

# 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 dep5
(
DEPT_ID int not null,
DEPT_NAME VARCHAR(30),
MANAGER_ID VARCHAR (30),
LOCATION_ID INT
);
```

## OUTPUT:

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

```
1 create table dep5
2 (
3     DEPT_ID int not null,
4     DEPT_NAME VARCHAR(30),
5     MANAGER_ID VARCHAR(30),
6     LOCATION_ID INT
7 );
```

The 'Results' tab displays the output of the query:

```
Table created.
0.04 seconds
```

At the bottom of the page, there are footer links: 220701040@rajalakshmi.edu.in, bhanu23, en, Copyright © 1999, 2023, Oracle and/or its affiliates., and Oracle APEX 23.2.4.

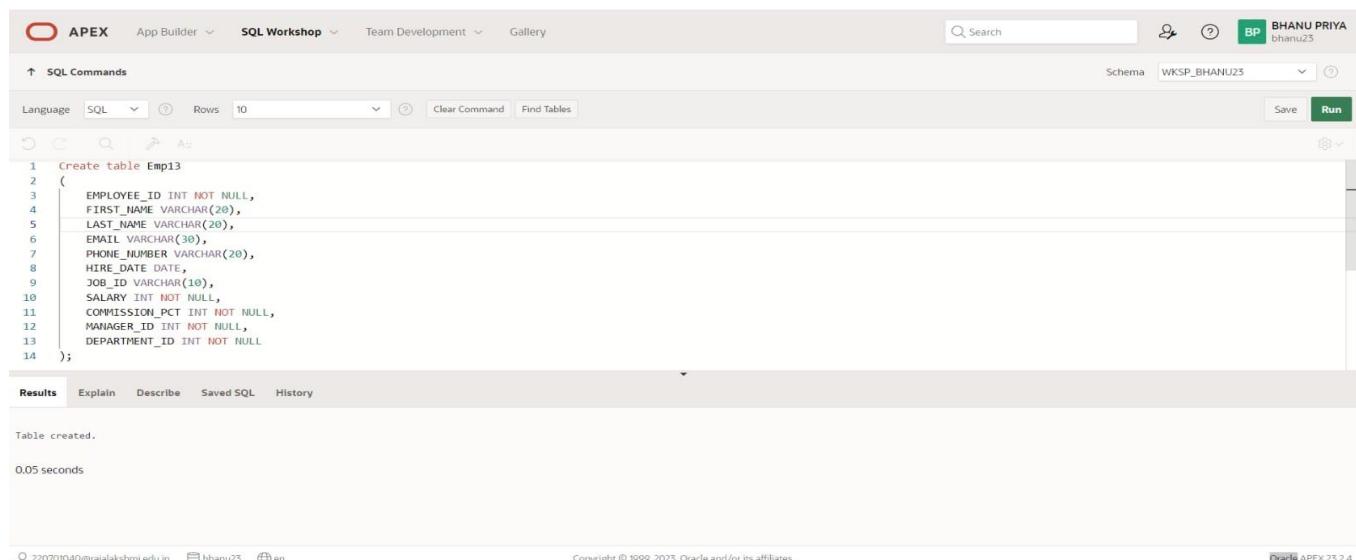
2.Create the EMP table based on the following instance chart. Confirm that the table is created.

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 Emp13
(
EMPLOYEE_ID INT NOT NULL,
FIRST_NAME VARCHAR (20),
LAST_NAME VARCHAR (20) ,
EMAIL VARCHAR (30),
PHONE_NUMBER VARCHAR (20) ,
HIRE_DATE DATE,
JOB_ID VARCHAR (10),
SALARY INT NOT NULL,
COMMISSION_PCT INT NOT NULL,
MANAGER_ID INT NOT NULL,
DEPARTMENT_ID INT NOT NULL
);
```

### OUTPUT:



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

```
1 Create table Emp13
2 (
3     EMPLOYEE_ID INT NOT NULL,
4     FIRST_NAME VARCHAR(20),
5     LAST_NAME VARCHAR(20),
6     EMAIL VARCHAR(30),
7     PHONE_NUMBER VARCHAR(20),
8     HIRE_DATE DATE,
9     JOB_ID VARCHAR(10),
10    SALARY INT NOT NULL,
11    COMMISSION_PCT INT NOT NULL,
12    MANAGER_ID INT NOT NULL,
13    DEPARTMENT_ID INT NOT NULL
14 );
```

Below the code, the results show:

```
Table created.
0.05 seconds
```

At the bottom, the footer includes:

220701040@rajalakshmi.edu.in bhanu23 en Copyright © 1999, 2025, Oracle and/or its affiliates. Oracle APEX 23.2.4

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

### QUERY:

```
ALTER TABLE my_emp3  
modify (last_name varchar (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. The right side shows the user's profile: BP BHANU PRIYA and the schema: WKSP\_BHANU23. The main area is titled "SQL Commands". The "Language" dropdown is set to "SQL". The "Rows" dropdown is set to 10. Below the input field, there are buttons for "Clear Command" and "Find Tables". The input field contains the following SQL code:

```
1 ALTER TABLE my_emp3  
2 modify(last_name varchar(50));
```

Below the input field, there are tabs for "Results", "Explain", "Describe", "Saved SQL", and "History". The "Results" tab is selected. The output section displays the message "Table altered." and "0.07 seconds". At the bottom of the page, there are footer links for "220701040@rajalakshmi.edu.in", "bhanu23", and "en", along with copyright information: "Copyright © 1999, 2023, Oracle and/or its affiliates." and "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 emp21
(
ID int not null,
first_name varchar (20),
last_name varchar (20),
salary int not null,
dept_id int not null
);
```

### 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. On the right, there's a search bar, user profile, and a workspace named 'BHANU PRIYA bhanu23'. The main area is titled 'SQL Commands' with tabs for Language (set to SQL), Rows (set to 10), and various icons for clear command, find tables, and run. Below this is a code editor containing the SQL script to create the 'emp21' table. The code is as follows:

```
1 create table emp21
2 (
3     ID int not null,
4     first_name varchar(20),
5     last_name varchar(20),
6     salary int not null,
7     dept_id int not null
8 );
```

Below the code editor, there are tabs for Results, Explain, Describe, Saved SQL, and History. The 'Results' tab is selected, showing the output 'Table created.' and a execution time of '0.04 seconds'. At the bottom, the footer includes user information (220701040@rajalakshmi.edu.in, bhanu23, en), copyright notice (Copyright © 1999, 2023, Oracle and/or its affiliates), and the version 'Oracle APEX 23.2.4'.

5.Drop the EMP table.

**QUERY:**

```
drop table emp13;
```

The screenshot shows the Oracle APEX interface with the SQL Workshop module selected. In the SQL Commands section, the command `drop table emp13;` is entered. The Results tab is selected, showing the output: "Table dropped." and "0.09 seconds". The top right corner shows the user's name as "Bhanu Priya".

**OUTPUT:**

6.Rename the EMPLOYEES2 table as EMP.

**QUERY:**

```
rename emp21 to emp13;
```

**OUTPUT:**

The screenshot shows the Oracle APEX interface with the SQL Workshop module selected. In the SQL Commands section, the command `rename emp21 to emp13;` is entered. The Results tab is selected, showing the output: "Statement processed." and "0.06 seconds". The top right corner shows the user's name as "Bhanu Priya".

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

### QUERY:

comment on table dep2 is 'Department info';

### OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. In the top navigation bar, 'SQL Workshop' is selected. The schema dropdown shows 'WKSP\_BHANU23'. The SQL command window contains the following code:

```
1 comment on table dep2 is 'Department info';
2
3
4
5
```

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

```
Statement processed.
0.03 seconds
```

At the bottom, the footer includes the URL '220701040@rajalakshmi.edu.in', the session ID 'bhnu23', and the environment 'en'.

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

### QUERY:

alter table emp13  
drop column first\_name;

### OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. In the top navigation bar, 'SQL Workshop' is selected. The schema dropdown shows 'WKSP\_BHANU23'. The SQL command window contains the following code:

```
1 alter table emp13
2 drop column first_name;
```

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

```
Table altered.
0.07 seconds
```

At the bottom, the footer includes the URL '220701040@rajalakshmi.edu.in', the session ID 'bhnu23', and the environment 'en'.

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_EMPLOYEE2(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. The right side shows a user profile for 'Bhanu Priya' with the schema 'WKSP\_BHANU23'. The main area is titled 'SQL Commands' and contains a single command: 'create table MY\_EMPLOYEE2(ID Number(4) Not null, Last\_name Varchar(25), First\_name Varchar(25), Userid Varchar(25), Salary Number(9,2));'. Below the command, the 'Results' tab is selected, displaying the output: 'Table created.' and '0.04 seconds'. The bottom footer includes copyright information for Oracle and the APEX version.

```
create table MY_EMPLOYEE2(ID Number(4) Not null, Last_name Varchar(25), First_name Varchar(25), Userid Varchar(25), Salary Number(9,2));
```

Table created.  
0.04 seconds

Copyright © 1999, 2025, Oracle and/or its affiliates.  
Oracle APEX 25.2.4

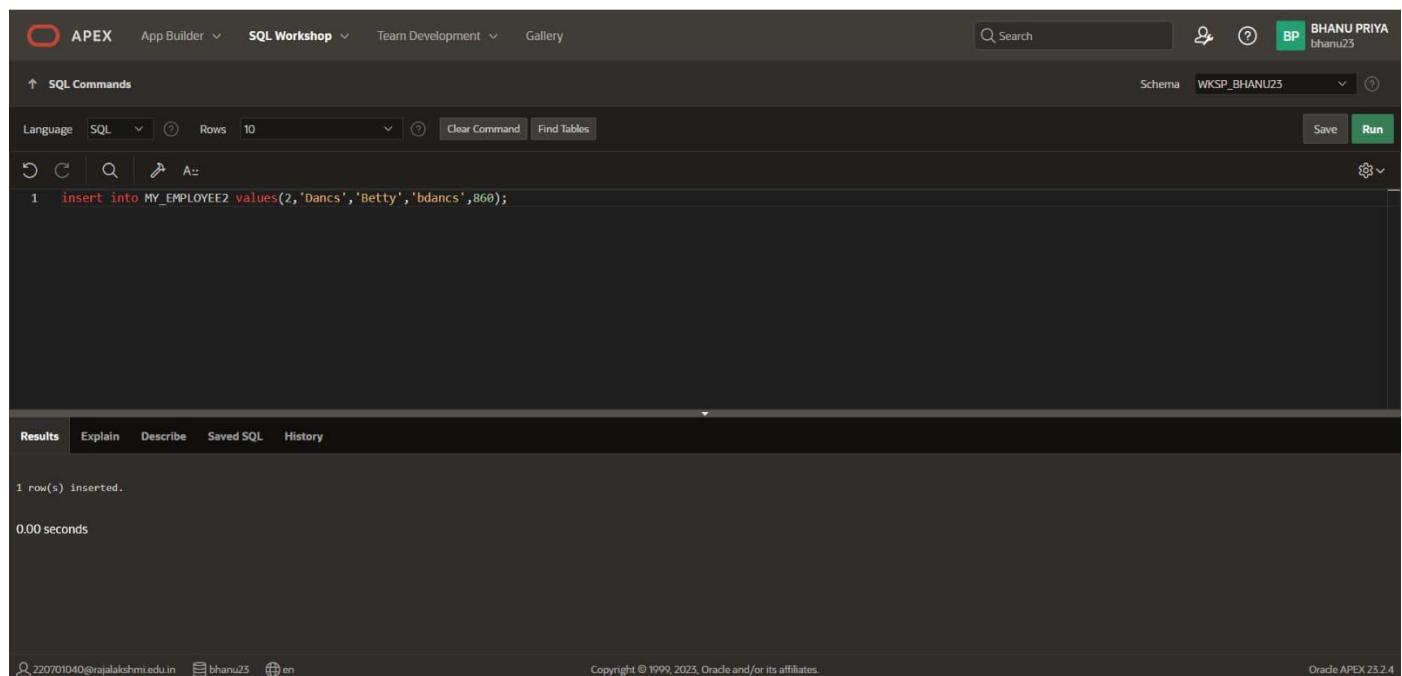
2.Add the first and second rows data to MY\_EMPLOYEE table from the following sample data.

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_EMPLOYEE2 values(2, 'Dancs', 'Betty', 'bdancs', 860);
```

### OUTPUT:



The screenshot shows the Oracle APEX SQL Workshop interface. In the SQL Commands section, the following SQL statement is entered:

```
1 insert into MY_EMPLOYEE2 values(2, 'Dancs', 'Betty', 'bdancs', 860);
```

Below the command, the results are displayed:

1 row(s) inserted.  
0.00 seconds

At the bottom, the footer includes the user information 220701040@rajalakshmi.edu.in, session ID bhanu25, and Oracle APEX 25.2.4.

3. Display the table with values.

#### QUERY:

```
select*from MY_EMPLOYEE2;
```

#### 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 the user 'BHUANU PRIYA bhanu23'. The main area has tabs for 'SQL Commands' and 'Results' (selected). The SQL command entered is 'select\*from MY\_EMPLOYEE2;'. The results table has columns: ID, LAST\_NAME, FIRST\_NAME, USERID, and SALARY. Two rows are displayed:

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

Below the table, it says '2 rows returned in 0.03 seconds' and there is a 'Download' link. The bottom footer includes the URL '220701040@rajalakshmi.edu.in', session 'bhanu23', and page 'en'. Copyright information and 'Oracle APEX 23.2.4' are also present.

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_EMPLOYEE2 values (3, 'Biri', 'Ben', 'bbiri' ,1100);
```

#### 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 the user 'BHUANU PRIYA bhanu23'. The main area has tabs for 'SQL Commands' and 'Results' (selected). The SQL command entered is 'insert into MY\_EMPLOYEE2 values(3, 'Biri', 'Ben', 'bbiri', 1100);'. The results table shows the message '1 row(s) inserted.' and a timestamp '0.01 seconds'. The bottom footer includes the URL '220701040@rajalakshmi.edu.in', session 'bhanu23', and page 'en'. Copyright information and 'Oracle APEX 23.2.4' are also present.

5. Make the data additions permanent.

### QUERY:

select\*from MY\_EMPLOYEE2;

### 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 the user 'BHANDU PRIYA' and schema 'WKSP\_BHANU23'. The main area has tabs for 'SQL Commands' and 'Results'. The SQL command entered is 'select\*from MY\_EMPLOYEE2;'. The results section displays a table with two rows of data:

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

Below the table, it says '2 rows returned in 0.03 seconds' and there is a 'Download' link. The bottom of the page includes copyright information and the text 'Oracle APEX 23.2.4'.

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

### QUERY:

update MY\_EMPLOYEE2 set last\_name ='Drexler'where id=3;

### 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 the user 'BHANDU PRIYA' and schema 'WKSP\_BHANU23'. The main area has tabs for 'SQL Commands' and 'Results'. The SQL command entered is 'update MY\_EMPLOYEE2 set last\_name ='Drexler'where id=3;'. The results section shows the message '1 row(s) updated.' and '0.01 seconds'. The bottom of the page includes copyright information and the text 'Oracle APEX 23.2.4'.

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

### QUERY:

```
update MY_EMPLOYEE2 set Salary =1000 where Salary<900
```

### 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 the user 'BHANU PRIYA' and schema 'WKSP\_BHANUZ3'. The main area is titled 'SQL Commands' with a sub-section '↑ SQL Commands'. It contains a text input field with the following SQL command:

```
1 update MY_EMPLOYEE2 set Salary =1000 where Salary<900
```

Below the command, the 'Results' tab is selected, showing the output:

```
3 row(s) updated.  
0.01 seconds
```

At the bottom, the footer includes the URL '220701040@rajalakshmi.edu.in', session ID 'bhau25', and language 'en'. The copyright notice 'Copyright © 1999, 2023, Oracle and/or its affiliates.' and the version 'Oracle APEX 23.2.4' are also present.

8.Delete Betty from MY\_EMPLOYEE table.

### QUERY:

```
delete from MY_EMPLOYEE2 where First_name= 'Betty';
```

### OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface, identical to the previous one but with a different query. The top navigation bar and user information are the same. The main area is titled 'SQL Commands' with a sub-section '↑ SQL Commands'. It contains the following SQL command:

```
1 delete from MY_EMPLOYEE2 where First_name='Betty';
```

Below the command, the 'Results' tab is selected, showing the output:

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

At the bottom, the footer includes the URL '220701040@rajalakshmi.edu.in', session ID 'bhau25', and language 'en'. The copyright notice 'Copyright © 1999, 2023, Oracle and/or its affiliates.' and the version 'Oracle APEX 23.2.4' are also present.

9.Empty the fourth row of the emp table.

### QUERY:

```
delete from MY_EMPLOYEE2 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 'BP BHANU PRIYA' with the schema 'WKSP\_BHANU25'. The SQL Commands tab is active, showing the query: '1 delete from MY\_EMPLOYEE2 where id=4'. The results section below shows the output: '1 row(s) deleted.' and '0.01 seconds'. The bottom status bar displays the user's email ('220701040@rajalakshmi.edu.in'), session ID ('bhnu25'), and the Oracle APEX version ('Oracle APEX 23.2.4').

```
delete from MY_EMPLOYEE2 where id=4
```

1 row(s) deleted.  
0.01 seconds

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 COMPANY(ID number(6), last_name varchar2(25) not null, email varchar2(25),  
salary number(8,2), constraint my_emp_id_pk PRIMARY KEY(ID));
```

**OUTPUT:**

The screenshot shows the Oracle APEX SQL Workshop interface. In the top navigation bar, 'APEX' is selected. The main area is titled 'SQL Commands'. The 'Language' dropdown is set to 'SQL'. The query entered is:

```
1 CREATE TABLE COMPANY(ID number(6),last_name varchar2(25) not null,email varchar2(25),salary number(8,2), constraint my_emp_id_pk PRIMARY KEY(ID));
```

In the results section, the message 'Table created.' is displayed. At the bottom, the copyright notice 'Copyright © 1999, 2025, Oracle and/or its affiliates.' and the version 'Oracle APEX 23.2.4' are visible.

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

## QUERY:

```
CREATE TABLE DEPTS(ID number(6), last_name varchar2(25) not null,email varchar2(25), constraint my_dept5_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. The right side shows a user profile for 'BHANU PRIYA bhanu23'. The main workspace has tabs for SQL Commands, Explain, Describe, Saved SQL, and History. The SQL Commands tab is active, showing the following SQL command:

```
1 CREATE TABLE DEPTS(ID number(6),last_name varchar2(25) not null,email varchar2(25), constraint my_dept5_id_pk PRIMARY KEY(ID));
```

Below the command, the results section displays the message "Table created." and a execution time of "0.05 seconds". The bottom of the screen shows copyright information for Oracle and the APEX version.

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 COMPANY ADD CONSTRAINT MY_COMPANY_DEPT4_ID_FK  
FOREIGN KEY(DEPT_ID) REFERENCES COMPANY (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. On the right, there's a search bar, user profile, and session information (Schema: WKSP\_BHANU25, User: BHANU PRIYA). The main workspace is titled 'SQL Commands' and contains a single line of SQL code: 'ALTER TABLE COMPANY ADD CONSTRAINT MY\_COMPANY\_DEPT4\_ID\_FK FOREIGN KEY(DEPT\_ID) REFERENCES COMPANY (ID);'. Below the code, the results tab is selected, showing the output: 'Table altered.' and '0.07 seconds'. The bottom footer displays copyright information: 'Copyright © 1999, 2023, Oracle and/or its affiliates.' and 'Oracle APEX 23.2.4'.

```
ALTER TABLE COMPANY ADD CONSTRAINT MY_COMPANY_DEPT4_ID_FK FOREIGN KEY(DEPT_ID) REFERENCES COMPANY (ID);
```

Table altered.  
0.07 seconds

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

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 COMPANY 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. The right side shows the user profile 'BHANU PRIYA' and schema 'WKSP\_BHANU23'. The main workspace is titled 'SQL Commands' and contains the following SQL statement:

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

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

```
Table altered.
```

Execution time is listed as 0.05 seconds. The bottom of the screen displays copyright information and the version '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:

DATE:

1. The following statement executes successfully.

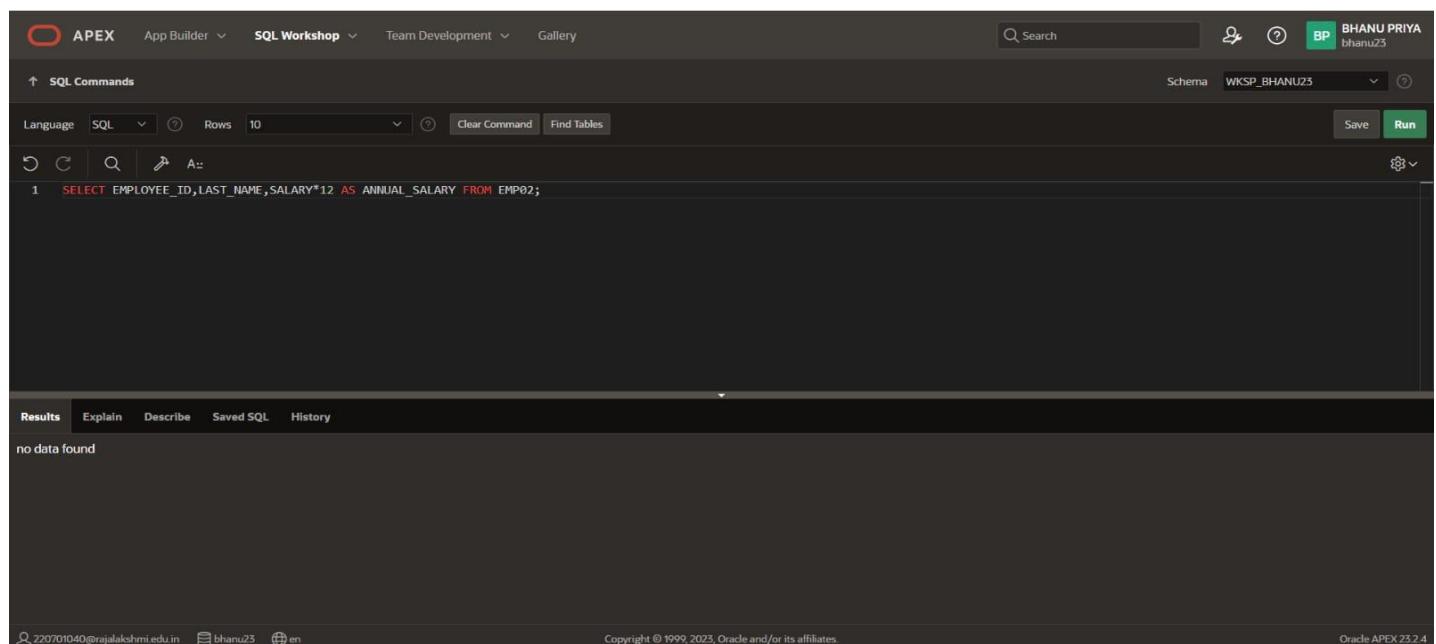
## Identify the Errors

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

## QUERY:

```
SELECT EMPLOYEE_ID, LAST_NAME, SALARY*12 AS ANNUAL_SALARY FROM EMP02;
```

## 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. On the right, there's a user profile for 'BHANU PRIYA' with the session ID 'bhanu23'. The main workspace is titled 'SQL Commands'. It features a toolbar with icons for Undo, Redo, Find, Replace, and Run. Below the toolbar, the language is set to 'SQL' and the number of rows to '10'. The command input field contains the following SQL query:

```
1 SELECT EMPLOYEE_ID, LAST_NAME, SALARY*12 AS ANNUAL_SALARY FROM EMP02;
```

The results section is currently empty, displaying the message 'no data found'. At the bottom of the page, footer information includes the email '220701040@rajalakshmi.edu.in', the session ID 'bhanu23', and the copyright notice 'Copyright © 1999, 2023, Oracle and/or its affiliates.' followed by 'Oracle APEX 23.2.4'.

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

### QUERY:

```
DESC DEPARTMENTS39;
```

### 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 'BHANU PRIYA' with schema 'WKSP\_BHANU23'. The main area displays the results of the SQL command 'DESC DEPARTMENTS39;'. Below the command, the table structure is shown in a grid format:

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
DEPARTMENTS39	DEPT_ID	NUMBER	22	-	0	-	-	-	-
	DEPT_NAME	VARCHAR2	40	-	-	-	✓	-	-
	MANAGER_ID	NUMBER	22	-	0	-	-	-	-

At the bottom of the interface, there are tabs for Results, Explain, Describe (which is selected), Saved SQL, and History. The status bar at the bottom shows the user's email (220701040@rajalakshmi.edu.in), session ID (bhangu23), and language (en). Copyright information and the Oracle APEX version (23.2.4) are also present.

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_NUMBER, LAST_NAME, JOB_CODE, HIRE_DATE FROM EMP07;
```

### 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 the user 'BHANU PRIYA bhanu23'. The main area has tabs for 'SQL Commands' and 'Results'. Under 'SQL Commands', the query is displayed:

```
1 SELECT EMPLOYEE_NUMBER, LAST_NAME, JOB_CODE, HIRE_DATE FROM EMP07;
2
3
4
```

Under 'Results', the output is shown in a table:

EMPLOYEE_NUMBER	LAST_NAME	JOB_CODE	HIRE_DATE
4	BHANU	1	7

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

4.Provide an alias STARTDATE for the hire date.

