

# 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	Varchar
Length	7	25

## QUERY:

```
CREATE TABLE DEPT(ID NUMBER(7) NOT NULL, NAME VARCHAR(25));
```

## OUTPUT:

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

```
1 create table department(
2     id int,
3     name varchar(25)
4 );
```

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

Table created.  
0.04 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	Varchar	Varchar	Number
<b>Length</b>	7	25	25	7

### QUERY:

```
CREATE TABLE EMP(ID NUMBER(7) NOT NULL,  
LAST_NAME VARCHAR(25) NOT NULL,  
FIRST_NAME VARCHAR(25), DEPT_ID NUMBER(7));
```

### 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. On the right, there's a user profile for 'Agila shree 15' and a workspace dropdown set to 'WKSP\_AS15'. The main area is titled 'SQL Commands' with a schema dropdown also set to 'WKSP\_AS15'. Below this are language selection ('SQL'), row count ('10'), and command buttons ('Clear Command', 'Find Tables', 'Save', 'Run'). The SQL editor contains the following code:

```
6  create table EMP(id number(7),last_name varchar2(25),first_name varchar2(25),dept_id number(7));
```

The results tab shows the output: 'Table created.' and a execution time of '0.04 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 VARCHAR(50));
```

**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 'A1 Agila shree 15 as15'. The main area is titled 'SQL Commands' with a schema dropdown set to 'WKSP\_AS15'. Below the title are buttons for 'Language' (set to 'SQL'), 'Rows' (set to 10), 'Clear Command', 'Find Tables', 'Save', and 'Run'. The SQL command entered is 'alter table EMP modify(last\_name varchar(50));'. The results section shows the output: 'Table altered.' and a execution time of '0.06 seconds'.

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(6) NOT NULL,  
FIRST_NAME VARCHAR(20), LAST_NAME VARCHAR(25) NOT NULL,  
SALARY NUMBER(8,2), DEPT_ID NUMBER(6) NOT NULL);
```

**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 'A1 Agila shree 15 as15'. The main area is titled 'SQL Commands' with a schema dropdown set to 'WKSP\_AS15'. Below the title are buttons for 'Language' (set to 'SQL'), 'Rows' (set to 10), 'Clear Command', 'Find Tables', 'Save', and 'Run'. The SQL command entered is 'create table employees2(id number(6)not null, first\_name varchar(20),last\_name varchar(25)not null, salary number(8,2),dept\_id number(6)not null);'. The results section shows the output: 'Table created.' and a execution time of '0.06 seconds'.

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', 'Team Development', 'Gallery', a search bar, and a user profile 'A1 Agila shree 15 as15'. The main area is titled 'SQL Commands' with a schema dropdown set to 'WKSP\_AS15'. Below the title, there are buttons for 'Language' (set to 'SQL'), 'Rows' (set to 10), 'Clear Command', 'Find Tables', 'Save', and 'Run'. The SQL command entered is '14 drop table EMP;'. The results tab is selected, showing the output 'Table dropped.' and a execution time of '0.08 seconds'.

6. Rename the EMPLOYEES2 table as EMP.

### QUERY:

RENAME EMPLOYEES2 TO EMP;

### OUTPUT:

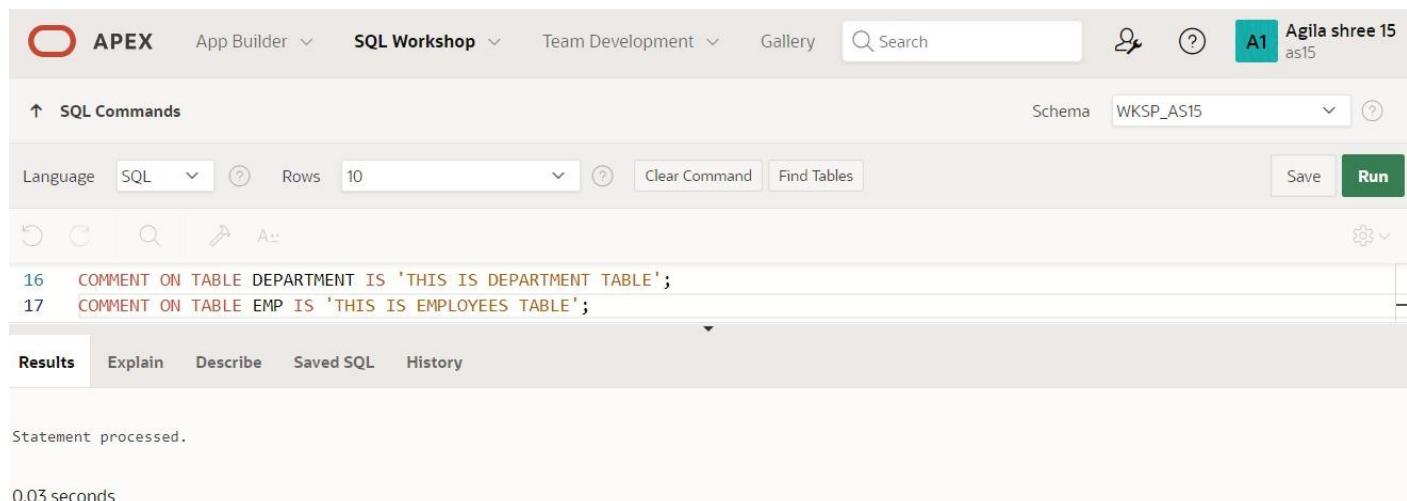
The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', 'Gallery', a search bar, and a user profile 'A1 Agila shree 15 as15'. The main area is titled 'SQL Commands' with a schema dropdown set to 'WKSP\_AS15'. Below the title, there are buttons for 'Language' (set to 'SQL'), 'Rows' (set to 10), 'Clear Command', 'Find Tables', 'Save', and 'Run'. The SQL command entered is '21 rename employees2 to EMP;'. The results tab is selected, showing the output 'Statement processed.' and a execution time of '0.06 seconds'.

