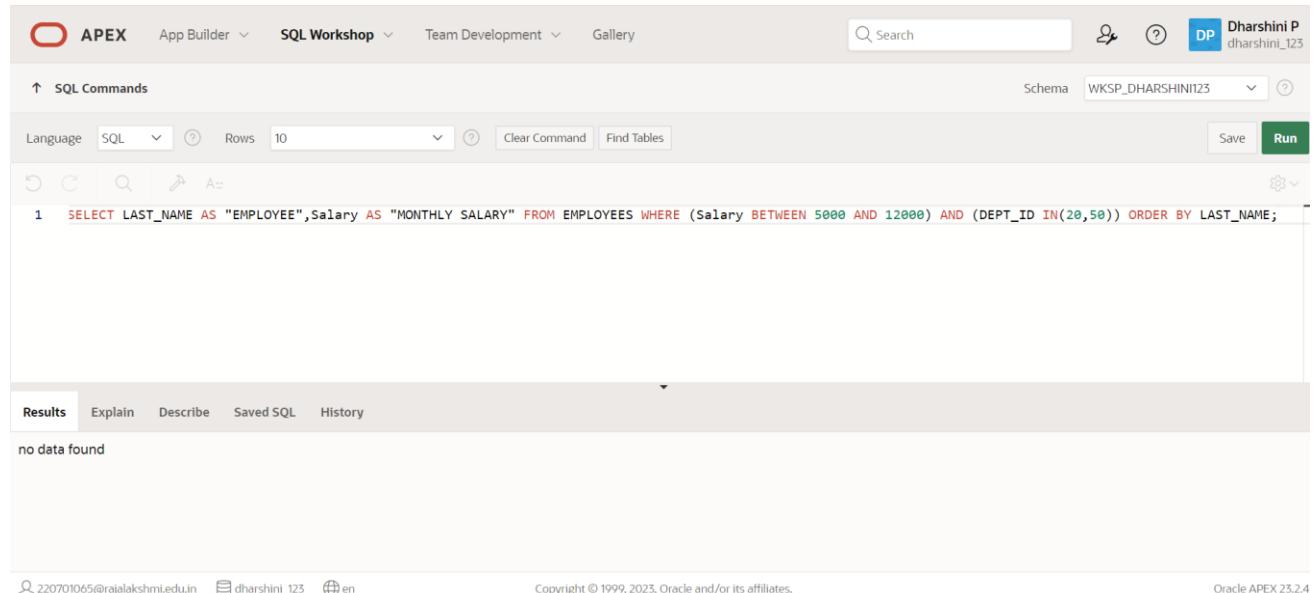
Below the table, it says '2 rows returned in 0.01 seconds'. The bottom of the page includes copyright information for Oracle and the APEX version.

6. Display the last name and salary of all employees who earn between 5000 and 12000 and are in departments 20 and 50 in alphabetical order by name. Label the columns EMPLOYEE, MONTHLY SALARY respectively.(hints: between, in)

**QUERY:**

```
SELECT LAST_NAME AS "EMPLOYEE", Salary AS "MONTHLY SALARY" FROM EMPLOYEES WHERE (Salary BETWEEN 5000 AND 12000) AND (DEPT_ID IN(20,50)) ORDER BY LAST_NAME;
```

**OUTPUT:**



The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. The SQL Workshop tab is selected. The schema dropdown is set to 'WKSP\_DHARSHINI123'. The main area displays the SQL command:

```
1 SELECT LAST_NAME AS "EMPLOYEE", Salary AS "MONTHLY SALARY" FROM EMPLOYEES WHERE (Salary BETWEEN 5000 AND 12000) AND (DEPT_ID IN(20,50)) ORDER BY LAST_NAME;
```

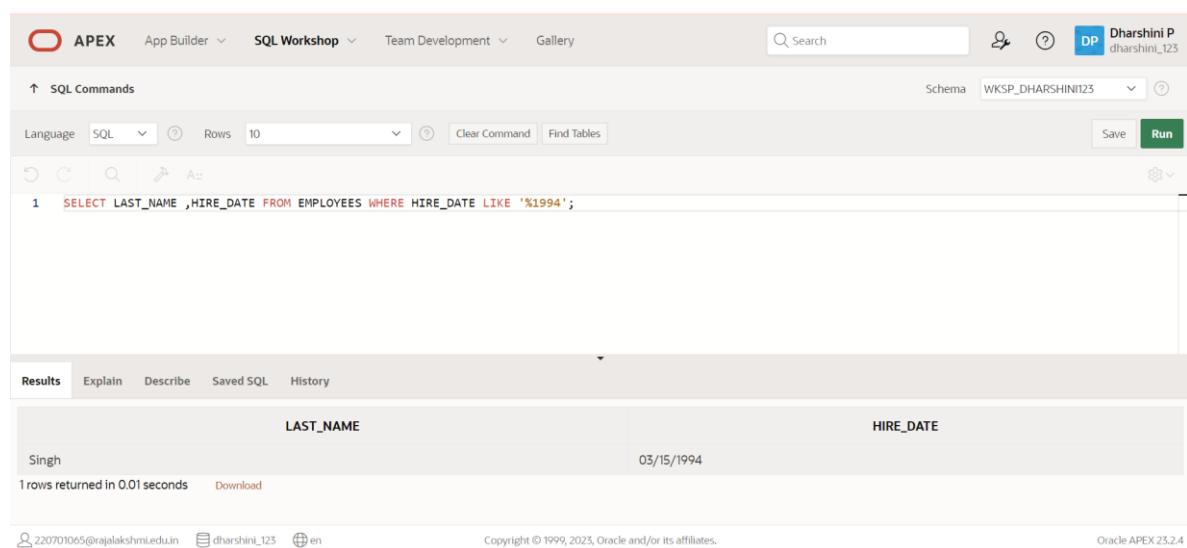
The results section shows the message 'no data found'.

7. Display the last name and hire date of every employee who was hired in 1994.(hints: like)

**QUERY:**

```
SELECT LAST_NAME ,HIRE_DATE FROM EMPLOYEES WHERE HIRE_DATE LIKE '%1994'
```

**OUTPUT:**



The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. The SQL Workshop tab is selected. The schema dropdown is set to 'WKSP\_DHARSHINI123'. The main area displays the SQL command:

```
1 SELECT LAST_NAME ,HIRE_DATE FROM EMPLOYEES WHERE HIRE_DATE LIKE '%1994';
```

The results section displays the output:

LAST_NAME	HIRE_DATE
Singh	03/15/1994

1 rows returned in 0.01 seconds

8. Display the last name and job title of all employees who do not have a manager.(hints: is null)

**QUERY:**

```
SELECT LAST_NAME ,JOB_TITLE FROM EMPLOYEES WHERE MANAGER_ID IS NULL;
```

**OUTPUT:**

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. The top right shows a user profile for 'Dharshini P dharshini\_123'. The main area has tabs for 'SQL Commands' and 'Results'. The SQL command entered is: `SELECT LAST_NAME ,JOB_TITLE FROM EMPLOYEES WHERE MANAGER_ID IS NULL;`. The results table has columns 'LAST\_NAME' and 'JOB\_TITLE', displaying three rows: Arora (MAKETING HEAD), Patel (Sales Manager), and Singh (Graph MANAGER).

LAST_NAME	JOB_TITLE
Arora	MAKETING HEAD
Patel	Sales Manager
Singh	Graph MANAGER

9. Display the last name, salary, and commission for all employees who earn commissions.

Sort data in descending order of salary and commissions.(hints: is not nul,orderby)

**QUERY:**

```
Select last_name ,Salary,commission from employees where commission is not null order by Salary,commission DESC;
```

**OUTPUT:**

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. The top right shows a user profile for 'Dharshini P dharshini\_123'. The main area has tabs for 'SQL Commands' and 'Results'. The SQL command entered is: `Select last_name ,Salary,comission from employees where comission is not null order by Salary,comission DESC;`. The results table has columns 'LAST\_NAME', 'SALARY', and 'COMISSION', displaying three rows: Singh (10000, 2000), Arora (13000, 700), and Patel (20000, 4500).

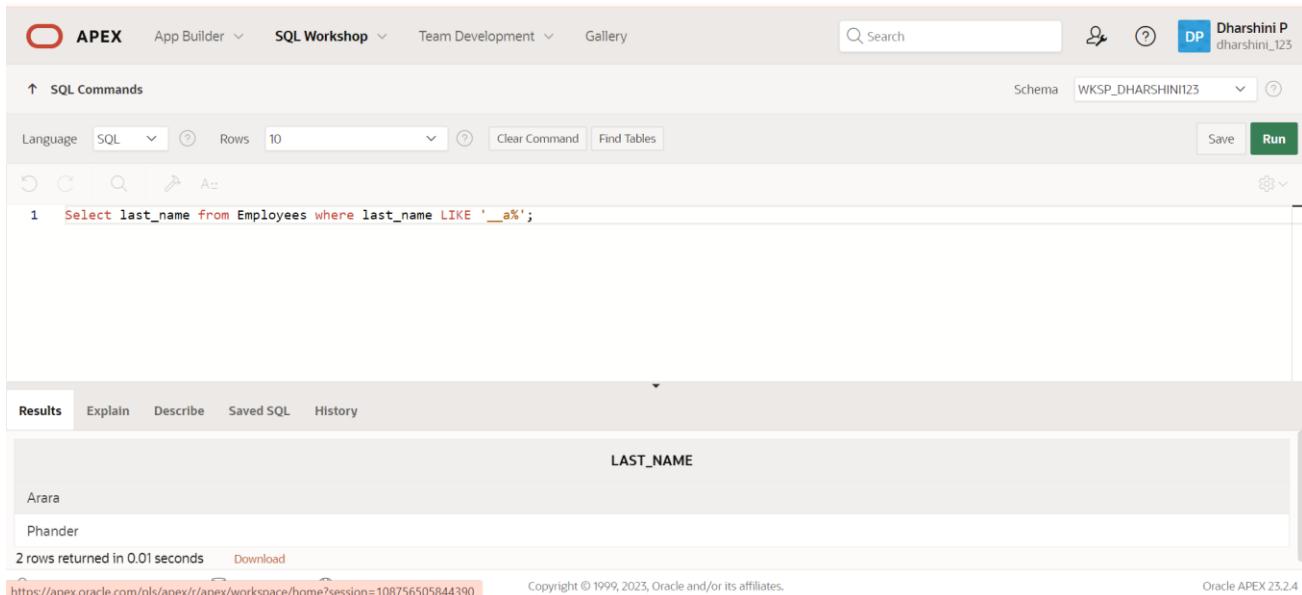
LAST_NAME	SALARY	COMISSION
Singh	10000	2000
Arora	13000	700
Patel	20000	4500

10. Display the last name of all employees where the third letter of the name is a.(hints:like)

**QUERY:**

Select last\_name from Employees where last\_name LIKE '\_\_a%';

**OUTPUT:**



The screenshot shows the Oracle APEX SQL Workshop interface. The query entered is "Select last\_name from Employees where last\_name LIKE '\_\_a%'". The results pane displays two rows: "Arara" and "Phander". The output is timestamped at "2 rows returned in 0.01 seconds".

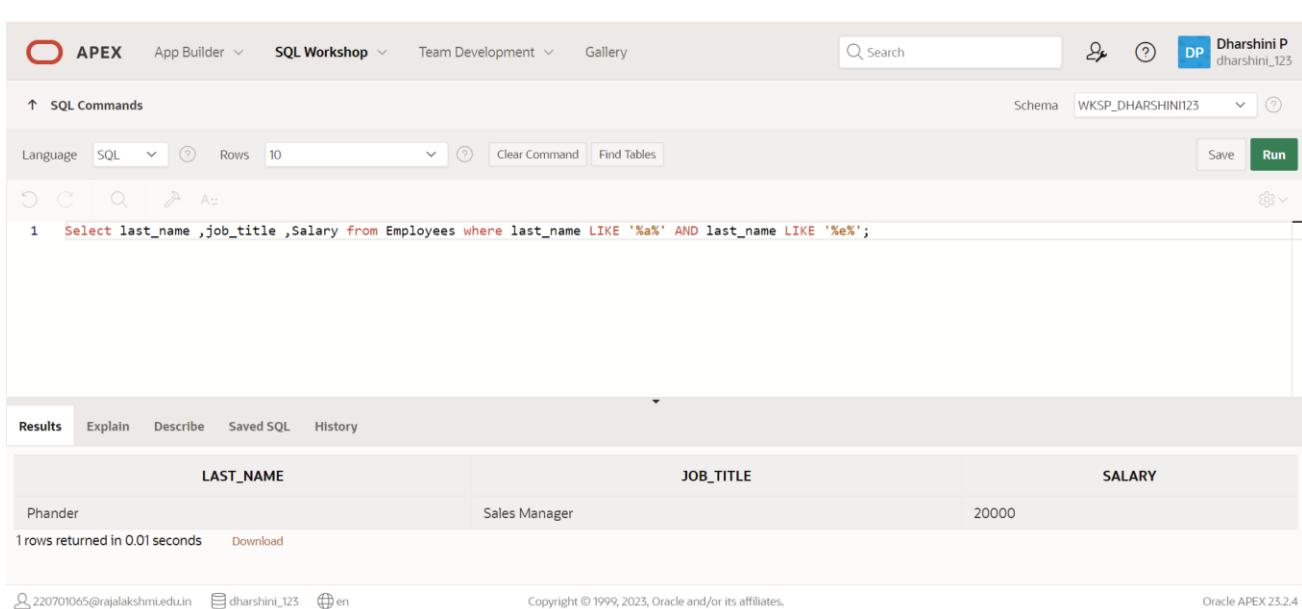
LAST_NAME
Arara
Phander

11. Display the last name of all employees who have an a and an e in their last name.(hints: like)

**QUERY:**

Select last\_name from Employees where last\_name LIKE '%a%' AND last\_name LIKE '%e%';

**OUTPUT:**



The screenshot shows the Oracle APEX SQL Workshop interface. The query entered is "Select last\_name ,job\_title ,Salary from Employees where last\_name LIKE '%a%' AND last\_name LIKE '%e%'". The results pane displays one row: "Phander" with job title "Sales Manager" and salary "20000". The output is timestamped at "1 rows returned in 0.01 seconds".

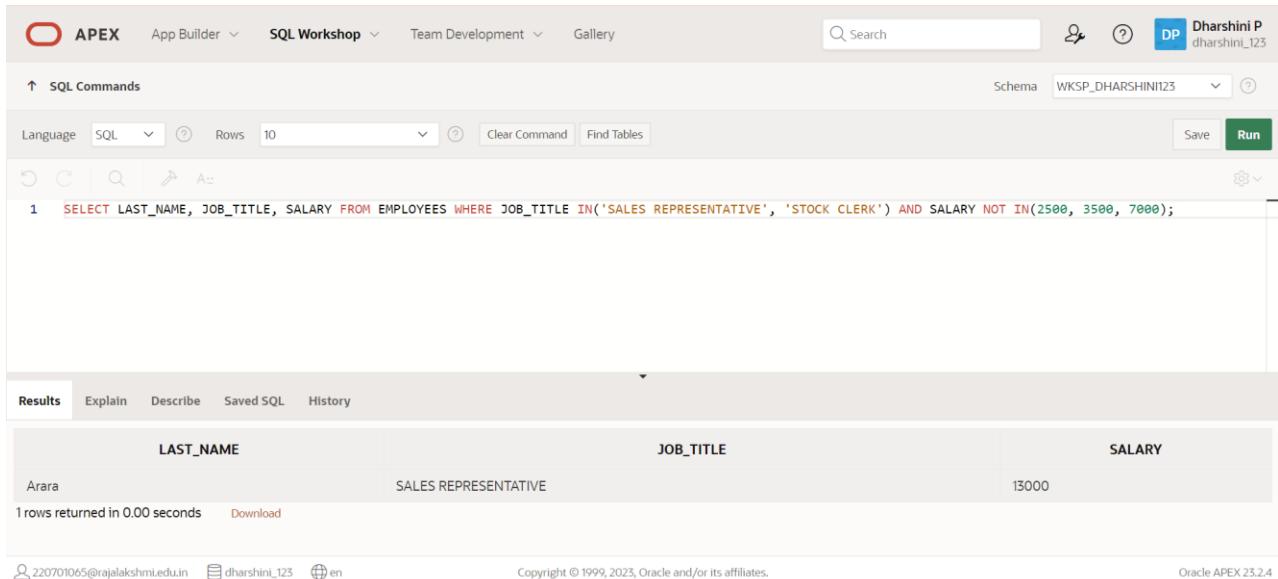
LAST_NAME	JOB_TITLE	SALARY
Phander	Sales Manager	20000

12. Display the last name and job and salary for all employees whose job is sales representative or stock clerk and whose salary is not equal to 2500 ,3500 or 7000.(hints:in,not in)

#### QUERY:

```
SELECT LAST_NAME, JOB_TITLE, SALARY FROM EMPLOYEES WHERE JOB_TITLE IN('SALES REPRESENTATIVE', 'STOCK CLERK') AND SALARY NOT IN(2500, 3500, 7000);
```

#### OUTPUT:



The screenshot shows the Oracle APEX SQL Workshop interface. The query executed is:

```
1  SELECT LAST_NAME, JOB_TITLE, SALARY FROM EMPLOYEES WHERE JOB_TITLE IN('SALES REPRESENTATIVE', 'STOCK CLERK') AND SALARY NOT IN(2500, 3500, 7000);
```

The results table shows one row:

LAST_NAME	JOB_TITLE	SALARY
Arara	SALES REPRESENTATIVE	13000

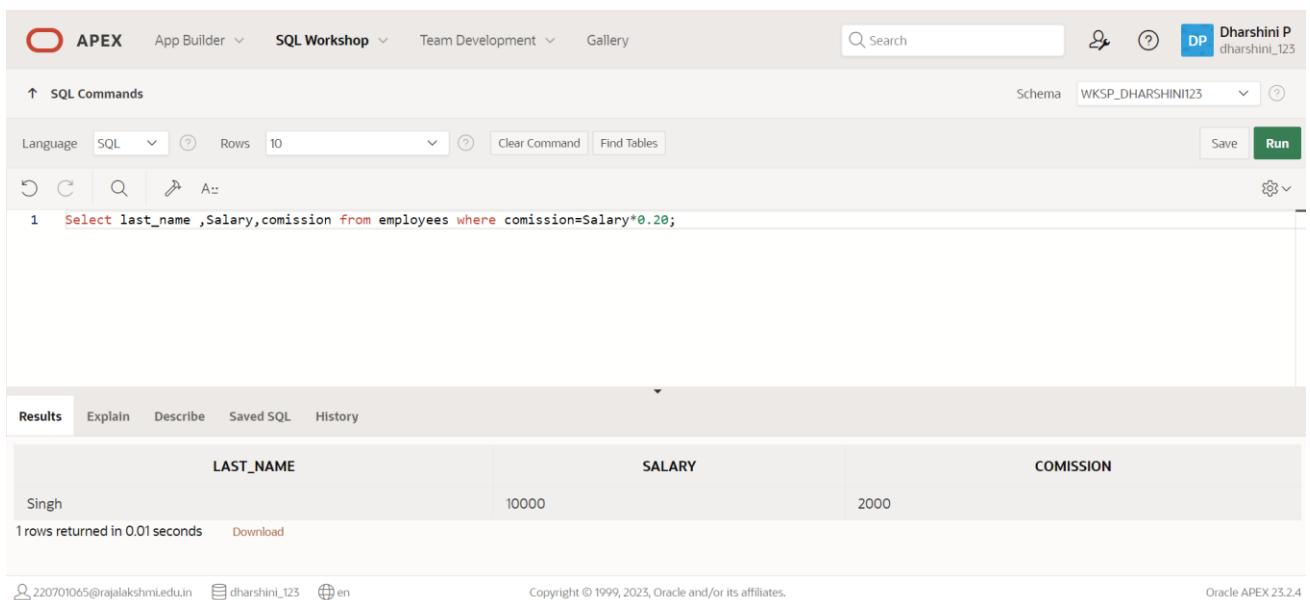
1 rows returned in 0.00 seconds

13. Display the last name, salary, and commission for all employees whose commission amount is 20%.(hints:use predicate logic)

#### QUERY:

```
Select last_name ,Salary,commission from employees where commission=Salary*0.20;
```

#### OUTPUT:



The screenshot shows the Oracle APEX SQL Workshop interface. The query executed is:

```
1  Select last_name ,Salary,comission from employees where comission=Salary*0.20;
```

The results table shows one row:

LAST_NAME	SALARY	COMISSION
Singh	10000	2000

1 rows returned in 0.01 seconds

Evaluation Procedure	Marks awarded
Query(5)	
Execution (5)	
Viva(5)	
Total (15)	
Faculty Signature	

## **RESULT:**

# SINGLE ROW FUNCTIONS

**EX.NO.6**

**DATE:**

**Find the Solution for the following:**

1. Write a query to display the current date. Label the column Date.

**QUERY:**

```
SELECT SYSDATE AS "DATE" FROM DUAL;
```

**OUTPUT:**

The screenshot shows the Oracle APEX SQL Workshop interface. The query `SELECT SYSDATE AS "DATE" FROM DUAL;` is entered in the command line. The results pane displays the output: `DATE` with the value `03/15/2024`. The status bar at the bottom indicates `1 rows returned in 0.02 seconds`.

2. The HR department needs a report to display the employee number, last name, salary, and increase by 15.5% (expressed as a whole number) for each employee. Label the column New Salary.

**QUERY:**

```
SELECT emp_id, last_name, Salary, Salary+(15.5/100*Salary) "NEW_SALARY" FROM EMPLOYEES;
```

**OUTPUT:**

The screenshot shows the Oracle APEX SQL Workshop interface. The query `SELECT emp_id, last_name, Salary, Salary+(15.5/100*Salary) "NEW_SALARY" FROM EMPLOYEES;` is entered in the command line. The results pane displays a report with four columns: `EMP_ID`, `LAST_NAME`, `SALARY`, and `NEW_SALARY`. The data rows are:

EMP_ID	LAST_NAME	SALARY	NEW_SALARY
176	Arara	13000	15015
170	Phander	20000	23100
175	Singh	10000	11550

3. Modify your query lab\_03\_02.sql to add a column that subtracts the old salary from the new salary. Label the column Increase.

**QUERY:**

```
SELECT emp_id, last_name, Salary, (Salary+(Salary*15.5/100))-Salary "Increase" From Employees;
```

**OUTPUT:**

EMP_ID	LAST_NAME	SALARY	Increase
176	Arara	13000	2015
170	Phander	20000	3100
175	Singh	10000	1550

4. Write a query that displays the last name (with the first letter uppercase and all other letters lowercase) and the length of the last name for all employees whose name starts with the letters J, A, or M. Give each column an appropriate label. Sort the results by the employees' last names.

**QUERY:**

```
Select initcap(last_name) "Name", length(last_name) "Length of Name"  
from Employees  
where last_name like 'J%' or last_name like 'A%' or last_name like 'M%'  
order by last_name;
```

**OUTPUT:**

Name	Length of Name
Arara	5
Jhander	7

5.Rewrite the query so that the user is prompted to enter a letter that starts the last name. For example, if the user enters H when prompted for a letter, then the output should show all employees whose last name starts with the letter H.

**QUERY:**

```
select initcap(last_name) "Name", length(last_name) "Length of Name" from employees  
where last_name like 'J%' order by last_name;
```

**OUTPUT:**

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. The right side shows a user profile for 'Dharshini P dharshini\_123'. The main area has tabs for SQL Commands, Results, Explain, Describe, Saved SQL, and History. The SQL Commands tab contains the following code:

```
1 select initcap(last_name) "Name", length(last_name) "Length of Name" from employees  
2 where last_name like 'J%' order by last_name;
```

The Results tab displays the output:

Name	Length of Name
Jhander	7

Below the results, it says '1 rows returned in 0.01 seconds' and provides a 'Download' link. The bottom of the page includes copyright information for Oracle and APEX version 23.2.4.

6.The HR department wants to find the length of employment for each employee. For each employee, display the last name and calculate the number of months between today and the date on which the employee was hired. Label the column MONTHS\_WORKED. Order your results by the number of months employed. Round the number of months up to the closest whole number.

**QUERY:**

```
select last_name, round(months_between(sysdate,hire_date),0) Months_worked  
from employees order by 2;
```

**OUTPUT:**

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. The right side shows a user profile for 'Dharshini P dharshini\_123'. The main area has tabs for SQL Commands, Results, Explain, Describe, Saved SQL, and History. The SQL Commands tab contains the following code:

```
1 select last_name, round(months_between(sysdate,hire_date),0) Months_worked from employees order by 2;
```

The Results tab displays the output:

LAST_NAME	MONTHS_WORKED
Harini	9
Jhander	12
Arara	13

Below the results, it says '3 rows returned in 0.00 seconds' and provides a 'Download' link. The bottom of the page includes copyright information for Oracle and APEX version 23.2.4.

7.Create a report that produces the following for each employee:  
<employee last name> earns <salary> monthly but wants <3 times salary>. Label the column Dream Salaries.

#### QUERY:

Select last\_name||' earns \$'||salary||' monthly but wants \$'||salary\*3 "Dream Salary" from employees;

#### OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The query entered is:

```
1 Select last_name||' earns $'||salary||' monthly but wants $'||salary*3 "Dream Salary" from employees;
```

The results section displays the output:

Dream Salary	
Arara	earns \$13000 monthly but wants \$39000
Jhander	earns \$20000 monthly but wants \$60000
Harini	earns \$10000 monthly but wants \$30000

3 rows returned in 0.01 seconds

8.Create a query to display the last name and salary for all employees. Format the salary to be 15 characters long, left-padded with the \$ symbol. Label the column SALARY.

#### QUERY:

Select last\_name, lpad(salary,15,'\$') Salary from employees;

#### OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The query entered is:

```
1 Select last_name, lpad(salary,15,'$') Salary from employees;
```

The results section displays the output:

LAST_NAME	SALARY
Arara	\$\$\$\$\$\$\$\$\$\$13000
Jhander	\$\$\$\$\$\$\$\$\$\$20000
Harini	\$\$\$\$\$\$\$\$\$\$10000

3 rows returned in 0.01 seconds

9. Display each employee's last name, hire date, and salary review date, which is the first Monday after six months of service. Label the column REVIEW. Format the dates to appear in the format similar to "Monday, the Thirty-First of July, 2000."

**QUERY:**

```
select last_name, hire_date, to_char((next_day(hire_date,'Monday')),'fmday," the  
"ddspth "of" month,yyyy') "REVIEW" from employees;
```

**OUTPUT:**

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. A search bar and a user profile icon for 'Dharshini P' are also present. The main area displays a SQL command in the editor:

```
1 select last_name, hire_date, to_char((next_day(hire_date,'Monday')),'fmday," the  
"ddspth "of" month,yyyy') "REVIEW" from employees;
```

Below the editor, the results tab is selected, showing the output of the query:

LAST_NAME	HIRE_DATE	REVIEW
Arara	02/20/2023	monday, the twenty-seventh of february,2023
Jhander	03/05/2023	monday, the sixth of march,2023
Harini	05/31/2023	monday, the fifth of june,2023

At the bottom of the results page, it says '3 rows returned in 0.00 seconds'. The footer includes copyright information for Oracle and the APEX version.

10. Display the last name, hire date, and day of the week on which the employee started. Label the column DAY. Order the results by the day of the week, starting with Monday.

**QUERY:**

Select Last\_name, hire\_date, to\_char(hire\_date,'Day') "Day" from employees order by to\_char(hire\_date-1,'d');

**OUTPUT:**

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. A search bar and a user profile icon for 'Dharshini P' are also present. The main area displays a SQL command in the editor:

```
1 Select Last_name, hire_date, to_char(hire_date,'Day') "Day" from employees order by to_char(hire_date-1,'d');
```

Below the editor, the results tab is selected, showing the output of the query:

LAST_NAME	HIRE_DATE	Day
Arara	02/20/2023	Monday
Harini	05/31/2023	Wednesday
Jhander	03/05/2023	Sunday

At the bottom of the results page, it says '3 rows returned in 0.01 seconds'. The footer includes copyright information for Oracle and the APEX version.

Evaluation Procedure	Marks awarded
Query(5)	
Execution (5)	
Viva(5)	
Total (15)	
Faculty Signature	

## **RESULT:**

# DISPLAYING DATA FROM MULTIPLE TABLES

**EX\_NO:**7

**DATE:**

1. Write a query to display the last name, department number, and department name for all employees.

**QUERY:**

```
Select e.last_name,e.department_number,d.dept_id from empo21 e,dept23 d where  
e.department_number=d.dept_id;
```

**OUTPUT:**

The screenshot shows the Oracle SQL Workshop interface. In the SQL Commands pane, the following query is entered:

```
1 select e.last_name, e.dept_no , d.dept_name from emp08 e , dept08 d where e.dept_no=d.dept_no;
```

In the Results pane, the output is displayed as a table:

LAST_NAME	DEPT_NO	DEPT_NAME
lia	1	cse
jeln	2	cse

2 rows returned in 0.01 seconds

2. Create a unique listing of all jobs that are in department 80. Include the location of the department in the output.

**QUERY:**

```
select distinct job_id,loc_id from empo21 e,dept23 d where e.department_number=d.dept_id and  
e.department_number=80;
```

**OUTPUT:**

The screenshot shows the Oracle SQL Workshop interface. In the SQL Commands pane, the following query is entered:

```
1 select distinct job_id, loc_id from emp08 e, dept08 d where e.dept_no =d.dept_no and e.dept_no =80;
```

In the Results pane, the output is displayed as a table:

JOB_ID	LOC_ID
11	22

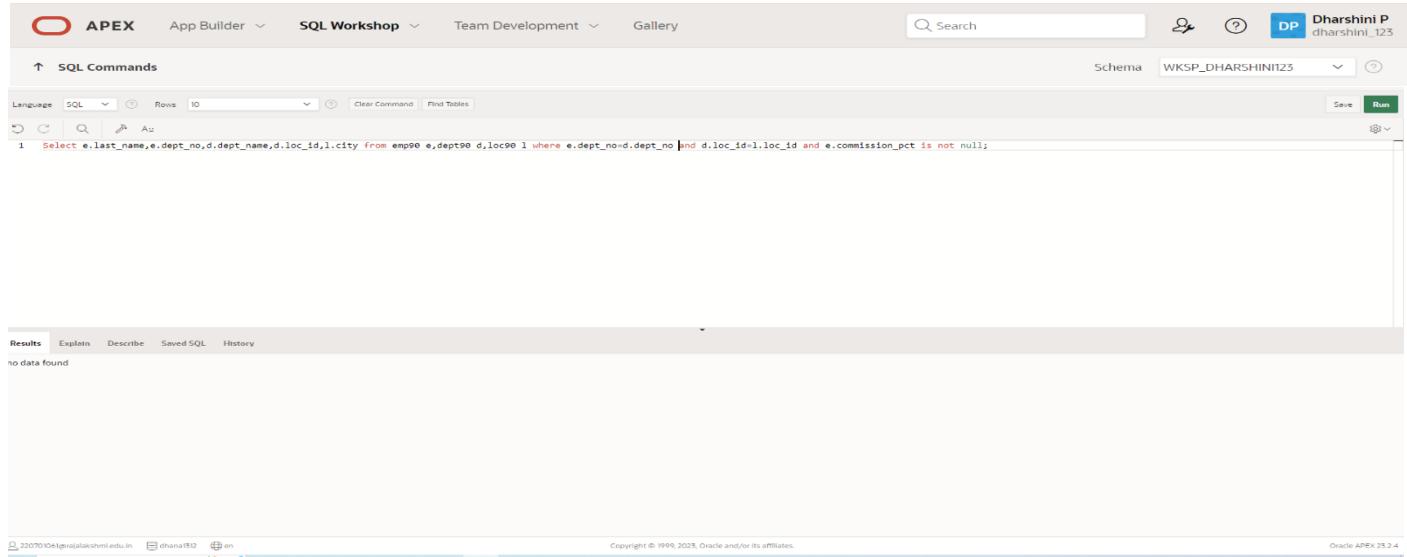
1 rows returned in 0.00 seconds

3. Write a query to display the employee last name, department name, location ID, and city of all employees who earn a commission

#### QUERY:

```
Select e.last_name,e.department_number,d.dept_name,d.loc_id,l.city from empo21 e,dept23 d,location l  
where e.department_number=d.dept_id and d.loc_id=l.location_id and e.commission_pct is not null;
```

#### OUTPUT:



```
1 Select e.last_name,e.dept_no,d.dept_name,d.loc_id,l.city from empo21 e,dept23 d,location l where e.dept_no=d.dept_no and d.loc_id=l.location_id and e.commission_pct is not null;
```

Results Explain Describe Saved SQL History

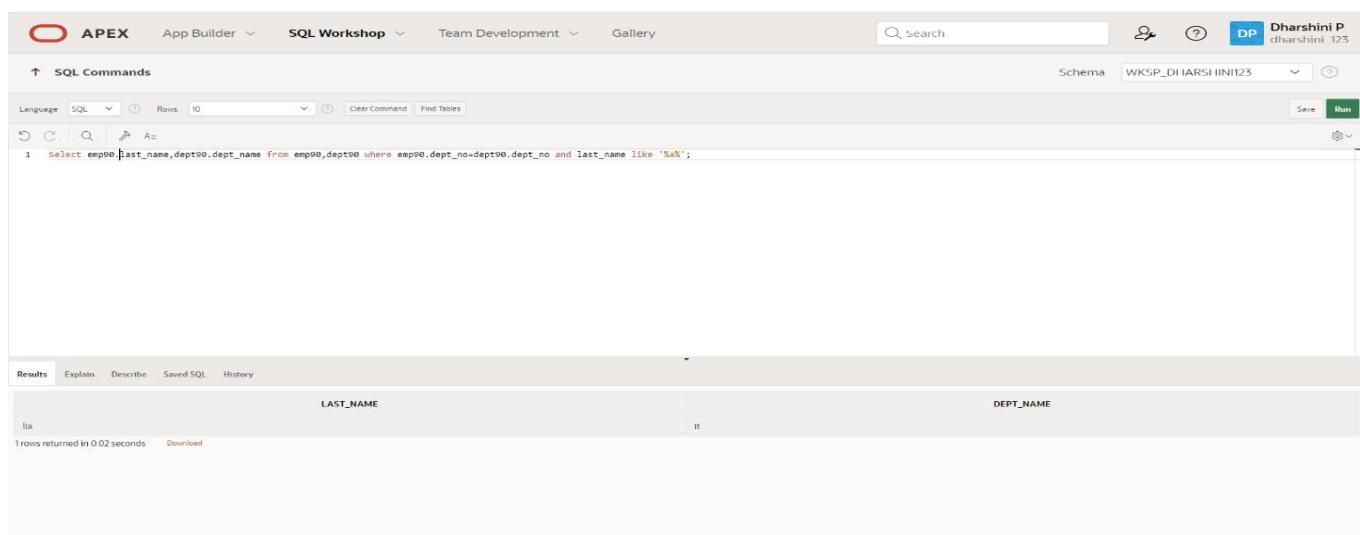
no data found

4. Display the employee last name and department name for all employees who have an a(lowercase) in their last names.

#### QUERY:

```
Select empo21.last_name,dept23.dept_name from empo21,dept23  
where empo21.department_number=dept23.dept_id and last_name like '%a%';
```

#### OUTPUT:



```
1 Select emp01.last_name,dept00.dept_name from emp01,dept00 where emp01.dept_no=dept00.dept_no and last_name like '%a%';
```

LAST_NAME	DEPT_NAME
It	IT

1 rows returned in 0.02 seconds Download

5. Write a query to display the last name, job, department number, and department name for all employees who work in Toronto.

