

# Creating and Managing Tables

EX\_NO:1

DATE:

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

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

## QUERY:

```
Create table dept(id number(7),name varchar2(25));
```

## OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there's a search bar, a user icon for 'deepthi\_p', and a schema dropdown set to 'WKSP\_DEEPTHIP22070105'. The main area is titled 'SQL Commands' and contains a command input field with the SQL statement 'Create table dept(id number(7),name varchar2(25));'. Below the input field, there are tabs for 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History'. The results section displays the message 'Table created.' and '0.03 seconds'.

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

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

### QUERY:

```
Create table emp(id number(7),Last_Name varchar2(25),First_Name varchar2(25),Dept_id number(7));
```

### OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. In the top navigation bar, 'APEX' is selected. The main area is titled 'SQL Commands'. A single command is entered in the text area: 'Create table emp(id number(7),Last\_Name varchar2(25),First\_Name varchar2(25),Dept\_id number(7));'. The 'Run' button is highlighted in green at the bottom right. Below the command, the results section displays the message 'Table created.' and a execution time of '0.03 seconds'.

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

#### QUERY:

```
Alter table emp modify(Last_Name varchar2(25));
```

#### OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. In the top navigation bar, 'APEX' is selected. The main area is titled 'SQL Commands'. A single line of SQL code is entered: '1 Alter table emp modify(Last\_Name varchar2(25));'. Below the command, the results section shows the output: 'Table altered.' and '0.06 seconds'. The schema dropdown at the top right shows 'WKSP\_DEEPTHIP22070105'.

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

#### QUERY:

```
Create table employees2(id number(7),first_name varchar2(25),Last_name varchar2(25),Salary int,Dept_id number(7));
```

#### OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. In the top navigation bar, 'APEX' is selected. The main area is titled 'SQL Commands'. A single line of SQL code is entered: '1 Create table employees2(id number(7),first\_name varchar2(25),Last\_name varchar2(25),Salary int,Dept\_id number(7));'. Below the command, the results section shows the output: 'Table created.' and '0.03 seconds'. The schema dropdown at the top right shows 'WKSP\_DEEPTHIP22070105'.

5.Drop the EMP table.

**QUERY:**

```
Drop table emp;
```

**OUTPUT:**

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (which is selected), 'Team Development', and 'Gallery'. A search bar and user profile 'deepthi p deepthi\_p\_220701059' are also present. The main area is titled 'SQL Commands' with a 'Language' dropdown set to 'SQL'. The query 'Drop table emp;' is entered in the command input field. Below the input field, there are icons for refresh, undo, redo, search, and a dropdown menu. The results tab is selected, showing the output: 'Table dropped.' and a execution time of '0.07 seconds'.

6.Rename the EMPLOYEES2 table as EMP.

**QUERY:**

```
Rename employees2 to emp;
```

**OUTPUT:**

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. A search bar and user profile 'deepthi p deepthi\_p\_220701059' are visible. The main area is titled 'SQL Commands' with a 'Language' dropdown set to 'SQL'. The query 'Rename employees2 to emp;' is entered in the command input field. Below the input field, there are icons for refresh, undo, redo, search, and a dropdown menu. The results tab is selected, showing the output: 'Statement processed.' and a execution time of '0.04 seconds'.

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

**QUERY:**

```
comment on table dept is 'Department info';
comment on table emp is Employee info';
```

**OUTPUT:**

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', 'Gallery', a search bar, and a user profile for 'deepthi p'. The main area is titled 'SQL Commands'. It shows the following command in the editor:

```
1 comment on table dept is 'Department info'
```

The results tab shows the output:

```
Statement processed.  
0.02 seconds
```

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

**QUERY:**

```
Alter table emp drop column first_name;
```

**OUTPUT:**

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', 'Gallery', a search bar, and a user profile for 'deepthi p'. The main area is titled 'SQL Commands'. It shows the following command in the editor:

```
1 Alter table emp drop column first_name;
```

The results tab shows the output:

```
Table altered.  
0.05 seconds
```

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

**RESULT:**

# MANIPULATING DATA

EX\_NO:2

DATE:

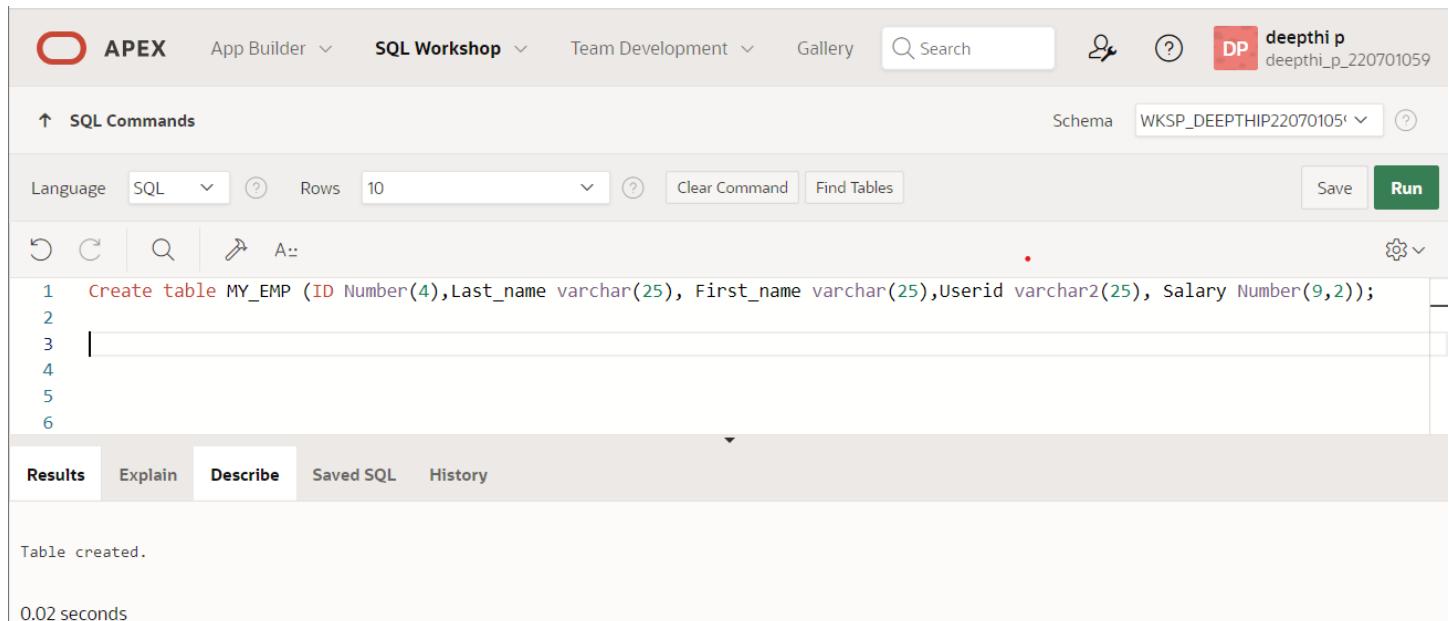
1.Create MY\_EMPLOYEE table with the following structure

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_EMP (ID Number(4),Last_name varchar(25), First_name varchar(25),Userid varchar2(25), Salary Number(9,2));
```

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, Gallery, a search bar, and a user profile for 'deepthi p'. The main area is titled 'SQL Commands' and shows the schema 'WKSP\_DEEPTHIP22070105'. The SQL editor contains the following code:

```
1 Create table MY_EMP (ID Number(4),Last_name varchar(25), First_name varchar(25),Userid varchar2(25), Salary Number(9,2));
```

The 'Results' tab at the bottom shows the output:

```
Table created.
```

Execution time: 0.02 seconds

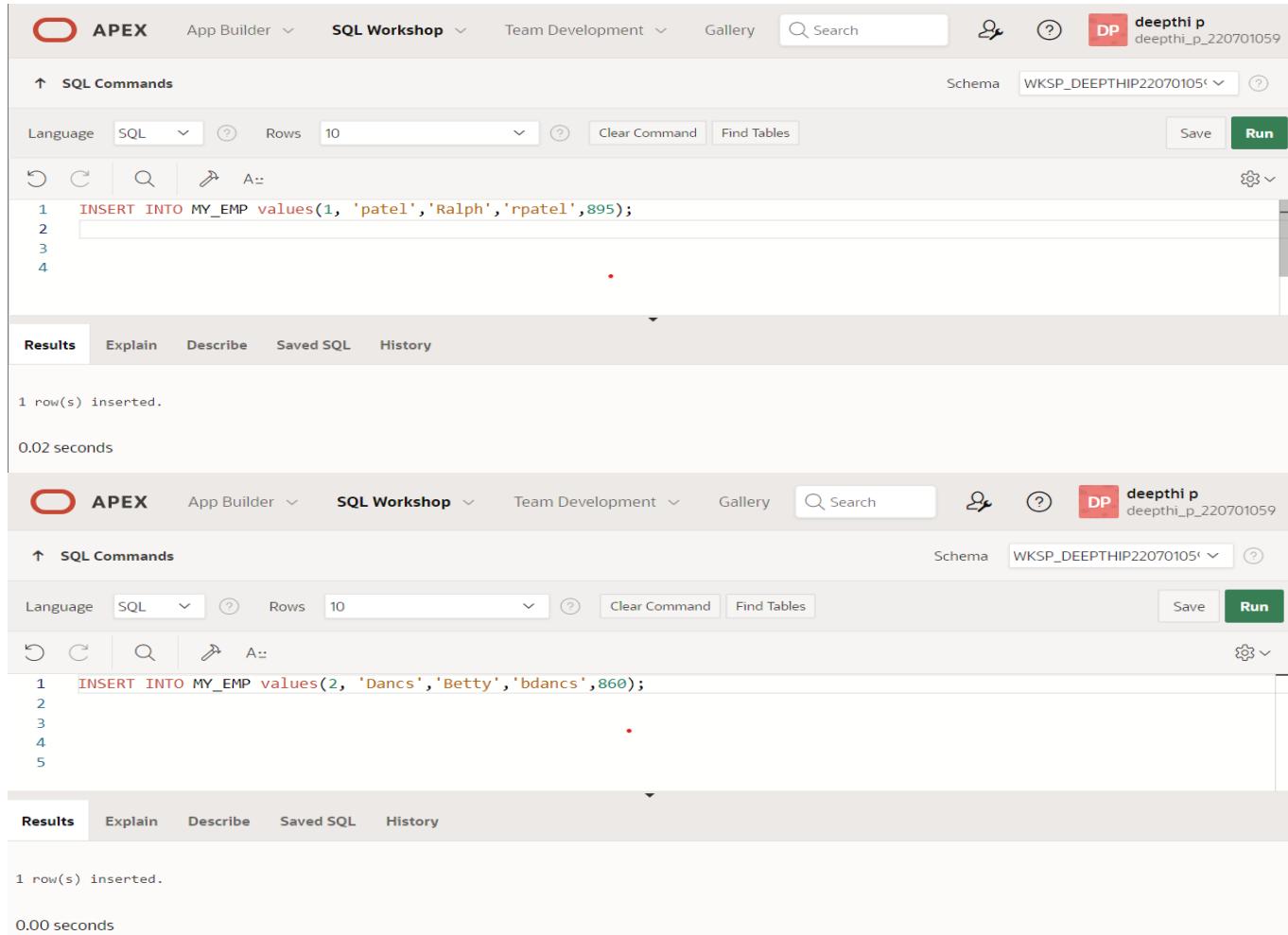
2.Add the first and second rows data to MY\_EMPLOYEE table from the following sample data.

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_EMP values(1, 'patel','Ralph','rpatel',895);
INSERT INTO MY_EMP values(2, 'Dancs','Betty','bdancs',860); OUTPUT
```

## OUTPUT:



The image shows two separate sessions in the Oracle SQL Workshop interface. Both sessions are titled "SQL Workshop" and belong to the schema "WKSP\_DEEPTHIP22070105". The top session has a user icon and the name "deepthi p" next to it. The bottom session also has a user icon and the same name. Both sessions have the same SQL command entered:

```
1 INSERT INTO MY_EMP values(1, 'patel','Ralph','rpatel',895);
2
3
4
5
```

After running the command, both sessions show the output:

1 row(s) inserted.  
0.02 seconds

1 row(s) inserted.  
0.00 seconds

3. Display the table with values.

**QUERY:**

Select \* from MY\_EMP;

**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 'deepthi p'.

In the SQL Commands section, the schema is set to 'WKSP\_DEEPTHIP22070105'. The query 'Select \* from MY\_EMP;' is entered in the command editor. The 'Run' button is highlighted in green.

The Results tab is selected, displaying the following data:

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

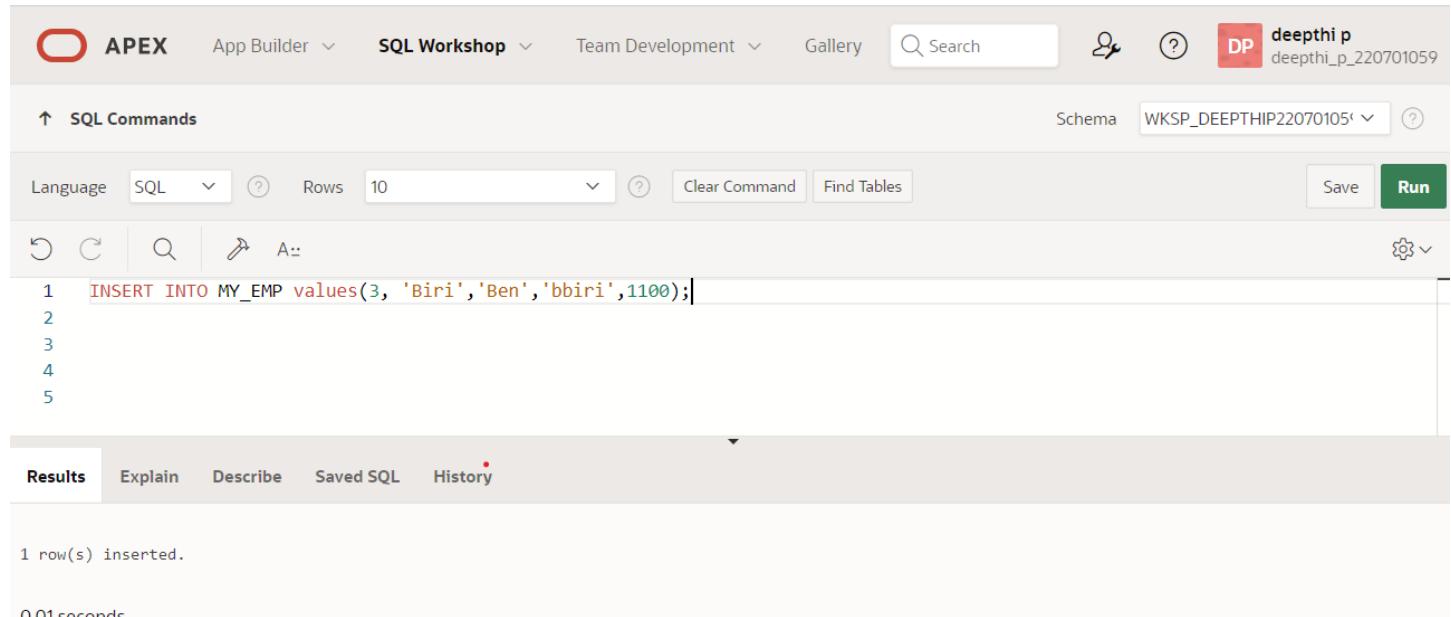
Below the table, a message indicates '2 rows returned in 0.01 seconds' and a 'Download' link is available.

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_EMP values(3, 'Biri','Ben','bbiri',1100);
INSERT INTO MY_EMP values(4, 'Newman','Chad','Cnewman',750);
```

### OUTPUT:



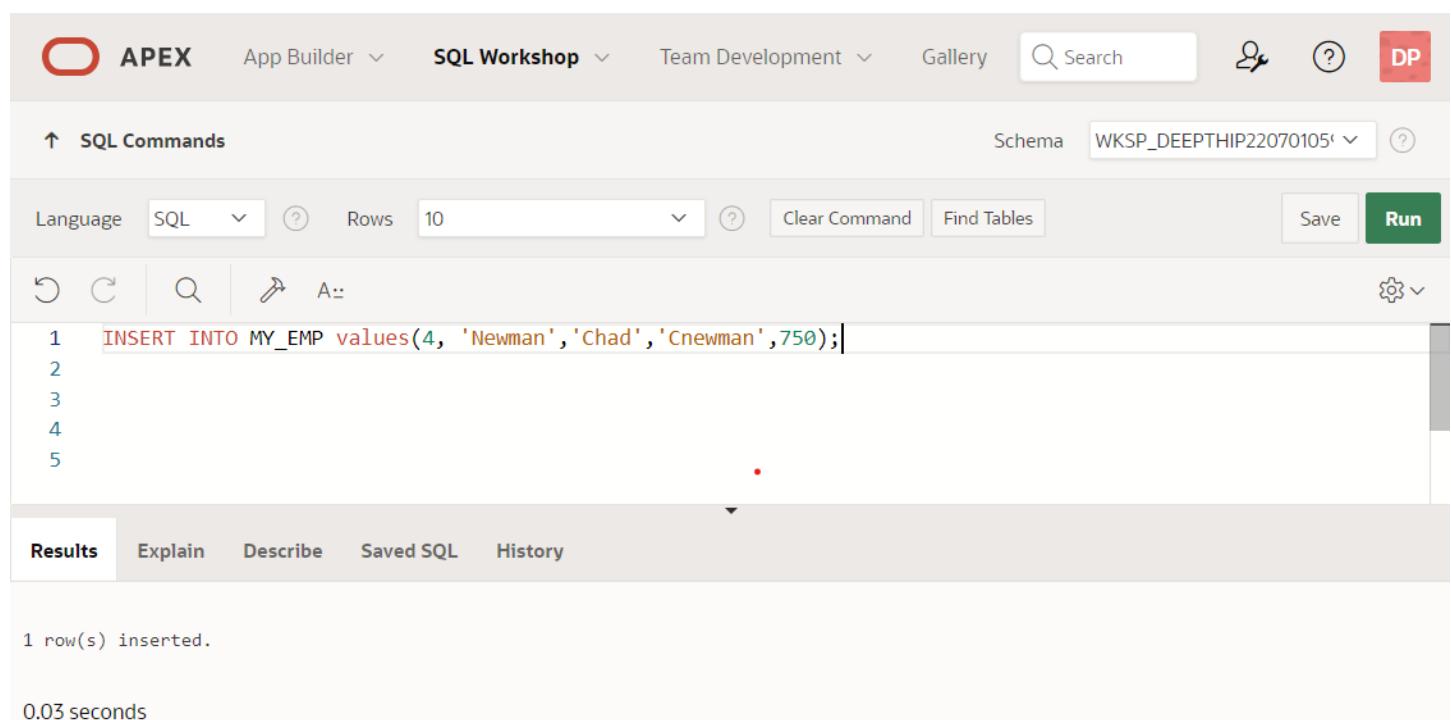
The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, Gallery, a search bar, and user information for 'deepthi p' (deepthi\_p\_220701059). The SQL Commands tab is selected. The schema is set to 'WKSP\_DEEPTHIP22070105'. The SQL editor contains the following code:

```
1 INSERT INTO MY_EMP values(3, 'Biri','Ben','bbiri',1100);
2
3
4
5
```

The results section shows the output of the query:

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

Execution time: 0.01 seconds.



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 user information for 'deepthi p' (deepthi\_p\_220701059). The SQL Commands tab is selected. The schema is set to 'WKSP\_DEEPTHIP22070105'. The SQL editor contains the following code:

```
1 INSERT INTO MY_EMP values(4, 'Newman','Chad','Cnewman',750);
2
3
4
5
```

The results section shows the output of the query:

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

Execution time: 0.03 seconds.

5. Make the data additions permanent.

**QUERY:**

```
SELECT * from MY_EMP  
Order by id;
```

**OUTPUT:**

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, Gallery, and a search bar. The user is logged in as 'deepthi p' with schema 'WKSP\_DEEPTHIP22070105'. The SQL Commands tab is active, displaying the following SQL code:

```
1 SELECT * from MY_EMP  
2 Order by id;  
3
```

The Results tab shows the output of the query:

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

Below the table, it says '4 rows returned in 0.01 seconds' and there is a 'Download' link.

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

**QUERY:**

```
UPDATE MY_EMP SET Last_Name='Drexler' where id=3;
```

**OUTPUT:**

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, Gallery, and a search bar. The user is logged in as 'deepthi p' with schema 'WKSP\_DEEPTHIP22070105'. The SQL Commands tab is active, displaying the following SQL code:

```
1 UPDATE MY_EMP SET Last_Name='Drexler' where id=3;  
2
```

The Results tab shows the output of the query:

1 row(s) updated.

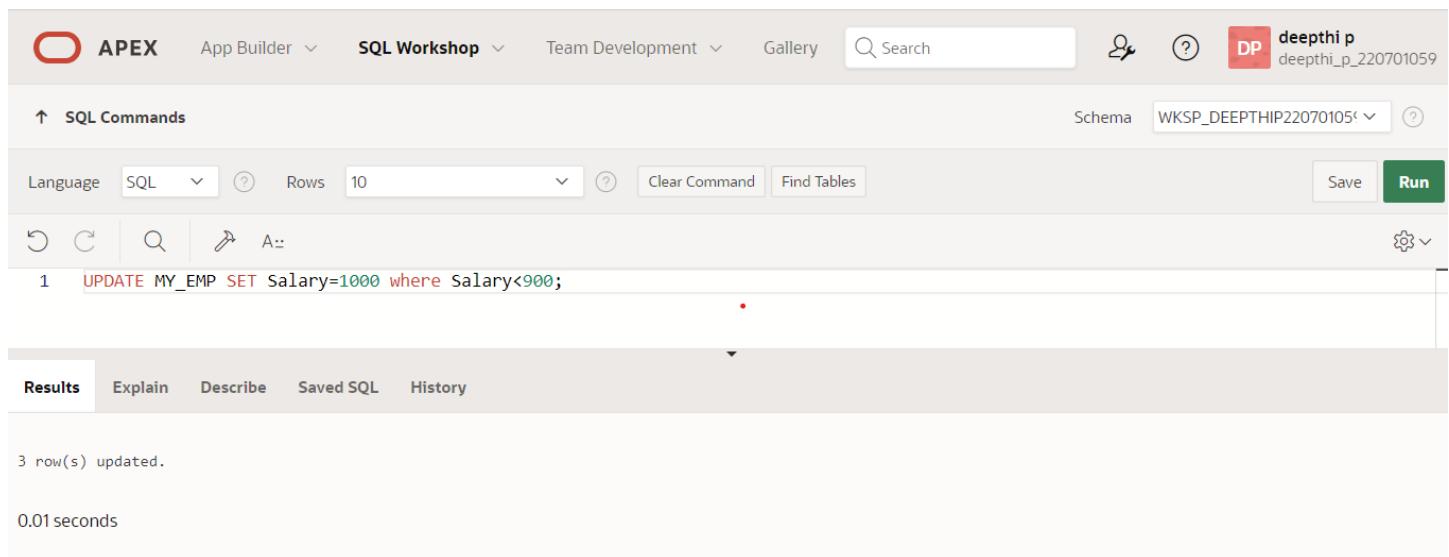
0.01 seconds

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

**QUERY:**

UPDATE MY\_EMP SET Salary=1000 where Salary<900;

**OUTPUT:**



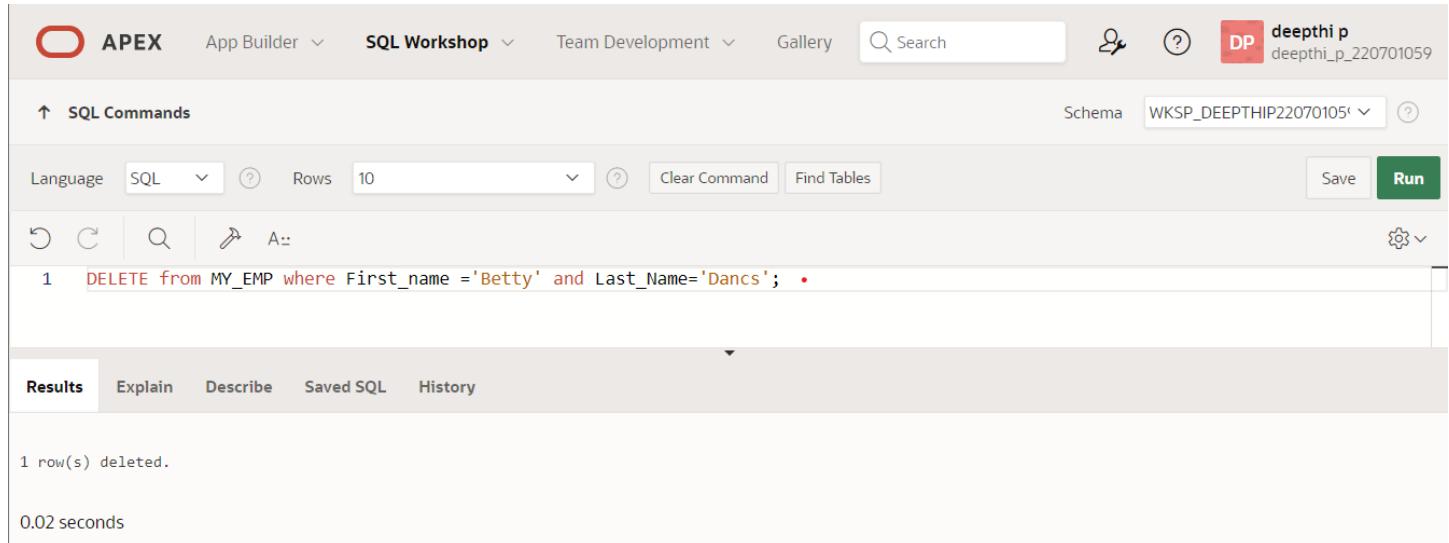
The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', 'Gallery', a search bar, and a user profile for 'deepthi p'. The 'Schema' dropdown is set to 'WKSP\_DEEPTHIP22070105'. The main area is titled 'SQL Commands' with tabs for 'Language' (set to 'SQL'), 'Rows' (set to 10), 'Clear Command', 'Find Tables', 'Save', and 'Run'. The command entered is '1 UPDATE MY\_EMP SET Salary=1000 where Salary<900;'. Below the command, the results section shows '3 row(s) updated.' and a execution time of '0.01 seconds'.

8.Delete Betty dancs from MY\_EMPLOYEE table.

**QUERY:**

DELETE from MY\_EMP where First\_name ='Betty' and Last\_Name='Dancs';

**OUTPUT:**



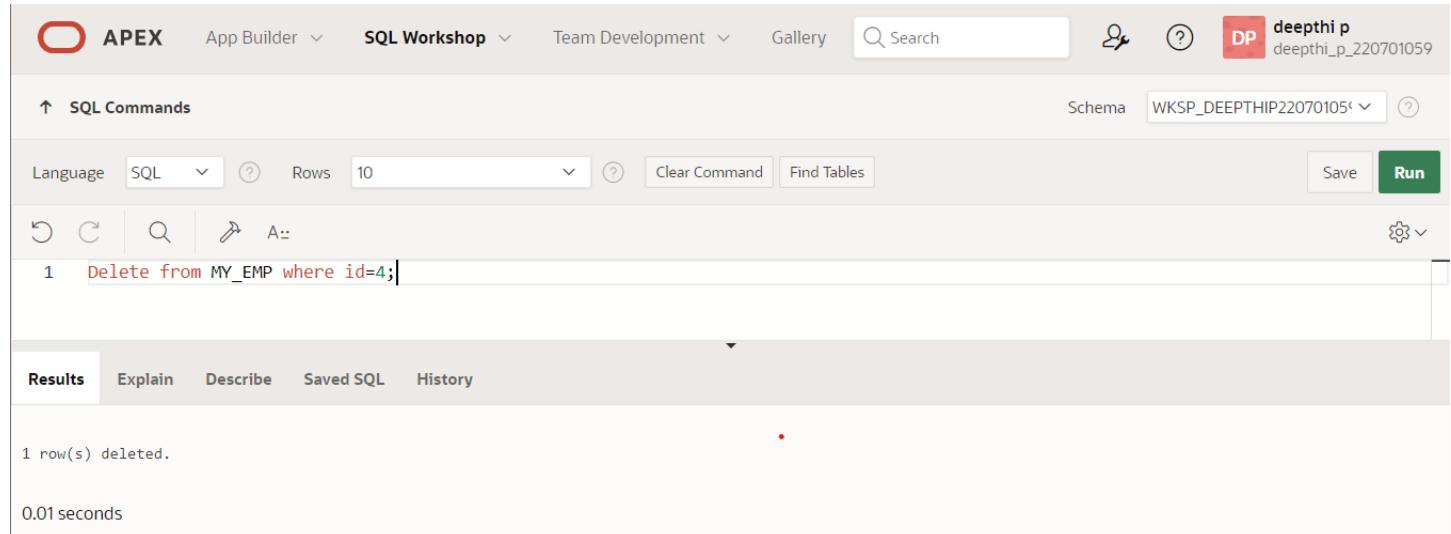
The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', 'Gallery', a search bar, and a user profile for 'deepthi p'. The 'Schema' dropdown is set to 'WKSP\_DEEPTHIP22070105'. The main area is titled 'SQL Commands' with tabs for 'Language' (set to 'SQL'), 'Rows' (set to 10), 'Clear Command', 'Find Tables', 'Save', and 'Run'. The command entered is '1 DELETE from MY\_EMP where First\_name ='Betty' and Last\_Name='Dancs';'. Below the command, the results section shows '1 row(s) deleted.' and a execution time of '0.02 seconds'.

9.Empty the fourth row of the emp table.

**QUERY:**

Delete from MY\_EMP where id=4;

**OUTPUT:**



The screenshot shows the Oracle SQL Workshop interface. In the top navigation bar, 'APEX' is selected. The main area is titled 'SQL Commands'. The language is set to 'SQL'. A single command is entered in the text area: 'Delete from MY\_EMP where id=4;'. Below the results, it shows '1 row(s) deleted.' and a execution time of '0.01 seconds'.

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

**RESULT:**

# INCLUDING CONSTRAINTS

**EX\_NO:3**

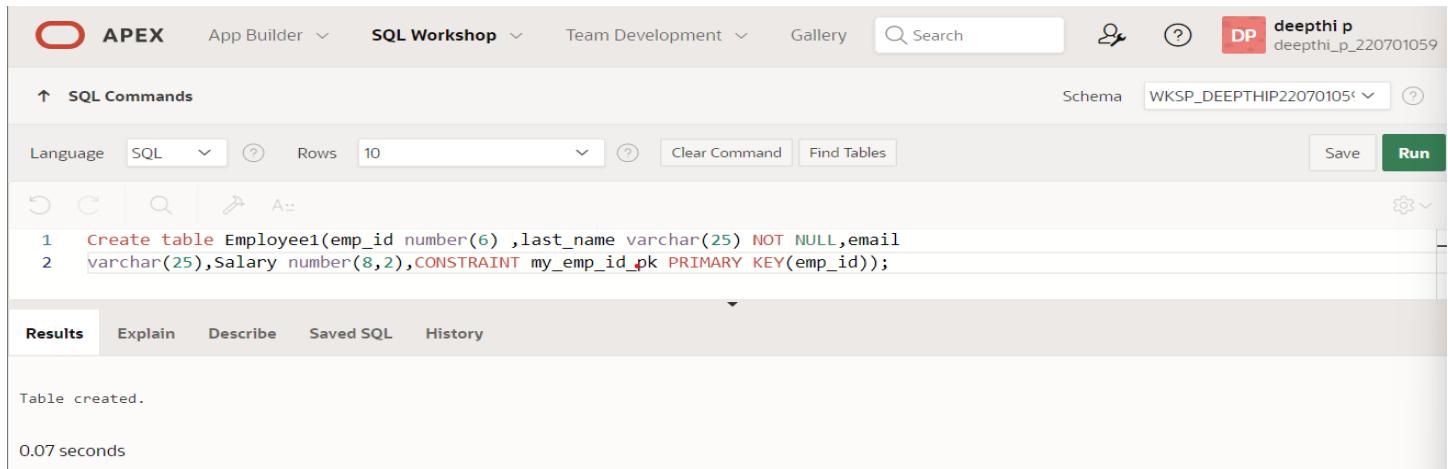
**DATE:**

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 Employee1(emp_id number(6) ,last_name varchar(25) NOT NULL,email  
varchar(25),Salary number(8,2),CONSTRAINT my_emp_id_pk PRIMARY KEY(emp_id));
```

## OUTPUT:



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

```
1 Create table Employee1(emp_id number(6) ,last_name varchar(25) NOT NULL,email  
2      varchar(25),Salary number(8,2),CONSTRAINT my_emp_id_pk PRIMARY KEY(emp_id));
```

The Results tab displays the output:

```
Table created.  
0.07 seconds
```

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

#### QUERY:

```
Create table MYDEPT (dept_id number(6) ,first_name varchar(25) NOT NULL,email varchar(25),Salary number(8,2),CONSTRAINT my_dept_id_pk PRIMARY KEY(dept_id));
```

#### 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 'deepthi p'. The SQL Workshop tab is selected. The schema dropdown shows 'WKSP\_DEEPTHIP22070105'. The main area is titled 'SQL Commands' and contains the following SQL code:

```
1 Create table MYDEPT (dept_id number(6) ,first_name varchar(25) NOT NULL,email varchar(25),Salary number(8,2)
2 ,CONSTRAINT my_dept_id_pk PRIMARY KEY(dept_id));
```

Below the code, the results section displays the message 'Table created.' and a execution time of '0.07 seconds'.