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

### QUERY:

```
COMMENT ON TABLE DEPT IS 'THIS IS DEPARTMENT TABLE';
COMMENT ON TABLE EMP IS 'THIS IS EMPLOYEES TABLE';
```

### 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 'A1 Agila shree 15 as15'. The main area is titled 'SQL Commands' with tabs for 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History'. The results pane displays the following SQL commands:

```
16 COMMENT ON TABLE DEPARTMENT IS 'THIS IS DEPARTMENT TABLE';
17 COMMENT ON TABLE EMP IS 'THIS IS EMPLOYEES TABLE';
```

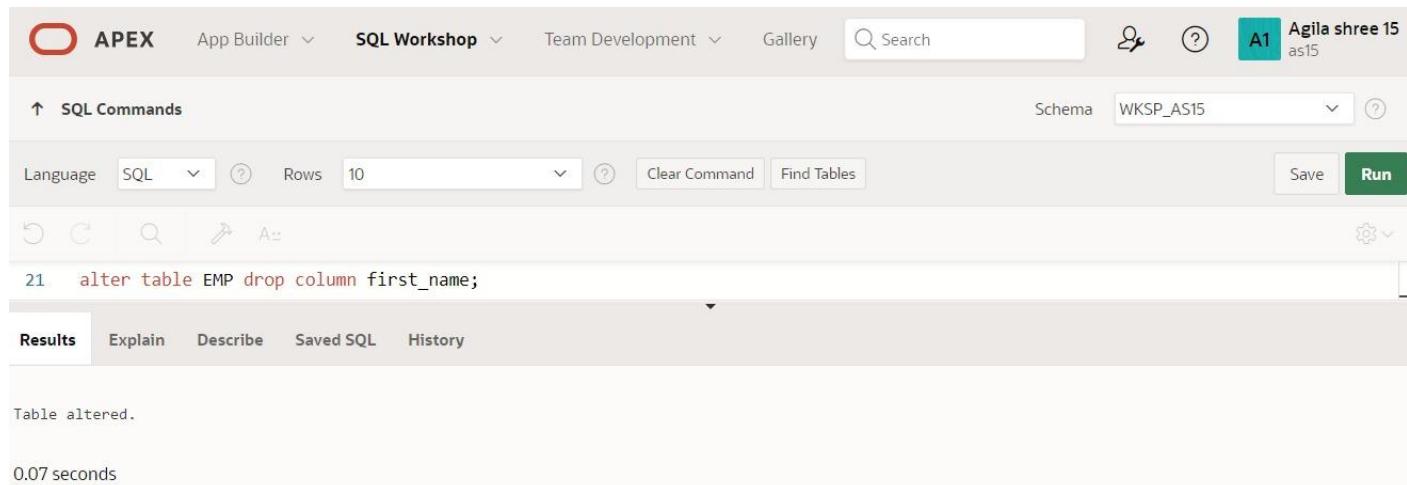
Below the results, a message states 'Statement processed.' and '0.03 seconds'.

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', 'Team Development', 'Gallery', a search bar, and a user profile 'A1 Agila shree 15 as15'. The main area is titled 'SQL Commands' with tabs for 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History'. The results pane displays the following SQL command:

```
21 alter table EMP drop column first_name;
```

Below the results, a message states 'Table altered.' and '0.07 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_EMPLOYEE(ID NUMBER(4) NOT NULL,  
LAST_NAME VARCHAR(25),FIRST_NAME VARCHAR(25),  
USERID VARCHAR(25), SALARY NUMBER(9,2));
```

## OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, Gallery, a search bar, and a user profile A1 Agila shree 15. The SQL Workshop tab has 'SQL Commands' selected. The schema dropdown is set to WKSP\_AS15. The main area contains the SQL command for creating the table:

```
1 create table my_employee(id number(4)not null,last_name varchar(25),  
2 first_name varchar(25),user_id varchar(25),salary number(9,2));
```

The results section at the bottom displays the message "Table created." and "0.05 seconds".

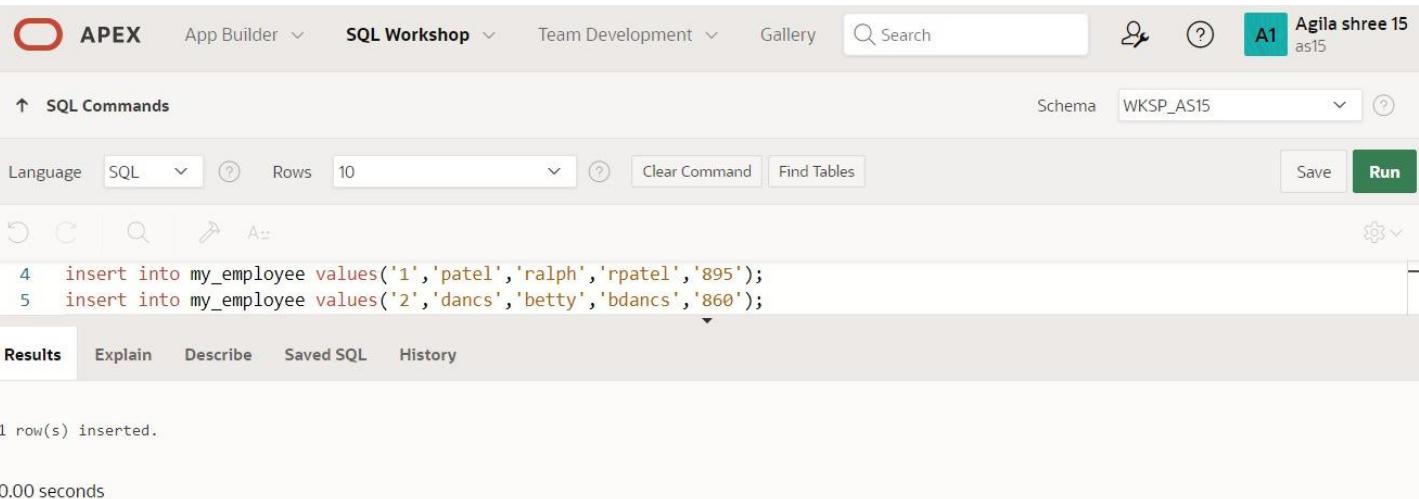
2. Add the first and second rows data to MY\_EMPLOYEE table from the following sample data.

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_EMPLOYEE VALUES(1,'PATEL','RALPH','RPATEL',895);
INSERT INTO MY_EMPLOYEE VALUES(2,'DANCS','BETTY','BDANCS',860);
```

## 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 Agila shree 15 as15. The main area is titled "SQL Commands". The schema dropdown is set to WKSP\_AS15. The command input field contains the following SQL code:

```
4 insert into my_employee values('1','patel','ralph','rpatel','895');
5 insert into my_employee values('2','dancs','betty','bdancs','860');
```

The "Results" tab is selected, showing the output: "1 row(s) inserted." Below it, the time taken is listed as "0.00 seconds".

3. Display the table with values.

### QUERY:

```
SELECT * FROM MY_EMPLOYEE;
```

### 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 A1 Agila shree 15 as15. The SQL Workshop tab is selected. The schema dropdown is set to WKSP\_AS15. The main area shows the following SQL command:

```
7 select*from my_employee;
```

Below the command, the Results tab is selected, displaying the following table:

ID	LAST_NAME	FIRST_NAME	USER_ID	SALARY
1	patel	ralph	rpatel	895
2	dancs	betty	bdancs	860

At the bottom left, it says "2 rows returned in 0.03 seconds".