### QUERY:

```
SELECT HIRE_DATE AS "STARTDATE" FROM EMP07;
```

### 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 the user 'BHANU PRIYA bhanu23'. The main area has tabs for 'SQL Commands' and 'Results'. Under 'SQL Commands', the query is displayed:

```
1 SELECT HIRE_DATE AS "STARTDATE" FROM EMP07;
```

Under 'Results', the output is shown in a table:

STARTDATE
7

Below the table, it says '1 rows returned in 0.00 seconds' and has a 'Download' link. The bottom of the page includes copyright information and the text 'Oracle APEX 23.2.4'.

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

### QUERY:

```
SELECT DISTINCT JOB_CODE FROM EMP07;
```

### 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 'BHANU PRIYA' with the schema 'WKSP\_BHANU23'. The SQL Commands section contains the query: 'SELECT DISTINCT JOB\_CODE FROM EMP07;'. The Results section displays the output: 'JOB\_CODE' with one row containing '1'. Below the results, it says '1 rows returned in 0.02 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'.

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 EMP07;
```

### 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 'BHANU PRIYA' with the schema 'WKSP\_BHANU23'. The SQL Commands section contains the query: 'SELECT LAST\_NAME || ',' || JOB\_CODE AS "EMPLOYEE AND TITLE" FROM EMP07;'. The Results section displays the output: 'EMPLOYEE AND TITLE' with one row containing 'BHANU,1'. 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 the APEX version 'Oracle APEX 23.2.4'.

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_NUMBER || ',' || HIRE_DATE AS "THE OUTPUT" FROM EMP07;
```

### OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. In the top navigation bar, 'APEX' is selected. Below it, 'SQL Workshop' is active. The main area contains the following SQL command:

```
1 SELECT LAST_NAME || ',' || JOB_CODE || ',' || EMPLOYEE_NUMBER || ',' || HIRE_DATE AS "THE OUTPUT" FROM EMP07;
```

Below the command, the results tab is selected. The output is displayed in a table with a single row:

THE OUTPUT
BHANU,1,4,7

Below the table, it says '1 rows returned in 0.00 seconds'. At the bottom of the page, there are footer links for user information, copyright notice, and version information.

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

**RESULT:**

# RESTRICTING AND SORTING DATA

EX\_NO:5

DATE:

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

QUERY:

Select last\_name,salary from empo21 where salary>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. On the right, there's a user profile for 'BHANU PRIYA bhanu23'. The main area has tabs for SQL Commands, Results, Explain, Describe, Saved SQL, and History. The SQL Commands tab contains the query: 'select last\_name,salary from empo21 where salary >12000;'. The Results tab displays the output in a table:

LAST_NAME	SALARY
appu	100000
priya	100000
dhruv	20000

Below the table, it says '3 rows returned in 0.03 seconds' and 'Download'.

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

QUERY:

select last\_name,department\_number from empo21 where emp\_id=176;

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface, similar to the previous one. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there's a user profile for 'BHANU PRIYA bhanu23'. The main area has tabs for SQL Commands, Results, Explain, Describe, Saved SQL, and History. The SQL Commands tab contains the query: 'select last\_name,department\_number from empo21 where emp\_id=176;'. The Results tab displays the output in a table:

LAST_NAME	DEPARTMENT_NUMBER
appu	111

Below the table, it says '1 rows returned in 0.03 seconds' and 'Download'.

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,salary from empo21 where salary not between 5000 and 12000;
```

#### 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. On the right, there's a user profile for 'BHANU PRIYA bhanu23'. The main area has tabs for SQL Commands, Results (selected), Explain, Describe, Saved SQL, and History. The SQL Commands tab contains the query: '1 select last\_name ,salary from empo21 where salary not between 5000 and 12000;'. The Results tab displays the output in a table:

LAST_NAME	SALARY
appu	100000
joe	200
priya	100000
dhruv	20000

Below the table, it says '4 rows returned in 0.01 seconds' and has a 'Download' link. At the bottom, it shows the URL '220701040@rajalakshmi.edu.in bhanu23 en', the copyright notice 'Copyright © 1999, 2023, Oracle and/or its affiliates.', and the 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 empo21 where hire_date between '02/20/1998' and '05/01/1998' order by hire_date asc;
```

#### 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. On the right, there's a user profile for 'BHANU PRIYA bhanu23'. The main area has tabs for SQL Commands, Results (selected), Explain, Describe, Saved SQL, and History. The SQL Commands tab contains the query: '1 select last\_name,job\_id,Hire\_Date from empo21 where Hire\_Date between '02/20/1998' and '05/01/1998' order by Hire\_date asc;'. The Results tab displays the output in a table:

LAST_NAME	JOB_ID	HIRE_DATE
appu	1234	02/21/1998
dhruv	9101	03/07/1998
joe	8978	04/22/1998

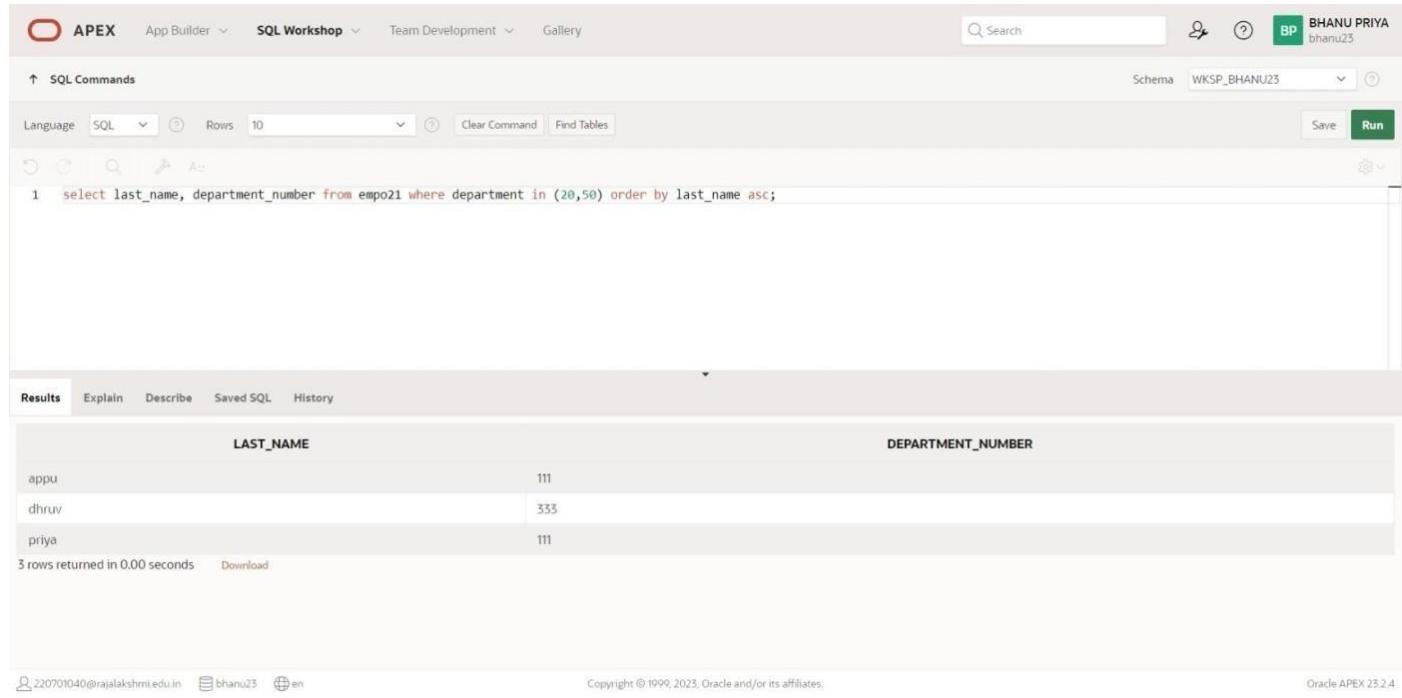
Below the table, it says '3 rows returned in 0.01 seconds' and has a 'Download' link. At the bottom, it shows the URL '220701040@rajalakshmi.edu.in bhanu23 en', the copyright notice 'Copyright © 1999, 2023, Oracle and/or its affiliates.', and the version 'Oracle APEX 23.2.4'.

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, department_number from empo21 where department in (20,50) order by last_name asc;
```

#### 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. On the right, a user profile for 'BHANU PRIYA' is shown with the schema 'WKSP\_BHANU23'. The main area displays the SQL command:

```
1 select last_name, department_number from empo21 where department in (20,50) order by last_name asc;
```

Below the command, the results are presented in a table:

LAST_NAME	DEPARTMENT_NUMBER
appu	111
dhruv	333
priya	111

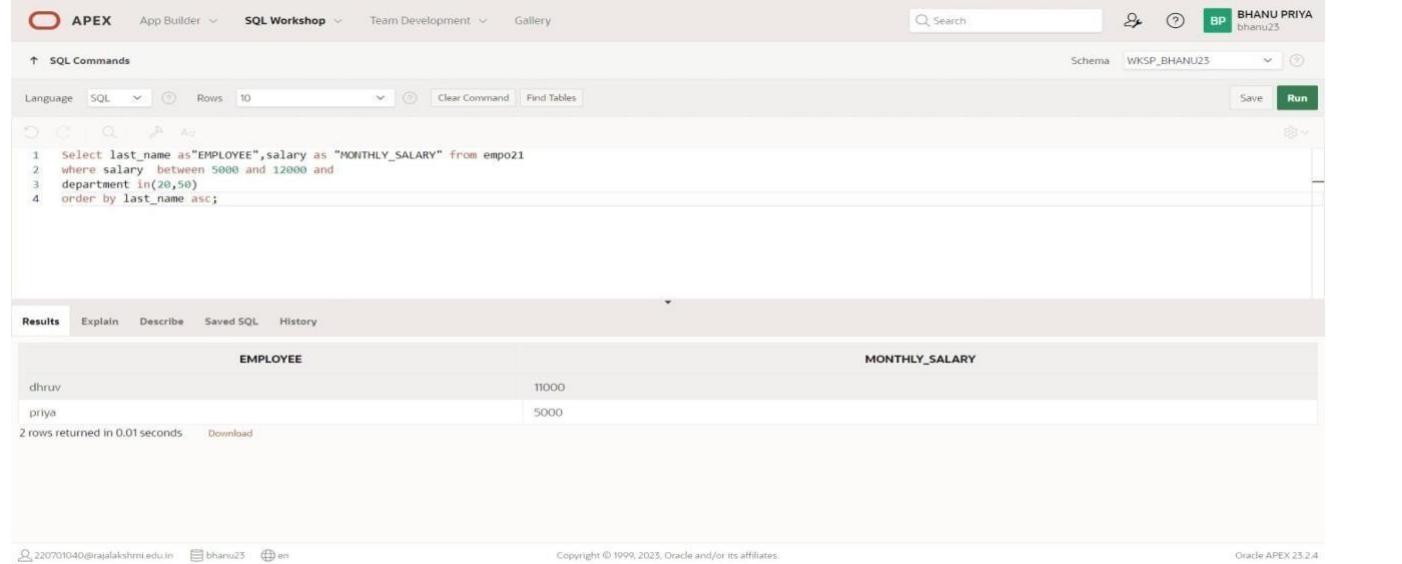
Text at the bottom of the results pane indicates "3 rows returned in 0.00 seconds". The footer of the page shows the user's email (220701040@rajalakshmi.edu.in), the session ID (bhangu23), and the copyright notice "Copyright © 1999, 2023, Oracle and/or its affiliates".

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 empo21 where salary between 5000 and 12000 and department in(20,50) order by Name_emp asc;
```

#### 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. On the right, a user profile for 'BHANU PRIYA' is shown with the schema 'WKSP\_BHANU23'. The main area displays the SQL command:

```
1 select last_name as "EMPLOYEE", salary as "MONTHLY_SALARY" from empo21
2 where salary between 5000 and 12000 and
3 department in(20,50)
4 order by last_name asc;
```

Below the command, the results are presented in a table:

EMPLOYEE	MONTHLY_SALARY
dhruv	11000
priya	5000

Text at the bottom of the results pane indicates "2 rows returned in 0.01 seconds". The footer of the page shows the user's email (220701040@rajalakshmi.edu.in), the session ID (bhangu23), and the copyright notice "Copyright © 1999, 2023, Oracle and/or its affiliates".

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 empo21 WHERE hire_date like '%94';
```

### 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 the user 'BHANDU PRIYA bhanu25'. The main area has tabs for 'SQL Commands' and 'Results'. The SQL command entered is:

```
1 SELECT last_name, hire_date
2 FROM empo21
3 WHERE hire_date like '%94';
```

The results table has columns 'LAST\_NAME' and 'HIRE\_DATE'. The data returned is:

LAST_NAME	HIRE_DATE
appu	02/21/1994
joe	04/22/1994

Below the table, it says '2 rows returned in 0.01 seconds' and has a 'Download' link. The bottom footer includes the URL '220701040@rajalakshmi.edu.in', session 'bhanu25', and locale 'en'. Copyright information 'Copyright © 1999, 2023, Oracle and/or its affiliates.' and the version 'Oracle APEX 23.2.4' are also present.

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_id FROM empo21 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 the user 'BHANDU PRIYA bhanu25'. The main area has tabs for 'SQL Commands' and 'Results'. The SQL command entered is:

```
1 SELECT last_name, job_id
2 FROM empo21
3 WHERE manager_id IS NULL;
```

The results table has columns 'LAST\_NAME' and 'JOB\_ID'. The data returned is:

LAST_NAME	JOB_ID
joe	8978
dhruv	9101

Below the table, it says '2 rows returned in 0.01 seconds' and has a 'Download' link. The bottom footer includes the URL '220701040@rajalakshmi.edu.in', session 'bhanu25', and locale 'en'. Copyright information 'Copyright © 1999, 2023, Oracle and/or its affiliates.' and the version 'Oracle APEX 23.2.4' are also present.

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_pct FROM empo21 WHERE commission_pct IS NOT NULL ORDER BY salary DESC, commission_pct 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'. On the right, there's a user profile for 'Bhanu Priya' (bhanu25). The main area has tabs for 'SQL Commands' and 'Results'. Under 'SQL Commands', the following SQL code is entered:

```
1 SELECT last_name, salary, commission_pct
2 FROM empo21
3 WHERE commission_pct IS NOT NULL
4 ORDER BY salary DESC, commission_pct DESC;
```

The 'Results' tab displays the output:

LAST_NAME	SALARY	COMMISSION_PCT
priya	5000	33
appu	1212	11
joe	200	22

Below the table, it says '3 rows returned in 0.01 seconds' and has a 'Download' link.

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

#### QUERY:

```
SELECT last_name, hire_date FROM empo21 WHERE hire_date like '%94';
```

#### 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'. On the right, there's a user profile for 'Bhanu Priya' (bhanu25). The main area has tabs for 'SQL Commands' and 'Results'. Under 'SQL Commands', the following SQL code is entered:

```
1 SELECT last_name
2 FROM empo21
3 WHERE last_name like '_a%';
```

The 'Results' tab displays the output:

LAST_NAME
prayae
dhauvae

Below the table, it says '2 rows returned in 0.01 seconds' and has a 'Download' link.

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 empo21 WHERE last_name LIKE '%a%' AND last_name LIKE '%e%';
```

#### 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 'Bhanu Priya' (bhanu23). The main area has tabs for 'SQL Commands' and 'Results'. The SQL command entered is:

```
1 SELECT last_name
2 FROM empo21
3 WHERE last_name LIKE '%a%'
4 AND last_name LIKE '%e%';
```

The results section displays the output:

LAST_NAME
priyae
dhruvae

2 rows returned in 0.00 seconds. The bottom of the screen shows copyright information and the version 'Oracle APEX 23.2.4'.

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_id,salary from empo21 where job_id in ('sales representative','stock clerk') and salary not in(2500,3500,7000);
```

#### 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 'Bhanu Priya' (bhanu23). The main area has tabs for 'SQL Commands' and 'Results'. The SQL command entered is:

```
1 select last_name,job_id,salary from empo21 where job_id in ('sales representative','stock clerk') and salary not in(2500,3500,7000);
```

The results section displays the output:

no data found

At the bottom, the copyright notice reads 'Copyright © 1999, 2023, Oracle and/or its affiliates.' and the version 'Oracle APEX 23.2.4'.

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_pct from employees where commission_pct=0.2;
```

### 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. On the right, there's a search bar, user information for 'Bhanu Priya' (bhanu23), and a 'Run' button. The main workspace is titled 'SQL Commands' and contains the following SQL code:

```
1 select last_name,salary,commission_pct from empo21 where commission_pct=0.2;
2
```

Below the code, the 'Results' tab is selected, showing the message 'no data found'. At the bottom of the page, there are footer links for user details (220701040@rajalakshmi.edu.in, bhanu23, en), copyright information (Copyright © 1999, 2023, Oracle and/or its affiliates.), and the software version (Oracle APEX 23.2.4).

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

**RESULT:**

# SINGLE ROW FUNCTIONS

**EX\_NO:6**

**DATE:**

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

**QUERY:**

```
select sysdate from dual;
```

**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 'BHUANU PRIYA bhanu23'. The main area has a search bar and a schema dropdown set to 'WKSP\_BHANU23'. Below is a toolbar with icons for undo, redo, search, and run. The SQL command input field contains: '1 select sysdate from dual;'. The results section shows the output: 'SYSDATE' followed by the date '03/12/2024'. It also indicates '1 rows returned in 0.02 seconds' and a 'Download' link. The bottom footer includes copyright information and the version 'Oracle APEX 23.2.4'.

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

**QUERY:**

```
select employee_id, last_name, salary, salary+(15.5/100*salary) "new_salary" from empo21;
```

**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 'BHUANU PRIYA bhanu23'. The main area has a search bar and a schema dropdown set to 'WKSP\_BHANU23'. Below is a toolbar with icons for undo, redo, search, and run. The SQL command input field contains: '1 select employee\_id, last\_name, salary, salary+(15.5/100\*salary) "new\_salary" from empo21;'. The results section displays a table with columns 'EMP\_ID', 'LAST\_NAME', 'SALARY', and 'new\_salary'. The data is as follows:

EMP_ID	LAST_NAME	SALARY	new_salary
12	appu	1212	1399.86
4	joe	200	231
1	prayae	5000	5775
3	dhauvae	11000	12705

It also indicates '4 rows returned in 0.03 seconds' and a 'Download' link. The bottom footer includes copyright information and the version 'Oracle APEX 23.2.4'.

**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 employee_id, last_name, salary, salary+(15.5/100*salary) "new_salary", new_salary-salary as "Increase" from empo21;
```

### OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The query has been run, and the results are displayed in a table format. The table has columns: EMP\_ID, LAST\_NAME, SALARY, new\_salary, and Increase. The data is as follows:

EMP_ID	LAST_NAME	SALARY	new_salary	Increase
12	appu	1212	1399.86	187.86
4	joe	200	231	31
1	prayoe	5000	5775	775
3	dhauvae	11000	12705	1705

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

**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 empo21 whose name starts with the letters J, A, or M. Give each column an appropriate label. Sort the results by the empo21' last names.

### QUERY:

```
select initcap(last_name),length(last_name) as "Length_of_last_name" from empo21 where last_name like 'J%' or last_name like 'A%' or last_name like 'M%' order by last_name asc;
```

### OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The query has been run, and the results are displayed in a table format. The table has columns: INITCAP(LAST\_NAME) and Length\_of\_last\_name. The data is as follows:

INITCAP(LAST_NAME)	Length_of_last_name
Joe	3

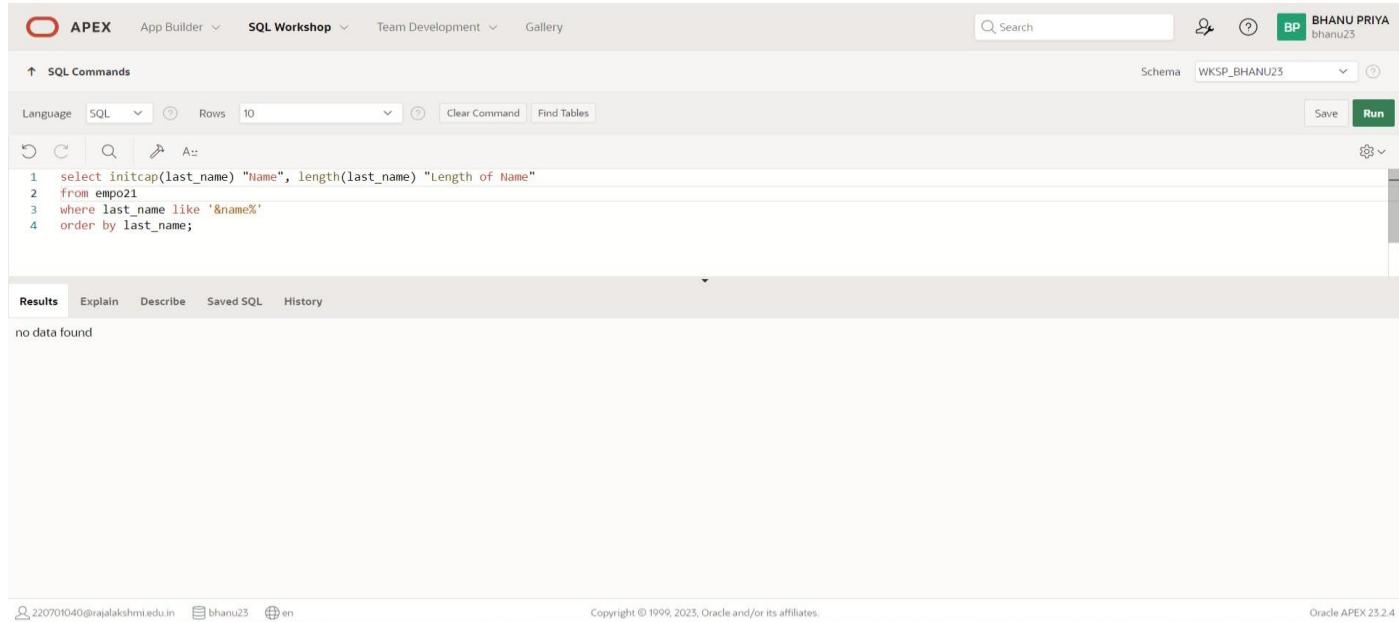
1 rows returned in 0.01 seconds [Download](#)

**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 empo21 whose last name starts with the letter H.

### QUERY:

```
select initcap(last_name),length(last_name) as "Length_of_last_name" from empo21 where last_name like 'J%' or last_name like 'A%' or last_name like 'M%' 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'. On the right, it shows the schema 'WKSP\_BHANU23' and a user icon 'BP BHANU PRIYA bhanu23'. The main area has tabs for 'SQL Commands' and 'Results'. Under 'SQL Commands', the query is displayed:

```
1 select initcap(last_name) "Name", length(last_name) "Length of Name"
2 from empo21
3 where last_name like '&name%'
4 order by last_name;
```

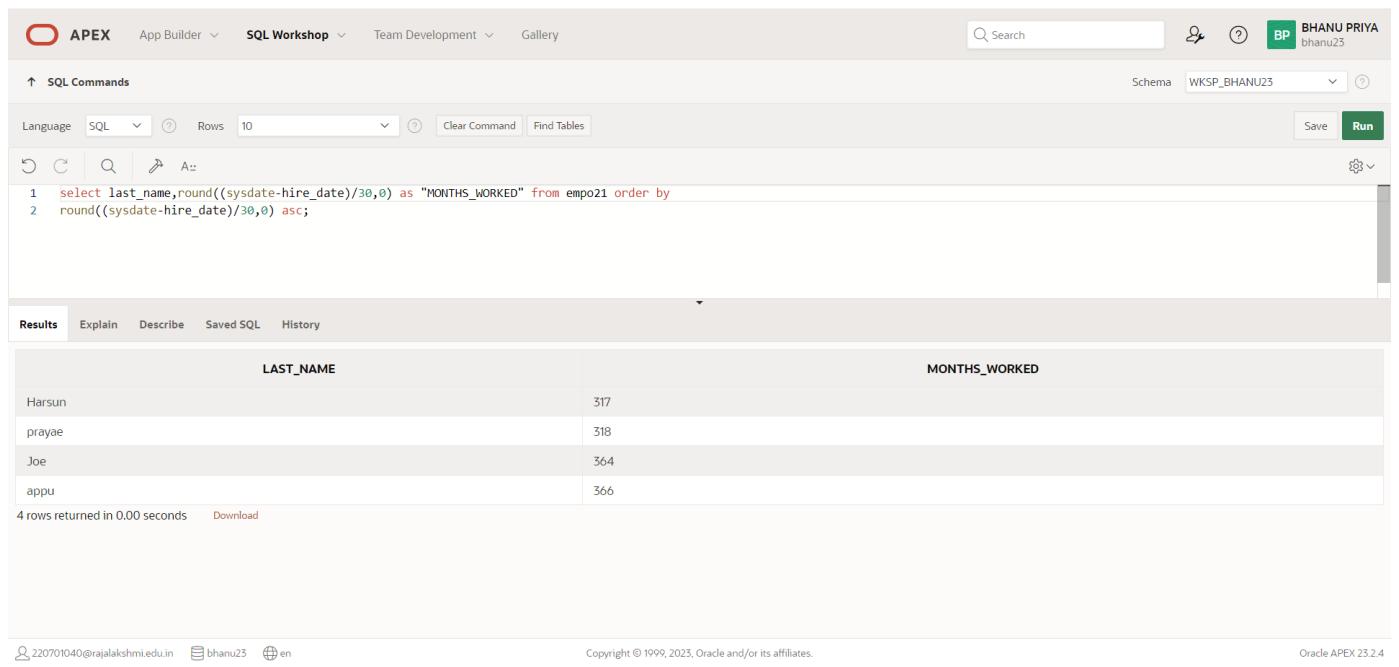
The 'Results' tab is selected, showing the message 'no data found'.