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 Employee1 ADD DEPT1_ID number(6); alter table Employee1 ADD CONSTRAINT  
my_emp_dept_id_fk FOREIGN KEY(dept1_id) REFERENCES Employee1(emp_id);
```

#### 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 'deepthi p'. The SQL Workshop tab is selected. The schema dropdown shows 'WKSP\_DEEPTHIP22070105'. The main area is titled 'SQL Commands' and contains the following SQL code:

```
1 ALTER TABLE Employee1 ADD DEPT1_ID number(6); alter table Employee1 ADD CONSTRAINT
2 my_emp_dept_id_fk FOREIGN KEY(dept1_id) REFERENCES Employee1(emp_id);
```

Below the code, the results section displays the message 'Table created.' and a execution time of '0.07 seconds'.

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 Employee1 ADD COMMISSION number (2,2) check (COMMISSION > 0);
```

#### OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, Gallery, a search bar, and user information (DP deepthi p deepthi\_p\_220701059). The main area is titled 'SQL Commands' with a sub-section 'Language: SQL'. It displays the following SQL command:

```
1 alter table Employee1 ADD COMMISSION number (2,2) check (COMMISSION > 0);
```

Below the command, the results section shows the output:

```
Table created.
```

Execution time: 0.07 seconds

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

#### RESULT:

# Writing Basic SQL SELECT Statements

EX\_NO:4

DATE:

- 1.The following statement executes successfully.

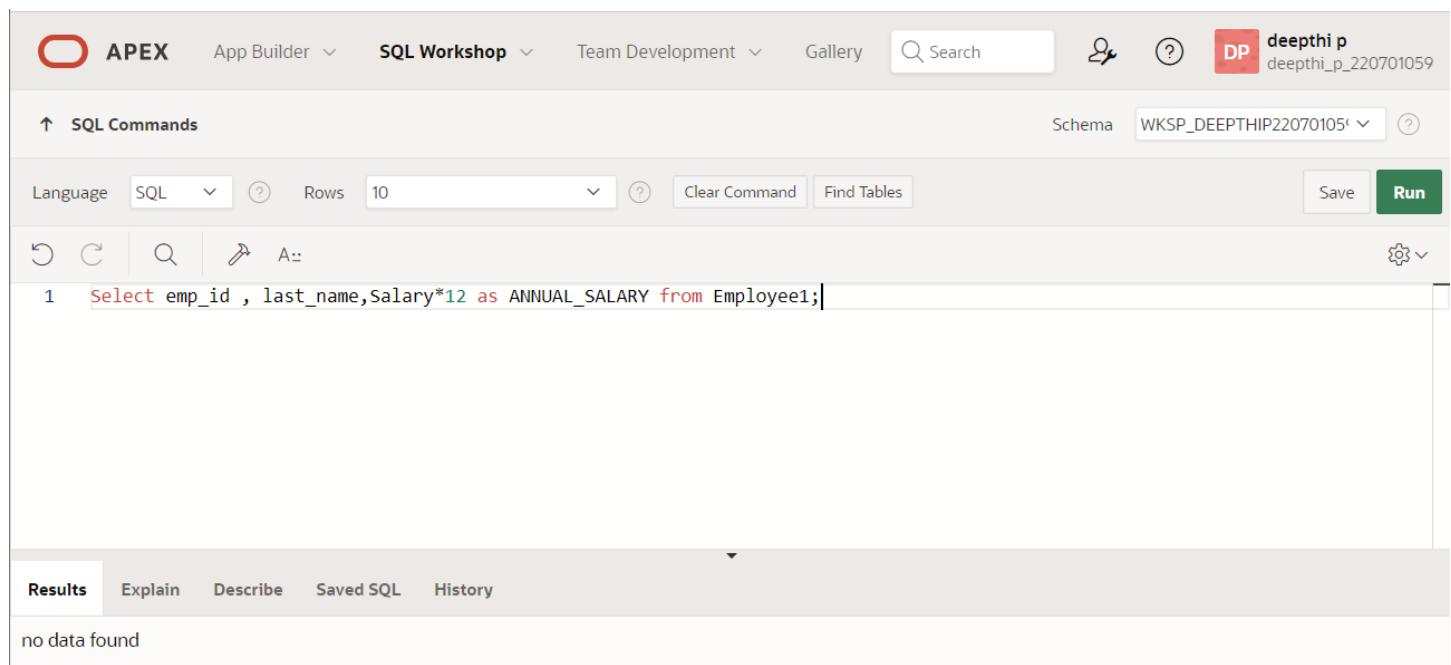
## Identify the Errors

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

## QUERY:

```
Select emp_id , last_name,Salary*12 as ANNUAL_SALARY from Employee1;
```

## OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, Gallery, a search bar, and a user profile for 'deepthi p'. The SQL Commands tab is active. The query editor window contains the following command:

```
1 Select emp_id , last_name,Salary*12 as ANNUAL_SALARY from Employee1;
```

The results section below shows the message "no data found".

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

**QUERY:**

Create table DEPARTMENT (dept\_id number(6), first\_name varchar(25) NOT NULL ,email varchar(25),Salary

number(8,2));

DESC DEPARTMENT;

**OUTPUT:**

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', 'Gallery', and a search bar. The user is connected to 'deepthi\_p\_220701059'. The main area is titled 'SQL Commands' with a schema dropdown set to 'WKSP\_DEEPTHIP220701059'. The command entered is:

```
1 Create table DEPARTMENT (dept_id number(6), first_name varchar(25) NOT
2 NULL ,email varchar(25),Salary number(8,2));
3 
```

The results tab shows the message 'Table created.' and a execution time of '0.04 seconds'.

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', 'Gallery', and a search bar. The user is connected to 'deepthi\_p\_220701059'. The main area is titled 'SQL Commands' with a schema dropdown set to 'WKSP\_DEEPTHIP220701059'. The command entered is:

```
1 DESC DEPARTMENT;
```

The results tab shows the table structure for 'DEPARTMENT' with columns: DEPT\_ID, FIRST\_NAME, EMAIL, and SALARY. The 'Describe' tab is selected.

Object Type	TABLE	Object	DEPARTMENT						
Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
DEPARTMENT	DEPT_ID	NUMBER	-	6	0	-	✓	-	-
	FIRST_NAME	VARCHAR2	25	-	-	-	-	-	-
	EMAIL	VARCHAR2	25	-	-	-	✓	-	-
	SALARY	NUMBER	-	8	2	-	✓	-	-

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:**

```
Create table Employees(emp_id number(6), job_id varchar(25) NOT NULL,email varchar(25),Salary number(8,2),hire_date date);
```

```
Insert into Employees values(1,'AAA','abc@gmail.com',10000,01/03/2024);
```

```
SELECT emp_id, last_name, job_id, hire_date from Employees;
```

**OUTPUT:**

The screenshot shows two sessions in the Oracle SQL Workshop interface. Both sessions have the same schema: WKSP\_DEEPTHIP22070105.

**Session 1 (Top):**

- SQL Commands tab selected.
- SQL language selected.
- Rows set to 10.
- Command:

```
1 Create table Employees(emp_id number(6), job_id varchar(25) NOT NULL,
2 email varchar(25),Salary number(8,2),hire_date date);
```
- Results tab selected.
- Output:

```
Table created.
```
- Time taken: 0.04 seconds.

**Session 2 (Bottom):**

- SQL Commands tab selected.
- SQL language selected.
- Rows set to 10.
- Command:

```
1 Insert into Employees values(1,'AAA','abc@gmail.com',10000,2024-01-03);
2 |
```
- Results tab selected.
- Output:

```
1 row(s) inserted.
```
- Time taken: 0.05 seconds.

The screenshot shows the Oracle APEX SQL Workshop interface. At the top, there are tabs for App Builder, SQL Workshop, Team Development, and Gallery. On the right, there's a search bar, user profile, and schema information (WKSP\_DEEPTHIP22070105). Below the tabs, the SQL Commands section has a language dropdown set to SQL, row limit set to 10, and buttons for Clear Command and Find Tables. The main area contains a single line of SQL code: "1 SELECT emp\_id, last\_name, job\_id, hire\_date from Employees;". The Results tab is selected, showing a table with four columns: EMP\_ID, LAST\_NAME, JOB\_ID, and HIRE\_DATE. One row is returned, with values 1, Smith, AAA, and 2020 respectively. A message at the bottom says "1 rows returned in 0.01 seconds" and includes a Download link.

EMP_ID	LAST_NAME	JOB_ID	HIRE_DATE
1	Smith	AAA	2020

4. Provide an alias STARTDATE for the hire date.

**QUERY:**

Select Hire\_date as STARTDATE from Employees;

**OUTPUT:**

The screenshot shows the Oracle APEX SQL Workshop interface. The setup is identical to the previous one, with the SQL Commands section containing the query "1 Select Hire\_date as STARTDATE from Employees;". The Results tab is selected, displaying a table with a single column named STARTDATE. The value 2020 is listed. A message at the bottom indicates "1 rows returned in 0.01 seconds" and provides a Download link.

STARTDATE
2020

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

**QUERY:**

SELECT DISTINCT job\_id from Employees;

**OUTPUT:**

The screenshot shows the Oracle SQL Workshop interface. In the top navigation bar, 'APEX' is selected. The main area is titled 'SQL Commands'. A query is entered: 'SELECT DISTINCT job\_id from Employees;'. The results tab is selected, showing a single row with the value 'AAA' under the 'JOB\_ID' column. The output is labeled 'EMPLOYEE' and 'TITLE'.

JOB_ID
AAA

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

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_id AS "EMPLOYEE and TITLE" FROM Employees;
```

#### OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. A query is entered: 'Select last\_name||', '||job\_id AS "EMPLOYEE and TITLE" FROM Employees;'. The results tab is selected, showing a single row with the value 'Smith, AAA' under the 'EMPLOYEE and TITLE' column. The output is labeled 'EMPLOYEE and TITLE'.

EMPLOYEE and TITLE
Smith, AAA

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

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 emp_id||'.'|| last_name||','||job_id|| email||','||Salary||','|| Hire_date AS "THE OUTPUT" FROM Employees;
```

**OUTPUT:**

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

```
1 Select emp_id||'.'|| last_name||','||job_id|| email||','||Salary||','|| Hire_date AS "THE OUTPUT"
2 FROM Employees;
```

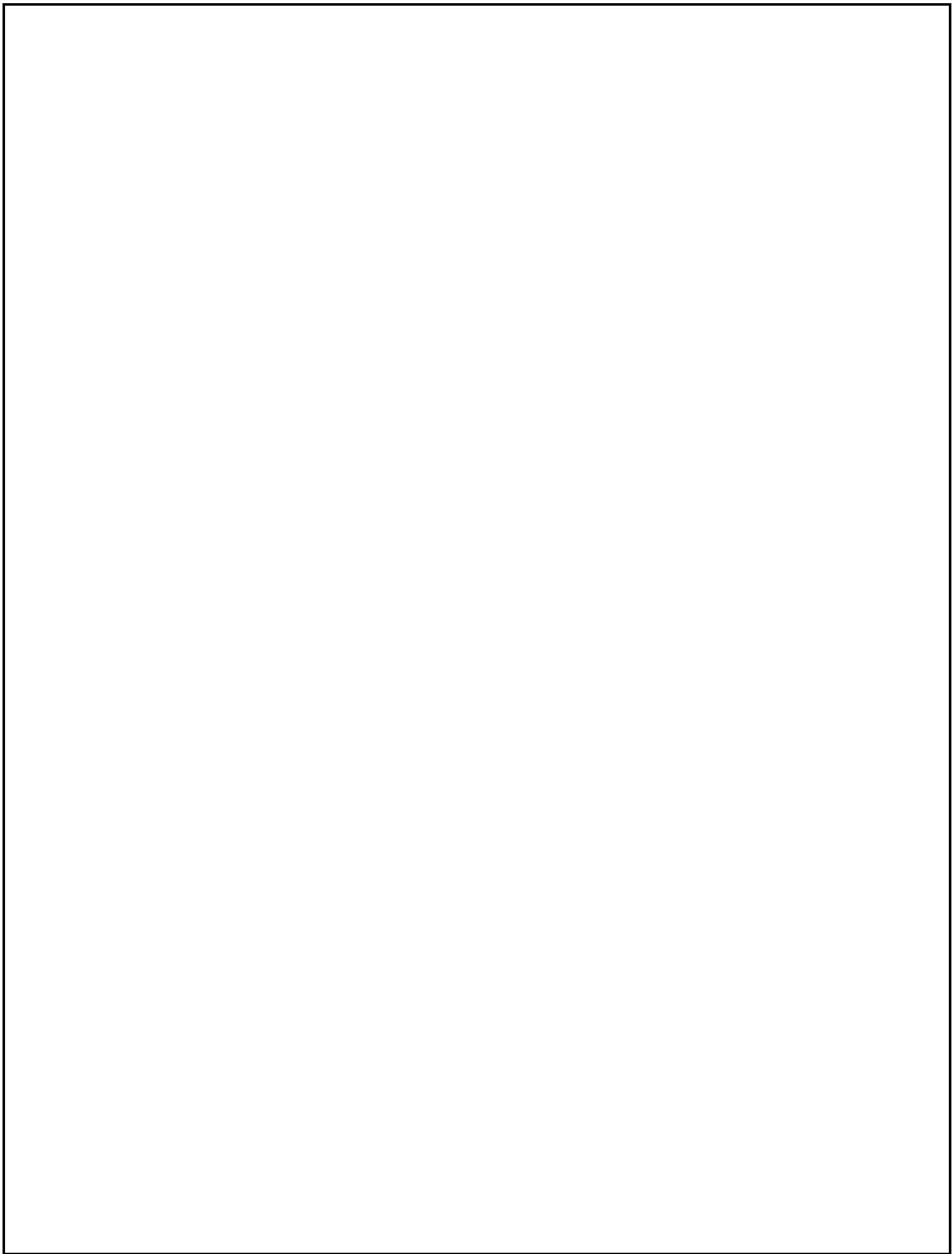
Below the command, the results are shown in a table:

THE OUTPUT	
1,Smith,AAAabc@gmail.com,10000,2020	

Text at the bottom indicates "1 rows returned in 0.01 seconds" and a "Download" link.

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

**RESULT:**



# **RESTRICTING AND SORTING DATA**

**EX.NO.5**

**DATE:**

Find the Solution for the following:

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

**QUERY:**

Select last\_name ,Salary from Employee where salary>12000

**OUTPUT:**

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

```
1 select last_name,salary from employee
2 where salary>12000;
```

The results table displays the following data:

LAST_NAME	SALARY
Snow	30000
Martin	14000
Smith	20000

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

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

**QUERY:**

Select last\_name, dept\_no from Employee where id=176;

**OUTPUT:**

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

```
1 Select last_name,dept_no from employee
2 where id=176;
```

The results table displays the following data:

LAST_NAME	DEPT_NO
Smith	11

1 rows returned in 0.01 seconds [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 Employee
```

```
Where Salary NOT BETWEEN 5000 AND 12000
```

#### OUTPUT:

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

```
1 Select last_name,salary from employee
2 where salary not between 5000 and 12000;
```

The results table displays the following data:

LAST_NAME	SALARY
Snow	30000
Martin	14000
Smith	20000

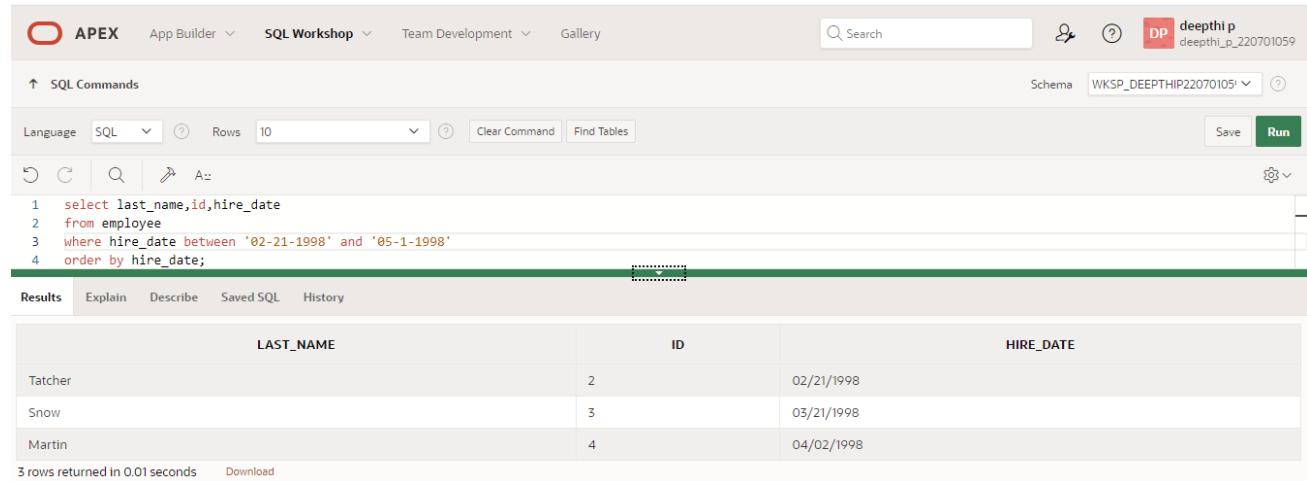
3 rows returned in 0.01 seconds    Download

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, id, hire_date from Employee  
where hire_date between 'February 20,1998' AND 'May 1,1998';
```

### OUTPUT:



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

```
1 select last_name,id,hire_date  
2 from employee  
3 where hire_date between '02-21-1998' and '05-1-1998'  
4 order by hire_date;
```

The results section displays the following data:

LAST_NAME	ID	HIRE_DATE
Tatcher	2	02/21/1998
Snow	3	03/21/1998
Martin	4	04/02/1998

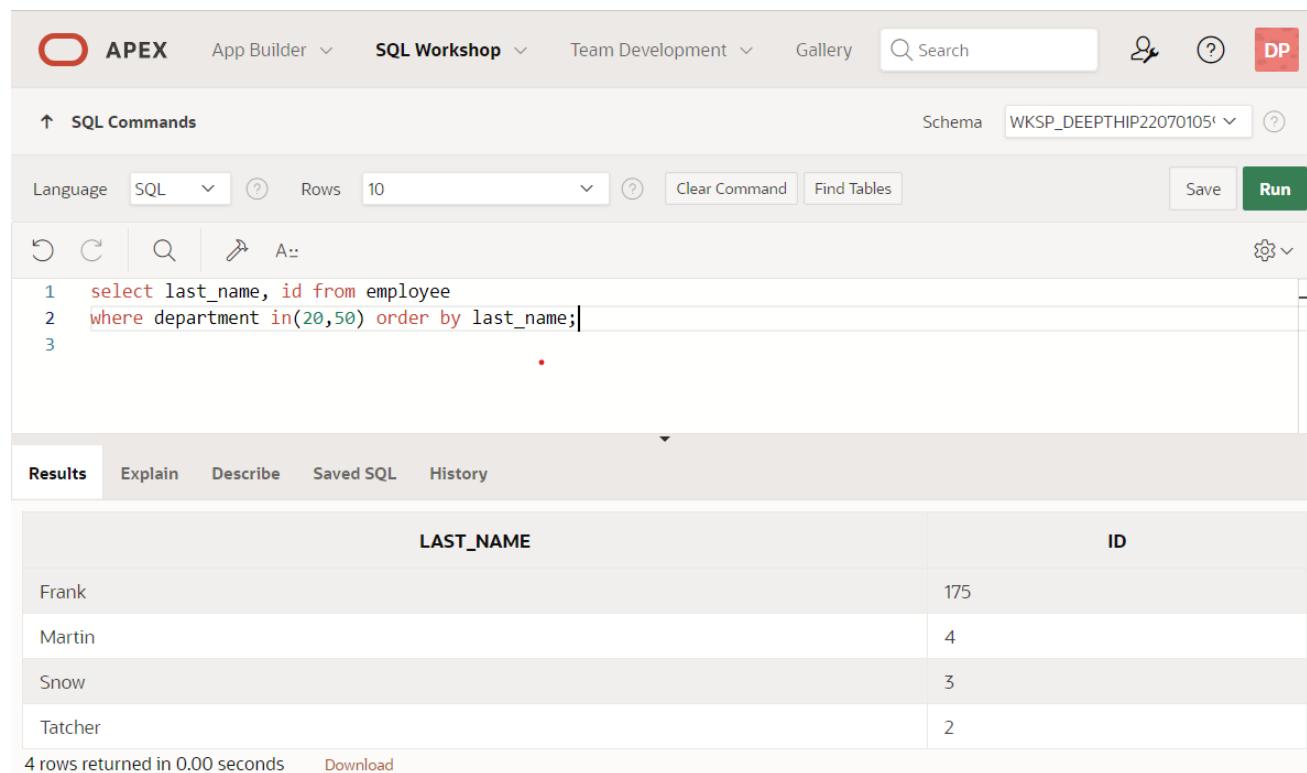
3 rows returned in 0.01 seconds. A Download button is also present.

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, id from employee where department in(20,50) order by last_name;
```

**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 user profile icons. The SQL Workshop tab is selected. The schema dropdown shows WKSP\_DEEPTHIP22070105. The main area displays the SQL command:

```
1 select last_name, id from employee
2 where department in(20,50) order by last_name;
3 .
```

The Results tab is selected, showing the output:

LAST_NAME	ID
Frank	175
Martin	4
Snow	3
Tatcher	2

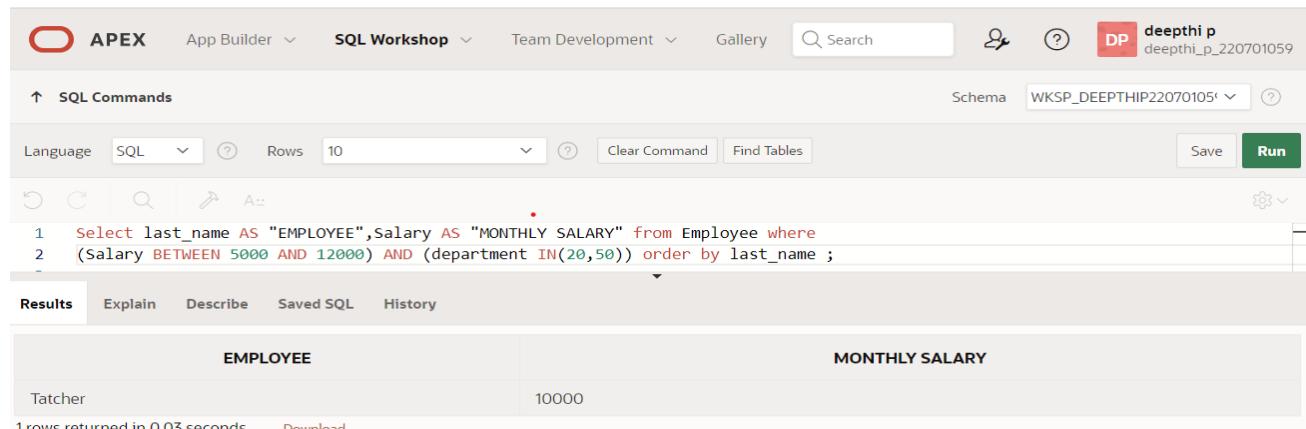
Below the table, it says "4 rows returned in 0.00 seconds" and there is a "Download" link.

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 Employee where (Salary BETWEEN 5000 AND 12000) AND (dept\_id IN(20,50)) order by last\_name ;

**OUTPUT:**



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, Gallery, and a search bar. The right side shows a user profile for 'deepthi p' with a session ID. The main area has tabs for SQL Commands, Results, Explain, Describe, Saved SQL, and History. The SQL Commands tab contains the following query:

```
1 select last_name AS "EMPLOYEE",Salary AS "MONTHLY SALARY" from Employee where
2 (Salary BETWEEN 5000 AND 12000) AND (department IN(20,50)) order by last_name ;
```

The Results tab displays the output:

EMPLOYEE	MONTHLY SALARY
Tatcher	10000

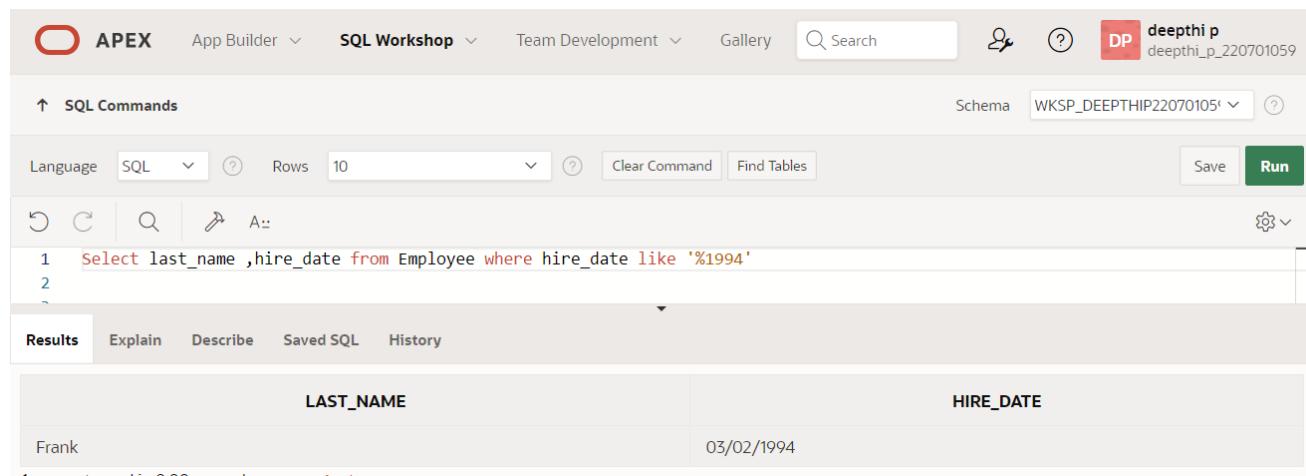
1 rows returned in 0.03 seconds [Download](#)

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 Employee where hire\_date like '%1994'

**OUTPUT:**



The screenshot shows the Oracle SQL Workshop interface, similar to the previous one. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, Gallery, and a search bar. The right side shows a user profile for 'deepthi p' with a session ID. The main area has tabs for SQL Commands, Results, Explain, Describe, Saved SQL, and History. The SQL Commands tab contains the following query:

```
1 Select last_name ,hire_date from Employee where hire_date like '%1994'
2
```

The Results tab displays the output:

LAST_NAME	HIRE_DATE
Frank	03/02/1994

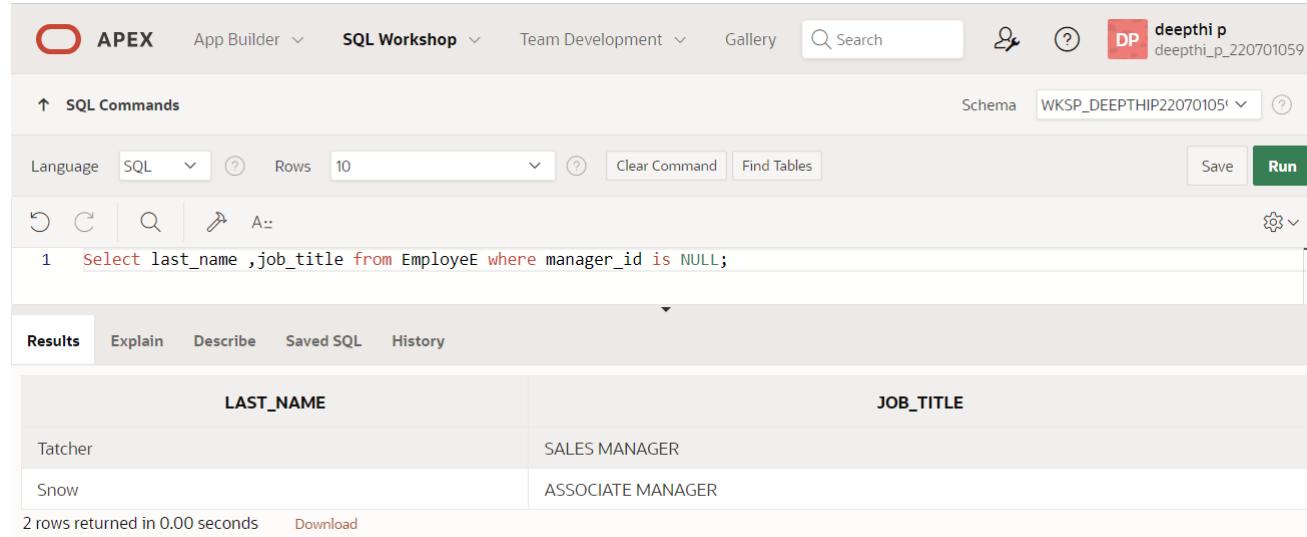
1 rows returned in 0.00 seconds [Download](#)

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

**QUERY:**

Select last\_name ,job\_title from Employee where manager\_id is NULL;

**OUTPUT:**



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', 'Gallery', a search bar, and a user profile for 'deepthi p'. The 'Schema' dropdown is set to 'WKSP\_DEEPTHIP22070105'. The main area is titled 'SQL Commands' with a sub-section '1'. The command entered is 'Select last\_name ,job\_title from Employee where manager\_id is NULL;'. Below the command, there are tabs for 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History'. The 'Results' tab is selected, displaying a table with two rows. The columns are 'LAST\_NAME' and 'JOB\_TITLE'. The data is as follows:

LAST_NAME	JOB_TITLE
Tatcher	SALES MANAGER
Snow	ASSOCIATE MANAGER

At the bottom left, it says '2 rows returned in 0.00 seconds'. There is also a 'Download' link.

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,commision from employee where commision is not null  
order by Salary,commision DESC;
```

**OUTPUT:**

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, Gallery, Search, and a user profile for 'deepthi p'. The SQL Commands section contains the following code:

```
1 Select last_name ,Salary,commision from employee where commision is not null  
2 order by Salary,commision DESC;  
3  
4 |
```

The Results tab is selected, displaying the following table output:

LAST_NAME	SALARY	COMMISION
Tatcher	10000	23
Snow	30000	34

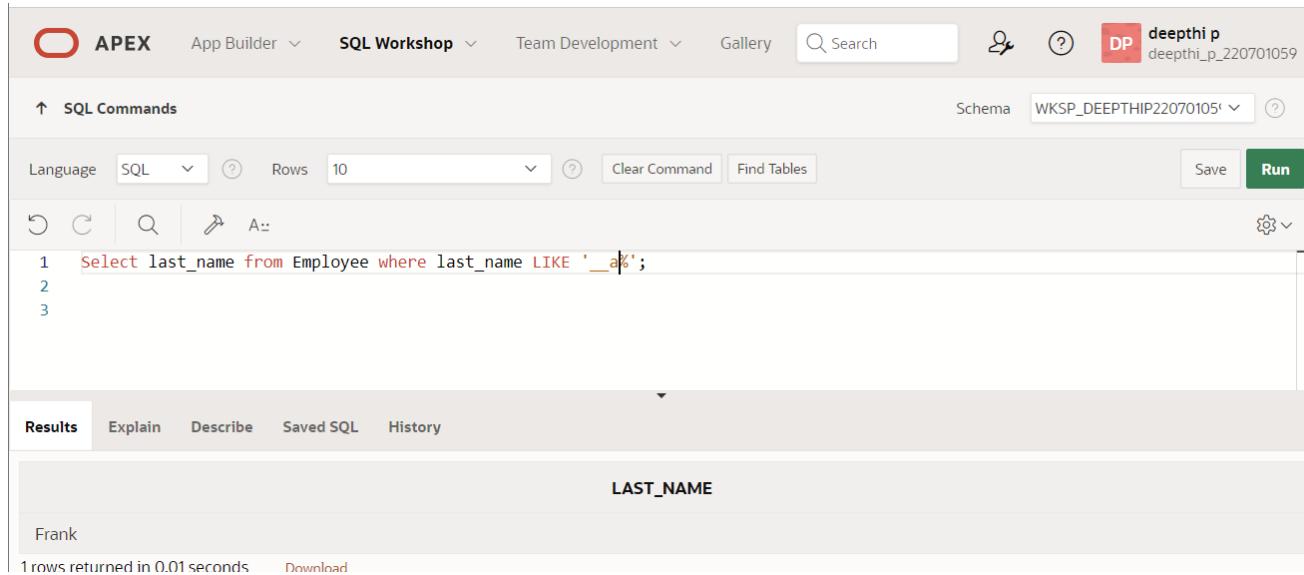
Below the table, it says '2 rows returned in 0.01 seconds' and there is a 'Download' link.

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

**QUERY:**

Select last\_name from Employee where last\_name LIKE '\_\_A%';

**OUTPUT:**



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', 'Gallery', a search bar, and user information 'deepthi p deepthi\_p\_22070105'. The main area is titled 'SQL Commands' with a schema dropdown set to 'WKSP\_DEEPTHIP22070105'. The SQL editor contains the following code:

```
1 Select last_name from Employee where last_name LIKE '__a%';
2
3
```

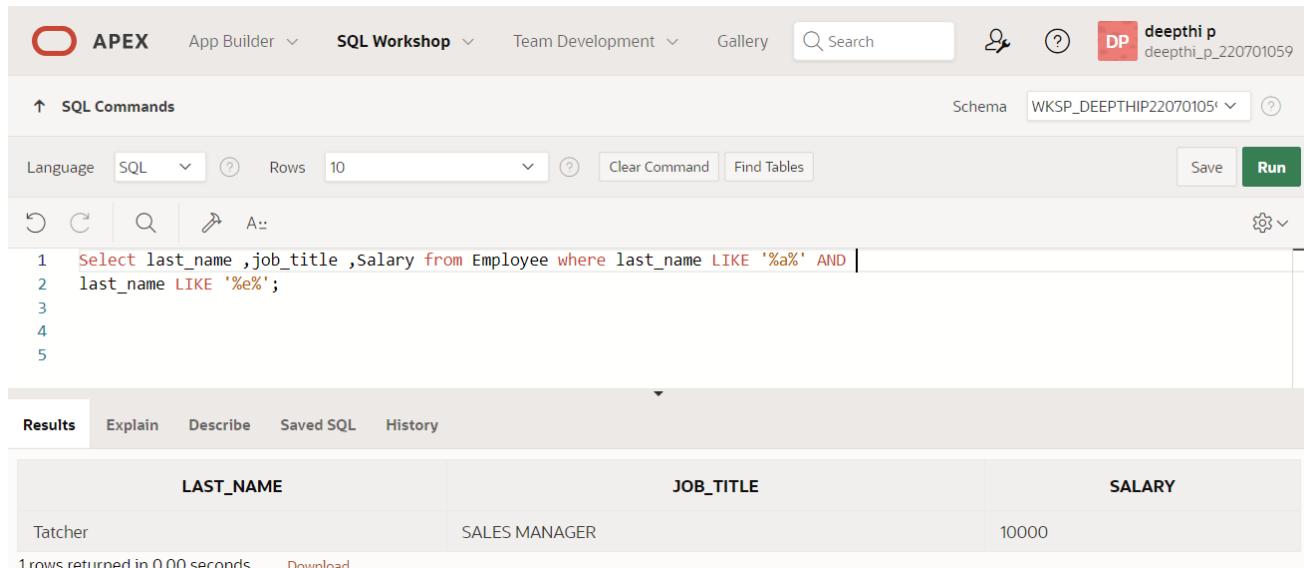
The results section shows a single row with the column 'LAST\_NAME' containing 'Frank'. Below the results, it says '1 rows returned in 0.01 seconds'.

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 ,job_title ,Salary from Employee where last_name LIKE '%A%' AND  
last_name LIKE '%E%';
```

**OUTPUT:**



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, Gallery, a search bar, and a user profile for 'deepthi p'.

In the SQL Commands section, the schema is set to 'WKSP\_DEEPTHIP22070105'. The code entered is:

```
1 Select last_name ,job_title ,Salary from Employee where last_name LIKE '%A%' AND  
2 last_name LIKE '%E%';  
3  
4  
5
```

The 'Run' button is highlighted in green. Below the code, the results tab is selected, showing the output:

LAST_NAME	JOB_TITLE	SALARY
Tatcher	SALES MANAGER	10000

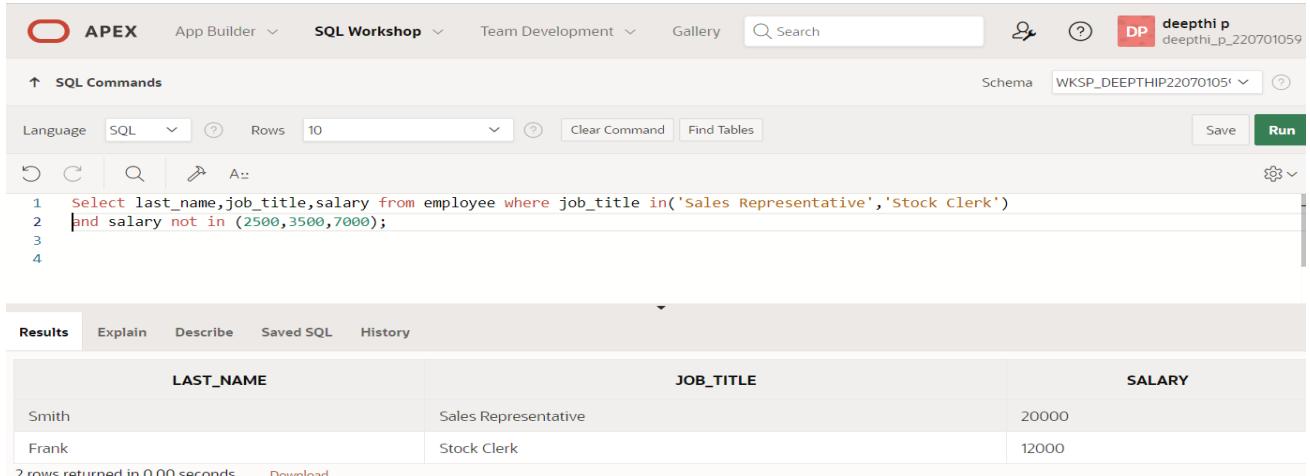
1 rows returned in 0.00 seconds [Download](#)

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

#### QUERY:

Select last\_name,job\_title,salary from employee where job\_title in('Sales Representative','Stock Clerk') and salary not in (2500,3500,7000);

#### OUTPUT:



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

```
1 select last_name,job_title,salary from employee where job_title in('Sales Representative','Stock Clerk')
2 and salary not in (2500,3500,7000);
3
4
```

The Results pane displays the following output:

LAST_NAME	JOB_TITLE	SALARY
Smith	Sales Representative	20000
Frank	Stock Clerk	12000

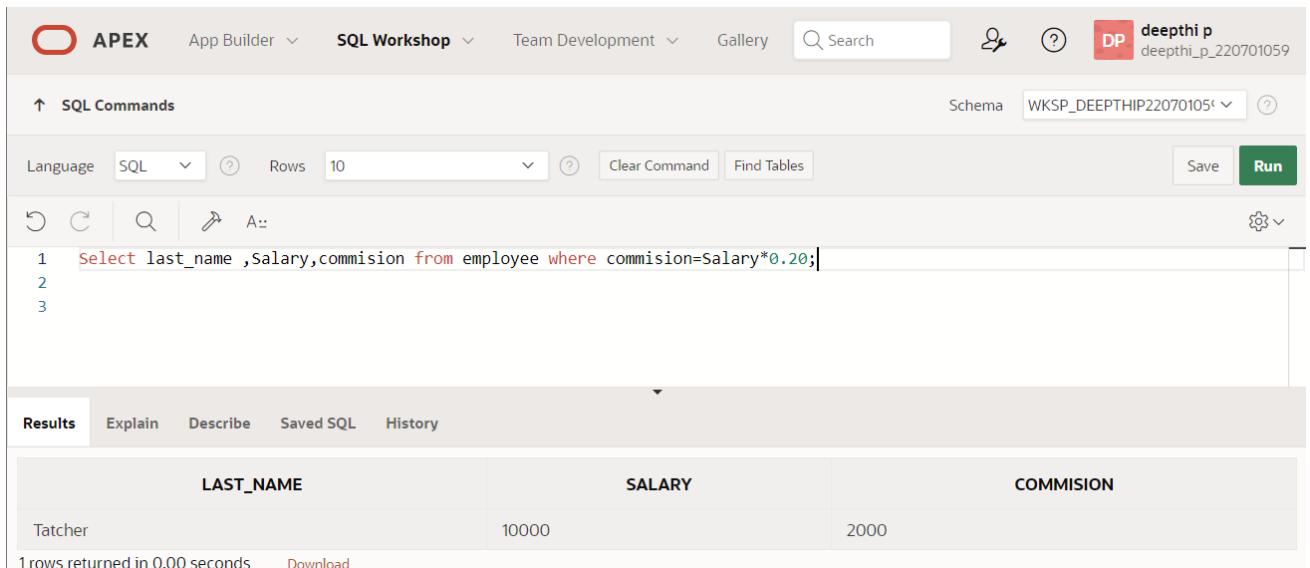
2 rows returned in 0.00 seconds

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

#### QUERY:

Select last\_name ,Salary,commission from employee where commission=Salary\*0.20;

#### OUTPUT:



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

```
1 Select last_name ,Salary,commission from employee where commision=Salary*0.20;
2
3
```

The Results pane displays the following output:

LAST_NAME	SALARY	COMMISION
Tatcher	10000	2000

1 rows returned in 0.00 seconds

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

## **RESULT:**

# SINGLE ROW FUNCTIONS

**EX.NO.6**

**DATE:**

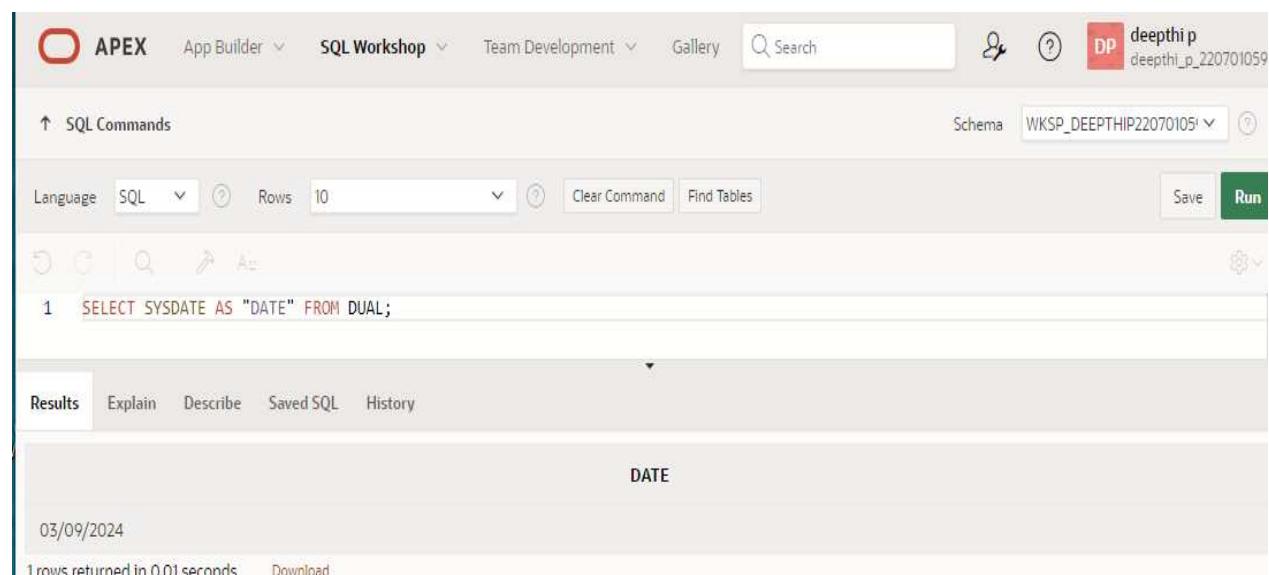
**Find the Solution for the following:**

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

**QUERY:**

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

**OUTPUT:**



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, Gallery, a search bar, and a user profile for 'deepthi p' (deepthi\_p\_220701059). The main workspace is titled 'SQL Commands'. It features a toolbar with icons for Undo, Redo, Find, Replace, and Save. Below the toolbar, the command 'SELECT SYSDATE AS "DATE" FROM DUAL;' is entered in the command input field. The results tab is selected, displaying a single row with the column 'DATE' containing the value '03/09/2024'. The status bar at the bottom indicates '1 rows returned in 0.01 seconds' and provides a 'Download' link.

DATE
03/09/2024

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 id, last_name, Salary, Salary+(15.5/100*Salary) "NEW_SALARY" From Employee;
```

**OUTPUT:**

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, Gallery, a search bar, and a user profile for 'deepthi p'.

In the SQL Commands pane, the query is displayed:

```
1 select id, last_name, salary, salary+(salary*15.5/100) as "NEW SALARY" from employee;
```

The Results pane displays the output of the query:

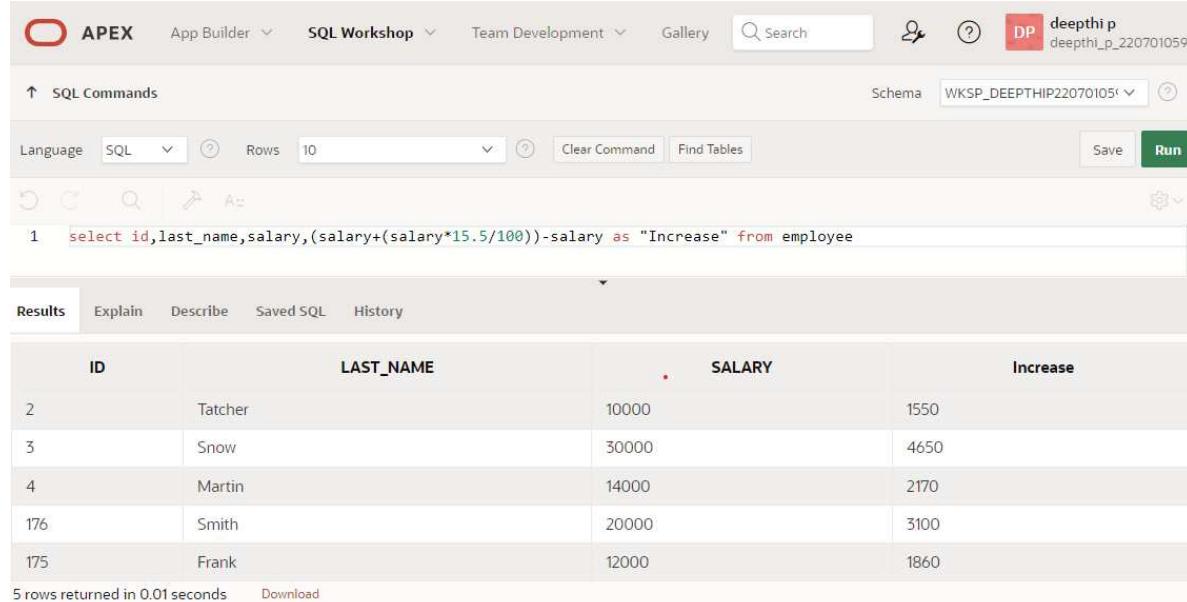
ID	LAST_NAME	SALARY	NEW SALARY
2	Tatcher	10000	11550
3	Snow	30000	34650
4	Martin	14000	16170
176	Smith	20000	23100
175	Frank	12000	13860

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 e_id, last_name, Salary, (Salary+(Salary*15.5/100))-Salary "Increase"  
From Employee;
```

**OUTPUT:**



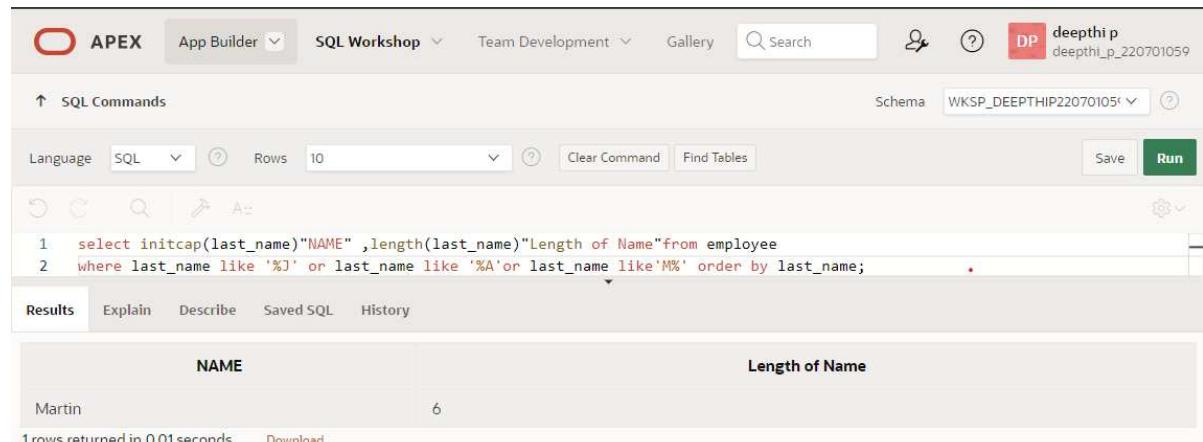
ID	LAST_NAME	SALARY	Increase
2	Tatcher	10000	1550
3	Snow	30000	4650
4	Martin	14000	2170
176	Smith	20000	3100
175	Frank	12000	1860

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

**QUERY:**

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

**OUTPUT:**



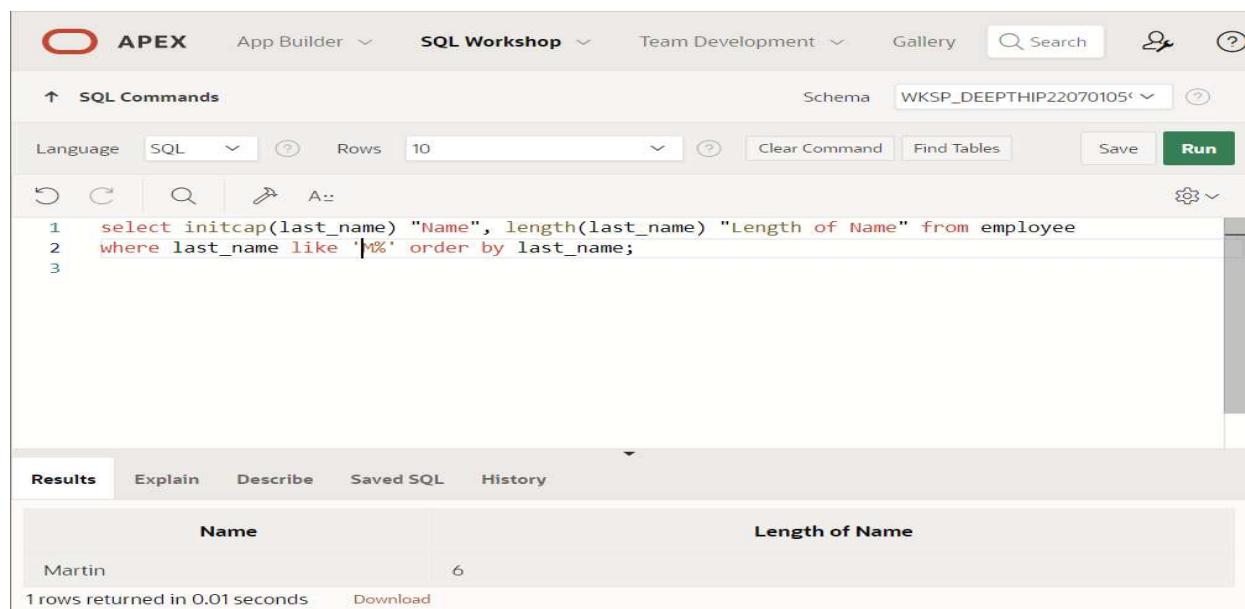
NAME	Length of Name
Martin	6

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

**QUERY:**

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

**OUTPUT:**



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', 'Gallery', and a search bar. The 'Schema' dropdown is set to 'WKSP\_DEEPTHIP22070105'. The main area is titled 'SQL Commands' with a sub-section '1 SQL Commands'. It contains the following SQL code:

```
1 select initcap(last_name) "Name", length(last_name) "Length of Name" from employee  
2 where last_name like 'M%' order by last_name;  
3
```

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

Name	Length of Name
Martin	6

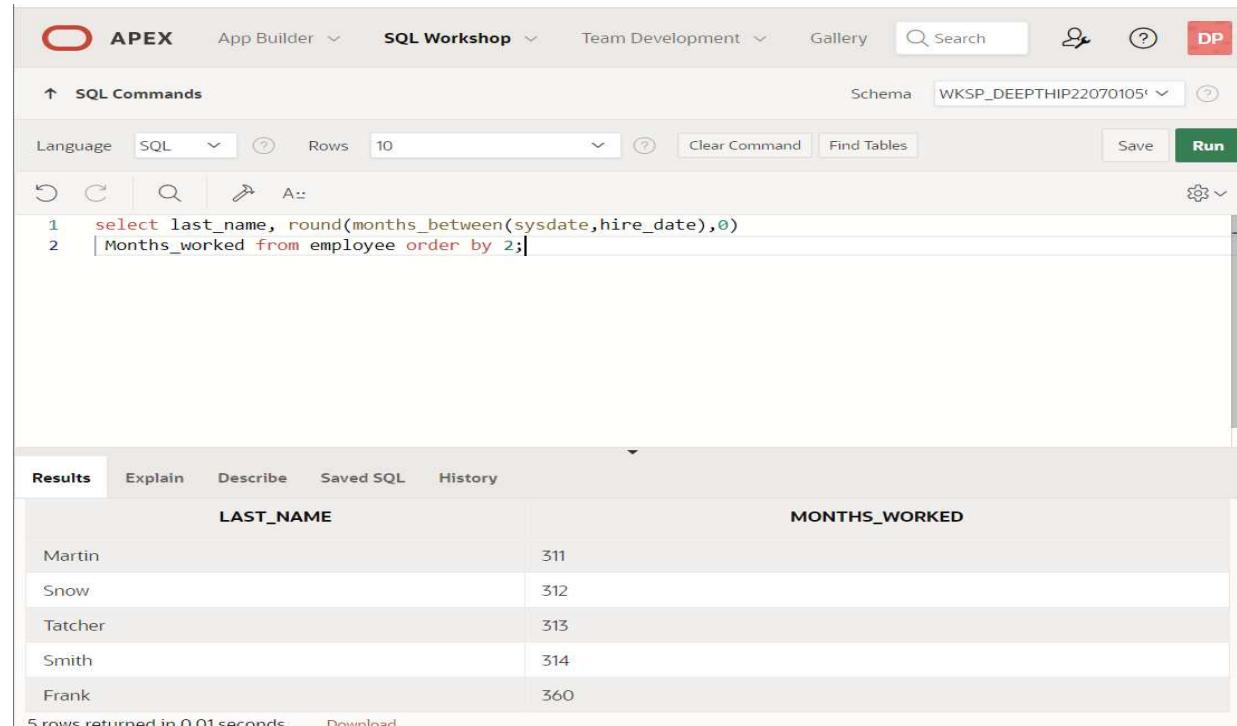
The results table has two columns: 'Name' and 'Length of Name'. The single row returned is 'Martin' with a length of 6. A note at the bottom says '1 rows returned in 0.01 seconds'.

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

**QUERY:**

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

**OUTPUT:**



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, Gallery, a search bar, and user profile icons. The SQL Commands tab is active, showing the following SQL code:

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

The Results tab is selected, displaying the output of the query:

LAST_NAME	MONTHS_WORKED
Martin	311
Snow	312
Tatcher	313
Smith	314
Frank	360

Below the table, it says "5 rows returned in 0.01 seconds" and there is a "Download" link.

7. Create a report that produces the following for each employee:

<employee last name> earns <salary> monthly but wants <3 times salary>. Label the column Dream Salaries.

**QUERY:**

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

**OUTPUT:**

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

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

The Results tab displays the output:

Dream Salary
Tatcher earns \$10000 monthly but wants \$30000
Snow earns \$30000 monthly but wants \$90000
Martin earns \$14000 monthly but wants \$42000
Smith earns \$20000 monthly but wants \$60000
Frank earns \$12000 monthly but wants \$36000

5 rows returned in 0.01 seconds

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

**QUERY:**

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

**OUTPUT:**

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

```
1 select last_name, lpad(salary,15,'$') Salary from employee;
```

The Results tab displays the output:

LAST_NAME	SALARY
Tatcher	\$\$\$\$\$\$\$\$\$\$10000
Snow	\$\$\$\$\$\$\$\$\$\$30000
Martin	\$\$\$\$\$\$\$\$\$\$14000
Smith	\$\$\$\$\$\$\$\$\$\$20000
Frank	\$\$\$\$\$\$\$\$\$\$12000

5 rows returned in 0.01 seconds

Page footer: Copyright © 1999, 2023, Oracle and/or its affiliates. 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(hire_date,'Monday')),'fmday," the  
"ddspth "of" month,yyyy') "REVIEW" from employee;
```

**OUTPUT:**

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, Gallery, and a search bar. The right side shows a user profile for 'deepthi p' with the schema 'WKSP\_DEEPTHIP22070105'. The main area is titled 'SQL Commands' with a language dropdown set to 'SQL'. The command entered is:

```
1 select last_name, hire_date, to_char(next_day(hire_date,'Monday')),'fmday," the "ddspth "of" month,yyyy')  
2 "REVIEW" from employee;  
3
```

Below the command, the results section displays the output:

LAST_NAME	HIRE_DATE	REVIEW
Tatcher	02/21/1998	monday, the twenty-third of february,1998
Snow	03/21/1998	monday, the twenty-third of march,1998
Martin	04/02/1998	monday, the sixth of april,1998
Smith	01/21/1998	monday, the twenty-sixth of january,1998
Frank	03/02/1994	monday, the seventh of march,1994

A note at the bottom indicates the results were returned in 0.01 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') "Day"  
from employe  
order by to_char(hire_date-1,'d')
```

**OUTPUT:**

APEX App Builder SQL Workshop Team Development Gallery Search

SQL Commands Schema WKSP\_DEEPTHIP220701055

Language SQL Rows 10 Clear Command Find Tables Save Run

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

Results Explain Describe Saved SQL History

LAST_NAME	HIRE_DATE	Day
Smith	01/21/1998	Wednesday
Frank	03/02/1994	Wednesday
Martin	04/02/1998	Thursday
Tatcher	02/21/1998	Saturday
Snow	03/21/1998	Saturday

5 rows returned in 0.01 seconds Download

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

## **RESULT:**

# DISPLAYING DATA FROM MULTIPLE TABLES

**EXP\_NO:**7

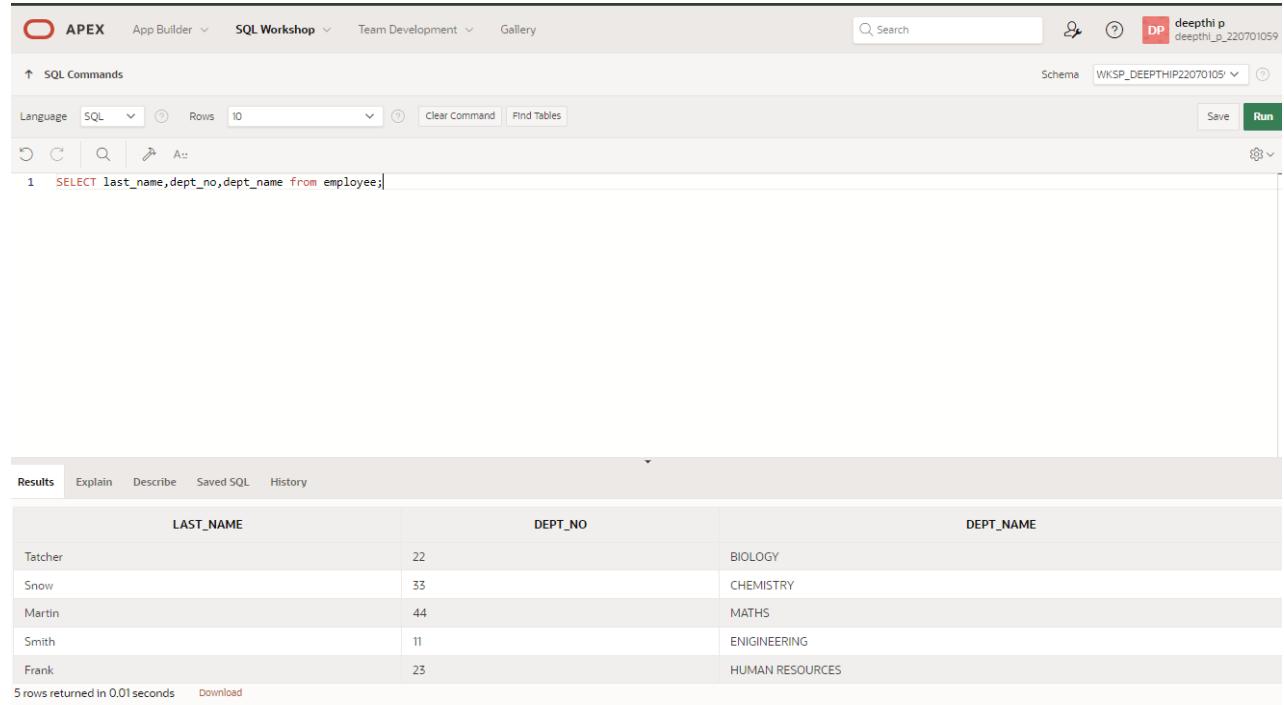
**DATE:**

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

**QUERY:**

Select last\_name,dept\_no,dept\_name from employee;

**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 'deepthi p' (deepthi\_p\_22070105). The main area has tabs for SQL Commands, Results (selected), Explain, Describe, Saved SQL, and History. The SQL Commands tab shows the query: 'SELECT last\_name,dept\_no,dept\_name from employee;'. The Results tab displays the following data:

LAST_NAME	DEPT_NO	DEPT_NAME
Tatcher	22	BIOLOGY
Snow	33	CHEMISTRY
Martin	44	MATHS
Smith	11	ENGINEERING
Frank	23	HUMAN RESOURCES

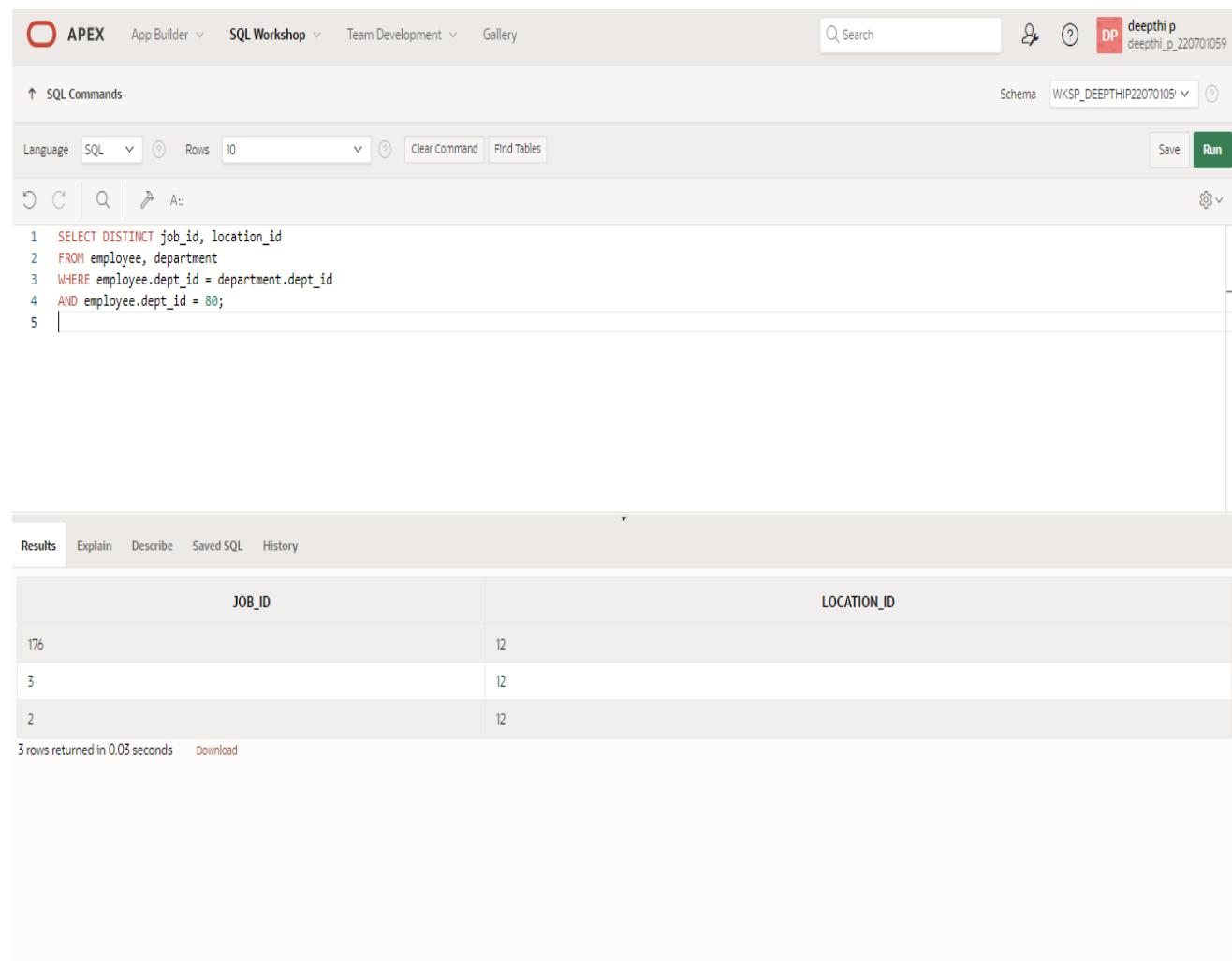
Below the table, it says '5 rows returned in 0.01 seconds' and there is a 'Download' link.

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, location_id  
FROM employee, department  
WHERE employee.dept_id = department.dept_id  
AND employee.dept_id = 80;
```

### 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 'deepthi p' and a schema dropdown set to 'WKSP\_DEEPTHIP22070105'. The main area has tabs for SQL Commands, Explain, Describe, Saved SQL, and History. The SQL Commands tab displays the query:

```
1 SELECT DISTINCT job_id, location_id  
2 FROM employee, department  
3 WHERE employee.dept_id = department.dept_id  
4 AND employee.dept_id = 80;  
5
```

The Results tab shows the output of the query:

JOB_ID	LOCATION_ID
176	12
3	12
2	12

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

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, d.department_name, d.location_id, l.city  
FROM employees e, departments d, locations l  
WHERE e.department_id = d.department_id  
AND  
d.location_id = l.location_id  
AND e.commission IS NOT NULL;
```

### 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 'deepthi\_p' and a schema dropdown for 'WKSP\_DEEPTHIP22070105'. The main area has tabs for SQL Commands, Explain, Describe, Saved SQL, and History. The SQL Commands tab displays the following code:

```
1 SELECT e.last_name, d.department_name, d.location_id, l.city  
2 FROM employees e, departments d, locations l  
3 WHERE e.department_id = d.department_id  
4 AND  
5 d.location_id = l.location_id  
6 AND e.commission IS NOT NULL;
```

Below the code, the Results tab is selected, showing a table with four columns: LAST\_NAME, DEPARTMENT\_NAME, LOCATION\_ID, and CITY. The data returned is:

LAST_NAME	DEPARTMENT_NAME	LOCATION_ID	CITY
Tatcher	BIOLOGY	12	Toronto
Snow	BIOLOGY	12	Toronto

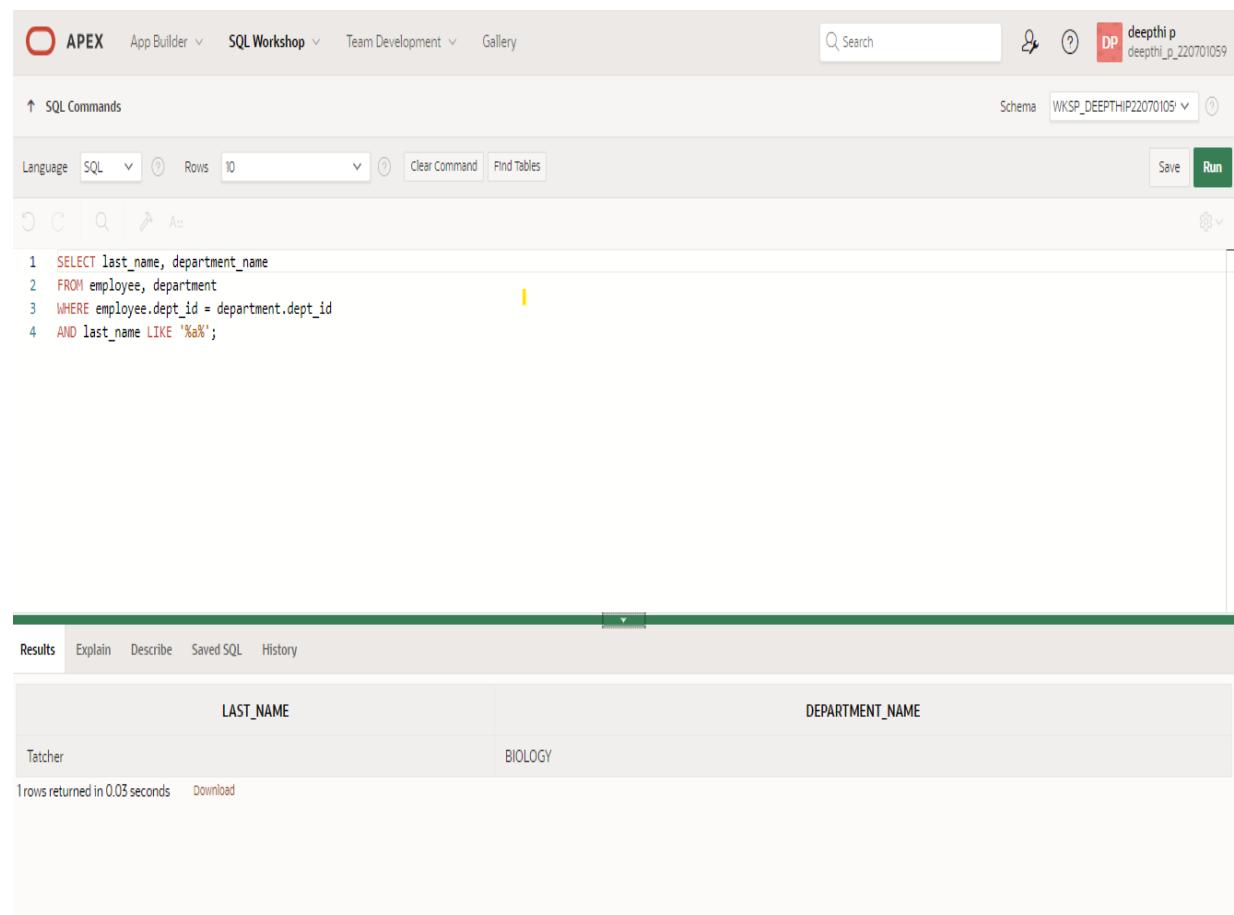
At the bottom left, it says '2 rows returned in 0.01 seconds' and there is a 'Download' link.

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

### QUERY:

```
SELECT last_name, department_name  
FROM employees, departments  
WHERE employees.department_id = departments.department_id  
AND last_name LIKE '%a%';
```

### OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. At the top, there are tabs for APEX, App Builder, SQL Workshop (which is selected), Team Development, and Gallery. On the right side, there are icons for search, help, and a user profile, along with the text "deepthi p deepthi\_p\_220701059". Below the tabs, there's a search bar and a schema dropdown set to "WKSP\_DEEPTHIP220701059". The main area contains a SQL command editor with the following code:

```
1 SELECT last_name, department_name  
2 FROM employee, department  
3 WHERE employee.dept_id = department.dept_id  
4 AND last_name LIKE '%a%';
```

Below the editor, there are buttons for Save and Run. The results tab is selected at the bottom, showing a single row of data:

LAST_NAME	DEPARTMENT_NAME
Tatcher	BIOLOGY

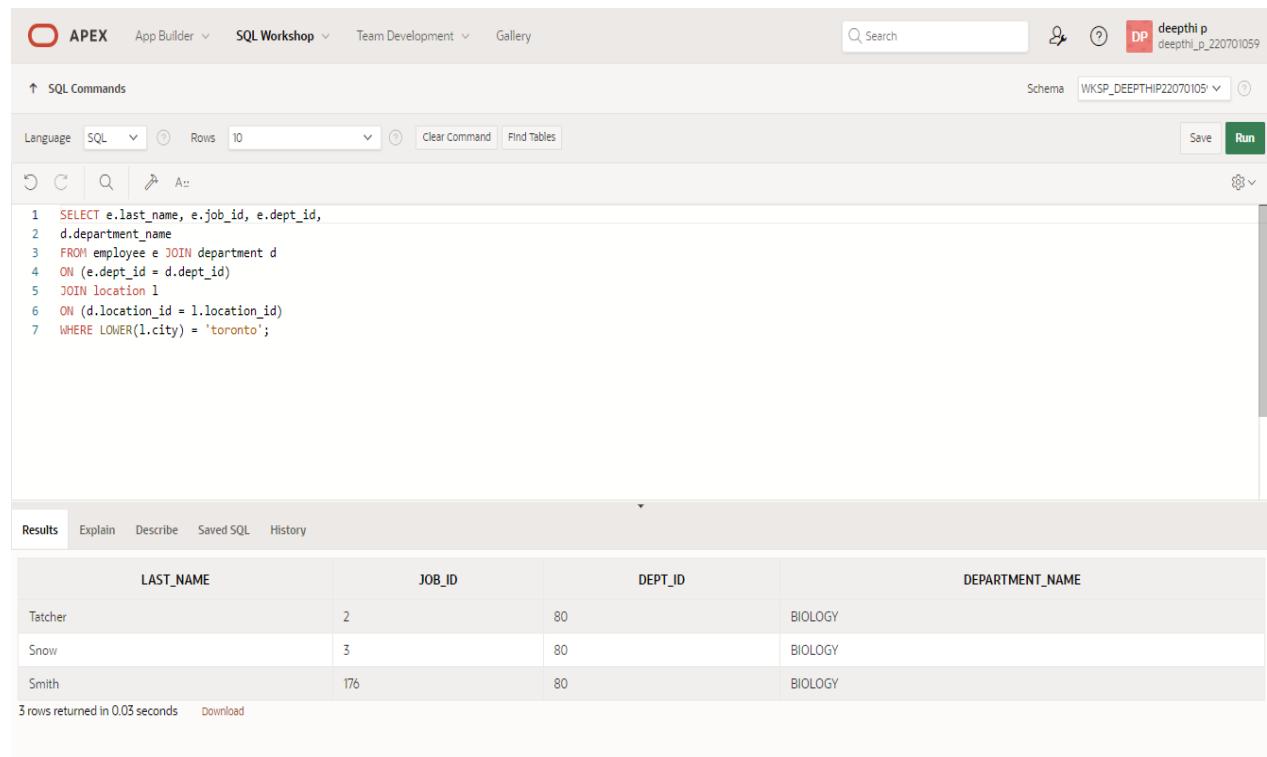
At the bottom left of the results panel, it says "1 rows returned in 0.03 seconds" and has a "Download" link.

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.job_id, e.dept_id,
d.department_name
FROM employee e JOIN department d
ON (e.dept_id = d.dept_id)
JOIN location l
ON (d.location_id = l.location_id)
WHERE LOWER(l.city) = 'toronto';
```

### 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 'deepthi\_p' and the schema 'WKSP\_DEEPTHIP22070105'. The main area has tabs for SQL Commands, Explain, Describe, Saved SQL, and History. The SQL Commands tab displays the query code. The Results tab shows the output of the query, which retrieves three rows from the database:

LAST_NAME	JOB_ID	DEPT_ID	DEPARTMENT_NAME
Tatcher	2	80	BIOLOGY
Snow	3	80	BIOLOGY
Smith	176	80	BIOLOGY

At the bottom, it says '3 rows returned in 0.03 seconds' and has a 'Download' link.

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.manager_name "manager", m.manager_id "Mgr#"  
FROM employee w join manager m  
ON (w.manager_id = m.manager_id);
```

### OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, Gallery, a search bar, and a user profile for 'deepthi p'.

In the SQL Commands section, the schema is set to 'WKSP\_DEEPTHIP22070105'. The code entered is:

```
1 SELECT w.last_name "employee", w.emp_id "EMP#",  
2      m.manager_name "manager", m.manager_id "Mgr#"  
3 FROM employee w join manager m  
4 ON (w.manager_id = m.manager_id);
```

The Results tab is selected, displaying the following output:

employee	EMP#	manager	Mgr#
Martin	3	MR.SINGH	123
Frank	5	MR.MOHANRAJ	456

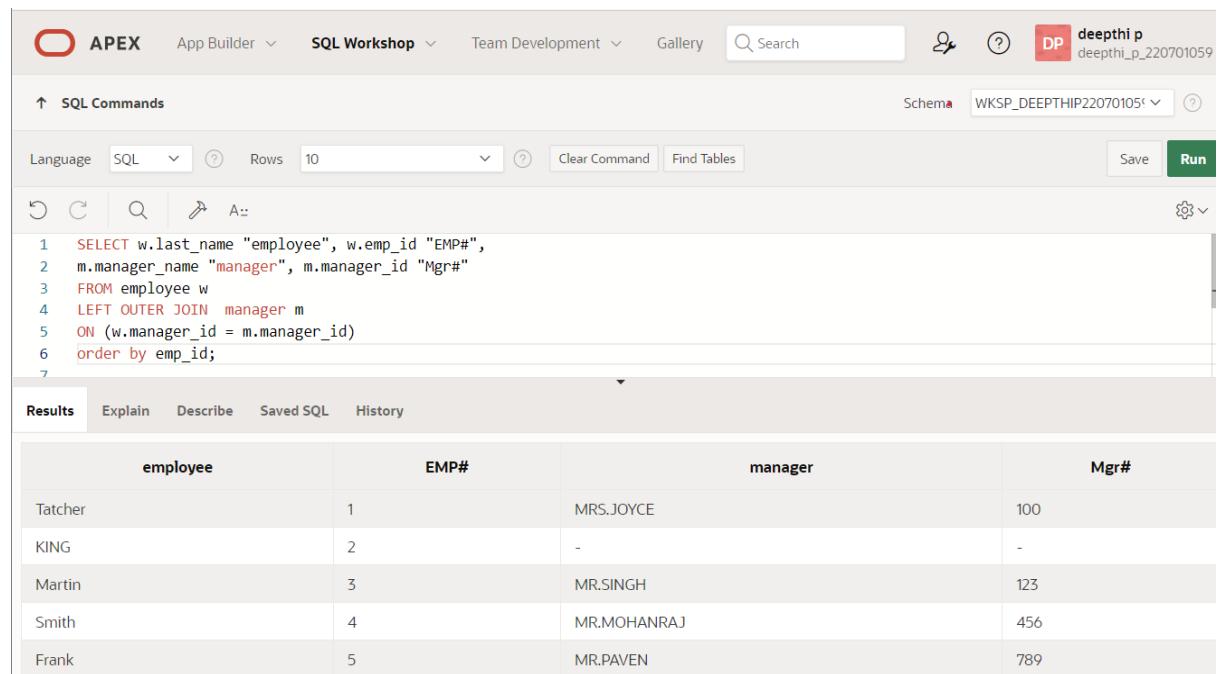
Below the table, it says '2 rows returned in 0.02 seconds' and there is a 'Download' link.

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.manager_name "manager", m.manager_id "Mgr#"  
FROM employee w  
LEFT OUTER JOIN manager m  
ON (w.manager_id = m.manager_id)  
order by emp_id;
```

### OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, Gallery, a search bar, and a user profile for 'deepthi p'.

The SQL Commands tab is active, displaying the following SQL code:

```
1 SELECT w.last_name "employee", w.emp_id "EMP#",  
2 m.manager_name "manager", m.manager_id "Mgr#"  
3 FROM employee w  
4 LEFT OUTER JOIN manager m  
5 ON (w.manager_id = m.manager_id)  
6 order by emp_id;  
7
```

The Results tab is selected, showing the output of the query:

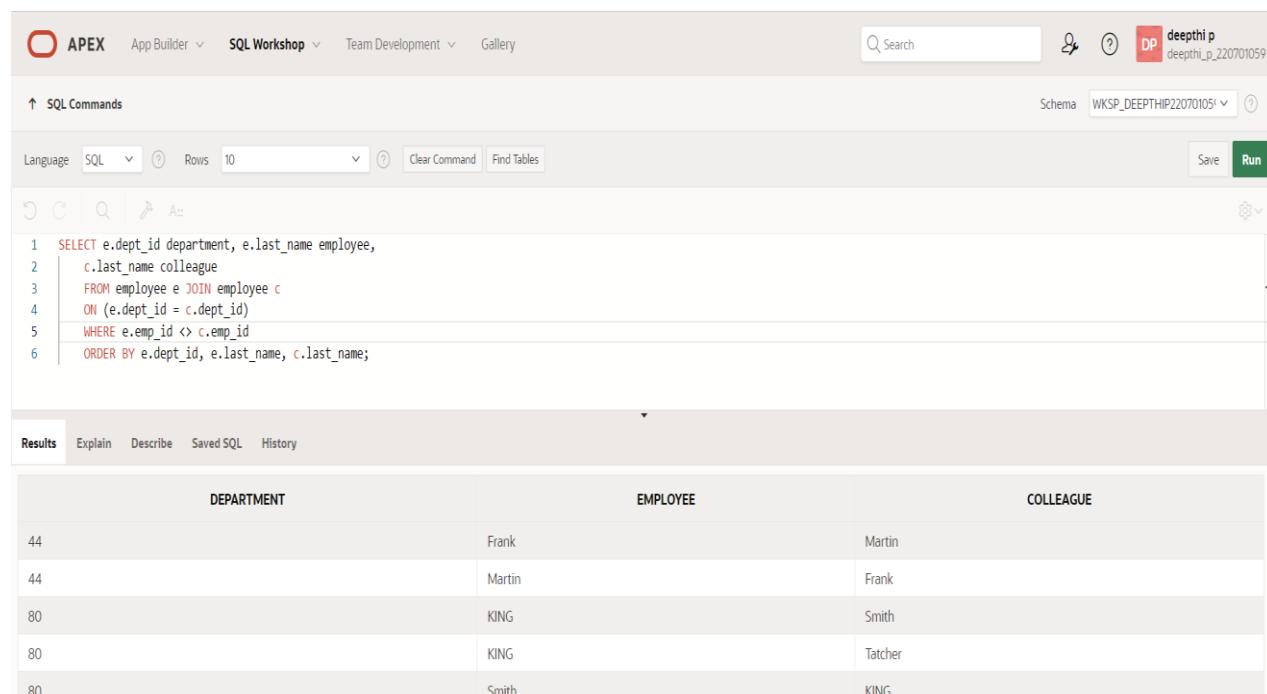
employee	EMP#	manager	Mgr#
Tatcher	1	MRS.JOYCE	100
KING	2	-	-
Martin	3	MR.SINGH	123
Smith	4	MR.MOHANRAJ	456
Frank	5	MR.PAVEN	789

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.dept_id department, e.last_name employee,  
c.last_name colleague  
FROM employee e JOIN employee c  
ON (e.dept_id = c.dept_id)  
WHERE e.emp_id <> c.emp_id  
ORDER BY e.dept_id, e.last_name, c.last_name;
```

### 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 'deepthi\_p' and the schema 'WKSP\_DEEPTHIP22070105'. The main area has tabs for SQL Commands, Explain, Describe, Saved SQL, and History. The SQL Commands tab contains the query code. The Results tab is selected, displaying the output in a grid format.

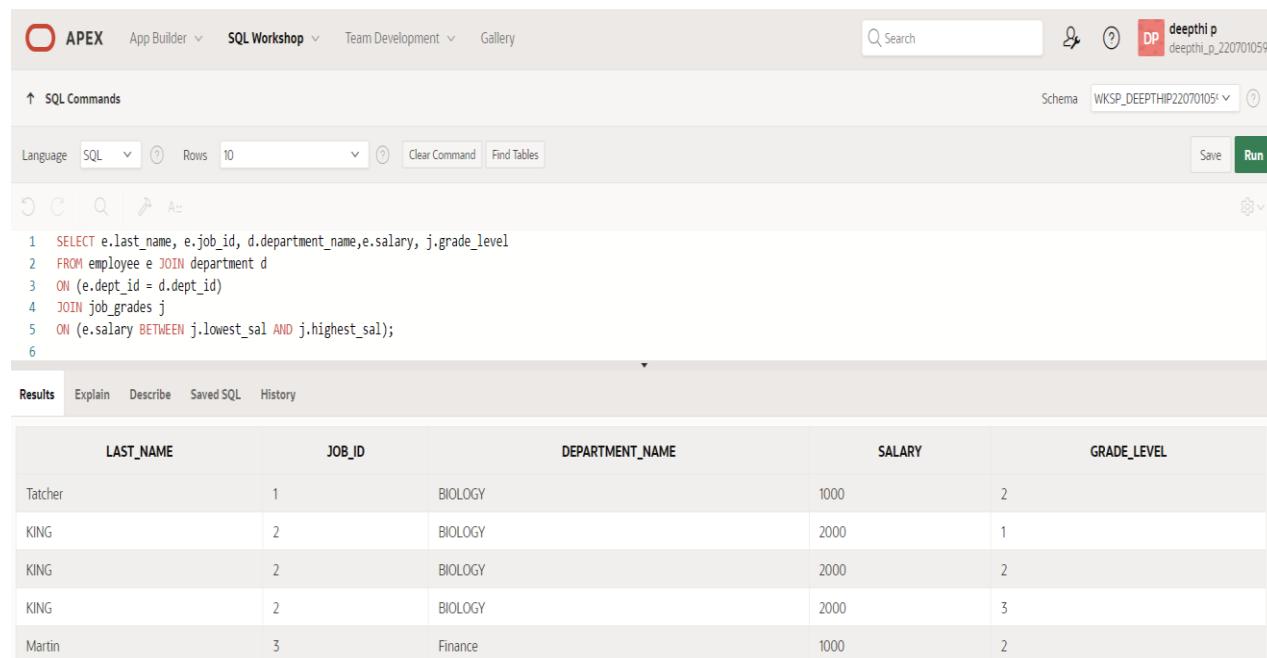
DEPARTMENT	EMPLOYEE	COLLEAGUE
44	Frank	Martin
44	Martin	Frank
80	KING	Smith
80	KING	Tatcher
80	Smith	KING

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.department_name, e.salary, j.grade_level  
FROM employee e JOIN department d  
ON (e.dept_id = d.dept_id)  
JOIN job_grades j  
ON (e.salary BETWEEN j.lowest_sal AND j.highest_sal);
```

### OUTPUT:



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

```
1 SELECT e.last_name, e.job_id, d.department_name, e.salary, j.grade_level  
2 FROM employee e JOIN department d  
3 ON (e.dept_id = d.dept_id)  
4 JOIN job_grades j  
5 ON (e.salary BETWEEN j.lowest_sal AND j.highest_sal);  
6
```

The Results tab displays the output of the query:

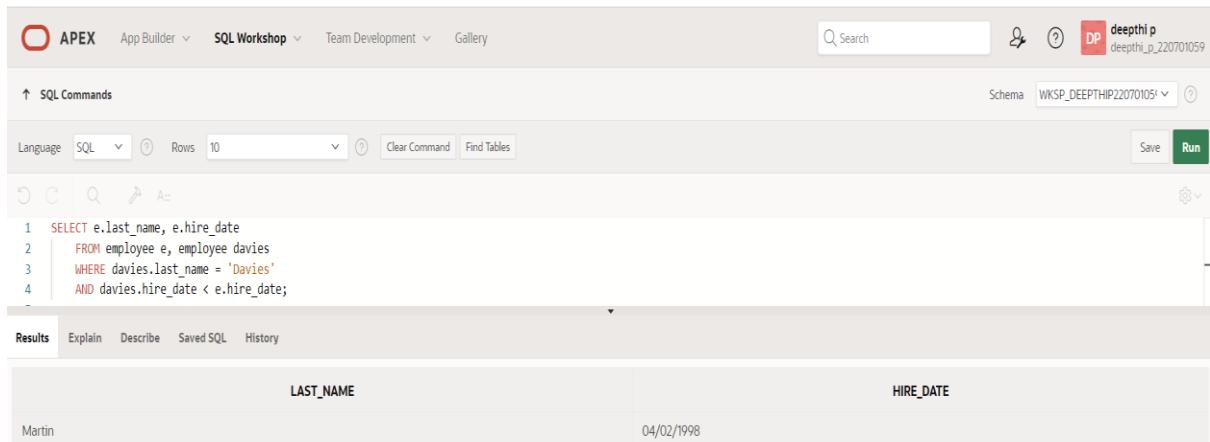
LAST_NAME	JOB_ID	DEPARTMENT_NAME	SALARY	GRADE_LEVEL
Tatcher	1	BIOLOGY	1000	2
KING	2	BIOLOGY	2000	1
KING	2	BIOLOGY	2000	2
KING	2	BIOLOGY	2000	3
Martin	3	Finance	1000	2

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 employee e, employee davies  
WHERE davies.last_name = 'Davies'  
AND davies.hire_date < e.hire_date;
```

## OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. On the right, there's a search bar, a user icon for 'deepthi p', and a schema dropdown set to 'WKSP\_DEEPTHIP220701051'. The main area is titled 'SQL Commands' with a 'Language' dropdown set to 'SQL'. A code editor contains the following query:

```
1 SELECT e.last_name, e.hire_date
2   FROM employee e, employee davies
3  WHERE davies.last_name = 'Davies'
4  AND davies.hire_date < e.hire_date;
```

Below the code editor, a results grid displays the output:

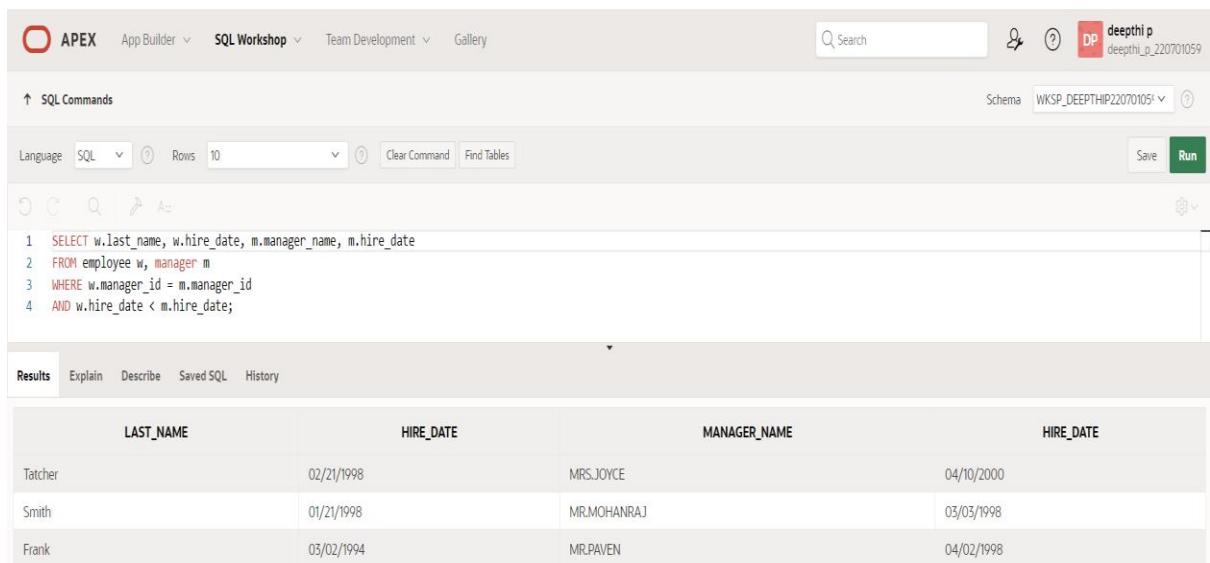
LAST_NAME	HIRE_DATE
Martin	04/02/1998

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 w.last_name, w.hire_date, m.manager_name, m.hire_date
FROM employee w, manager m
WHERE w.manager_id = m.manager_id
AND w.hire_date < m.hire_date;
```

## OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. On the right, there's a search bar, a user icon for 'deepthi p', and a schema dropdown set to 'WKSP\_DEEPTHIP220701051'. The main area is titled 'SQL Commands' with a 'Language' dropdown set to 'SQL'. A code editor contains the following query:

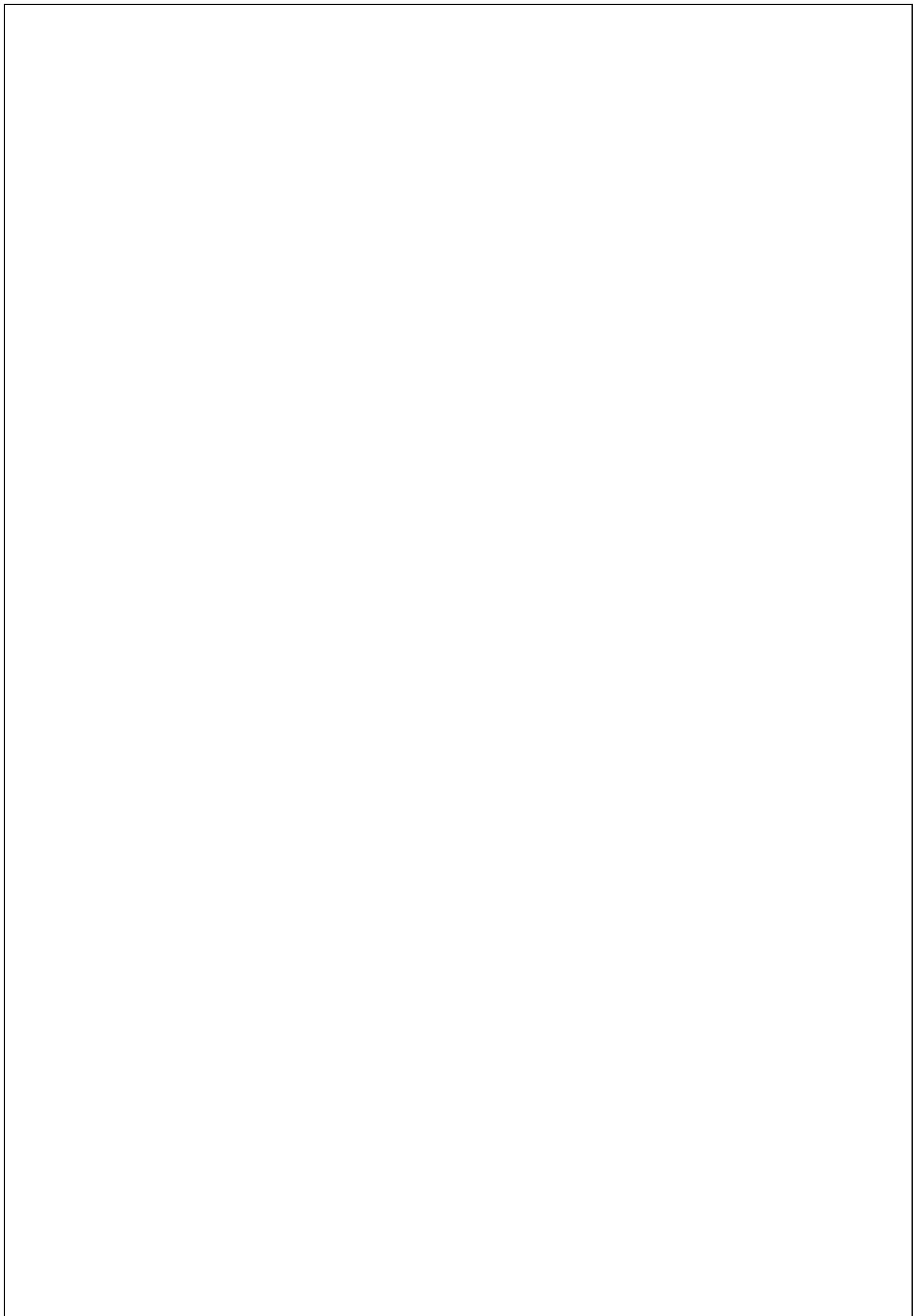
```
1 SELECT w.last_name, w.hire_date, m.manager_name, m.hire_date
2   FROM employee w, manager m
3  WHERE w.manager_id = m.manager_id
4  AND w.hire_date < m.hire_date;
```

Below the code editor, a results grid displays the output:

LAST_NAME	HIRE_DATE	MANAGER_NAME	HIRE_DATE
Tatcher	02/21/1998	MRS.JOYCE	04/10/2000
Smith	01/21/1998	MR.MOHANRAJ	03/03/1998
Frank	03/02/1994	MR.PAVEN	04/02/1998

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 EMPLOYEES;
```

## OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. On the right, there are user profile icons for 'deepthi p' and 'deepthi\_p\_22070105'. The main area is titled 'SQL Commands' with a search bar and a 'Run' button. The code entered is:

```
1 select MAX(salary) "Maximum",MIN(salary) "Minimum",SUM(salary) "Sum",AVG(salary) "Average" from employees;
```

The results section displays the following summary statistics:

	Maximum	Minimum	Sum	Average
11000	4000	21900	7300	

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

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 EMPLOYEES group by job_id;
```

## 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 profile are the same. The main area is titled 'SQL Commands' with a search bar and a 'Run' button. The code entered is:

```
1 select MAX(salary) "Maximum",MIN(salary) "Minimum",SUM(salary) "Sum",AVG(salary) "Average" from employees group by job_id;
```

The results section displays the following summary statistics:

	Maximum	Minimum	Sum	Average
11000	4000	21900	7300	

Below the table, it says '1 rows returned in 0.00 seconds' and provides a 'Download' link. The bottom footer includes copyright information for Oracle and the APEX version.

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

**OUTPUT:**

The screenshot shows the Oracle APEX interface with the SQL Workshop module selected. In the SQL Commands panel, the following query is entered:

```
1 select count(*) "No_of_people" from employees group by job_id;
```

The results panel displays the output:

No_of_people
3

Below the results, it says "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 employees;
```

**OUTPUT:**

A screenshot of the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. The SQL Workshop tab is selected. The main area shows a SQL command line with the following query:

```
1 select count(manager_id) "Number of Managers" from employees where manager_id is not null;
```

The results section displays the output:

Number of Managers	
3	

Below the results, it says "1 rows returned in 0.01 seconds" and provides a "Download" link. The bottom of the screen shows user information (deepthi\_p\_220701059) and copyright information (Copyright © 1999, 2025, Oracle and/or its affiliates.).

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

**QUERY:**

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

**OUTPUT:**

A screenshot of the Oracle APEX SQL Workshop interface, identical to the previous one but with a different query. The SQL command line shows:

```
1 select MAX(salary)-MIN(salary) "DIFFERENCE" from employees;
```

The results section displays the output:

DIFFERENCE	
7000	

Below the results, it says "1 rows returned in 0.01 seconds" and provides a "Download" link. The bottom of the screen shows user information (deepthi\_p\_220701059) and copyright information (Copyright © 1999, 2025, Oracle and/or its affiliates.).

**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 employees 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 interface with the SQL Workshop module selected. The query entered is:

```
1 SELECT manager_id,MIN(salary) AS lowest_salary FROM employees WHERE manager_id IS NOT NULL GROUP BY manager_id HAVING MIN(salary) > 6000 ORDER BY lowest_salary DESC;
```

The results page displays the following data:

MANAGER_ID	LOWEST_SALARY
100	11000
205	6900

2 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(*) as 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 employees;
```

**OUTPUT:**

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

```
1 select count(*) as 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 employees;
```

The results page displays the following data:

TOTAL	1995	1996	1997	1998
3	0	0	0	1

1 rows returned in 0.04 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:**

**OUTPUT:**

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

```
1 select department_id,job_id,sum(salary) from employees where department_id in (20,50,80,9) group by rollup(department_id,job_id);
```

The results window displays the output of the query:

DEPARTMENT_ID	JOB_ID	SUM(SALARY)
80	ac_account	11000
80	-	11000
-	-	11000

3 rows returned in 0.03 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 departments d inner join employees e on(d.dpt_id =e.department_id ) group by d.dept_name ,d.loc;
```

**OUTPUT:**

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

```
1 select d.dept_name as "Department name",l.location_id as "Location",count(e.department_id) as "Number of people",round(avg(e.salary),2) as "Salary"
2 from departments d,employees e,location l where d.dept_id=e.department_id group by d.dept_name,l.location_id,e.department_id;
```

The results window displays the output of the query:

Department name	Location	Number of people	Salary
Public Relations	4598	1	6900
Public Relations	1231	1	6900
finance	4598	1	11000
finance	1231	1	11000

4 rows returned in 0.02 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 SQL Workshop interface. In the SQL Commands tab, a subquery is executed to find employees in the same department as 'Janu' but with a different last name. The results are displayed in a table with columns LAST\_NAME and HIRE\_DATE, showing one row for 'DURRANT'.

LAST_NAME	HIRE_DATE
DURRANT	1985-01-15

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 schema is set to 'WKSP\_DEEPTHIP22070105'. The SQL command entered is:

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

The results table shows two rows:

EMPLOYEE_ID	LAST_NAME	SALARY
142	Doe	30000
1001	Smith	70000

2 rows returned in 0.01 seconds. There is a 'Download' button.

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 schema is set to 'WKSP\_DEEPTHIP22070105'. The SQL command entered is:

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

The results table shows two rows:

EMPLOYEE_ID	LAST_NAME
113	Janu
142	Doe

2 rows returned in 0.03 seconds. There is a 'Download' button.

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 links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows a user profile for 'deepthi p' and the schema 'WKSP\_DEEPTHIP22070105'. The main area displays a SQL command:

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

The results table has columns LAST\_NAME, DEPARTMENT\_ID, and JOB\_ID. It contains two rows:

LAST_NAME	DEPARTMENT_ID	JOB_ID
Janu	100	ac_account
Doe	100	ac_account

Below the table, it says '2 rows returned in 0.04 seconds' and there is a 'Download' link.

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 top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows a user profile for 'deepthi\_p\_22070105' and the schema 'WKSP\_DEEPTHIP22070105'. The main area displays a SQL command:

```
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. It contains two rows:

LAST_NAME	SALARY
Davies	11000
Mohan	4000

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

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 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 'deepthi p', and a schema dropdown 'WKSP\_DEEPTHIP22070105'. The main area is titled 'SQL Commands' with a 'Run' button. The code entered is:

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

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

DEPARTMENT_ID	LAST_NAME	JOB_ID
80	Smith	sales_rep
80	Davies	ac_account
20	Doe	ac_account

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 search bar, user profile 'deepthi p', and a schema dropdown 'WKSP\_DEEPTHIP22070105'. The main area is titled 'SQL Commands' with a 'Run' button. The code entered is:

```
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, showing the output:

EMPLOYEE_ID	LAST_NAME	SALARY
1001	Smith	70000
142	Doeu	30000

2 rows returned in 0.01 seconds

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

**RESULT:**

# USING THE SET OPERATORS

**EX\_NO:10**

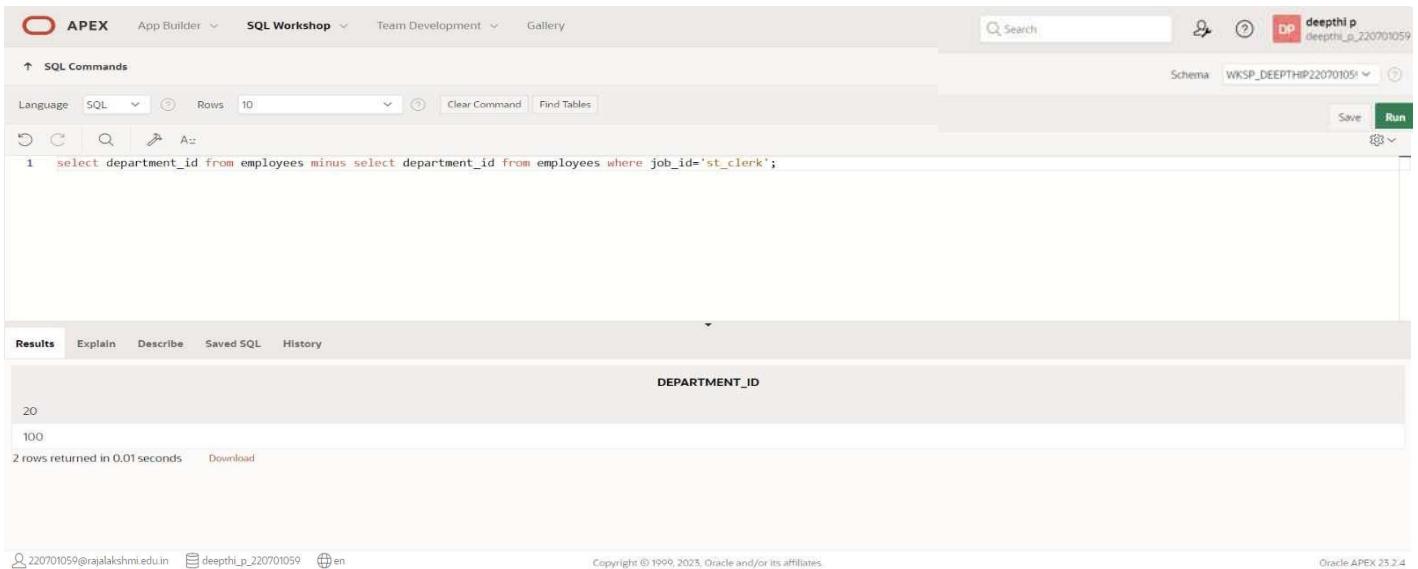
**DATE:**

- 1.) The HR department needs a list of department IDs for departments that do not contain the job DST\_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 top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. The right side shows a user profile for 'deepthi\_p' and the schema 'WKSP\_DEEPTHI\_P220701051'. The main area is titled 'SQL Commands' with a 'Run' button. The SQL editor contains the following query:

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

The results section shows the output of the query:

DEPARTMENT_ID
20
100

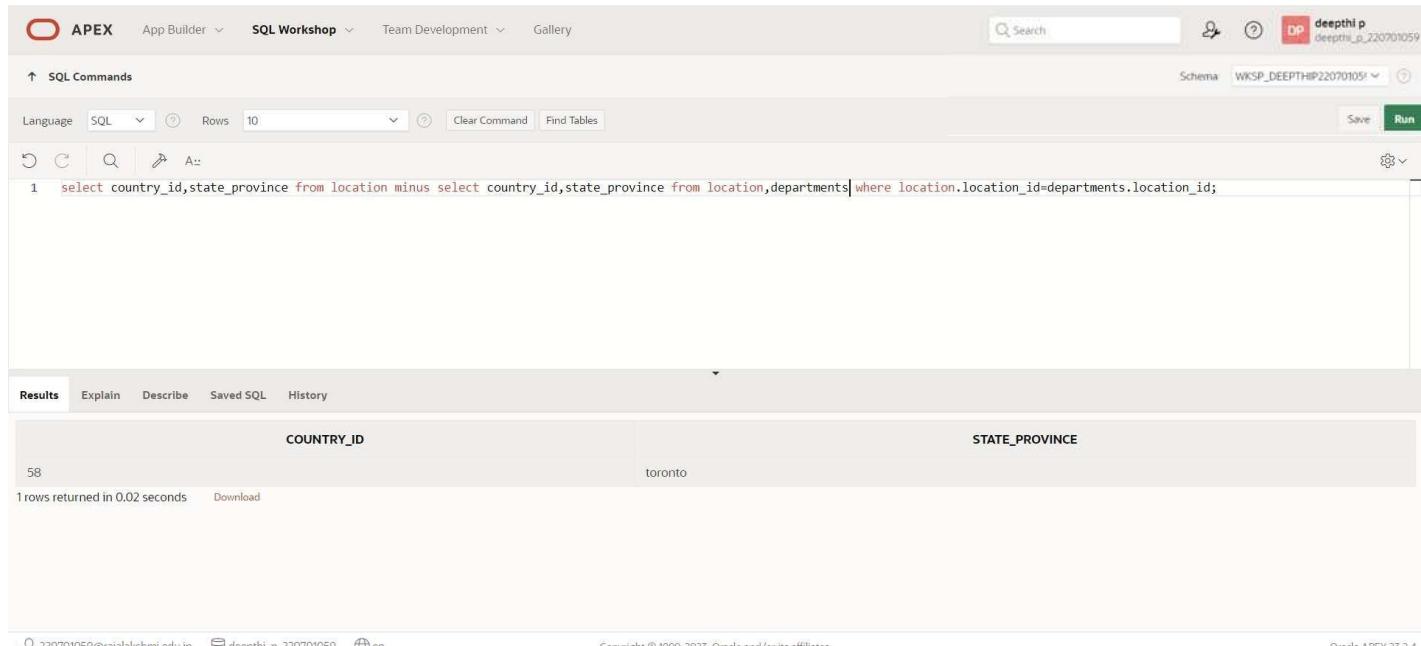
Below the table, it says '2 rows returned in 0.01 seconds' and has a 'Download' link. At the bottom, there are footer links for '220701059@rajalakshmi.edu.in', 'deepthi\_p\_220701059', 'en', 'Copyright © 1999, 2023, Oracle and/or its affiliates.', and 'Oracle APEX 23.2.4'.

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 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 'deepthi p', and a schema dropdown set to 'WKSP\_DEEPTHIP22070105'. The main workspace displays a SQL command: 'select country\_id,state\_province from location minus select country\_id,state\_province from location,departments where location.location\_id=departments.location\_id;'. Below the command, the results section shows a single row: COUNTRY\_ID 58 and STATE\_PROVINCE toronto. A note indicates '1 rows returned in 0.02 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. In the top navigation bar, the 'SQL Workshop' tab is selected. The main area contains three SQL statements separated by UNION operators:

```

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 section displays the output of these queries:

JOB_ID	DEPARTMENT_ID
ac_account	20
hr_rep	20

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

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

#### QUERY:

```
select 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 with the same query as above:

```
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 section displays the output of the intersect query:

JOB_ID	EMPLOYEE_ID
ac_account	113
ac_account	142
sales_rep	1001

Below the table, it says "3 rows returned in 0.03 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 'deepthi p', and a schema dropdown set to 'WKSP\_DEEPTHP220701051'. The main workspace displays a SQL command: 'select first\_name||' '||last\_name as "Name",department\_id from employees union all select dept\_name,dept\_id from departments;'. Below the command is a results grid with two columns: 'Name' and 'DEPARTMENT\_ID'. The results list ten entries, each consisting of a name and its corresponding department ID. At the bottom left, it says '10 rows returned in 0.01 seconds'. At the bottom right, it says 'Oracle APEX 23.2.4'.

Name	DEPARTMENT_ID
John Smith	80
Emily Johnson	20
Jaunty Janu	100
den Davies	80
Jane Doe	20
Vijaya Mohan	150
Public Relations	100
finance	80
Executive	80
Executive	20

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

**RESULT:**

## CREATING VIEWS

EX.NO.11

DATE:

Find the Solution for the following:

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 SQL Workshop interface. In the SQL Commands pane, the following SQL code is entered:

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

The 'Results' tab is selected, and the output shows:

```
View created.
```

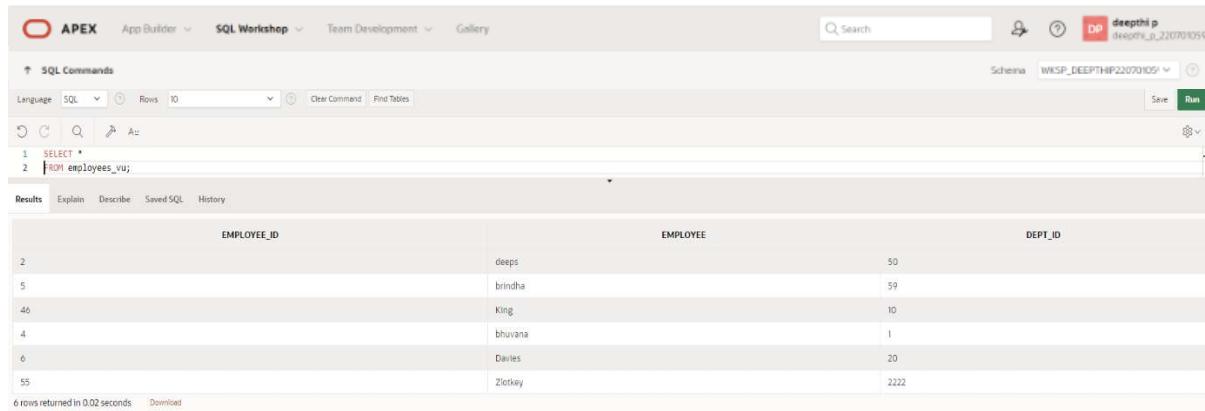
Execution time: 0.08 seconds

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

**QUERY:**

```
select * from employees_vu;
```

**OUTPUT:**



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

```
1 SELECT *
2 FROM employees_vu;
```

The 'Results' tab is selected, and the output shows a table with the following data:

EMPLOYEE_ID	EMPLOYEE	DEPT_ID
2	deeps	50
5	brinda	59
46	King	10
4	bhuvana	1
6	Davies	20
55	Zlotkey	2222

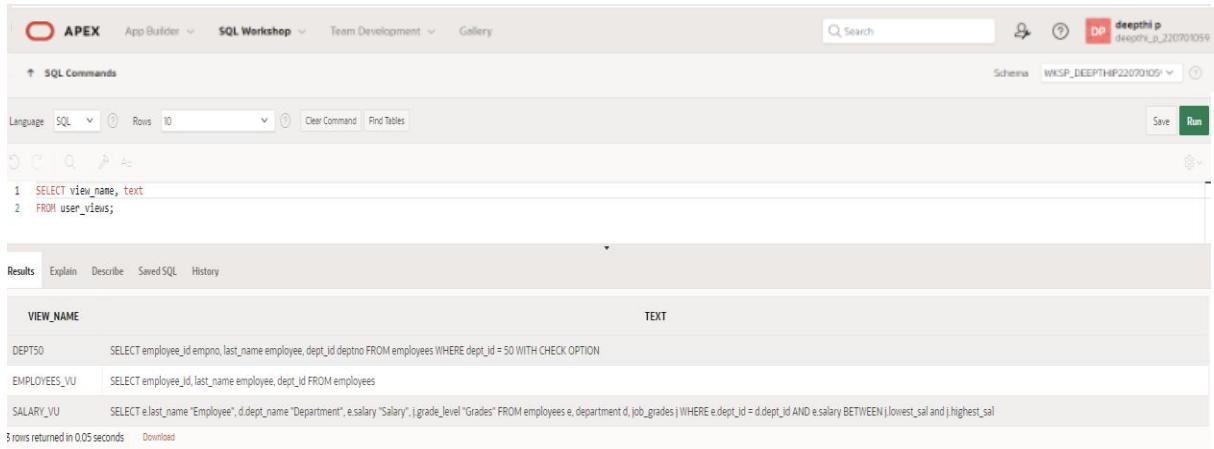
6 rows returned in 0.02 seconds    Download

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 SQL Workshop interface. In the SQL Commands pane, the following query is entered:

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

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

VIEW_NAME	TEXT
DEPT50	SELECT employee_id empno, last_name employee, dept_id deptno FROM employees WHERE dept_id = 50 WITH CHECK OPTION
EMPLOYEES_VU	SELECT employee_id, last_name employee, dept_id FROM employees
SALARY_VU	SELECT e.last_name "Employee", d.dept_name "Department", e.salary "Salary", j.grade_level "Grades" FROM employees e, department d, job_grades j WHERE e.dept_id = d.dept_id AND e.salary BETWEEN j.lowest_sal AND j.highest_sal

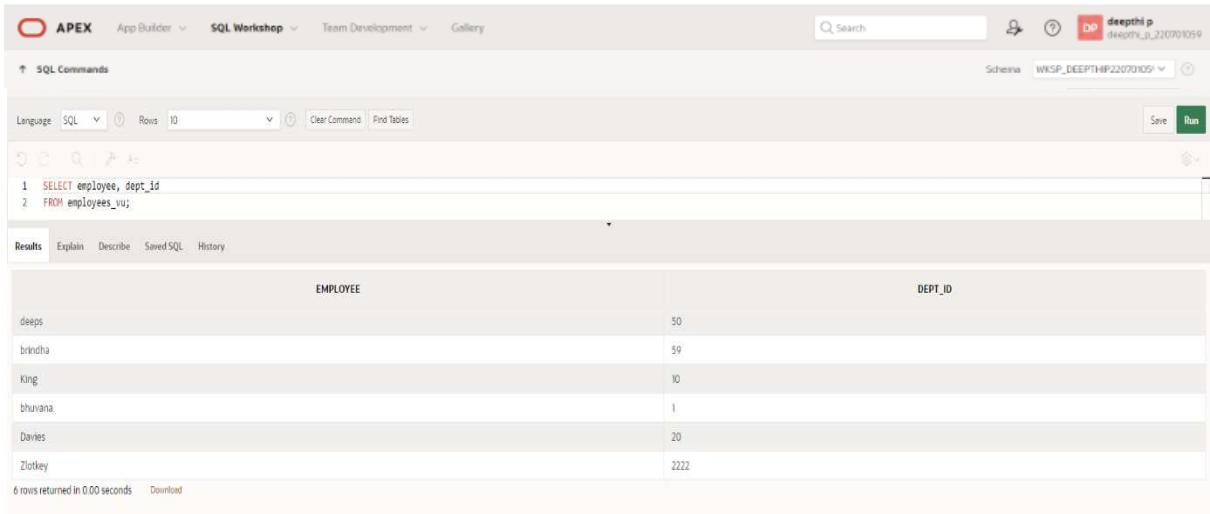
3 rows returned in 0.05 seconds [Download](#)

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 SQL Workshop interface. In the SQL Commands pane, the following query is entered:

```
1 SELECT employee, dept_id  
2 FROM employees_vu;
```

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

EMPLOYEE	DEPT_ID
deep	50
brinda	59
King	10
bharana	1
Davies	20
Zlotkey	2222

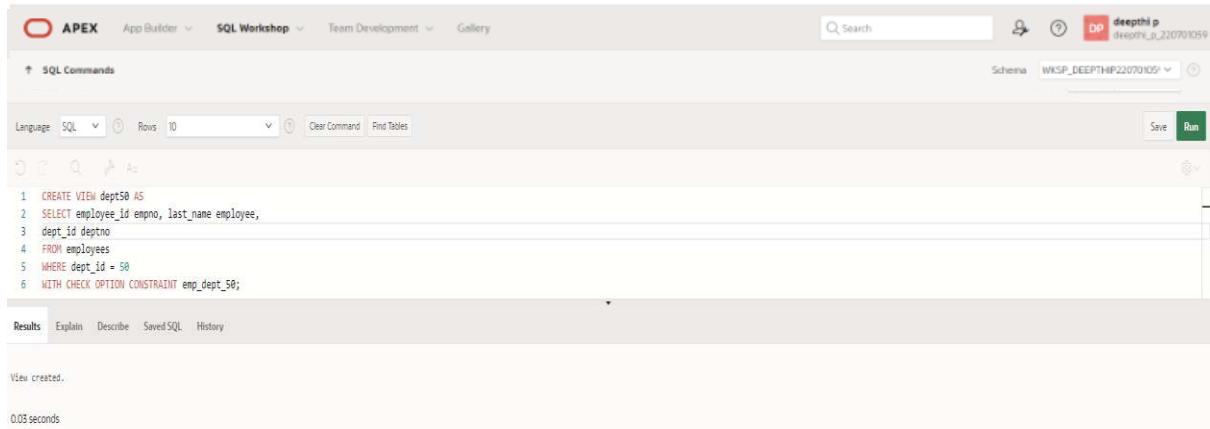
6 rows returned in 0.00 seconds [Download](#)

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 SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows a workspace titled 'deepthi p' with schema 'WKSP\_DEEPTHIP220701059'. The main area is a SQL editor with the following content:

```
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;
```

The 'Results' tab is selected at the bottom, displaying the message 'View created.' and a execution time of '0.03 seconds'.