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

### QUERY:

```
INSERT INTO MY_EMPLOYEE VALUES(3,'BIRI','BEN','BBIRI',1100);
INSERT INTO MY_EMPLOYEE VALUES(4,'NEWMAN','CHAD','CNEWMAN',750);
INSERT INTO MY_EMPLOYEE VALUES(5,'ROPEBUR','AUDREY','AROPEBUR',1550);
```

### 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 A1 Agila shree 15 as15. The SQL Workshop tab is selected. The schema dropdown is set to WKSP\_AS15. The main area shows the following SQL commands:

```
10 insert into my_employee values('3','biri','ben','bbiri','1100');
11 insert into my_employee values('4','newmann','chad','cnewmann','750');
12 insert into my_employee values('5','ropebur','audrey','aropebur','1550');
```

Below the commands, the Results tab is selected, displaying the message "1 row(s) inserted." and "0.00 seconds".

5. Make the data additions permanent.

### QUERY:

```
SELECT * FROM MY_EMPLOYEE;
```

### 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 'A1 Agila shree 15 as15'. The main area is titled 'SQL Commands' with a schema dropdown set to 'WKSP\_AS15'. The query entered is 'select\*from my\_employee;'. The results tab displays the following data:

ID	LAST_NAME	FIRST_NAME	USER_ID	SALARY
1	patel	ralph	rpatel	895
4	newmann	chad	cnewmann	750
5	ropebur	audrey	aropebur	1550
2	dancs	betty	bdancs	860
3	biri	ben	bbiri	1100

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

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

### QUERY:

```
UPDATE MY_EMPLOYEE SET LAST_NAME='DREXLER' WHERE ID=3;
```

### OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', 'Gallery', a search bar, and a user profile 'A1 Agila shree 15 as15'. The main area is titled 'SQL Commands' with a schema dropdown set to 'WKSP\_AS15'. The query entered is 'update my\_employee set last\_name='Drelex' where id=3;'. The results tab displays the message '1 row(s) updated.' and a time of '0.01 seconds'.

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

### QUERY:

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

### OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', 'Gallery', a search bar, and a user profile 'A1 Agila shree 15 as15'. The main area is titled 'SQL Commands' with a schema dropdown set to 'WKSP\_AS15'. Below the title are buttons for 'Language' (set to 'SQL'), 'Rows' (set to 10), 'Clear Command', 'Find Tables', 'Save', and a large green 'Run' button. The command history shows line 18: 'update my\_employee set salary='1000' where salary<900;'. The results tab is selected, showing the message '3 row(s) updated.' and a execution time of '0.01 seconds'.

8. Delete Betty Dancs from MY\_EMPLOYEE table.

### QUERY:

```
DELETE FROM MY_EMPLOYEE WHERE FIRST_NAME='BETTY' AND LAST_NAME='DANCS';
```

### OUTPUT:

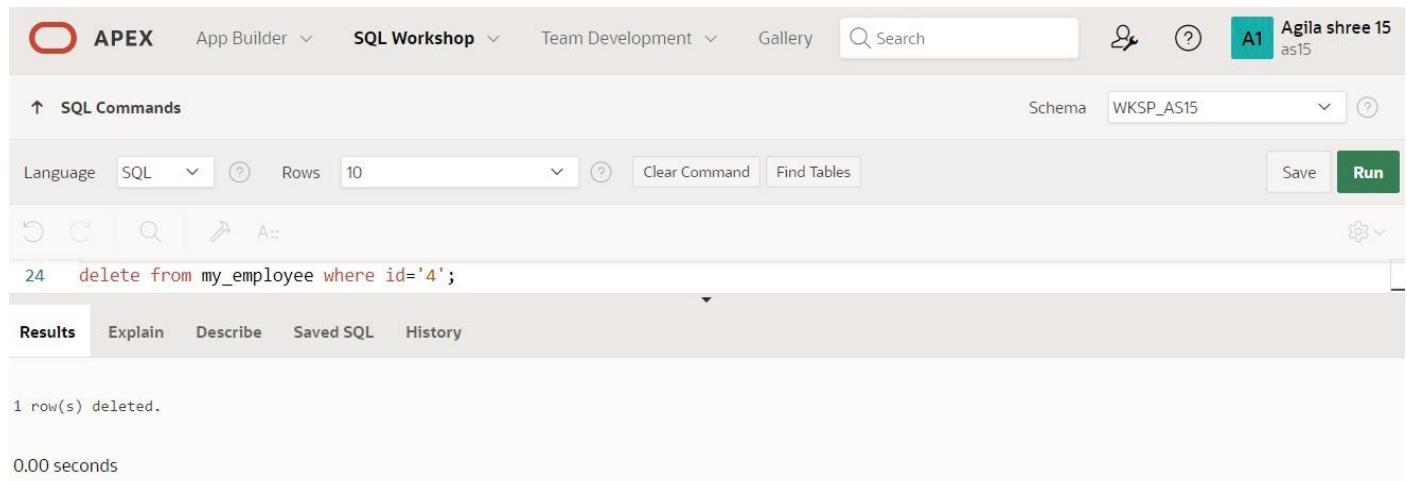
The screenshot shows the Oracle SQL Workshop interface, identical to the previous one but with a different command. The command history shows line 21: 'delete from my\_employee where first\_name='betty';'. The results tab is selected, showing the message '1 row(s) deleted.' and a execution time of '0.01 seconds'.

9. Empty the fourth row of the emp table.

### QUERY:

DELETE FROM MY\_EMPLOYEE WHERE ID='4';

### 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 'Agila shree 15' and 'A1 as15'. The main area is titled 'SQL Commands'. It has dropdowns for 'Language' (set to 'SQL'), 'Rows' (set to '10'), and buttons for 'Clear Command', 'Find Tables', 'Save', and 'Run'. Below these are icons for undo, redo, search, and refresh. The SQL command entered is '24 delete from my\_employee where id='4';'. The results tab is selected, showing the output: '1 row(s) deleted.' and '0.00 seconds' execution time.

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

### RESULT :

# INCLUDING CONSTRAINTS

EX-NO : 3

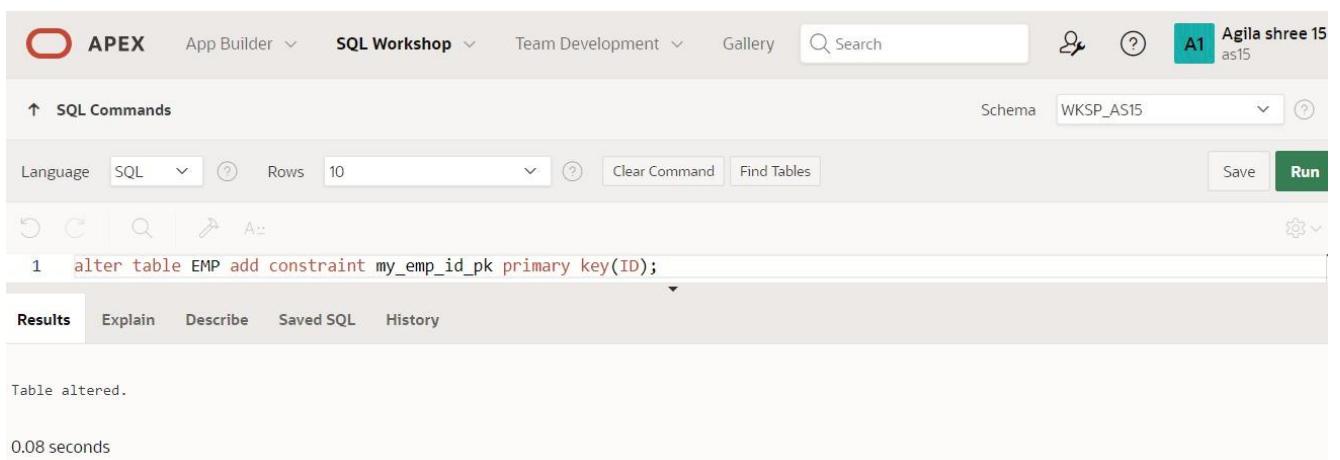
DATE:

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:

```
ALTER TABLE EMP ADD CONSTRAINT my_emp_id_pk PRIMARY KEY(ID);
```

## OUTPUT:



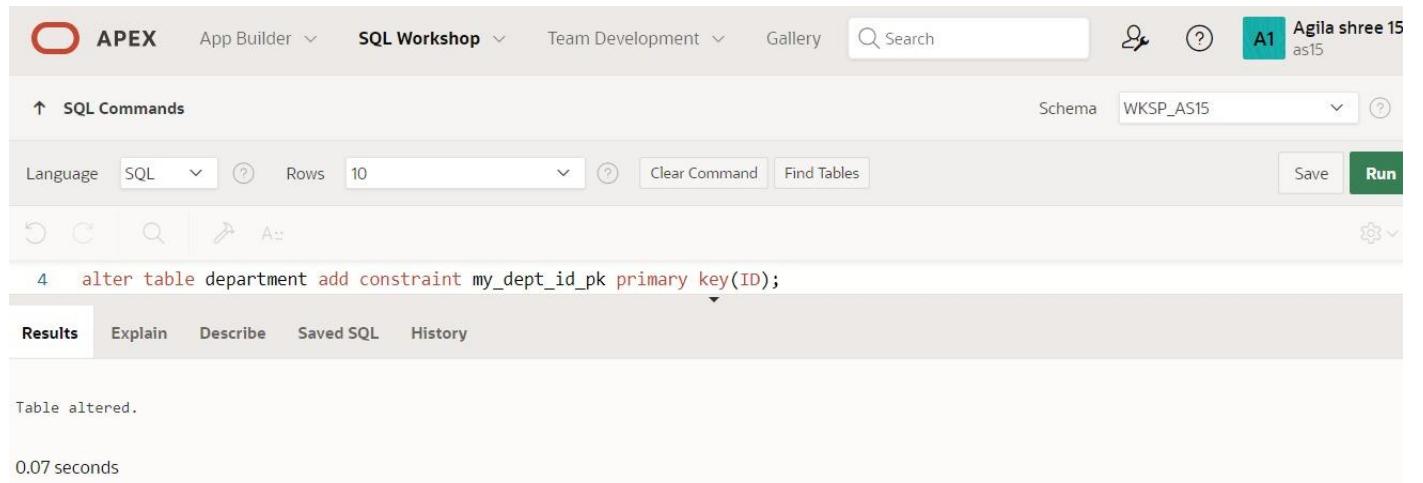
The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (which is selected), Team Development, Gallery, and a search bar. On the right, there's a user profile for 'Agila shree 15' and a workspace dropdown set to 'WKSP\_AS15'. The main workspace is titled 'SQL Commands'. It features a toolbar with icons for Undo, Redo, Find, and Run. Below the toolbar, the schema is set to 'WKSP\_AS15'. The SQL command entered is: `alter table EMP add constraint my_emp_id_pk primary key(ID);`. The 'Results' tab is active, displaying the output: 'Table altered.' and '0.08 seconds'. Other tabs available include Explain, Describe, Saved SQL, and History.

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:

```
ALTER TABLE DEPARTMENT ADD CONSTRAINT my_dept_id_pk PRIMARY  
KEY(ID);
```

### OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, Gallery, and a search bar. On the right, there's a user icon for 'Agila shree 15' and a workspace identifier 'A1 as15'. The main workspace is titled 'SQL Commands'. It features a toolbar with icons for Undo, Redo, Find, Replace, and Run. Below the toolbar, the schema is set to 'WKSP\_AS15'. The SQL editor contains the following command:

```
4 alter table department add constraint my_dept_id_pk primary key(ID);
```

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

```
Table altered.  
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 EMP ADD CONSTRAINT MY_EMP_DEPT_ID_FK FOREIGN  
KEY(DEPT_ID) REFERENCES DEPARTMENT(ID);
```

### OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery, along with a search bar and user information (A1 Agila shree 15 as15). The main workspace is titled "SQL Commands" and contains the following SQL code:

```
6  ALTER TABLE EMP ADD CONSTRAINT MY_EMP_DEPT_ID_FK FOREIGN KEY(DEPT_ID) REFERENCES DEPARTMENT(ID);  
7
```

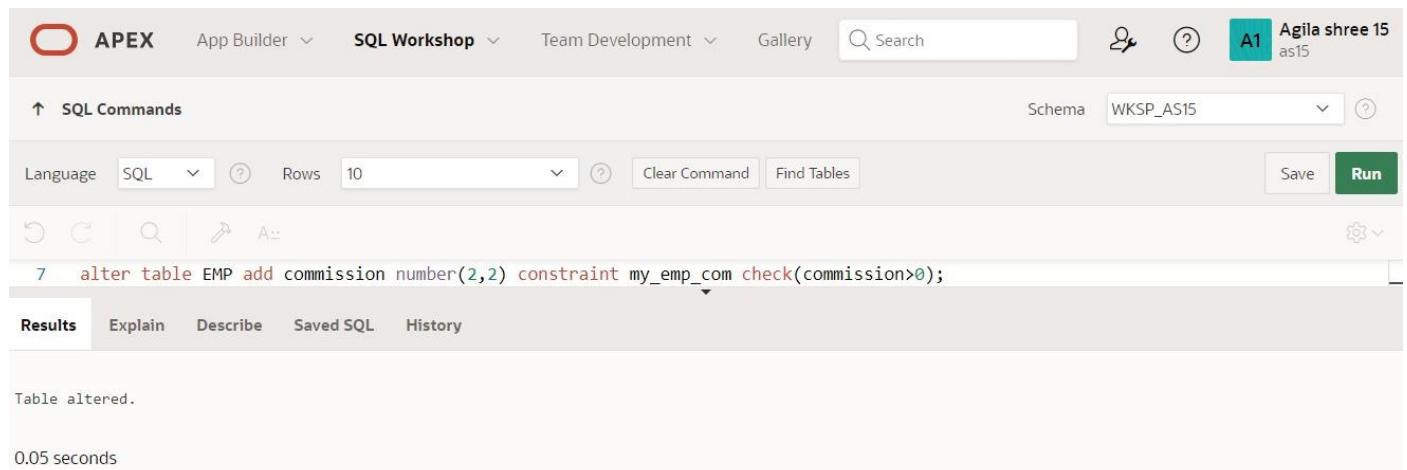
The "Results" tab is selected, displaying the output "Table altered." Below the results, a note indicates "0.07 seconds".