**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((sysdate-hire_date)/30,0) as "MONTHS_WORKED" from empo21 order by round((sysdate-hire_date)/30,0) asc;
```

### 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'. On the right, it shows the schema 'WKSP\_BHANU23' and a user icon 'BP BHANU PRIYA bhanu23'. The main area has tabs for 'SQL Commands' and 'Results'. Under 'SQL Commands', the query is displayed:

```
1 select last_name,round((sysdate-hire_date)/30,0) as "MONTHS_WORKED" from empo21 order by
2 round((sysdate-hire_date)/30,0) asc;
```

The 'Results' tab is selected, displaying a table with two columns: 'LAST\_NAME' and 'MONTHS\_WORKED'. The data is as follows:

LAST_NAME	MONTHS_WORKED
Harsun	317
prayae	318
Joe	364
appu	366

At the bottom left, it says '4 rows returned in 0.00 seconds'. At the bottom right, it says 'Download'.

**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 as "DREAM_SALARIES" from empo21;
```

**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 'Bhanu Priya' with schema 'WKSP\_BHANU23'. The SQL Commands tab is active, displaying the query: 'select last\_name||' earns'||salary||' monthly but wants'||salary\*3 as "DREAM\_SALARIES" from empo21;'. The Results tab shows the output:

DREAM_SALARIES	
appu	earns 1212 monthly but wants 3636
Joe	earns 200 monthly but wants 600
prayae	earns 5000 monthly but wants 15000
Harsun	earns 11000 monthly but wants 33000

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

At the bottom, the footer includes the URL '220701040@rajalakshmi.edu.in', session ID 'bhanu23', language 'en', copyright notice 'Copyright © 1999, 2023, Oracle and/or its affiliates.', and 'Oracle APEX 23.2.4'.

**8.** Create a query to display the last name and salary for all empo21. Format the salary to be 15 characters long, left-padded with the \$ symbol. Label the column SALARY.

**QUERY:**

```
select last_name, lpad(salary, 15, '$') as "SALARY" from empo21;
```

**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 'Bhanu Priya' with schema 'WKSP\_BHANU23'. The SQL Commands tab is active, displaying the query: 'select last\_name, lpad(salary, 15, '\$') as "SALARY" from empo21;'. The Results tab shows the output:

LAST_NAME	SALARY
appu	\$\$\$\$\$\$\$\$\$\$1212
Joe	\$\$\$\$\$\$\$\$\$\$200
prayae	\$\$\$\$\$\$\$\$\$\$5000
Harsun	\$\$\$\$\$\$\$\$\$\$11000

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

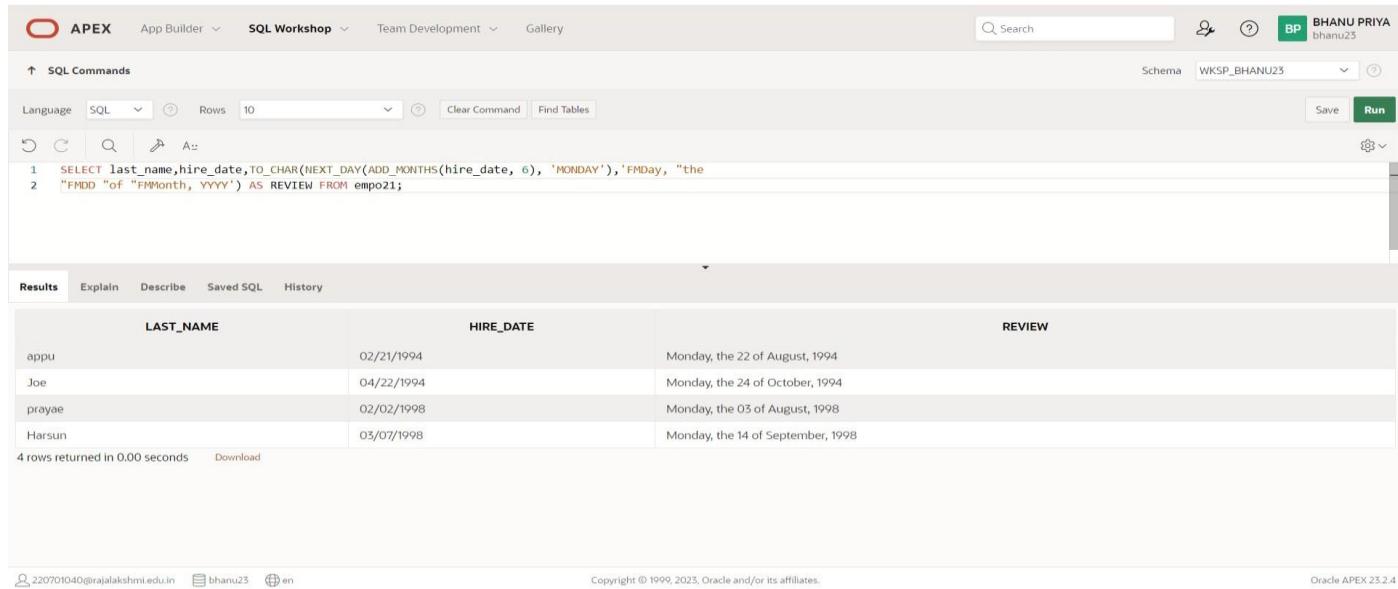
At the bottom, the footer includes the URL '220701040@rajalakshmi.edu.in', session ID 'bhanu23', language 'en', copyright notice 'Copyright © 1999, 2023, Oracle and/or its affiliates.', and 'Oracle APEX 23.2.4'.

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(ADD_MONTHS(hire_date, 6),  
'MONDAY'),'FMDD "the "FMMonth, YYYY') AS REVIEW FROM empo21;
```

### 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'. On the right, there is a user profile for 'Bhanu Priya' (bhanu23) and a schema dropdown set to 'WKSP\_BHANU23'. The main area has tabs for 'SQL Commands' and 'Results'. Under 'SQL Commands', the query is displayed:

```
1 SELECT last_name,hire_date,TO_CHAR(NEXT_DAY(ADD_MONTHS(hire_date, 6),  
2 'MONDAY'),'FMDD "the "FMMonth, YYYY') AS REVIEW FROM empo21;
```

Under 'Results', the output is shown in a table:

LAST_NAME	HIRE_DATE	REVIEW
appu	02/21/1994	Monday, the 22 of August, 1994
Joe	04/22/1994	Monday, the 24 of October, 1994
prayae	02/02/1998	Monday, the 03 of August, 1998
Harsun	03/07/1998	Monday, the 14 of September, 1998

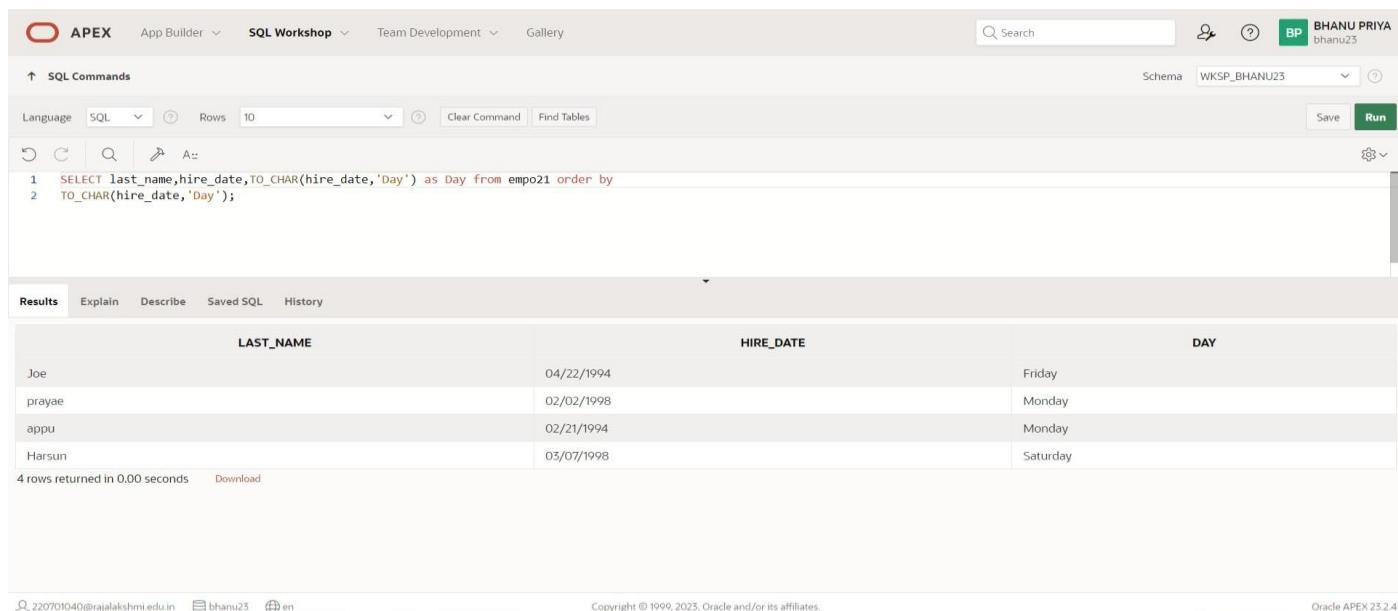
Below the table, it says '4 rows returned in 0.00 seconds'.

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') as Day from empo21 order by  
TO_CHAR(hire_date,'Day');
```

### 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'. On the right, there is a user profile for 'Bhanu Priya' (bhanu23) and a schema dropdown set to 'WKSP\_BHANU23'. The main area has tabs for 'SQL Commands' and 'Results'. Under 'SQL Commands', the query is displayed:

```
1 SELECT last_name,hire_date,TO_CHAR(hire_date,'Day') as Day from empo21 order by  
2 TO_CHAR(hire_date,'Day');
```

Under 'Results', the output is shown in a table:

LAST_NAME	HIRE_DATE	DAY
Joe	04/22/1994	Friday
prayae	02/02/1998	Monday
appu	02/21/1994	Monday
Harsun	03/07/1998	Saturday

Below the table, it says '4 rows returned in 0.00 seconds'.

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 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 'Bhanu Priya' (bhanu23). The main area has tabs for 'SQL Commands' and 'Results'. The SQL command entered is:

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

The results section displays the output:

LAST_NAME	DEPARTMENT_NUMBER	DEPT_ID
Joe	80	80

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

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 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 'Bhanu Priya' (bhanu23). The main area has tabs for 'SQL Commands' and 'Results'. The SQL command entered is:

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

The results section displays the output:

JOB_ID	LOC_ID
8978	49

Below the table, it says '1 rows returned in 0.03 seconds' and provides a 'Download' link. The bottom footer includes copyright information and the version 'Oracle APEX 23.2.4'.

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:**

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. On the right, there's a search bar, a user icon for 'BHANU PRIYA bhanu23', and a schema dropdown set to 'WKSP\_BHANU23'. The main workspace is titled 'SQL Commands' and contains the following SQL query:

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

Below the query, the 'Results' tab is selected, showing the output:

JOB_ID	LOC_ID
8978	49

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

At the bottom, the footer includes the URL '220701040@rajalakshmi.edu.in', the session identifier 'bhanu23', and the language 'en'. It also states 'Copyright © 1999, 2023, Oracle and/or its affiliates.' and 'Oracle APEX 23.2.4'.

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:**

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. On the right, there's a search bar, a user icon for 'BHANU PRIYA bhanu23', and a schema dropdown set to 'WKSP\_BHANU23'. The main workspace is titled 'SQL Commands' and contains the following SQL query:

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

Below the query, the 'Results' tab is selected, showing the output:

JOB_ID	LOC_ID
8978	49

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

At the bottom, the footer includes the URL '220701040@rajalakshmi.edu.in', the session identifier 'bhanu23', and the language 'en'. It also states 'Copyright © 1999, 2023, Oracle and/or its affiliates.' and 'Oracle APEX 23.2.4'.

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 empo21 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 executed is:

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

The results table displays two columns: JOB\_ID and LOC\_ID. The data returned is:

JOB_ID	LOC_ID
8978	49

1 rows returned in 0.03 seconds

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 empo21  
m on (w.manager_id=m.emp_id);
```

**OUTPUT:**

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

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

The results table displays two columns: JOB\_ID and LOC\_ID. The data returned is:

JOB_ID	LOC_ID
8978	49

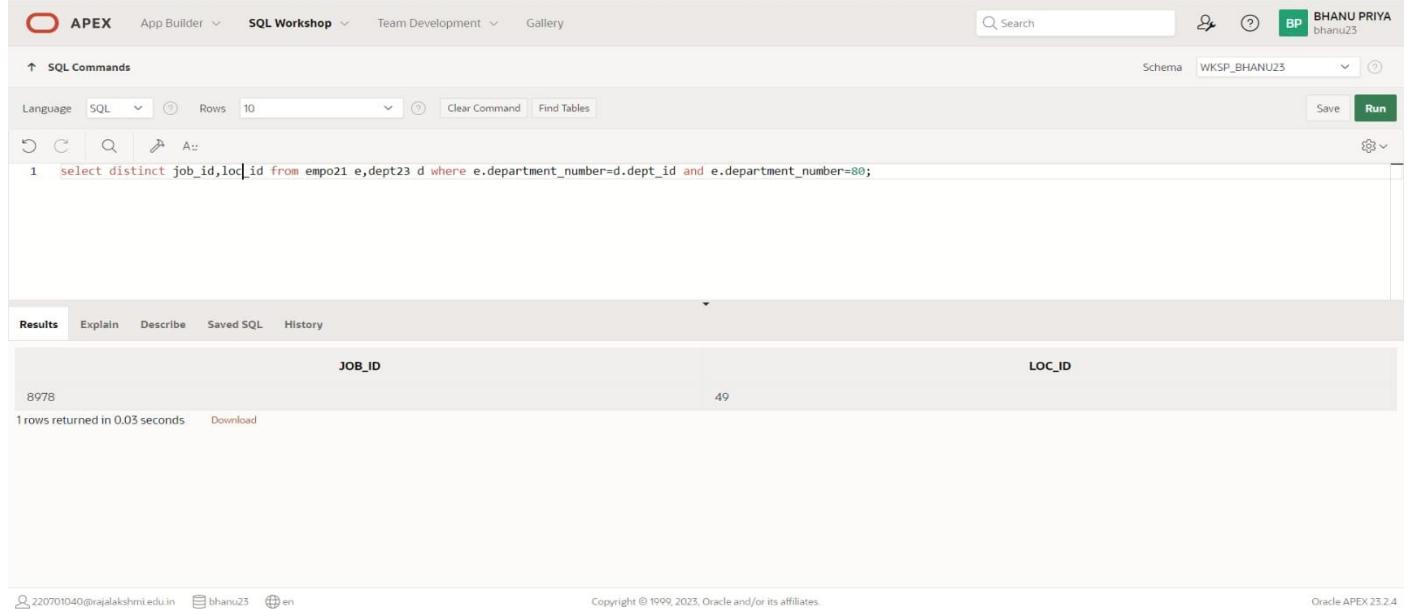
1 rows returned in 0.03 seconds

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 top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. The right side shows a user profile for 'Bhanu Priya' (bhanu25). The main area is titled 'SQL Commands' with tabs for Language (set to SQL), Rows (set to 10), and buttons for Clear Command and Find Tables. A search bar is at the top right. Below the tabs, there are icons for Undo, Redo, Search, and Run. The SQL command entered is:

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

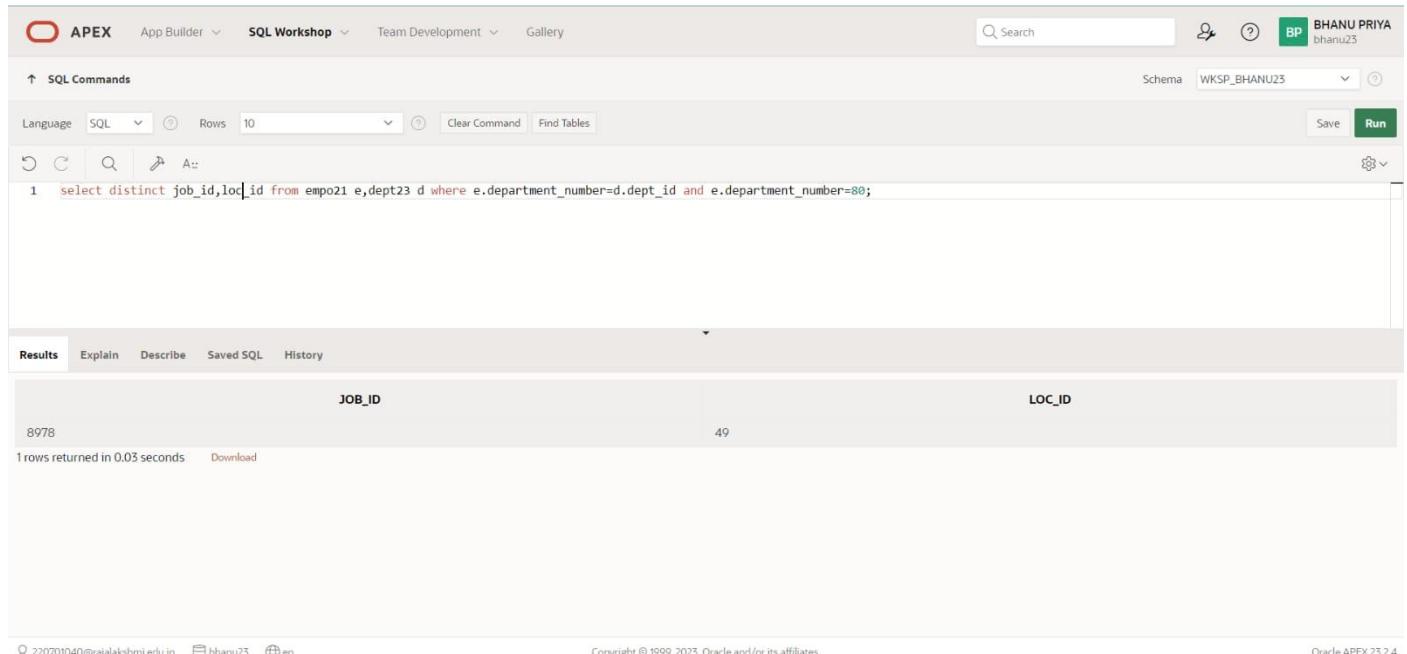
The results section shows a single row with columns 'JOB\_ID' and 'LOC\_ID'. The value for 'JOB\_ID' is 8978 and for 'LOC\_ID' is 49. Below the results, it says '1 rows returned in 0.03 seconds' and provides a 'Download' link. At the bottom of the page, there are footer links for 220701040@rajalakshmi.edu.in, bhanu25, and en, along with copyright information for Oracle and Oracle APEX 23.2.4.

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:**



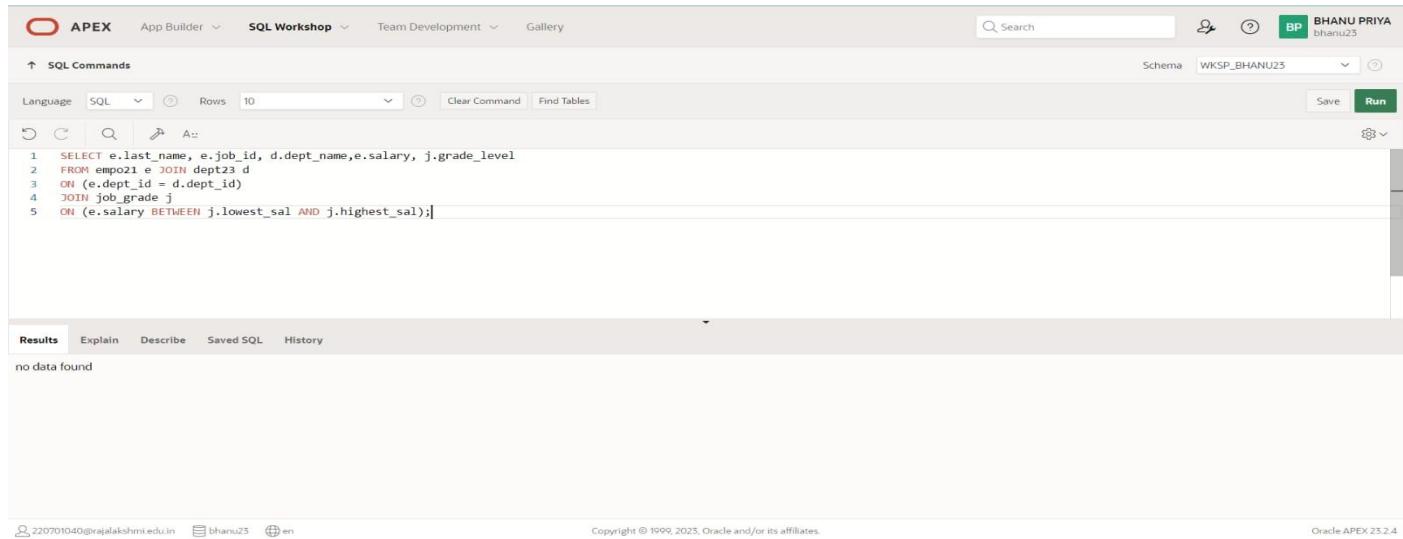
The screenshot shows the Oracle APEX SQL Workshop interface, identical to the previous one. The top navigation bar, user profile, and SQL command entry area are the same. The results section shows a single row with columns 'JOB\_ID' and 'LOC\_ID'. The value for 'JOB\_ID' is 8978 and for 'LOC\_ID' is 49. Below the results, it says '1 rows returned in 0.03 seconds' and provides a 'Download' link. At the bottom of the page, there are footer links for 220701040@rajalakshmi.edu.in, bhanu25, and en, along with copyright information for Oracle and Oracle APEX 23.2.4.

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 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 'Bhanu Priya' (bhanu23) and a 'Run' button. The main area has tabs for 'SQL Commands' and 'Results'. Under 'SQL Commands', the query is pasted. Under 'Results', it says 'no data found'. At the bottom, the footer includes the URL '220701040@rajalakshmi.edu.in', the schema 'bhanu23', and the copyright notice 'Copyright © 1999, 2023, Oracle and/or its affiliates. Oracle APEX 25.2.4'.

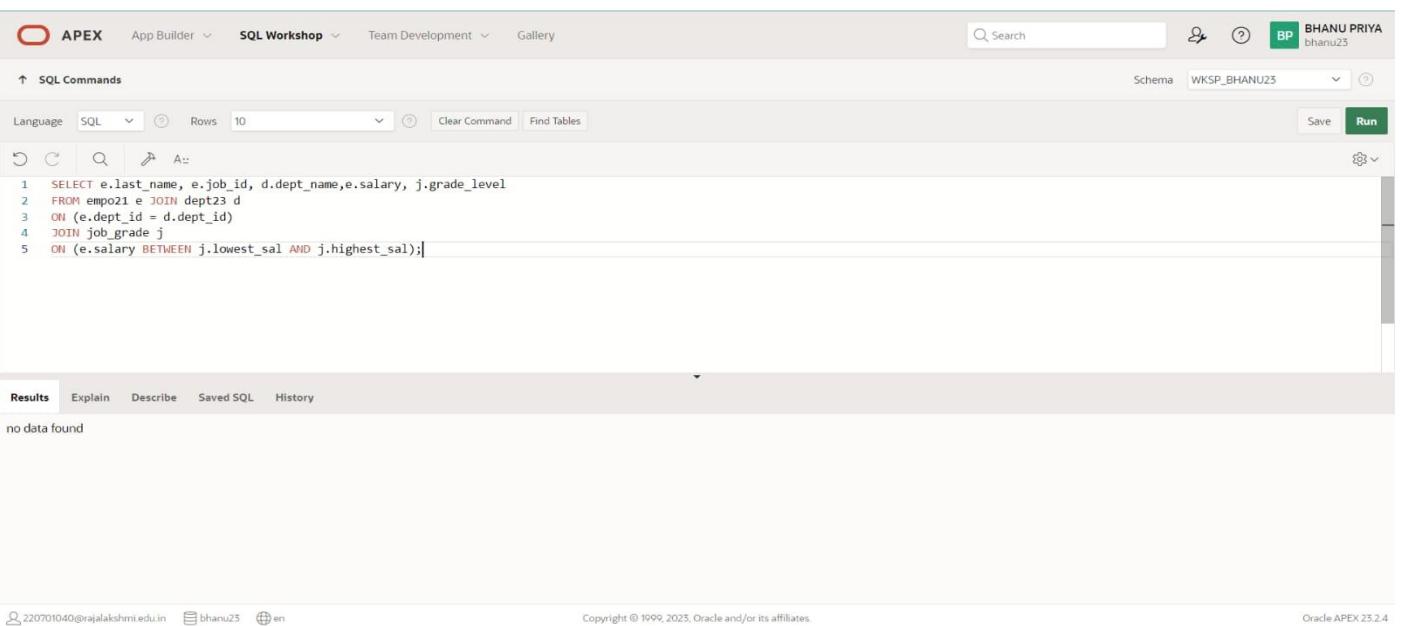
```
1 SELECT e.last_name, e.job_id, d.dept_name, e.salary, j.grade_level
2 FROM emp021 e JOIN dept23 d
3 ON (e.dept_id = d.dept_id)
4 JOIN job_grade j
5 ON (e.salary BETWEEN j.lowest_sal AND j.highest_sal);
```

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 SQL Workshop interface, identical to the previous one but with a different query. The top navigation bar, user profile, and footer are the same. The main area shows the new query under 'SQL Commands'. Under 'Results', it says 'no data found'. The footer includes the URL '220701040@rajalakshmi.edu.in', the schema 'bhanu23', and the copyright notice 'Copyright © 1999, 2023, Oracle and/or its affiliates. Oracle APEX 25.2.4'.

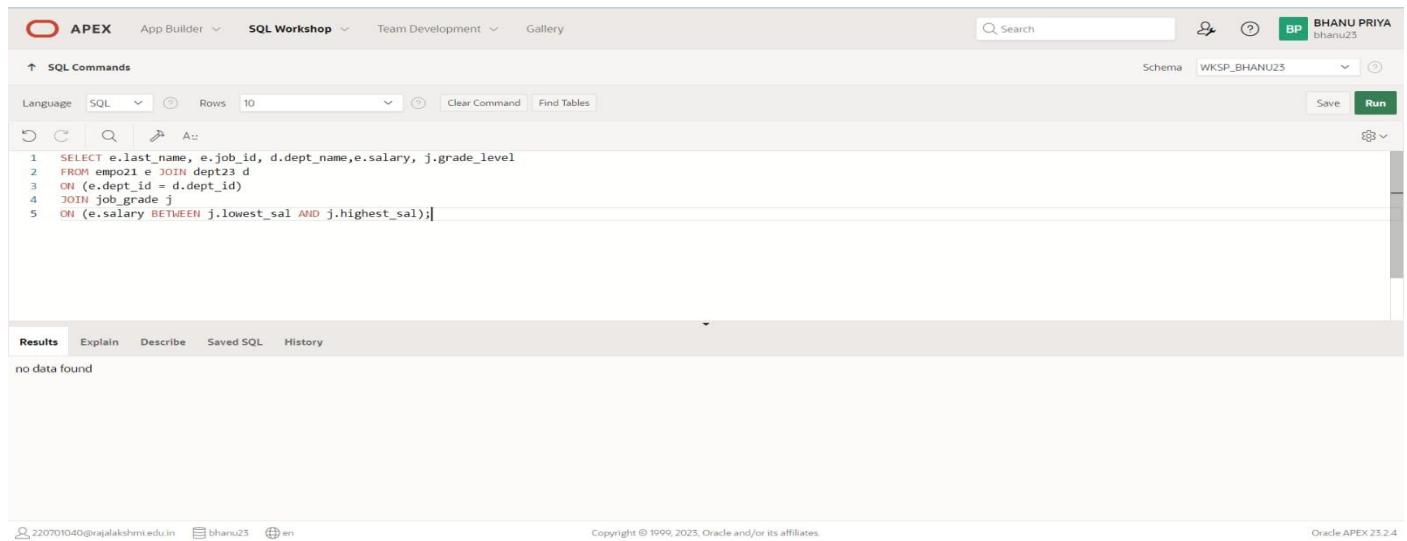
```
1 SELECT e.last_name, e.job_id, d.dept_name, e.salary, j.grade_level
2 FROM emp021 e JOIN dept23 d
3 ON (e.dept_id = d.dept_id)
4 JOIN job_grade j
5 ON (e.salary BETWEEN j.lowest_sal AND j.highest_sal);
```

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 APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows a user profile for 'Bhanu Priya' (bhanu23) and the schema 'WKSP\_BHANU23'. The main area is titled 'SQL Commands' with tabs for Language (SQL selected), Rows (10), Clear Command, and Find Tables. Below this is a code editor containing the following SQL query:

```
1 SELECT e.last_name, e.job_id, d.dept_name, e.salary, j.grade_level
2 FROM emp021 e JOIN dept23 d
3 ON (e.dept_id = d.dept_id)
4 JOIN job_grade j
5 ON (e.salary BETWEEN j.lowest_sal AND j.highest_sal);
```

Below the code editor, there are tabs for Results, Explain, Describe, Saved SQL, and History. The 'Results' tab is selected, showing the message 'no data found'. At the bottom of the page, footer information includes the URL '220701040@rajalakshmi.edu.in', session ID 'bhans23', language 'en', copyright notice 'Copyright © 1999, 2025, Oracle and/or its affiliates.', and the version 'Oracle APEX 25.2.4'.

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 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 'BHUU PRIYA bhanu23'. The main workspace has a 'SQL Commands' tab selected. The SQL editor contains the following query:

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

The results section displays the output of the query:

	Maximum	Minimum	sum	Average
90000	60000	300000	75000	

Below the results, it says '1 rows returned in 0.01 seconds' and provides a 'Download' link. The bottom of the page includes footer links for user information and copyright notice.

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 SQL Workshop interface. The query executed is:

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

The results are displayed in a table:

JOB_ID	MAXIMUM	MINIMUM	SUM	AVERAGE
44	70000	70000	70000	70000
65	90000	90000	90000	90000
46	60000	60000	60000	60000
47	80000	80000	80000	80000

4 rows returned in 0.01 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 SQL Workshop interface. The query executed is:

```
1 select job_id, count(*) from EMPB where job_id='47' group by job_id ;  
2  
3
```

The results are displayed in a table:

JOB_ID	COUNT(*)
47	1

1 rows returned in 0.01 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 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 'BP BHANU PRIYA bhanu23'. The schema is set to 'WKSP\_BHANU23'. The SQL Commands tab is active, displaying the following SQL code:

```
1 select max(salary)-min(salary) difference from empB;
2
3
```

The Results tab shows the output of the query:

DIFFERENCE
30000

1 rows returned in 0.00 seconds. The bottom of the screen displays copyright information for Oracle and the APEX version.

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 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 'BP BHANU PRIYA bhanu23'. The schema is set to 'WKSP\_BHANU23'. The SQL Commands tab is active, displaying the following SQL code:

```
1 select count(distinct manager_id )"Number of managers" from empB;
2
3
```

The Results tab shows the output of the query:

Number of managers
3

1 rows returned in 0.00 seconds. The bottom of the screen displays copyright information for Oracle and the APEX version.

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 APEX SQL Workshop interface. The query is executed and returns three rows of data:

MANAGER_ID	MIN(SALARY)
3	80000
6	70000
5	60000

3 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 APEX SQL Workshop interface. The query is executed and returns one row of data:

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:

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

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

The results are displayed in a matrix format:

job	Dept20	dept50	dept80	dept90	TOTAL
44	-	-	-	-	60000
65	-	-	-	-	90000
46	-	-	-	-	80000
47	-	-	70000	-	70000

4 rows returned in 0.01 seconds

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:

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

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

The results are displayed in a matrix format:

dept_name	department location	Number of people	salary
CS	CHENNAI	1	70000
IT	MUMBAI	1	90000
IT	TORANTO	1	80000
CS	BANGLORE	1	60000

4 rows returned in 0.07 seconds

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

**RESULT:**

# SUB QUERIES

**EX\_NO:9**

**DATE:**

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 department_id=(select department_id from employees where last_name='Janu') and last_name not in('Janu');
```

**OUTPUT:**

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

```
1 select last_name,hire_date from emp021 where dept_id=(select dept_id from emp021 where last_name='Janu') and last_name not in('Janu');
```

The results section displays the output:

LAST_NAME	HIRE_DATE
Jones	05/01/1998
Williams	02/19/1994

2 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 APEX SQL Workshop interface. The SQL command window contains the following code:

```
1 select emp_id,last_name,salary from emp021 where salary>(select avg(salary) from emp021) order by salary;
```

The results section displays the output:

EMP_ID	LAST_NAME	SALARY
2	Janu	60000
5	Brown	80000

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 department_id=(select department_id from employees where last_name like'%u%');
```

#### 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'. On the right, there's a user profile for 'Bhanu Priya' (bhanu25). The main area has tabs for 'SQL Commands' and 'Results'. Under 'SQL Commands', the query is displayed:

```
1 select employee_id, last_name from employees where department_id=(select department_id from employees where last_name like'%u%');
```

Under 'Results', the output is shown in a table:

EMPLOYEE_ID	LAST_NAME
2	Janu
4	Jones
3	Williams

Below the table, it says '3 rows returned in 0.03 seconds' and has a 'Download' link. At the bottom, it shows the URL 'https://apex.oracle.com/pls/apex/r/apex/workspace/home?session=10064742540...', copyright information 'Copyright © 1999, 2023, Oracle and/or its affiliates.', and 'Oracle APEX 23.2.4'.

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, department_id, job_id from employees where department_id=(select dept_id from departments where location_id=1700);
```

#### 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'. On the right, there's a user profile for 'Bhanu Priya' (bhanu25). The main area has tabs for 'SQL Commands' and 'Results'. Under 'SQL Commands', the query is displayed:

```
1 select last_name, dept_id, job_id from emp021 where dept_id=(select dept_id from location where location_id=1700);
```

Under 'Results', the output is shown in a table:

LAST_NAME	DEPT_ID	JOB_ID
Janu	80	102
davies	20	101
Jones	80	104
Brown	5	105
Williams	80	103

Below the table, it says '5 rows returned in 0.06 seconds' and has a 'Download' link. At the bottom, it shows the URL 'https://apex.oracle.com/pls/apex/r/apex/workspace/home?session=10064742540...', copyright information 'Copyright © 1999, 2023, Oracle and/or its affiliates.', and 'Oracle APEX 23.2.4'.

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_id=(select manager_id from employees where manager_name='King');
```

### OUTPUT:

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

```
1 select last_name,salary from employees where manager_id in(select manager_id from employees where manager_name='King'));
```

The results table has columns LAST\_NAME and SALARY, displaying data for three employees:

LAST_NAME	SALARY
davies	50000
Williams	20000
Jones	48000

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

### OUTPUT:

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

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

The results table has columns DEPARTMENT\_ID, LAST\_NAME, and JOB\_ID, displaying data for three employees:

DEPARTMENT_ID	LAST_NAME	JOB_ID
80	Janu	ac_account
80	Jones	ac_account
80	Williams	hr_rep

3 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 where last_name like '%u%');
```

### 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. On the right, there's a user profile for 'Bhanu Priya' (bhanu23). The main workspace is titled 'SQL Commands' and contains the following SQL code:

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

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

EMPLOYEE_ID	LAST_NAME	SALARY
5	Brown	80000

Text at the bottom of the results pane indicates "1 rows returned in 0.01 seconds" and provides a "Download" link. The footer of the page includes user information (220701040@rajalakshmi.edu.in, bhanu23, en), copyright notice (Copyright © 1999, 2023, Oracle and/or its affiliates.), and the version (Oracle APEX 23.2.4).

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

**RESULT:**

# USING THE SET OPERATORS

EX\_NO:10

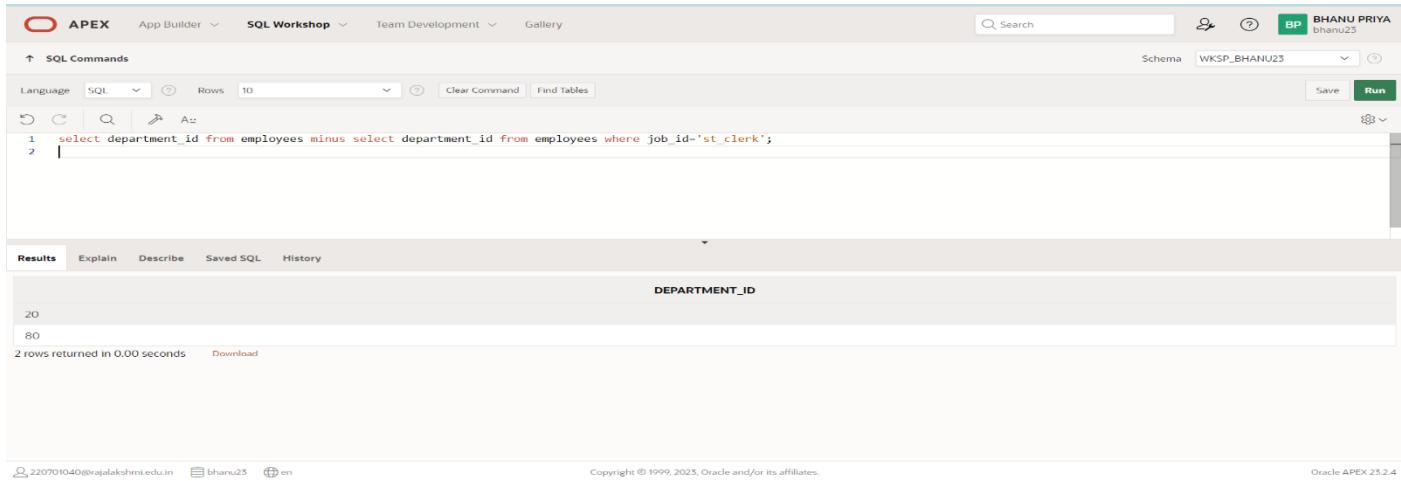
DATE:

- 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 department_id from employees minus select department_id from employees where job_id='st_clerk';
```

**OUTPUT:**



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

```
1 select department_id from employees minus select department_id from employees where job_id='st_clerk';
2 |
```

The results section displays the output:

DEPARTMENT_ID
20
80

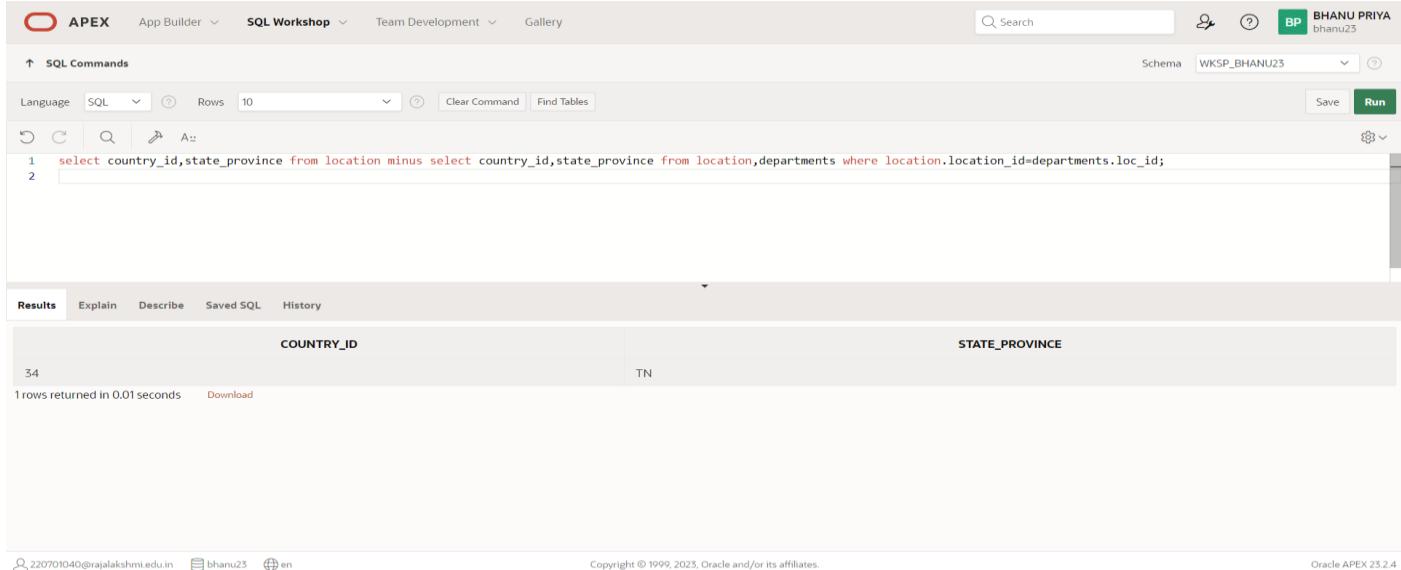
2 rows returned in 0.00 seconds

- 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,state_province from location minus select country_id,state_province from location,departments where location.location_id=departments.location_id;
```

**OUTPUT:**



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

```
1 select country_id,state_province from location minus select country_id,state_province from location,departments where location.location_id=departments.location_id;
2 |
```

The results section displays the output:

COUNTRY_ID	STATE_PROVINCE
34	TN

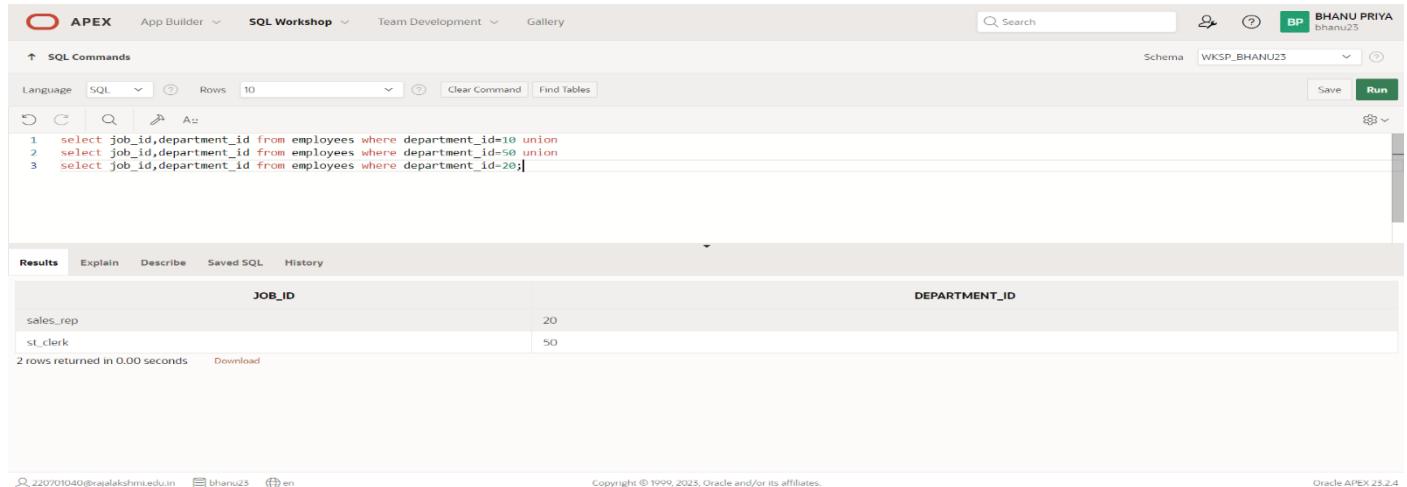
1 rows returned in 0.01 seconds

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 job_id,department_id from employees where department_id=10 union
select job_id,department_id from employees where department_id=50 union
select job_id,department_id from employees where department_id=20;
```

#### OUTPUT:



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

```
1 select job_id,department_id from employees where department_id=10 union
2 select job_id,department_id from employees where department_id=50 union
3 select job_id,department_id from employees where department_id=20;
```

The Results tab displays the output:

JOB_ID	DEPARTMENT_ID
sales_rep	20
st_clerk	50

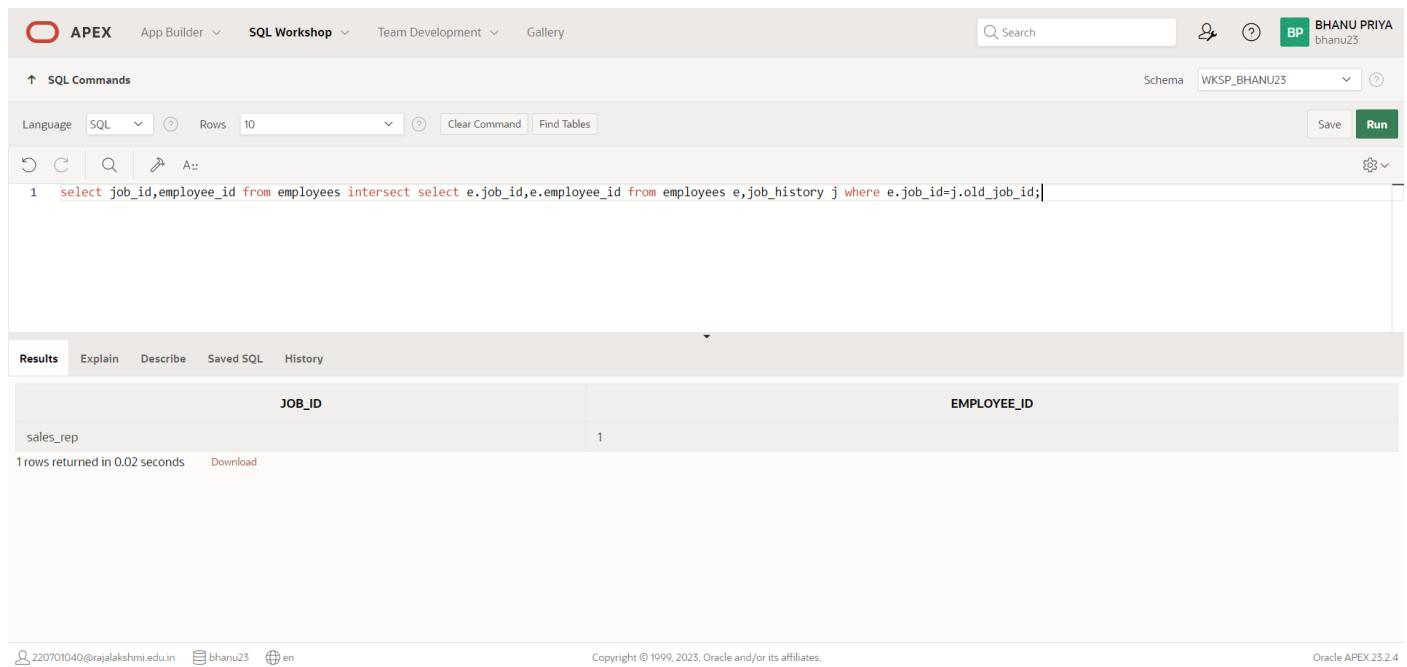
2 rows returned in 0.00 seconds

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 job_id,employee_id from employees intersect select e.job_id,e.employee_id from employees
e.job_history j where e.job_id=j.old_job_id;
```

#### OUTPUT:



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

```
1 select job_id,employee_id from employees intersect select e.job_id,e.employee_id from employees e,job_history j where e.job_id=j.old_job_id;
```

The Results tab displays the output:

JOB_ID	EMPLOYEE_ID
sales_rep	1

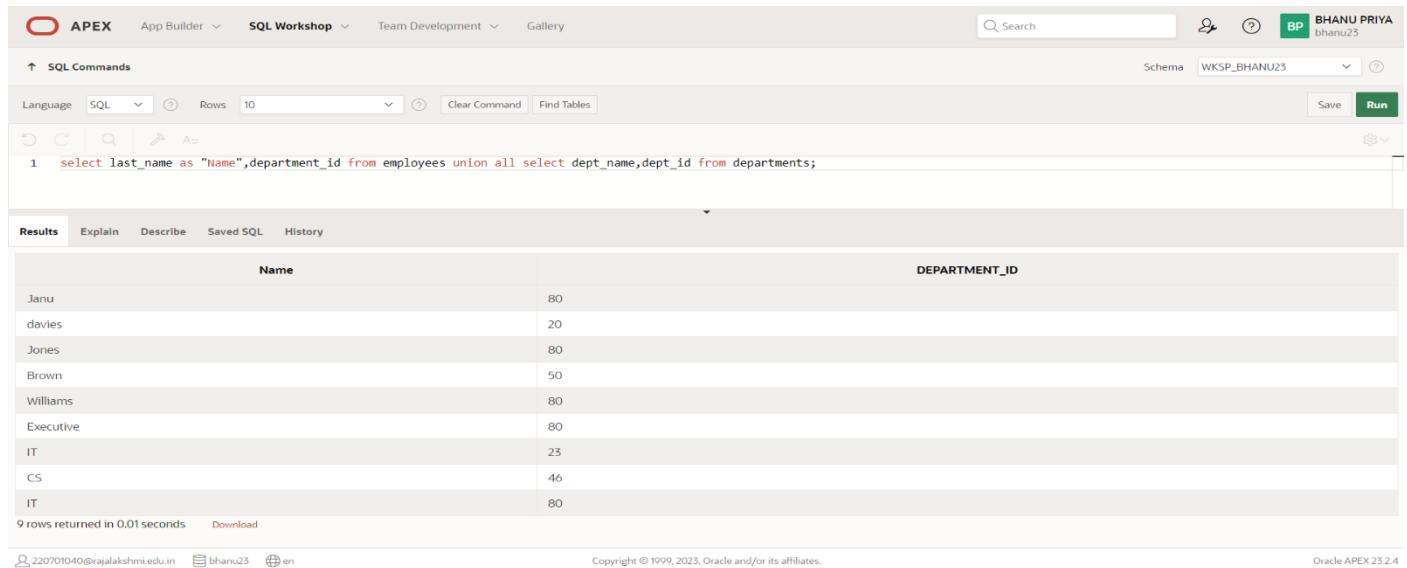
1 rows returned in 0.02 seconds

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 first_name||' '||last_name as "Name",department_id from employees union all select  
dept_name,dept_id from departments;
```

### 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. On the right, there's a search bar, a user icon for 'BHANU PRIYA bhanu23', and a 'Run' button. The main workspace is titled 'SQL Commands' and contains the following SQL code:

```
1  select last_name as "Name",department_id from employees union all select dept_name,dept_id from departments;
```

Below the code, the results are displayed in a table:

Name	DEPARTMENT_ID
Janu	80
davies	20
Jones	80
Brown	50
Williams	80
Executive	80
IT	23
CS	46
IT	80

At the bottom of the results pane, it says '9 rows returned in 0.01 seconds' and has a 'Download' link. The footer of the page includes copyright information: 'Copyright © 1999, 2023, Oracle and/or its affiliates.' and 'Oracle APEX 23.2.4'.

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

**RESULT:**

# CREATING VIEWS

**EX\_NO:11**

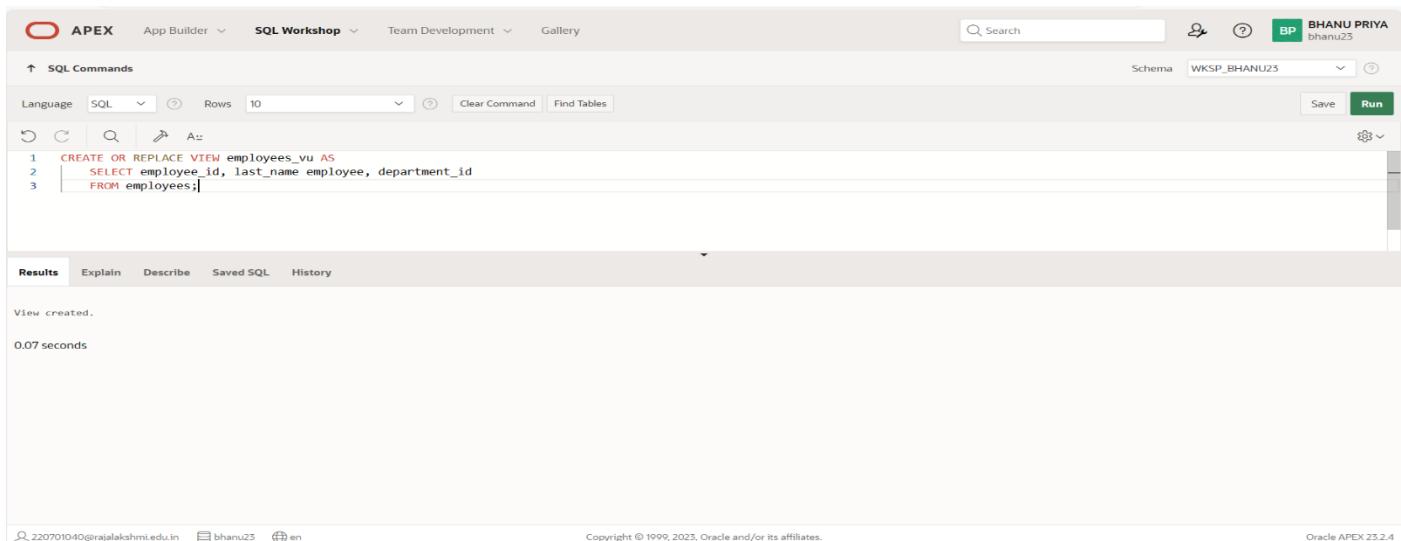
**DATE:**

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 OR REPLACE VIEW employees_vu AS SELECT employee_id, last_name employee,
department_id FROM employees;
```

## OUTPUT:



The screenshot shows the Oracle APEX SQL Workshop interface. In the top navigation bar, 'SQL Workshop' is selected. The main area displays the following SQL code:

```
1 CREATE OR REPLACE VIEW employees_vu AS
2   SELECT employee_id, last_name employee, department_id
3     FROM employees;
```

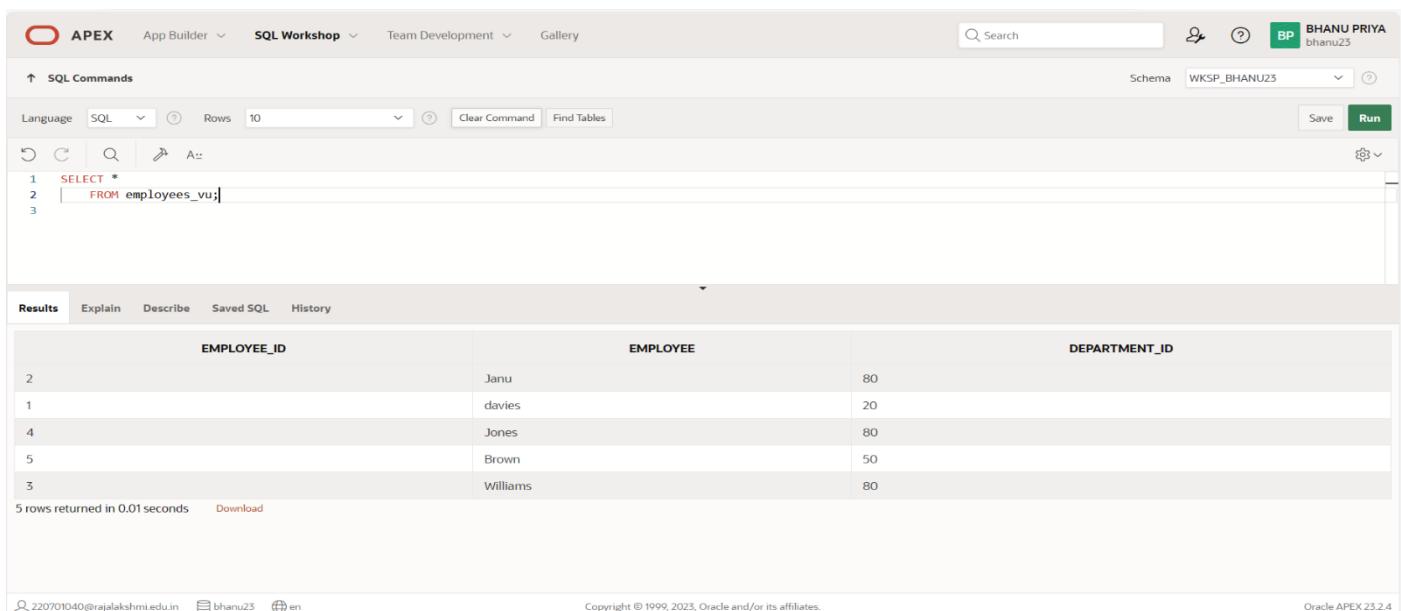
Below the code, the 'Results' tab is active, showing the message "View created." and a execution time of "0.07 seconds". The bottom status bar indicates the user is bhanu23 and the session is en.

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

## QUERY:

```
select * from employees_vu;
```

## OUTPUT:



The screenshot shows the Oracle APEX SQL Workshop interface with the 'Results' tab active. The results of the query are displayed in a table:

EMPLOYEE_ID	EMPLOYEE	DEPARTMENT_ID
2	Janu	80
1	davies	20
4	Jones	80
5	Brown	50
3	Williams	80

Below the table, it says "5 rows returned in 0.01 seconds". The bottom status bar indicates the user is bhanu23 and the session is en.

3.)Select the view name and text from the USER\_VIEWS data dictionary views

#### QUERY:

```
SELECT view_name, text FROM user_views;
```

#### OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The query `SELECT view_name, text FROM user_views;` has been run. The results are displayed in a table with two columns: `VIEW_NAME` and `TEXT`. The single row returned is for the view `EMPLOYEES_VU`, which contains the SQL statement `SELECT employee_id, last_name employee, department_id FROM employees`.

VIEW_NAME	TEXT
EMPLOYEES_VU	SELECT employee_id, last_name employee, department_id FROM employees

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

#### QUERY:

```
SELECT employee, department_id FROM employees_vu;
```

#### OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The query `SELECT employee, department_id FROM employees_vu;` has been run. The results are displayed in a table with two columns: `EMPLOYEE` and `DEPARTMENT_ID`. The data shows five employees and their corresponding department IDs.

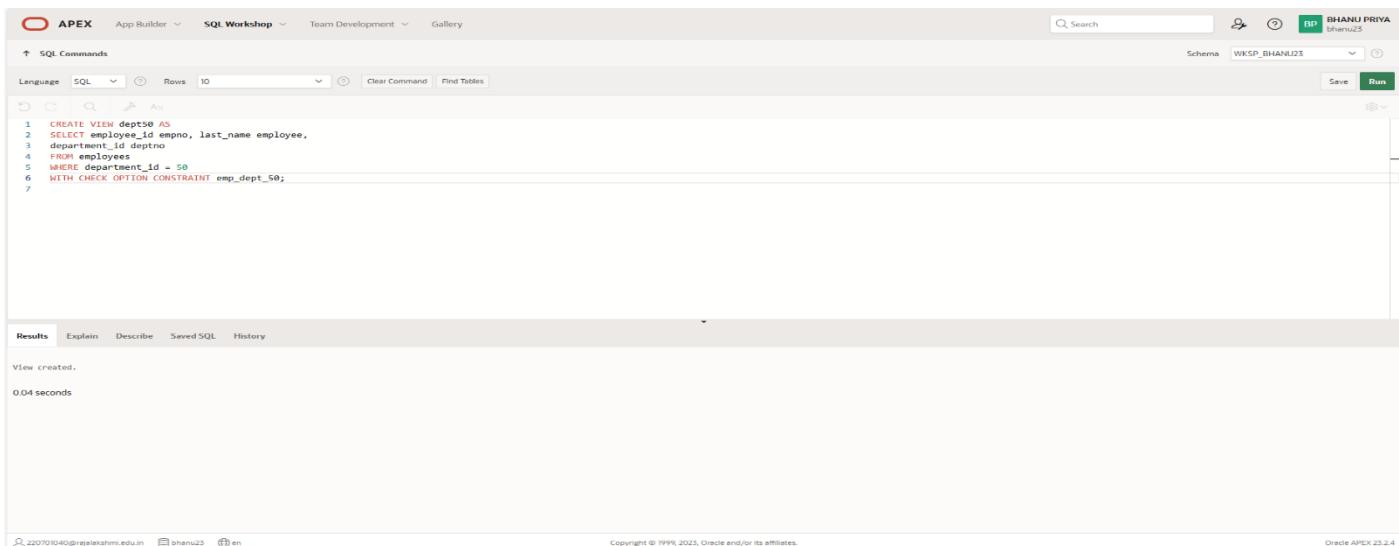
EMPLOYEE	DEPARTMENT_ID
Janu	80
davles	20
Jones	80
Brown	50
Williams	80

5.)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 employee_id empno, last_name employee, department_id deptno
FROM employees WHERE department_id = 50 WITH CHECK OPTION CONSTRAINT emp_dept_50;
```

### OUTPUT:



The screenshot shows the Oracle APEX SQL Workshop interface. In the top navigation bar, 'APEX' is selected. Below it, 'SQL Workshop' is chosen from the dropdown menu. The main area displays the SQL command for creating the view:

```
1 CREATE VIEW dept50 AS
2  SELECT employee_id empno, last_name employee,
3  department_id deptno
4  FROM employees
5  WHERE department_id = 50
6  WITH CHECK OPTION CONSTRAINT emp_dept_50;
7
```

Below the command, the results show:

View created.  
0.04 seconds

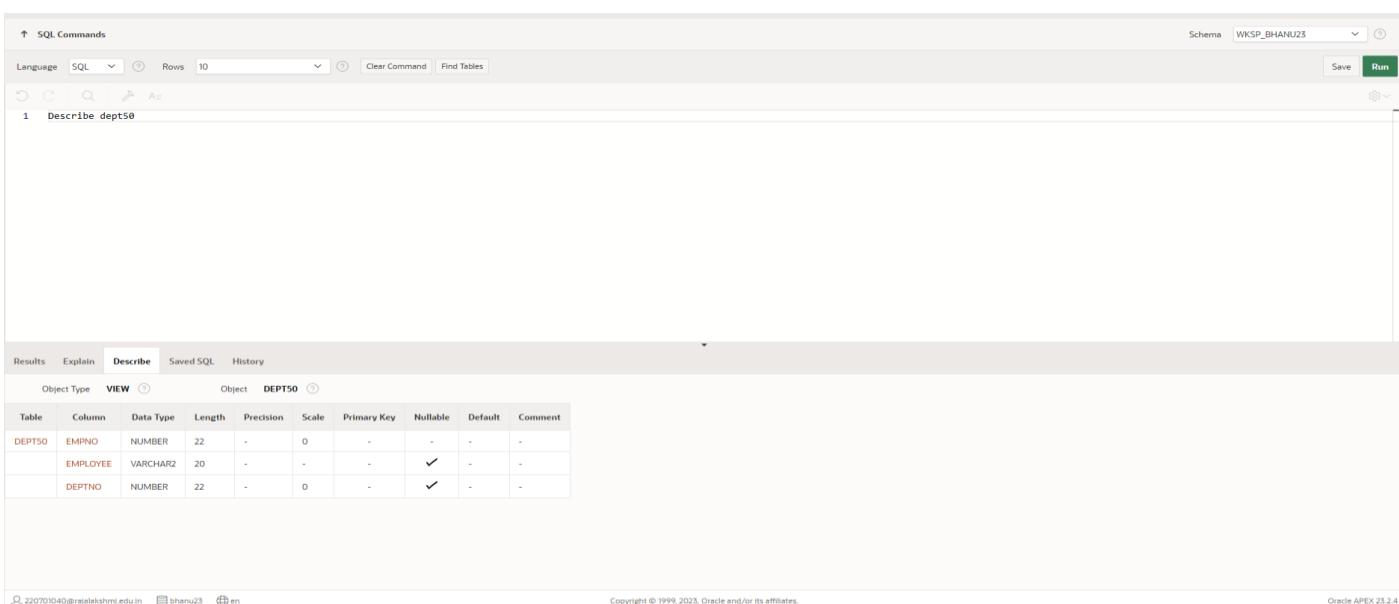
At the bottom, the status bar includes the session ID (220701040@rajalakshmi.edu.in), the schema (bhangu23), and the language (en). The copyright notice for Oracle (1999-2023) and the APEX version (23.2.4) are also present.

6.)Display the structure and contents of the DEPT50 view.

### QUERY:

```
Describe dept50;
```

### OUTPUT:



The screenshot shows the Oracle APEX SQL Workshop interface with the 'Describe' tab selected. The command entered is:

```
1 Describe dept50
```

The results table shows the structure of the DEPT50 view:

Object Type	VIEW	Object	DEPT50						
Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
DEPT50	EMPNO	NUMBER	22	-	0	-	-	-	-
	EMPLOYEE	VARCHAR2	20	-	-	-	✓	-	-
	DEPTNO	NUMBER	22	-	0	-	✓	-	-

At the bottom, the status bar includes the session ID (220701040@rajalakshmi.edu.in), the schema (bhangu23), and the language (en). The copyright notice for Oracle (1999-2023) and the APEX version (23.2.4) are also present.

7.) Attempt to reassign Matos to department 80

**QUERY:**

```
UPDATE dept50 SET deptno=80 WHERE employee='Matos';
```

**OUTPUT:**

The screenshot shows the Oracle APEX SQL Workshop interface. In the top navigation bar, 'APEX' is selected. Below it, 'SQL Workshop' is active. The main area contains the following SQL command:

```
1 UPDATE dept50
2 SET deptno=80
3 WHERE employee='Matos';
4
```

Below the command, the results section shows:

0 row(s) updated.  
0.05 seconds

At the bottom of the interface, the copyright notice reads "Copyright © 1999, 2023, Oracle and/or its affiliates." and "Oracle APEX 23.2.4".

8.) 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 or replace view salary_vu as select e.last_name "Employee",d.dept_name Department, e.salary "Salary",j.grade_level "Grades" from employees e,departments d,job_grade j where e.department_id=d.dept_id and e.salary between j.lowest_sal and j.highest_sal;
```

**OUTPUT:**

The screenshot shows the Oracle APEX SQL Workshop interface. In the top navigation bar, 'APEX' is selected. Below it, 'SQL Workshop' is active. The main area contains the following SQL command:

```
1 create or replace view salary_vu as
2 select e.last_name "Employee",d.dept_name "Department",e.salary "Salary",j.grade_level "Grades"
3 from employees e,departments d,job_grade j
4 where e.department_id=d.dept_id and e.salary between j.lowest_sal and j.highest_sal;
```

Below the command, the results section shows:

View created.  
0.04 seconds

At the bottom of the interface, the copyright notice reads "Copyright © 1999, 2023, Oracle and/or its affiliates." and "Oracle APEX 23.2.4".

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.

```
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

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

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 tablename 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 tablename 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	

# EXERCISE 13

## Creating Views

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';
```

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

Results	Explain	Describe	Saved SQL	History
ID	Song Title		ARTIST	
47	Hurrah for Today		The Jubilant Trio	
49	Lets Celebrate		The Celebrants	
2 rows returned in 0.00 seconds		<a href="#">Download</a>		

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';
```

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);
```

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);
```

# 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';
```

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;
```

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);
```

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;
```

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;
```

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';**

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;**

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

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

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 ASSELECT title, artistFROM  
copy_d_songs;SELECT * FROM view_copy_d_songs;
```

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**

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;
```

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_idFROM(SELECT dpt.department_id,  
MAX(NVL(emp.salary,0)) max_dpt_salFROM departments dpt LEFT OUTER JOIN employees emp ON  
dpt.department_id = emp.department_idGROUP BY dpt.department_id) dptmx LEFT OUTER JOIN  
employees empm ON dptmx.department_id = empm.department_idWHERE NVL(empm.salary,0) =  
dptmx.max_dpt_sal;
```

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

```
SELECT ROWNUM, last_name, salaryFROM(SELECT * FROM f_staffs ORDER BY SALARY);
```

# **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);**

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';
```

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';
```

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;
```

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\_idxON d\_partners(LOWER(last\_name));**

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;**

**SELECT \* FROM user\_synonyms WHERE table\_NAME = UPPER('d\_track\_listings');**

10. Drop the synonym that you created in question

**DROP SYNONYM dj\_tracks2;**

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

**RESULT:**

# OTHER DATABASE OBJECTS

**EX\_NO:14**

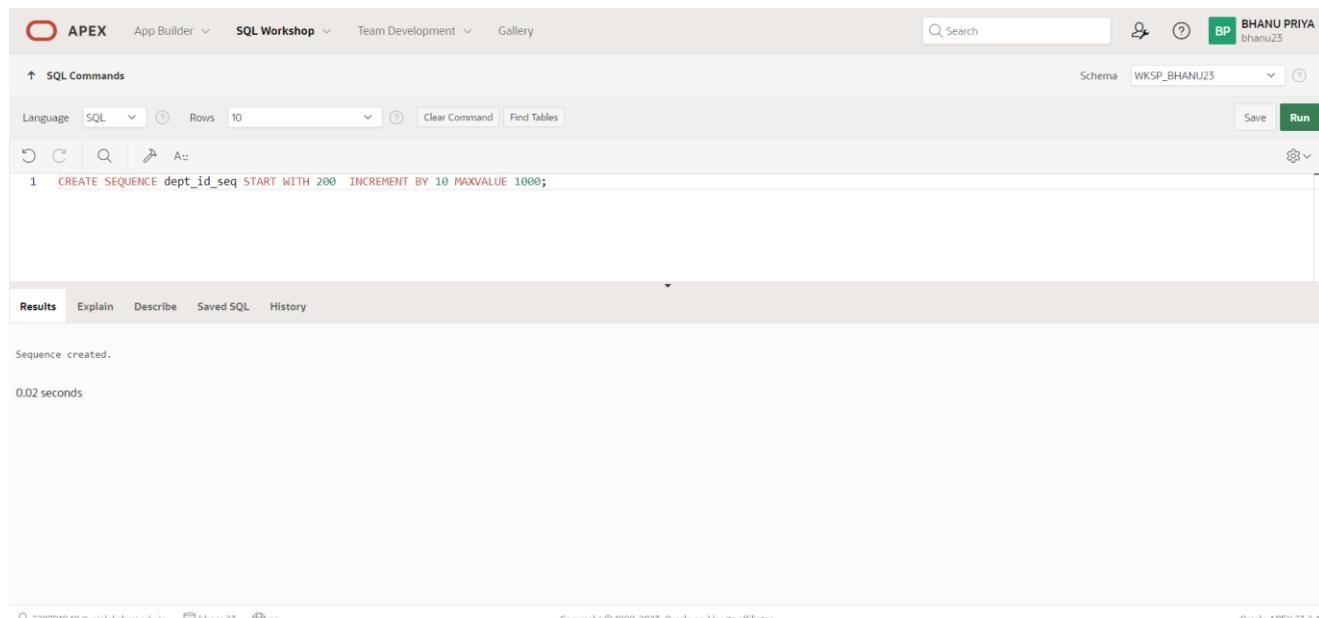
**DATE:**

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:**



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

```
1 CREATE SEQUENCE dept_id_seq START WITH 200 INCREMENT BY 10 MAXVALUE 1000;
```

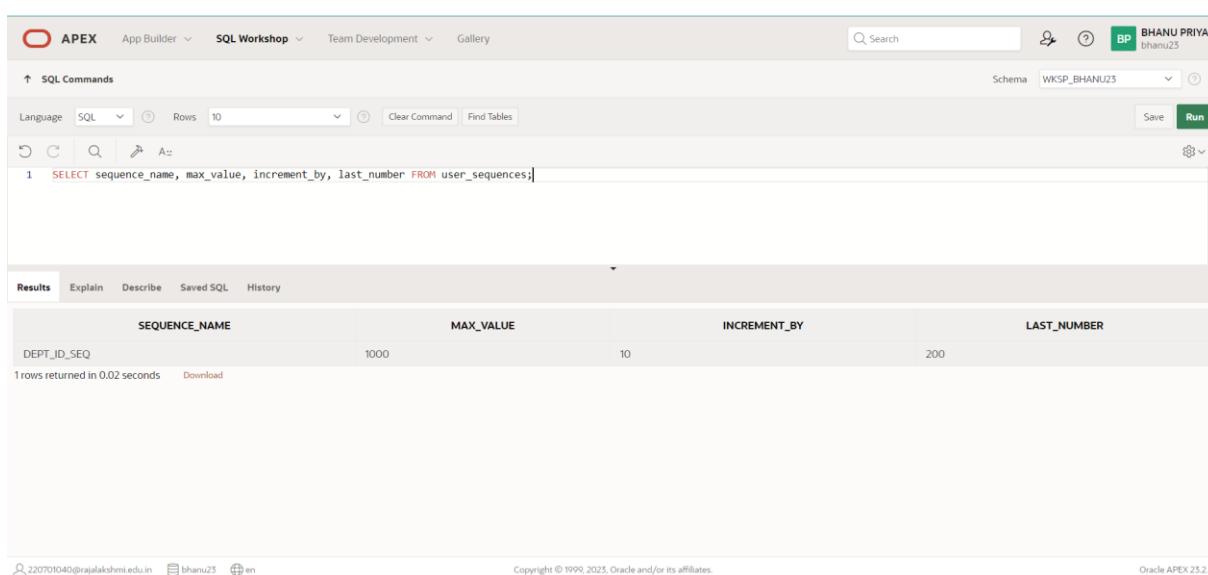
Below the command, the results pane displays the message "Sequence created." and "0.02 seconds". The bottom of the screen shows the URL "220701040@rajalakshmi.edu.in", the schema "bhanu23", and the copyright notice "Copyright © 1999, 2023, Oracle and/or its affiliates. Oracle APEX 23.2.4".

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:**



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

```
1 SELECT sequence_name, max_value, increment_by, last_number FROM user_sequences;
```

Below the command, the results pane displays a table with the following data:

SEQUENCE_NAME	MAX_VALUE	INCREMENT_BY	LAST_NUMBER
DEPT_ID_SEQ	1000	10	200

The table has a single row. The bottom of the screen shows the URL "220701040@rajalakshmi.edu.in", the schema "bhanu23", and the copyright notice "Copyright © 1999, 2023, Oracle and/or its affiliates. Oracle APEX 23.2.4".

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 APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. On the right, there is a user profile for 'Bhanu Priya' (bhanu23). The main area displays the results of a script named 'ex14'. The 'Summary' view is selected, showing 1 row inserted. Below this, a detailed table shows the single statement executed: 'INSERT INTO dept VALUES (dept\_id\_seq.nextval, 'Education')'. The table includes columns for Number, Elapsed time (0.02), Statement, Feedback (1 row(s) inserted), and Rows (1). At the bottom, performance metrics are shown: 1 statement processed, 1 successful, and 0 with errors. The footer contains copyright information for Oracle and the APEX version (25.2.4).