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

### QUERY:

**Describe dept50;**

**SELECT \* from dept50;**

### 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 Commands tab is selected. The schema dropdown is set to WKSP\_DEEPTHIP220701059. The command entered is 'DESCRIBE dept50'. The results pane shows the structure of the view, including columns EMPNO, EMPLOYEE, and DEPTNO, along with their data types and constraints.

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
DEPT50	EMPNO	NUMBER	22	-	-	-	✓	-	-
	EMPLOYEE	VARCHAR2	25	-	-	-	-	-	-
	DEPTNO	NUMBER	-	6	0	-	✓	-	-

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, and Gallery. The SQL Commands tab is selected. The schema dropdown is set to WKSP\_DEEPTHIP220701059. The commands entered are 'SELECT \* FROM dept50;'. The results pane displays the data from the DEPT50 view, showing one row with EMPNO 2, EMPLOYEE 'deeps', and DEPTNO 50.

EMPNO	EMPLOYEE	DEPTNO
2	deeps	50

7. Attempt to reassign Matos to department 80.

### QUERY:

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

### OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, and Gallery. The SQL Commands tab is selected. The schema dropdown is set to WKSP\_DEEPTHIP220701059. The commands entered are 'UPDATE dept50 SET deptno=80 WHERE employee='Matos''. The results pane shows an error message: 'ORA-01402: view WITH CHECK OPTION where-clause violation'.

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
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. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. The right side shows a user profile for 'deepthi p' and a schema dropdown for 'WKSP\_DEEPTHIP22070105'. The main area is titled 'SQL Commands' with a 'Language' dropdown set to 'SQL'. Below the code editor, there are buttons for 'Save' and 'Run'. The code editor contains the SQL command provided above, with line numbers 1 through 11. The results tab is selected, showing the message 'View created.' and a execution time of '0.06 seconds'. The bottom footer includes copyright information for Oracle and the APEX version 'Oracle APEX 23.2.4'.

```
1 CREATE OR REPLACE VIEW salary_vu
2 AS
3 SELECT e.last_name "Employee",
4 d.dept_name "Department",
5 e.salary "Salary",
6 j.grade_level "Grades"
7 FROM employees e,
8 department d,
9 job_grades j
10 WHERE e.dept_id = d.dept_id
11 AND e.salary BETWEEN j.lowest_sal and j.highest_sal;
```

Results Explain Describe Saved SQL History

View created.

0.06 seconds

220701046@rajalakshmi.edu.in mbhuven Copyright © 1999, 2023, Oracle and/or its affiliates. Oracle APEX 23.2.4

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

**RESULT:**

# Intro to Constraints; NOT NULL and UNIQUE Constraints

**EX\_NO:**12

**DATE:**

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;
```

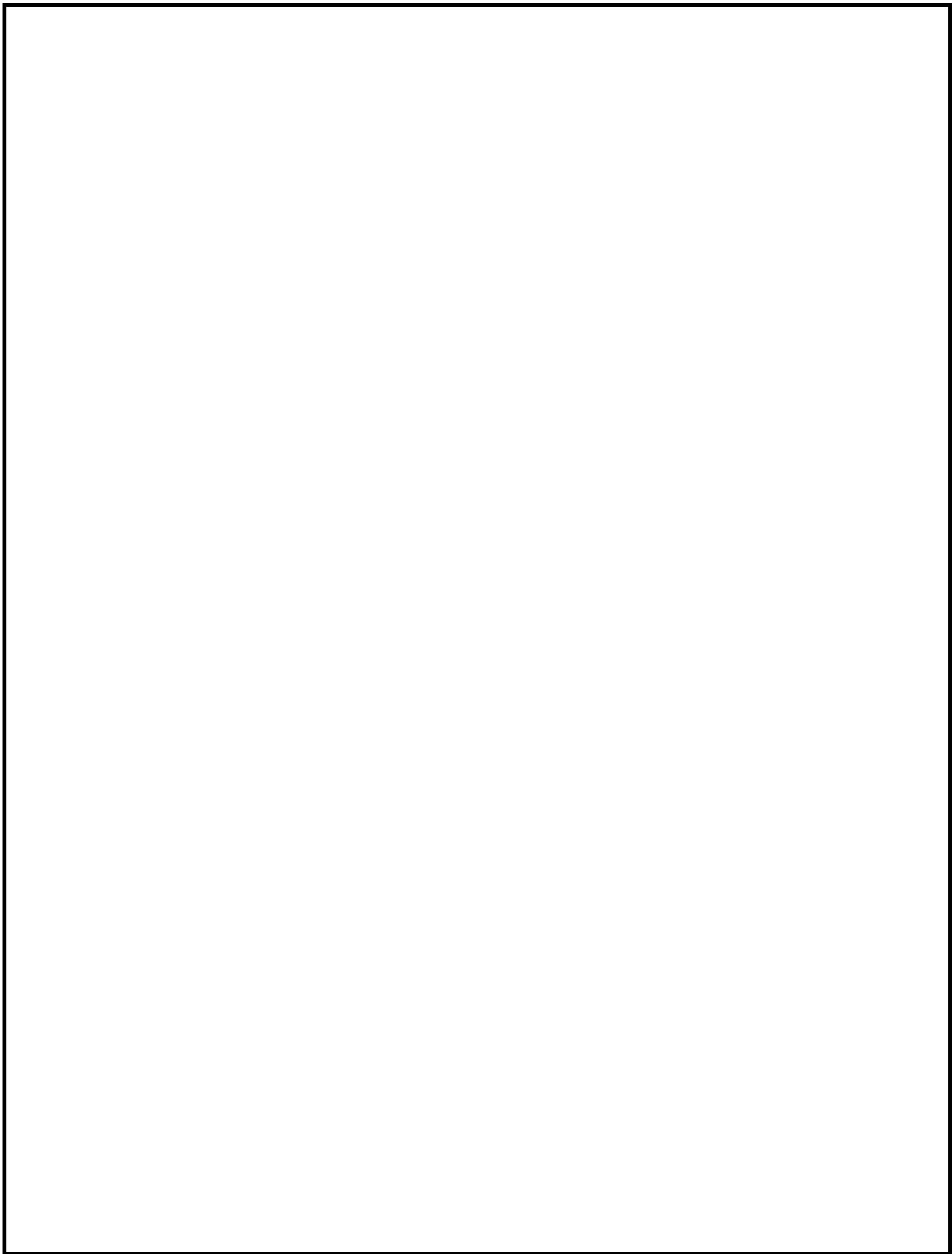
9. Rewrite the CREATE TABLE statement for the Global Fast Foods locations table to define the UNIQUE constraints at the table level. Do not execute this statement.

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)
);

```



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

## **RESULT:**

## Creating Views

EX.NO.13

DATE:

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

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

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

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



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

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

The 'Results' tab is active, showing the message "View created." and a execution time of "0.03 seconds".

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



The screenshot shows the Oracle SQL Workshop interface with the same navigation bar and SQL Workshop tab. The SQL command is now:

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

The 'Results' tab is active, displaying the output of the query:

ID	Song Title	Artist
54	YYY	SHREYA GOSHAL

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

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

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



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

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

The 'Results' tab is selected, showing the message 'View created.' and a execution time of '0.01 seconds'.

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

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



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

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

The 'Results' tab is selected, showing the message 'View created.' and a execution time of '0.02 seconds'.

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

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



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, and Gallery. The SQL Workshop tab is selected. The main area is titled 'SQL Commands' with a 'Language' dropdown set to 'SQL'. Below the dropdown are buttons for 'Rows' (set to 10), 'Clear Command', and 'Find Tables'. On the right side, there are 'Save' and 'Run' buttons. The code area contains the SQL command for creating the view. The results section below shows the message 'View created.' and a execution time of '0.02 seconds'.

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

Results Explain Describe Saved SQL History

View created.

0.02 seconds



The screenshot shows the Oracle SQL Workshop interface, identical to the previous one but with a different query in the command area. The code is a simple select statement from the view: 'SELECT \* FROM view\_min\_max\_avg\_dpt\_salary;'. The results section displays the data from the view, which includes three rows of department information: BIO, cse, and Executive, along with their respective salary statistics.

```
1 SELECT * FROM view_min_max_avg_dpt_salary;
2
3
```

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

Results Explain Describe Saved SQL History

3 rows returned in 0.02 seconds Download

## DML Operations and Views

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

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

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

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

```
SELECT owner, table_name, column_name, updatable,insertable, deletable
```

```
FROM user_updatable_columns WHERE LOWER(table_name) = 'copy_d_cds';
```



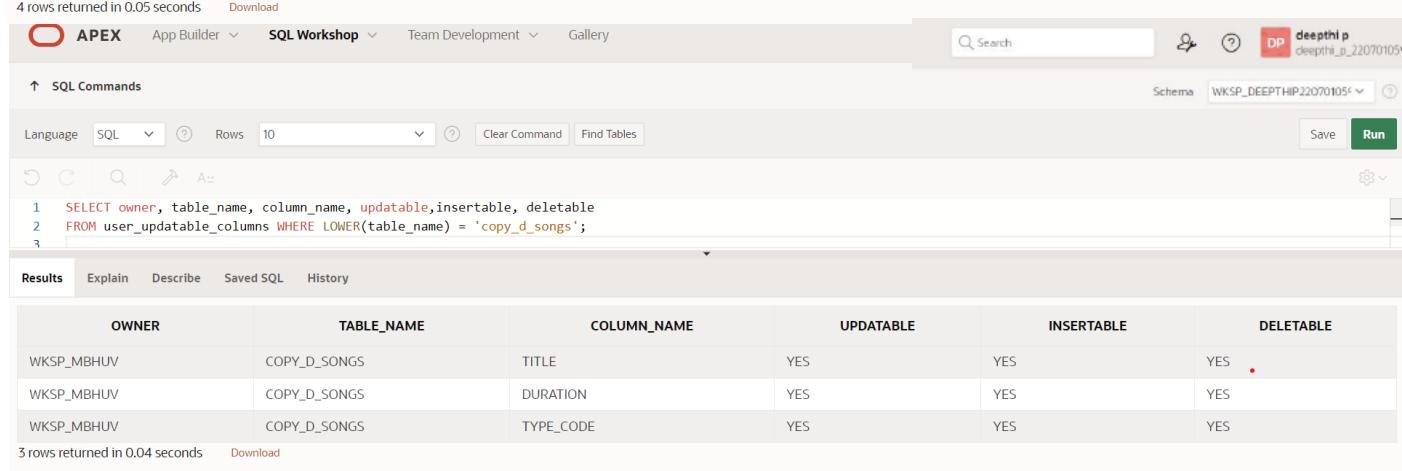
The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, and Gallery. On the right, there's a search bar, a user icon, and a session identifier 'deepthi p deepthi\_p\_22070105e'. Below the navigation is a toolbar with icons for Undo, Redo, Find, and Save/Run. The main area is titled 'SQL Commands' and contains a code editor with the following SQL:

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

Below the code editor is a results grid with the following columns: OWNER, TABLE\_NAME, COLUMN\_NAME, UPDATABLE, INSERTABLE, and DELETABLE. The data returned is:

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

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



The screenshot shows the Oracle SQL Workshop interface, identical to the one above but with a different query. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, and Gallery. On the right, there's a search bar, a user icon, and a session identifier 'deepthi p deepthi\_p\_22070105e'. Below the navigation is a toolbar with icons for Undo, Redo, Find, and Save/Run. The main area is titled 'SQL Commands' and contains the following SQL:

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

Below the code editor is a results grid with the following columns: OWNER, TABLE\_NAME, COLUMN\_NAME, UPDATABLE, INSERTABLE, and DELETABLE. The data returned is:

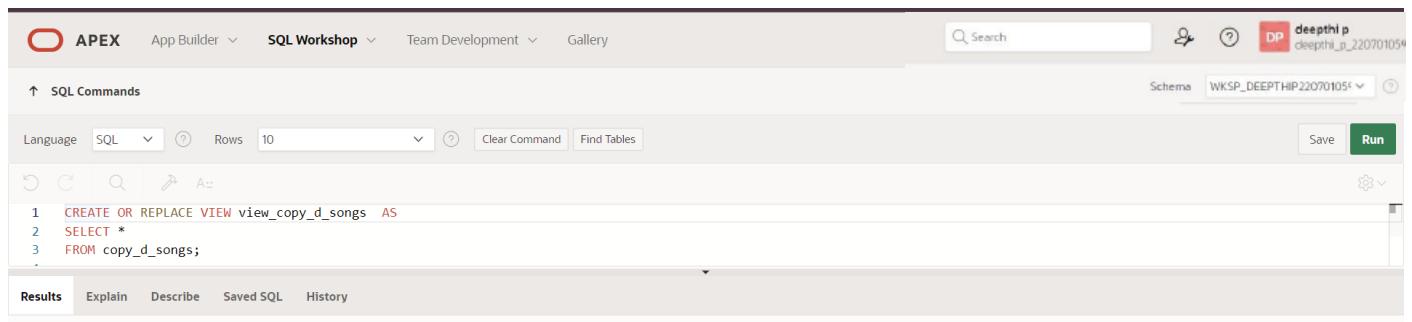
OWNER	TABLE_NAME	COLUMN_NAME	UPDATABLE	INSERTABLE	DELETABLE
WKSP_MBHVU	COPY_D_SONGS	TITLE	YES	YES	YES
WKSP_MBHVU	COPY_D_SONGS	DURATION	YES	YES	YES
WKSP_MBHVU	COPY_D_SONGS	TYPE_CODE	YES	YES	YES

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

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;
```

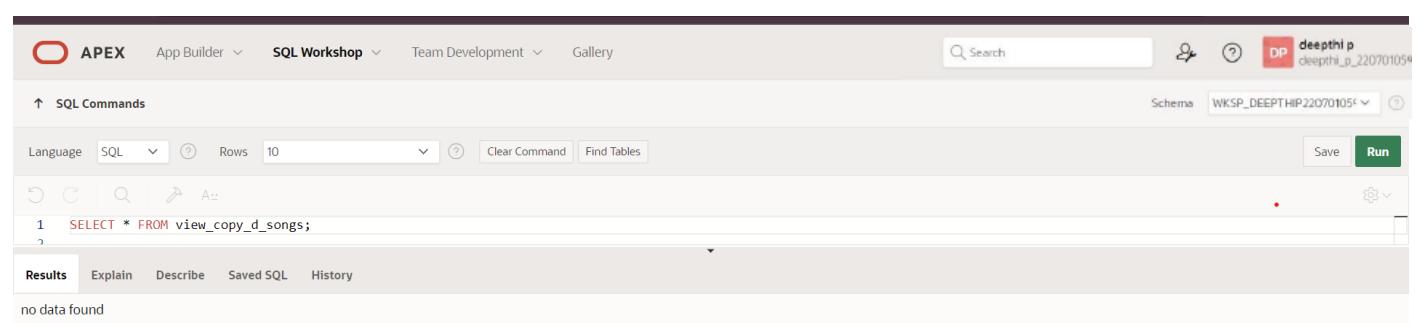
```
SELECT * FROM view_copy_d_songs;
```



This screenshot shows the Oracle SQL Workshop interface. In the top navigation bar, 'APEX' is selected. The main area is titled 'SQL Commands'. A code editor window contains the following SQL statements:

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

The 'Results' tab is active, showing the message 'View created.' and a execution time of '0.02 seconds'.



This screenshot shows the Oracle SQL Workshop interface. In the top navigation bar, 'APEX' is selected. The main area is titled 'SQL Commands'. A code editor window contains the following SQL statement:

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

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

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

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

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

The screenshot shows the Oracle SQL Workshop interface. In the top navigation bar, 'APEX' is selected. Below it, 'SQL Workshop' is active. The main area is titled 'SQL Commands'. A query is being run against schema 'WKSP\_DEEPTHIP220701054'. The code entered is:

```

1 INSERT INTO view_copy_d_songs(id,title,duration,artist,type_code)
2 VALUES(88,'Mello Jello','2 min','The What',4);
3

```

Below the code, the 'Results' tab is selected. It displays the message '1 row(s) inserted.' and a execution time of '0.03 seconds'.

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

```

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

```

```
SELECT * FROM read_copy_d_cds;
```

The screenshot shows the Oracle SQL Workshop interface. In the top navigation bar, 'APEX' is selected. Below it, 'SQL Workshop' is active. The main area is titled 'SQL Commands'. A query is being run against schema 'WKSP\_DEEPTHIP220701054'. The code entered is:

```

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

```

Below the code, the 'Results' tab is selected. It displays the message 'View created.' and a execution time of '0.04 seconds'.

The screenshot shows the Oracle SQL Workshop interface. In the top navigation bar, 'APEX' is selected. Below it, 'SQL Workshop' is active. The main area is titled 'SQL Commands'. A query is being run against schema 'WKSP\_DEEPTHIP220701054'. The code entered is:

```

1 SELECT * FROM read_copy_d_cds;

```

Below the code, the 'Results' tab is selected. It displays the message 'no data found'.

4. 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**

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

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

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there's a search bar, a user icon for 'deepthi p', and a schema dropdown set to 'WKSP\_DEEPTHIP22070105'. The main area is titled 'SQL Commands' with a sub-section '↑ SQL Commands'. It shows the following SQL code:

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

Below the code, there are tabs for Results, Explain, Describe, Saved SQL, and History. The 'Results' tab is selected, displaying the message 'View created.' and a execution time of '0.04 seconds'.

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

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

The screenshot shows the Oracle SQL Workshop interface, identical to the previous one but with a different query. The top navigation bar and schema selection are the same. The main area shows the following SQL code:

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

The 'Results' tab is selected, displaying the message '0 row(s) deleted.' and an execution time of '0.02 seconds'.

7. 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;
```

A screenshot of the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. A search bar and user profile are also present. The main area is titled 'SQL Commands' with a sub-section 'Rows: 10'. The code editor contains the following SQL command:

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

The results tab shows the output:

```
0 row(s) deleted.  
0.02 seconds
```

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

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

A screenshot of the Oracle SQL Workshop interface, identical to the previous one but with a different SQL command in the editor:

```
1  DELETE FROM read_copy_d_cds WHERE year = '2001';
```

The results show:

```
0 row(s) deleted.  
0.00 seconds
```

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

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

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

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

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

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

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

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

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

## Managing Views

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

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

```
SELECT * FROM view_copy_d_songs;
```

The screenshot shows two consecutive executions of SQL commands in Oracle SQL Workshop. In the first part, a view named 'view\_copy\_d\_songs' is created with the command:

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

In the second part, a SELECT \* statement is run against the newly created view:

```
SELECT * FROM view_copy_d_songs;
```

The results show one row returned:

TITLE	ARTIST
Mello Jello	The What

Execution details: 1 rows returned in 0.01 seconds.

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

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

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

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

```
DROP VIEW view_copy_d_songs;
```

The select statement is:

```
SELECT * FROM view_copy_d_songs;
```

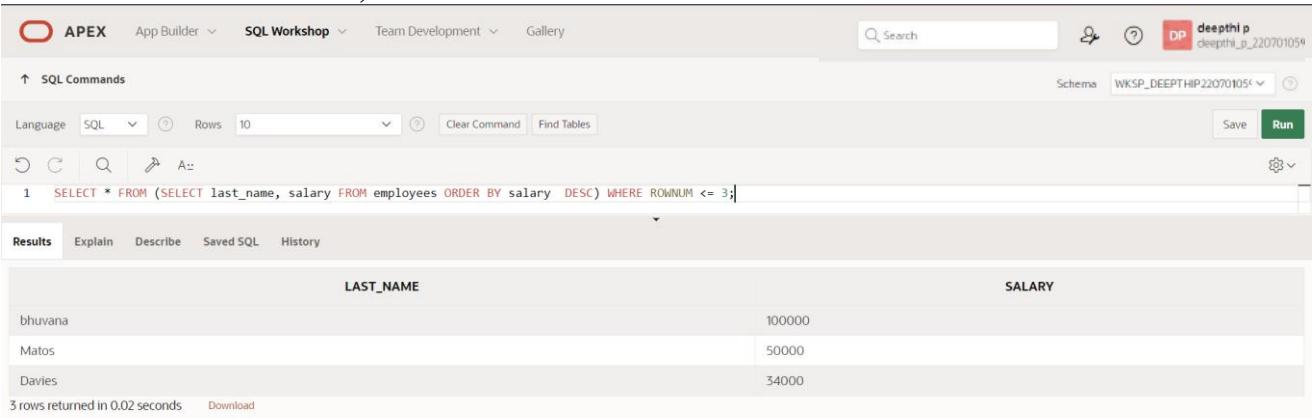
The results indicate that the view no longer exists:

ORA-00942: table or view does not exist

Execution details: View dropped.

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

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

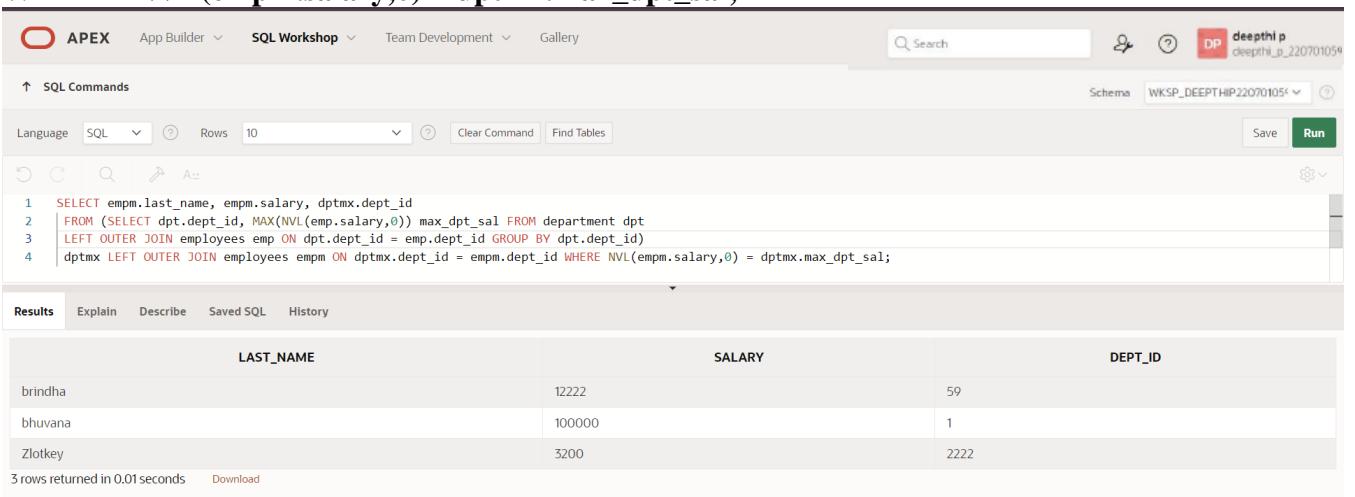


The screenshot shows the Oracle SQL Workshop interface. The query `SELECT * FROM (SELECT last_name, salary FROM employees ORDER BY salary DESC) WHERE ROWNUM <= 3;` is entered in the command line. The results table has columns `LAST_NAME` and `SALARY`, displaying three rows: bhuvana (100000), Matos (50000), and Davies (34000). The total execution time is 0.02 seconds.

LAST_NAME	SALARY
bhuvana	100000
Matos	50000
Davies	34000

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

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

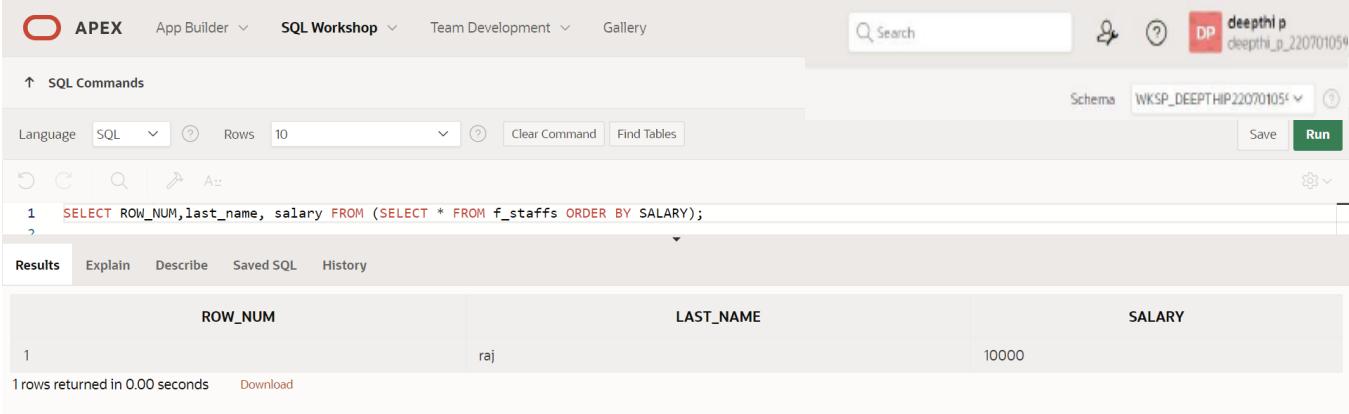


The screenshot shows the Oracle SQL Workshop interface. The query uses an inline view to calculate the maximum salary for each department and then joins it with the employees table to find employees whose salary matches their department's maximum salary. The results table has columns `LAST_NAME`, `SALARY`, and `DEPT_ID`, displaying three rows: brindha (12222, DEPT\_ID 59), bhuvana (100000, DEPT\_ID 1), and Zlotkey (3200, DEPT\_ID 2222). The total execution time is 0.01 seconds.

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

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

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



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, and Gallery. On the right, there's a search bar, user profile, and schema dropdown set to WKSP\_DEEPTHIP22070105. The main area has tabs for SQL Commands, Results, Explain, Describe, Saved SQL, and History. The SQL Commands tab shows the executed query:

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

The Results tab displays the output:

ROW_NUM	LAST_NAME	SALARY
1	raj	10000

1 rows returned in 0.00 seconds [Download](#)

## Indexes and Synonyms

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

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

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

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

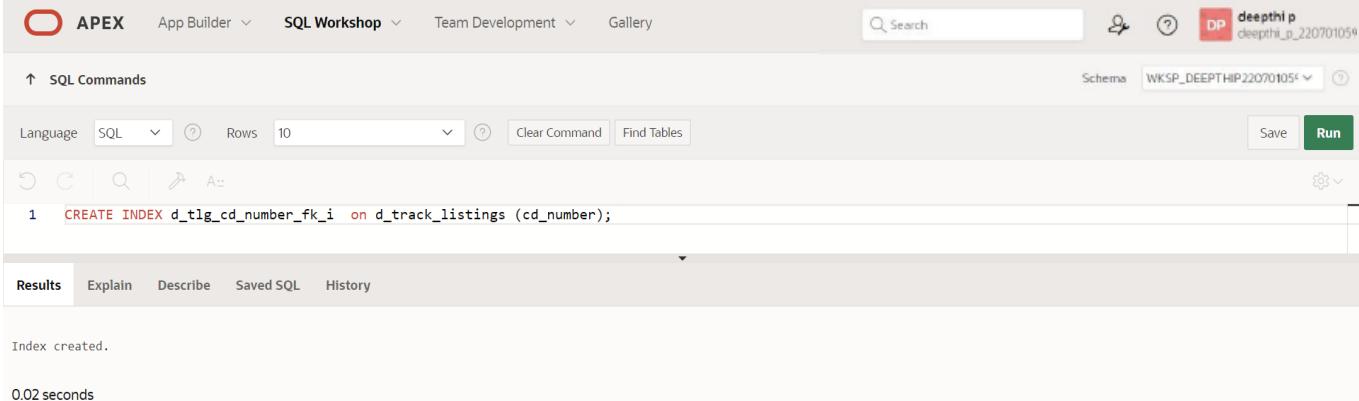
**Indexes use ROWID's (base 64 string representation of the row address containing block identifier, row location in the block and the database file identifier) which is the fastest way to access any particular row.**

3. When will an index be created automatically?

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

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

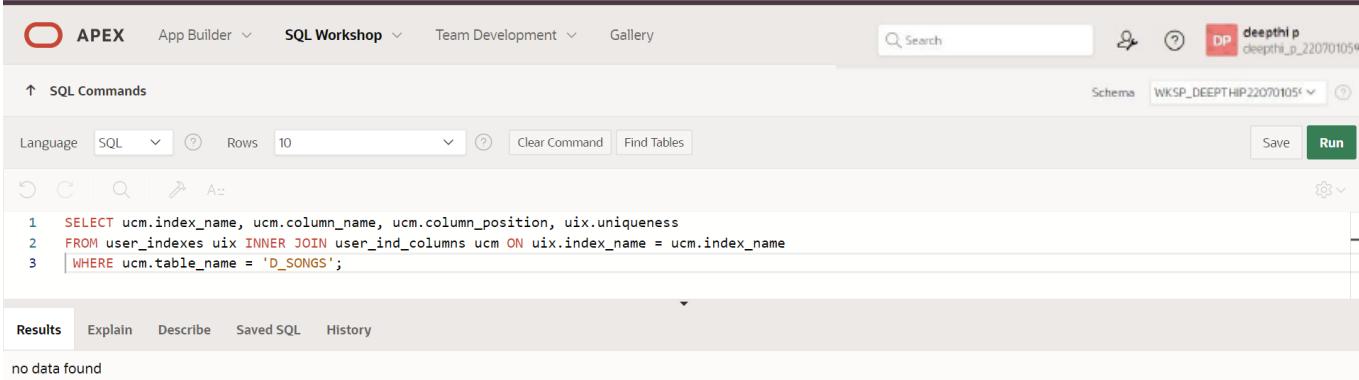
**CREATE INDEX d\_tlg\_cd\_number\_fk\_i  
on d\_track\_listings (cd\_number);**



The screenshot shows the Oracle Application Express SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, and Gallery. The SQL Workshop tab is selected. The schema dropdown shows 'WKSP\_DEEPTHIP22070105'. The main area displays the SQL command: 'CREATE INDEX d\_tlg\_cd\_number\_fk\_i ON d\_track\_listings (cd\_number);'. Below the command, the message 'Index created.' is displayed, followed by '0.02 seconds' execution time.