### QUERY:

```
Select e.last_name,e.department_number,e.job_id,d.dept_name from emp021 e join dept d  
on(e.department_number=d.dept_id) join location on (d.location_id=location.location_id) where  
lower(location.city)=’toronto’;
```

### OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The query entered is:

```
Select e.last_name,e.department_number,e.job_id,d.dept_name from emp021 e join dept d  
on(e.department_number=d.dept_id) join location on (d.location_id=location.location_id) where  
lower(location.city)=’toronto’;
```

The results section displays one row:

LAST_NAME	DEPT_NO	JOB_ID	DEPT_NAME
Leijn	2	22	CSE

Copyright © 1999, 2023, Oracle and/or its affiliates. Oracle APEX 23.2.4

6. Display the employee last name and employee number along with their manager’s last name and manager number. Label the columns Employee, Emp#, Manager, and Mgr#, Respectively

### QUERY:

```
Select w.last_name “Employee”,w.emp_id “emp#”,m.last_name ‘manager’,m.emp_id “Mgr#” from emp021  
m on (w.manager_id=m.emp_id);
```

### OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The query entered is:

```
Select w.last_name “Employee”,w.emp_id “emp#”,m.last_name ‘manager’,m.emp_id “Mgr#” from emp021 w JOIN emp021 m on (w.manager_id=m.emp_id);
```

The results section displays one row:

LAST_NAME	DEPT_NAME
Ila	IT

Copyright © 1999, 2023, Oracle and/or its affiliates. Oracle APEX 23.2.4

7. Modify lab4\_6.sql to display all employees including King, who has no manager. Order the results by the employee number.

#### QUERY:

```
Select w.last_name "Employee", w.emp_id "emp#", m.last_name 'manager', m.emp_id "Mgr#" from empo21  
w left outer join empo21 m on (w.manager_id=m.emp_id);
```

#### OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The SQL Commands tab contains the following SQL code:

```
1 SELECT W.LAST_NAME "EMPLOYEE", W.EMP_ID "EMP#",  
2 M.LAST_NAME "MANAGER", M.EMP_ID "MGR#"  
3 FROM EMP90 W  
4 LEFT OUTER JOIN EMP90 M  
5 ON (W.MANAGER_ID = M.EMP_ID)  
6 ORDER BY W.EMP_ID;  
7
```

The Results tab displays the output:

EMPLOYEE	EMP#	MANAGER	MGR#
lia	70	-	-
liu-biung	-	-	-
jejin	-	-	-

3 rows returned in 0.02 seconds [Download](#)

8. Create a query that displays employee last names, department numbers, and all the employees who work in the same department as a given employee. Give each column an appropriate label

#### QUERY:

```
select e.department_number dept23,e.last_name colleague from empo21 e join empo21 c on  
(e.department_number=c.department_number) where e.emp_id <> c.emp_id order by  
e.department_number,e.last_name,c.last_name;
```

#### OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The SQL Commands tab contains the following SQL code:

```
1 SELECT E.LAST_NAME EMPLOYEE, E.DEPT_NO DEPARTMENT,  
2 C.LAST_NAME COLLEAGUE  
3 FROM EMP90 E JOIN EMP90 C  
4 ON (E.DEPT_NO = C.DEPT_NO)  
5 WHERE E.EMP_ID <> C.EMP_ID  
6 ORDER BY E.DEPT_NO, E.LAST_NAME;
```

The Results tab displays the output:

EMPLOYEE	DEPARTMENT	COLLEAGUE
lia	70	-
liu-biung	-	-
jejin	-	-

3 rows returned in 0.02 seconds [Download](#)

9. Show the structure of the JOB\_GRADES table. Create a query that displays the name, job, department name, salary, and grade for all employees

### QUERY:

```
SELECT e.last_name, e.job_id, d.dept_name, e.salary, j.grade_level
FROM emp18 e JOIN dept18 d
ON (e.dept_id = d.dept_id)
JOIN job_grade j
ON (e.salary BETWEEN j.lowest_sal AND j.highest_sal);
```

### OUTPUT:

The screenshot shows the Oracle APEX interface with the SQL Workshop tab selected. The query from step 9 is pasted into the command editor. The results pane displays three rows of data:

EMPLOYEE	EMP#	MANAGER	MGR#
Ila	70	-	-
Iku-blung	-	-	-
Jin	-	-	-

3 rows returned in 0.02 seconds

10. Create a query to display the name and hire date of any employee hired after employee Davies.

### QUERY:

```
SELECT e.last_name, e.hire_date
FROM emp18 e, emp18 davies
WHERE davies.last_name = 'Davies'
AND davies.hire_date < e.hire_date;
```

### OUTPUT:

The screenshot shows the Oracle APEX interface with the SQL Workshop tab selected. The query from step 10 is pasted into the command editor. The results pane displays one row of data:

LAST_NAME	DEPT_NAME
Ila	IT

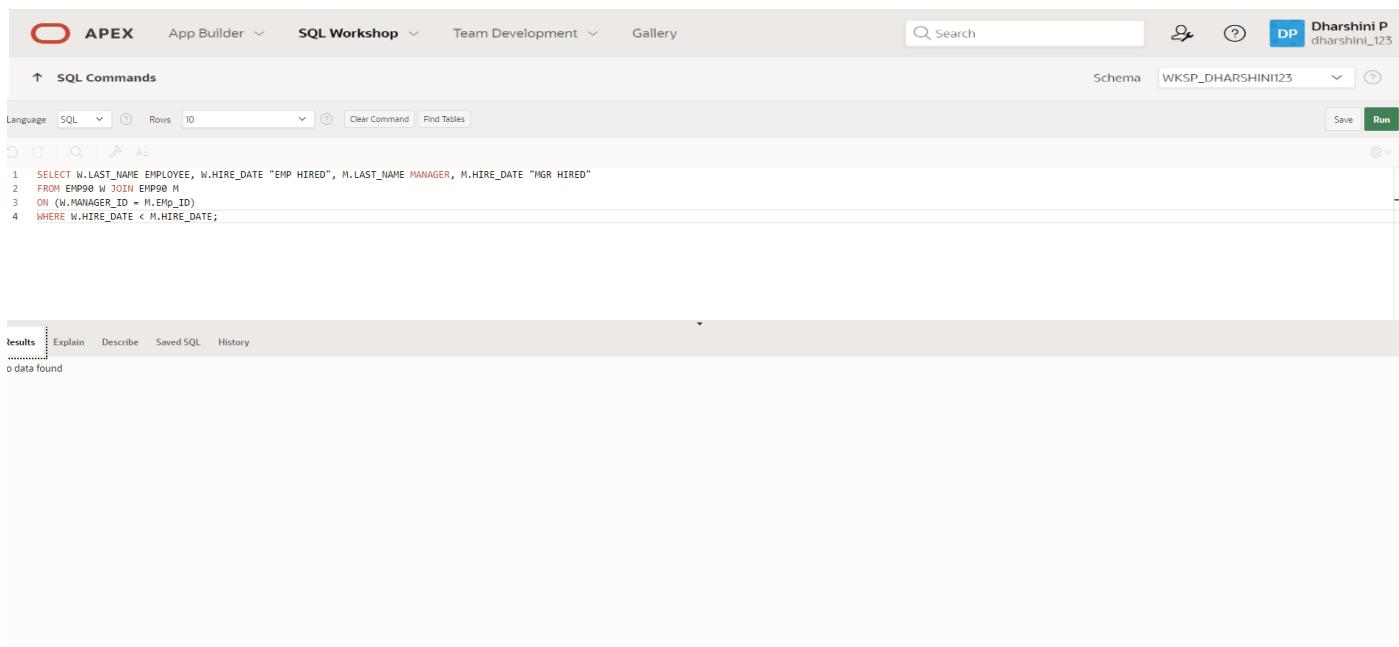
1 rows returned in 0.04 seconds

11. Display the names and hire dates for all employees who were hired before their managers, along with their manager's names and hire dates. Label the columns Employee, Emp Hired, Manager, and Mgr Hired, respectively.

### QUERY:

```
SELECT e.last_name AS Employee, e.hire_date AS Emp_Hired,
e.manager_name AS Manager, m.hire_date AS Mgr_Hired
FROM emp18 e
JOIN emp18|m ON e.manager_name = m.last_name
WHERE e.hire_date < m.hire_date;
```

### OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows the user Dharshini P with schema WKSP\_DHARSHINI123. The main area is titled "SQL Commands" with a "Language" dropdown set to SQL. The query editor contains the following code:

```
1 SELECT W.LAST_NAME EMPLOYEE, W.HIRE_DATE "EMP HIRED", M.LAST_NAME MANAGER, M.HIRE_DATE "MGR HIRED"
2 FROM EMP98 W JOIN EMP98 M
3 ON (W.MANAGER_ID = M.EMP_ID)
4 WHERE W.HIRE_DATE < M.HIRE_DATE;
```

The results tab is selected, showing the message "0 data found".

Evaluation Procedure	Marks awarded
Query(5)	
Execution (5)	
Viva(5)	
Total (15)	
Faculty Signature	

### RESULT

# AGGREGATING DATA USING GROUP FUNCTIONS

EX\_NO : 8

DATE:

1. Group functions work across many rows to produce one result per group.  
True/False

**TRUE**

2. Group functions include nulls in calculations.  
True/False

**FALSE**

3. The WHERE clause restricts rows prior to inclusion in a group calculation.  
True/False

**FALSE**

4. Find the highest, lowest, sum, and average salary of all employees. Label the columns Maximum, Minimum, Sum, and Average, respectively. Round your results to the nearest whole number

QUERY:

```
select Round(Max (salary),0)"Maximum", Round (Min (salary),0) "Minimum",
round(sum(salary),0)"sum", round (avg(salary),0) "Average" from EMPB;
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows a user profile for Dharshini P dharshini\_123. The main area is titled 'SQL Commands' with a schema dropdown set to WKSP\_DHARSHINI123. The command input field contains the following SQL query:

```
1 select round(MAX(Salary),0)"MAXIMUM",round(MIN(SALARY),0)"MINIMUM",ROUND(SUM(SALARY),0)"SUM",ROUND(AVG(SALARY),0)"AVERAGE" FROM EMPB;
```

The results section shows the output of the query:

MAXIMUM	MINIMUM	SUM	AVERAGE
45000	12000	91000	30333

Below the table, it says "1 rows returned in 0.01 seconds" and has a "Download" link.

5.Modify the above query to display the minimum, maximum, sum, and average salary for each job type.

### QUERY:

```
select job_id ,Round(MAX(salary),0) "MAXIMUM",Round (Min(salary),0)"Minimum",Round  
(SUM(Salary),0)"sum" ,Round (AVG (salary),0)"average" from EMPB group by job_id;
```

### OUTPUT:

The screenshot shows the Oracle APEX interface with the SQL Workshop tab selected. A single SQL command is entered in the command editor:

```
1 SELECT JOB_ID,ROUND(MAX(SALARY),0)"MAXIMUM",ROUND(MIN(SALARY),0)"MINIMUM",ROUND(SUM(SALARY),0)"SUM",ROUND(AVG(SALARY),0)"AVERAGE" FROM EMPA GROUP BY JOB_ID;
```

The results are displayed in a grid format:

JOB_ID	MAXIMUM	MINIMUM	SUM	AVERAGE
44	34000	34000	34000	34000
65	-	-	-	-
46	12000	12000	12000	12000
47	45000	45000	45000	45000

Below the results, it says "4 rows returned in 0.00 seconds".

6.Write a query to display the number of people with the same job. Generalize the query so that the user in the HR department is prompted for a job title.

### QUERY:

```
select job_id, count(*) from EMPB group by job_id ;  
select job_id, count(*) from EMPB where job_id='47' group by job_id ;
```

### OUTPUT:

The screenshot shows the Oracle APEX interface with the SQL Workshop tab selected. A single SQL command is entered in the command editor:

```
1 SELECT JOB_ID COUNT(*) FROM EMPA GROUP BY JOB_ID;
```

The results are displayed in a grid format:

MAXIMUM	MINIMUM	SUM	AVERAGE
45000	12000	91000	30333

Below the results, it says "1 rows returned in 0.00 seconds".

7. Determine the number of managers without listing them. Label the column Number of Managers. Hint: Use the MANAGER\_ID column to determine the number of managers.

### QUERY:

```
select count(distinct manager_id )"Number of managers" from empb;
```

### OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. A search bar and user profile 'Dharshini P dharshini\_123' are also present. The main area is titled 'SQL Commands' with a sub-section 'Number of managers'. The query 'select count(distinct manager\_id )"Number of managers" from empb;' is entered in the command line. The results section shows a single row with the value '8' under the heading 'Number of managers'. The status bar at the bottom indicates '1 rows returned in 0.01 seconds'.

8. Find the difference between the highest and lowest salaries. Label the column DIFFERENCE

### QUERY:

```
select max(salary)-min(salary) difference from empb;
```

### OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. A search bar and user profile 'Dharshini P dharshini\_123' are also present. The main area is titled 'SQL Commands' with a sub-section 'DIFFERENCE'. The query 'select max(salary)-min(salary) difference from empb;' is entered in the command line. The results section shows a single row with the value '65000' under the heading 'DIFFERENCE'. The status bar at the bottom indicates '1 rows returned in 0.01 seconds'.

9.Create a report to display the manager number and the salary of the lowest-paid employee for that manager. Exclude anyone whose manager is not known. Exclude any groups where the minimum salary is \$6,000 or less. Sort the output in descending order of salary.

#### QUERY:

```
select manager_id ,MIN(salary) from empb where manager_id is not null group by manager_id having min(salary) >6000 order by min(salary) desc;
```

#### OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The SQL Commands tab contains the following query:

```
1 selectv manager_id ,min(salary) from empb where manger_id is not null group by manager_id having min(salary) >6000 order by min(salary) desc;
```

The Results tab displays the output:

DIFFERENCE
65000

1 rows returned in 0.01 seconds

10.Create a query to display the total number of employees and, of that total, the number of employees hired in 1995, 1996, 1997, and 1998. Create appropriate column headings

#### QUERY:

```
select count(*) total ,sum(decode(to_char(hire_date,'YYYY'),1995,1,0))"1995",sum(decode(to_char(hire_date,'YYYY'),1996,1,0))"1996",sum(decode(to_char(hire_date,'YYYY'),1997,1,0))"1997",sum(decode(to_char(hire_date,'YYYY'),1998,1,0)) "1998" from empb;
```

#### OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The SQL Commands tab contains the following query:

```
1 select count(*) total ,sum(decode(to_char(hire_date,'YYYY'),1995,1,0))"1995",
2 sum(decode(to_char(hire_date,'YYYY'),1996,1,0))"1996",sum(decode(to_char(hire_date,'YYYY'),1997,1,0))"1997",sum(decode(to_char(hire_date,'YYYY'),1998,1,0)) "1998" from empb;
```

The Results tab displays the output:

TOTAL	1995	1996	1997	1998
4	1	0	1	0

1 rows returned in 0.01 seconds

11. Create a matrix query to display the job, the salary for that job based on department number, and the total salary for that job, for departments 20, 50, 80, and 90, giving each column an appropriate heading

### QUERY:

```
select job_id "job", sum(decode(dept_id,20,salary))"Dept20",sum (decode(dept_id ,50, salary))  
"dept50",sum (decode(dept_id ,80, salary)) "dept80",sum (decode(dept_id ,90, salary)) "dept90",sum(salary)  
"TOTAL" from empb group by job_id
```

### OUTPUT:

job	Dept20	dept50	dept80	dept90	TOTAL
44	-	-	-	-	35000
55	-	-	-	-	80000
46	-	-	15000	-	15000
47	-	-	-	-	60000

12. Write a query to display each department's name, location, number of employees, and the average salary for all the employees in that department. Label the column name-Location, Number of people, and salary respectively. Round the average salary to two decimal places.

### QUERY:

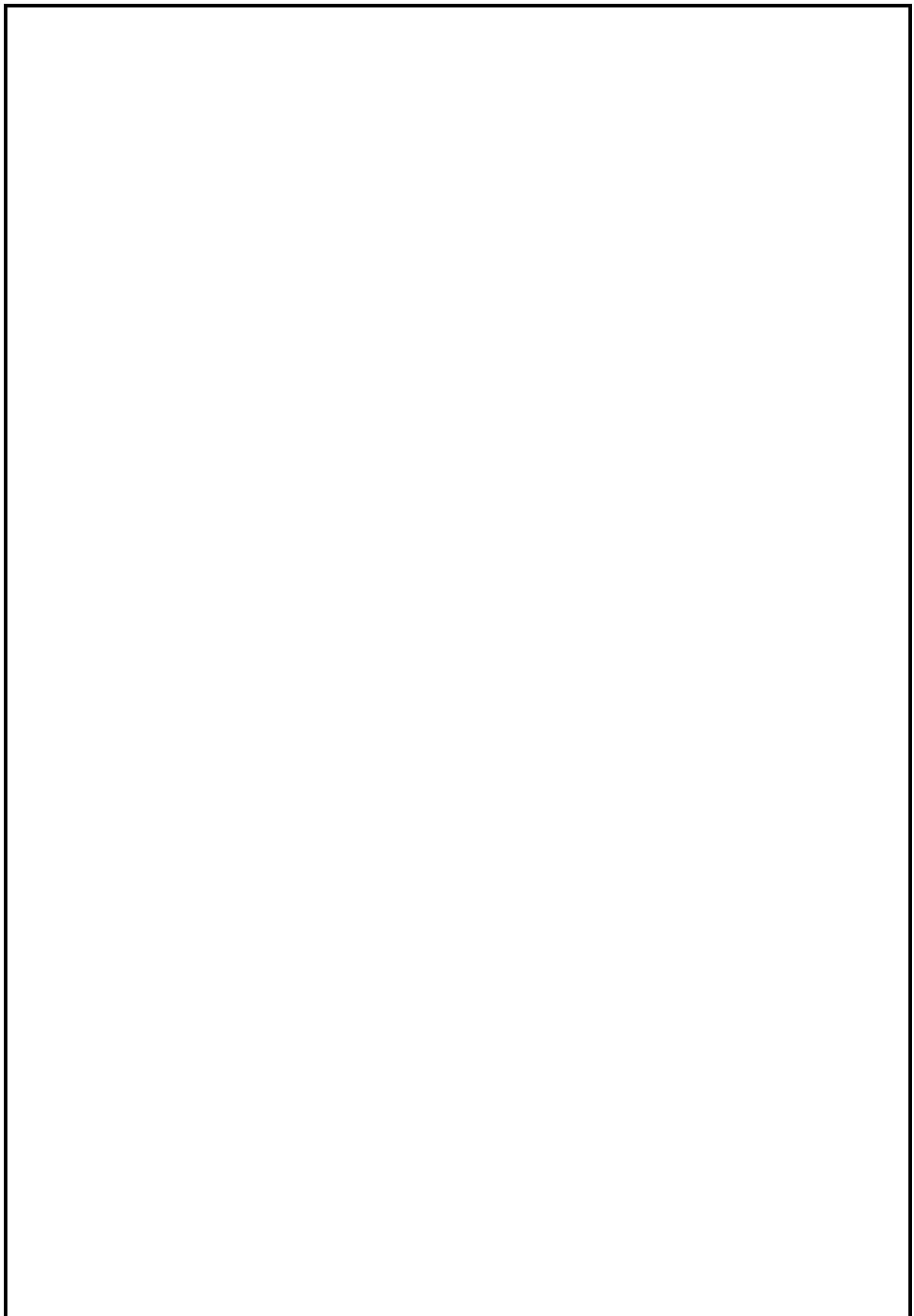
```
select d.dept_name as "dept_name",d.loc as "department location", count(*) "Number of  
people",round(avg(salary),2) "salary" from dept111 d inner join empb e on(d.dpt_id =e.dept_id ) group by  
d.dept_name ,d.loc;
```

### OUTPUT:

dept_name	department location	Number of people	salary
cs	chennai	1	15000
it	mumbai	1	60000
cs	benglore	1	80000
ft	tomato	1	35000

Evaluation Procedure	Marks awarded
Query(5)	
Execution (5)	
Viva(5)	
Total (15)	
Faculty Signature	

**RESULT:**



## SUB-QUERIES

**EX.NO:9**

**DATE:**

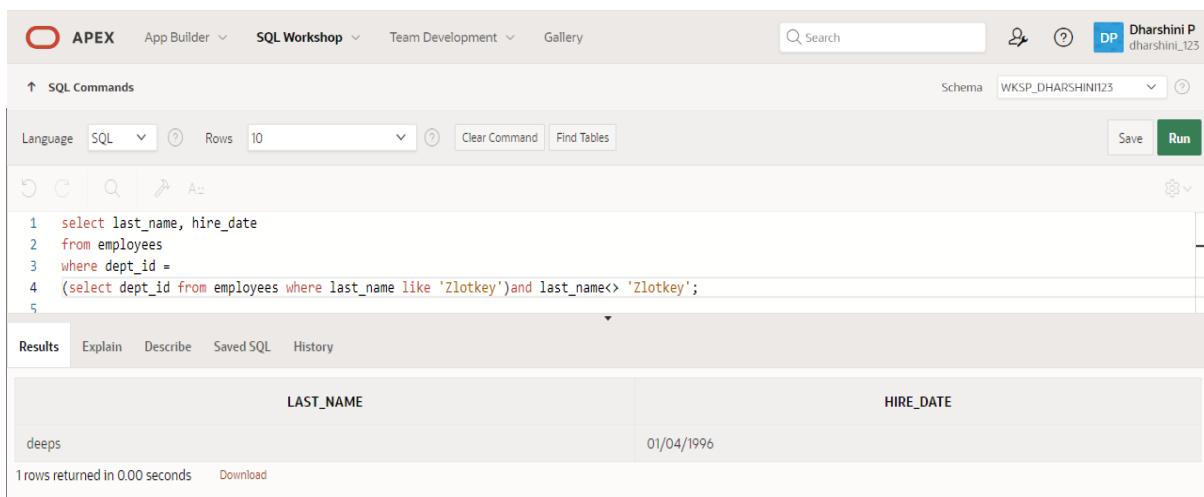
Find the Solution for the following:

1. The HR department needs a query that prompts the user for an employee last name. The query then displays the last name and hire date of any employee in the same department as the employee whose name they supply (excluding that employee). For example, if the user enters Zlotkey, find all employees who work with Zlotkey (excluding Zlotkey).

### **QUERY:**

```
Select last_name, hire_date  
from employees  
where dept_id =  
(select dept_id from employees where last_name like 'Zlotkey')and last_name  
<> 'Zlotkey';
```

### **OUTPUT:**



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, and Gallery. On the right, there's a user profile for 'Dharshini P' with a session ID 'dharshini\_123'. The main workspace has a toolbar with icons for Undo, Redo, Search, and Run. Below the toolbar, the schema is set to 'WKSP\_DHARSHINI123'. The SQL command window contains the following code:

```
1 select last_name, hire_date  
2 from employees  
3 where dept_id =  
4 (select dept_id from employees where last_name like 'Zlotkey')and last_name<> 'Zlotkey';  
5
```

The results window shows a single row of data:

LAST_NAME	HIRE_DATE
deeps	01/04/1996

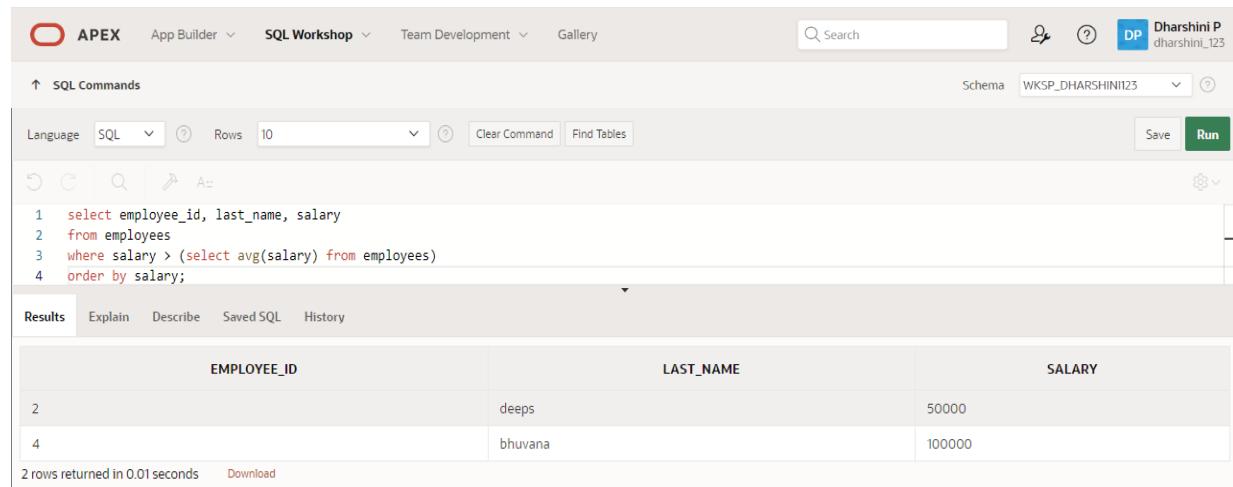
Below the results, it says '1 rows returned in 0.00 seconds'.

2. Create a report that displays the employee number, last name, and salary of all employees who earn more than the average salary. Sort the results in order of ascending salary.

### QUERY:

```
select employee_id, last_name, salary  
from employees  
where salary > (select avg(salary) from employees) order by salary;
```

### OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The SQL command entered is:

```
1 select employee_id, last_name, salary  
2   from employees  
3  where salary > (select avg(salary) from employees)  
4  order by salary;
```

The results table shows two rows:

EMPLOYEE_ID	LAST_NAME	SALARY
2	deeps	50000
4	bhuvana	100000

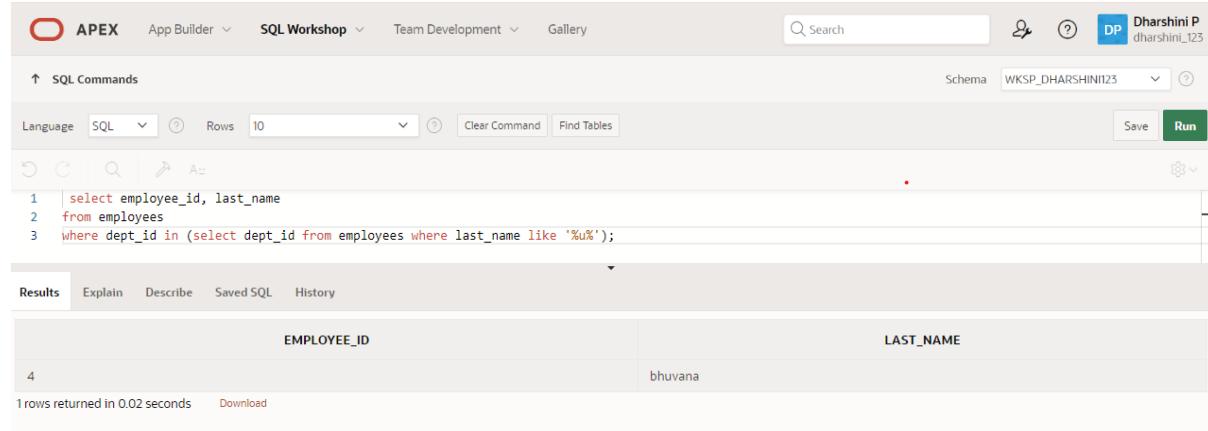
2 rows returned in 0.01 seconds

3. Write a query that displays the employee number and last name of all employees who work in a department with any employee whose last name contains a u.

### QUERY:

```
select employee_id, last_name from employees  
where dept_id in (select dept_id from employees where last_name like '%u%');
```

### OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The SQL command entered is:

```
1 | select employee_id, last_name  
2   from employees  
3  where dept_id in (select dept_id from employees where last_name like '%u%');
```

The results table shows one row:

EMPLOYEE_ID	LAST_NAME
4	bhuvana

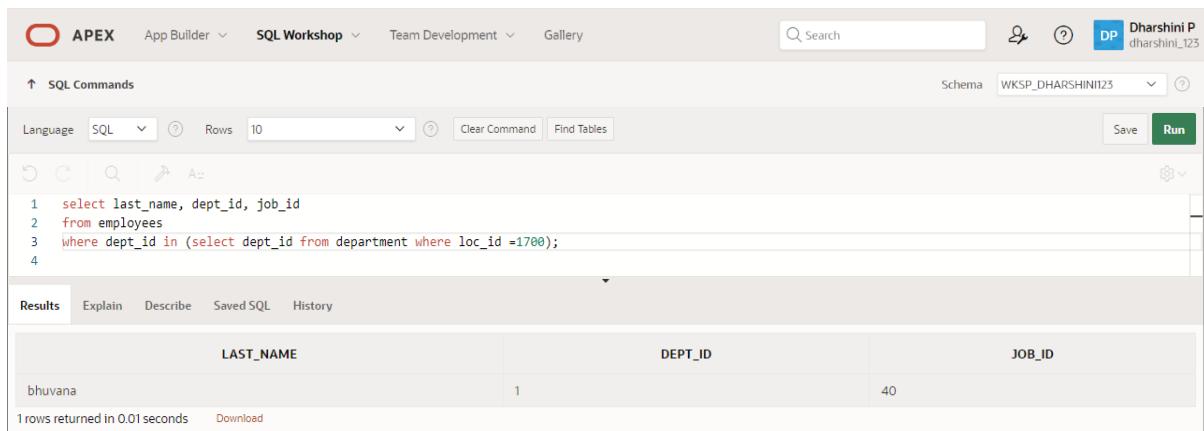
1 rows returned in 0.02 seconds

4. The HR department needs a report that displays the last name, department number, and job ID of all employees whose department location ID is 1700.

### QUERY:

```
select last_name, dept_id, job_id  
from employees  
where dept_id in (select dept_id from department where loc_id =1700);
```

### OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, and Gallery. The right side shows a user profile for 'Dharshini P' with the schema 'WKSP\_DHARSHINI123'. The main area has tabs for SQL Commands, Results, Explain, Describe, Saved SQL, and History. The SQL Commands tab contains the following query:

```
1 select last_name, dept_id, job_id  
2 from employees  
3 where dept_id in (select dept_id from department where loc_id =1700);  
4
```

The Results tab displays the output:

LAST_NAME	DEPT_ID	JOB_ID
bhuvana	1	40

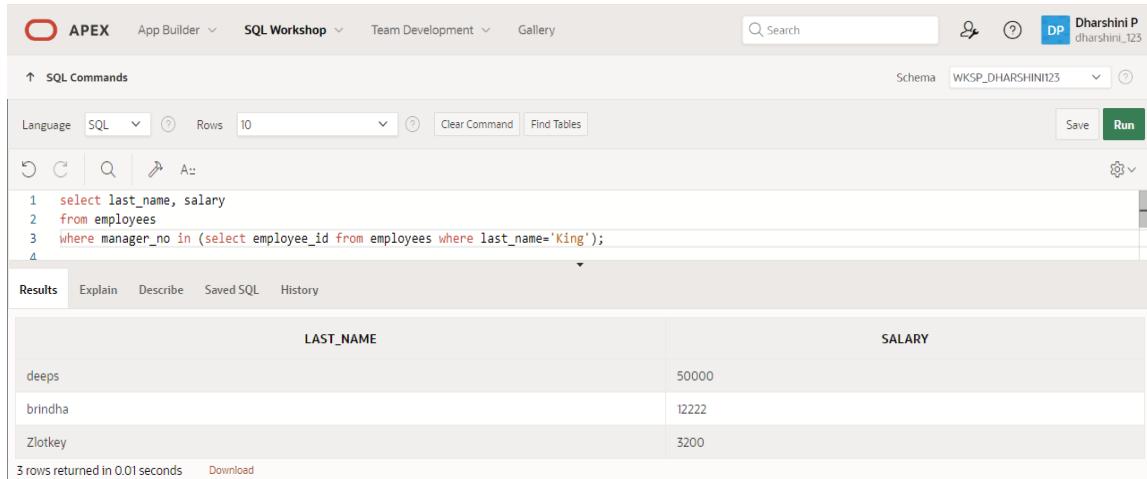
1 rows returned in 0.01 seconds

5. Create a report for HR that displays the last name and salary of every employee who reports to King.

### QUERY:

```
select last_name, salary  
from employees where manager_no in (select employee_id from employees  
where last_name='King');
```

### OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, and Gallery. The right side shows a user profile for 'Dharshini P' with the schema 'WKSP\_DHARSHINI123'. The main area has tabs for SQL Commands, Results, Explain, Describe, Saved SQL, and History. The SQL Commands tab contains the following query:

```
1 select last_name, salary  
2 from employees  
3 where manager_no in (select employee_id from employees where last_name='King');  
4
```

The Results tab displays the output:

LAST_NAME	SALARY
deepa	50000
brindha	12222
Zlotkey	3200

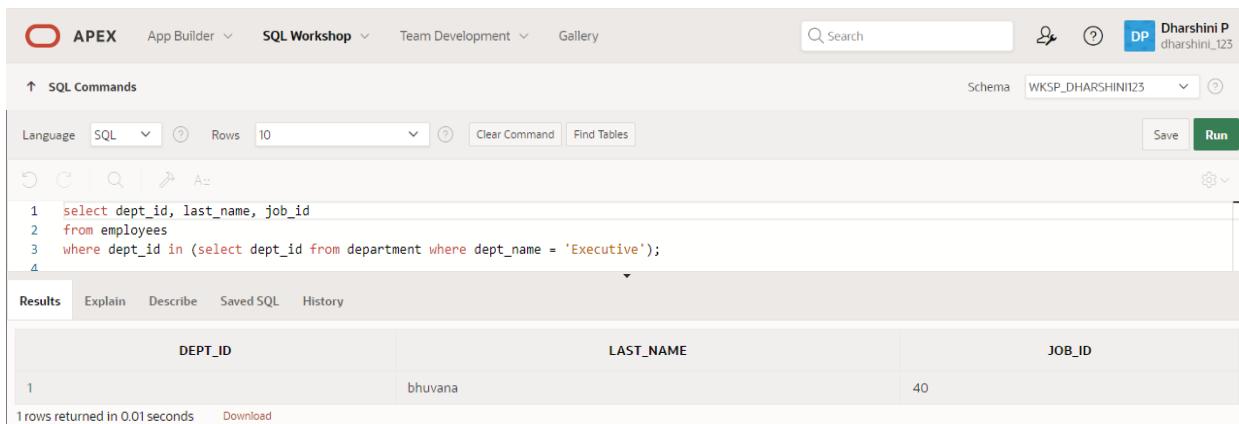
3 rows returned in 0.01 seconds

6. Create a report for HR that displays the department number, last name, and job ID for every employee in the Executive department.

### QUERY:

```
select dept_id, last_name, job_id  
from employees  
where dept_id in (select dept_id from department where dept_name =  
'Executive');
```

### OUTPUT :



The screenshot shows the Oracle SQL Workshop interface. The query window contains the following SQL code:

```
1 select dept_id, last_name, job_id  
2   from employees  
3  where dept_id in (select dept_id from department where dept_name =  
4 'Executive');
```

The results pane shows a single row of data:

DEPT_ID	LAST_NAME	JOB_ID
1	bhuvana	40

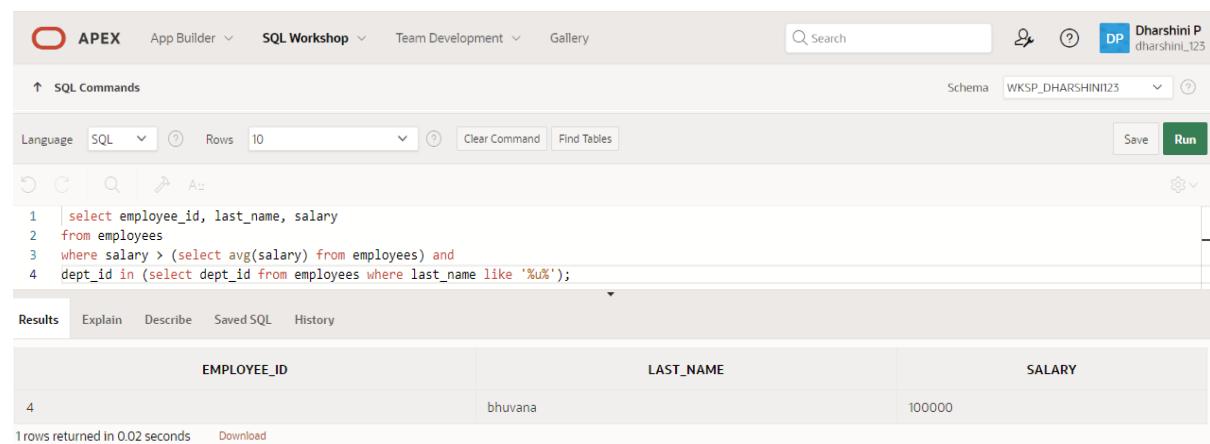
1 rows returned in 0.01 seconds

7. Modify the query 3 to display the employee number, last name, and salary of all employees who earn more than the average salary and who work in a department with any employee whose last name contains a u.

### QUERY:

```
select employee_id, last_name, salary  
from employees where salary > (select avg(salary) from employees) and  
dept_id in (select dept_id from employees where last_name like '%u%');
```

### OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The query window contains the following SQL code:

```
1 | select employee_id, last_name, salary  
2 |   from employees  
3 | where salary > (select avg(salary) from employees) and  
4 | dept_id in (select dept_id from employees where last_name like '%u%');
```

The results pane shows a single row of data:

EMPLOYEE_ID	LAST_NAME	SALARY
4	bhuvana	100000

1 rows returned in 0.02 seconds

Evaluation Procedure	Marks Awarded
Query(5)	
Execution (5)	
Viva(5)	
Total (15)	
Faculty Signature	

**RESULT:**

## USING THE SET OPERATORS

EX.NO:10

DATE:

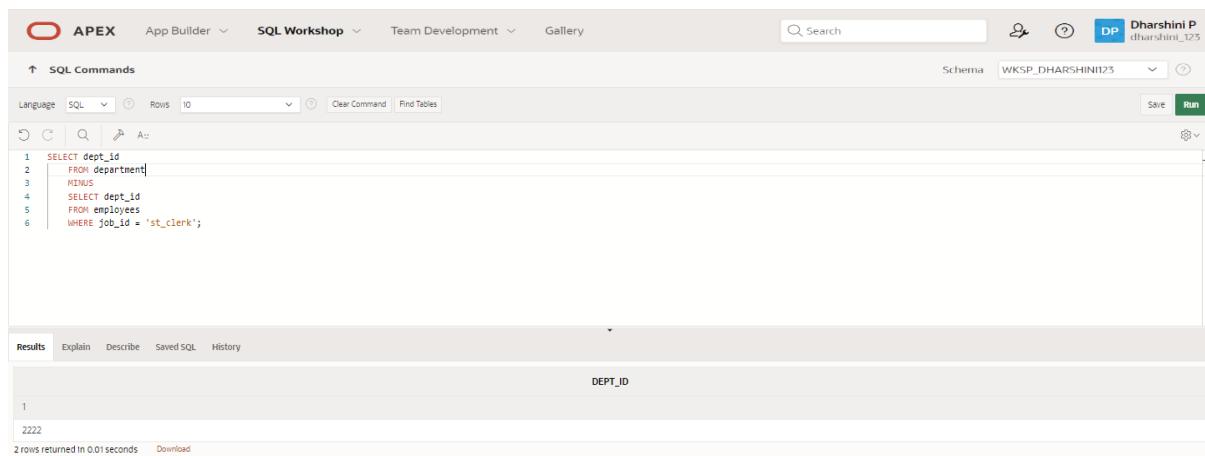
Find the Solution for the following:

1. The HR department needs a list of department IDs for departments that do not contain the job ID ST\_CLERK. Use set operators to create this report.

**QUERY:**

```
SELECT dept_id
FROM department
MINUS
SELECT dept_id
FROM employees
WHERE job_id = 'st_clerk';
```

**OUTPUT:**



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there's a user profile for 'Dharshini P' (dharsinhir\_123). The main area has tabs for SQL Commands, Results, Explain, Describe, Saved SQL, and History. The SQL Commands tab contains the following code:

```
1 SELECT dept_id
2   FROM department
3 MINUS
4   SELECT dept_id
5     FROM employees
6    WHERE job_id = 'st_clerk';
```

The Results tab displays the output:

DEPT_ID
1
2222

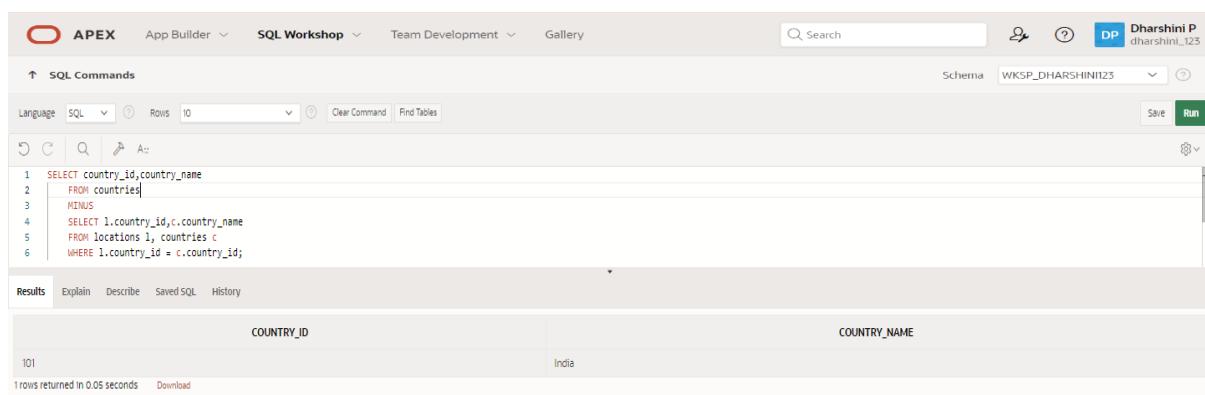
Below the table, it says "2 rows returned in 0.01 seconds".

2. The HR department needs a list of countries that have no departments located in them. Display the country ID and the name of the countries. Use set operators to create this report.

### QUERY:

```
SELECT country_id,country_name
FROM countries
MINUS
SELECT l.country_id,c.cOUNTRY_NAME
FROM locations l, countries c
WHERE l.country_id = c.country_id;
```

### OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows a user profile for 'Dharshini P dharshini\_123'. The main area has tabs for SQL Commands, Results, Explain, Describe, Saved SQL, and History. The SQL Commands tab displays the following query:

```
1 SELECT country_id,country_name
2   FROM countries
3
4 MINUS
5
6 SELECT l.country_id,c.cOUNTRY_NAME
7   FROM locations l, countries c
8 WHERE l.country_id = c.country_id;
```

The Results tab shows the output:

COUNTRY_ID	COUNTRY_NAME
101	India

Below the table, it says '1 rows returned in 0.05 seconds' and has a 'Download' link.

3. Produce a list of jobs for departments 10, 50, and 20, in that order. Display job ID and department ID using set operators.

#### QUERY:

```
SELECT DISTINCT job_no, dept_id  
FROM employees  
WHERE dept_id = 10
```

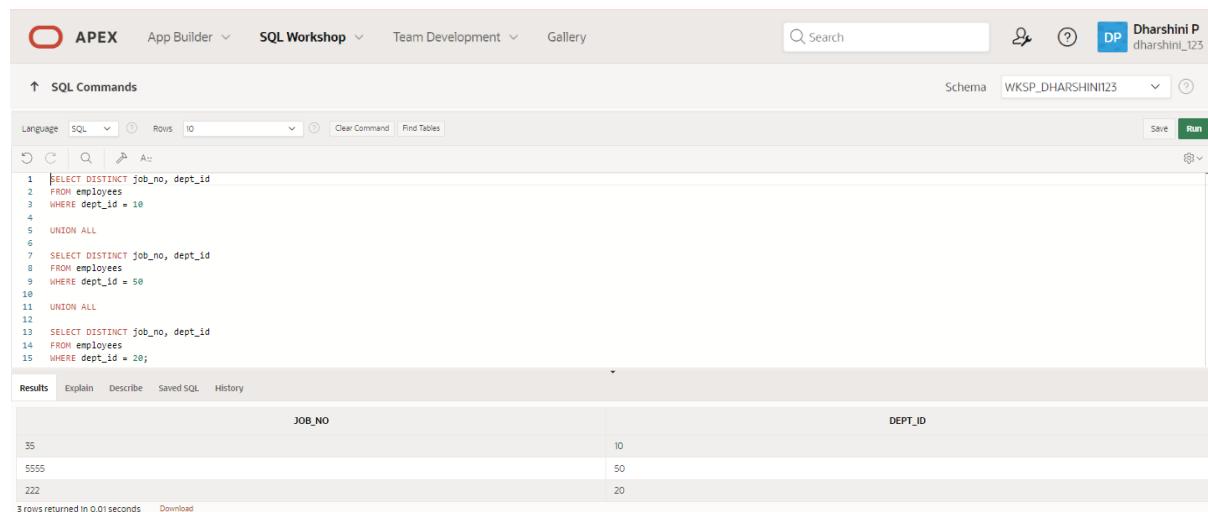
UNION ALL

```
SELECT DISTINCT job_no, dept_id  
FROM employees  
WHERE dept_id = 50
```

UNION ALL

```
SELECT DISTINCT job_no, dept_id  
FROM employees  
WHERE dept_id = 20;
```

#### OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. A search bar and user profile 'Dharshini P dharshini123' are also present. The main area has tabs for 'SQL Commands' and 'Results'. The SQL Commands tab displays the following code:

```
1 SELECT DISTINCT job_no, dept_id  
2 FROM employees  
3 WHERE dept_id = 10  
4  
5 UNION ALL  
6  
7 SELECT DISTINCT job_no, dept_id  
8 FROM employees  
9 WHERE dept_id = 50  
10  
11 UNION ALL  
12  
13 SELECT DISTINCT job_no, dept_id  
14 FROM employees  
15 WHERE dept_id = 20;
```

The Results tab shows the output in a table:

JOB_NO	DEPT_ID
35	10
5555	50
222	20

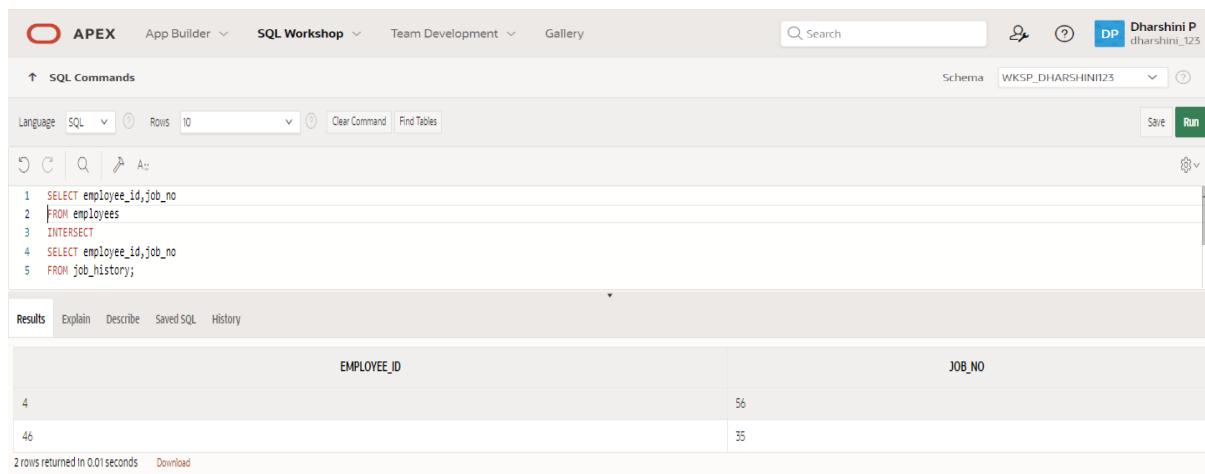
At the bottom left, it says '3 rows returned in 0.01 seconds'.

4. Create a report that lists the employee IDs and job IDs of those employees who currently have a job title that is the same as their job title when they were initially hired by the company (that is, they changed jobs but have now gone back to doing their original job).

### QUERY:

```
SELECT employee_id,job_no
FROM employees
INTERSECT
SELECT employee_id,job_no
FROM job_history;
```

### OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows a user profile for Dharshini P (dharshini\_123). The main area has tabs for SQL Commands and Results. In the SQL Commands tab, the following query is entered:

```
1 SELECT employee_id,job_no
2 FROM employees
3 INTERSECT
4 SELECT employee_id,job_no
5 FROM job_history;
```

The Results tab displays the output as a table:

EMPLOYEE_ID	JOB_NO
4	56
46	35

Below the table, it says "2 rows returned in 0.01 seconds".

5. The HR department needs a report with the following specifications:

- Last name and department ID of all the employees from the EMPLOYEES table, regardless of whether or not they belong to a department.
- Department ID and department name of all the departments from the DEPARTMENTS table, regardless of whether or not they have employees working in them Write a compound query to accomplish this.

#### QUERY:

```
SELECT last_name,dept_id,TO_CHAR(null)
FROM employees
UNION
SELECT TO_CHAR(null),dept_id,dept_name
FROM department;
```

#### OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The SQL command entered is:

```
1 SELECT last_name,dept_id,TO_CHAR(null)
2 FROM employees
3 UNION
4 SELECT TO_CHAR(null),dept_id,dept_name
5 FROM department;
```

The results section displays the following data:

LAST_NAME	DEPT_ID	TO_CHAR(NULL)
Davies	20	-
King	10	-
Zlotkey	2222	-
bhuvana	1	-
brindha	59	-
deepas	50	-
-	1	Executive
-	59	BIO
-	2222	cse

9 rows returned in 0.01 seconds

Evaluation Procedure	Marks awarded
Query(5)	
Execution (5)	
Viva(5)	
Total (15)	
Faculty Signature	

#### RESULT:

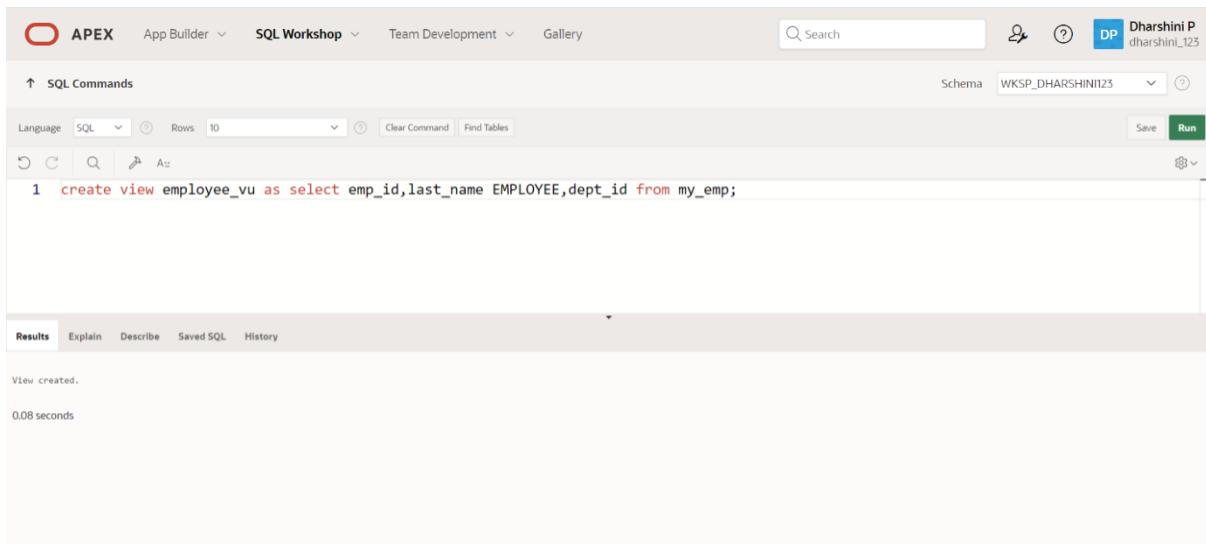
**EXNO:11**

## **CREATING VIEWS**

**DATE:**

**1.Create a view called EMPLOYEE\_VU based on the employee numbers, employee names and department numbers from the EMPLOYEES table. Change the heading for the employee name to EMPLOYEE.**

**QUERY: create view employee\_vu as select emp\_id,last\_name EMPLOYEE,dept\_id from my\_emp;**  
**OUTPUT:**

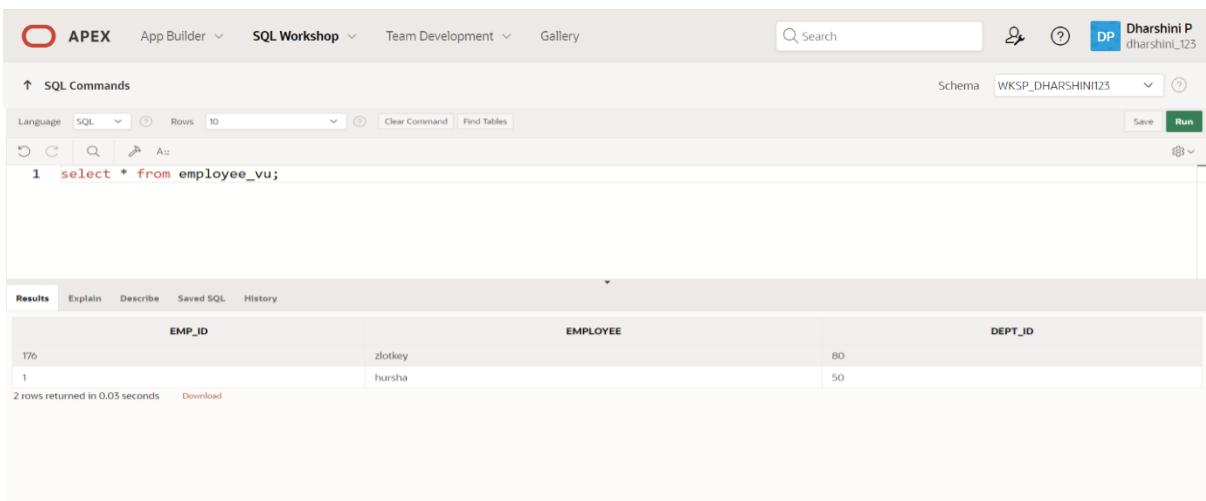


The screenshot shows the Oracle SQL Workshop interface. In the top navigation bar, 'APEX' is selected. The main area is titled 'SQL Commands'. A single line of SQL code is entered: 'create view employee\_vu as select emp\_id,last\_name EMPLOYEE,dept\_id from my\_emp;'. Below the code, the results pane shows the message 'View created.' and a execution time of '0.08 seconds'.

**2.Display the contents of the EMPLOYEES\_VU view.**

**QUERY: select \* from employee\_vu;**

**OUTPUT:**



The screenshot shows the Oracle SQL Workshop interface. In the top navigation bar, 'APEX' is selected. The main area is titled 'SQL Commands'. A single line of SQL code is entered: 'select \* from employee\_vu;'. Below the code, the results pane displays the following table:

EMP_ID	EMPLOYEE	DEPT_ID
176	zlotkey	80
1	hursha	50

At the bottom of the results pane, it says '2 rows returned in 0.03 seconds'.

**3.Using your EMPLOYEES\_VU view, enter a query to display all employees names and department.**

**QUERY:** select employee, dept\_id from employee\_vu;

**OUTPUT:**

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, and Gallery. The schema dropdown is set to WKSP\_DHARSHINI123. The main area shows a SQL command window with the following content:

```
1 select employee, dept_id from employee_vu;
```

Below the command window, the results tab is selected, displaying the output:

EMPLOYEE	DEPT_ID
zlotkey	80
durma	50

2 rows returned in 0.01 seconds

**4.Create a view named DEPT50 that contains the employee number, employee last names and department numbers for all employees in department 50.Label the view columns EMPNO, EMPLOYEE and DEPTNO. Do not allow an employee to be reassigned to another department through the view.**

**QUERY:** create view DEPT50 as select emp\_id EMPNO,last\_name EMPLOYEE,dept\_id DEPTNO from my\_emp where dept\_id=50;

**OUTPUT:**

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, and Gallery. The schema dropdown is set to WKSP\_DHARSHINI123. The main area shows a SQL command window with the following content:

```
1 create view DEPT50 as select emp_id EMPNO,last_name EMPLOYEE,dept_id DEPTNO from my_emp where dept_id=50;
```

Below the command window, the results tab is selected, displaying the output:

View created.

0.05 seconds

## 5. Display the structure and contents of the DEPT50 view.

QUERY: select \* from dept50;

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. In the top navigation bar, the 'SQL Workshop' tab is selected. The schema dropdown shows 'WKSP\_DHARSHINI123'. The main area displays the results of the query 'select \* from DEPT50;'. The results table has three columns: EMPNO, EMPLOYEE, and DEPTNO. One row is returned, with values 1, hursha, and 50 respectively. A message at the bottom indicates '1 rows returned in 0.00 seconds'.

EMPNO	EMPLOYEE	DEPTNO
1	hursha	50

## 6. Attempt to reassign Matos to department 80.

QUERY: update dept50 set deptno=80 where employee='Matos';

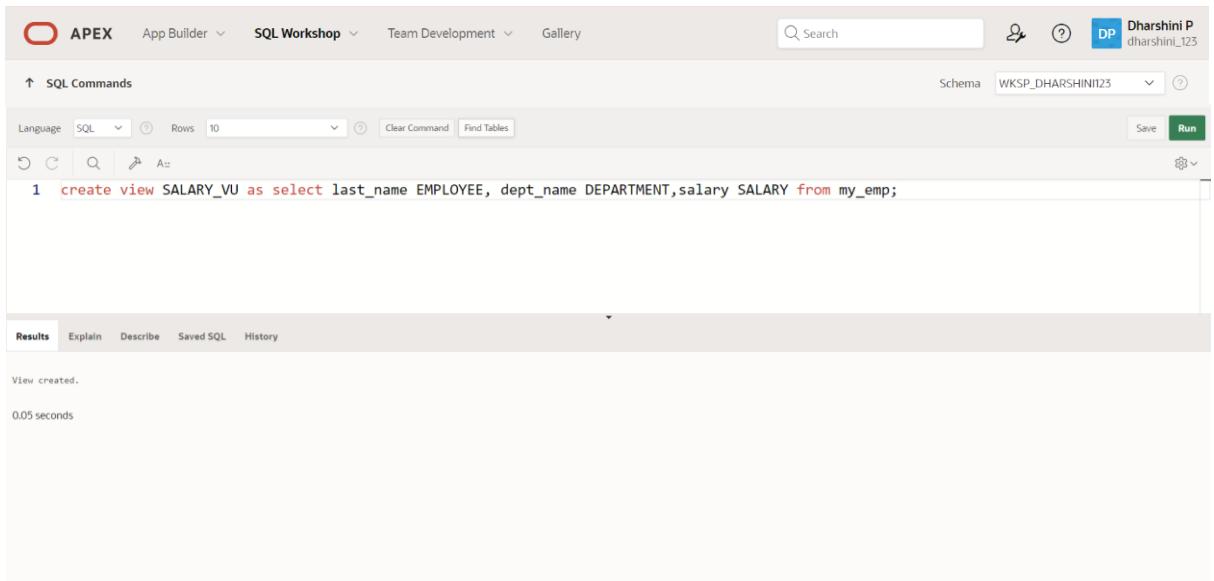
OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. In the top navigation bar, the 'SQL Workshop' tab is selected. The schema dropdown shows 'WKSP\_DHARSHINI123'. The main area displays the results of the query 'update dept50 set deptno=80 where employee='Matos';'. The message at the bottom indicates '0 row(s) updated.' and '0.04 seconds'.

**7.Create a view called SALARY\_VU based on the employee last names, department names, salaries, and salary grades for all employees. Use the Employees, DEPARTMENTS and JOB\_GRADE tables. Label the column Employee, Department, salary, and Grade respectively.**

**QUERY:** create view salary\_vu as select lname EMPLOYEE, dept\_name DEPARTMENT, salary SALARY, sal\_grade GRADE from my\_emp;

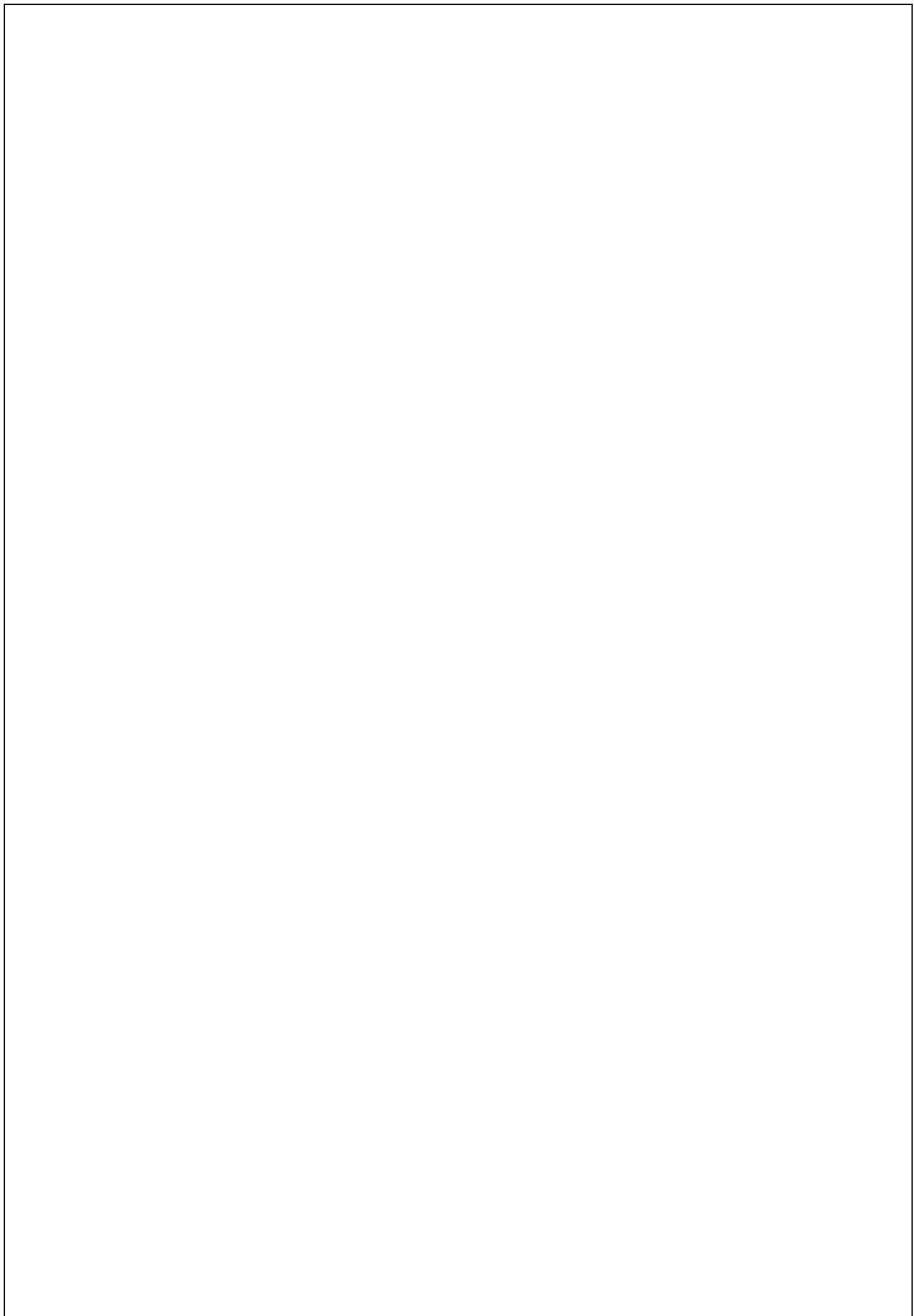
**OUTPUT:**



The screenshot shows the Oracle SQL Workshop interface. In the top navigation bar, 'APEX' is selected. The main area is titled 'SQL Commands'. A single line of SQL code is entered: 'create view SALARY\_VU as select last\_name EMPLOYEE, dept\_name DEPARTMENT, salary SALARY, sal\_grade GRADE from my\_emp;'. Below the code, the 'Results' tab is active, displaying the message 'View created.' and '0.05 seconds' execution time.

Evaluation Procedure	Marks awarded
Query(5)	
Execution (5)	
Viva(5)	
Total (15)	
Faculty Signature	

**RESULT:**



# EXERCISE 12

## PRACTICE QUESTIONS

### Intro to Constraints; NOT NULL and UNIQUE Constraints

Global Fast Foods has been very successful this past year and has opened several new stores. They need to add a table to their database to store information about each of their store's locations. The owners want to make sure that all entries have an identification number, date opened, address, and city and that no other entry in the table can have the same email address. Based on this information, answer the following questions about the global\_locations table. Use the table for your answers.

Global Fast Foods global_locations Table						
NAME	TYPE	LENGTH	PRECISION	SCALE	NULLABLE	DEFAULT
Id						
name						
date_opened						
address						
city						
zip/postal code						
phone						
email						
manager_id						
Emergency contact						

1. What is a “constraint” as it relates to data integrity?

Database can be as reliable as the data in it, and database rules are implemented as Constraint to maintain data integrity.

2. What are the limitations of constraints that may be applied at the column level and at the table level?

- Constraints referring to more than one column are defined at Table Level
- NOT NULL constraint must be defined at column level as per ANSI/ISO SQL standard.

3. Why is it important to give meaningful names to constraints?

- If a constraint is violated in a SQL statement execution, it is easy to identify the cause with user-named constraints.
- It is easy to alter names/drop constraint.

4. Based on the information provided by the owners, choose a datatype for each column. Indicate the length, precision, and scale for each NUMBER datatype.

Global Fast Foods global_locations Table						
NAME	TYPE	DataType	LENGTH	PRECISION	SCALE	NULLABLE
id	pk	NUMBER	6	0		No
name		VARCHAR2	50			
date_opened		DATE				No
address		VARCHAR2	50			No
city		VARCHAR2	30			No
zip_postal_code		VARCHAR2	12			
phone		VARCHAR2	20			
email	uk	VARCHAR2	75			
manager_id		NUMBER	6	0		
emergency_contact		VARCHAR2	20			

5. Use "(nullable)" to indicate those columns that can have null values.

Global Fast Foods global_locations Table						
NAME	TYPE	DataType	LENGTH	PRECISION	SCALE	NULLABLE
id	pk	NUMBER	6	0		No
name		VARCHAR2	50			Yes
date_opened		DATE				No
address		VARCHAR2	50			No
city		VARCHAR2	30			No
zip_postal_code		VARCHAR2	12			Yes
phone		VARCHAR2	20			Yes
email	uk	VARCHAR2	75			Yes
manager_id		NUMBER	6	0		Yes
emergency_contact		VARCHAR2	20			Yes

6. Write the CREATE TABLE statement for the Global Fast Foods locations table to define the constraints at the column level.

```
CREATE TABLE f_global_locations
( id NUMBER(6,0) CONSTRAINT f_gln_id_pk PRIMARY KEY ,
  name VARCHAR2(50),
  date_opened DATE CONSTRAINT f_gln_dt_opened_nn NOT NULL ENABLE,
  address VARCHAR2(50) CONSTRAINT f_gln_add_nn NOT NULL ENABLE,
  city VARCHAR2(30) CONSTRAINT f_gln_city_nn NOT NULL ENABLE,
  zip_postal_code VARCHAR2(12),
  phone VARCHAR2(20),
  email VARCHAR2(75) CONSTRAINT f_gln_email_uk UNIQUE,
  manager_id NUMBER(6,0),
  emergency_contact VARCHAR2(20)
);
```

7. Execute the CREATE TABLE statement in Oracle Application Express.

Table Created.

8. Execute a DESCRIBE command to view the Table Summary information.

DESCRIBE f\_global\_locations;

9. Rewrite the CREATE TABLE statement for the Global Fast Foods locations table to define the UNIQUE constraints at the table level. Do not execute this statement.

NAME	TYPE	LENGTH	PRECISION	SCALE	NULLABLE	DEFAULT
id	number	4				
loc_name	varchar2	20			X	
	date					
address	varchar2	30				
city	varchar2	20				
zip_postal	varchar2	20			X	
phone	varchar2	15			X	
email	varchar2	80			X	
manager_id	number	4			X	
contact	varchar2	40			X	

```
CREATE TABLE f_global_locations
( id NUMBER(6,0) CONSTRAINT f_gln_id_pk PRIMARY KEY ,
name VARCHAR2(50),
date_opened DATE CONSTRAINT f_gln_dt_opened_nn NOT NULL ENABLE,
address VARCHAR2(50) CONSTRAINT f_gln_add_nn NOT NULL ENABLE,
city VARCHAR2(30) CONSTRAINT f_gln_city_nn NOT NULL ENABLE,
zip_postal_code VARCHAR2(12),
phone VARCHAR2(20),
email VARCHAR2(75) ,
manager_id NUMBER(6,0),
emergency_contact VARCHAR2(20),
CONSTRAINT f_gln_email_uk UNIQUE(email)
);
```

# PRIMARY KEY, FOREIGN KEY, and CHECK Constraints

1. What is the purpose of a

- PRIMARY KEY
- FOREIGN KEY
- CHECK CONSTRAINT

**a. PRIMARY KEY**

Uniquely identify each row in table.

**b. FOREIGN KEY**

Referential integrity constraint links back parent table's primary/unique key to child table's column.

**c. CHECK CONSTRAINT**

Explicitly define condition to be met by each row's fields. This condition must be returned as true or unknown.

2. Using the column information for the animals table below, name constraints where applicable at the table level, otherwise name them at the column level. Define the primary key (animal\_id). The license\_tag\_number must be unique. The admit\_date and vaccination\_date columns cannot contain null values.

animal_id NUMBER(6)	- PRIMARY KEY
name VARCHAR2(25)	
license_tag_number NUMBER(10)	- UNIQUE
admit_date DATE	-NOT NULL
adoption_id NUMBER(5),	
vaccination_date DATE	-NOT NULL

3. Create the animals table. Write the syntax you will use to create the table.

```
CREATE TABLE animals
( animal_id NUMBER(6,0) CONSTRAINT anl_anl_id_pk PRIMARY KEY ,
  name VARCHAR2(25),
  license_tag_number NUMBER(10,0) CONSTRAINT anl_l_tag_num_uk UNIQUE,
  admit_date DATE CONSTRAINT anl_adt_dat_nn NOT NULL ENABLE,
  adoption_id NUMBER(5,0),
  vaccination_date DATE CONSTRAINT anl_vcc_dat_nn NOT NULL ENABLE
);
```

4. Enter one row into the table. Execute a SELECT \* statement to verify your input. Refer to the graphic below for input.

ANIMAL_ID	NAME	LICENSE_TAG_NUMBE R	ADMIT_DATE	ADOPTION_ID	VACCINATION_DATE
101	Spot	35540	10-Oct-2004	205	12-Oct-2004

```
INSERT INTO animals (animal_id, name, license_tag_number, admit_date, adoption_id, vaccination_date)
VALUES( 101,'Spot', 35540, TO_DATE('10-Oct-2004', 'DD-Mon-YYYY'), 205, TO_DATE('12-Oct-2004', 'DD-Mon-YYYY'));
```

```
SELECT * FROM animals;
```

5. Write the syntax to create a foreign key (adoption\_id) in the animals table that has a corresponding primary-key reference in the adoptions table. Show both the column-level and table-level syntax. Note that because you have not actually created an adoptions table, no adoption\_id primary key exists, so the foreign key cannot be added to the animals table.

**COLUMN LEVEL STATEMENT:**

```
ALTER TABLE animals
```

```
MODIFY ( adoption_id NUMBER(5,0) CONSTRAINT anl_adopt_id_fk REFERENCES adoptions(id)
ENABLE );
```

**TABLE LEVEL STATEMENT:**

```
ALTER TABLE animals ADD CONSTRAINT anl_adopt_id_fk FOREIGN KEY (adoption_id)
REFERENCES adoptions(id) ENABLE;
```

6. What is the effect of setting the foreign key in the ANIMAL table as:

a. ON DELETE CASCADE

```
ALTER TABLE animals
ADD CONSTRAINT anl_adopt_id_fk FOREIGN KEY (adoption_id)
REFERENCES adoptions(id) ON DELETE CASCADE ENABLE ;
```

b. ON DELETE SET NULL

```
ALTER TABLE animals
ADD CONSTRAINT anl_adopt_id_fk FOREIGN KEY (adoption_id)
REFERENCES adoptions(id) ON DELETE SET NULL ENABLE ;
```

7. What are the restrictions on defining a CHECK constraint?

- I cannot specify check constraint for a view however in this case I could use WITH CHECK OPTION clause
- I am restricted to columns from self table and fields in self row.
- I cannot use subqueries and scalar subquery expressions.

Evaluation Procedure	Marks awarded
Query(5)	
Execution (5)	
Viva(5)	
Total (15)	
Faculty Signature	

# PRACTICE PROBLEM

## Managing Constraints

Using Oracle Application Express, click the SQL Workshop tab in the menu bar. Click the Object Browser and verify that you have a table named copy\_d\_clients and a table named copy\_d\_events. If you don't have these tables in your schema, create them before completing the exercises below. Here is how the original tables are related. The d\_clients table has a primary key client\_number. This has a primary-key constraint and it is referenced in the foreign-key constraint on the d\_events table.

**NOTE:** The practice exercises use the d\_clients and d\_events tables in the DJs on Demand database. Students will work with copies of these two tables named copy\_d\_clients and copy\_d\_events. Make sure they have new copies of the tables (without changes made from previous exercises). Remember, tables copied using a subquery do not have the integrity constraints as established in the original tables. When using the SELECT statement to view the constraint name, the tablenames must be all capital letters.

1. What are four functions that an ALTER statement can perform on constraints?

- ADD
- DROP
- ENABLE
- DISABLE

2. Since the tables are copies of the original tables, the integrity rules are not passed onto the new tables; only the column datatype definitions remain. You will need to add a PRIMARY KEY constraint to the copy\_d\_clients table. Name the primary key copy\_d\_clients\_pk . What is the syntax you used to create the PRIMARY KEY constraint to the copy\_d\_clients.table?