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 EMP ADD COMMISSION NUMBER(2,2) CONSTRAINT  
my_emp_com CHECK(COMMISSION>0);
```

### OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, Gallery, and a search bar. On the right, there are user profile icons for 'A1 Agila shree 15' and 'as15'. The main workspace is titled 'SQL Commands' and shows the schema 'WKSP\_AS15'. The SQL editor contains the following command:

```
7 alter table EMP add commission number(2,2) constraint my_emp_com check(commission>0);
```

The 'Results' tab is selected, displaying the output: 'Table altered.' Below it, the execution time is shown as '0.05 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.

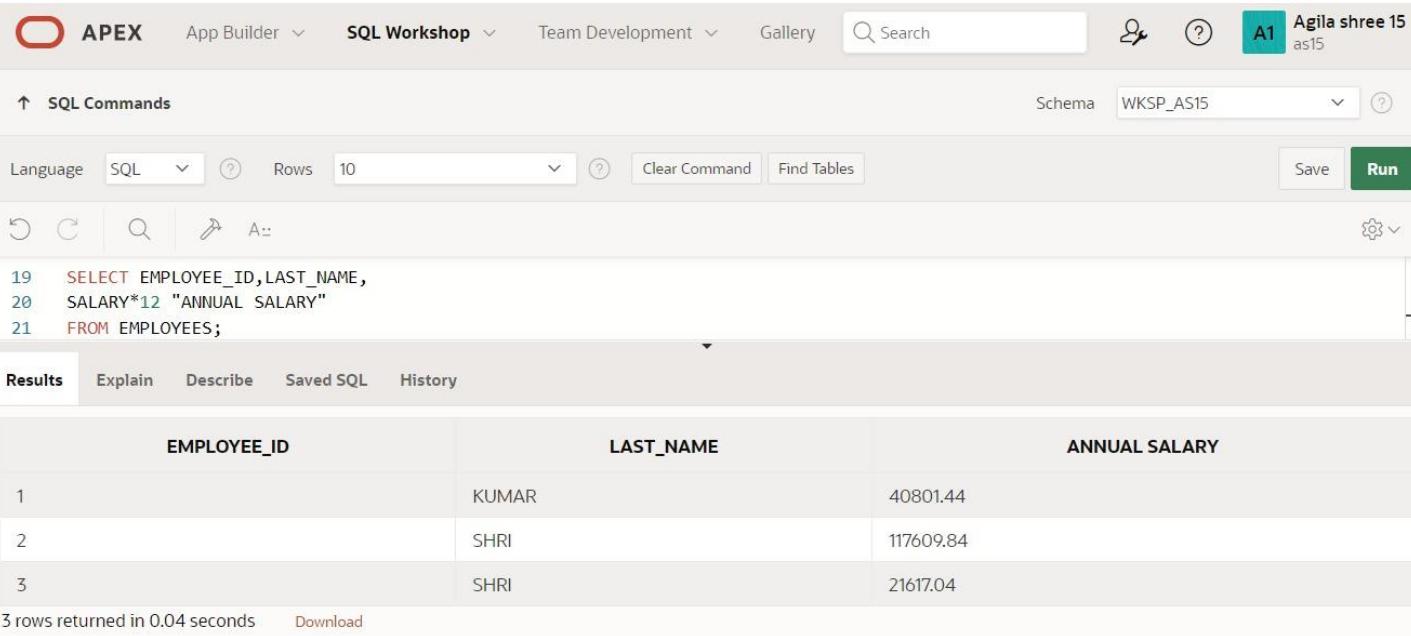
## Identify the Errors

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

## QUERY:

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

## 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 connected to schema WKSP\_AS15, indicated by a green button labeled 'A1 Agila shree 15 as15'.

In the SQL Commands section, the query is displayed:

```
19: SELECT EMPLOYEE_ID, LAST_NAME,  
20: SALARY*12 "ANNUAL SALARY"  
21: FROM EMPLOYEES;
```

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

EMPLOYEE_ID	LAST_NAME	ANNUAL SALARY
1	KUMAR	40801.44
2	SHRI	117609.84
3	SHRI	21617.04

At the bottom, it says '3 rows returned in 0.04 seconds' and has a 'Download' link.

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

### QUERY:

DESC DEPARTMENT;

### OUTPUT:

The screenshot shows the Oracle 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 information (A1 Agila shree 15 as15). The main area is titled "SQL Commands" and contains the command "27 DESC DEPARTMENTS;". Below this, there are tabs for Results, Explain, Describe (which is selected), Saved SQL, and History. The "Describe" tab displays the structure of the DEPARTMENTS table. The table has four columns: DEPT\_ID, DEPT\_NAME, MANAGER\_ID, and LOCATION\_ID. The DEPT\_ID column is defined as NUMBER(6,0) and is the primary key. The DEPT\_NAME column is defined as VARCHAR2(20) and is nullable. The MANAGER\_ID and LOCATION\_ID columns are also defined as NUMBER(6,0) and are nullable. There is a "Run" button at the bottom right of the SQL input area.

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
DEPARTMENTS	DEPT_ID	NUMBER	-	6	0	-	-	-	-
	DEPT_NAME	VARCHAR2	20	-	-	-	✓	-	-
	MANAGER_ID	NUMBER	-	6	0	-	✓	-	-
	LOCATION_ID	NUMBER	-	4	0	-	✓	-	-

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

### QUERY:

```
SELECT EMPLOYEE_ID, LAST_NAME, JOB_ID, HIRE_DATE  
FROM EMPLOYEES;
```

### 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 'Agila shree 15 as15'. The main area is titled 'SQL Commands' with a schema dropdown set to 'WKSP\_AS15'. The SQL editor contains the following code:

```
27  SELECT EMPLOYEE_ID, LAST_NAME, JOB_ID, HIRE_DATE  
28  FROM EMPLOYEES;
```

The 'Results' tab is selected, displaying the query results in a grid format:

EMPLOYEE_ID	LAST_NAME	JOB_ID	HIRE_DATE
1	KUMAR	456	1999-05-01
2	SHRI	67	1999-08-02
3	SHRI	67	1996-07-01

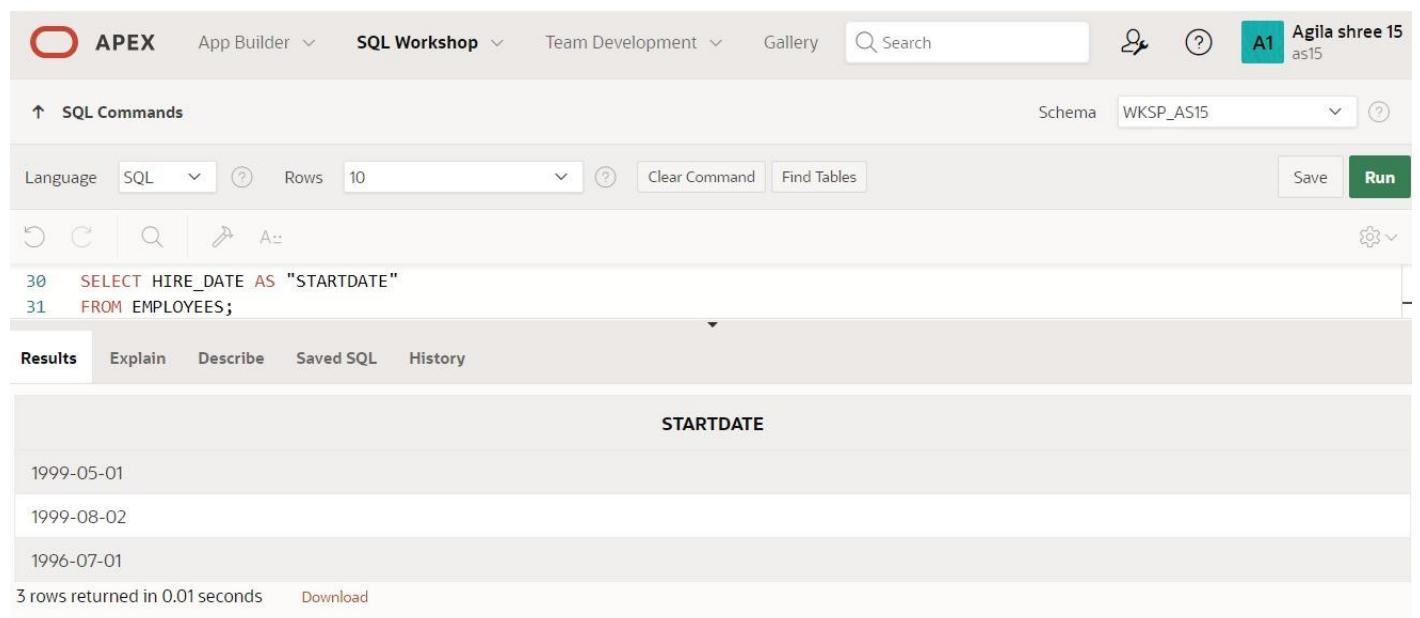
Below the table, it says '3 rows returned in 0.01 seconds' and there is a 'Download' link.

4. Provide an alias STARTDATE for the hire date.

**QUERY:**

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

**OUTPUT:**



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery, along with a search bar and user profile A1 Agila shree 15 as15. The main workspace is titled 'SQL Commands' and shows the following SQL code:

```
30  SELECT HIRE_DATE AS "STARTDATE"  
31  FROM EMPLOYEES;
```

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

STARTDATE
1999-05-01
1999-08-02
1996-07-01

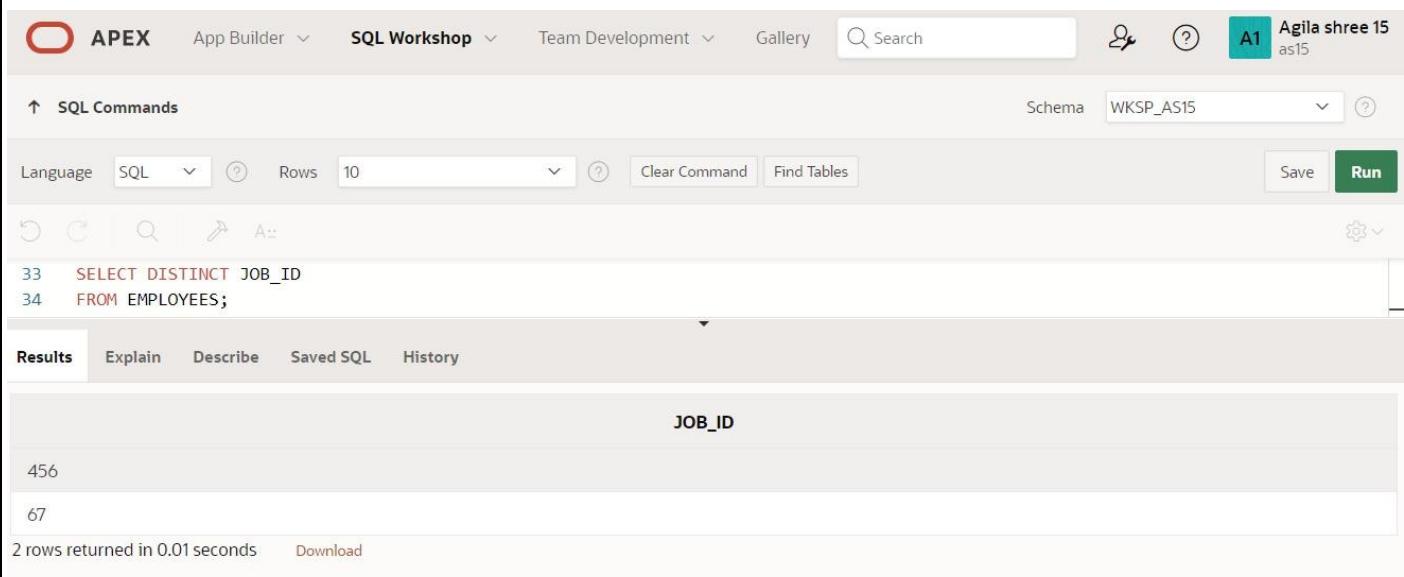
Below the results, it says '3 rows returned in 0.01 seconds' and has a 'Download' link.

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. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, Gallery, and a search bar. On the right, there's a user profile for 'Agila shree 15' and a workspace identifier 'A1 as15'. The main workspace is titled 'SQL Commands'. It features a toolbar with icons for Undo, Redo, Find, Replace, and Save/Run. Below the toolbar, the schema is set to 'WKSP\_AS15'. The SQL editor contains the following code:

```
33  SELECT DISTINCT JOB_ID  
34  FROM EMPLOYEES;
```

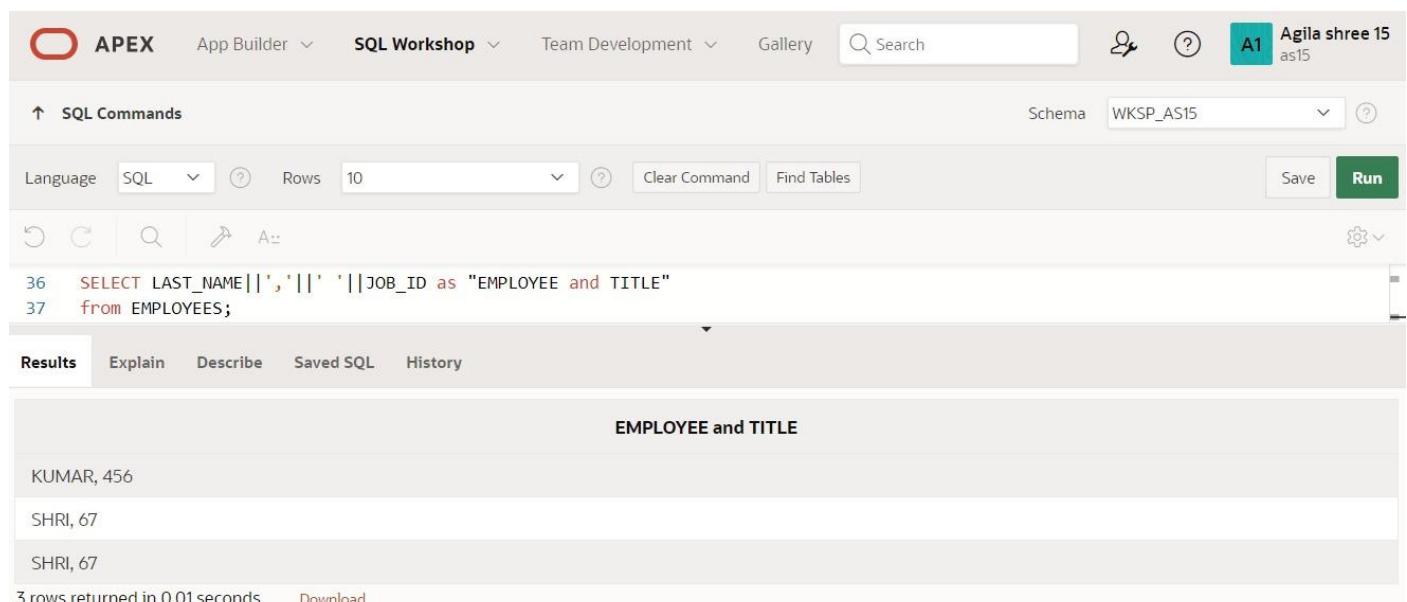
The results tab is selected, showing a single column named 'JOB\_ID' with two rows of data: 456 and 67. The status bar at the bottom indicates '2 rows returned in 0.01 seconds' and provides a 'Download' link.

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. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, Gallery, and a search bar. On the right, there's a user profile for 'Agila shree 15' and a workspace identifier 'A1 WKSP\_AS15'. The main workspace is titled 'SQL Commands'. It features a toolbar with buttons for Language (SQL selected), Rows (set to 10), Clear Command, Find Tables, Save, and Run. Below the toolbar, the SQL command is displayed:

```
36  SELECT LAST_NAME||','||''||JOB_ID as "EMPLOYEE and TITLE"  
37  from EMPLOYEES;
```

The results tab is active, showing the output of the query:

EMPLOYEE and TITLE	
KUMAR,	456
SHRI,	67
SHRI,	67

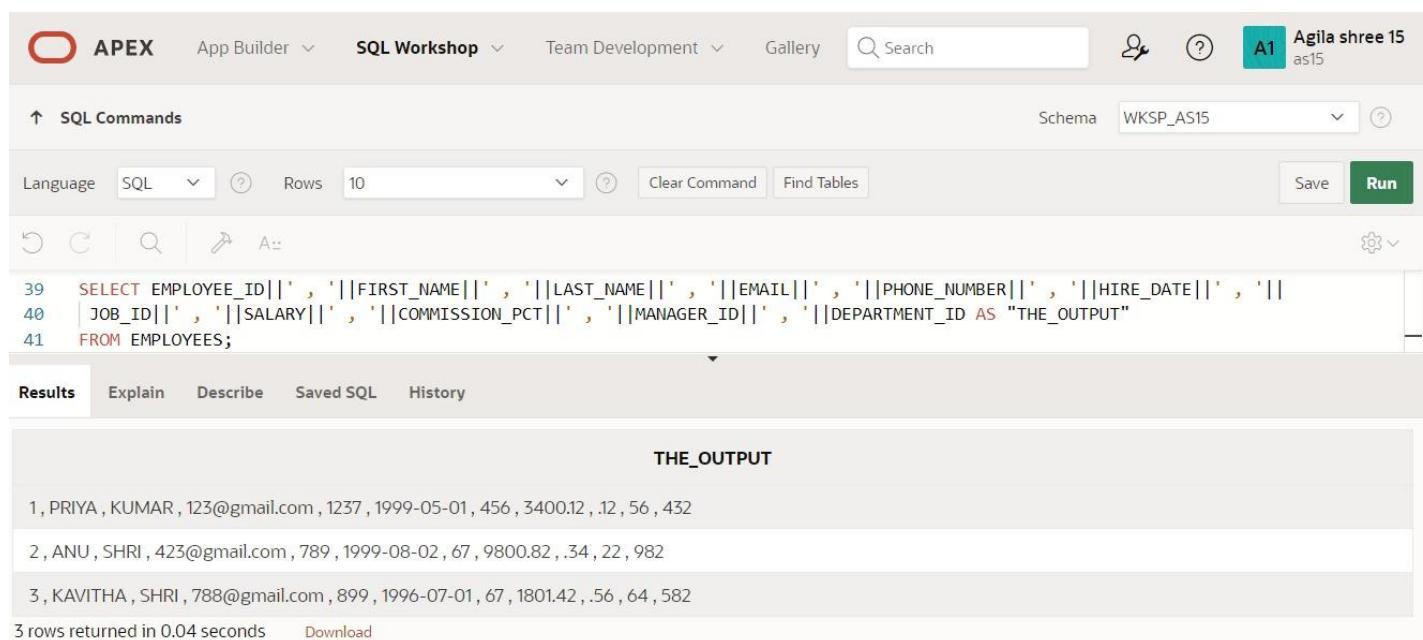
At the bottom left, it says '3 rows returned in 0.01 seconds'. There is also a 'Download' link.

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 EMPLOYEE_ID", "FIRST_NAME", "LAST_NAME", "EMAIL", "PHONE_NUMBER",
"||HIRE_DATE", "|| JOB_ID", "||SALARY", "||COMMISSION_PCT", "||MANAGER_ID",
"||DEPARTMENT_ID AS "THE_OUTPUT" FROM EMPLOYEES;
```

## 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 (Agila shree 15, A1, as15). The main area is titled 'SQL Commands'. The schema is set to WKSP\_AS15. The SQL editor contains the following code:

```
39  SELECT EMPLOYEE_ID", "FIRST_NAME", "LAST_NAME", "EMAIL", "PHONE_NUMBER",
40  "||HIRE_DATE", "|| JOB_ID", "||SALARY", "||COMMISSION_PCT", "||MANAGER_ID",
41  "||DEPARTMENT_ID AS "THE_OUTPUT"
FROM EMPLOYEES;
```