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

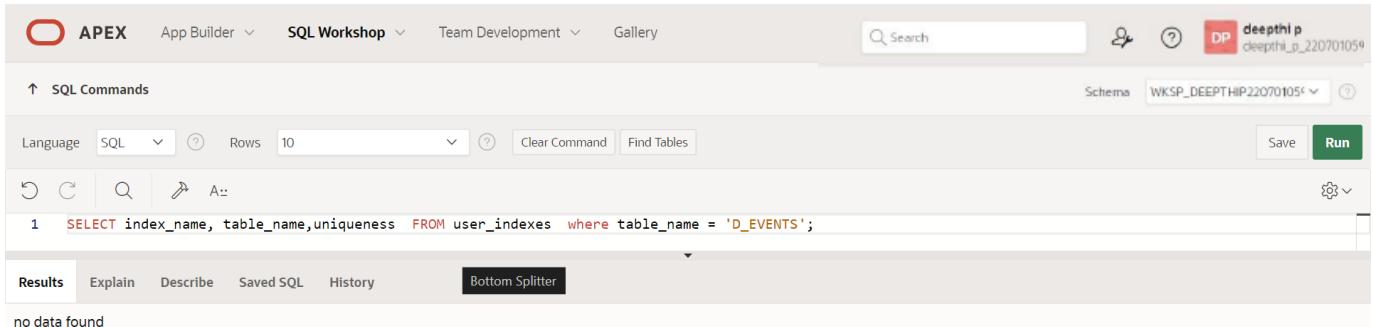
**SELECT ucm.index\_name, ucm.column\_name, ucm.column\_position, uix.uniqueness  
FROM user\_indexes uix INNER JOIN user\_ind\_columns ucm ON uix.index\_name =  
ucm.index\_name  
WHERE ucm.table\_name = 'D\_SONGS';**



The screenshot shows the Oracle Application Express SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, and Gallery. The SQL Workshop tab is selected. The schema dropdown shows 'WKSP\_DEEPTHIP22070105'. The main area displays the query: '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';'. Below the query, the message 'no data found' is displayed.

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';**



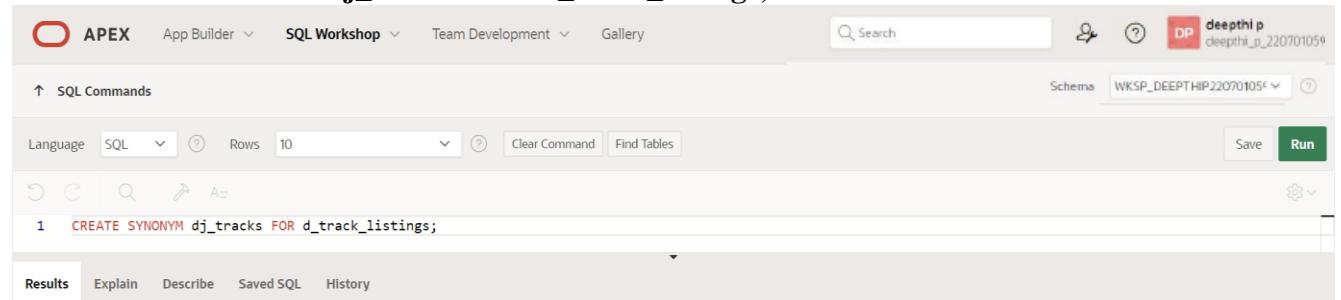
A screenshot of the Oracle SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. On the right side, there's a search bar, user profile information for 'deepthi p', and a schema dropdown set to 'WKSP\_DEEPTHIP220701054'. The main area shows a SQL command window with the following content:

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

The results tab shows the message "no data found".

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

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



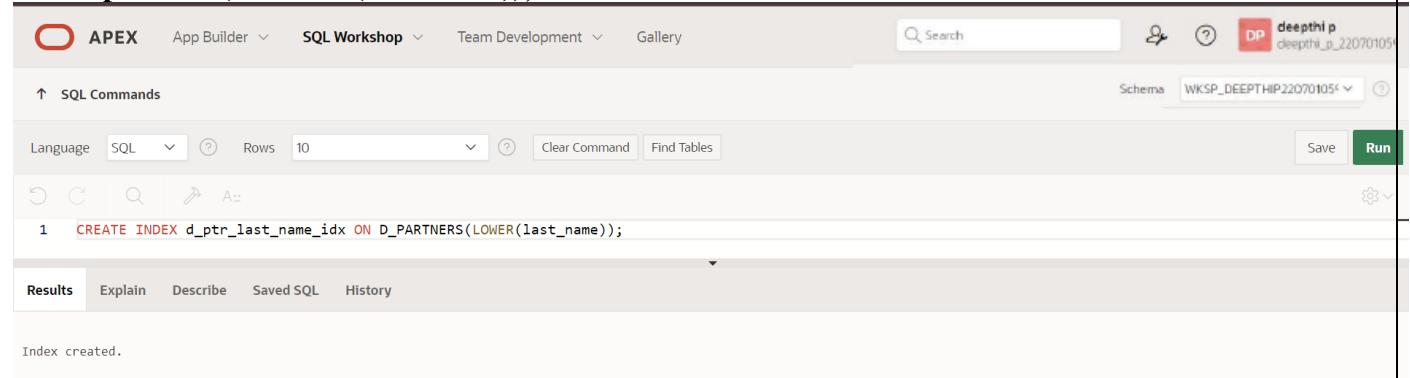
A screenshot of the Oracle SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. On the right side, there's a search bar, user profile information for 'deepthi p', and a schema dropdown set to 'WKSP\_DEEPTHIP220701054'. The main area shows a SQL command window with the following content:

```
1  CREATE SYNONYM dj_tracks FOR d_track_listings;
```

The results tab shows the message "Synonym created." and a execution time of "0.01 seconds".

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

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



A screenshot of the Oracle SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. On the right side, there's a search bar, user profile information for 'deepthi p', and a schema dropdown set to 'WKSP\_DEEPTHIP220701054'. The main area shows a SQL command window with the following content:

```
1  CREATE INDEX d_ptr_last_name_idx ON D_PARTNERS(LOWER(last_name));
```

The results tab shows the message "Index created." and a execution time of "0.02 seconds".

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

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

A screenshot of the Oracle SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. A search bar and a user icon for 'deepthi p' are also present. The main area shows a SQL command being run:

```
1 CREATE SYNONYM dj_tracks2 FOR d_track_listings;
```

The results section shows the message: "Synonym created." and a execution time of "0.01 seconds".

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

A screenshot of the Oracle SQL Workshop interface, identical to the previous one, but showing the results of the query:

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

The results table displays two rows:

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

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

10. Drop the synonym that you created in question

**DROP SYNONYM dj\_tracks2;**

A screenshot of the Oracle SQL Workshop interface. The SQL command is:

```
1 DROP SYNONYM dj_tracks2;
```

The results show the message: "Synonym dropped." and an execution time of "0.02 seconds".

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

**RESULT:**

## OTHER DATABASE OBJECTS

EX.NO:14

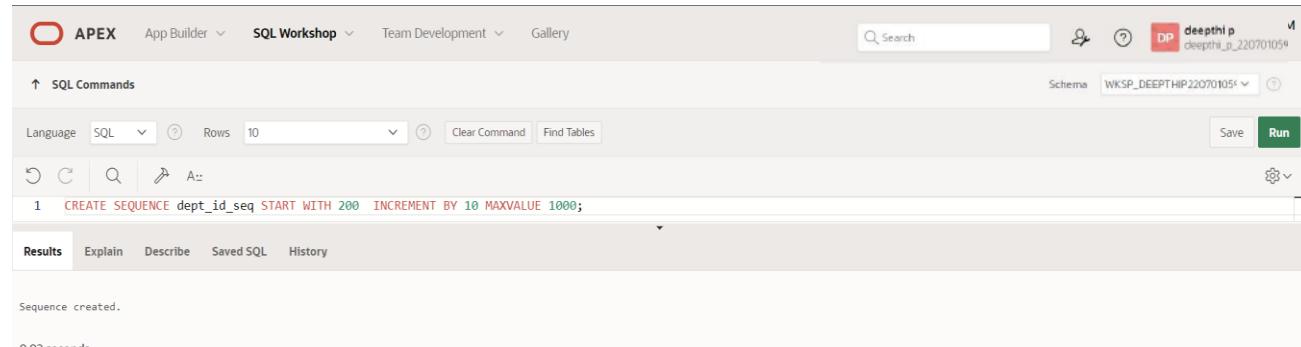
DATE:

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

**QUERY:**

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

**OUTPUT:**



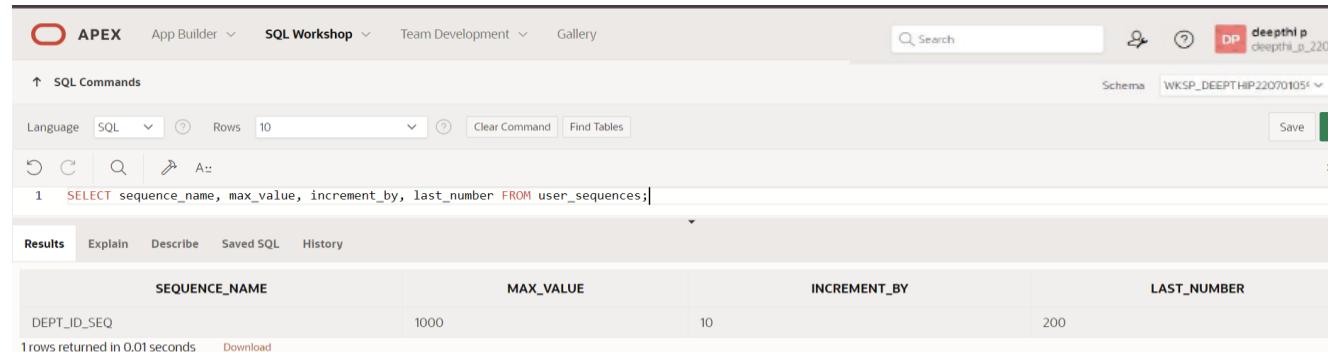
The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there's a user icon for 'deepthi p' and a schema dropdown set to 'WKSP\_DEEPTHIP220701054'. The main area is titled 'SQL Commands' with a search bar and a 'Run' button. Below the title, there are buttons for Undo, Redo, Find, and Paste. The SQL command 'CREATE SEQUENCE dept\_id\_seq START WITH 200 INCREMENT BY 10 MAXVALUE 1000;' is entered in the command line. The results tab shows the output: 'Sequence created.' and a execution time of '0.02 seconds'.

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

**QUERY:**

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

**OUTPUT:**



The screenshot shows the Oracle SQL Workshop interface again. The top navigation bar and schema selection are identical to the previous screenshot. The main area has 'SQL Commands' selected. The SQL command 'SELECT sequence\_name, max\_value, increment\_by, last\_number FROM user\_sequences;' is entered. The results tab displays the following table:

SEQUENCE_NAME	MAX_VALUE	INCREMENT_BY	LAST_NUMBER
DEPT_ID_SEQ	1000	10	200

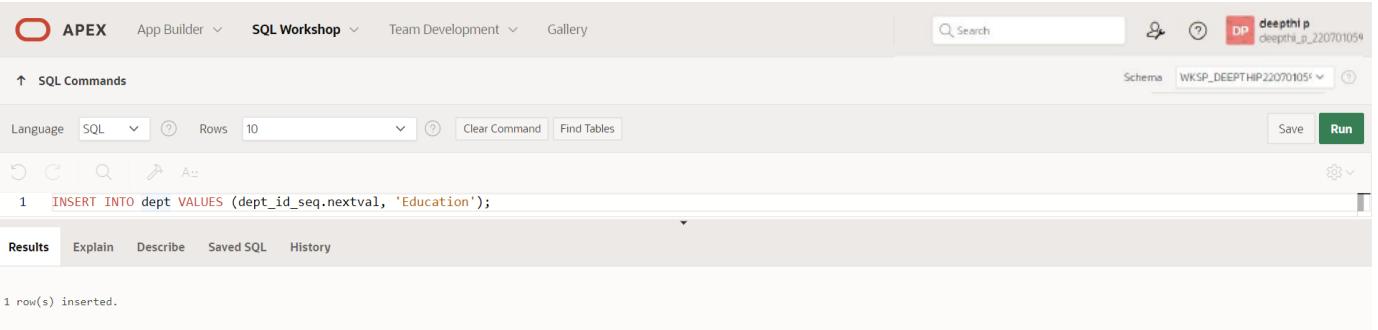
Below the table, it says '1 rows returned in 0.01 seconds' and has a 'Download' link.

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

**QUERY:**

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

**OUTPUT:**



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The search bar contains 'Search'. The schema dropdown shows 'WKSP\_DEEPTHIP22070105'. The main area displays the following SQL command and its execution results:

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

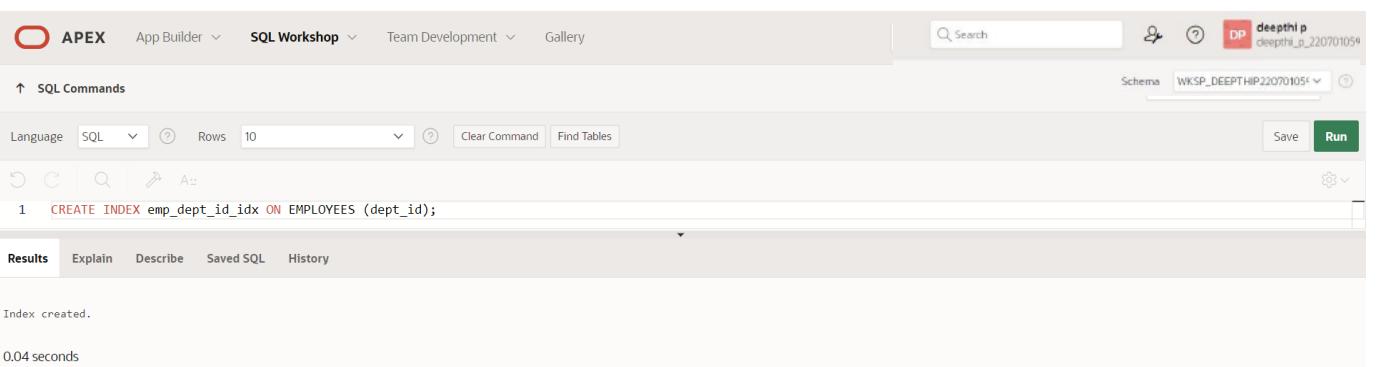
Results tab: 1 row(s) inserted.  
Time taken: 0.02 seconds

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

**QUERY:**

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

**OUTPUT:**



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The search bar contains 'Search'. The schema dropdown shows 'WKSP\_DEEPTHIP22070105'. The main area displays the following SQL command and its execution results:

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

Results tab: Index created.  
Time taken: 0.04 seconds

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

**QUERY:**

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

**OUTPUT:**

APEX App Builder SQL Workshop Team Development Gallery

↑ SQL Commands

Language SQL Rows 10 Clear Command Find Tables Save Run

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

Results Explain Describe Saved SQL History

INDEX_NAME	TABLE_NAME	UNIQUENESS
EMP_DEPT_ID_IDX	EMPLOYEES	NONUNIQUE

1 rows returned in 0.05 seconds Download

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

**RESULT:**

## **CONTROLLING USER ACCESS**

**EX.NO:15**

**DATE:**

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

The CREATE SESSION system privilege

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

The CREATE TABLE privilege

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

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

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

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

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

The ALTER USER statement

**6. Grant another user access to your DEPARTMENTS table. Have the user grant you query access**

**to his or her DEPARTMENTS table.**

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

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

**7. Query all the rows in your DEPARTMENTS table.**

SELECT \* FROM departments;

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

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

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

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

SELECT table\_name FROM user\_tables;

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

Team 1 revokes the privilege.

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

Team 2 revokes the privilege.

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

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

Team 1 executes this INSERT statement.

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

Team 2 executes this INSERT statement.

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

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

**RESULT:**

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

**EX.NO:16**

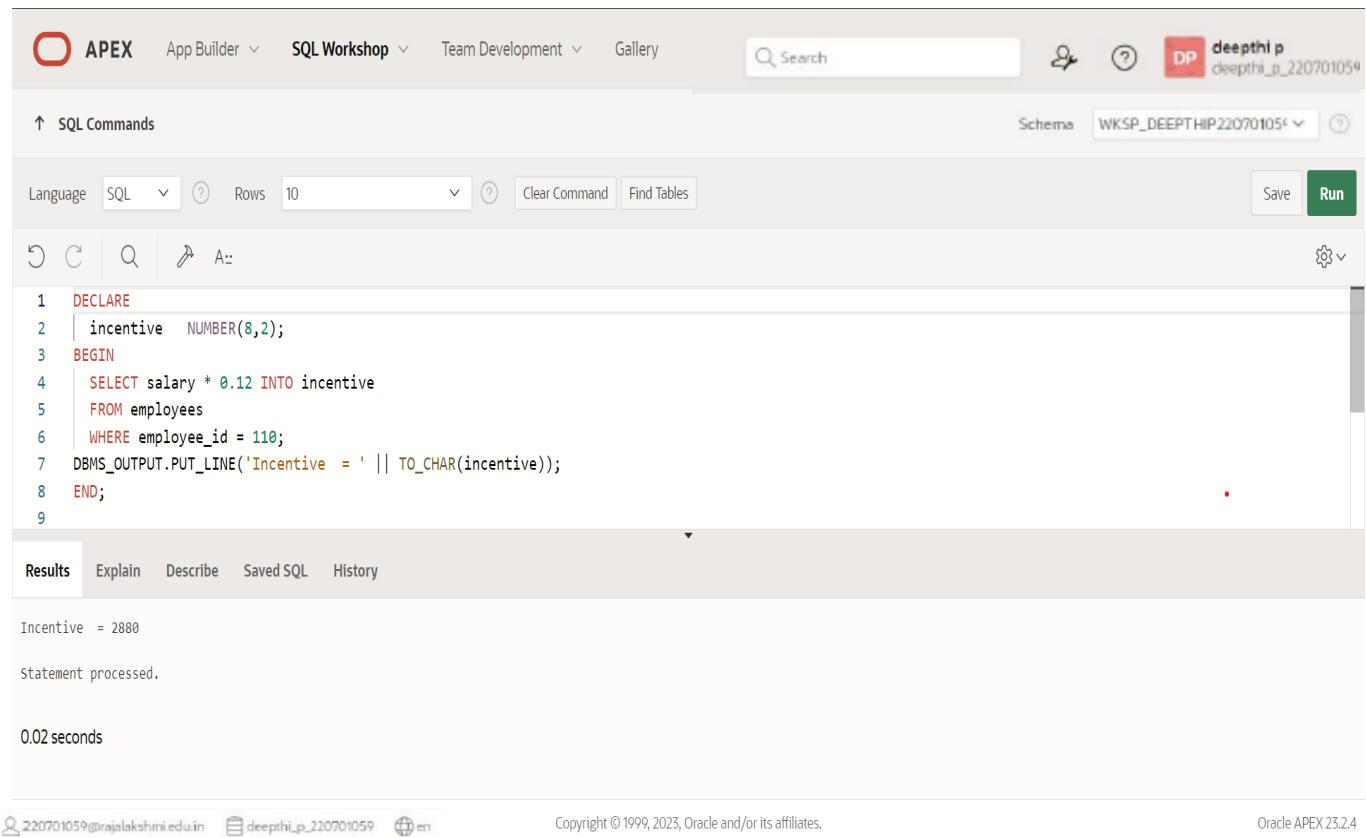
**DATE:**

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

**QUERY:**

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

**OUTPUT:**



The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for App Builder, SQL Workshop, Team Development, and Gallery, along with a search bar and user profile information. The main workspace is titled "SQL Commands" and shows the following content:

Language: SQL Rows: 10

```
1 DECLARE
2     incentive NUMBER(8,2);
3 BEGIN
4     SELECT salary * 0.12 INTO incentive
5     FROM employees
6     WHERE employee_id = 110;
7     DBMS_OUTPUT.PUT_LINE('Incentive = ' || TO_CHAR(incentive));
8 END;
9
```

Below the code, there are tabs for Results, Explain, Describe, Saved SQL, and History. The Results tab displays the output of the executed query:

```
Incentive = 2880
Statement processed.

0.02 seconds
```

At the bottom of the page, there are footer links for 220701059@rajalakshmi.edu.in, deepthi\_p\_220701059, and en, along with copyright and version information: Copyright © 1999, 2023, Oracle and/or its affiliates. Oracle APEX 23.2.4.

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 code editor, a PL/SQL block is written:

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

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
```

Below the code editor, the results pane shows the error message again:

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

At the bottom of the page, there are footer links for user information and copyright.

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:

```

1 DECLARE
2   salary_of_emp  NUMBER(8,2);
3
4 PROCEDURE approx_salary (
5   emp          NUMBER,
6   empsal      IN OUT NUMBER,
7   address      NUMBER
8 ) IS
9 BEGIN
10   empsal := empsal + address;

```

Results Explain Describe Saved SQL History

Before invoking procedure, salary\_of\_emp: 34000  
After invoking procedure, salary\_of\_emp: 35000  
Statement processed.

0.01 seconds

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

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

## QUERY:

```

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

```

## OUTPUT:

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

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

```

Below the code, there are tabs for Results, Explain, Describe, Saved SQL, and History. The Results tab shows the output: "Procedure created." and "0.01 seconds". The bottom of the page displays copyright information for Oracle and the APEX version.

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 APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. A search bar and user profile 'deepthi p' are also present. The main area is titled 'SQL Commands' and shows the following PL/SQL code:

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

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

```
TRUE
FALSE
```

Below the results, it says "Statement processed." and "0.01 seconds". At the bottom, there are footer links for '220701059@rajalakshmi.edu.in', 'deepthi\_p\_220701059', and 'en'. The copyright notice reads "Copyright © 1999, 2023, Oracle and/or its affiliates." and the version is "Oracle APEX 23.2.4".

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 APEX SQL Workshop interface. At the top, there are tabs for APEX, App Builder, SQL Workshop, Team Development, and Gallery. On the right, there's a search bar, user profile for 'deepthi p', and a schema dropdown set to 'WKSP\_DEEPTHI\_P\_22070105'. Below the tabs, there's a toolbar with Language (SQL), Rows (10), Clear Command, Find Tables, Save, and Run buttons. The main area contains a code editor with the following PL/SQL block:

```

1  DECLARE
2      num_small NUMBER := 8;
3      num_large NUMBER := 5;
4      num_temp NUMBER;
5      BEGIN
6
7          IF num_small > num_large THEN
8              num_temp := num_small;
9              num_small := num_large;
10             num_large := num_temp;
11         END IF;

```

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

```

num_small = 5
num_large = 8

Statement processed.

```

At the bottom, there are footer links for support, copyright information (Copyright © 1999, 2023, Oracle and/or its affiliates.), and the Oracle APEX version (Oracle APEX 23.2.4).

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 APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. A search bar and a user icon for 'deepthi p' are also present. The main area displays the following PL/SQL code:

```
1  DECLARE
2      PROCEDURE test1 (
3          sal_achieve NUMBER,
4          target_qty NUMBER,
5          emp_id NUMBER
6      )
7      IS
8          incentive NUMBER := 0;
9          updated VARCHAR2(3) := 'No';
10         BEGIN
11             IF sal_achieve > (target_qty + 200) THEN
12                 incentive := 150;
13             ELSE
14                 incentive := 75;
15             END IF;
16             DBMS_OUTPUT.NEW_LINE;
17             DBMS_OUTPUT.PUT_LINE (
18                 'Sale achieved : ' || sal_achieve || ', incentive : ' || incentive || ')';
19         END;
20     END test1;
```

Below the code, there are tabs for Results, Explain, Describe, Saved SQL, and History. The Results tab is selected, showing the output of the procedure execution:

```
Table updated? Yes, incentive = 75.
Table updated? Yes, incentive = 150.

1 row(s) updated.

0.02 seconds
```

At the bottom of the interface, the user's email (220701059@prajalakshmi.edu.in), session ID (deepthi\_p\_220701059), and language (en) are displayed, along with the copyright notice (Copyright © 1999, 2023, Oracle and/or its affiliates) and the version (Oracle APEX 23.2.4).

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

test1(28000);

END;

## **OUTPUT:**

The screenshot shows the Oracle APEX interface with the SQL Workshop tab selected. The code editor contains a PL/SQL procedure named `test1`. The procedure takes a parameter `sal_achieve` of type `NUMBER`. It initializes a variable `incentive` to 0. The procedure then uses a series of IF-ELSIF-ELSE statements to determine the incentive based on the value of `sal_achieve`. Finally, it outputs the results using `DBMS_OUTPUT.PUT_LINE`. The results pane at the bottom shows three executions of the procedure with different input values: 45000, 36000, and 28000, each resulting in an incentive of 1800, 800, and 500 respectively. The output also includes a statement processed message.

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

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

```
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. At the top, there are tabs for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there's a search bar, a user icon, and a schema dropdown set to 'WKSP\_DEEPTHIP220701054'. Below the tabs, there's a toolbar with Language (SQL), Rows (10), Clear Command, Find Tables, Save, and Run buttons. The main area contains a code editor with the following PL/SQL block:

```
1 DECLARE
2   CURSOR emp_cursor IS
3     SELECT e.employee_id, e.first_name, m.first_name AS manager_name
4       FROM employees e
5      LEFT JOIN employees m ON e.manager_id = m.employee_id;
6   emp_record emp_cursor%ROWTYPE;
```

Below the code editor, there are tabs for Results, Explain, Describe, Saved SQL, and History. The Results tab displays the output of the query:

```
Employee ID: 122
Employee Name: ramesh
Manager Name:
-----
Employee ID: 4
Employee Name:
Manager Name:
-----
Employee ID: 2
Employee Name: john
Manager Name:
-----
Employee ID: 5
Employee Name: sneha
Manager Name:
```

At the bottom of the interface, there are footer links for support, documentation, and Oracle APEX 23.2.

**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

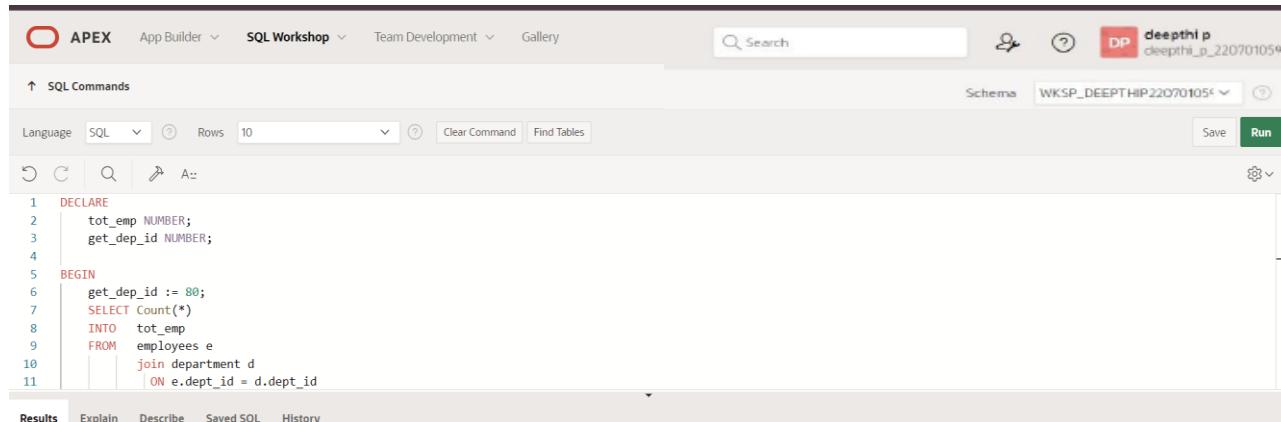
```
get_dep_id := 80;
SELECT Count(*)
INTO tot_emp
FROM employees e
join departments d
ON e.department_id = d.dept_id
WHERE e.department_id = get_dep_id;
```

```
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 code entered is:

```
1 DECLARE
2     tot_emp NUMBER;
3     get_dep_id NUMBER;
4
5 BEGIN
6     get_dep_id := 80;
7     SELECT Count(*)
8     INTO tot_emp
9     FROM employees e
10    join department d
11    ON e.dept_id = d.dept_id
```

The results pane displays the output of the query:

```
The employees are in the department 80 is: 0
There are 45 vacancies in department 80

Statement processed.
```

Execution details: 0.02 seconds

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

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:

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

```
8   SELECT employee_id, first_name || ' ' || last_name AS full_name, job_id, hire_date, salary
9   FROM employees;
10  BEGIN
11    DBMS_OUTPUT.PUT_LINE('Employee ID | Full Name | Job Title | Hire Date | Salary');
12    DBMS_OUTPUT.PUT_LINE('-----');
13    OPEN c_employees;
14    FETCH c_employees INTO v_employee_id, v_full_name, v_job_id, v_hire_date, v_salary;
15    WHILE c_employees%FOUND LOOP
16      DBMS_OUTPUT.PUT_LINE(v_employee_id || ' ' || v_full_name || ' ' || v_job_id || ' ' || v_hire_date || ' ' || v_salary);
17      FETCH c_employees INTO v_employee_id, v_full_name, v_job_id, v_hire_date, v_salary;

```

The "Results" tab is selected, displaying the output of the program:

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

Below the results, a message states "Statement processed." and the footer includes copyright information for Oracle and the APEX version.