```
ALTER TABLE copy_d_clients  
ADD CONSTRAINT copy_d_clt_client_number_pk PRIMARY KEY (client_number);
```

3. Create a FOREIGN KEY constraint in the copy\_d\_events table. Name the foreign key copy\_d\_events\_fk. This key references the copy\_d\_clients table client\_number column. What is the syntax you used to create the FOREIGN KEY constraint in the copy\_d\_events table?

```
ALTER TABLE copy_d_events  
ADD CONSTRAINT copy_d_eve_client_number_fk FOREIGN KEY (client_number) REFERENCES  
copy_d_clients (client_number) ENABLE;
```

4. Use a SELECT statement to verify the constraint names for each of the tables. Note that the tablenames must be capitalized.

```
SELECT constraint_name, constraint_type, table_name  
FROM user_constraints  
WHERE table_name = UPPER('copy_d_events');
```

a. The constraint name for the primary key in the copy\_d\_clients table is\_\_\_\_\_.

**COPY\_D\_CLT\_CLIENT\_NUMBER\_PK**

5. Drop the PRIMARY KEY constraint on the copy\_d\_clients table. Explain your results.

```
ALTER TABLE copy_d_clients  
DROP CONSTRAINT COPY_D_CLT_CLIENT_NUMBER_PK CASCADE ;
```

6. Add the following event to the copy\_d\_events table. Explain your results.

ID	NAME	EVENT_DATE	DESCRIPTION	COST	VENUE_ID	PACKAGE_CODE	THEME_CODE	CLIENT_NUMBER
140	Cline Bas Mitzvah	15-Jul-2004	Church and Private Home formal	4500	105	87	77	7125

```
INSERT INTO
copy_d_events(client_number,id,name,event_date,description,cost,venue_id,package_code,theme_code)
VALUES(7125,140,'Cline Bas Mitzvah',TO_DATE('15-Jul-2004','dd-Mon-yyyy'),'Church and Private Home formal',4500,105,87,77);
```

**RESULT:** ORA-02291: integrity constraint (HKUMAR.COPY\_D\_EVE\_CLIENT\_NUMBER\_FK) violated - parent key not found

7. Create an ALTER TABLE query to disable the primary key in the copy\_d\_clients table. Then add the values from #6 to the copy\_d\_events table. Explain your results.

```
ALTER TABLE copy_d_clients
DISABLE CONSTRAINT COPY_D_CLT_CLIENT_NUMBER_PK CASCADE;
```

8. Repeat question 6: Insert the new values in the copy\_d\_events table. Explain your results.

```
INSERT INTO
copy_d_events(client_number,id,name,event_date,description,cost,venue_id,package_code,theme_code)
VALUES(7125,140,'Cline Bas Mitzvah',TO_DATE('15-Jul-2004','dd-Mon-yyyy'),'Church and Private Home formal',4500,105,87,77);
```

**1 row(s) inserted.**

9. Enable the primary-key constraint in the copy\_d\_clients table. Explain your results.

```
ALTER TABLE copy_d_clients
ENABLE CONSTRAINT COPY_D_CLT_CLIENT_NUMBER_PK ;
```

10. If you wanted to enable the foreign-key column and reestablish the referential integrity between these two tables, what must be done?

```
DELETE FROM copy_d_events WHERE
client_number NOT IN ( SELECT client_number FROM copy_d_clients);
```

**1 row(s) deleted.**

```
ALTER TABLE copy_d_events
ENABLE CONSTRAINT COPY_D_EVE_CLIENT_NUMBER_FK;
```

**Table altered.**

11. Why might you want to disable and then re-enable a constraint?

Generally to make bulk operations fast, where my input data is diligently sanitized and I am sure, it is safe to save some time in this clumsy process.

12. Query the data dictionary for some of the constraints that you have created. How does the data dictionary identify each constraint type?

Queries are same as in point 2,3, 4 above.

- C - Check constraint
  - Sub-case - if I see SEARCH\_CONDITION something like "FIRST\_NAME" IS NOT NULL , its a NOT NULL constraint.
- P - Primary key
- R - Referential integrity (fk)
- U - Unique key

Evaluation Procedure	Marks awarded
Query(5)	
Execution (5)	
Viva(5)	
Total (15)	
Faculty Signature	

## CREATING VIEWS

EX.NO.13

DATE:

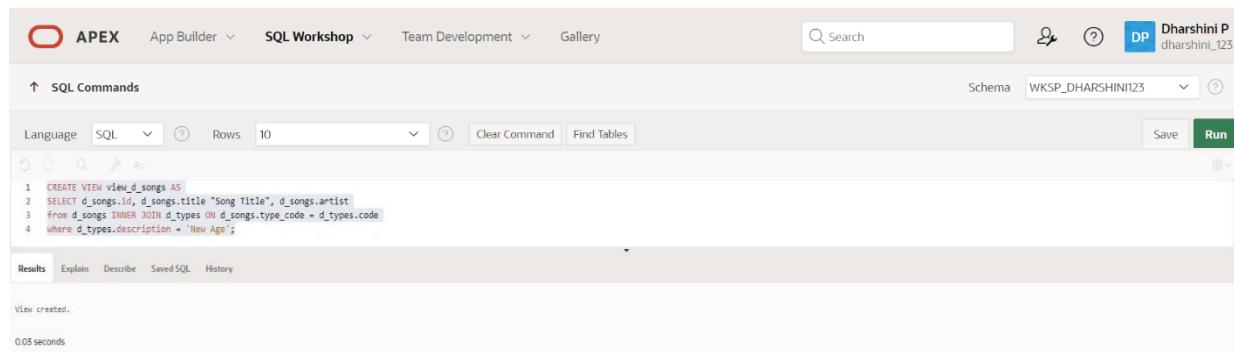
1. What are three uses for a view from a DBA's perspective?

- **Restrict access and display selective columns**
- **Reduce complexity of queries from other internal systems. So, providing a way to view same data in a different manner.**
- **Let the app code rely on views and allow the internal implementation of tables to be modified later.**

2. Create a simple view called view\_d\_songs that contains the ID, title and artist from the DJs on Demand table for each "New Age" type code. In the subquery, use the alias "Song Title" for the title column.

**CREATE VIEW view\_d\_songs AS**

```
SELECT d_songs.id, d_songs.title "Song Title", d_songs.artist
from d_songs INNER JOIN d_types ON d_songs.type_code = d_types.code
where d_types.description = 'New Age';
```

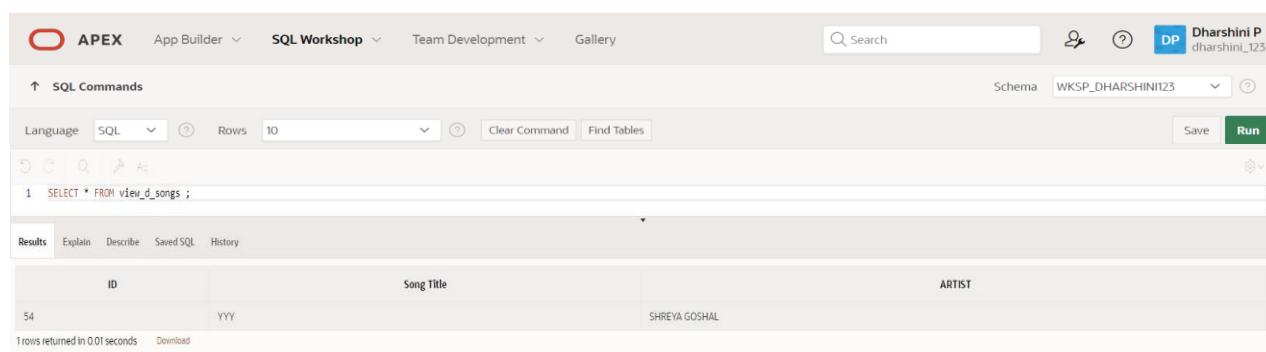


The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, and Gallery. The SQL Workshop tab is selected. The schema dropdown is set to WKSP\_DHARSHINI123. The main area displays the SQL command for creating the view:

```
1 CREATE VIEW view_d_songs AS
2 SELECT d_songs.id, d_songs.title "Song Title", d_songs.artist
3 FROM d_songs INNER JOIN d_types ON d_songs.type_code = d_types.code
4 WHERE d_types.description = 'New Age';
```

Below the command, the message "View created." is displayed, along with a timestamp of "0.03 seconds".

3. **SELECT \* FROM view\_d\_songs.** What was returned?



The screenshot shows the Oracle SQL Workshop interface with the same schema and workspace settings as the previous screenshot. The SQL command is:

```
1 SELECT * FROM view_d_songs;
```

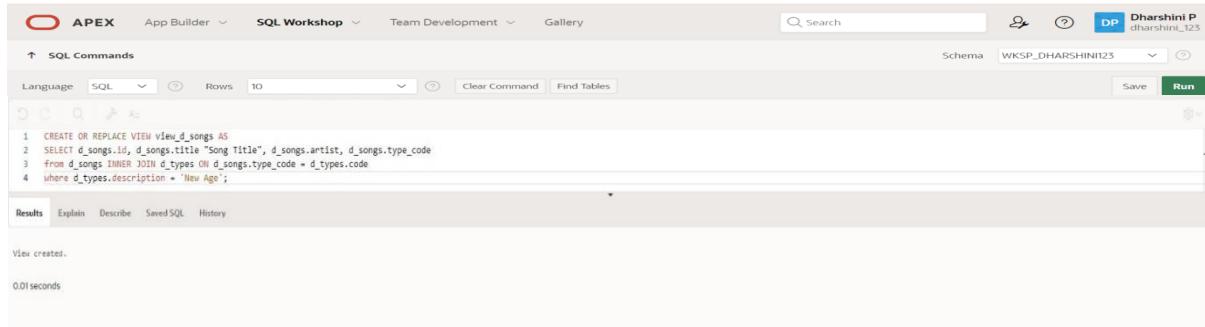
The results pane shows the output of the query:

ID	Song Title	ARTIST
54	YYY	SHREYA GOSHAL

At the bottom, it says "1 rows returned in 0.01 seconds".

4. REPLACE view\_d\_songs. Add type\_code to the column list. Use aliases for all columns. Or use alias after the CREATE statement as shown.

```
CREATE OR REPLACE VIEW view_d_songs AS
SELECT d_songs.id, d_songs.title "Song Title", d_songs.artist, d_songs.type_code
from d_songs INNER JOIN d_types ON d_songs.type_code = d_types.code
where d_types.description = 'New Age';
```



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows a user profile for Dharshini P. The main area is titled 'SQL Commands' with a 'Language' dropdown set to 'SQL'. The code editor contains the SQL query for creating the view. The results pane below shows the message 'View created.' and a execution time of '0.01 seconds'.

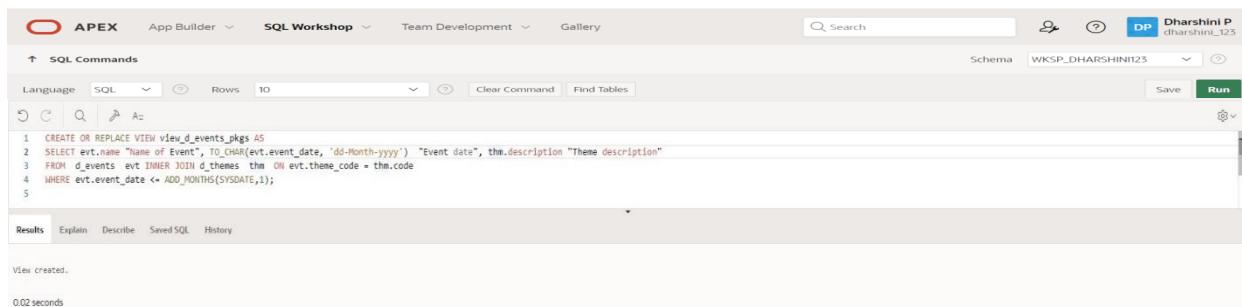
```
1 CREATE OR REPLACE VIEW view_d_songs AS
2 SELECT d_songs.id, d_songs.title "Song Title", d_songs.artist, d_songs.type_code
3 from d_songs INNER JOIN d_types ON d_songs.type_code = d_types.code
4 where d_types.description = 'New Age';
```

Results Explain Describe Saved SQL History

View created.  
0.01 seconds

5. Jason Tsang, the disk jockey for DJs on Demand, needs a list of the past events and those planned for the coming months so he can make arrangements for each event's equipment setup. As the company manager, you do not want him to have access to the price that clients paid for their events. Create a view for Jason to use that displays the name of the event, the event date, and the theme description. Use aliases for each column name.

```
CREATE OR REPLACE VIEW view_d_events_pkgs AS
SELECT evt.name "Name of Event", TO_CHAR(evt.event_date, 'dd-Month-
yyyy') "Event date", thm.description "Theme description"
FROM d_events evt INNER JOIN d_themes thm ON evt.theme_code = thm.code
WHERE evt.event_date <= ADD_MONTHS(SYSDATE,1);
```



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows a user profile for Dharshini P. The main area is titled 'SQL Commands' with a 'Language' dropdown set to 'SQL'. The code editor contains the SQL query for creating the view. The results pane below shows the message 'View created.' and a execution time of '0.02 seconds'.

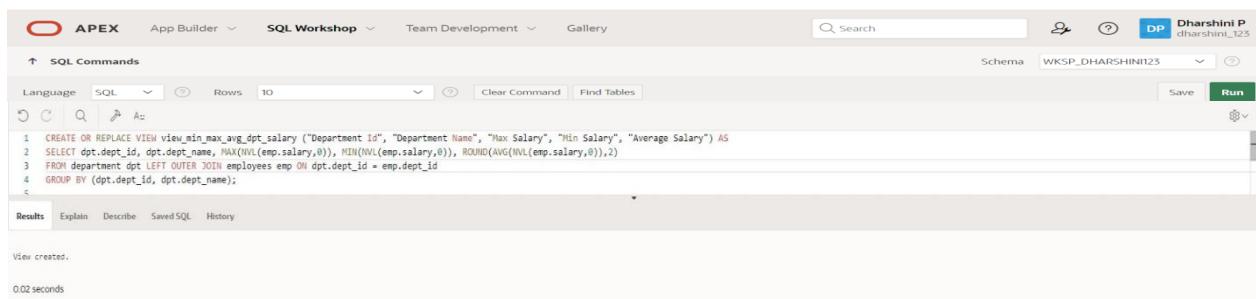
```
1 CREATE OR REPLACE VIEW view_d_events_pkgs AS
2 SELECT evt.name "Name of Event", TO_CHAR(evt.event_date, 'dd-Month-yyyy') "Event date", thm.description "Theme description"
3 FROM d_events evt INNER JOIN d_themes thm ON evt.theme_code = thm.code
4 WHERE evt.event_date <= ADD_MONTHS(SYSDATE,1);
5
```

Results Explain Describe Saved SQL History

View created.  
0.02 seconds

6. It is company policy that only upper-level management be allowed access to individual employee salaries. The department managers, however, need to know the minimum, maximum, and average salaries, grouped by department. Use the Oracle database to prepare a view that displays the needed information for department managers.

```
CREATE OR REPLACE VIEW view_min_max_avg_dpt_salary ("Department Id",  
"Department Name", "Max Salary", "Min Salary", "Average Salary") AS  
SELECT dpt.department_id, dpt.department_name, MAX(NVL(emp.salary,0)),  
MIN(NVL(emp.salary,0)), ROUND(AVG(NVL(emp.salary,0)),2)  
FROM departments dpt LEFT OUTER JOIN employees emp ON dpt.department_id =  
emp.department_id  
GROUP BY (dpt.department_id, dpt.department_name);
```

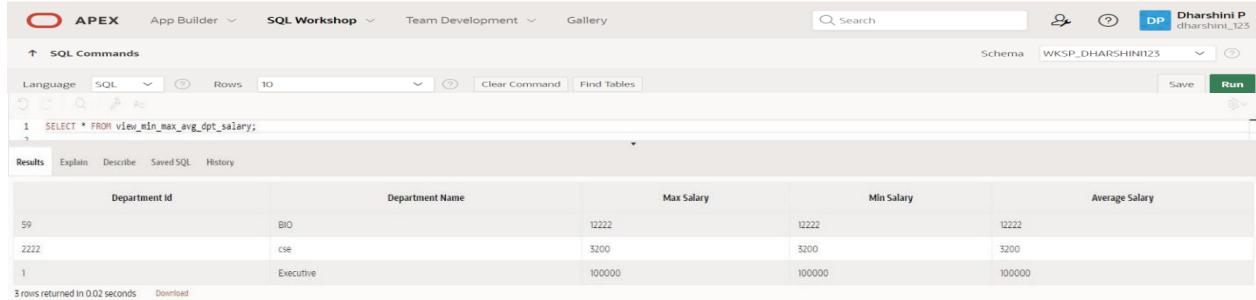


The screenshot shows the Oracle SQL Workshop interface. In the top navigation bar, 'SQL Workshop' is selected. The main area contains the SQL command for creating the view:

```
1 CREATE OR REPLACE VIEW view_min_max_avg_dpt_salary ("Department Id",  
2 "Department Name", "Max Salary", "Min Salary", "Average Salary") AS  
3 SELECT dpt.department_id, dpt.department_name, MAX(NVL(emp.salary,0)),  
4 MIN(NVL(emp.salary,0)), ROUND(AVG(NVL(emp.salary,0)),2)  
5 FROM departments dpt LEFT OUTER JOIN employees emp ON dpt.department_id =  
6 emp.department_id  
7 GROUP BY (dpt.department_id, dpt.department_name);
```

Below the command, the results show:

View created.  
0.02 seconds



The screenshot shows the Oracle SQL Workshop interface. In the top navigation bar, 'SQL Workshop' is selected. The main area contains the SQL command for selecting from the view:

```
1 SELECT * FROM view_min_max_avg_dpt_salary;
```

Below the command, the results show a table with the following data:

Department Id	Department Name	Max Salary	Min Salary	Average Salary
59	BIO	12222	12222	12222
2222	cse	3200	3200	3200
1	Executive	100000	100000	100000

3 rows returned in 0.02 seconds

## DML Operations and Views

Use the DESCRIBE statement to verify that you have tables named copy\_d\_songs, copy\_d\_events, copy\_d\_cds, and copy\_d\_clients in your schema. If you don't, write a query to create a copy of each.

1. Query the data dictionary USER\_UPDATABLE\_COLUMNS to make sure the columns in the base tables will allow UPDATE, INSERT, or DELETE. All table names in the data dictionary are stored in uppercase.

```
SELECT owner, table_name, column_name, updatable,insertable, deletable
FROM user_updatable_columns WHERE LOWER(table_name) = 'copy_d_songs';
```

```
SELECT owner, table_name, column_name, updatable,insertable, deletable
FROM user_updatable_columns WHERE LOWER(table_name) = 'copy_d_events';
```

```
SELECT owner, table_name, column_name, updatable,insertable, deletable
FROM user_updatable_columns WHERE LOWER(table_name) = 'copy_d_cds';
```

The screenshot shows the Oracle SQL Workshop interface with the 'Results' tab selected. The query executed was:

```
1 SELECT owner, table_name, column_name, updatable,insertable, deletable
2 FROM user_updatable_columns WHERE LOWER(table_name) = 'copy_d_songs';
3
```

The results table has the following structure and data:

OWNER	TABLE_NAME	COLUMN_NAME	UPDATABLE	INSERTABLE	DELETABLE
WKSP_MBHVU	COPY_D_SONGS	TITLE	YES	YES	YES
WKSP_MBHVU	COPY_D_SONGS	DURATION	YES	YES	YES
WKSP_MBHVU	COPY_D_SONGS	TYPE_CODE	YES	YES	YES

3 rows returned in 0.04 seconds [Download](#)

The screenshot shows the Oracle SQL Workshop interface with the 'Results' tab selected. The query executed was:

```
1 SELECT owner, table_name, column_name, updatable,insertable, deletable
2 FROM user_updatable_columns WHERE LOWER(table_name) = 'copy_d_cds';
3
```

The results table has the following structure and data:

OWNER	TABLE_NAME	COLUMN_NAME	UPDATABLE	INSERTABLE	DELETABLE
WKSP_MBHVU	COPY_D_CDS	CD_NUMBER	YES	YES	YES
WKSP_MBHVU	COPY_D_CDS	PRODUCER	YES	YES	YES
WKSP_MBHVU	COPY_D_CDS	TITLE	YES	YES	YES
WKSP_MBHVU	COPY_D_CDS	YEAR	YES	YES	YES

4 rows returned in 0.05 seconds [Download](#)

2. Use the CREATE or REPLACE option to create a view of *all* the columns in the copy\_d\_songs table called view\_copy\_d\_songs.

```
CREATE OR REPLACE VIEW view_copy_d_songs AS
SELECT *
FROM copy_d_songs;
```

**SELECT \* FROM view\_copy\_d\_songs**

The screenshot shows the Oracle SQL Workshop interface. In the top navigation bar, 'APEX' is selected. Below it, the 'SQL Workshop' tab is active. The schema dropdown shows 'WKSP\_DHARSHINI123'. The main area contains two code snippets. The first snippet creates a view named 'view\_copy\_d\_songs' with three columns: 'CREATE OR REPLACE VIEW view\_copy\_d\_songs AS SELECT \* FROM copy\_d\_songs;'. The second snippet selects all columns from this view: 'SELECT \* FROM view\_copy\_d\_songs;'. Both snippets have a timestamp of '0.02 seconds'. Below the code, the 'Results' tab is selected, showing the message 'no data found'.

3. Use view\_copy\_d\_songs to INSERT the following data into the underlying copy\_d\_songs table. Execute a SELECT \* from copy\_d\_songs to verify your DML command. See the graphic.

ID	TITLE	DURATION	ARTIST	TYPE_CODE
88	Mello Jello	2	The What	4

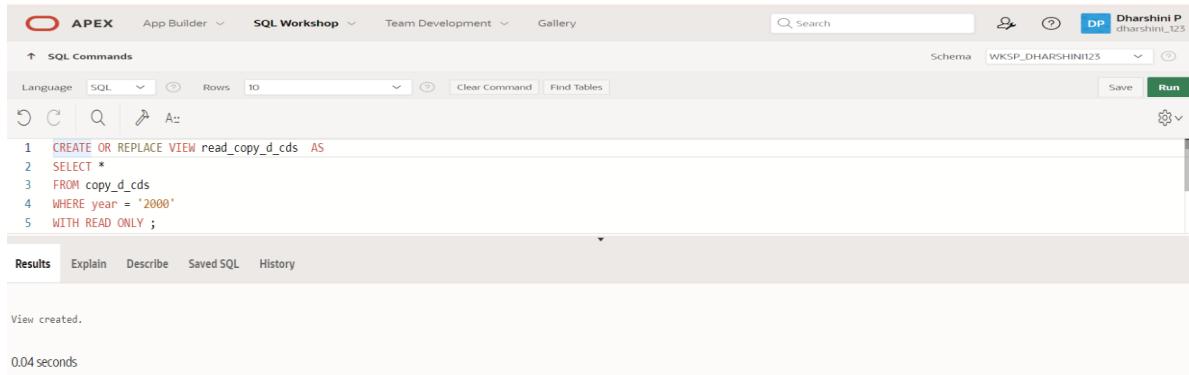
```
INSERT INTO view_copy_d_songs(id,title,duration,artist,type_code)
VALUES(88,'Mello Jello','2 min','The What',4);
```

The screenshot shows the Oracle SQL Workshop interface. The 'SQL Workshop' tab is active. The schema dropdown shows 'WKSP\_DHARSHINI123'. The main area contains the following code: '1 INSERT INTO view\_copy\_d\_songs(id,title,duration,artist,type\_code) 2 VALUES(88,'Mello Jello','2 min','The What',4);'. The timestamp is '0.03 seconds'. Below the code, the 'Results' tab is selected, showing the message '1 row(s) inserted.'

4. Create a view based on the DJs on Demand COPY\_D\_CDS table. Name the view read\_copy\_d\_cds. Select all columns to be included in the view. Add a WHERE clause to restrict the year to 2000. Add the WITH READ ONLY option.

```
CREATE OR REPLACE VIEW read_copy_d_cds AS
SELECT *
FROM copy_d_cds
WHERE year = '2000'
WITH READ ONLY;

SELECT * FROM read_copy_d_cds;
```



The screenshot shows the Oracle SQL Workshop interface. In the SQL Commands tab, the following SQL code is entered:

```
1 CREATE OR REPLACE VIEW read_copy_d_cds AS
2 SELECT *
3 FROM copy_d_cds
4 WHERE year = '2000'
5 WITH READ ONLY;
```

After running the command, the Results tab displays the message "View created." and a execution time of "0.04 seconds".



The screenshot shows the Oracle SQL Workshop interface. In the SQL Commands tab, the following SQL code is entered:

```
1 SELECT * FROM read_copy_d_cds;
```

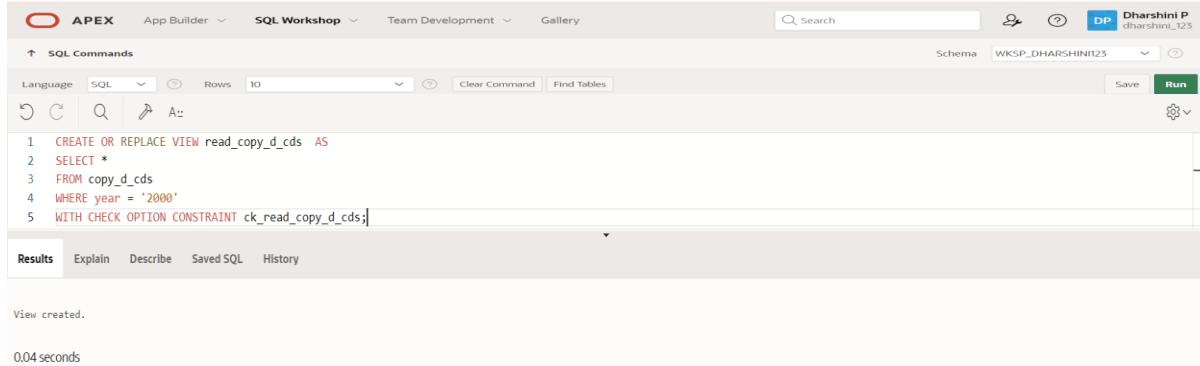
After running the command, the Results tab displays the message "no data found".

5. Using the read\_copy\_d\_cds view, execute a DELETE FROM read\_copy\_d\_cds WHERE cd\_number = 90;

**ORA-42399: cannot perform a DML operation on a read-only view**

6. Use REPLACE to modify read\_copy\_d\_cds. Replace the READ ONLY option with WITH CHECK OPTION CONSTRAINT ck\_read\_copy\_d\_cds. Execute a SELECT \* statement to verify that the view exists.

```
CREATE OR REPLACE VIEW read_copy_d_cds AS
SELECT *
FROM copy_d_cds
WHERE year = '2000'
WITH CHECK OPTION CONSTRAINT ck_read_copy_d_cds;
```



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows a user profile for 'Dharshini P' with the schema 'WKSP\_DHARSHINI123'. The main area displays the SQL command for creating the view:

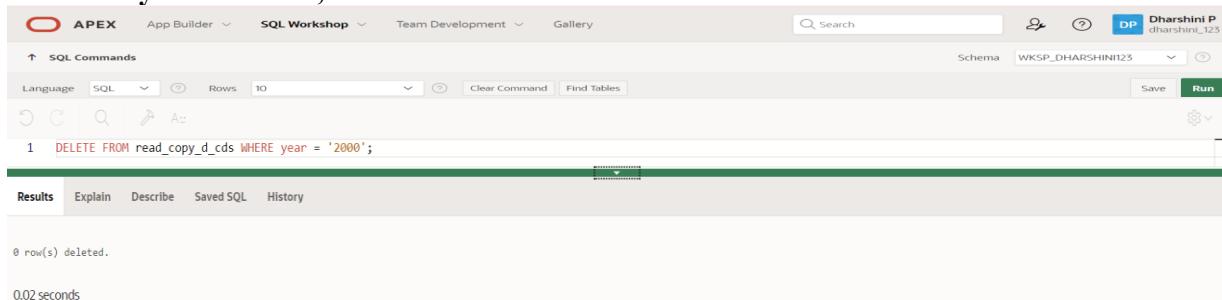
```
1 CREATE OR REPLACE VIEW read_copy_d_cds AS
2 SELECT *
3 FROM copy_d_cds
4 WHERE year = '2000'
5 WITH CHECK OPTION CONSTRAINT ck_read_copy_d_cds;
```

Below the command, the results show:

View created.  
0.04 seconds

7. Use the read\_copy\_d\_cds view to delete any CD of year 2000 from the underlying copy\_d\_cds.

```
DELETE FROM read_copy_d_cds
WHERE year = '2000';
```



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows a user profile for 'Dharshini P' with the schema 'WKSP\_DHARSHINI123'. The main area displays the SQL command for deleting a row from the view:

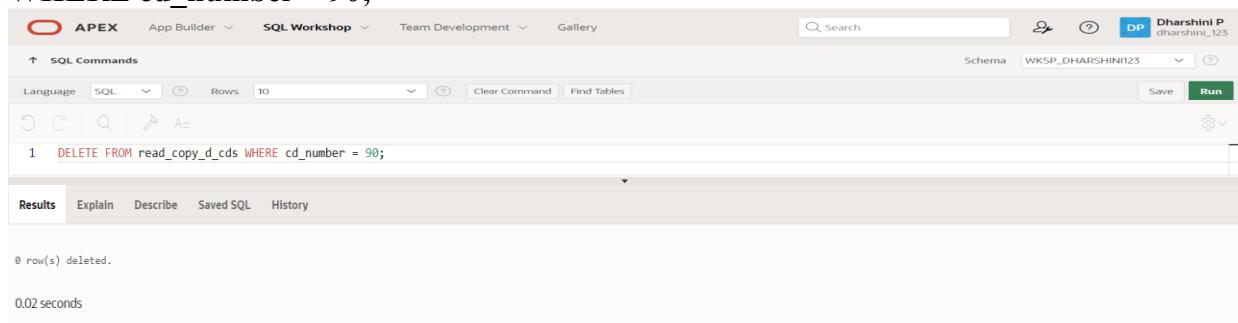
```
1 DELETE FROM read_copy_d_cds WHERE year = '2000';
```

Below the command, the results show:

8 row(s) deleted.  
0.02 seconds

8. Use the read\_copy\_d\_cds view to delete cd\_number 90 from the underlying copy\_d\_cds table.

```
DELETE FROM read_copy_d_cds
WHERE cd_number = 90;
```



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows a user profile for 'Dharshini P' with the schema 'WKSP\_DHARSHINI123'. The main area displays the SQL command for deleting a row from the view:

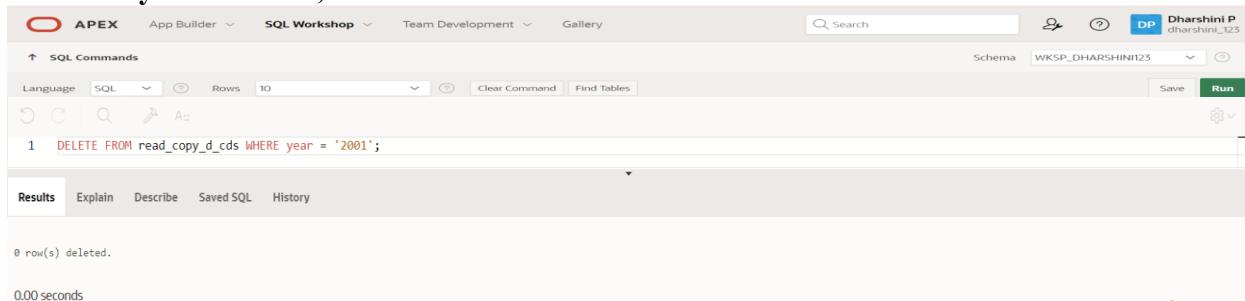
```
1 DELETE FROM read_copy_d_cds WHERE cd_number = 90;
```

Below the command, the results show:

0 row(s) deleted.  
0.02 seconds

9. Use the read\_copy\_d\_cds view to delete year 2001 records.

**DELETE FROM read\_copy\_d\_cds  
WHERE year = '2001';**



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows the user 'Dharshini P' and schema 'WKSP\_DHARSHINI123'. The main area has tabs for SQL Commands, Language (SQL selected), Rows (10), Clear Command, Find Tables, Save, and Run. Below these tabs, the command '1 DELETE FROM read\_copy\_d\_cds WHERE year = '2001'' is entered. The results tab shows '0 row(s) deleted.' and a duration of '0.00 seconds'.

10. Execute a SELECT \* statement for the base table copy\_d\_cds. What rows were deleted?

**Only the one in problem 7 above, not the one in 8 and 9**

11. What are the restrictions on modifying data through a view?

**DELETE,INSERT,MODIFY restricted if it contains:**

**Group functions  
GROUP BY CLAUSE  
DISTINCT  
pseudocolumn ROWNUM Keyword**

12. What is Moore's Law? Do you consider that it will continue to apply indefinitely? Support your opinion with research from the internet.

**It roughly predicted that computing power nearly doubles every year. But Moore also said in 2005 that as per nature of exponential functions, this trend may not continue forever.**

13. What is the "singularity" in terms of computing?

**Singularity is the hypothesis that the invention of artificial superintelligence will abruptly trigger runaway technological growth, resulting in unfathomable changes to human civilization**

## Managing Views

1. Create a view from the copy\_d\_songs table called view\_copy\_d\_songs that includes only the title and artist. Execute a SELECT \* statement to verify that the view exists.

```
CREATE OR REPLACE VIEW view_copy_d_songs AS
SELECT title, artist
FROM copy_d_songs;
```

```
SELECT * FROM view_copy_d_songs;
```

The screenshot shows two sessions in the Oracle SQL Workshop. The top session shows the creation of a view:

```
1 CREATE OR REPLACE VIEW view_copy_d_songs AS SELECT title, artist FROM copy_d_songs;
```

The message "View created." is displayed below the command. The bottom session shows the execution of the view:

```
1 SELECT * FROM view_copy_d_songs;
```

The results table displays one row:

TITLE	ARTIST
Mello Jello	The What

1 rows returned in 0.01 seconds

2. Issue a DROP view\_copy\_d\_songs. Execute a SELECT \* statement to verify that the view has been deleted.

```
DROP VIEW view_copy_d_songs;
SELECT * FROM view_copy_d_songs;
```

**ORA-00942: table or view does not exist**

The screenshot shows the execution of a drop command followed by a select statement:

```
1 DROP VIEW view_copy_d_songs;
```

The message "View dropped." is displayed below the command. The bottom session shows the execution of the view:

```
1 SELECT * FROM view_copy_d_songs;
```

The message "ORA-00942: table or view does not exist" is displayed, indicating the view no longer exists.

3. Create a query that selects the last name and salary from the Oracle database. Rank the salaries from highest to lowest for the top three employees.

```
SELECT * FROM
(SELECT last_name, salary FROM employees ORDER BY salary DESC)
WHERE ROWNUM <= 3;
```

The screenshot shows the Oracle SQL Workshop interface. The query is executed and returns three rows:

LAST_NAME	SALARY
bhuvana	100000
Matos	5000
Davies	34000

3 rows returned in 0.02 seconds [Download](#)

4. Construct an inline view from the Oracle database that lists the last name, salary, department ID, and maximum salary for each department. Hint: One query will need to calculate maximum salary by department ID.