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:

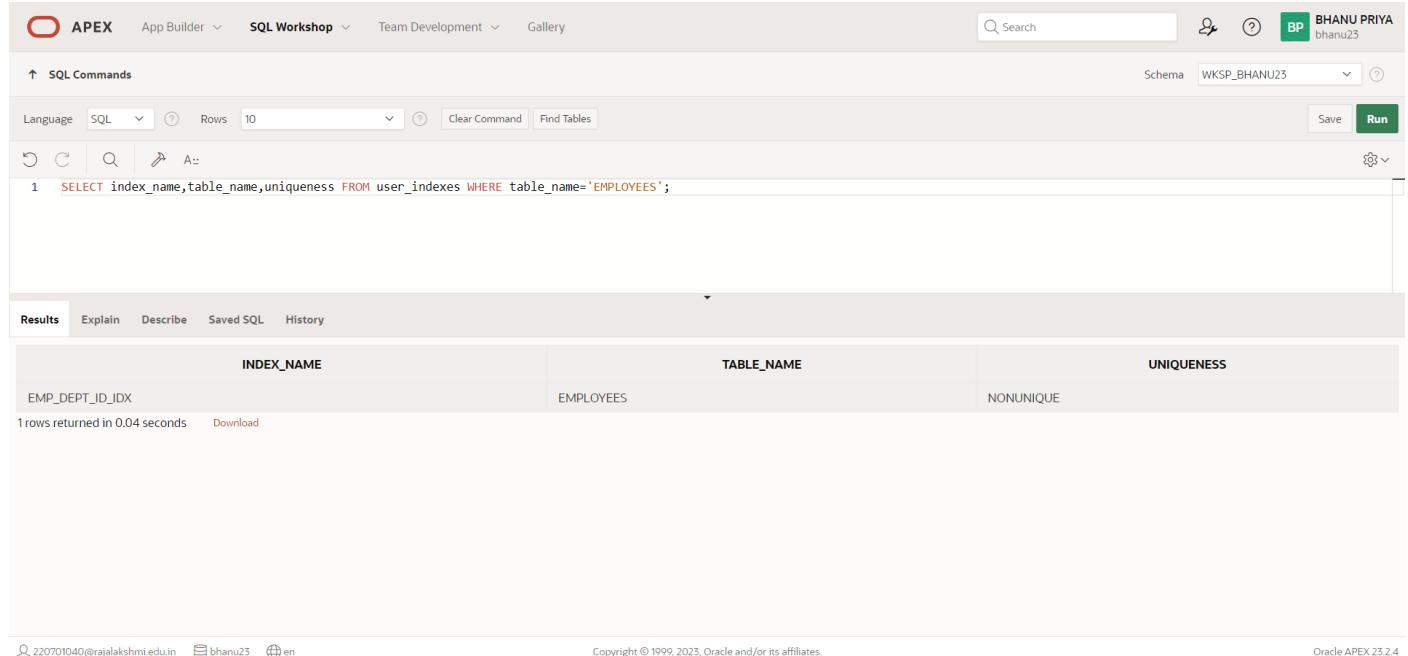
The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. On the right, there is a user profile for 'Bhanu Priya' (bhanu23). The main area displays the results of a SQL command to create an index. The 'SQL Commands' tab is active, showing the command 'CREATE INDEX emp\_dept\_id\_idx ON EMPLOYEES (department\_id);'. The 'Results' tab is selected, showing the output 'Index created.' and a timestamp of '0.03 seconds'. The footer contains copyright information for Oracle and the APEX version (25.2.4).

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 APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there's a search bar, a user icon for 'Bhanu Priya bhanu23', and a schema dropdown set to 'WKSP\_BHANU23'. The main workspace displays a SQL command line with the query: 'SELECT index\_name,table\_name,uniqueness FROM user\_indexes WHERE table\_name='EMPLOYEES';'. Below the command line, the results tab is selected, showing a single row of data in a table format:

INDEX_NAME	TABLE_NAME	UNIQUENESS
EMP_DEPT_ID_IDX	EMPLOYEES	NONUNIQUE

Below the table, it says '1 rows returned in 0.04 seconds' and has a 'Download' link. At the bottom of the page, there are footer links for email (220701040@rajalakshmi.edu.in), session (bhanu23), and language (en). The copyright notice 'Copyright © 1999, 2023, Oracle and/or its affiliates.' and the version 'Oracle APEX 23.2.4' are also present.

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
Query(5)	
Execution (5)	
Viva(5)	
Total (15)	
Faculty Signature	

**RESULT:**

# PL/SQL CONTROL STRUCTURES

**EX NO:16**

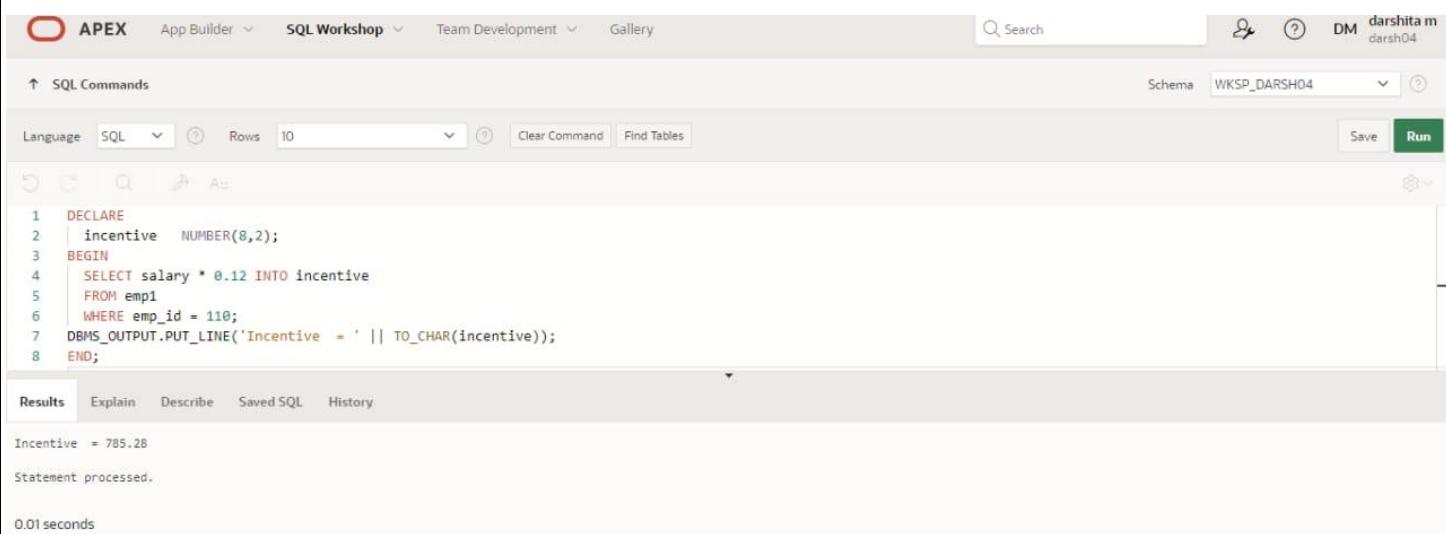
**DATE:**

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 top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. On the right, it shows the user 'darshita m' and schema 'WKSP\_DARSH04'. The main area has tabs for 'SQL Commands' (selected) and 'Results'. The SQL command entered is:

```
1 DECLARE
2     incentive NUMBER(8,2);
3 BEGIN
4     SELECT salary * 0.12 INTO incentive
5     FROM emp1
6     WHERE emp_id = 110;
7     DBMS_OUTPUT.PUT_LINE('Incentive = ' || TO_CHAR(incentive));
8 END;
```

The 'Results' tab shows the output:

```
Incentive = 785.28
Statement processed.
0.01 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 APEX SQL Workshop interface. In the SQL Commands tab, a PL/SQL block is entered:

```
1  DECLARE
2  | "WELCOME" varchar2(10) := 'welcome';
3  BEGIN
4  | DBMS_Output.Put_Line("Welcome");
5  END;
6  /
```

An error message is displayed in a yellow box:

```
Error at line 4/25: ORA-06550: line 4, column 25:
PLS-00201: identifier 'Welcome' must be declared
ORA-06512: at "SYS.WWV_DBMS_SQL_APEX_230200", line 801
ORA-06550: line 4, column 3:
PL/SQL: Statement ignored
```

The error message points to line 4, column 25, where the identifier 'Welcome' is used without being declared.

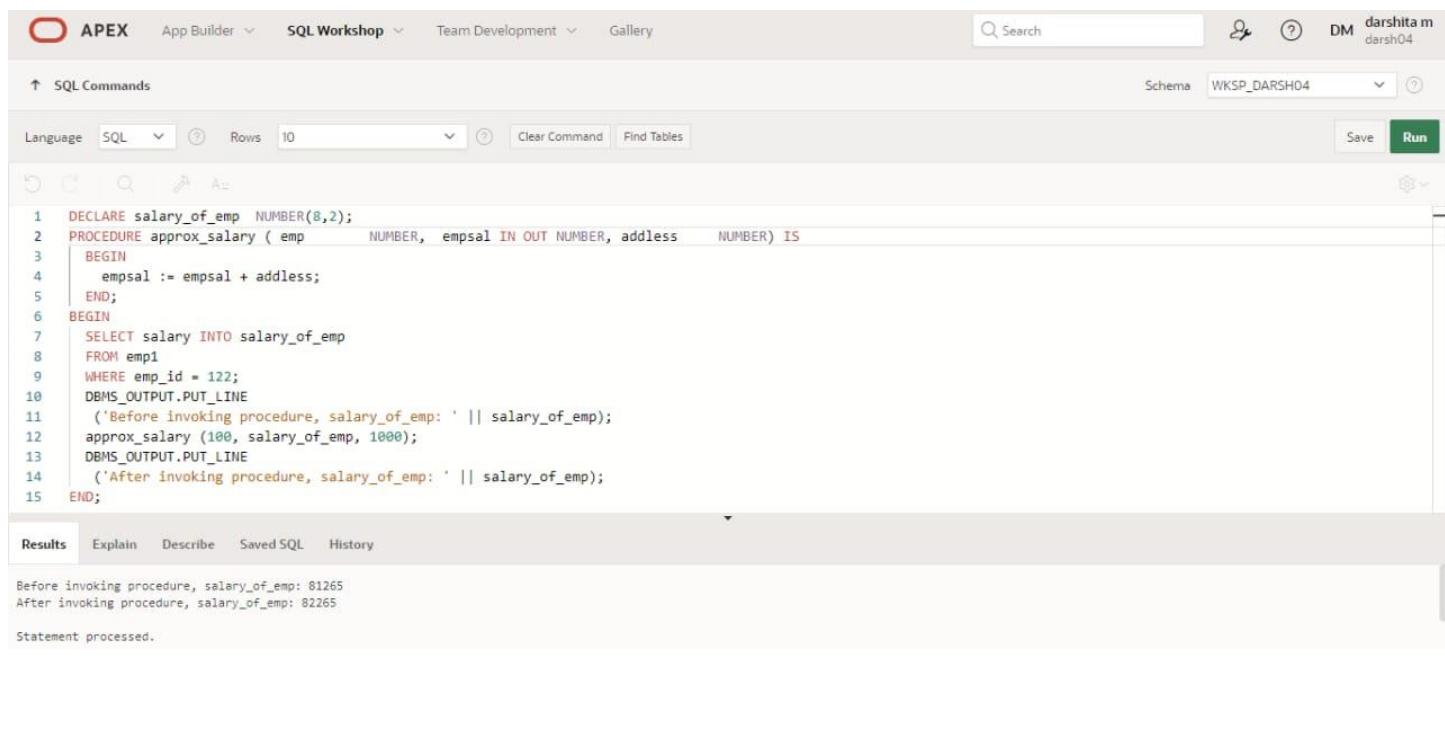
3.) Write a PL/SQL block to adjust the salary of the employee whose ID 122.

### 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 the user 'darshita m' and schema 'WKSP\_DARSH04'. The main area is titled 'SQL Commands' with tabs for Language (set to SQL), Rows (set to 10), Clear Command, Find Tables, Save, and Run. The code area contains the PL/SQL block from the previous step. The results tab at the bottom shows the output of the DBMS\_OUTPUT.PUT\_LINE statements: 'Before invoking procedure, salary\_of\_emp: 81265' and 'After invoking procedure, salary\_of\_emp: 82265'. A message 'Statement processed.' is also visible.

```
1 DECLARE salary_of_emp NUMBER(8,2);
2 PROCEDURE approx_salary ( emp      NUMBER, empsal IN OUT NUMBER, addless  NUMBER) IS
3 BEGIN
4     empsal := empsal + addless;
5 END;
6 BEGIN
7     SELECT salary INTO salary_of_emp
8     FROM emp1
9     WHERE emp_id = 122;
10    DBMS_OUTPUT.PUT_LINE
11    ('Before invoking procedure, salary_of_emp: ' || salary_of_emp);
12    approx_salary (100, salary_of_emp, 1000);
13    DBMS_OUTPUT.PUT_LINE
14    ('After invoking procedure, salary_of_emp: ' || salary_of_emp);
15 END;
```

Results Explain Describe Saved SQL History

Before invoking procedure, salary\_of\_emp: 81265  
After invoking procedure, salary\_of\_emp: 82265  
Statement processed.

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:

Results Explain Describe Saved SQL History

```
m = TRUE
m AND n = TRUE
----- FOR m TRUE AND n NULL -----
m = TRUE
n = NULL
m AND n = NULL
----- FOR m FALSE AND n NULL-----
m = FALSE
n = NULL
m AND n = FALSE
----- FOR m NULL AND n TRUE -----
m = NULL
n = TRUE
m AND n = NULL
----- FOR m NULL AND n FALSE -----
m = NULL
n = FALSE
m AND n = FALSE

Statement processed.

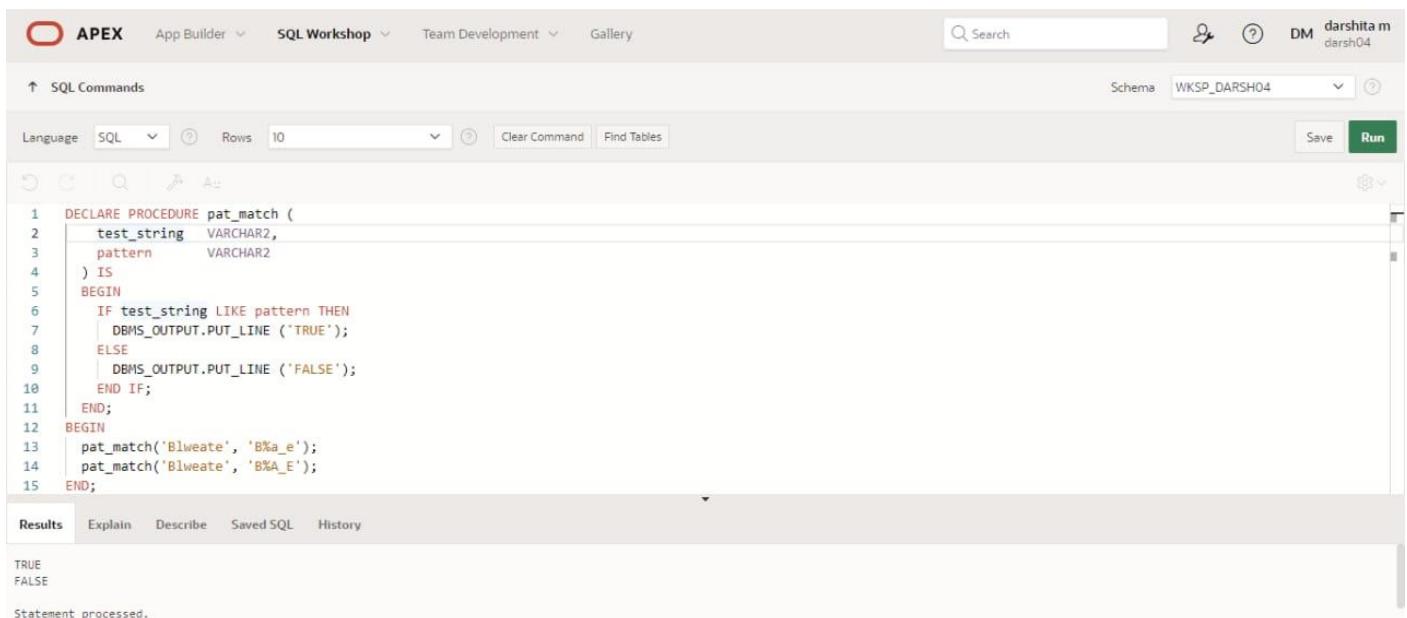
0.01 seconds
```

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, Team Development, and Gallery. The SQL Workshop tab is selected. The schema dropdown shows WKSP\_DARSH04. The main area displays the PL/SQL code for the pat\_match procedure. The code uses the LIKE operator with wildcards (%) to match strings. The output section shows the results of executing the procedure with two different patterns. The status bar at the bottom indicates "Statement processed."

```
1  DECLARE PROCEDURE pat_match (
2    test_string  VARCHAR2,
3    pattern      VARCHAR2
4  ) IS
5  BEGIN
6    IF test_string LIKE pattern THEN
7      DBMS_OUTPUT.PUT_LINE ('TRUE');
8    ELSE
9      DBMS_OUTPUT.PUT_LINE ('FALSE');
10   END IF;
11 END;
12 BEGIN
13   pat_match('Blweate', 'B%a_e');
14   pat_match('Blweate', 'B%A_E');
15 END;
```

Results Explain Describe Saved SQL History

TRUE  
FALSE

Statement processed.

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;
```

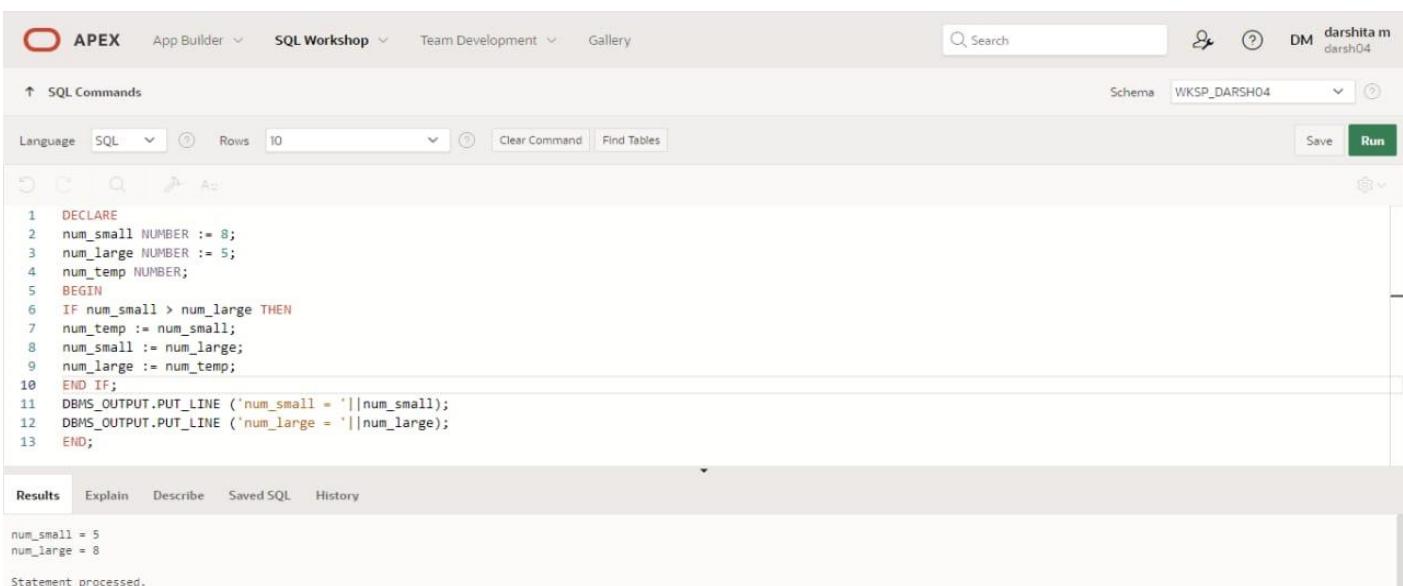
```
DBMS_OUTPUT.PUT_LINE ('num_small ='||num_small);
```

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

```
END;
```

```
/
```

### OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows a user profile for 'darshita m' and 'darsh04'. The main area is titled 'SQL Commands' with a schema dropdown set to 'WKSP\_DARSH04'. The code editor contains the provided PL/SQL block. The results tab shows the output 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  IF num_small > num_large THEN
7  num_temp := num_small;
8  num_small := num_large;
9  num_large := num_temp;
10 END IF;
11 DBMS_OUTPUT.PUT_LINE ('num_small ='||num_small);
12 DBMS_OUTPUT.PUT_LINE ('num_large ='||num_large);
13 END;
```

Results

```
num_small = 5
num_large = 8
Statement processed.
```

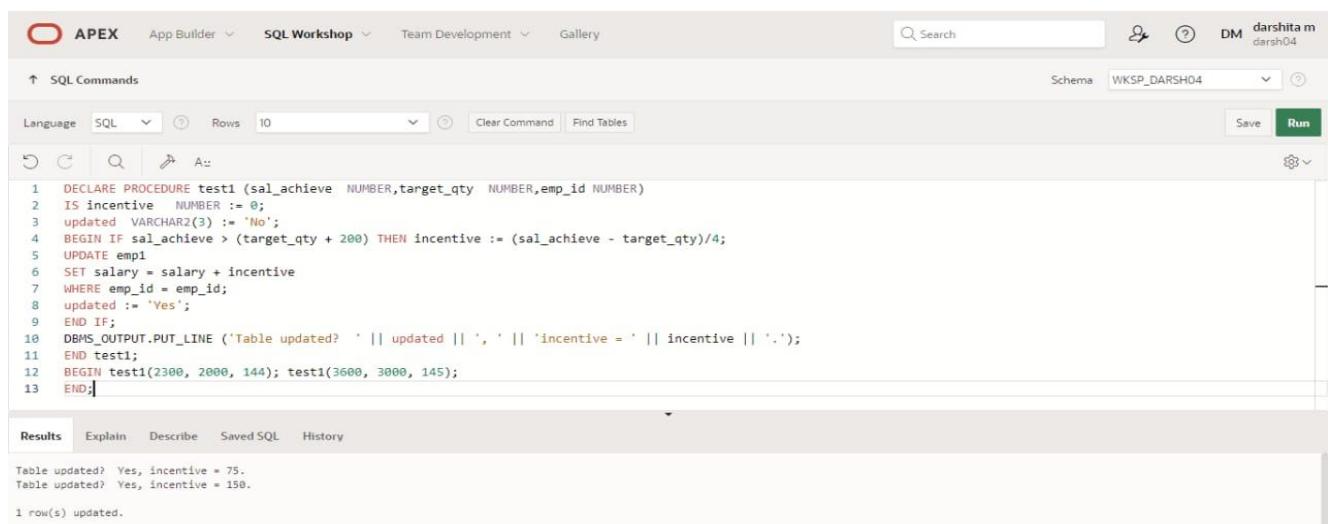
7.) Write a PL/SQL procedure to calculate the incentive on a target achieved and display the message either the record updated or not.

**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. 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\_DARSH04. The main area displays the PL/SQL code for the test1 procedure. The code is executed successfully, and the results pane shows the output: "Table updated? Yes, incentive = 75." and "Table updated? Yes, incentive = 150.". A note at the bottom indicates "1 row(s) updated."

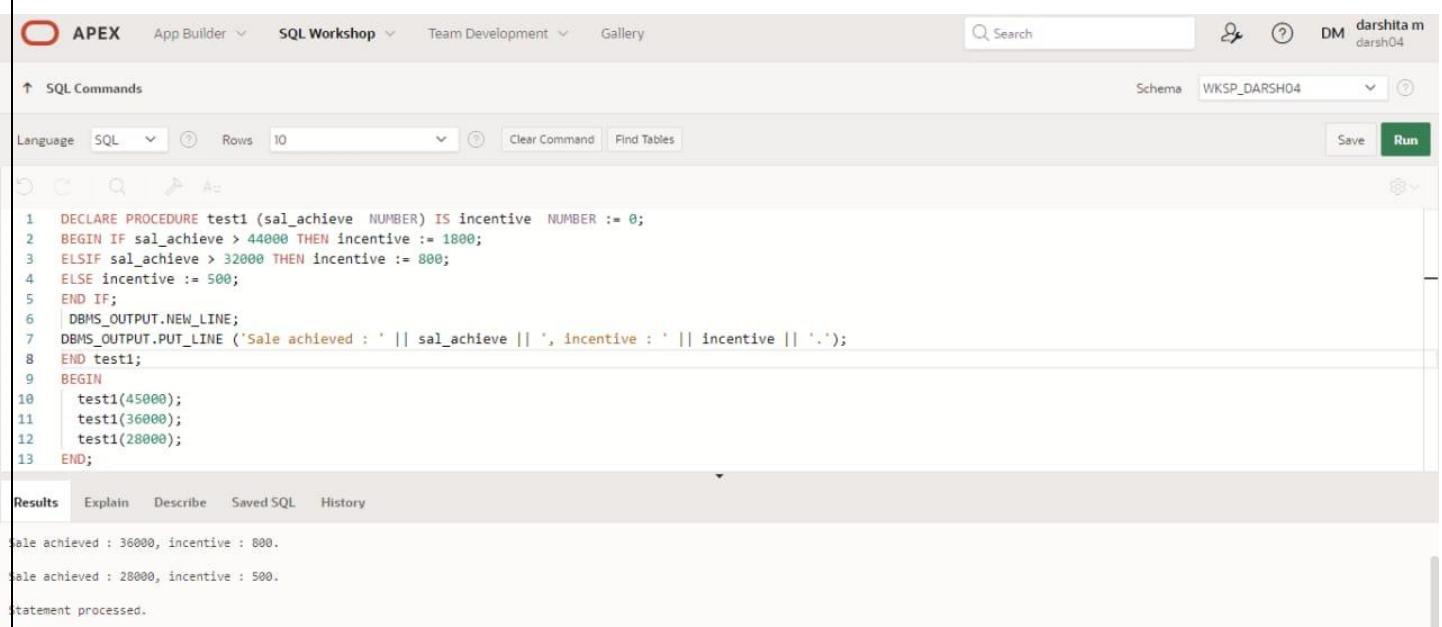
```
1  DECLARE PROCEDURE test1 (sal_achieve NUMBER,target_qty NUMBER,emp_id NUMBER)
2  IS incentive NUMBER := 0;
3  updated VARCHAR2(3) := 'No';
4  BEGIN IF sal_achieve > (target_qty + 200) THEN incentive := (sal_achieve - target_qty)/4;
5  UPDATE emp1
6  SET salary = salary + incentive
7  WHERE emp_id = emp_id;
8  updated := 'Yes';
9  END IF;
10 DBMS_OUTPUT.PUT_LINE ('Table updated? ' || updated || ',' ||
11 'incentive = ' || incentive || '.');
12 END test1;
13 BEGIN test1(2300, 2000, 144); test1(3600, 3000, 145);
14 END;
```

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. In the top navigation bar, 'APEX' is selected. The main area displays the PL/SQL code for the 'test1' procedure and its execution. The code includes logic for calculating incentives based on sales achievement and printing the results to the console. The output window at the bottom shows the executed statements and their results.