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 SQL Workshop tab is selected. The main area displays a PL/SQL code block and its execution results.

```
1  DECLARE
2      CURSOR emp_cursor IS
3          SELECT e.employee_id, e.first_name, m.first_name AS manager_name
4          FROM employees e
5              LEFT JOIN employees m ON e.manager_id = m.employee_id;
6      emp_record emp_cursor%ROWTYPE;
```

The results pane shows the output of the cursor query:

```
Employee ID: 122
Employee Name: ramesh
Manager Name:
-----
Employee ID: 4
Employee Name:
Manager Name:
-----
Employee ID: 2
Employee Name: john
Manager Name:
-----
Employee ID: 5
Employee Name: sneha
Manager Name:
```

At the bottom of the interface, there are footer links for support, legal notices, and copyright information.

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 SQL Workshop interface. In the top navigation bar, 'APEX' is selected. Below it, tabs for 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery' are visible. A search bar and user profile 'deepthi p deepthi\_p\_220701054' are on the right. The main area has a toolbar with icons for copy, cut, search, and refresh. The code editor contains the following PL/SQL block:

```

1  DECLARE
2    CURSOR job_cursor IS
3      SELECT e.job_id, j.lowest_sal
4      FROM job_grades j,employees e;
5    job_record job_cursor%ROWTYPE;
6    BEGIN
7      OPEN job_cursor;
8      FETCH job_cursor INTO job_record;
9      WHILE job_cursor%FOUND LOOP

```

The results pane shows the output of the query:

```

Job ID: st_clerk
Minimum Salary: 11000000
-----
Job ID: st_clerk
Minimum Salary: 11000000
-----
Job ID: st_joseph
Minimum Salary: 11000000
-----
Job ID: st_mary
Minimum Salary: 11000000

```

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

#### QUERY:

DECLARE

```

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

```

## OUTPUT:

The screenshot shows the Oracle APEX interface with the SQL Workshop tab selected. The code area contains a PL/SQL block:

```
1  DECLARE
2      CURSOR employees_cur IS
3          SELECT employee_id, last_name, job_id, start_date
4          FROM employees NATURAL JOIN job_history;
5          emp_start_date DATE;
6  BEGIN
7      dbms_output.Put_line(Rpad('Employee ID', 15) || Rpad('Last Name', 25) || Rpad('Job Id', 35) || 'Start Date');
8      dbms_output.Put_line('-----');
9      FOR emp_sal_rec IN employees_cur LOOP
10          -- find out most recent end_date in job_history
11          SELECT ...
```

The results section shows the output:

Employee ID	Last Name	Job Id	Start Date

Statement processed.  
0.03 seconds

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

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:

APEX App Builder SQL Workshop Team Development Gallery

Search

Schema: WKSP\_DEEPTHIP220701054

SQL Commands

Language: SQL Rows: 10 Clear Command Find Tables Save Run

```

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 END;

```

Results Explain Describe Saved SQL History

```

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

```

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

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

**RESULT:**

## PROCEDURES AND FUNCTIONS

EX.NO: 17

DATE:

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

**QUERY:**

DECLARE

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

**OUTPUT:**

The screenshot shows the Oracle 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 'deepthi\_p' and the schema 'WKSP\_DEEPTHIP22070105'. The main workspace is titled 'SQL Commands' and contains the following PL/SQL code:

```
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;
```

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

```
120
Statement processed.
0.01 seconds
```

At the bottom, footer information includes the URL '22070105@prajalakshmi.edu.in', the session ID 'deepthi\_p\_22070105', and the copyright notice 'Copyright © 1999, 2023, Oracle and/or its affiliates.' The page also indicates it is 'Oracle APEX 23.2.4'.

**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 links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. The SQL Workshop tab is selected. The schema dropdown shows 'WKSP\_DEEPTHIP220701054'. The main area displays the following SQL code:

```
1 CREATE TABLE books (
2     book_id NUMBER PRIMARY KEY,
3     title VARCHAR2(100),
4     author VARCHAR2(100),
5     year_published NUMBER
6 );
7 INSERT INTO books (book_id, title, author, year_published) VALUES (1, '1984', 'George Orwell', 1949);
8 INSERT INTO books (book_id, title, author, year_published) VALUES (2, 'To Kill a Mockingbird', 'Harper Lee', 1960);
9 INSERT INTO books (book_id, title, author, year_published) VALUES (3, 'The Great Gatsby', 'F. Scott Fitzgerald', 1925);
10
11 CREATE OR REPLACE PROCEDURE get_book_info (
12     p_book_id IN NUMBER,
13     p_title OUT VARCHAR2,
14     p_author OUT VARCHAR2,
15 );

```

The 'Results' tab is selected, showing the output of the executed query:

```
Title: 1984
Author: George Orwell
Year Published: 1949

Statement processed.
```

At the bottom, the URL is 220701059@prajalakshmi.edu.in, the session ID is deepthi\_p\_220701059, and the environment is en. Copyright information for Oracle APEX 23.2.4 is also present.

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

## RESULT:

# TRIGGER

EX\_NO: 18

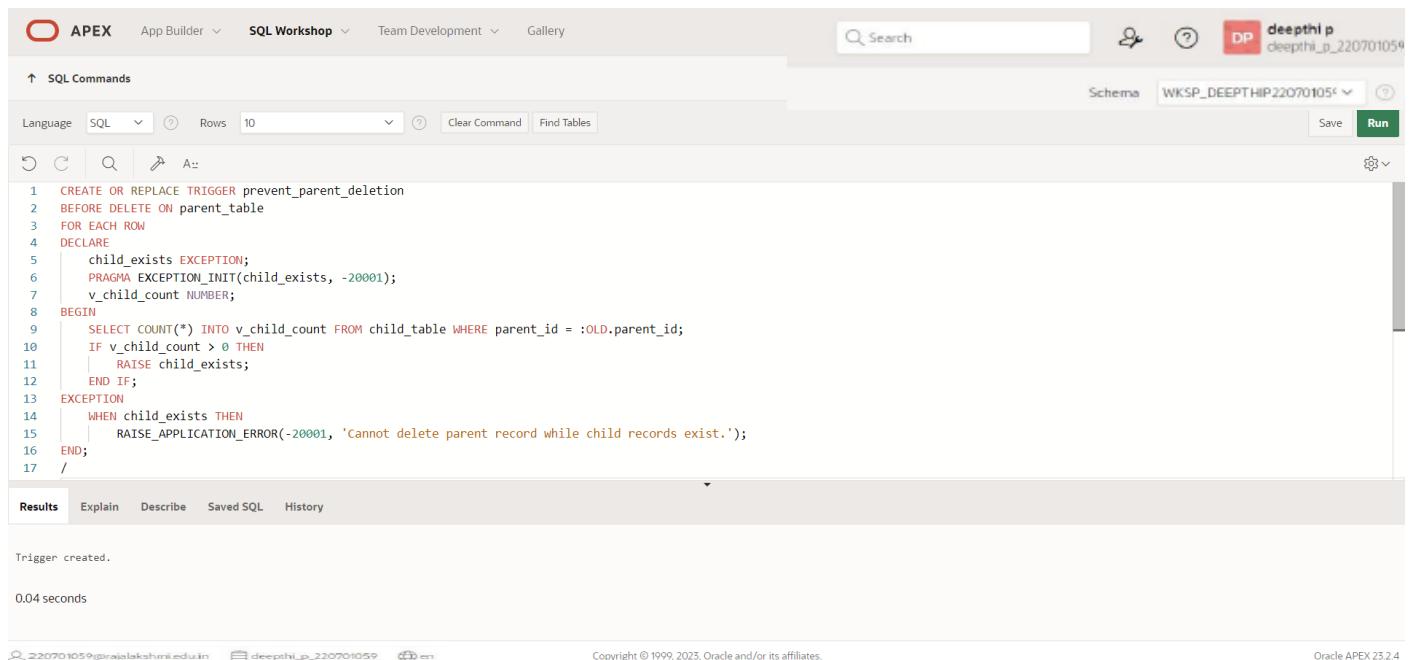
DATE:

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 SQL Workshop interface. In the top navigation bar, 'APEX' and 'SQL Workshop' are selected. The main area is titled 'SQL Commands'. The code for the trigger is pasted into the command editor. The 'Run' button at the bottom right is highlighted in green. Below the code, the results pane shows the message 'Trigger created.' and a execution time of '0.04 seconds'. The bottom footer includes copyright information for Oracle and a note about APEX version 23.2.4.

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

Trigger created.  
0.04 seconds

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

## QUERY:

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

## OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. In the top navigation bar, 'APEX' is selected. The main area displays the PL/SQL code for the trigger. Below the code, the 'Results' tab is active, showing the output: 'Trigger created.' and '0.04 seconds'. The bottom right corner indicates the version 'Oracle APEX 23.2.4'.

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

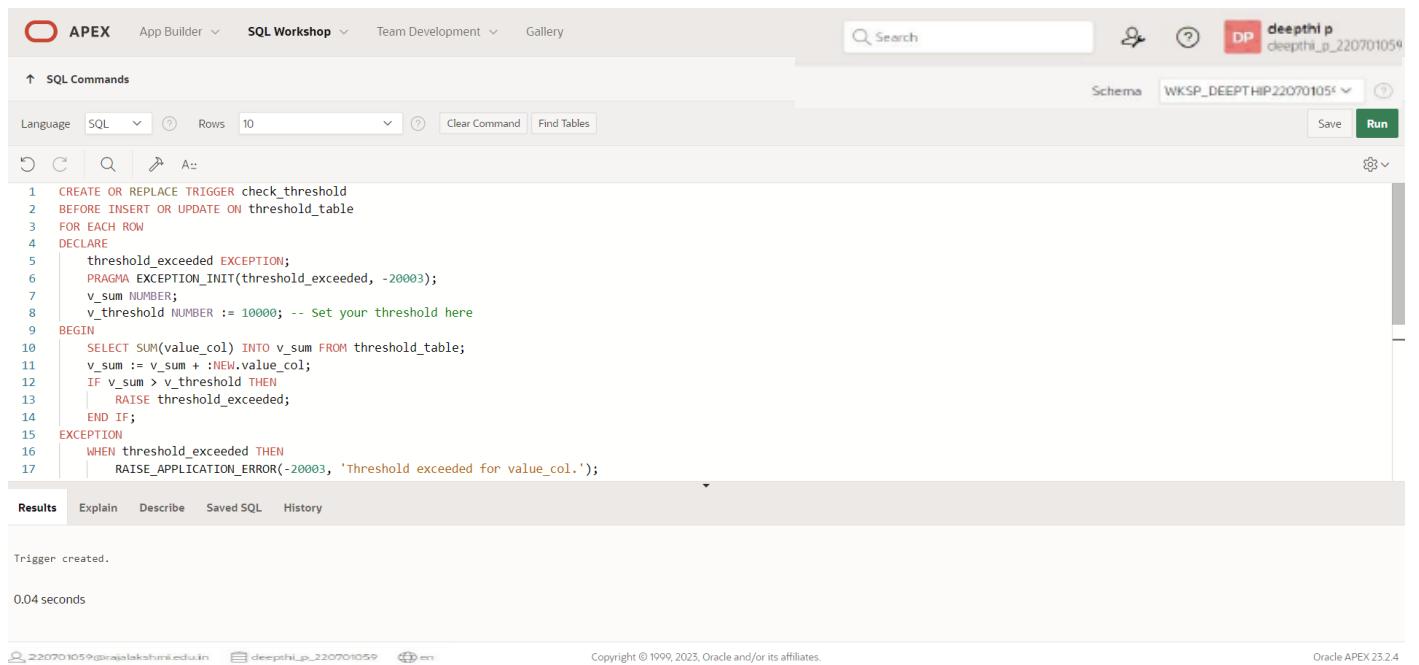
```

**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 'deepthi p' and a schema 'WKSP\_DEEPTHI\_P220701054'. The main area is titled 'SQL Commands' with a 'Language' dropdown set to 'SQL'. Below the code editor, there are buttons for 'Run' and 'Save'. The code editor contains the PL/SQL trigger definition shown above. The results tab at the bottom shows the message 'Trigger created.' and a execution time of '0.04 seconds'.

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

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

## QUERY:

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

## OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. In the top navigation bar, 'SQL Workshop' is selected. The main area contains the PL/SQL code for the trigger. Below the code, the 'Results' tab is active, showing the output: 'Trigger created.' and '0.03 seconds'. The bottom right corner indicates the session is connected to 'deepthi\_p\_22070105'.

```
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,
6     VALUES (audit_seq.NEXTVAL, :OLD.id, :OLD.col1, :NEW.col1, :OLD.col2, :NEW.col2,
7     SYSTIMESTAMP);
8 END;
9 /
```

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 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. The right side shows a user profile for 'deepthi p' and the schema 'WKSP\_DEEPTHIP22070105'. The main area is titled 'SQL Commands' with a 'Language' dropdown set to 'SQL'. Below the code editor, there are buttons for 'Save' and 'Run'. The code editor contains the following PL/SQL code:

```

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

```

Below the code editor, the 'Results' tab is selected, showing the message 'Trigger created.' and a execution time of '0.05 seconds'. The bottom of the screen displays copyright information for Oracle and the APEX version 'Oracle APEX 23.2.4'.

**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. In the top navigation bar, 'APEX' is selected. Below it, tabs for 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery' are visible. On the right side, a user profile for 'deepthi p' is shown, along with the schema name 'WKSP\_DEPTHI\_P\_220701054'. The main area is titled 'SQL Commands' and contains the following PL/SQL code:

```

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 /

```

Below the code, there are tabs for 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History'. The 'Results' tab is selected, showing the output: 'Trigger created.' and '0.03 seconds'. At the bottom, the session information '220701054@rajalakshmi.edu.in' and 'deepthi\_p\_220701054' is displayed, along with copyright information 'Copyright © 1999, 2023, Oracle and/or its affiliates.' and the version 'Oracle APEX 23.2.4'.

**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 links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The SQL Workshop tab has a dropdown menu showing 'deepthi\_p' and 'deepthi\_p\_220701054'. The main area is titled 'SQL Commands' with a 'Language' dropdown set to 'SQL'. The schema is set to 'WKSP\_DEEPTHIP220701054'. The 'Run' button is highlighted in green. Below the toolbar, there are icons for Undo, Redo, Search, and other common operations. The code editor contains the following PL/SQL trigger definition:

```
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
```

Below the code editor, the 'Results' tab is selected. The output shows the message 'Trigger created.' and a execution time of '0.05 seconds'. At the bottom, the session information shows '220701059@rajashankarshah.educare' and 'deepthi\_p\_220701059'. The copyright notice 'Copyright © 1999, 2023, Oracle and/or its affiliates.' and the version 'Oracle APEX 23.2.4' are also visible.

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:**

```
DEEPTHI_P_59> db.restaurants.find( { $or: [ { name: /^Wil/ }, { cuisine: { $nin: ['American', 'Chinese'] } } ] , { restaurant_id: 1, name: 1, borough: 1, cuisine: 1 } } );
[ {
  _id: ObjectId('664f3c798752f54dc3cdcf7'),
  borough: 'Bronx',
  cuisine: 'Bakery',
  name: 'Morris Park Bake Shop',
  restaurant_id: '30075445'
}]
DEEPTHI_P_59> |
```

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:**

```
DEEPTHI_P_59 > 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 } );
DEEPTHI_P_59 > |
```

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:

```
DEEPTHI_P_59 > db.restaurants.find({$and : [{"address.coord.1": {$gt : 42}}, {"address.coord.1": {$lte : 52}}]}, {_id:0, restaurant_id:1, name:1, address:1})
DEEPTHI_P_59 >
```

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:

```

| DEEPTHI_P_59> 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'
| }
|
| DEEPTHI_P_59 > |

```

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

#### QUERY:

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

#### OUTPUT:

```

| DEEPTHI_P_59> 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'
| }
|
| DEEPTHI_P_59 > |

```

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:

```

DEEPTHI_P_59> db.restaurants.find({}, { _id: 0 }).sort({ cuisine: 1, borough: -1 })
[
  {
    address: {
      building: '1007',
      coord: [ -73.856077, 40.848447 ],
      street: 'Morris Park Ave',
      zipcode: '10462'
    },
    borough: 'Bronx',
    cuisine: 'Bakery',
    grades: [
      {
        date: ISODate('2014-03-03T00:00:00.000Z'),
        grade: 'A',
        score: 2
      },
      {
        date: ISODate('2013-09-11T00:00:00.000Z'),
        grade: 'A',
        score: 6
      },
      {
        date: ISODate('2013-01-24T00:00:00.000Z'),
        grade: 'A',
        score: 10
      },
      {
        date: ISODate('2011-11-23T00:00:00.000Z'),
        grade: 'A',
        score: 9
      },
      {
        date: ISODate('2011-03-10T00:00:00.000Z'),
        grade: 'B',
        score: 14
      }
    ],
    name: 'Morris Park Bake Shop',
    restaurant_id: '30075445'
  }
]
E DEEPTHI_P_59 > |

```

**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:

```

DEEPTHI_P_59> db.restaurants.find({ "address.street": { $exists: true, $ne: "" } })
[
  {
    _id: ObjectId('664f3c798752f54dc3cdcdf7'),
    address: {
      building: '1007',
      coord: [ -73.856077, 40.848447 ],
      street: 'Morris Park Ave',
      zipcode: '10462'
    },
    borough: 'Bronx',
    cuisine: 'Bakery',
    grades: [
      {
        date: ISODate('2014-03-03T00:00:00.000Z'),
        grade: 'A',
        score: 2
      },
      {
        date: ISODate('2013-09-11T00:00:00.000Z'),
        grade: 'A',
        score: 6
      },
      {
        date: ISODate('2013-01-24T00:00:00.000Z'),
        grade: 'A',
        score: 10
      },
      {
        date: ISODate('2011-11-23T00:00:00.000Z'),
        grade: 'A',
        score: 9
      },
      {
        date: ISODate('2011-03-10T00:00:00.000Z'),
        grade: 'B',
        score: 14
      }
    ],
    name: 'Morris Park Bake Shop',
    restaurant_id: '30075445'
  }
]
E DEEPTHI_P_59 > |

```

**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:

```

DEEPTHI_P_59> db.restaurants.find({ "address.coord": { $elemMatch: { $type: "double" } } })
[ {
  _id: ObjectId('664f3c798752f54dc3cdcdf7'),
  address: {
    building: '1007',
    coord: [ -73.856077, 40.848447 ],
    street: 'Morris Park Ave',
    zipcode: '10462'
  },
  borough: 'Bronx',
  cuisine: 'Bakery',
  grades: [
    {
      date: ISODate('2014-03-03T00:00:00.000Z'),
      grade: 'A',
      score: 2
    },
    {
      date: ISODate('2013-09-11T00:00:00.000Z'),
      grade: 'A',
      score: 6
    },
    {
      date: ISODate('2013-01-24T00:00:00.000Z'),
      grade: 'A',
      score: 10
    },
    {
      date: ISODate('2011-11-23T00:00:00.000Z'),
      grade: 'A',
      score: 9
    },
    {
      date: ISODate('2011-03-10T00:00:00.000Z'),
      grade: 'B',
      score: 14
    }
  ],
  name: 'Morris Park Bake Shop',
  restaurant_id: '30075445'
}
]
DEEPTHI_P_59> |

```

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:

```

DEEPTHI_P_59> db.restaurants.find({ "grades.score": { $mod: [7, 0] } }, { restaurant_id: 1, name: 1, grades: 1 })
[ {
  _id: ObjectId('664f3c798752f54dc3cdcdf7'),
  grades: [
    {
      date: ISODate('2014-03-03T00:00:00.000Z'),
      grade: 'A',
      score: 2
    },
    {
      date: ISODate('2013-09-11T00:00:00.000Z'),
      grade: 'A',
      score: 6
    },
    {
      date: ISODate('2013-01-24T00:00:00.000Z'),
      grade: 'A',
      score: 10
    },
    {
      date: ISODate('2011-11-23T00:00:00.000Z'),
      grade: 'A',
      score: 9
    },
    {
      date: ISODate('2011-03-10T00:00:00.000Z'),
      grade: 'B',
      score: 14
    }
  ],
  name: 'Morris Park Bake Shop',
  restaurant_id: '30075445'
}
]
DEEPTHI_P_59> |

```

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:

```
DEEPTHI_P_59 > db.restaurants.find({ name: /mon/i }, { name: 1, borough: 1, "address.coord": 1, cuisine: 1 })  
DEEPTHI_P_59 > |
```

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:**

```
DEEPTHI_P_59 > db.restaurants.find({ name: /^Mad/i }, { name: 1, borough: 1, "address.coord": 1, cuisine: 1 })  
DEEPTHI_P_59 > |
```

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:**

```

DEEPTHI_P_59> db.restaurants.find({ "grades": { $elemMatch: { "score": { $lt: 5 } } } })
[ {
    _id: ObjectId('664f3c798752f54dc3cdcf7'),
    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'
}
]
DEEPTHI_P_59 > |

```

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:

```

DEEPTHI_P_59> db.restaurants.find({ "grades": { $elemMatch: { "score": { $lt: 5 } } }, "borough": "Manhattan" })
DEEPTHI_P_59 > |

```

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:

```
DEEPTHI_P_59:> db.restaurants.find({ "grades": { $elemMatch: { "score": { $lt: 5 } } }, $or: [{ "borough": "Manhattan" }, { "borough": "Brooklyn" }] })  
DEEPTHI_P_59:> |
```

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:

```
DEEPTHI_P_59:> db.restaurants.find({ "grades": { $elemMatch: { "score": { $lt: 5 } } }, $or: [{ "borough": "Manhattan" }, { "borough": "Brooklyn" }], "cuisine": { $ne: "American" } })  
DEEPTHI_P_59:> |
```

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:

```
| DEEPTHI_P_59> db.restaurants.find({ "grades": { $elemMatch: { "score": { $lt: 5 } } }, $or: [{ "borough": "Manhattan" }, { "borough": "Brooklyn" }], "cuisine": { $nin
$nin: ["American", "Chinese"] } })
| DEEPTHI_P_59> |
```

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:

```
| DEEPTHI_P_59> db.restaurants.find({ $and: [{ "grades.grade": "A", "grades.score": 2 }, { "grades.grade": "A", "grades.score": 6 }] })
[{"_id": ObjectId('664f3c798752f54dc3cdcdf7'),
  address: {
    building: '1007',
    coord: [-73.856077, 40.848447],
    street: 'Morris Park Ave',
    zipcode: '10462'
  },
  borough: 'Bronx',
  cuisine: 'Bakery',
  grades: [
    {
      date: ISODate('2014-03-03T00:00:00.000Z'),
      grade: 'A',
      score: 2
    },
    {
      date: ISODate('2013-09-11T00:00:00.000Z'),
      grade: 'A',
      score: 6
    },
    {
      date: ISODate('2013-01-24T00:00:00.000Z'),
      grade: 'A',
      score: 10
    },
    {
      date: ISODate('2011-11-23T00:00:00.000Z'),
      grade: 'A',
      score: 9
    },
    {
      date: ISODate('2011-03-10T00:00:00.000Z'),
      grade: 'B',
      score: 14
    }
  ],
  name: 'Morris Park Bake Shop',
  restaurant_id: '30075445'
}
| DEEPTHI_P_59> |
```

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:

```
| DEEPTHI_P_59> db.restaurants.find({ $and: [{ "grades.grade": "A", "grades.score": 2 }, { "grades.grade": "A", "grades.score": 6 }], "borough": "Manhattan" })  
| DEEPTHI_P_59> |
```

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:

```
| DEEPTHI_P_59> db.restaurants.find({ $and: [{ "grades.grade": "A", "grades.score": 2 }, { "grades.grade": "A", "grades.score": 6 }], $or: [{ "borough": "Manhattan" }, { "borough": "Brooklyn" }] })  
| DEEPTHI_P_59> |
```

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:

```
+ DEEPTHI_P_59> db.restaurants.find({ $and: [{ "grades.grade": "A", "grades.score": 2 }, { "grades.grade": "A", "grades.score": 6 }], $or: [{ "borough": "Manhattan" }, { "borough": "Brooklyn" }], "cuisine": { $ne: "American" } })
```

```
DEEPTHI_P_59> |
```

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:

```
+ DEEPTHI_P_59> 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"] } })
```

```
DEEPTHI_P_59> |
```

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:

```
[DEEPTHI_P_59] > db.restaurants.find({ $or: [{ "grades.score": 2 }, { "grades.score": 6 }] })  
[{"_id": ObjectId('664f3c798752f54dc3cdcdf7'),  
 "address": {  
   "building": '1007',  
   "coord": [-73.856077, 40.848447],  
   "street": 'Morris Park Ave',  
   "zipcode": '10462'  
 },  
 "borough": 'Bronx',  
 "cuisine": 'Bakery',  
 "grades": [  
   {  
     "date": ISODate('2014-03-03T00:00:00.000Z'),  
     "grade": 'A',  
     "score": 2  
   },  
   {  
     "date": ISODate('2013-09-11T00:00:00.000Z'),  
     "grade": 'A',  
     "score": 6  
   },  
   {  
     "date": ISODate('2013-01-24T00:00:00.000Z'),  
     "grade": 'A',  
     "score": 10  
   },  
   {  
     "date": ISODate('2011-11-23T00:00:00.000Z'),  
     "grade": 'A',  
     "score": 9  
   },  
   {  
     "date": ISODate('2011-03-10T00:00:00.000Z'),  
     "grade": 'B',  
     "score": 14  
   }  
,  
 "name": 'Morris Park Bake Shop',  
 "restaurant_id": '30075445'}]  
[DEEPTHI_P_59] > |
```

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

## RESULT:

# MONGO DB

**EX\_NO:** 20

**DATE:**

**1.) Find all movies with full information from the 'movies' collection that released in the year 1893.**

**QUERY:**

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

**OUTPUT:**

```
DEEPTHI_P_59> db.movies.find({ year: 1893 })
DEEPTHI_P_59 > |
```

**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:**

```
DEEPTHI_P_59 > db.movies.find({ runtime: { $gt: 120 } })  
DEEPTHI_P_59 > |
```

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

#### QUERY:

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

#### OUTPUT:

```
[{"_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/MV5BMTU3NjE5NzYtYTYYNs00MDVmLWIwYjgtMmYwYWIxZDYyNzU2XkEyXkFqcGdeQXVyNzQzNzQxNzI0._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')  
 }  
}  
]
```

### 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:

```
DEEPTHI_P_59> db.movies.find({ directors: 'William K.L. Dickson' })
DEEPTHI_P_59> |
```

**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:**

```
DEEPTHI_P_59> db.movies.find({ countries: 'USA' })
[
  {
    _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/MV5BMTU3NjE5NzYtYTYYNS00MDVmLWIwYjgtMmYwYWIxZDYyNzU2XkEyXkFqcGdeQXVyNzQzNzQzI0._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')
    }
  }
]
DEEPTHI_P_59> |
```

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

**QUERY:**

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

**OUTPUT:**

```
DEEPTHI_P_59 => db.movies.find({ rated: 'UNRATED' })  
DEEPTHI_P_59 => |
```

**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:**

```
DEEPTHI_P_59 => db.movies.find({ 'imdb.votes': { $gt: 1000 } })  
[  
  {  
    _id: ObjectId('573a1390f29313caabcd42e8'),  
    plot: 'A group of bandits stage a brazen train hold-up, only to find a determined posse hot on their heels.',  
    genres: [ 'Short', 'Western' ],  
    runtime: 11,  
    cast: [  
      'A.C. Abadie',  
      "Gilbert M. 'Broncho Billy' Anderson",  
      'George Barnes',  
      'Justus D. Barnes'  
    ],  
    poster: 'https://m.media-amazon.com/images/M/MV5BMTU3NjE5NzYtYTYYNS00MDVmLWIwYjgtMmYwYWIxZDYyNzU2XkEyXkFqcGdeQXVyNzQzNzQxNzI0._V1_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')  
    }  
  }  
]  
DEEPTHI_P_59 => |
```

**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:**

```

| DEEPTHI_P_59 : 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/MV5BMTU3NjE5NzYtYTYYNS00MDVmLWIwYjgtMmYwYIxZDYyNzU2XkEyXkFqcGdeQXVyNzQzNzQxNzI@._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: [ 'DEEPTHI_P_59' ],
|   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')
|   }
| }
| DEEPTHI_P_59 > |

```

**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:**

```

DEEPTHI_P_59 => db.movies.find({ 'tomatoes.viewer.rating': { $gt: 4 } })
DEEPTHI_P_59 `` |
```

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

**QUERY:**

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

**OUTPUT:**

```

DEEPTHI_P_59 > 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/MV5BMTU3NjESNzYtYTYYN500MDVmLWIwYjgtMmYwYIxZDYyNzU2XkEyXkFqcGdeQXVyNzQzNzQxNzI@._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')
  }
}
]
DEEPTHI_P_59 > |

```

**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:

```

DEEPTHI_P_59 > 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 }
... )
DEEPTHI_P_59 > |

```

**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:

```
DEEPTHI_P_59 > 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 }  
... )  
DEEPTHI_P_59 > |
```

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:

```
DEEPTHI_P_59 > db.movies.find(  
...   { released: ISODate("1893-05-09T00:00:00.000Z") },  
...   { title: 1, languages: 1, released: 1, directors: 1, writers: 1, countries: 1 }  
... )  
DEEPTHI_P_59 > |
```

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:

```
| DEEPTHI_P_59 ~> db.movies.find(  
...   { title: /scene/i },  
...   { title: 1, languages: 1, released: 1, directors: 1, writers: 1, countries: 1 }  
... )  
| DEEPTHI_P_59 > |
```

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

**RESULT:**