```
SELECT empm.last_name, empm.salary, dptmx.department_id
FROM
(SELECT dpt.department_id, MAX(NVL(emp.salary,0)) max_dpt_sal
FROM departments dpt LEFT OUTER JOIN employees emp ON dpt.department_id =
emp.department_id
GROUP BY dpt.department_id) dptmx LEFT OUTER JOIN employees empm ON
dptmx.department_id = empm.department_id
WHERE NVL(empm.salary,0) = dptmx.max_dpt_sal;
```

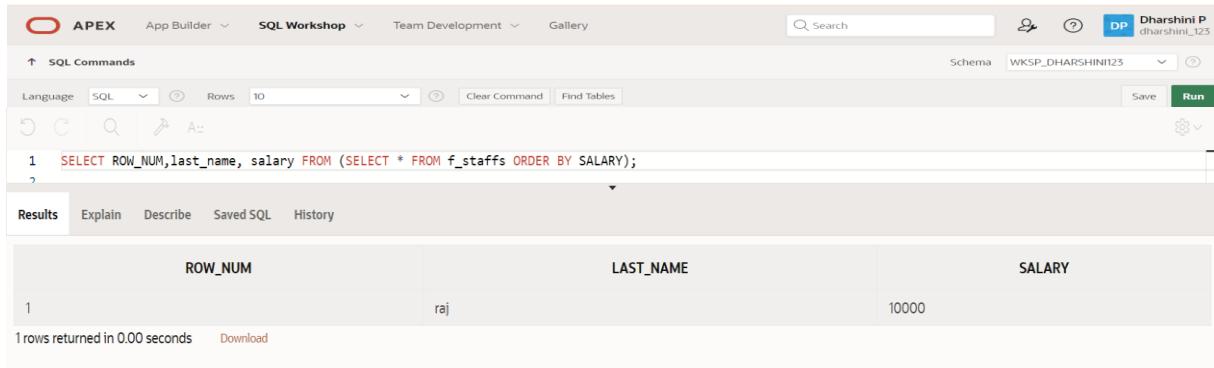
The screenshot shows the Oracle SQL Workshop interface. The query is executed and returns three rows:

LAST_NAME	SALARY	DEPT_ID
brindha	12222	59
bhuvana	100000	1
Zlotkey	3200	2222

3 rows returned in 0.01 seconds [Download](#)

5. Create a query that will return the staff members of Global Fast Foods ranked by salary from lowest to highest.

```
SELECT ROW_NUM,last_name, salary  
FROM  
(SELECT * FROM f_staffs ORDER BY SALARY);
```



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows a user profile for 'Dharshini P' with a schema dropdown set to 'WKSP\_DHARSHINI123'. The main area has tabs for SQL Commands, SQL (selected), and Rows (set to 10). Below these are icons for Undo, Redo, Search, and Find Tables. The SQL editor contains the query: 'SELECT ROW\_NUM, last\_name, salary FROM (SELECT \* FROM f\_staffs ORDER BY SALARY);'. The results tab is selected, displaying a single row: Row Num 1, Last Name raj, and Salary 10000. A note at the bottom says '1 rows returned in 0.00 seconds'.

## Indexes and Synonyms

1. What is an index and what is it used for?

**Definition:** These are schema objects which make retrieval of rows from table faster.

**Purpose:** An index provides direct and fast access to row in table. They provide indexed path to locate data quickly, so hereby reduce necessity of heavy disk input/output operations.

2. What is a ROWID, and how is it used?

**Indexes use ROWID's (base 64 string representation of the row address containing block identifier, row location in the block and the database file identifier) which is the fastest way to access any particular row.**

3. When will an index be created automatically?

**Primary key/unique key use already existing unique index but if index is not present already, it is created while applying unique/primary key constraint.**

4. Create a nonunique index (foreign key) for the DJs on Demand column (cd\_number) in the D\_TRACK\_LISTINGS table. Use the Oracle Application Express SQL Workshop Data Browser to confirm that the index was created.

```
CREATE INDEX d_tlg_cd_number_fk_i
on d_track_listings (cd_number);
```

The screenshot shows the Oracle Application Express SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. The right side shows a user profile for 'Dharshini P' with the schema 'WKSP\_DHARSHINI123'. The main area is titled 'SQL Commands' with a sub-section 'Language: SQL'. A command line shows the creation of an index:

```
1 CREATE INDEX d_tlg_cd_number_fk_i on d_track_listings (cd_number);
```

The results section below shows the output:

```
Index created.
0.02 seconds
```

5. Use the join statement to display the indexes and uniqueness that exist in the data dictionary for the DJs on Demand D\_SONGS table.

```
SELECT ucm.index_name, ucm.column_name, ucm.column_position, uix.uniqueness
FROM user_indexes uix INNER JOIN user_ind_columns ucm ON uix.index_name =
ucm.index_name
WHERE ucm.table_name = 'D_SONGS';
```

The screenshot shows the Oracle Application Express SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. The right side shows a user profile for 'Dharshini P' with the schema 'WKSP\_DHARSHINI123'. The main area is titled 'SQL Commands' with a sub-section 'Language: SQL'. A command line shows a query to select indexes from the data dictionary:

```
1 SELECT ucm.index_name, ucm.column_name, ucm.column_position, uix.uniqueness
2 FROM user_indexes uix INNER JOIN user_ind_columns ucm ON uix.index_name = ucm.index_name
3 WHERE ucm.table_name = 'D_SONGS';
```

The results section below shows the output:

```
no data found
```

6. Use a SELECT statement to display the index\_name, table\_name, and uniqueness from the data dictionary USER\_INDEXES for the DJs on Demand D\_EVENTS table.

```
SELECT index_name, table_name, uniqueness FROM user_indexes where table_name =
'D_EVENTS';
```

The screenshot shows the Oracle Application Express SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. The right side shows a user profile for 'Dharshini P' with the schema 'WKSP\_DHARSHINI123'. The main area is titled 'SQL Commands' with a sub-section 'Language: SQL'. A command line shows a query to select indexes from the data dictionary:

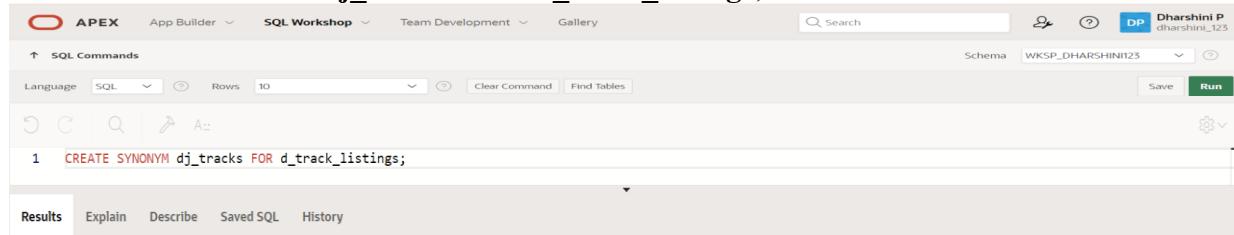
```
1 SELECT index_name, table_name, uniqueness FROM user_indexes where table_name = 'D_EVENTS';
```

The results section below shows the output:

```
no data found
```

7. Write a query to create a synonym called dj\_tracks for the DJs on Demand d\_track\_listings table.

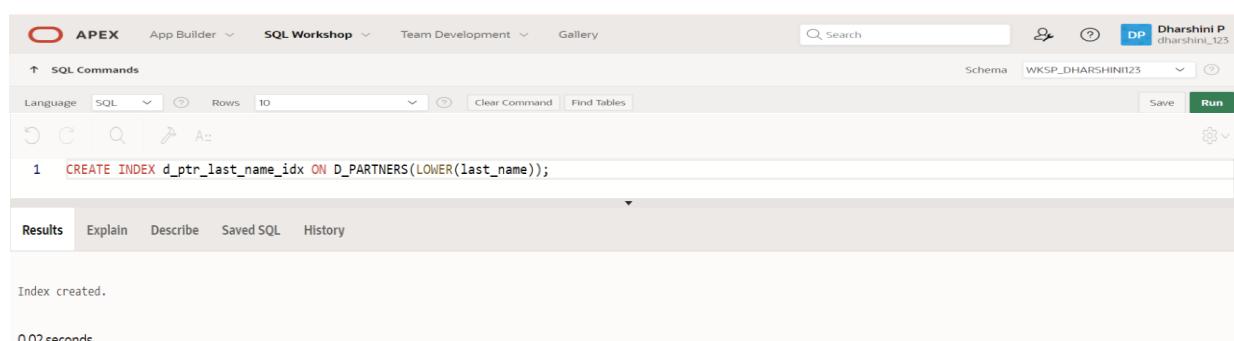
**CREATE SYNONYM dj\_tracks FOR d\_track\_listings;**



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there's a user profile for 'Dharshini P' and a schema dropdown set to 'WKSP\_DHARSHINI123'. The main area has tabs for SQL Commands, Results (selected), Explain, Describe, Saved SQL, and History. The SQL Commands tab contains the command: '1 CREATE SYNONYM dj\_tracks FOR d\_track\_listings;'. Below the command, the results show 'Synonym created.' and a execution time of '0.01 seconds'.

8. Create a function-based index for the last\_name column in DJs on Demand D\_PARTNERS table that makes it possible not to have to capitalize the table name for searches. Write a SELECT statement that would use this index.

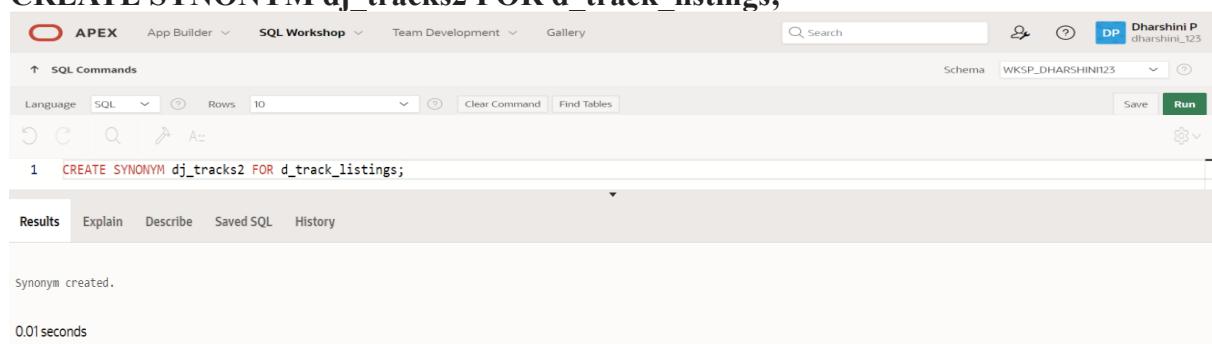
**CREATE INDEX d\_ptr\_last\_name\_idx  
ON d\_partners(LOWER(last\_name));**



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there's a user profile for 'Dharshini P' and a schema dropdown set to 'WKSP\_DHARSHINI123'. The main area has tabs for SQL Commands, Results (selected), Explain, Describe, Saved SQL, and History. The SQL Commands tab contains the command: '1 CREATE INDEX d\_ptr\_last\_name\_idx ON D\_PARTNERS(LOWER(last\_name));'. Below the command, the results show 'Index created.' and a execution time of '0.02 seconds'.

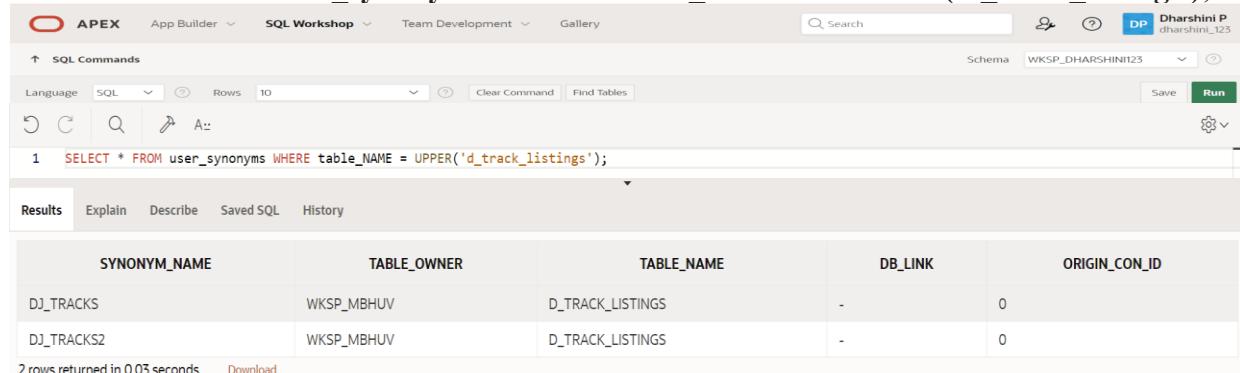
9. Create a synonym for the D\_TRACK\_LISTINGS table. Confirm that it has been created by querying the data dictionary.

**CREATE SYNONYM dj\_tracks2 FOR d\_track\_listings;**



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there's a user profile for 'Dharshini P' and a schema dropdown set to 'WKSP\_DHARSHINI123'. The main area has tabs for SQL Commands, Results (selected), Explain, Describe, Saved SQL, and History. The SQL Commands tab contains the command: '1 CREATE SYNONYM dj\_tracks2 FOR d\_track\_listings;'. Below the command, the results show 'Synonym created.' and a execution time of '0.01 seconds'.

```
SELECT * FROM user_synonyms WHERE table_NAME = UPPER('d_track_listings');
```



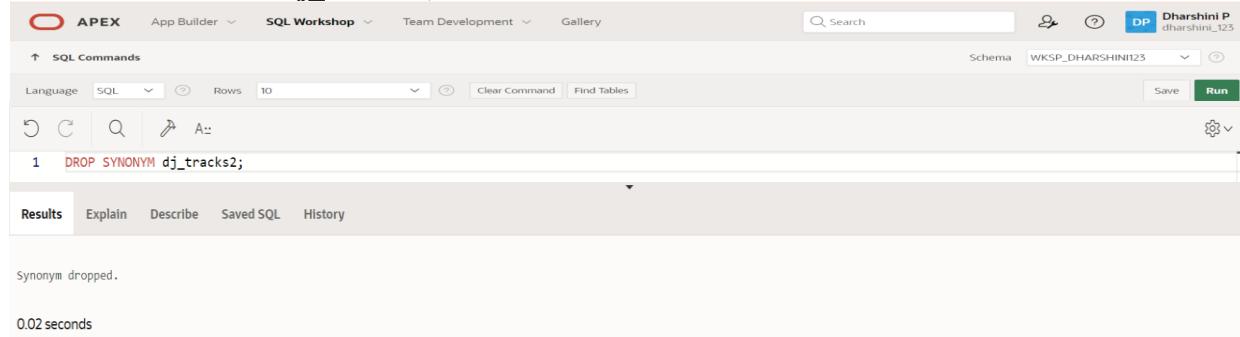
The screenshot shows the Oracle SQL Workshop interface. The query `SELECT * FROM user_synonyms WHERE table_NAME = UPPER('d_track_listings');` has been run. The results table has the following data:

SYNONYM_NAME	TABLE_OWNER	TABLE_NAME	DB_LINK	ORIGIN_CON_ID
DJ_TRACKS	WKSP_MBHV	D_TRACK_LISTINGS	-	0
DJ_TRACKS2	WKSP_MBHV	D_TRACK_LISTINGS	-	0

2 rows returned in 0.03 seconds [Download](#)

10. Drop the synonym that you created in question

```
DROP SYNONYM dj_tracks2;
```



The screenshot shows the Oracle SQL Workshop interface. The query `DROP SYNONYM dj_tracks2;` has been run. The results show the message "Synonym dropped." and a execution time of "0.02 seconds".

Evaluation Procedure	Marks awarded
Query(5)	
Execution (5)	
Viva(5)	
Total (15)	
Faculty Signature	

**RESULT:**

## OTHER DATABASE OBJECTS

EX.NO:14

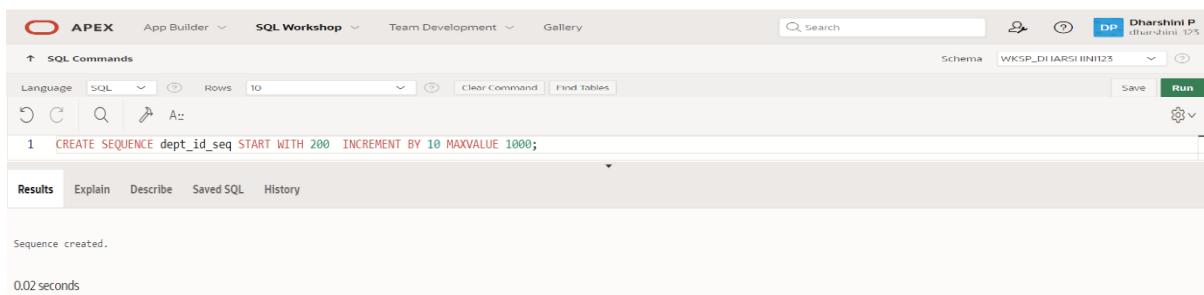
DATE:

1.)Create a sequence to be used with the primary key column of the DEPT table. The sequence should start at 200 and have a maximum value of 1000. Have your sequence increment by ten numbers. Name the sequence DEPT\_ID\_SEQ

**QUERY:**

**CREATE SEQUENCE dept\_id\_seq START WITH 200 INCREMENT BY 10 MAXVALUE 1000;**

**OUTPUT:**



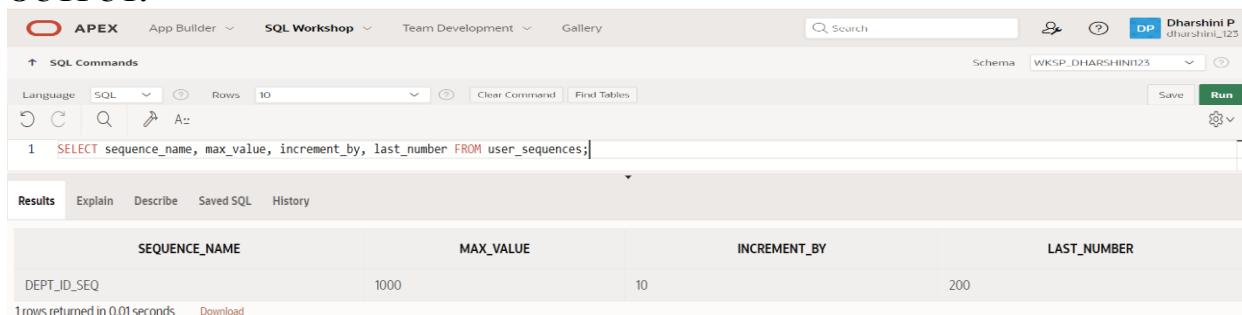
A screenshot of the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows a user profile for 'Dharshini P dharshini\_123'. The main area has tabs for SQL Commands, Results, Explain, Describe, Saved SQL, and History. The SQL Commands tab shows the command: 'CREATE SEQUENCE dept\_id\_seq START WITH 200 INCREMENT BY 10 MAXVALUE 1000;'. Below the command, the results show 'Sequence created.' and a execution time of '0.02 seconds'.

2.)Write a query in a script to display the following information about your sequences: sequence name, maximum value, increment size, and last number

**QUERY:**

**SELECT sequence\_name, max\_value, increment\_by, last\_number FROM user\_sequences;**

**OUTPUT:**



A screenshot of the Oracle SQL Workshop interface, similar to the previous one. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows a user profile for 'Dharshini P dharshini\_123'. The main area has tabs for SQL Commands, Results, Explain, Describe, Saved SQL, and History. The SQL Commands tab shows the command: 'SELECT sequence\_name, max\_value, increment\_by, last\_number FROM user\_sequences;'. Below the command, the results table displays the following data:

SEQUENCE_NAME	MAX_VALUE	INCREMENT_BY	LAST_NUMBER
DEPT_ID_SEQ	1000	10	200

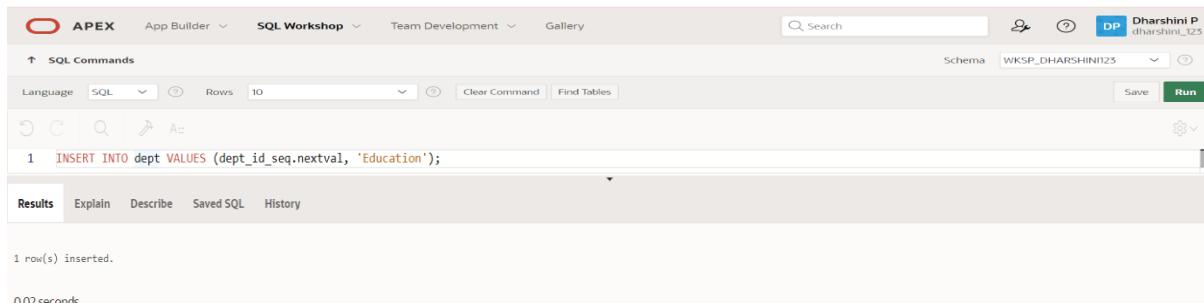
At the bottom, it says '1rows returned in 0.01 seconds' and has a 'Download' link.

3.) Write a script to insert two rows into the DEPT table. Name your script lab12\_3.sql. Be sure to use the sequence that you created for the ID column. Add two departments named Education and Administration. Confirm your additions. Run the commands in your script.

**QUERY:**

```
INSERT INTO dept VALUES (dept_id_seq.nextval, 'Education');
INSERT INTO dept VALUES (dept_id_seq.nextval, 'Administration');
```

**OUTPUT:**



The screenshot shows the Oracle SQL Workshop interface. In the SQL Commands pane, two INSERT statements are run:

```
1 INSERT INTO dept VALUES (dept_id_seq.nextval, 'Education');
```

The Results pane shows the output:

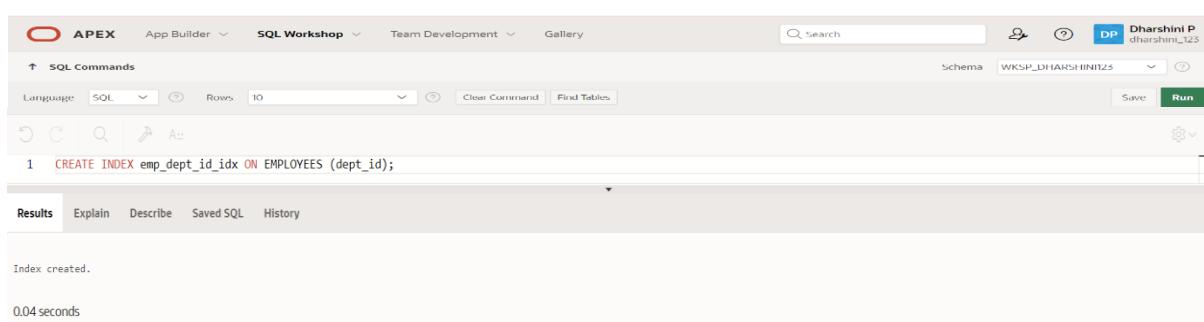
1 row(s) inserted.  
0.02 seconds

4.) Create a nonunique index on the foreign key column (DEPT\_ID) in the EMP table.

**QUERY:**

```
CREATE INDEX emp_dept_id_idx ON EMPLOYEES (department_id);
```

**OUTPUT:**



The screenshot shows the Oracle SQL Workshop interface. A CREATE INDEX statement is run:

```
1 CREATE INDEX emp_dept_id_idx ON EMPLOYEES (dept_id);
```

The Results pane shows the output:

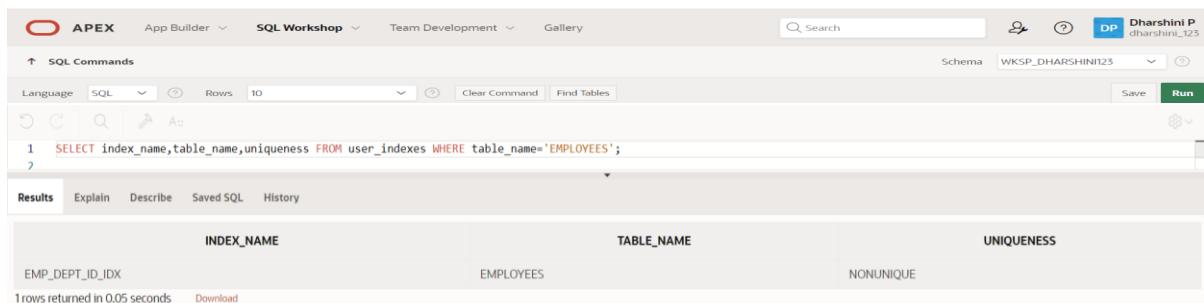
Index created.  
0.04 seconds

5.) Display the indexes and uniqueness that exist in the data dictionary for the EMP table.

**QUERY:**

```
SELECT index_name,table_name,uniqueness FROM user_indexes WHERE
table_name='EMPLOYEES';
```

**OUTPUT:**



The screenshot shows the Oracle SQL Workshop interface. A SELECT statement is run to query the user\_indexes data dictionary view:

```
1 SELECT index_name,table_name,uniqueness FROM user_indexes WHERE table_name='EMPLOYEES';
2
```

The Results pane shows the output:

INDEX_NAME	TABLE_NAME	UNIQUENESS
EMP_DEPT_ID_IDX	EMPLOYEES	NONUNIQUE

1 rows returned in 0.05 seconds    Download

Evaluation Procedure	Marks awarded
Query(5)	
Execution (5)	
Viva(5)	
Total (15)	
Faculty Signature	

**RESULT:**

## **CONTROLLING USER ACCESS**

**EX.NO:15**

**DATE:**

1. What privilege should a user be given to log on to the Oracle Server? Is this a system or an object privilege?

The CREATE SESSION system privilege

2. What privilege should a user be given to create tables?

The CREATE TABLE privilege

3. If you create a table, who can pass along privileges to other users on your table?

You can, or anyone you have given those privileges to by using the WITH GRANT OPTION.

4. You are the DBA. You are creating many users who require the same system privileges. What should you use to make your job easier?

Create a role containing the system privileges and grant the role to the users

5. What command do you use to change your password?

The ALTER USER statement

6. Grant another user access to your DEPARTMENTS table. Have the user grant you query access to his or her DEPARTMENTS table.

Team 2 executes the GRANT statement.      GRANT select ON departments TO <user1>;

Team 1 executes the GRANT statement.      GRANT select ON departments TO <user2>;

7. Query all the rows in your DEPARTMENTS table.

SELECT \* FROM departments;

8. Add a new row to your DEPARTMENTS table. Team 1 should add Education as department number 500. Team 2 should add Human Resources department number 510. Query the other team's table.

Team 1 executes this INSERT statement. INSERT INTO departments(department\_id, department\_name) VALUES (500, 'Education'); COMMIT;

Team 2 executes this INSERT statement. INSERT INTO departments(department\_id, department\_name) VALUES (510, 'Administration'); COMMIT;

9. Query the USER\_TABLES data dictionary to see information about the tables that you own.

```
SELECT table_name FROM user_tables;
```

10. Revoke the SELECT privilege on your table from the other team.

Team 1 revokes the privilege.

```
REVOKE select  
ON departments  
FROM user2;
```

Team 2 revokes the privilege.

```
REVOKE select  
ON departments  
FROM user1;
```

11. Remove the row you inserted into the DEPARTMENTS table in step 8 and save the changes.

Team 1 executes this INSERT statement.

```
DELETE FROM departments  
WHERE department_id = 500;  
COMMIT;
```

Team 2 executes this INSERT statement.

```
DELETE FROM departments  
WHERE department_id = 510;  
COMMIT;
```

Evaluation Procedure	Marks awarded
<u>Practice Evaluation (5)</u>	
<u>Viva(5)</u>	
<u>Total (10)</u>	
<u>Faculty Signature</u>	

**RESULT:**

## **PL/SQL** **CONTROL STRUCTURES**

**EX.NO:16**

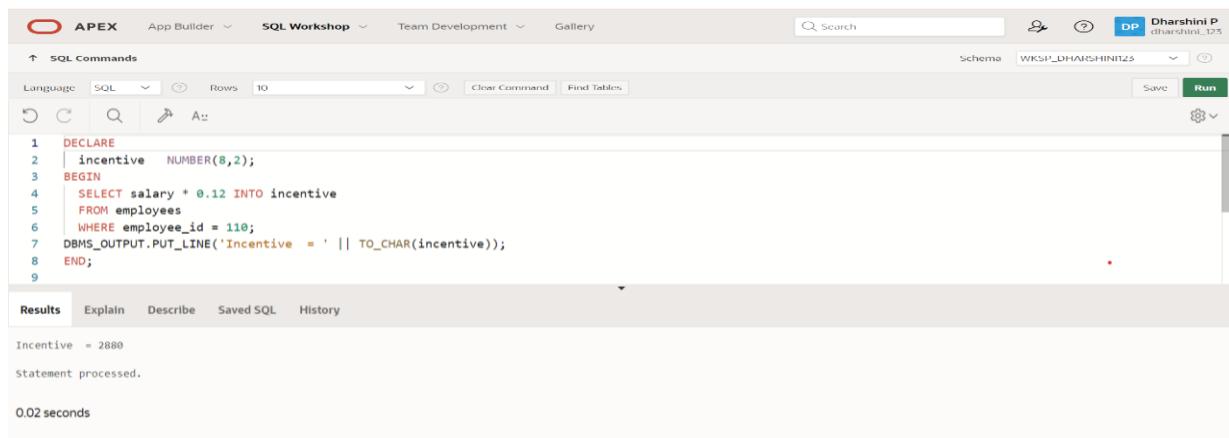
**DATE:**

1.) Write a PL/SQL block to calculate the incentive of an employee whose ID is 110.

**QUERY:**

```
DECLARE
incentive NUMBER(8,2);
BEGIN
SELECT salary*0.12 INTO incentive
FROM employees
WHERE employee_id = 110;
DBMS_OUTPUT.PUT_LINE('Incentive = ' || TO_CHAR(incentive));
END;
```

**OUTPUT:**



The screenshot shows the Oracle SQL Workshop interface. The command window displays the following PL/SQL code:

```
1 DECLARE
2   | incentive  NUMBER(8,2);
3 BEGIN
4   | SELECT salary * .12 INTO incentive
5   | FROM employees
6   | WHERE employee_id = 110;
7   DBMS_OUTPUT.PUT_LINE('Incentive = ' || TO_CHAR(incentive));
8 END;
9 
```

The results pane shows the output of the query:

```
Incentive = 2880
Statement processed.

0.02 seconds
```

2.) Write a PL/SQL block to show an invalid case-insensitive reference to a quoted and without quoted user-defined identifier

**QUERY:**

```
DECLARE
WELCOME varchar2(10) := 'welcome';
BEGIN
DBMS_Output.Put_Line("Welcome");
END;
/
```

```
DECLARE
WELCOME varchar2(10) := 'welcome';
BEGIN
DBMS_Output.Put_Line("Welcome");
END;
/
```

**OUTPUT:**

The screenshot shows the Oracle SQL Workshop interface. In the SQL Commands tab, a PL/SQL block is entered. The code includes a declaration of a variable named WELCOME without quotes, followed by a BEGIN block containing a DBMS\_OUTPUT.PUT\_LINE statement with the string "Welcome". The code ends with an END; statement and a final slash. A tooltip highlights the line 'WELCOME varchar2(10) := 'welcome'' with the note 'identifier without quotation'. The results section displays an error message: 'Error at line 4/25: ORA-06550: line 4, column 25: PLS-00201: identifier 'Welcome' must be declared ORA-06512: at "SYS.MVW\_DBMS\_SQL\_APEX\_230200", line 801 ORA-06550: line 4, column 3: PL/SQL: Statement ignored'. Below the error message, the original code is shown again.

```
1. DECLARE
2. | WELCOME varchar2(10) := 'welcome'; -- identifier without quotation
3. BEGIN
4. | DBMS_Output.Put_Line("Welcome"); --reference to the identifier with quotation and different case
5. END;
```

Error at line 4/25: ORA-06550: line 4, column 25:  
PLS-00201: identifier 'Welcome' must be declared  
ORA-06512: at "SYS.MVW\_DBMS\_SQL\_APEX\_230200", line 801  
ORA-06550: line 4, column 3:  
PL/SQL: Statement ignored

```
2. WELCOME varchar2(10) := 'welcome'; -- identifier without quotation
3. BEGIN
4. DBMS_Output.Put_Line("Welcome"); --reference to the identifier with quotation
   and different case
5. END;
```

3.) Write a PL/SQL block to adjust the salary of the employee whose ID 122.

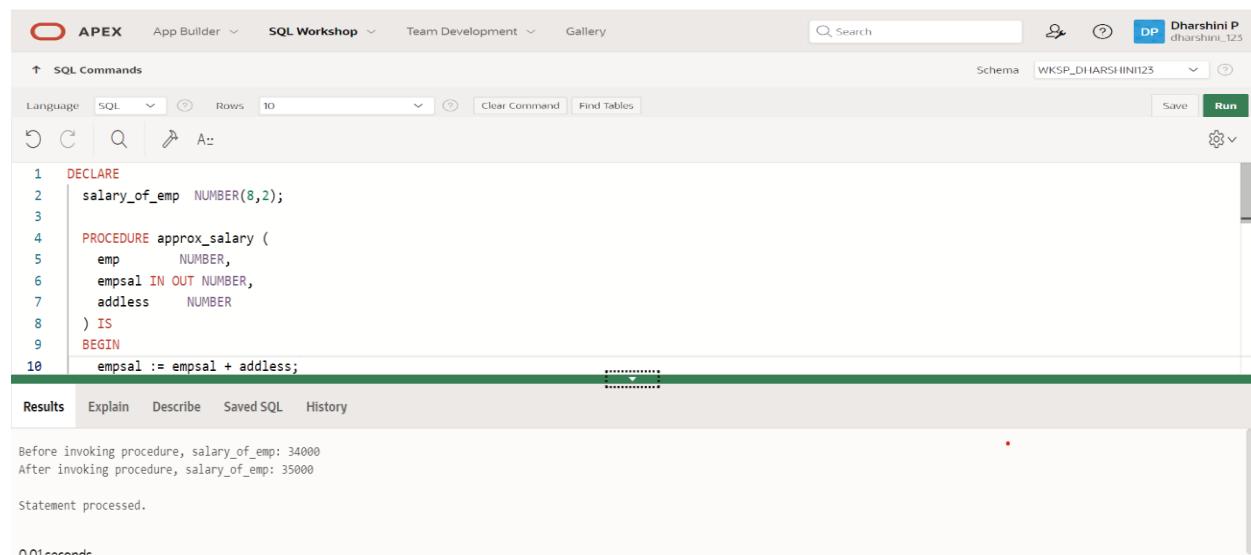
#### QUERY:

```
DECLARE
    salary_of_emp NUMBER(8,2);
PROCEDURE approx_salary (
    emp      NUMBER,
    empsal IN OUT NUMBER,
    addless  NUMBER
) IS
BEGIN
    empsal := empsal + addless;
END;
```

```
BEGIN
    SELECT salary INTO salary_of_emp
    FROM employees
    WHERE employee_id = 122;
    DBMS_OUTPUT.PUT_LINE
        ('Before invoking procedure, salary_of_emp: ' || salary_of_emp);
    approx_salary (100, salary_of_emp, 1000);
    DBMS_OUTPUT.PUT_LINE
        ('After invoking procedure, salary_of_emp: ' || salary_of_emp);
END;
/
```

#### OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows a user profile for 'Dharshini P dharshini\_123'. The main area is titled 'SQL Commands' with tabs for Language (SQL selected), Rows (10), and various icons. The code area contains the PL/SQL block from the previous section. The results tab at the bottom shows the output of the procedure execution:

```
Before invoking procedure, salary_of_emp: 34000
After invoking procedure, salary_of_emp: 35000
Statement processed.