```
1  DECLARE PROCEDURE test1 (sal_achieve NUMBER) IS incentive NUMBER := 0;
2  BEGIN IF sal_achieve > 44000 THEN incentive := 1800;
3  ELSIF sal_achieve > 32000 THEN incentive := 800;
4  ELSE incentive := 500;
5  END IF;
6  DBMS_OUTPUT.NEW_LINE;
7  DBMS_OUTPUT.PUT_LINE ('Sale achieved : ' || sal_achieve || ', incentive : ' || incentive || '.');
8  END test1;
9  BEGIN
10    test1(45000);
11    test1(36000);
12    test1(28000);
13  END;
```

**Results**

```
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
```

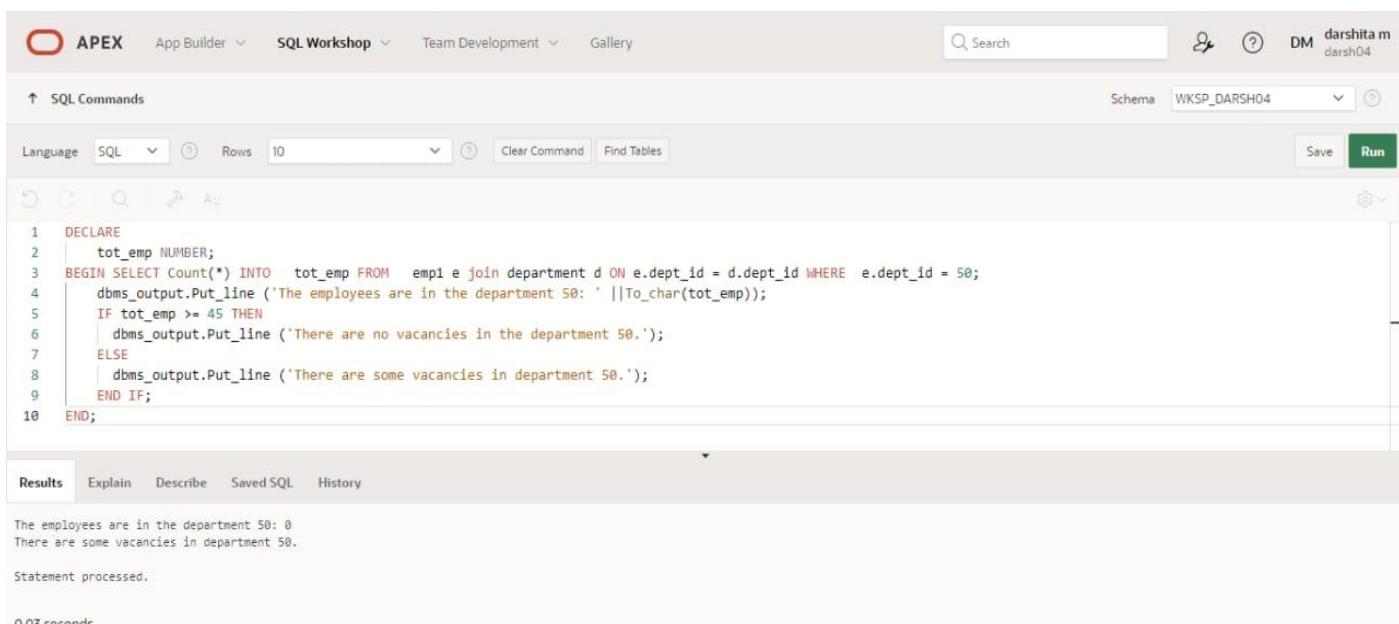
```
    dbms_output.Put_line ('There are'||to_char(45-tot_emp)||' vacancies in department'||get_dep_id  
);
```

```
END IF;
```

```
END;
```

```
/
```

**OUTPUT:**



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', 'Gallery', a search bar, and a user profile for 'darshita m darsh04'. The main workspace is titled 'SQL Commands'. The code area contains a PL/SQL block. The results tab at the bottom displays the output of the executed code.

```
1  DECLARE  
2  |      tot_emp NUMBER;  
3  BEGIN SELECT Count(*) INTO tot_emp FROM emp1 e join department d ON e.dept_id = d.dept_id WHERE e.dept_id = 50;  
4  dbms_output.Put_line ('The employees are in the department 50: ' ||To_char(tot_emp));  
5  IF tot_emp >= 45 THEN  
6  |      dbms_output.Put_line ('There are no vacancies in the department 50.');
```

The results tab shows the output:

```
The employees are in the department 50: 0  
There are some vacancies in department 50.  
Statement processed.  
0.03 seconds
```

**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:**

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery'. On the right, there's a user icon for 'Bhanu Priya' and a 'Run' button. The main workspace is titled 'SQL Commands' and contains the following PL/SQL code:

```
11  WHERE e.department_id = get_dep_id;
12  dbms_output.Put_line ('The employees are in the department'||get_dep_id||' is: '
13  ||To_char(tot_emp));
14  IF tot_emp >= 45 THEN
15  dbms_output.Put_line ('There are no vacancies in the department'||get_dep_id);
16  ELSE
17  dbms_output.Put_line ('There are'||to_char(45-tot_emp)||' vacancies in department'|| get_dep_id
);
18  get_dep_id );
19  END IF;
20 END;
21 /
22 
```

Below the code, the 'Results' tab is selected, showing the output:

```
The employees are in the department 80 is: 6
There are 39 vacancies in department 80
Statement processed.

0.01 seconds
```

At the bottom, footer information includes the URL '220701040@satyaledhmi.edu.in', the session ID 'bhani075', and the page number '1 of 1'. The copyright notice 'Copyright © 1999-2013, Oracle and/or its affiliates.' and the page 'Oracle APEX 7.2.4' are also present.

**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;

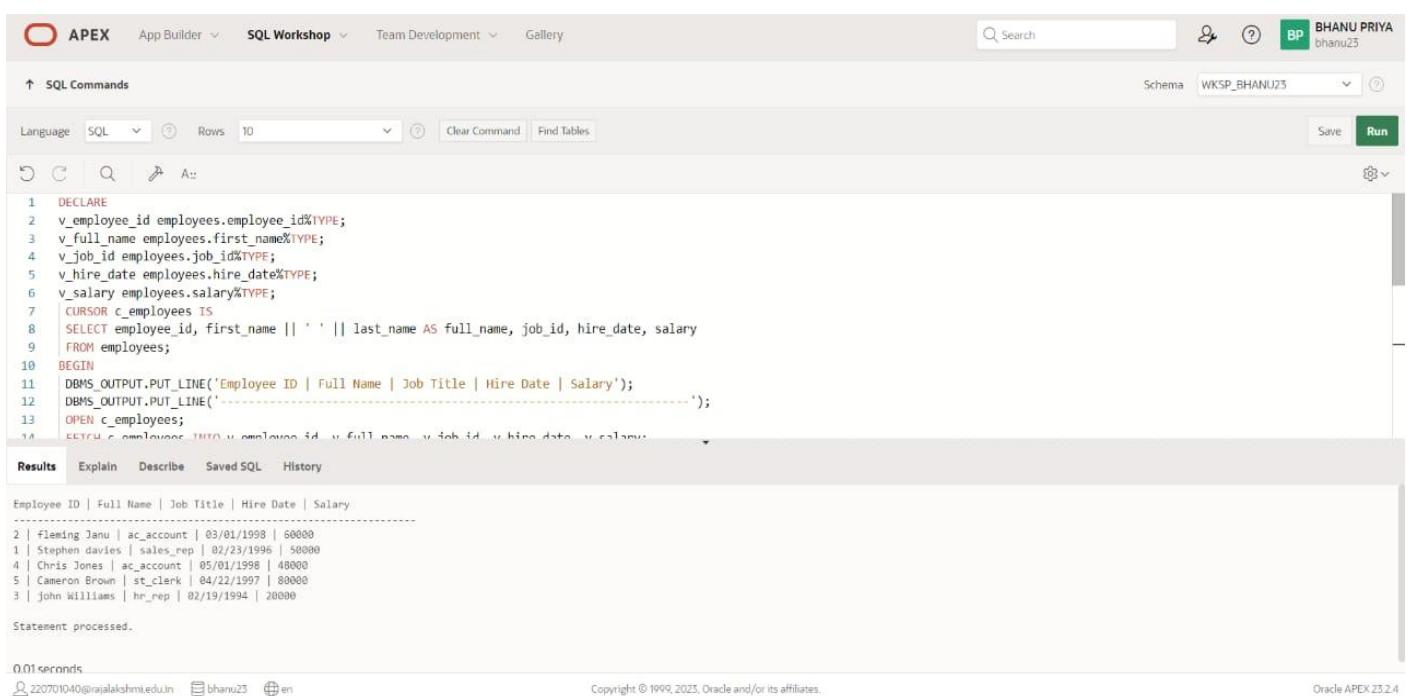
END LOOP;

CLOSE c\_employees;

END;

/

**OUTPUT:**



```
Employee ID | Full Name | Job Title | Hire Date | Salary
-----|-----|-----|-----|-----
2 | Fleming Janu | ac_account | 03/01/1998 | 60000
1 | Stephen davies | sales_rep | 02/23/1996 | 50000
4 | Chris Jones | ac_account | 05/01/1998 | 48000
5 | Cameron Brown | st_clerk | 04/22/1997 | 80000
3 | John Williams | hr_rep | 02/19/1994 | 20000
```

Statement processed.

0.01 seconds

220701040@rajalakshmi.edu.in bhanu25 en

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

Oracle APEX 23.2.4

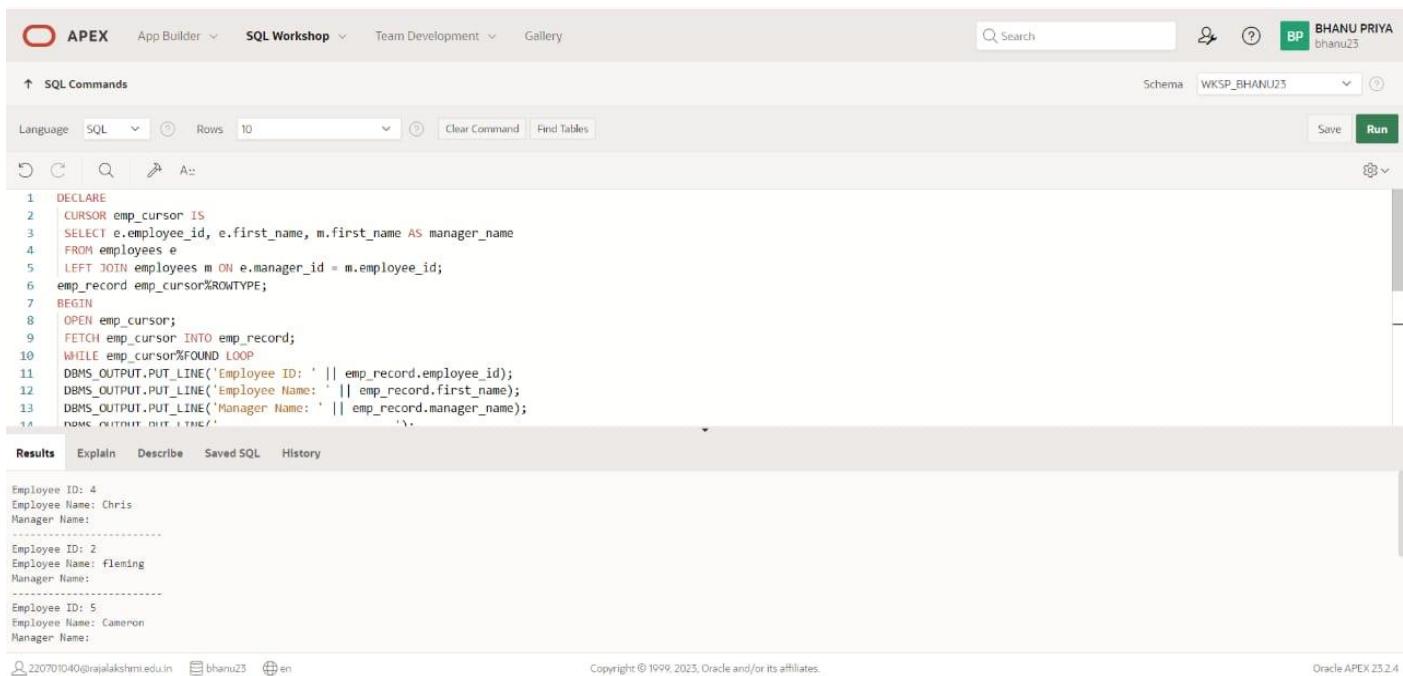
**12.) Write a PL/SQL program to display the employee IDs, names, and department names of all employees.**

**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 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 'Bhanu Priya' with session ID 'bhanu23'. The schema is set to 'WKSP\_BHANU23'. The main workspace is titled 'SQL Commands' and contains the PL/SQL code provided above. Below the code, the 'Results' tab is selected, displaying the output of the program. The output shows three records from the employees table, each with their Employee ID, First Name, and Manager Name, separated by a horizontal line.

Employee ID	Employee Name	Manager Name
4	Chris	
2	fleming	
5	Cameron	

**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:**

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 connected to schema WKSP\_BHANU23. The SQL Commands tab is active, showing the PL/SQL code. The Results tab is selected, displaying the output of the code execution.

```
1  DECLARE
2  CURSOR job_cursor IS
3  SELECT e.job_id, j.lowest_sal
4  FROM job_grade 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   DBMS_OUTPUT.PUT_LINE('Job ID: ' || job_record.job_id);
11   DBMS_OUTPUT.PUT_LINE('Minimum Salary: ' || job_record.lowest_sal);
12   DBMS_OUTPUT.PUT_LINE('-----');
13   FETCH job_cursor INTO job_record;
14 END LOOP;
```

Results

Job ID	Minimum Salary
ac_account	48000
sales_rep	48000
ac_account	48000
st_clerk	48000

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

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

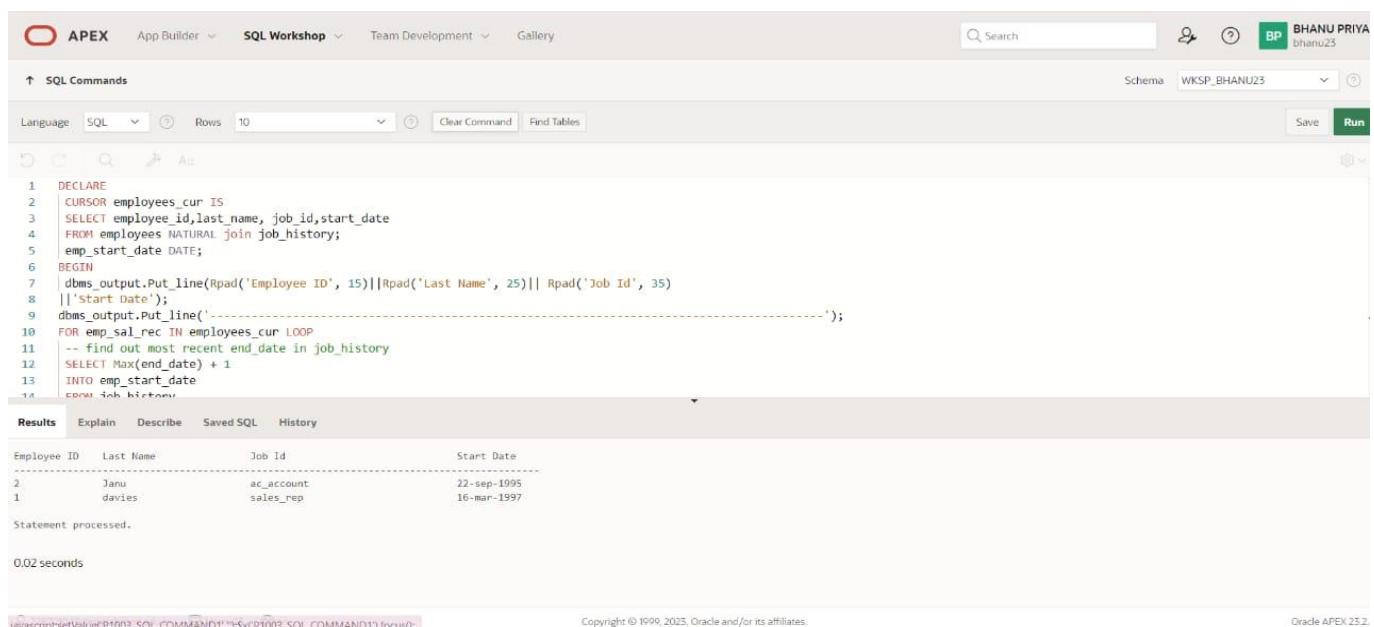
**QUERY:**

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 APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. The right side of the header shows the user 'BHANU PRIYA' and the schema 'WKSP\_BHANU23'. The main workspace has tabs for SQL Commands, Results, Explain, Describe, Saved SQL, and History. The SQL Commands tab contains the PL/SQL code provided above. The Results tab displays the output of the code execution, which includes the following table:

Employee ID	Last Name	Job Id	Start Date
2	Janet	sa_sales	22-sep-1995
1	Davies	sa_sales	16-mar-1997

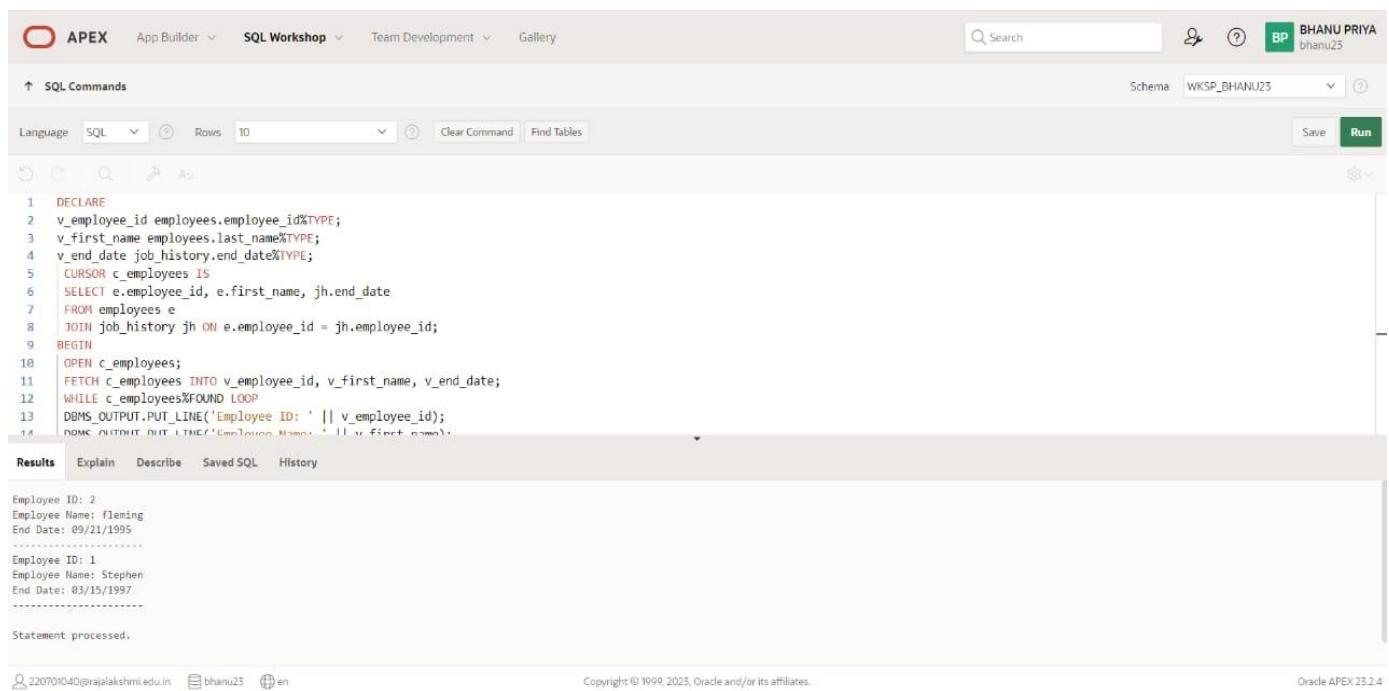
Below the table, the message 'Statement processed.' is shown, followed by '0.02 seconds'.

**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 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 'Bhanu Priya' with the schema 'WKSP\_BHANU25'. 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;
12      WHILE c_employees%FOUND LOOP
13        DBMS_OUTPUT.PUT_LINE('Employee ID: ' || v_employee_id);
14        DBMS_OUTPUT.PUT_LINE('Employee Name: ' || v_first_name);
15        DBMS_OUTPUT.PUT_LINE('End Date: ' || v_end_date);
16        DBMS_OUTPUT.PUT_LINE('-----');
```

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

```
Employee ID: 2
Employee Name: Fleming
End Date: 09/21/1995
-----
Employee ID: 1
Employee Name: Stephen
End Date: 03/15/1997
-----
Statement processed.
```

At the bottom, the footer includes the URL '220701040@rajalakshmi.edu.in', the session identifier 'bhnu23', and the copyright notice 'Copyright © 1999, 2023, Oracle and/or its affiliates.' The page also indicates it is 'Oracle APEX 23.2.4'.

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;

$n \equiv n - 1$

END LOOP.

DBMS\_OUTPUT.PUT\_LINE(fac);

END.

### **OUTPUT:**

The screenshot shows the Oracle APEX interface with the 'SQL Workshop' selected in the top navigation bar. The main area displays a PL/SQL block for calculating a factorial:

```
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;
```

The code is syntax-highlighted, with keywords in blue and identifiers in green. The 'Run' button is visible at the bottom right of the code editor.

**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 SQL Workshop interface. The top navigation bar includes 'APEX' (with a red circle icon), 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. On the right, there's a search bar, a user icon for 'Bhanu Priya', and a schema dropdown set to 'WKSP\_BHANU23'. The main workspace is titled 'SQL Commands' and contains the following PL/SQL code:

```
1 CREATE OR REPLACE PROCEDURE get_book_info (
2     p_book_id IN NUMBER,
3     p_title OUT VARCHAR2,
4     p_author OUT VARCHAR2,
5     p_year_published OUT NUMBER
6 )
7 AS
8 BEGIN
9     SELECT title, author, year_published INTO p_title, p_author, p_year_published
10    FROM books
11   WHERE book_id = p_book_id;

```

Below the code, the 'Results' tab is selected, showing the message 'Procedure created.' and a execution time of '0.04 seconds'. The bottom footer includes user information (220701040@rajalakshmi.edu.in, bhanu23, en), copyright notice (Copyright © 1999, 2022, Oracle and/or its affiliates.), and the software version (Oracle APEX 23.2.4).