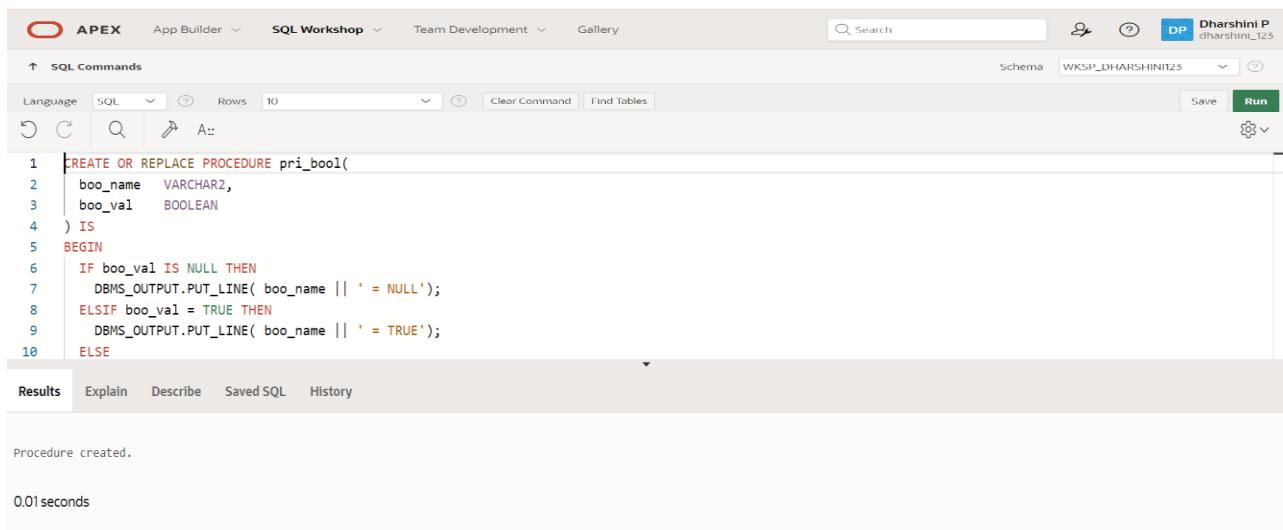
0.01 seconds
```

4.) Write a PL/SQL block to create a procedure using the "IS [NOT] NULL Operator" and show AND operator returns TRUE if and only if both operands are TRUE.

#### QUERY:

```
CREATE OR REPLACE PROCEDURE pri_bool(
    boo_name  VARCHAR2,
    boo_val   BOOLEAN
) IS
BEGIN
    IF boo_val IS NULL THEN
        DBMS_OUTPUT.PUT_LINE( boo_name || ' = NULL');
    ELSIF boo_val = TRUE THEN
        DBMS_OUTPUT.PUT_LINE( boo_name || ' = TRUE');
    ELSE
        DBMS_OUTPUT.PUT_LINE( boo_name || ' = FALSE');
    END IF;
END;
/
```

#### OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. In the top navigation bar, 'APEX' is selected. The main area is titled 'SQL Commands'. The code entered is the PL/SQL block provided in the question. After running the command, the results pane shows the message 'Procedure created.' and a execution time of '0.01 seconds'.

```
1 CREATE OR REPLACE PROCEDURE pri_bool(
2     boo_name  VARCHAR2,
3     boo_val   BOOLEAN
4 ) IS
5 BEGIN
6     IF boo_val IS NULL THEN
7         DBMS_OUTPUT.PUT_LINE( boo_name || ' = NULL');
8     ELSIF boo_val = TRUE THEN
9         DBMS_OUTPUT.PUT_LINE( boo_name || ' = TRUE');
10    ELSE
11    END IF;
12 END;
13 /
```

Results Explain Describe Saved SQL History

Procedure created.  
0.01 seconds

5.) Write a PL/SQL block to describe the usage of LIKE operator including wildcard characters and escape character.

#### QUERY:

DECLARE

```
PROCEDURE pat_match (
    test_string  VARCHAR2,
    pattern      VARCHAR2
) IS
BEGIN
    IF test_string LIKE pattern THEN
        DBMS_OUTPUT.PUT_LINE ('TRUE');
    ELSE
        DBMS_OUTPUT.PUT_LINE ('FALSE');
    END IF;
END;
BEGIN
    pat_match('Blweate', 'B%a_e');
    pat_match('Blweate', 'B%A_E');
END;
/
```

#### OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows the user Dharshini P (dharshini\_123). The main area displays the PL/SQL code for the pat\_match procedure. Below the code, the Results tab is selected, showing the output: TRUE and FALSE, indicating the procedure correctly handles both patterns. The status bar at the bottom indicates the statement was processed in 0.01 seconds.

```
1  DECLARE
2  PROCEDURE pat_match (
3      test_string  VARCHAR2,
4      pattern      VARCHAR2
5  ) IS
6  BEGIN
7      IF test_string LIKE pattern THEN
8          DBMS_OUTPUT.PUT_LINE ('TRUE');
9      ELSE
```

Results	Explain	Describe	Saved SQL	History
TRUE FALSE Statement processed. 0.01 seconds				

6.) Write a PL/SQL program to arrange the number of two variable in such a way that the small number will store in num\_small variable and large number will store in num\_large variable

#### QUERY:

DECLARE

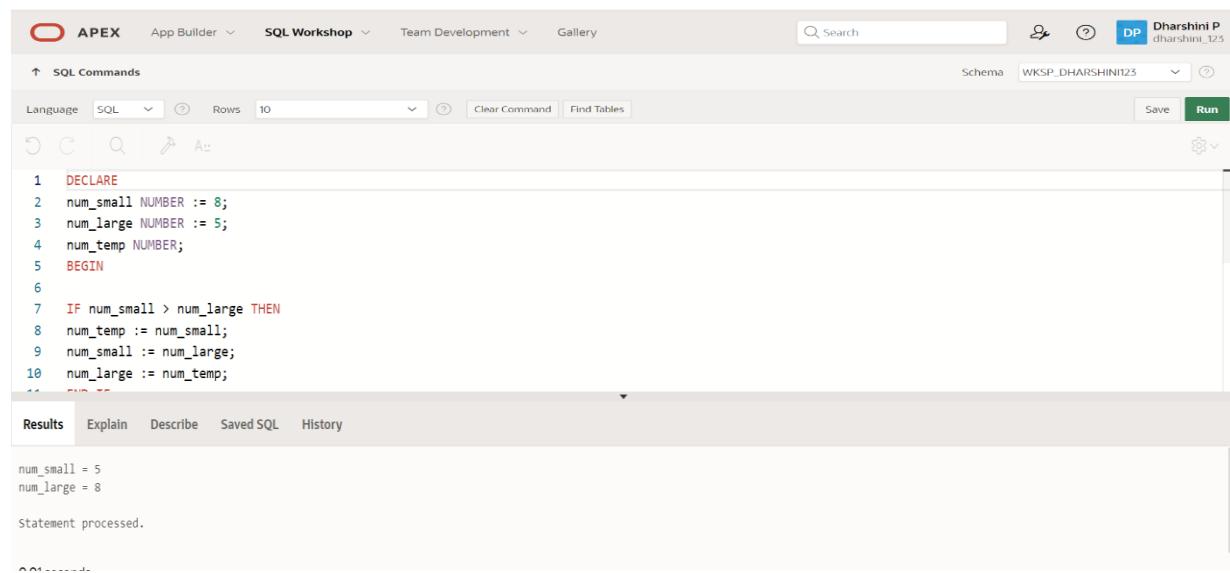
```
num_small NUMBER := 8;
num_large NUMBER := 5;
num_temp NUMBER;
```

BEGIN

```
IF num_small > num_large THEN
    num_temp := num_small;
    num_small := num_large;
    num_large := num_temp;
END IF;
```

```
DBMS_OUTPUT.PUT_LINE ('num_small ='||num_small);
DBMS_OUTPUT.PUT_LINE ('num_large ='||num_large);
END;
```

#### OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. In the SQL Commands pane, a PL/SQL block is written and executed. The output pane displays the results of the DBMS\_OUTPUT.PUT\_LINE statements.

```
1 DECLARE
2 num_small NUMBER := 8;
3 num_large NUMBER := 5;
4 num_temp NUMBER;
5 BEGIN
6
7 IF num_small > num_large THEN
8 num_temp := num_small;
9 num_small := num_large;
10 num_large := num_temp;
11 END IF;
12
13 DBMS_OUTPUT.PUT_LINE ('num_small ='||num_small);
14 DBMS_OUTPUT.PUT_LINE ('num_large ='||num_large);
15
16 END;
```

Results

```
num_small = 5
num_large = 8

Statement processed.
```

7.) Write a PL/SQL procedure to calculate the incentive on a target achieved and display the message either the record updated or not.

**QUERY:**

DECLARE

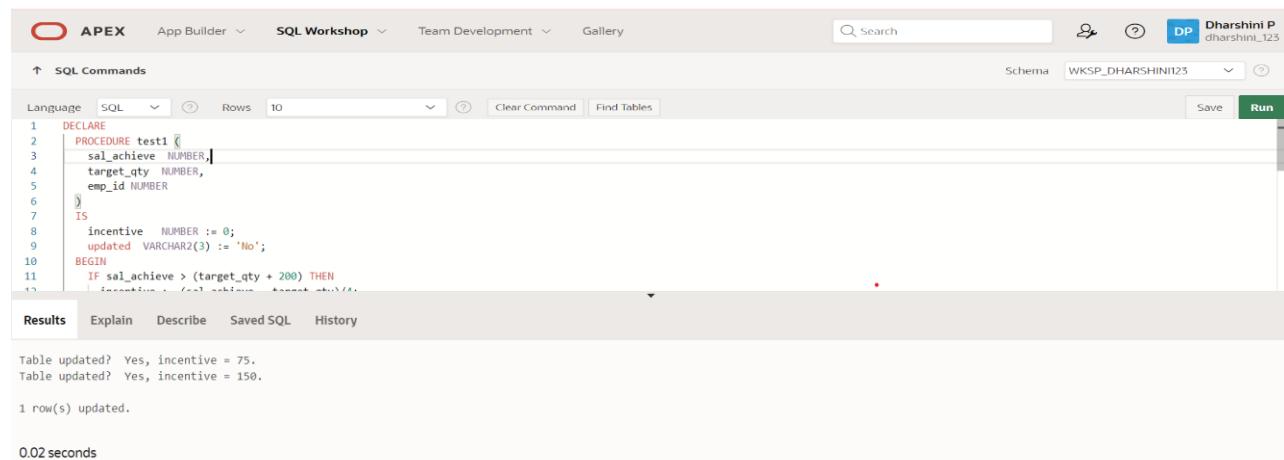
```
PROCEDURE test1 (
    sal_achieve NUMBER,
    target_qty NUMBER,
    emp_id NUMBER
)
IS
    incentive NUMBER := 0;
    updated VARCHAR2(3) := 'No';
BEGIN
    IF sal_achieve > (target_qty + 200) THEN
        incentive := (sal_achieve - target_qty)/4;
        UPDATE employees
        SET salary = salary + incentive
        WHERE employee_id = emp_id;
        updated := 'Yes';
    END IF;
    DBMS_OUTPUT.PUT_LINE (
        'Table updated? ' || updated || ',' ||
        'incentive = ' || incentive || !
    );
END test1;
```

BEGIN

```
test1(2300, 2000, 144);
test1(3600, 3000, 145);
```

END;

**/OUTPUT:**



The screenshot shows the Oracle SQL Workshop interface. In the top navigation bar, 'APEX' is selected. The main area has tabs for 'SQL Commands' (which is active), 'Explain', 'Describe', 'Saved SQL', and 'History'. The SQL Commands pane contains the PL/SQL code provided above. The Results pane displays the following output:

```
Table updated? Yes, incentive = 75.
Table updated? Yes, incentive = 150.
1 row(s) updated.
0.02 seconds
```

8.) Write a PL/SQL procedure to calculate incentive achieved according to the specific sale limit

**QUERY:**

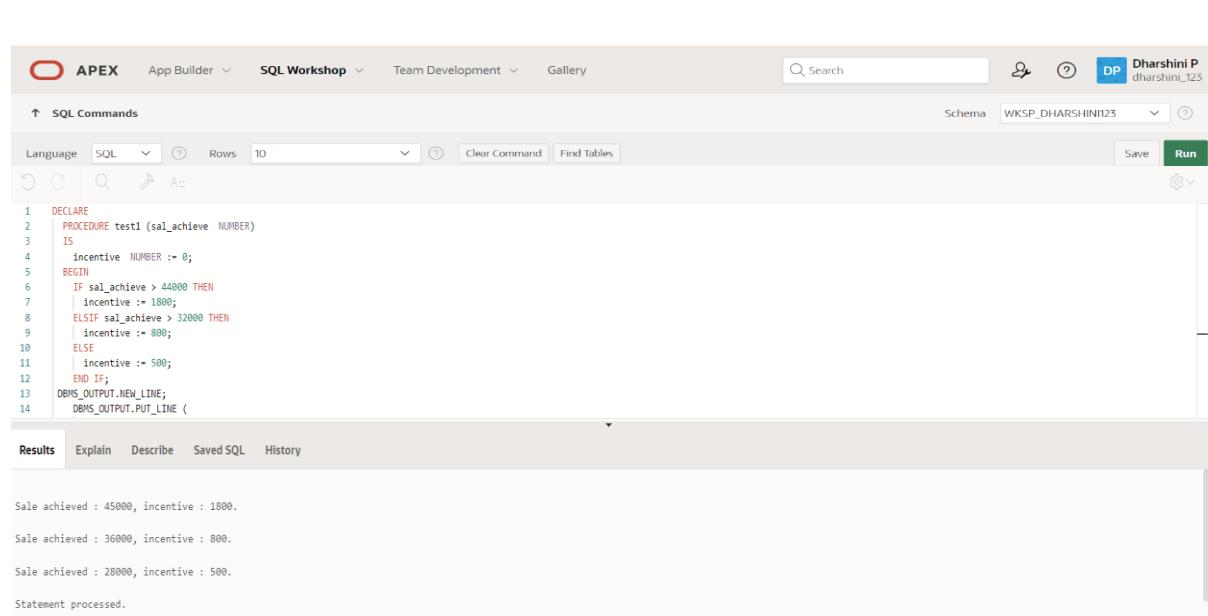
DECLARE

```
PROCEDURE test1 (sal_achieve NUMBER)
IS
    incentive NUMBER := 0;
BEGIN
    IF sal_achieve > 44000 THEN
        incentive := 1800;
    ELSIF sal_achieve > 32000 THEN
        incentive := 800;
    ELSE
        incentive := 500;
    END IF;
    DBMS_OUTPUT.NEW_LINE;
    DBMS_OUTPUT.PUT_LINE (
        'Sale achieved : ' || sal_achieve || ', incentive : ' || incentive || !
    );
END test1;
```

BEGIN

```
    test1(45000);
    test1(36000);
    test1(28000);
END;
```

**OUTPUT:**



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows the user profile 'Dharsini P' and the schema 'WKSP\_DHARSHINI123'. The main area is titled 'SQL Commands' with tabs for Language (set to SQL), Rows (set to 10), and Clear Command. Below this is a toolbar with icons for Undo, Redo, Find, Replace, and Save/Run. The code editor contains the PL/SQL procedure 'test1' and its call from the 'BEGIN' block. The results pane at the bottom displays the output of the executed code.

```
1  DECLARE
2      PROCEDURE test1 (sal_achieve NUMBER)
3  IS
4      incentive NUMBER := 0;
5      BEGIN
6          IF sal_achieve > 44000 THEN
7              incentive := 1800;
8          ELSIF sal_achieve > 32000 THEN
9              incentive := 800;
10         ELSE
11             incentive := 500;
12         END IF;
13     DBMS_OUTPUT.NEW_LINE;
14     DBMS_OUTPUT.PUT_LINE (
```

Sale achieved : 45000, incentive : 1800.  
Sale achieved : 36000, incentive : 800.  
Sale achieved : 28000, incentive : 500.  
Statement processed.

**9.)** Write a PL/SQL program to count number of employees in department 50 and check whether this department have any vacancies or not. There are 45 vacancies in this department.

**QUERY:**

```
SET SERVEROUTPUT ON
DECLARE
    tot_emp NUMBER;
    get_dep_id NUMBER;

BEGIN
    get_dep_id := 80;
    SELECT Count(*)
    INTO tot_emp
    FROM employees e
        join departments d
            ON e.department_id = d.department_id
    WHERE e.department_id = get_dep_id;
    dbms_output.Put_line ('The employees are in the department'||get_dep_id||' is: '
        ||To_char(tot_emp));
    IF tot_emp >= 45 THEN
        dbms_output.Put_line ('There are no vacancies in the department'||get_dep_id);
    ELSE
        dbms_output.Put_line ('There are'||to_char(45-tot_emp)||' vacancies in department'||get_dep_id );
    END IF;
END;
/
```

**OUTPUT:**

**10.)** Write a PL/SQL program to count number of employees in a specific department and check whether this department have any vacancies or not. If any vacancies, how many vacancies are in that department.

**QUERY:**

DECLARE

```
tot_emp NUMBER;  
get_dep_id NUMBER;
```

BEGIN

```
get_dep_id := 80;  
SELECT Count(*)  
INTO tot_emp  
FROM employees e  
join departments d  
ON e.department_id = d.dept_id  
WHERE e.department_id = get_dep_id;
```

```
dbms_output.Put_line ('The employees are in the department'||get_dep_id|| is: '  
||To_char(tot_emp));
```

IF tot\_emp >= 45 THEN

```
dbms_output.Put_line ('There are no vacancies in the department'||get_dep_id);
```

ELSE

```
dbms_output.Put_line ('There are'||to_char(45-tot_emp)|| vacancies in department'||  
get_dep_id );
```

END IF;

END;

/

**OUTPUT:**

```
1  DECLARE  
2      tot_emp NUMBER;  
3      get_dep_id NUMBER;  
4  BEGIN  
5      get_dep_id := 80;  
6      SELECT Count(*)  
7      INTO tot_emp  
8      FROM employees e  
9      join department d  
10     ON e.dept_id = d.dept_id  
11  
12  dbms_output.Put_line ('The employees are in the department 80 is: 0'  
13  dbms_output.Put_line ('There are 45 vacancies in department 80'  
14  
15  Statement processed.  
16  
17  0.02 seconds
```

11.) Write a PL/SQL program to display the employee IDs, names, job titles, hire dates, and salaries of all employees

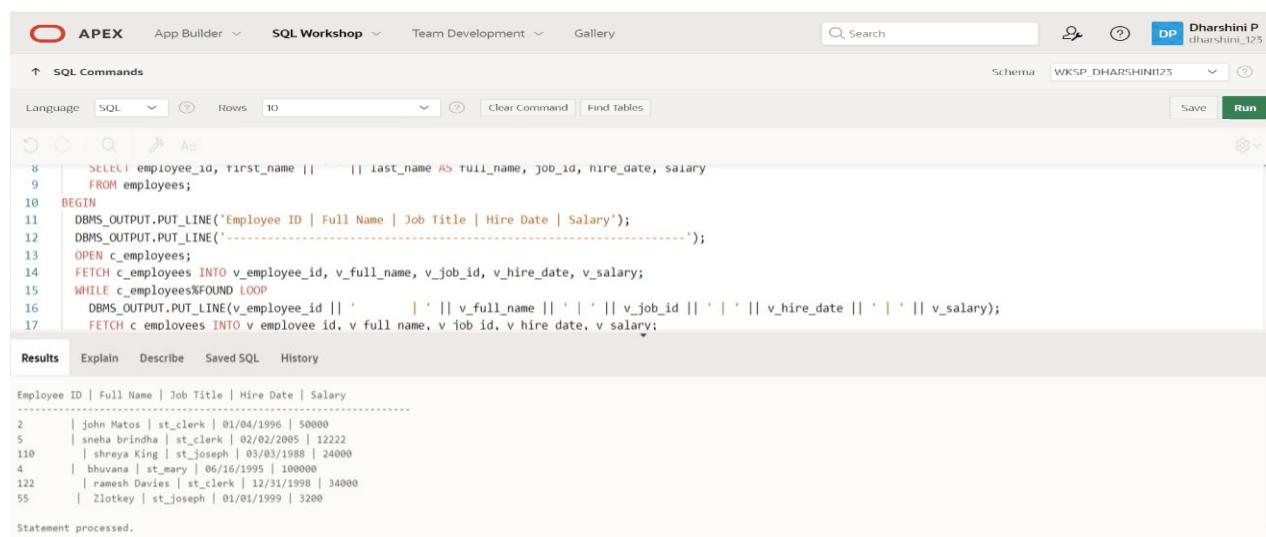
**QUERY:**

DECLARE

```
v_employee_id employees.employee_id%TYPE;
v_full_name employees.first_name%TYPE;
v_job_id employees.job_id%TYPE;
v_hire_date employees.hire_date%TYPE;
v_salary employees.salary%TYPE;

CURSOR c_employees IS
    SELECT employee_id, first_name || ' ' || last_name AS full_name, job_id, hire_date, salary
    FROM employees;
BEGIN
    DBMS_OUTPUT.PUT_LINE('Employee ID | Full Name | Job Title | Hire Date | Salary');
    DBMS_OUTPUT.PUT_LINE('-----');
    OPEN c_employees;
    FETCH c_employees INTO v_employee_id, v_full_name, v_job_id, v_hire_date, v_salary;
    WHILE c_employees%FOUND LOOP
        DBMS_OUTPUT.PUT_LINE(v_employee_id || ' ' || v_full_name || ' ' || v_job_id || ' '
        || v_hire_date || ' ' || v_salary);
        FETCH c_employees INTO v_employee_id, v_full_name, v_job_id, v_hire_date,
        v_salary;
    END LOOP;
    CLOSE c_employees;
END;
/
```

**OUTPUT:**



```
Employee ID | Full Name | Job Title | Hire Date | Salary
-----|-----|-----|-----|-----
2 | John Matos | st_clerk | 01/04/1996 | 50000
5 | sneha brindha | st_clerk | 02/02/2005 | 12222
110 | shreya King | st_joseph | 03/03/1988 | 24000
4 | bhuvana | st_mary | 06/16/1995 | 100000
122 | ramesh Davies | st_clerk | 12/31/1998 | 34000
55 | Zlotkey | st_joseph | 01/01/1999 | 3200
```

**12.)** Write a PL/SQL program to display the employee IDs, names, and department names of all employees.

## QUERY:

**DECLARE**

```
CURSOR emp_cursor IS
  SELECT e.employee_id, e.first_name, m.first_name AS manager_name
  FROM employees e
  LEFT JOIN employees m ON e.manager_id = m.employee_id;
emp_record emp_cursor%ROWTYPE;
BEGIN
  OPEN emp_cursor;
  FETCH emp_cursor INTO emp_record;
  WHILE emp_cursor%FOUND LOOP
    DBMS_OUTPUT.PUT_LINE('Employee ID: ' || emp_record.employee_id);
    DBMS_OUTPUT.PUT_LINE('Employee Name: ' || emp_record.first_name);
    DBMS_OUTPUT.PUT_LINE('Manager Name: ' || emp_record.manager_name);
    DBMS_OUTPUT.PUT_LINE('-----');
    FETCH emp_cursor INTO emp_record;
  END LOOP;
  CLOSE emp_cursor;
END;
/
```

## OUTPUT:

The screenshot shows the APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. A search bar is located at the top right. The user is logged in as Dharshini P (dharshini\_123). The current page is titled "SQL Commands". The main area displays a SQL code editor with the following query:

```
1 DECLARE
2   CURSOR emp_cursor IS
3     SELECT e.employee_id, e.first_name, m.first_name AS manager_name
4       FROM employees e
5      LEFT JOIN employees m ON e.manager_id = m.employee_id;
6   emp_record emp_cursor%ROWTYPE;
7
8   BEGIN
9     OPEN emp_cursor;
10    LOOP
11      FETCH emp_cursor INTO emp_record;
12      EXIT WHEN emp_cursor%NOTFOUND;
13      DBMS_OUTPUT.PUT_LINE('Employee ID: '||emp_record.employee_id);
14      DBMS_OUTPUT.PUT_LINE('Employee Name: '||emp_record.first_name);
15      DBMS_OUTPUT.PUT_LINE('Manager Name: '||emp_record.manager_name);
16      DBMS_OUTPUT.PUT_LINE('-----');
17    END LOOP;
18    CLOSE emp_cursor;
19  END;
```

The results tab is selected, showing the output of the query for four employees:

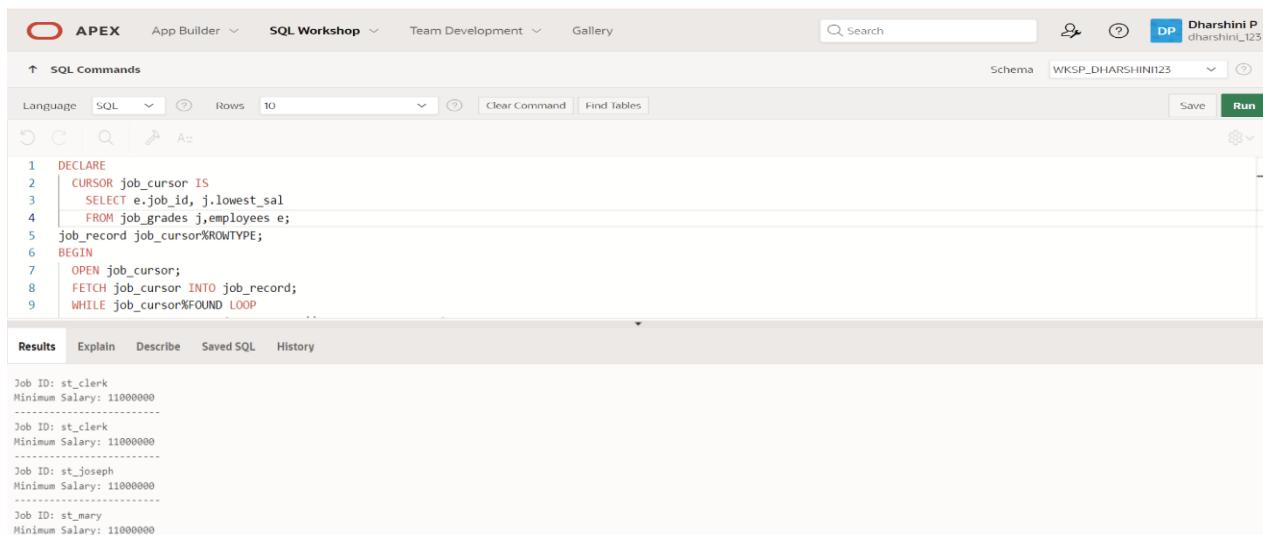
Employee ID	Employee Name	Manager Name
122	ramesh	
4		
2	john	
5	sneha	

13.) Write a PL/SQL program to display the job IDs, titles, and minimum salaries of all jobs

**QUERY:**

```
DECLARE
  CURSOR job_cursor IS
    SELECT e.job_id, j.lowest_sal
      FROM job_grade j,employees e;
  job_record job_cursor%ROWTYPE;
BEGIN
  OPEN job_cursor;
  FETCH job_cursor INTO job_record;
  WHILE job_cursor%FOUND LOOP
    DBMS_OUTPUT.PUT_LINE('Job ID: ' || job_record.job_id);
    DBMS_OUTPUT.PUT_LINE('Minimum Salary: ' || job_record.lowest_sal);
    DBMS_OUTPUT.PUT_LINE('-----');
    FETCH job_cursor INTO job_record;
  END LOOP;
  CLOSE job_cursor;
END;
/
```

**OUTPUT:**



```
APEX App Builder SQL Workshop Team Development Gallery Search Schema WKSP_DHARSHINI123 Run
SQL Commands Language SQL Rows 10 Clear Command Find Tables
1 DECLARE
2   CURSOR job_cursor IS
3     SELECT e.job_id, j.lowest_sal
4       FROM job_grades j,employees e;
5   job_record job_cursor%ROWTYPE;
6 BEGIN
7   OPEN job_cursor;
8   FETCH job_cursor INTO job_record;
9   WHILE job_cursor%FOUND LOOP
10
11   Job ID: st_clerk
12   Minimum Salary: 11000000
13   -----
14   Job ID: st_clerk
15   Minimum Salary: 11000000
16   -----
17   Job ID: st_joseph
18   Minimum Salary: 11000000
19   -----
20   Job ID: st_mary
21   Minimum Salary: 11000000
```

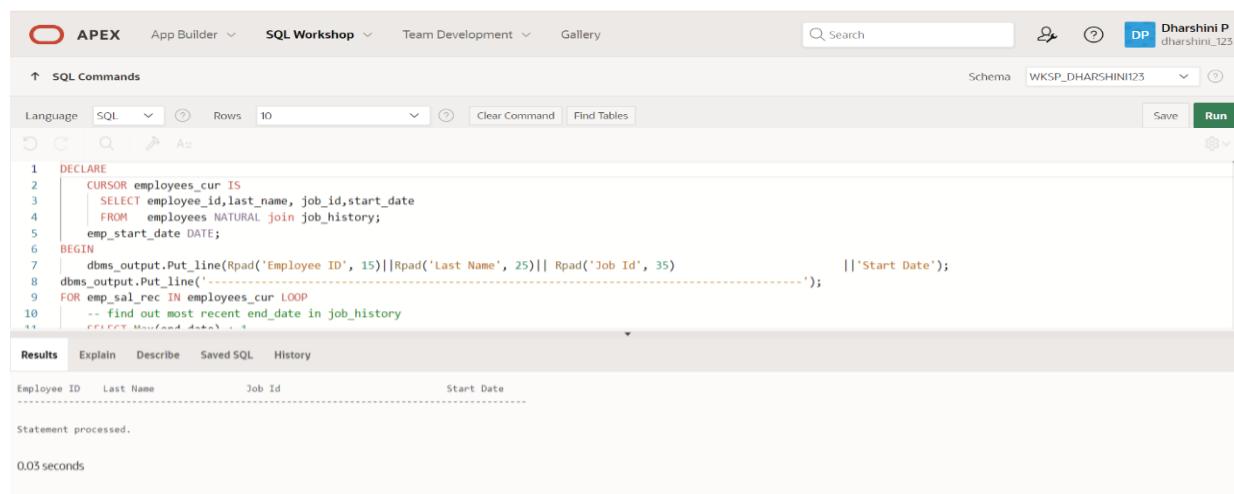
**14.)** Write a PL/SQL program to display the employee IDs, names, and job history start dates of all employees.

**QUERY:**

DECLARE

```
CURSOR employees_cur IS
  SELECT employee_id, last_name, job_id, start_date
  FROM employees NATURAL JOIN job_history;
  emp_start_date DATE;
BEGIN
  dbms_output.Put_line(Rpad('Employee ID', 15) || Rpad('Last Name', 25) || Rpad('Job Id', 35)
  || 'Start Date');
  dbms_output.Put_line('-----');
FOR emp_sal_rec IN employees_cur LOOP
  -- find out most recent end_date in job_history
  SELECT Max(end_date) + 1
  INTO emp_start_date
  FROM job_history
  WHERE employee_id = emp_sal_rec.employee_id;
  IF emp_start_date IS NULL THEN
    emp_start_date := emp_sal_rec.start_date;
  END IF;
  dbms_output.Put_line(Rpad(emp_sal_rec.employee_id, 15)
    || Rpad(emp_sal_rec.last_name, 25)
    || Rpad(emp_sal_rec.job_id, 35)
    || To_char(emp_start_date, 'dd-mon-yyyy'));
END LOOP;
END;
```

**/OUTPUT:**



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows the user profile 'Dharshini P dharshini\_123'. The main area displays the PL/SQL code in the 'SQL Commands' tab. The code is as follows:

```
1 DECLARE
2   CURSOR employees_cur IS
3     SELECT employee_id, last_name, job_id, start_date
4     FROM employees NATURAL JOIN job_history;
5   emp_start_date DATE;
6 BEGIN
7   dbms_output.Put_line(Rpad('Employee ID', 15) || Rpad('Last Name', 25) || Rpad('Job Id', 35)
8   || 'Start Date');
9   dbms_output.Put_line('-----');
10  FOR emp_sal_rec IN employees_cur LOOP
11    -- find out most recent end_date in job_history
12    SELECT Max(end_date) + 1
13    INTO emp_start_date
14    FROM job_history
15    WHERE employee_id = emp_sal_rec.employee_id;
16    IF emp_start_date IS NULL THEN
17      emp_start_date := emp_sal_rec.start_date;
18    END IF;
19    dbms_output.Put_line(Rpad(emp_sal_rec.employee_id, 15)
20      || Rpad(emp_sal_rec.last_name, 25)
21      || Rpad(emp_sal_rec.job_id, 35)
22      || To_char(emp_start_date, 'dd-mon-yyyy'));
23  END LOOP;
24END;
```

The results section shows the output of the query:

Employee ID	Last Name	Job Id	Start Date

Statement processed.  
0.03 seconds

**15.)** Write a PL/SQL program to display the employee IDs, names, and job history end dates of all employees.

**QUERY:**

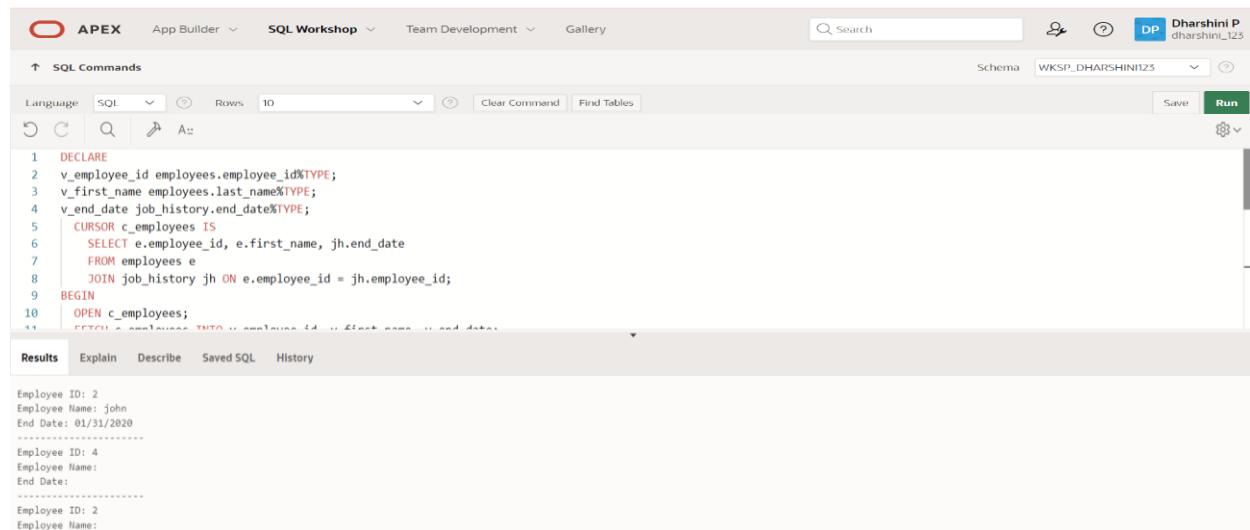
DECLARE

```
v_employee_id employees.employee_id%TYPE;
v_first_name employees.last_name%TYPE;
v_end_date job_history.end_date%TYPE;

CURSOR c_employees IS
  SELECT e.employee_id, e.first_name, jh.end_date
    FROM employees e
   JOIN job_history jh ON e.employee_id = jh.employee_id;

BEGIN
  OPEN c_employees;
  FETCH c_employees INTO v_employee_id, v_first_name, v_end_date;
  WHILE c_employees%FOUND LOOP
    DBMS_OUTPUT.PUT_LINE('Employee ID: ' || v_employee_id);
    DBMS_OUTPUT.PUT_LINE('Employee Name: ' || v_first_name);
    DBMS_OUTPUT.PUT_LINE('End Date: ' || v_end_date);
    DBMS_OUTPUT.PUT_LINE('-----');
    FETCH c_employees INTO v_employee_id, v_first_name, v_end_date;
  END LOOP;
  CLOSE c_employees;
END;
```

**OUTPUT:**



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. A search bar and a user profile for 'Dharshini P dharshini\_123' are also present. The main workspace is titled 'SQL Commands' and contains the following PL/SQL code:

```
1 DECLARE
2   v_employee_id employees.employee_id%TYPE;
3   v_first_name employees.last_name%TYPE;
4   v_end_date job_history.end_date%TYPE;
5   CURSOR c_employees IS
6     SELECT e.employee_id, e.first_name, jh.end_date
7       FROM employees e
8      JOIN job_history jh ON e.employee_id = jh.employee_id;
9 BEGIN
10  OPEN c_employees;
11  FETCH c_employees INTO v_employee_id, v_first_name, v_end_date;
```

The 'Results' tab is selected, displaying the output of the executed code:

```
Employee ID: 2
Employee Name: john
End Date: 01/31/2020
-----
Employee ID: 4
Employee Name:
End Date:
-----
Employee ID: 2
Employee Name:
```

Evaluation Procedure	Marks awarded
Query(5)	
Execution (5)	
Viva(5)	
Total (15)	
Faculty Signature	

**RESULT:**

## PROCEDURES AND FUNCTIONS

EX.NO: 17

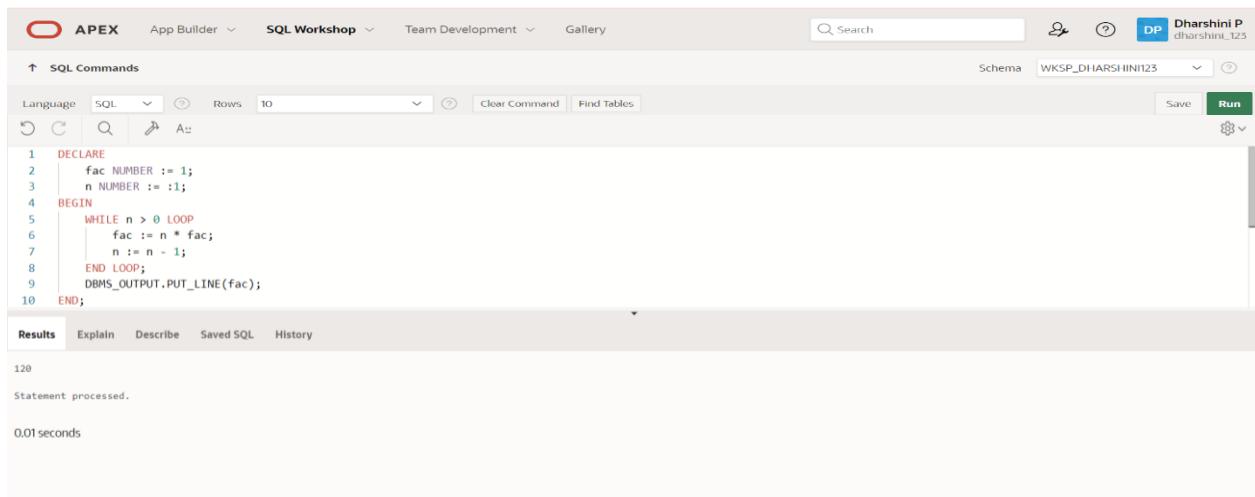
DATE:

### 1.) Factorial of a number using function.

**QUERY:**

```
DECLARE
    fac NUMBER := 1;
    n NUMBER := :1;
BEGIN
    WHILE n > 0 LOOP
        fac := n * fac;
        n := n - 1;
    END LOOP;
    DBMS_OUTPUT.PUT_LINE(fac);
END;
```

**OUTPUT:**



The screenshot shows the Oracle SQL Workshop interface. In the SQL Commands tab, a PL/SQL block is written and executed. The code declares variables fac and n, initializes them, and then enters a loop to calculate the factorial of n. The output is displayed in the Results tab, showing the processed statement and the execution time.

```
1  DECLARE
2      fac NUMBER := 1;
3      n NUMBER := :1;
4  BEGIN
5      WHILE n > 0 LOOP
6          fac := n * fac;
7          n := n - 1;
8      END LOOP;
9      DBMS_OUTPUT.PUT_LINE(fac);
10 END;
```

Results

Statement processed.  
0.01 seconds

**2.) Write a PL/SQL program using Procedures IN,INOUT,OUT parameters to retrieve the corresponding book information in library.**

**QUERY:**

```
CREATE OR REPLACE PROCEDURE get_book_info (
    p_book_id IN NUMBER,
    p_title IN OUT VARCHAR2,
    p_author OUT VARCHAR2,
    p_year_published OUT NUMBER
)
AS
BEGIN
    SELECT title, author, year_published INTO p_title, p_author, p_year_published
    FROM books
    WHERE book_id = p_book_id;

    p_title := p_title || ' - Retrieved';
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        p_title := NULL;
        p_author := NULL;
        p_year_published := NULL;
END;
```

```
DECLARE
    v_book_id NUMBER := 1;
    v_title VARCHAR2(100);
    v_author VARCHAR2(100);
    v_year_published NUMBER;
BEGIN
    v_title := 'Initial Title';

    get_book_info(p_book_id => v_book_id, p_title => v_title, p_author => v_author,
    p_year_published => v_year_published);

    DBMS_OUTPUT.PUT_LINE('Title: ' || v_title);
    DBMS_OUTPUT.PUT_LINE('Author: ' || v_author);
    DBMS_OUTPUT.PUT_LINE('Year Published: ' || v_year_published);
END;
```

## OUTPUT:

The screenshot shows the Oracle APEX interface with the SQL Workshop tab selected. The schema is set to 'WKSP\_DHARSHINI123'. The code area contains the following SQL statements:

```
1 CREATE TABLE books (
2     book_id NUMBER PRIMARY KEY,
3     title VARCHAR2(100),
4     author VARCHAR2(100),
5     year_published NUMBER
6 );
7 INSERT INTO books (book_id, title, author, year_published) VALUES (1, '1984', 'George Orwell', 1949);
8 INSERT INTO books (book_id, title, author, year_published) VALUES (2, 'To Kill a Mockingbird', 'Harper Lee', 1960);
9 INSERT INTO books (book_id, title, author, year_published) VALUES (3, 'The Great Gatsby', 'F. Scott Fitzgerald', 1925);
10
11 CREATE OR REPLACE PROCEDURE get_book_info (
12     p_book_id IN NUMBER,
13     p_title OUT VARCHAR2,
14     p_author OUT VARCHAR2;
```

The results section shows the output of the last query:

```
Title: 1984
Author: George Orwell
Year Published: 1949
Statement processed.
```

Evaluation Procedure	Marks awarded
Query(5)	
Execution (5)	
Viva(5)	
Total (15)	
Faculty Signature	

## RESULT:

# TRIGGER

EX\_NO: 18

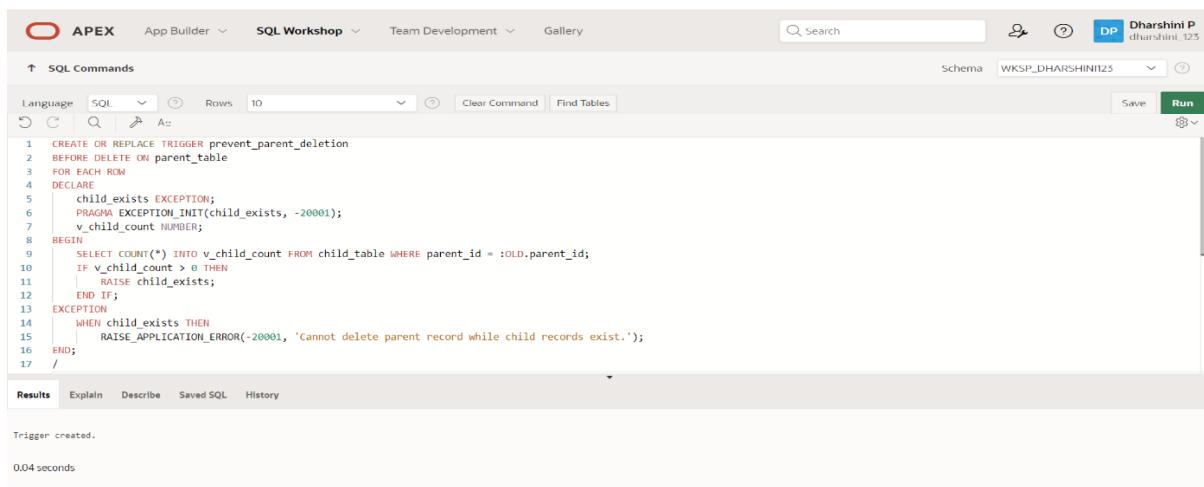
DATE:

1.) Write a code in PL/SQL to develop a trigger that enforces referential integrity by preventing the deletion of a parent record if child records exist

QUERY:

```
CREATE OR REPLACE TRIGGER prevent_parent_deletion
BEFORE DELETE ON parent_table
FOR EACH ROW
DECLARE
    child_exists EXCEPTION;
    PRAGMA EXCEPTION_INIT(child_exists, -20001);
    v_child_count NUMBER;
BEGIN
    SELECT COUNT(*) INTO v_child_count FROM child_table WHERE parent_id =
:OLD.parent_id;
    IF v_child_count > 0 THEN
        RAISE child_exists;
    END IF;
EXCEPTION
    WHEN child_exists THEN
        RAISE_APPLICATION_ERROR(-20001, 'Cannot delete parent record while
child records exist.');
END;
```

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. A search bar is at the top right. The main area is titled "SQL Commands". The schema dropdown shows "WKSP\_DHARSHINIT25". The code editor contains the PL/SQL trigger definition provided above. The "Run" button is visible on the right. The results tab at the bottom shows the message "Trigger created." and a execution time of "0.04 seconds".

```
1 CREATE OR REPLACE TRIGGER prevent_parent_deletion
2 BEFORE DELETE ON parent_table
3 FOR EACH ROW
4 DECLARE
5     child_exists EXCEPTION;
6     PRAGMA EXCEPTION_INIT(child_exists, -20001);
7     v_child_count NUMBER;
8 BEGIN
9     SELECT COUNT(*) INTO v_child_count FROM child_table WHERE parent_id =
:OLD.parent_id;
10    IF v_child_count > 0 THEN
11        RAISE child_exists;
12    END IF;
13 EXCEPTION
14    WHEN child_exists THEN
15        RAISE_APPLICATION_ERROR(-20001, 'Cannot delete parent record while child records exist.');
16 END;
17 /
```

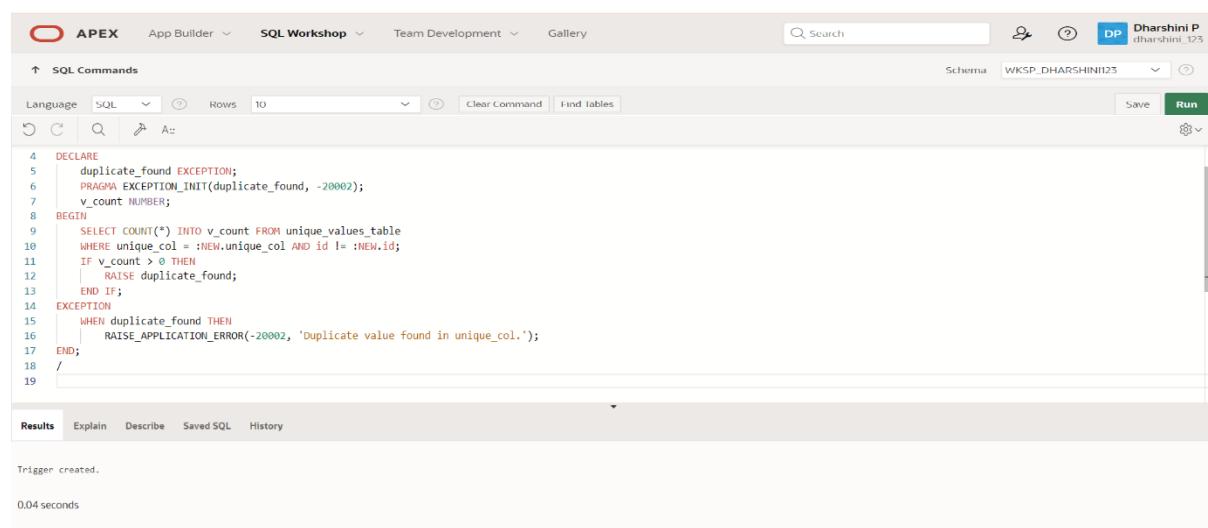
Trigger created.  
0.04 seconds

**2.) Write a code in PL/SQL to create a trigger that checks for duplicate values in a specific column and raises an exception if found**

**QUERY:**

```
CREATE OR REPLACE TRIGGER check_duplicates
BEFORE INSERT OR UPDATE ON unique_values_table
FOR EACH ROW
DECLARE
    duplicate_found EXCEPTION;
    PRAGMA EXCEPTION_INIT(duplicate_found, -20002);
    v_count NUMBER;
BEGIN
    SELECT COUNT(*) INTO v_count FROM unique_values_table
    WHERE unique_col = :NEW.unique_col AND id != :NEW.id;
    IF v_count > 0 THEN
        RAISE duplicate_found;
    END IF;
EXCEPTION
    WHEN duplicate_found THEN
        RAISE_APPLICATION_ERROR(-20002, 'Duplicate value found in
unique_col.');
END;
```

**OUTPUT:**



The screenshot shows the Oracle SQL Workshop interface. In the top navigation bar, 'APEX' is selected. The main area displays the PL/SQL code for the trigger. The code is as follows:

```
4  DECLARE
5      duplicate_found EXCEPTION;
6      PRAGMA EXCEPTION_INIT(duplicate_found, -20002);
7      v_count NUMBER;
8  BEGIN
9      SELECT COUNT(*) INTO v_count FROM unique_values_table
10     WHERE unique_col = :NEW.unique_col AND id != :NEW.id;
11     IF v_count > 0 THEN
12         RAISE duplicate_found;
13     END IF;
14 EXCEPTION
15     WHEN duplicate_found THEN
16         RAISE_APPLICATION_ERROR(-20002, 'Duplicate value found in unique_col.');
17 END;
18 /
19
```

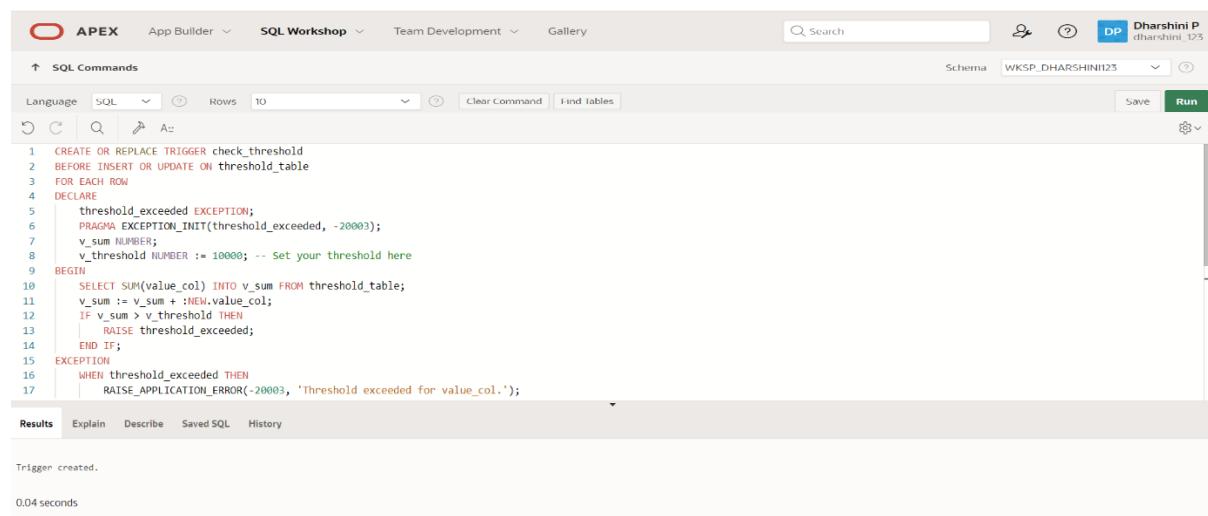
Below the code, the 'Results' tab is active, showing the message "Trigger created." and a execution time of "0.04 seconds".

**3.) Write a code in PL/SQL to create a trigger that restricts the insertion of new rows if the total of a column's values exceeds a certain threshold**

**QUERY:**

```
CREATE OR REPLACE TRIGGER check_threshold
BEFORE INSERT OR UPDATE ON threshold_table
FOR EACH ROW
DECLARE
    threshold_exceeded EXCEPTION;
    PRAGMA EXCEPTION_INIT(threshold_exceeded, -20003);
    v_sum NUMBER;
    v_threshold NUMBER := 10000; -- Set your threshold here
BEGIN
    SELECT SUM(value_col) INTO v_sum FROM threshold_table;
    v_sum := v_sum + :NEW.value_col;
    IF v_sum > v_threshold THEN
        RAISE threshold_exceeded;
    END IF;
EXCEPTION
    WHEN threshold_exceeded THEN
        RAISE_APPLICATION_ERROR(-20003, 'Threshold exceeded for value_col.');
END;
```

**OUTPUT:**



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows a user profile for 'Dharsini P' (dharsini\_123). The main area displays the PL/SQL code for the 'check\_threshold' trigger. The code is identical to the one provided in the question, including the exception handling and the RAISE\_APPLICATION\_ERROR call. Below the code, the results pane shows the message 'Trigger created.' and a execution time of '0.04 seconds'.

```
1 CREATE OR REPLACE TRIGGER check_threshold
2 BEFORE INSERT OR UPDATE ON threshold_table
3 FOR EACH ROW
4 DECLARE
5     threshold_exceeded EXCEPTION;
6     PRAGMA EXCEPTION_INIT(threshold_exceeded, -20003);
7     v_sum NUMBER;
8     v_threshold NUMBER := 10000; -- Set your threshold here
9 BEGIN
10    SELECT SUM(value_col) INTO v_sum FROM threshold_table;
11    v_sum := v_sum + :NEW.value_col;
12    IF v_sum > v_threshold THEN
13        RAISE threshold_exceeded;
14    END IF;
15 EXCEPTION
16    WHEN threshold_exceeded THEN
17        RAISE_APPLICATION_ERROR(-20003, 'Threshold exceeded for value_col.');

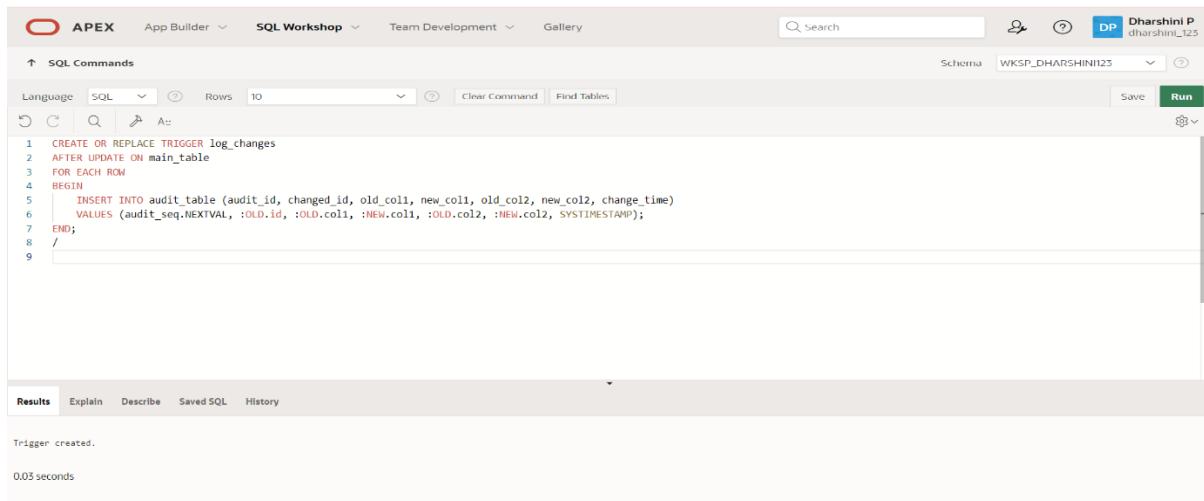
```

**4.) Write a code in PL/SQL to design a trigger that captures changes made to specific columns and logs them in an audit table.**

**QUERY:**

```
CREATE OR REPLACE TRIGGER log_changes
AFTER UPDATE ON main_table
FOR EACH ROW
BEGIN
    INSERT INTO audit_table (audit_id, changed_id, old_col1, new_col1,
old_col2, new_col2, change_time)
    VALUES (audit_seq.NEXTVAL, :OLD.id, :OLD.col1, :NEW.col1, :OLD.col2,
:NEW.col2, SYSTIMESTAMP);
END;
```

**OUTPUT:**



The screenshot shows the Oracle SQL Workshop interface. In the top navigation bar, 'APEX' is selected. The main area is titled 'SQL Commands'. The schema dropdown shows 'WKSP\_DHARSHINI125'. The code editor contains the PL/SQL trigger creation script. The results tab at the bottom displays the message 'Trigger created.' and a execution time of '0.03 seconds'.

```
1 CREATE OR REPLACE TRIGGER log_changes
2 AFTER UPDATE ON main_table
3 FOR EACH ROW
4 BEGIN
5     INSERT INTO audit_table (audit_id, changed_id, old_col1, new_col1, old_col2, new_col2, change_time)
6     VALUES (audit_seq.NEXTVAL, :OLD.id, :OLD.col1, :NEW.col1, :OLD.col2, :NEW.col2, SYSTIMESTAMP);
7 END;
8 /
```

Results Explain Describe Saved SQL History

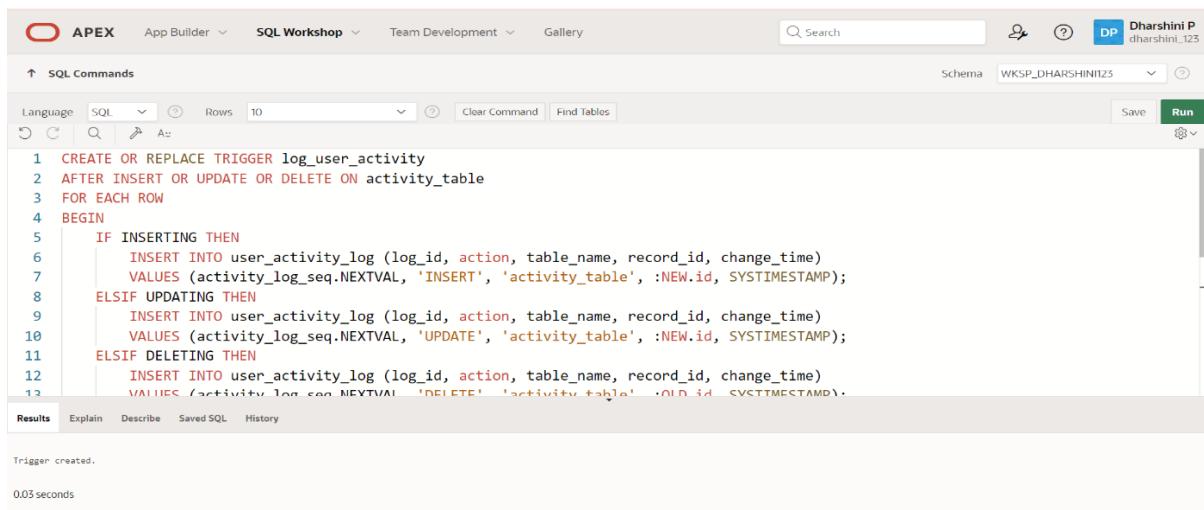
Trigger created.  
0.03 seconds

**5.) Write a code in PL/SQL to implement a trigger that records user activity (inserts, updates, deletes) in an audit log for a given set of tables.**

**QUERY:**

```
CREATE OR REPLACE TRIGGER log_user_activity
AFTER INSERT OR UPDATE OR DELETE ON activity_table
FOR EACH ROW
BEGIN
    IF INSERTING THEN
        INSERT INTO user_activity_log (log_id, action, table_name, record_id,
change_time)
        VALUES (activity_log_seq.NEXTVAL, 'INSERT', 'activity_table', :NEW.id,
SYSTIMESTAMP);
    ELSIF UPDATING THEN
        INSERT INTO user_activity_log (log_id, action, table_name, record_id,
change_time)
        VALUES (activity_log_seq.NEXTVAL, 'UPDATE', 'activity_table', :NEW.id,
SYSTIMESTAMP);
    ELSIF DELETING THEN
        INSERT INTO user_activity_log (log_id, action, table_name, record_id,
change_time)
        VALUES (activity_log_seq.NEXTVAL, 'DELETE', 'activity_table', :OLD.id,
SYSTIMESTAMP);
    END IF;
END;
```

**OUTPUT:**



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, and Gallery. The SQL Workshop tab is selected. The schema dropdown shows 'WKSP\_DHARSHINI123'. The main area displays the PL/SQL code for the trigger, which is highlighted in red. The code is identical to the one provided in the question. Below the code, the 'Results' tab is active, showing the message 'Trigger created.' and a execution time of '0.03 seconds'.

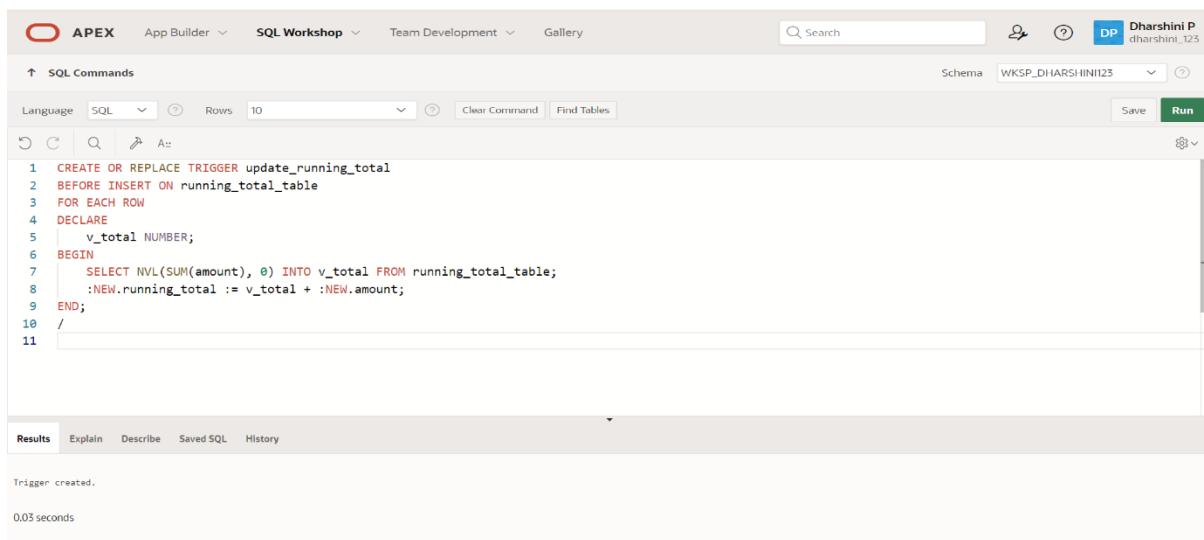
```
1 CREATE OR REPLACE TRIGGER log_user_activity
2 AFTER INSERT OR UPDATE OR DELETE ON activity_table
3 FOR EACH ROW
4 BEGIN
5     IF INSERTING THEN
6         INSERT INTO user_activity_log (log_id, action, table_name, record_id, change_time)
7         VALUES (activity_log_seq.NEXTVAL, 'INSERT', 'activity_table', :NEW.id, SYSTIMESTAMP);
8     ELSIF UPDATING THEN
9         INSERT INTO user_activity_log (log_id, action, table_name, record_id, change_time)
10        VALUES (activity_log_seq.NEXTVAL, 'UPDATE', 'activity_table', :NEW.id, SYSTIMESTAMP);
11    ELSIF DELETING THEN
12        INSERT INTO user_activity_log (log_id, action, table_name, record_id, change_time)
13        VALUES (activity_log_seq.NEXTVAL, 'DELETE', 'activity_table', :OLD.id, SYSTIMESTAMP);
```

**6.) Write a code in PL/SQL to implement a trigger that automatically calculates and updates a running total column for a table whenever new rows are inserted**

**QUERY:**

```
CREATE OR REPLACE TRIGGER update_running_total
BEFORE INSERT ON running_total_table
FOR EACH ROW
DECLARE
    v_total NUMBER;
BEGIN
    SELECT NVL(SUM(amount), 0) INTO v_total FROM running_total_table;
    :NEW.running_total := v_total + :NEW.amount;
END;
```

**OUTPUT:**



The screenshot shows the Oracle SQL Workshop interface. In the top navigation bar, 'APEX' is selected. The main area displays the SQL command for creating the trigger:

```
1 CREATE OR REPLACE TRIGGER update_running_total
2 BEFORE INSERT ON running_total_table
3 FOR EACH ROW
4 DECLARE
5     v_total NUMBER;
6 BEGIN
7     SELECT NVL(SUM(amount), 0) INTO v_total FROM running_total_table;
8     :NEW.running_total := v_total + :NEW.amount;
9 END;
10 /
11
```

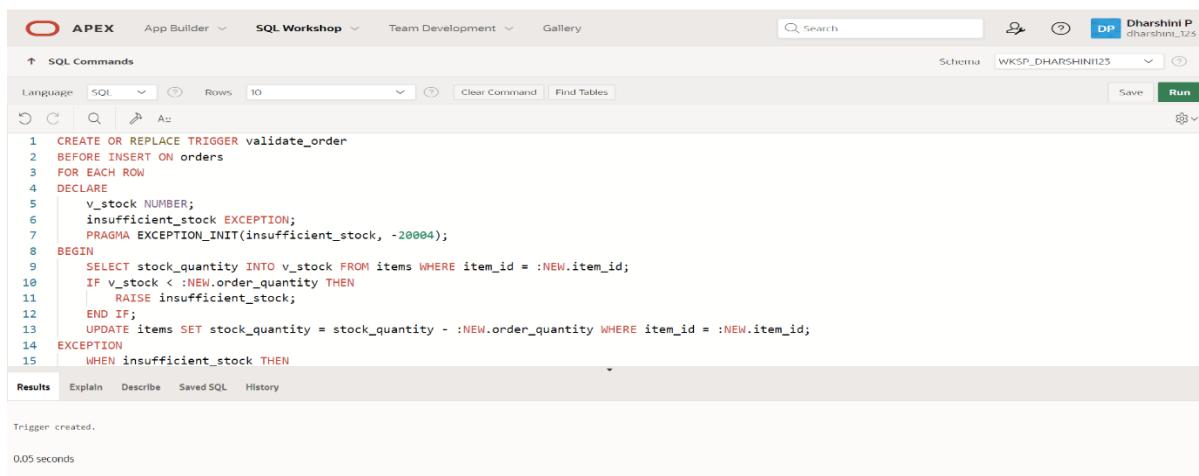
Below the code, the 'Results' tab is active, showing the output: "Trigger created." and "0.03 seconds".

**7.) Write a code in PL/SQL to create a trigger that validates the availability of items before allowing an order to be placed, considering stock levels and pending orders**

**QUERY:**

```
CREATE OR REPLACE TRIGGER validate_order
BEFORE INSERT ON orders
FOR EACH ROW
DECLARE
    v_stock NUMBER;
    insufficient_stock EXCEPTION;
    PRAGMA EXCEPTION_INIT(insufficient_stock, -20004);
BEGIN
    SELECT stock_quantity INTO v_stock FROM items WHERE item_id =
    :NEW.item_id;
    IF v_stock < :NEW.order_quantity THEN
        RAISE insufficient_stock;
    END IF;
    UPDATE items SET stock_quantity = stock_quantity - :NEW.order_quantity
    WHERE item_id = :NEW.item_id;
EXCEPTION
    WHEN insufficient_stock THEN
        RAISE_APPLICATION_ERROR(-20004, 'Insufficient stock for the item.');
END;
```

**OUTPUT:**

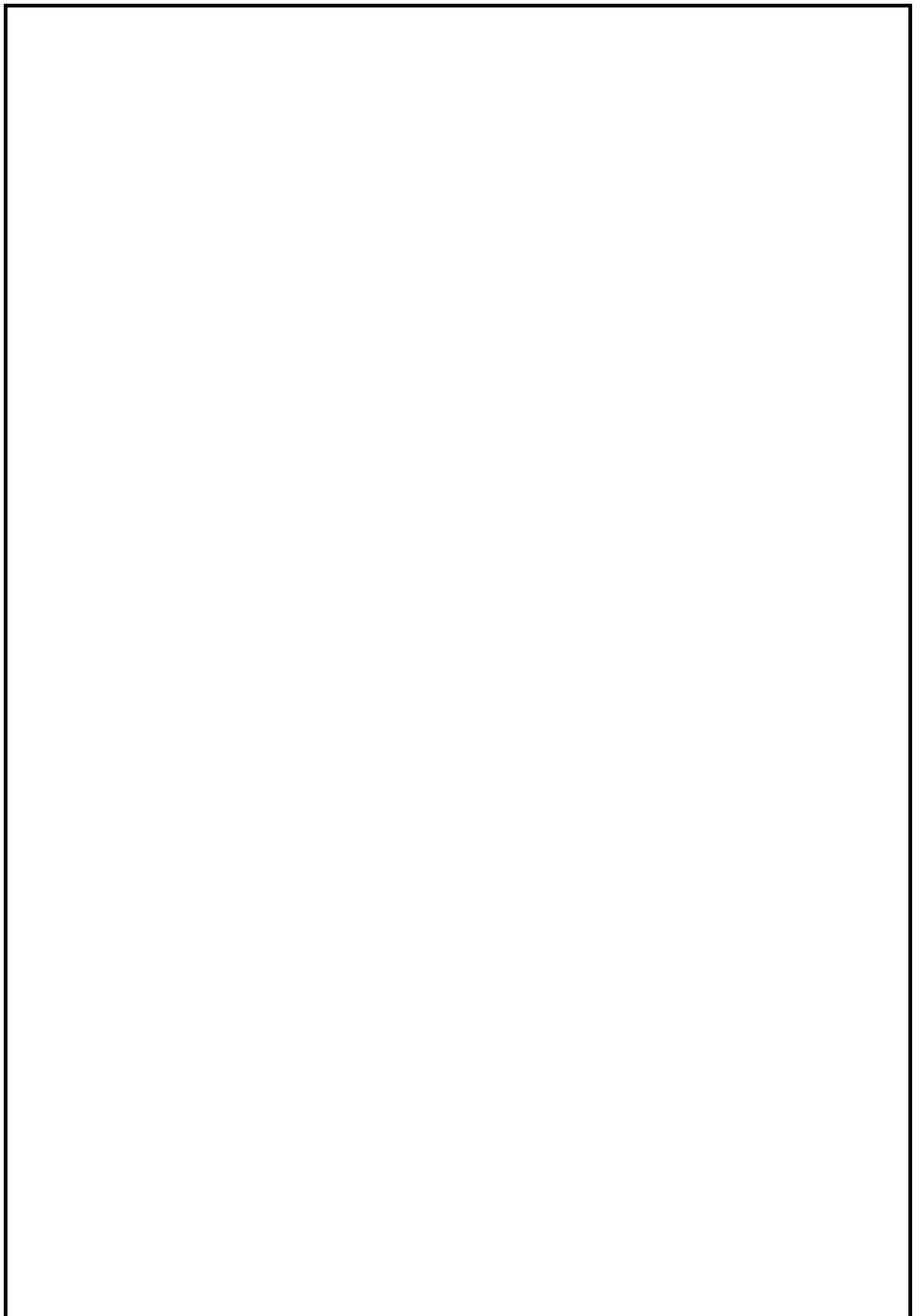


The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows the user profile 'Dharsini P' and the schema 'WKSP\_DHARSHINI123'. The main area displays the PL/SQL code for the 'validate\_order' trigger. The code is identical to the one provided in the question, including the declaration of variables, the BEGIN block with a SELECT statement and IF condition, the UPDATE statement, and the EXCEPTION block with a RAISE\_APPLICATION\_ERROR. Below the code, the 'Results' tab is active, showing the message 'Trigger created.' and a execution time of '0.05 seconds'.