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 APEX interface with the SQL Workshop tab selected. In the SQL Commands pane, the trigger code is pasted. The code creates a trigger named 'prevent\_parent\_deletion' that fires before a delete operation on the 'parent\_table'. It declares an exception 'child\_exists' and initializes it with code number -20001 via 'PRAGMA EXCEPTION\_INIT'. It also declares a variable 'v\_child\_count' of type NUMBER. The trigger body begins with a 'BEGIN' block, which contains a 'SELECT' statement to count child records for the deleted parent. If the count is greater than zero, it raises the 'child\_exists' exception. An 'EXCEPTION' block handles this raise, causing an application error with message 'Cannot delete parent record while child records exist.' The 'Results' tab at the bottom shows the output '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 END;
```

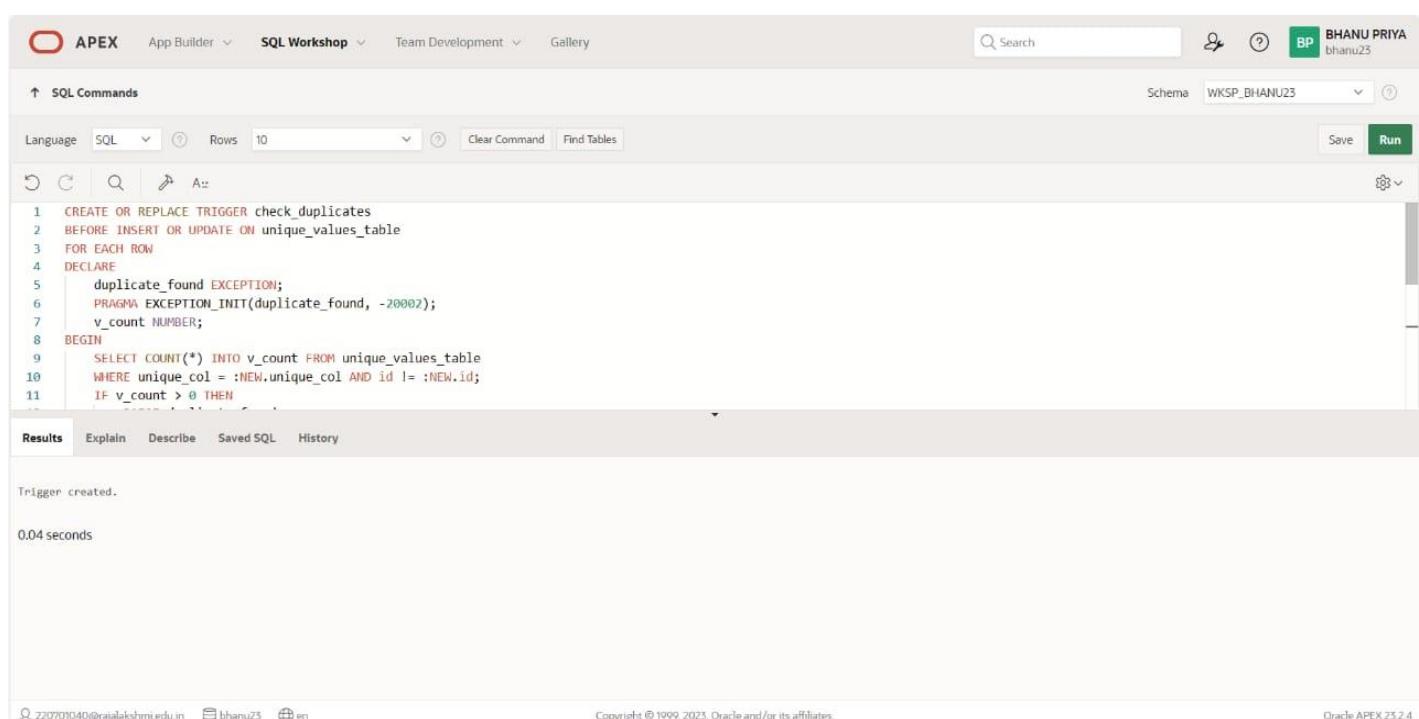
Trigger created.  
0.04 seconds

**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 APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery'. The right side of the header shows the user 'BHANDU PRIYA' and the schema 'WKSP\_BHANU23'. The main workspace is titled 'SQL Commands' and contains the PL/SQL code for the trigger. The code is highlighted in blue and black. Below the code, the 'Results' tab is selected, showing the message 'Trigger created.' and a execution time of '0.04 seconds'. The bottom footer includes copyright information for Oracle and the APEX version 'Oracle APEX 23.2.4'.

```
1 CREATE OR REPLACE TRIGGER check_duplicates
2 BEFORE INSERT OR UPDATE ON unique_values_table
3 FOR EACH ROW
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

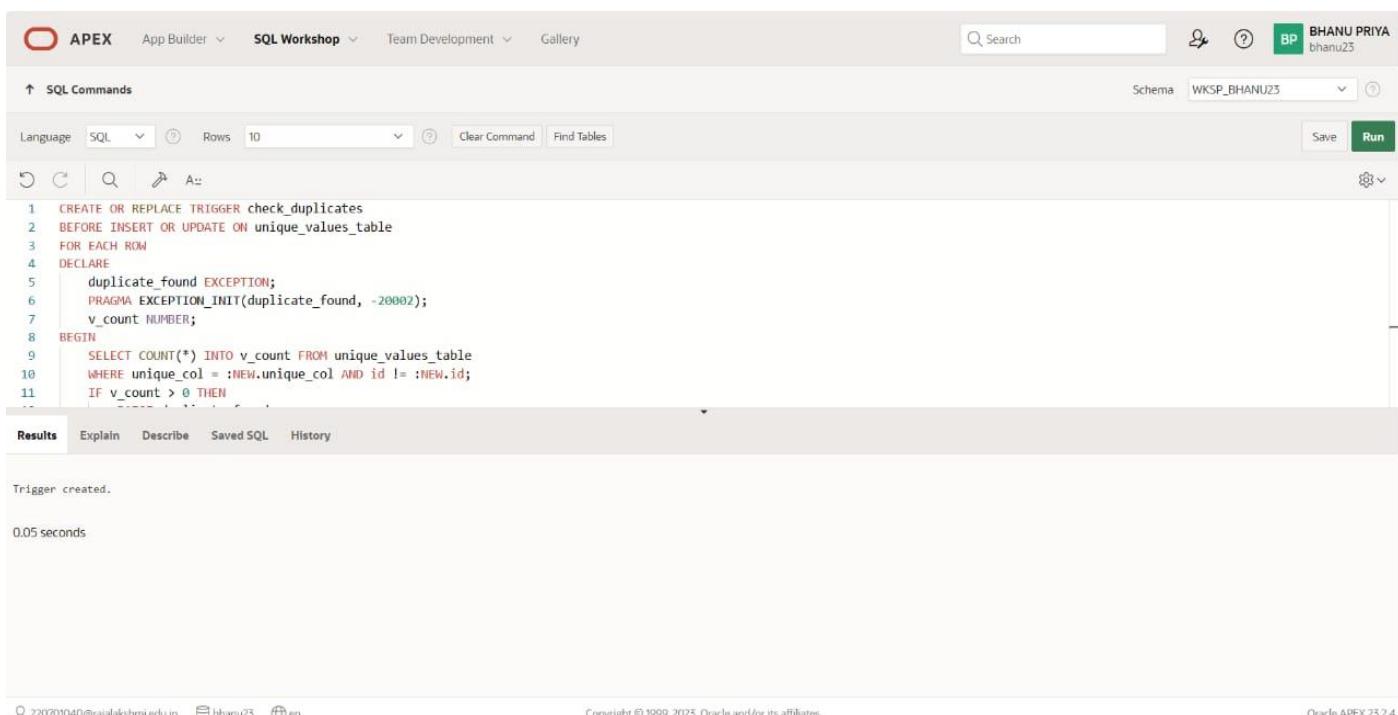
```

**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 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 'Bhanu Priya' (bhanu25). The main area is titled 'SQL Commands' with tabs for Language (SQL selected), Rows (10), Clear Command, Find Tables, Save, and Run. Below the tabs is a toolbar with icons for Undo, Redo, Search, and others. The SQL editor contains the PL/SQL code provided in the question. The results tab at the bottom shows the output: 'Trigger created.' and '0.05 seconds'. The footer includes copyright information for Oracle and the APEX version.

```
1 CREATE OR REPLACE TRIGGER check_duplicates
2 BEFORE INSERT OR UPDATE ON unique_values_table
3 FOR EACH ROW
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 END;
```

**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 APEX SQL Workshop interface. In the top navigation bar, 'APEX' is selected. Below it, tabs for 'App Builder', 'SQL Workshop' (which is active), and 'Team Development' are visible. On the right side, a user profile for 'Bhanu Priya' (bhanu23) is shown. The main area is titled 'SQL Commands'. The language is set to 'SQL'. The command entered is the PL/SQL code for creating the 'log\_changes' trigger. The code is as follows:

```
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
```

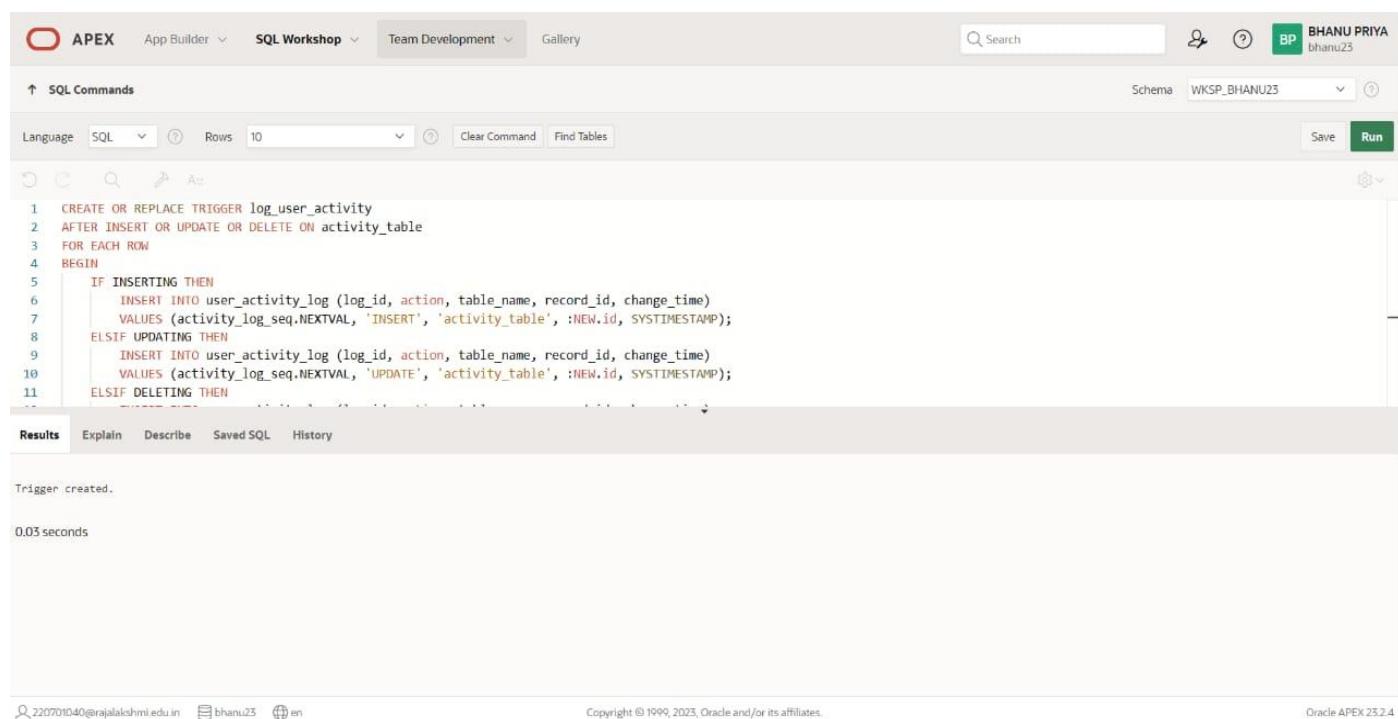
Below the command, the 'Results' tab is selected. The output shows the message 'Trigger created.' and a execution time of '0.05 seconds'. At the bottom of the page, there are footer links for copyright information and Oracle APEX version 23.2.4.

**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 APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side of the header shows a user profile for 'BHANU PRIYA' and the schema 'WKSP\_BHANU23'. The main workspace is titled 'SQL Commands' and contains the PL/SQL code for the trigger. The code is highlighted in red and blue, corresponding to keywords and identifiers. Below the code, there are tabs for 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History'. The results section displays the message 'Trigger created.' and a execution time of '0.03 seconds'. At the bottom, footer information includes the URL '220701040@rajalakshmi.edu.in', the session ID 'bhnu23', and the language 'en'. The copyright notice 'Copyright © 1999, 2023, Oracle and/or its affiliates.' and the version 'Oracle APEX 23.2.4' are also present.

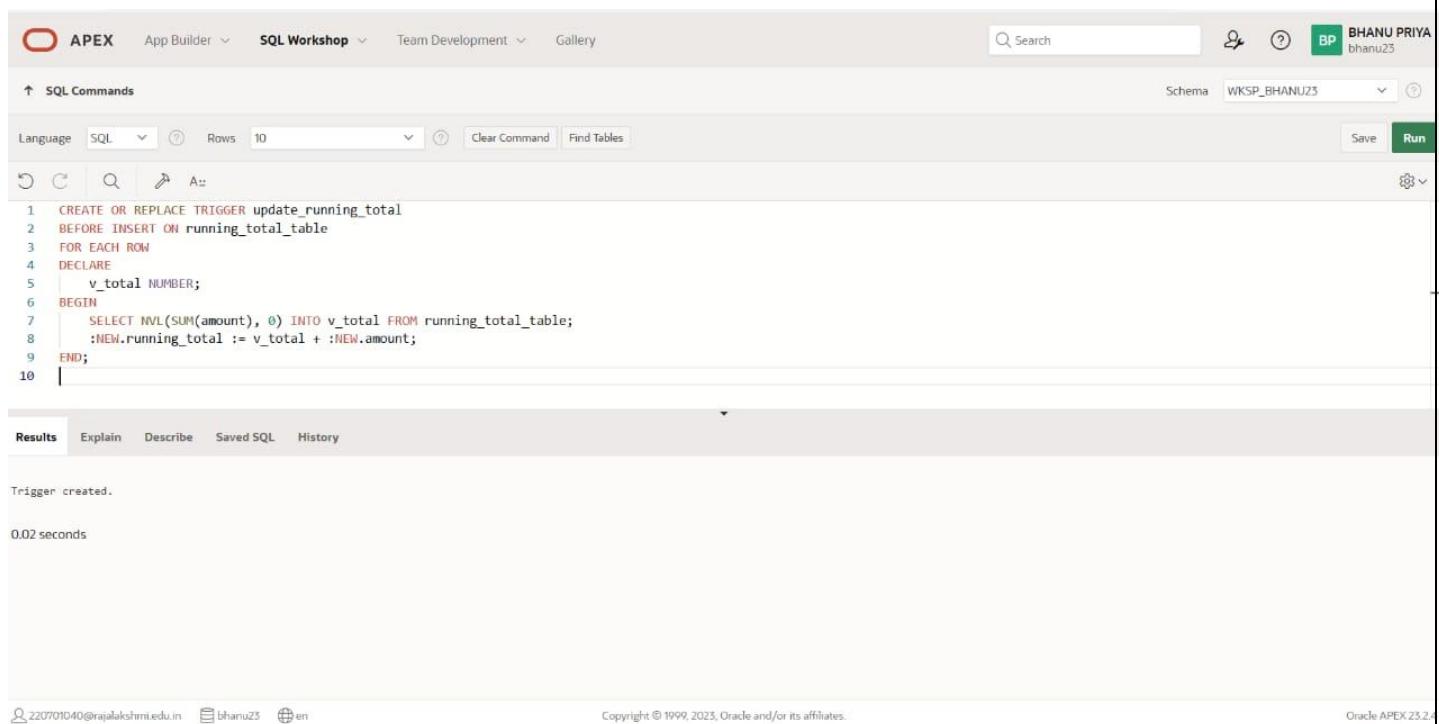
```
1 CREATE OR REPLACE TRIGGER log_user_activity
2 AFTER INSERT OR UPDATE OR DELETE ON activity_table
3 FOR EACH ROW
4 BEGIN
5     IF INSERTING THEN
6         INSERT INTO user_activity_log (log_id, action, table_name, record_id, change_time)
7         VALUES (activity_log_seq.NEXTVAL, 'INSERT', 'activity_table', :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
```

**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 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 'Bhanu Priya' (bhanu23). The main area is titled 'SQL Commands' with a 'Language' dropdown set to 'SQL'. The code editor contains the PL/SQL trigger creation script. The results tab shows the output: 'Trigger created.' and a execution time of '0.02 seconds'. The bottom footer includes copyright information for Oracle and the APEX version.

```
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 |
```

Results Explain Describe Saved SQL History

Trigger created.  
0.02 seconds

Copyright © 1999, 2023, Oracle and/or its affiliates. Oracle APEX 23.2

**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 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 'Bhanu Priya' (bhanu23). The main area is titled 'SQL Commands' with a 'Schema' dropdown set to 'WKSP\_BHANU23'. The code area contains the PL/SQL trigger definition shown above. Below the code, the 'Results' tab is selected, displaying the message 'Trigger created.' and a execution time of '0.04 seconds'. The bottom footer includes copyright information for Oracle and the APEX version.

```
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 = :NEW.item_id;
10    IF v_stock < :NEW.order_quantity THEN
11        RAISE insufficient_stock;
12    END IF;
13    UPDATE items SET stock_quantity = stock_quantity - :NEW.order_quantity WHERE
item_id = :NEW.item_id;
14 EXCEPTION
15     WHEN insufficient_stock THEN
16         RAISE_APPLICATION_ERROR(-20004, 'Insufficient stock for the item.');
17 END;
```

Trigger created.  
0.04 seconds

Copyright © 1999, 2025, Oracle and/or its affiliates. Oracle APEX 23.4

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:**

```
Bhanupriya_40> db.restaurants.find({$and:[{ $or: [{cuisine: { $nin: ["American", "Chinese"] }}, {name: /"Wil/}]}], {restaurant_id: 1, name: 1, borough: 1, cuisine: 1})
[
  {
    _id: ObjectId('6650a544c5dcab95e3cdcdf6'),
    borough: 'Bronx',
    cuisine: 'Bakery',
    name: 'Morris Park Bake Shop',
    restaurant_id: '30075445'
  }
]
Bhanupriya_40> |
```

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:**

```
Bhanupriya_40> db.restaurants.find({ "grades": { $elemMatch: { "grade": "A", "score": 11, "date": ISODate("2014-08-11T00:00:00Z") } } }, { restaurant_id: 1, name: 1, grades: 1 })
Bhanupriya_40>
```

**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:**

```
Bhanupriya_40> 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 } )
```

```
Bhanupriya_40>
```

**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:**

```
Bhanupriya_40> db.restaurants.find({$and : [{"address.coord.1": {$gt : 42}}, {"address.coord.1": {$lte : 52}}]}, {_id:0, restaurant_id:1, name:1, address:1})
```

```
Bhanupriya_40>
```

**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:**

```
Bhanupriya_40> 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'
}
]
Bhanupriya_40>
```

**6.) Write a MongoDB query to arrange the name of the restaurants in descending order along with all the columns.**

**QUERY:**

```
db.restaurants.find({}, { _id: 0 }).sort({ name: -1 })
```

**OUTPUT:**

```
Bhanupriya_40> 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'
}
]
Bhanupriya_40>
```

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:

```
Bhanupriya_40> db.restaurants.find({ "address.street": { $exists: true, $ne: "" } })
[
  {
    _id: ObjectId('6650a544c5dcab95e3cdcdf6'),
    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'
  }
]
Bhanupriya_40>
```

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:

```
Bhanupriya_40> db.restaurants.find({ "address.street": { $exists: true, $ne: "" } })
[
  {
    _id: ObjectId('6650a544c5dcab95e3cdcdf6'),
    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'
  }
]
Bhanupriya_40>
```

**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:**

```
Bhanupriya_40> db.restaurants.find({ "address.coord": { $elemMatch: { $type: "double" } } })
[ {
  _id: ObjectId('6650a544c5dcab95e3cdcdf6'),
  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'
}
]
Bhanupriya_40>
```

**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:**

```
Bhanupriya_40> db.restaurants.find({ "grades.score": { $mod: [7, 0] } }, { restaurant_id: 1, name: 1, grades: 1 })
[ {
  _id: ObjectId('6650a544c5dcab95e3cdcdf6'),
  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'
}
]
Bhanupriya_40>
```

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:**

```
Bhanupriya_40> db.restaurants.find({ name: /mon/i }, { name: 1, borough: 1, "address.coord": 1, cuisine: 1 })  
Bhanupriya_40>
```

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:**

```
Bhanupriya_40> db.restaurants.find({ name: /mon/i }, { name: 1, borough: 1, "address.coord": 1, cuisine: 1 })  
Bhanupriya_40>
```

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:**

```
Bhanupriya_40> db.restaurants.find({ "grades": { $elemMatch: { "score": { $lt: 5 } } } })
[ {
    _id: ObjectId('6650a544c5dcab95e3cdcdf6'),
    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'
}
]
Bhanupriya_40>
```

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:**

```
Bhanupriya_40> db.restaurants.find({ "grades": { $elemMatch: { "score": { $lt: 5 } } }, "borough": "Manhattan" })
Bhanupriya_40>
```

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:**

```
Bhanupriya_40> 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:**

```
Bhanupriya_40> 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:**

```
Bhanupriya_40> db.restaurants.find({ "grades": { $elemMatch: { "score": { $lt: 5 } } }, $or: [{ "borough": "Manhattan" }, { "borough": "Brooklyn" }], "cuisine": { $nin: ["American", "Chinese"] } })  
Bhanupriya_40>
```

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:**

```
Bhanupriya_40> db.restaurants.find({ $and: [{ "grades.grade": "A", "grades.score": 2 }, { "grades.grade": "A", "grades.score": 6 }] })  
[  
  {  
    _id: ObjectId('6650a544c5dcab95e3cdcdf6'),  
    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'  
  }  
]  
Bhanupriya_40>
```

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:**

```
Bhanupriya_40> db.restaurants.find({ $and: [{ "grades.grade": "A", "grades.score": 2 }, { "grades.grade": "A", "grades.score": 6 }], $or: [{ "borough": "Manhattan" }, { "borough": "Brooklyn" }] })
```

```
Bhanupriya_40>
```

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:**

```
Bhanupriya_40> db.restaurants.find({ $and: [{ "grades.grade": "A", "grades.score": 2 }, { "grades.grade": "A", "grades.score": 6 }], $or: [{ "borough": "Manhattan" }, { "borough": "Brooklyn" }] })
```

```
Bhanupriya_40>
```

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:**

```
Bhanupriya_40> db.restaurants.find({ $and: [{ "grades.grade": "A", "grades.score": 2 }, { "grades.grade": "A", "grades.score": 6 }], $or: [{ "borough": "Manhattan" }, { "borough": "Brooklyn" }], "cuisine": { $ne: "American" } })
```

```
Bhanupriya_40>
```

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:**

```
Bhanupriya_40> 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"] } })
```

```
Bhanupriya_40>
```

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:**

```
Bhanupriya_40> db.restaurants.find({ $or: [{ "grades.score": 2 }, { "grades.score": 6 }] })
[ {
  _id: ObjectId('6650a544c5dcab95e3cdcdf6'),
  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'
}
]
Bhanupriya_40>
```

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 year.

```
Bhanupriya_40> db.movies.find({ year: 1893 })  
Bhanupriya_40>
```

- 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:**

```
Bhanupriya_40> db.movies.find({ runtime: { $gt: 120 } })  
Bhanupriya_40>
```

### 3.) Find all movies with full information from the 'movies' collection that have "Short" genre.

**QUERY:**

```
db.movies.find({ genres: 'Short' })
```

**OUTPUT:**

```
Bhanupriya_40> 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/MVSBMTU3NjE5NzYtYTYYN500MDVmLWIwYjgtMmYwYWIxZDYyNzU2XkEyXkFqcGdeQXVyNzQzNzQxNzI@._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')
    }
  }
]
Bhanupriya_40>
```

### 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:**

```
Bhanupriya_40> 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/MVSBMTU3NjE5NzYtYTYYN500MDVmLWIwYjgtMmYwYWIxZDYyNzU2XkEyXkFqcGdeQXVyNzQzNzQxNzI@._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')
    }
  }
]
Bhanupriya_40>
```

**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:**

```
Bhanupriya_40> db.movies.find({ directors: 'William K.L. Dickson' })  
Bhanupriya_40>
```

**6.) Retrieve all movies from the 'movies' collection that have complete information and are rated as "UNRATED".**

**QUERY:**

```
db.movies.find({ rated: 'UNRATED' })
```

**OUTPUT:**

```
Bhanupriya_40> db.movies.find({ directors: 'William K.L. Dickson' })  
Bhanupriya_40>
```

**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:**

```
Bhanupriya_40> db.movies.find({ directors: 'William K.L. Dickson' })  
Bhanupriya_40>
```

**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**

```
Bhanupriya_40> 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/MV5BMHU3NjE5NzYtYTYYNS00MDVmlWlTwYjgtMmYwYWIxZDYyNzU2XkEyXkFqcGdeQXVyNzQzNzQxNzI@._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('2015-08-08T19:16:18.000Z')  
    }  
  }  
]  
Bhanupriya_40>
```

**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:**

```
Bhanupriya_40> db.movies.find({ 'tomatoes.viewer.rating': { $gt: 4 } })  
Bhanupriya_40>
```

**10.) Retrieve all movies from the 'movies' collection that have received an award.**

**QUERY:**

```
db.movies.find({ 'awards.wins': { $gt: 0 } })
```

**OUTPUT:**

```
Bhanupriya_40> 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/MVSBMTU3NjE5NzYtYTYYNS00MDVnLWIwYjgtMmYwYWIxZDYyNzU2XkEyXkFqcGdeQXVyNzQzNzQzNzI@._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')  
    }  
  }  
]  
Bhanupriya_40> |
```

**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:**

```
Bhanupriya_40> 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 }
... )
```

Bhanupriya\_40>

**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:**

```
Bhanupriya_40> 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 }
... )
```

Bhanupriya\_40>

**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:**

```
Bhanupriya_40> db.movies.find(
...   { released: ISODate("1893-05-09T00:00:00.000Z") },
...   { title: 1, languages: 1, released: 1, directors: 1, writers: 1, countries: 1 }
... )
```

Bhanupriya\_40>

**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:**

```
Bhanupriya_40> db.movies.find(
...   { title: /scene/i },
...   { title: 1, languages: 1, released: 1, directors: 1, writers: 1, countries: 1 }
... )
```

Bhanupriya\_40>

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

**RESULT:**