```
1 CREATE OR REPLACE TRIGGER validate_order
2 BEFORE INSERT ON orders
3 FOR EACH ROW
4 DECLARE
5     v_stock NUMBER;
6     insufficient_stock EXCEPTION;
7     PRAGMA EXCEPTION_INIT(insufficient_stock, -20004);
8 BEGIN
9     SELECT stock_quantity INTO v_stock FROM items WHERE item_id =
10    :NEW.item_id;
11    IF v_stock < :NEW.order_quantity THEN
12        RAISE insufficient_stock;
13    END IF;
14    UPDATE items SET stock_quantity = stock_quantity - :NEW.order_quantity WHERE item_id = :NEW.item_id;
15 EXCEPTION
16     WHEN insufficient_stock THEN
17         RAISE_APPLICATION_ERROR(-20004, 'Insufficient stock for the item.');
END;
```

Evaluation Procedure	Marks awarded
Query(5)	
Execution (5)	
Viva(5)	
Total (15)	
Faculty Signature	

**RESULT:**



# MONGO DB

**EX\_NO: 19**

**DATE:**

1.) Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which prepared dish except 'American' and 'Chinees' or restaurant's name begins with letter 'Wil'.

**QUERY:**

```
db.restaurants.find( { $or: [ { name: /^Wil/ }, { cuisine: { $nin: ['American', 'Chinese'] } } ] , { restaurant_id: 1, name: 1, borough: 1, cuisine: 1 } );
```

**OUTPUT:**

```
DHARSHINI_65> db.restaurants.find( { $or: [ { name: /^Wil/ }, { cuisine: { $nin: ['American', 'Chinese'] } } ] , { restaurant_id: 1, name: 1, borough: 1, cuisine: 1 } );
[
  {
    _id: ObjectId('664f3c798752f54dc3cdcdf7'),
    borough: 'Bronx',
    cuisine: 'Bakery',
    name: 'Morris Park Bake Shop',
    restaurant_id: '30075445'
  }
]
DHARSHINI_65>|
```

2.) Write a MongoDB query to find the restaurant Id, name, and grades for those restaurants which achieved a grade of "A" and scored 11 on an ISODate "2014-08-11T00:00:00Z" among many of survey dates.

**QUERY:** db.restaurants.find( { grades: { \$elemMatch: { grade: "A", score: 11, date: ISODate("2014-08-11T00:00:00Z") } } }, { restaurant\_id: 1, name: 1, grades: 1 } );

**OUTPUT:**

```
DHARSHINI_65> db.restaurants.find( { grades: { $elemMatch: { grade: "A", score: 11, date: ISODate("2014-08-11T00:00:00Z") } } }, { restaurant_id: 1, name: 1, grades: 1 } );
DHARSHINI_65>|
```

3.)Write a MongoDB query to find the restaurant Id, name and grades for those restaurants where the 2nd element of grades array contains a grade of "A" and score 9 on an ISODate "2014-08-11T00:00:00Z".

#### QUERY:

```
db.restaurants.find( {"grades.1.grade": "A", "grades.1.score": 9, "grades.1.date": ISODate("2014-08-11T00:00:00Z") }, { restaurant_id: 1, name: 1, grades: 1 } );
```

#### OUTPUT:

```
DHARSHINI_65> db.restaurants.find( { "grades.1.grade": "A", "grades.1.score": 9, "grades.1.date": ISODate("2014-08-11T00:00:00Z") }, { restaurant_id: 1, name: 1, grades: 1 } );
DHARSHINI_65> |
```

4.)Write a MongoDB query to find the restaurant Id, name, address and geographical location for those restaurants where 2nd element of coord array contains a value which is more than 42 and upto 52

#### QUERY:

```
db.restaurants.find({$and : [{"address.coord.1": {$gt : 42}}, {"address.coord.1": {$lte : 52}}]}, { _id:0, restaurant_id:1, name:1, address:1})
```

#### OUTPUT:

```
DHARSHINI_65> db.restaurants.find({$and : [{"address.coord.1": {$gt : 42}}, {"address.coord.1": {$lte : 52}}]}, { _id:0, restaurant_id:1, name:1, address:1 })
DHARSHINI_65> |
```

5.) Write a MongoDB query to arrange the name of the restaurants in ascending order along with all the columns.

#### QUERY:

```
db.restaurants.find({}, { _id: 0 }).sort({ name: 1 });
```

#### OUTPUT:

```
DHARSHINI_65> db.restaurants.find({}, { _id: 0 }).sort({ name: 1 })
[
  {
    address: {
      building: '1007',
      coord: [ -73.856077, 40.848447 ],
      street: 'Morris Park Ave',
      zipcode: '10462'
    },
    borough: 'Bronx',
    cuisine: 'Bakery',
    grades: [
      {
        date: ISODate('2014-03-03T00:00:00.000Z'),
        grade: 'A',
        score: 2
      },
      {
        date: ISODate('2013-09-11T00:00:00.000Z'),
        grade: 'A',
        score: 6
      },
      {
        date: ISODate('2013-01-24T00:00:00.000Z'),
        grade: 'A',
        score: 10
      },
      {
        date: ISODate('2011-11-23T00:00:00.000Z'),
        grade: 'A',
        score: 9
      },
      {
        date: ISODate('2011-03-10T00:00:00.000Z'),
        grade: 'B',
        score: 14
      }
    ],
    name: 'Morris Park Bake Shop',
    restaurant_id: '30075445'
  }
]
DHARSHINI_65>|
```

6.) Write a MongoDB query to arrange the name of the restaurants in descending along with all the columns.

#### QUERY:

```
db.restaurants.find({}, { _id: 0 }).sort({ name: -1 })
```

#### OUTPUT:

```
DHARSHINI_65> db.restaurants.find({}, { _id: 0 }).sort({ name: -1 })
[
  {
    address: {
      building: '1007',
      coord: [ -73.856077, 40.848447 ],
      street: 'Morris Park Ave',
      zipcode: '10462'
    },
    borough: 'Bronx',
    cuisine: 'Bakery',
    grades: [
      {
        date: ISODate('2014-03-03T00:00:00.000Z'),
        grade: 'A',
        score: 2
      },
      {
        date: ISODate('2013-09-11T00:00:00.000Z'),
        grade: 'A',
        score: 6
      },
      {
        date: ISODate('2013-01-24T00:00:00.000Z'),
        grade: 'A',
        score: 10
      },
      {
        date: ISODate('2011-11-23T00:00:00.000Z'),
        grade: 'A',
        score: 9
      },
      {
        date: ISODate('2011-03-10T00:00:00.000Z'),
        grade: 'B',
        score: 14
      }
    ],
    name: 'Morris Park Bake Shop',
    restaurant_id: '30075445'
  }
]
DHARSHINI_65>|
```

7.) Write a MongoDB query to arranged the name of the cuisine in ascending order and for that same cuisine borough should be in descending order.

#### QUERY:

```
db.restaurants.find({}, { _id: 0 }).sort({ cuisine: 1, borough: -1 })
```

#### OUTPUT:

```
DHARSHINI_65> db.restaurants.find({}, { _id: 0 }).sort({ cuisine: 1, borough: -1 })
[
  {
    address: {
      building: '1007',
      coord: [ -73.856077, 40.848447 ],
      street: 'Morris Park Ave',
      zipcode: '10462'
    },
    borough: 'Bronx',
    cuisine: 'Bakery',
    grades: [
      {
        date: ISODate('2014-03-03T00:00:00.000Z'),
        grade: 'A',
        score: 2
      },
      {
        date: ISODate('2013-09-11T00:00:00.000Z'),
        grade: 'A',
        score: 6
      },
      {
        date: ISODate('2013-01-24T00:00:00.000Z'),
        grade: 'A',
        score: 10
      },
      {
        date: ISODate('2011-11-23T00:00:00.000Z'),
        grade: 'A',
        score: 9
      },
      {
        date: ISODate('2011-03-10T00:00:00.000Z'),
        grade: 'B',
        score: 14
      }
    ],
    name: 'Morris Park Bake Shop',
    restaurant_id: '30075445'
  }
]
DHARSHINI_65>|
```

8.) Write a MongoDB query to know whether all the addresses contains the street or not.

#### QUERY:

```
db.restaurants.find({ "address.street": { $exists: true, $ne: "" } })
```

#### OUTPUT:

```
DHARSHINI_65> db.restaurants.find({ "address.street": { $exists: true, $ne: "" } })
[
  {
    _id: ObjectId('664f3c798752f54dc3cdcdf7'),
    address: {
      building: '1007',
      coord: [ -73.856077, 40.848447 ],
      street: 'Morris Park Ave',
      zipcode: '10462'
    },
    borough: 'Bronx',
    cuisine: 'Bakery',
    grades: [
      {
        date: ISODate('2014-03-03T00:00:00.000Z'),
        grade: 'A',
        score: 2
      },
      {
        date: ISODate('2013-09-11T00:00:00.000Z'),
        grade: 'A',
        score: 6
      },
      {
        date: ISODate('2013-01-24T00:00:00.000Z'),
        grade: 'A',
        score: 10
      },
      {
        date: ISODate('2011-11-23T00:00:00.000Z'),
        grade: 'A',
        score: 9
      },
      {
        date: ISODate('2011-03-10T00:00:00.000Z'),
        grade: 'B',
        score: 14
      }
    ],
    name: 'Morris Park Bake Shop',
    restaurant_id: '30075445'
  }
]
DHARSHINI_65>|
```

**9.)** Write a MongoDB query which will select all documents in the restaurants collection where the coord field value is Double.

#### QUERY:

```
db.restaurants.find({ "address.coord": { $elemMatch: { $type: "double" } } })
```

#### OUTPUT:

```
DHARSHINI_65> db.restaurants.find({ "address.coord": { $elemMatch: { $type: "double" } } })
[ {
  _id: ObjectId('664f3c798752f54dc3cdcdf7'),
  address: {
    building: '1007',
    coord: [-73.856077, 40.848447],
    street: 'Morris Park Ave',
    zipcode: '10462'
  },
  borough: 'Bronx',
  cuisine: 'Bakery',
  grades: [
    {
      date: ISODate('2014-03-03T00:00:00.000Z'),
      grade: 'A',
      score: 2
    },
    {
      date: ISODate('2013-09-11T00:00:00.000Z'),
      grade: 'A',
      score: 6
    },
    {
      date: ISODate('2013-01-24T00:00:00.000Z'),
      grade: 'A',
      score: 10
    },
    {
      date: ISODate('2011-11-23T00:00:00.000Z'),
      grade: 'A',
      score: 9
    },
    {
      date: ISODate('2011-03-10T00:00:00.000Z'),
      grade: 'B',
      score: 14
    }
  ],
  name: 'Morris Park Bake Shop',
  restaurant_id: '30075445'
}
]
DHARSHINI_65>
```

**10.** Write a MongoDB query which will select the restaurant Id, name and grades for those restaurants which returns 0 as a remainder after dividing the score by 7.

#### QUERY:

```
db.restaurants.find({ "grades.score": { $mod: [7, 0] } }, { restaurant_id: 1, name: 1, grades: 1 });
```

#### OUTPUT:

```
DHARSHINI_65> db.restaurants.find({ "grades.score": { $mod: [7, 0] } }, { restaurant_id: 1, name: 1, grades: 1 })
[ {
  _id: ObjectId('664f3c798752f54dc3cdcdf7'),
  grades: [
    {
      date: ISODate('2014-03-03T00:00:00.000Z'),
      grade: 'A',
      score: 2
    },
    {
      date: ISODate('2013-09-11T00:00:00.000Z'),
      grade: 'A',
      score: 6
    },
    {
      date: ISODate('2013-01-24T00:00:00.000Z'),
      grade: 'A',
      score: 10
    },
    {
      date: ISODate('2011-11-23T00:00:00.000Z'),
      grade: 'A',
      score: 9
    },
    {
      date: ISODate('2011-03-10T00:00:00.000Z'),
      grade: 'B',
      score: 14
    }
  ],
  name: 'Morris Park Bake Shop',
  restaurant_id: '30075445'
}
]
DHARSHINI_65>
```

11. Write a MongoDB query to find the restaurant name, borough, longitude and attitude and cuisine for those restaurants which contains 'mon' as three letters somewhere in its name.

**QUERY:**

```
db.restaurants.find({ name: /mon/i }, { name: 1, borough: 1, "address.coord": 1, cuisine: 1 })
```

**OUTPUT:**

```
DHARSHINI_65> db.restaurants.find({ name: /mon/i }, { name: 1, borough: 1, "address.coord": 1, cuisine: 1 })  
DHARSHINI_65> |
```

12. Write a MongoDB query to find the restaurant name, borough, longitude and latitude and cuisine for those restaurants which contain 'Mad' as first three letters of its name.

**QUERY:**

```
db.restaurants.find({ name: /^Mad/i }, { name: 1, borough: 1, "address.coord": 1, cuisine: 1 })
```

**OUTPUT:**

```
DHARSHINI_65> db.restaurants.find({ name: /^Mad/i }, { name: 1, borough: 1, "address.coord": 1, cuisine: 1 })  
DHARSHINI_65> |
```

13. Write a MongoDB query to find the restaurants that have at least one grade with a score of less than 5.

#### QUERY:

```
db.restaurants.find({ "grades": { $elemMatch: { "score": { $lt: 5 } } } })
```

#### OUTPUT:

```
DHARSHINI_65>db.restaurants.find({ "grades": { $elemMatch: { "score": { $lt: 5 } } } })
[ {
  _id: ObjectId('664f3c798752f54dc3cdcdf7'),
  address: {
    building: '1007',
    coord: [ -73.856077, 40.848447 ],
    street: 'Morris Park Ave',
    zipcode: '10462'
  },
  borough: 'Bronx',
  cuisine: 'Bakery',
  grades: [
    {
      date: ISODate('2014-03-03T00:00:00.000Z'),
      grade: 'A',
      score: 2
    },
    {
      date: ISODate('2013-09-11T00:00:00.000Z'),
      grade: 'A',
      score: 6
    },
    {
      date: ISODate('2013-01-24T00:00:00.000Z'),
      grade: 'A',
      score: 10
    },
    {
      date: ISODate('2011-11-23T00:00:00.000Z'),
      grade: 'A',
      score: 9
    },
    {
      date: ISODate('2011-03-10T00:00:00.000Z'),
      grade: 'B',
      score: 14
    }
  ],
  name: 'Morris Park Bake Shop',
  restaurant_id: '30075445'
}
]
DHARSHINI_65>
```

14. Write a MongoDB query to find the restaurants that have at least one grade with a score of less than 5 and that are located in the borough of Manhattan.

#### QUERY:

```
db.restaurants.find({ "grades": { $elemMatch: { "score": { $lt: 5 } } }, "borough": "Manhattan" })
```

#### OUTPUT:

```
DHARSHINI_65> db.restaurants.find({ "grades": { $elemMatch: { "score": { $lt: 5 } } }, "borough": "Manhattan" })
DHARSHINI_65>
```

15. Write a MongoDB query to find the restaurants that have at least one grade with a score of less than 5 and that are located in the borough of Manhattan or Brooklyn.

**QUERY:**

```
db.restaurants.find({ "grades": { $elemMatch: { "score": { $lt: 5 } } }, $or: [{ "borough": "Manhattan" }, { "borough": "Brooklyn" }] })
```

**OUTPUT:**

```
DHARSHINI_65>db.restaurants.find({ "grades": { $elemMatch: { "score": { $lt: 5 } } }, $or: [{ "borough": "Manhattan" }, { "borough": "Brooklyn" }] })
```

16. Write a MongoDB query to find the restaurants that have at least one grade with a score of less than 5 and that are located in the borough of Manhattan or Brooklyn, and their cuisine is not American.

**QUERY:**

```
db.restaurants.find({ "grades": { $elemMatch: { "score": { $lt: 5 } } }, $or: [{ "borough": "Manhattan" }, { "borough": "Brooklyn" }], "cuisine": { $ne: "American" } })
```

**OUTPUT:**

```
DHARSHINI_65>db.restaurants.find({ "grades": { $elemMatch: { "score": { $lt: 5 } } }, $or: [{ "borough": "Manhattan" }, { "borough": "Brooklyn" }], "cuisine": { $ne: "American" } })
```

17. Write a MongoDB query to find the restaurants that have at least one grade with a score of less than 5 and that are located in the borough of Manhattan or Brooklyn, and their cuisine is not American or Chinese.

#### QUERY:

```
db.restaurants.find({ "grades": { $elemMatch: { "score": { $lt: 5 } } }, $or: [{ "borough": "Manhattan" }, { "borough": "Brooklyn" }], "cuisine": { $nin: ["American", "Chinese"] } })
```

#### OUTPUT:

```
DHARSHINI_65>db.restaurants.find({ "grades": { $elemMatch: { "score": { $lt: 5 } } }, $or: [{ "borough": "Manhattan" }, { "borough": "Brooklyn" }], "cuisine": { $nin: ["American", "Chinese"] } })  
DHARSHINI_65>|
```

18. Write a MongoDB query to find the restaurants that have a grade with a score of 2 and a grade with a score of 6.

#### QUERY:

```
db.restaurants.find({ $and: [{ "grades.grade": "A", "grades.score": 2 }, { "grades.grade": "A", "grades.score": 6 }] })
```

#### OUTPUT:

```
DHARSHINI_65>db.restaurants.find({ $and: [{ "grades.grade": "A", "grades.score": 2 }, { "grades.grade": "A", "grades.score": 6 }] })  
[  
  {  
    _id: ObjectId('664f3c798752f54dc3cdcdf7'),  
    address: {  
      building: '1007',  
      coord: [-73.856077, 40.848447],  
      street: 'Morris Park Ave',  
      zipcode: '10462'  
    },  
    borough: 'Bronx',  
    cuisine: 'Bakery',  
    grades: [  
      {  
        date: ISODate('2014-03-03T00:00:00.000Z'),  
        grade: 'A',  
        score: 2  
      },  
      {  
        date: ISODate('2013-09-11T00:00:00.000Z'),  
        grade: 'A',  
        score: 6  
      },  
      {  
        date: ISODate('2013-01-24T00:00:00.000Z'),  
        grade: 'A',  
        score: 10  
      },  
      {  
        date: ISODate('2011-11-23T00:00:00.000Z'),  
        grade: 'A',  
        score: 9  
      },  
      {  
        date: ISODate('2011-03-10T00:00:00.000Z'),  
        grade: 'B',  
        score: 14  
      }  
    ],  
    name: 'Morris Park Bake Shop',  
    restaurant_id: '30075445'  
  }  
]  
DHARSHINI_65>|
```

19. Write a MongoDB query to find the restaurants that have a grade with a score of 2 and a grade with a score of 6 and are located in the borough of Manhattan.

**QUERY:**

```
db.restaurants.find({ $and: [{ "grades.grade": "A", "grades.score": 2 }, { "grades.grade": "A", "grades.score": 6 }], "borough": "Manhattan" })
```

**OUTPUT:**

```
DHARSHINI_65>db.restaurants.find({ $and: [{ "grades.grade": "A", "grades.score": 2 }, { "grades.grade": "A", "grades.score": 6 }], "borough": "Manhattan" })  
DHARSHINI_65>|
```

20. Write a MongoDB query to find the restaurants that have a grade with a score of 2 and a grade with a score of 6 and are located in the borough of Manhattan or Brooklyn.

**QUERY:**

```
db.restaurants.find({ $and: [{ "grades.grade": "A", "grades.score": 2 }, { "grades.grade": "A", "grades.score": 6 }], $or: [{ "borough": "Manhattan" }, { "borough": "Brooklyn" }] })
```

**OUTPUT:**

```
DHARSHINI_65>db.restaurants.find({ $and: [{ "grades.grade": "A", "grades.score": 2 }, { "grades.grade": "A", "grades.score": 6 }], $or: [{ "borough": "Manhattan" }, { "borough": "Brooklyn" }] })  
DHARSHINI_65>|
```

21. Write a MongoDB query to find the restaurants that have a grade with a score of 2 and a grade with a score of 6 and are located in the borough of Manhattan or Brooklyn, and their cuisine is not American.

#### QUERY:

```
db.restaurants.find({ $and: [{ "grades.grade": "A", "grades.score": 2 }, { "grades.grade": "A", "grades.score": 6 }], $or: [{ "borough": "Manhattan" }, { "borough": "Brooklyn" }], "cuisine": { $ne: "American" } })
```

#### OUTPUT:

```
DHARSHINI_65>db.restaurants.find({$and:[{"grades.grade": "A", "grades.score": 2}, {"grades.grade": "A", "grades.score": 6}], $or:[{"borough": "Manhattan"}, {"borough": "Brooklyn"}], "cuisine": {"$ne": "American"}})
```

```
DHARSHINI_65>
```

22. Write a MongoDB query to find the restaurants that have a grade with a score of 2 and a grade with a score of 6 and are located in the borough of Manhattan or Brooklyn, and their cuisine is not American or Chinese.

#### QUERY:

```
db.restaurants.find({ $and: [{ "grades.grade": "A", "grades.score": 2 }, { "grades.grade": "A", "grades.score": 6 }], $or: [{ "borough": "Manhattan" }, { "borough": "Brooklyn" }], "cuisine": { $nin: ["American", "Chinese"] } })
```

#### OUTPUT:

```
DHARSHINI_65>db.restaurants.find({$and:[{"grades.grade": "A", "grades.score": 2}, {"grades.grade": "A", "grades.score": 6}], $or:[{"borough": "Manhattan"}, {"borough": "Brooklyn"}], "cuisine": {"$nin": ["American", "Chinese"]}})
```

```
DHARSHINI_65>
```

23. Write a MongoDB query to find the restaurants that have a grade with a score of 2 or a grade with a score of 6.

#### QUERY:

```
db.restaurants.find({ $or: [{ "grades.score": 2 }, { "grades.score": 6 }] })
```

#### OUTPUT:

```
DHARSHINI_65> db.restaurants.find({ $or: [{ "grades.score": 2 }, { "grades.score": 6 }] })
[ {
  _id: ObjectId('664f3c798752f54dc3cdcdf7'),
  address: {
    building: '1007',
    coord: [ -73.856077, 40.848447 ],
    street: 'Morris Park Ave',
    zipcode: '10462'
  },
  borough: 'Bronx',
  cuisine: 'Bakery',
  grades: [
    {
      date: ISODate('2014-03-03T00:00:00.000Z'),
      grade: 'A',
      score: 2
    },
    {
      date: ISODate('2013-09-11T00:00:00.000Z'),
      grade: 'A',
      score: 6
    },
    {
      date: ISODate('2013-01-24T00:00:00.000Z'),
      grade: 'A',
      score: 10
    },
    {
      date: ISODate('2011-11-23T00:00:00.000Z'),
      grade: 'A',
      score: 9
    },
    {
      date: ISODate('2011-03-10T00:00:00.000Z'),
      grade: 'B',
      score: 14
    }
  ],
  name: 'Morris Park Bake Shop',
  restaurant_id: '30075445'
}
]
DHARSHINI_65>
```

Evaluation Procedure	Marks awarded
Query(5)	
Execution (5)	
Viva(5)	
Total (15)	
Faculty Signature	

#### RESULT:

# MONGO DB

**EX\_NO: 20**

**DATE:**

**1.) Find all movies with full information from the 'movies' collection that released in the year 1893.**

**QUERY:**

```
db.movies.find({ year: 1893 })
```

**OUTPUT:**

```
DHARSHINI_65> db.movies.find({ year: 1893 })
DHARSHINI_65> |
```

**2.) Find all movies with full information from the 'movies' collection that have a runtime greater than 120 minutes.**

**QUERY:**

```
db.movies.find({ runtime: { $gt: 120 } })
```

**OUTPUT:**

```
DHARSHINI_65> db.movies.find({ runtime: { $gt: 120 } })
DHARSHINI_65> |
```

**3.) Find all movies with full information from the 'movies' collection that have "Short" genre.**

**QUERY:**

```
db.movies.find({ genres: 'Short' })
```

**OUTPUT:**

```
DHARSHINI_65> db.movies.find({ genres: 'Short' })
[ {
  _id: ObjectId('573a1390f29313caabcd42e8'),
  plot: 'A group of bandits stage a brazen train hold-up, only to find a determined posse hot on their heels.',
  genres: [ 'Short', 'Western' ],
  runtime: 11,
  cast: [
    'A.C. Abadie',
    'Gilbert M. 'Broncho Billy' Anderson',
    'George Barnes',
    'Justus D. Barnes'
  ],
  poster: 'https://m.media-amazon.com/images/M/MV5BMTU3NjE5NzYtYTYYNS00MDVmLWIwYjgtMmYwYWIxZDYyNzU2XkEyXkFqcGdeQXVyNzQzQxNzI0._V1_SY1000_SX677_AL_.jpg',
  title: 'The Great Train Robbery',
  fullplot: "Among the earliest existing films in American cinema - notable as the first film that presented a narrative story to tell - it depicts a group of cowboy outlaws who hold up a train and rob the passengers. They are then pursued by a Sheriff's posse. Several scenes have color included - all hand tinted.",
  languages: [ 'English' ],
  released: ISODate('1903-12-01T00:00:00.000Z'),
  directors: [ 'Edwin S. Porter' ],
  rated: 'TV-G',
  awards: { wins: 1, nominations: 0, text: '1 win.' },
  lastupdated: '2015-08-13 00:27:59.177000000',
  year: 1903,
  imdb: { rating: 7.4, votes: 9847, id: 439 },
  countries: [ 'USA' ],
  type: 'movie',
  tomatoes: {
    viewer: { rating: 3.7, numReviews: 2559, meter: 75 },
    fresh: 6,
    critic: { rating: 7.6, numReviews: 6, meter: 100 },
    rotten: 0,
    lastUpdated: ISODate('2015-08-08T19:16:10.000Z')
  }
}
]
DHARSHINI_65> |
```

**4.) Retrieve all movies from the 'movies' collection that were directed by "William K.L. Dickson" and include complete information for each movie.**

**QUERY:**

```
db.movies.find({ directors: 'William K.L. Dickson' })
```

**OUTPUT:**

```
DHARSHINI_65> db.movies.find({ directors: 'William K.L. Dickson' })
DHARSHINI_65> |
```

**5.) Retrieve all movies from the 'movies' collection that were released in the USA and include complete information for each movie.**

**QUERY:**

```
db.movies.find({ countries: 'USA' })
```

**OUTPUT:**

```
DHARSHINI_65> db.movies.find({ countries: 'USA' })
[
  {
    _id: ObjectId('573a1390f29313caabed42e8'),
    plot: 'A group of bandits stage a brazen train hold-up, only to find a determined posse hot on their heels.',
    genres: [ 'Short', 'Western' ],
    runtime: 11,
    cast: [
      'A.C. Abadie',
      "Gilbert M. 'Broncho Billy' Anderson",
      'George Barnes',
      'Justus D. Barnes'
    ],
    poster: 'https://m.media-amazon.com/images/M/MV5BMTU3NjE5NzYtYTYYNS00MDVmLWIwYjgtMmYwYWIxZDYyNzU2XkEyXkFqcGdeQXVyNzQzQxNzI0._V1_SY1000_SX677_AL_.jpg',
    title: 'The Great Train Robbery',
    fullplot: "Among the earliest existing films in American cinema - notable as the first film that presented a narrative story to tell - it depicts a group of cowboy outlaws who hold up a train and rob the passengers. They are then pursued by a Sheriff's posse. Several scenes have color included - all hand tinted.",
    languages: [ 'English' ],
    released: ISODate('1903-12-01T00:00:00.000Z'),
    directors: [ 'Edwin S. Porter' ],
    rated: 'TV-G',
    awards: { wins: 1, nominations: 0, text: '1 win.' },
    lastupdated: '2015-08-13 00:27:59.177000000',
    year: 1903,
    imdb: { rating: 7.4, votes: 9847, id: 439 },
    countries: [ 'USA' ],
    type: 'movie',
    tomatoes: {
      viewer: { rating: 3.7, numReviews: 2559, meter: 75 },
      fresh: 6,
      critic: { rating: 7.6, numReviews: 6, meter: 100 },
      rotten: 0,
      lastUpdated: ISODate('2015-08-08T19:16:10.000Z')
    }
  }
]
DHARSHINI_65>
```

**6.) Retrieve all movies from the 'movies' collection that have complete information and are rated as "UNRATED".**

**QUERY:**

```
db.movies.find({ rated: 'UNRATED' })
```

**OUTPUT:**

```
DHARSHINI_65> db.movies.find({ rated: 'UNRATED' })
DHARSHINI_65> |
```

**7.) Retrieve all movies from the 'movies' collection that have complete information and have received more than 1000 votes on IMDb.**

**QUERY:**

```
db.movies.find({ 'imdb.votes': { $gt: 1000 } })
```

**OUTPUT:**

```
DHARSHINI_65> db.movies.find({ 'imdb.votes': { $gt: 1000 } })
[
  {
    _id: ObjectId('573a1390f29313caabcd42e8'),
    plot: 'A group of bandits stage a brazen train hold-up, only to find a determined posse hot on their heels.',
    genres: [ 'Short', 'Western' ],
    runtime: 11,
    cast: [
      'A.C. Abadie',
      'Gilbert M. "Broncho Billy" Anderson',
      'George Barnes',
      'Justus D. Barnes'
    ],
    poster: 'https://m.media-amazon.com/images/M/MV5BMTU3NjE5NzYtYTYYNS00MDVmLWIwYjgtMmYwYWIxZDYyNzU2XkEyXkFqcGdeQXVyNzQzNzQzI@._V1_SY1000_SX677_AL_.jpg',
    title: 'The Great Train Robbery',
    fullplot: "Among the earliest existing films in American cinema - notable as the first film that presented a narrative story to tell - it depicts a group of cowboy outlaws who hold up a train and rob the passengers. They are then pursued by a Sheriff's posse. Several scenes have color included - all hand tinted.",
    languages: [ 'English' ],
    released: ISODate('1903-12-01T00:00:00.000Z'),
    directors: [ 'Edwin S. Porter' ],
    rated: 'TV-G',
    awards: { wins: 1, nominations: 0, text: '1 win.' },
    lastupdated: '2015-08-13 00:27:59.177000000',
    year: 1903,
    imbd: { rating: 7.4, votes: 9847, id: 439 },
    countries: [ 'USA' ],
    type: 'movie',
    tomatoes: {
      viewer: { rating: 3.7, numReviews: 2559, meter: 75 },
      fresh: 6,
      critic: { rating: 7.6, numReviews: 6, meter: 100 },
      rotten: 0,
      lastUpdated: ISODate('2015-08-08T19:16:10.000Z')
    }
  }
]
DHARSHINI_65> |
```

**8.) Retrieve all movies from the 'movies' collection that have complete information and have an IMDb rating higher than 7.**

**QUERY:**

```
db.movies.find({ 'imdb.rating': { $gt: 7 } })
```

**OUTPUT:**

```
DHARSHINI_65> db.movies.find({ 'imdb.rating': { $gt: 7 } })
[
  {
    _id: ObjectId('573a1390f29313caabcd42e8'),
    plot: 'A group of bandits stage a brazen train hold-up, only to find a determined posse hot on their heels.',
    genres: [ 'Short', 'Western' ],
    runtime: 11,
    cast: [
      'A.C. Abadie',
      'Gilbert M. "Broncho Billy" Anderson',
      'George Barnes',
      'Justus D. Barnes'
    ],
    poster: 'https://m.media-amazon.com/images/M/MV5BMTU3NjE5NzYtYTYYNS00MDVmLWIwYjgtMmYwYWIxZDYyNzU2XkEyXkFqcGdeQXVyNzQzNzQzI@._V1_SY1000_SX677_AL_.jpg',
    title: 'The Great Train Robbery',
    fullplot: "Among the earliest existing films in American cinema - notable as the first film that presented a narrative story to tell - it depicts a group of cowboy outlaws who hold up a train and rob the passengers. They are then pursued by a Sheriff's posse. Several scenes have color included - all hand tinted.",
    languages: [ 'English' ],
    released: ISODate('1903-12-01T00:00:00.000Z'),
    directors: [ 'Edwin S. Porter' ],
    rated: 'TV-G',
    awards: { wins: 1, nominations: 0, text: '1 win.' },
    lastupdated: '2015-08-13 00:27:59.177000000',
    year: 1903,
    imbd: { rating: 7.4, votes: 9847, id: 439 },
    countries: [ 'USA' ],
    type: 'movie',
    tomatoes: {
      viewer: { rating: 3.7, numReviews: 2559, meter: 75 },
      fresh: 6,
      critic: { rating: 7.6, numReviews: 6, meter: 100 },
      rotten: 0,
      lastUpdated: ISODate('2015-08-08T19:16:10.000Z')
    }
  }
]
DHARSHINI_65> |
```

**9.) Retrieve all movies from the 'movies' collection that have complete information and have a viewer rating higher than 4 on Tomatoes.**

**QUERY:**

```
db.movies.find({ 'tomatoes.viewer.rating': { $gt: 4 } })
```

**OUTPUT:**

```
DHARSHINI_65> db.movies.find({ 'tomatoes.viewer.rating': { $gt: 4 } })  
DHARSHINI_65> |
```

**10.) Retrieve all movies from the 'movies' collection that have received an award.**

**QUERY:**

```
db.movies.find({ 'awards.wins': { $gt: 0 } })
```

**OUTPUT:**

```
DHARSHINI_65> db.movies.find({ 'awards.wins': { $gt: 0 } })  
[  
  {  
    _id: ObjectId('573a1390f29313caabcd42e8'),  
    plot: 'A group of bandits stage a brazen train hold-up, only to find a determined posse hot on their heels.',  
    genres: [ 'Short', 'Western' ],  
    runtime: 11,  
    cast: [  
      'A.C. Abadie',  
      "Gilbert M. 'Broncho Billy' Anderson",  
      'George Barnes',  
      'Justus D. Barnes'  
    ],  
    poster: 'https://m.media-amazon.com/images/M/MV5BMTU3NjE5NzYtYTYYNS00MDVmLWIwYjgtMmYwYIxZDYyNzU2XkEyXkFqcGdeQXVyNzQzNzI@._V1_SY1000_SX677_AL_.jpg',  
    title: 'The Great Train Robbery',  
    fullplot: "Among the earliest existing films in American cinema - notable as the first film that presented a narrative story to tell - it depicts a group of cowboy outlaws who hold up a train and rob the passengers. They are then pursued by a Sheriff's posse. Several scenes have color included - all hand tinted.",  
    languages: [ 'English' ],  
    released: ISODate('1903-12-01T00:00:00.000Z'),  
    directors: [ 'Edwin S. Porter' ],  
    rated: 'TV-G',  
    awards: { wins: 1, nominations: 0, text: '1 win.' },  
    lastupdated: '2015-08-13 00:27:59.177000000',  
    year: 1903,  
    imdb: { rating: 7.4, votes: 9847, id: 439 },  
    countries: [ 'USA' ],  
    type: 'movie',  
    tomatoes: {  
      viewer: { rating: 3.7, numReviews: 2559, meter: 75 },  
      fresh: 6,  
      critic: { rating: 7.6, numReviews: 6, meter: 100 },  
      rotten: 0,  
      lastUpdated: ISODate('2015-08-08T19:16:10.000Z')  
    }  
  }  
]  
DHARSHINI_65> |
```

**11.) Find all movies with title, languages, released, directors, writers, awards, year, genres, runtime, cast, countries from the 'movies' collection in MongoDB that have at least one nomination.**

**QUERY:**

```
db.movies.find( { 'awards.nominations': { $gt: 0 } }, { title: 1, languages: 1, released: 1, directors: 1, writers: 1, awards: 1, year: 1, genres: 1, runtime: 1, cast: 1, countries: 1 } )
```

**OUTPUT:**

```
DHARSHINI_65> db.movies.find(
...   { 'awards.nominations': { $gt: 0 } },
...   { title: 1, languages: 1, released: 1, directors: 1, writers: 1, awards: 1, year: 1, genres: 1, runtime: 1, cast: 1, countries: 1 }
... )
```

```
DHARSHINI_65> |
```

**12.) Find all movies with title, languages, released, directors, writers, awards, year, genres, runtime, cast, countries from the 'movies' collection in MongoDB with cast including "Charles Kayser".**

**QUERY:**

```
db.movies.find( { cast: 'Charles Kayser' }, { title: 1, languages: 1, released: 1, directors: 1, writers: 1, awards: 1, year: 1, genres: 1, runtime: 1, cast: 1, countries: 1 } )
```

**OUTPUT:**

```
DHARSHINI_65> db.movies.find(
...   { cast: 'Charles Kayser' },
...   { title: 1, languages: 1, released: 1, directors: 1, writers: 1, awards: 1, year: 1, genres: 1, runtime: 1, cast: 1, countries: 1 }
... )
```

```
DHARSHINI_65> |
```

**13.) Retrieve all movies with title, languages, released, directors, writers, countries from the 'movies' collection in MongoDB that released on May 9, 1893.**

**QUERY:**

```
db.movies.find( { released: ISODate("1893-05-09T00:00:00.000Z") }, { title: 1, languages: 1, released: 1, directors: 1, writers: 1, countries: 1 } )
```

**OUTPUT:**

```
DHARSHINI_65> db.movies.find(
...   { released: ISODate('1893-05-09T00:00:00.000Z') },
...   { title: 1, languages: 1, released: 1, directors: 1, writers: 1, countries: 1 }
... )
```

```
DHARSHINI_65> |
```

**14.) Retrieve all movies with title, languages, released, directors, writers, countries from the 'movies' collection in MongoDB that have a word "scene" in the title.**

**QUERY:**

```
db.movies.find( { title: /scene/i }, { title: 1, languages: 1, released: 1, directors: 1, writers: 1, countries: 1 } )
```

**OUTPUT:**

```
DHARSHINI_65> db.movies.find(
...   { title: /scene/i },
...   { title: 1, languages: 1, released: 1, directors: 1, writers: 1, countries: 1 }
... )
```

```
DHARSHINI_65> |
```

Evaluation Procedure	Marks awarded
Query(5)	
Execution (5)	
Viva(5)	
Total (15)	
Faculty Signature	

**RESULT:**