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

THE_OUTPUT
1, PRIYA, KUMAR, 123@gmail.com, 1237, 1999-05-01, 456, 3400.12, .12, 56, 432
2, ANU, SHRI, 423@gmail.com, 789, 1999-08-02, 67, 9800.82, .34, 22, 982
3, KAVITHA, SHRI, 788@gmail.com, 899, 1996-07-01, 67, 1801.42, .56, 64, 582

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

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

**RESULT :**

## RESTRICTING AND SORTING DATA

EX NO:

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;**

**OUTPUT:**

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

```
8  SELECT LAST_NAME,SALARY
9  FROM EMPLOYEES
10 WHERE SALARY>12000;
```

The results table displays two rows:

LAST_NAME	SALARY
JANE	23400.12
JAY	13400.12

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

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

**QUERY:**

**Select last\_name, dept\_id from Employee where e\_id=176;**

**OUTPUT:**

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

```
13  SELECT LAST_NAME,DEPARTMENT_ID
14  FROM EMPLOYEES
15  WHERE EMPLOYEE_ID='176';
```

The results table displays one row:

LAST_NAME	DEPARTMENT_ID
EMANUEL	30

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

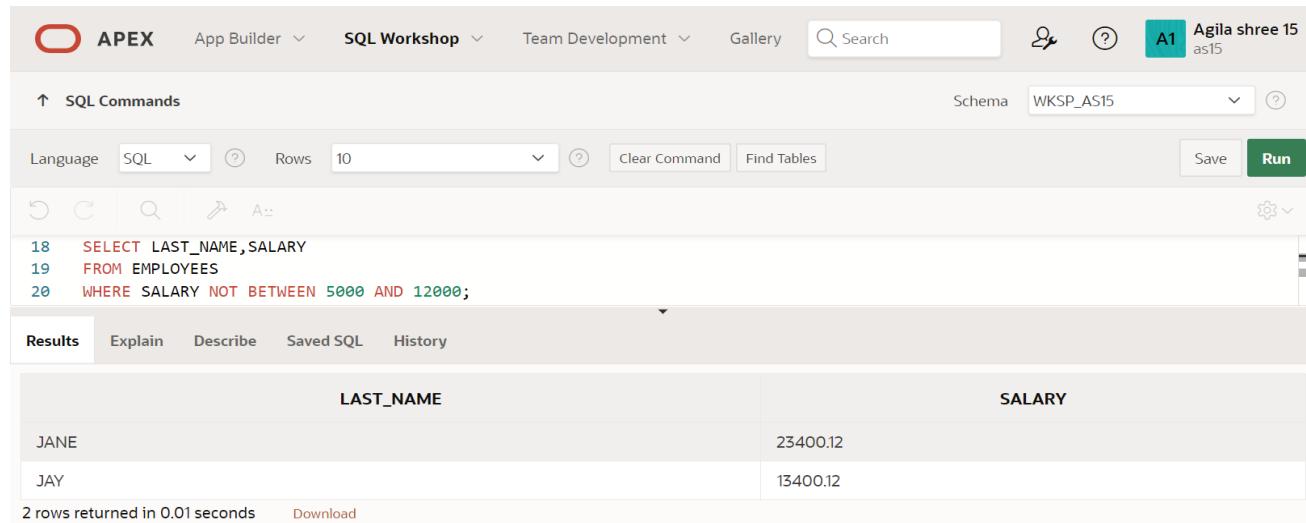
3. Create a query to display the last name and salary of employees whose salary is not in the range of 5000 and 12000. (hints: not between )

**QUERY:**

**SELECT last\_name , dept\_id from Employee**

**Where Salary NOT BETWEEN 5000 AND 12000**

**OUTPUT:**



APEX App Builder SQL Workshop Team Development Gallery Search Schema WKSP\_AS15 A1 Agila shree 15 as15

↑ SQL Commands Schema: WKSP\_AS15

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

LAST\_NAME SALARY

LAST_NAME	SALARY
JANE	23400.12
JAY	13400.12

2 rows returned in 0.01 seconds Download

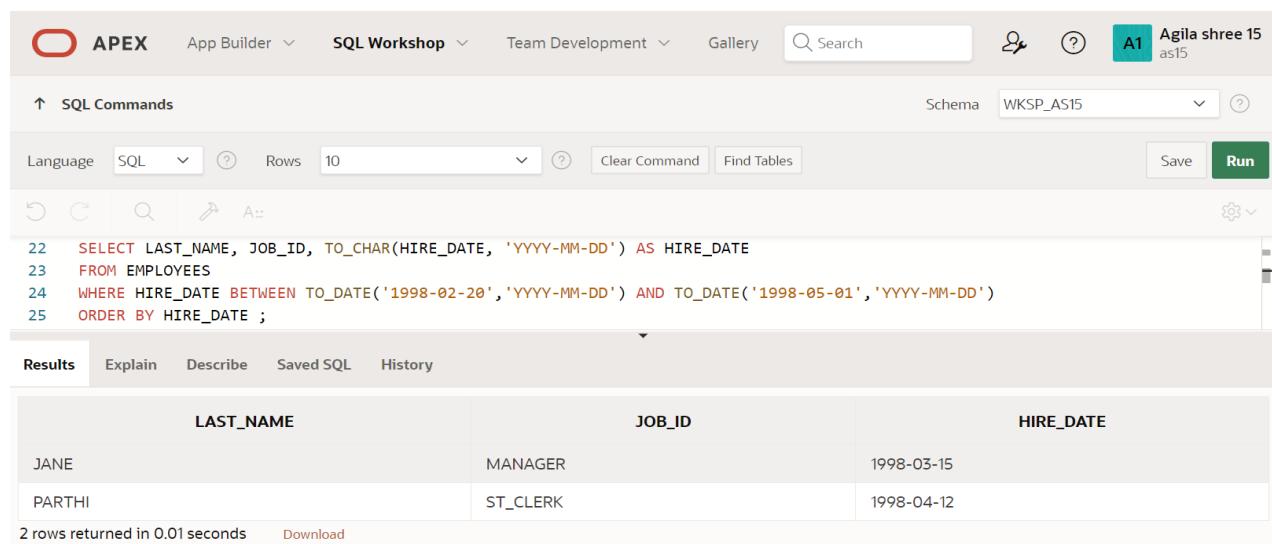
4. Display the employee last name, job ID, and start date of employees hired between February 20,1998 and May 1,1998.order the query in ascending order by start date.(hints: between)

**QUERY:**

**Select last\_name, Job\_id, hire\_date from Employee**

**where hire\_date between 'February 20,1998' AND 'May 1,1998';**

**OUTPUT:**



APEX App Builder SQL Workshop Team Development Gallery Search Schema WKSP\_AS15 A1 Agila shree 15 as15

↑ SQL Commands Schema: WKSP\_AS15

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

LAST\_NAME JOB\_ID HIRE\_DATE

LAST_NAME	JOB_ID	HIRE_DATE
JANE	MANAGER	1998-03-15
PARTHI	ST_CLERK	1998-04-12

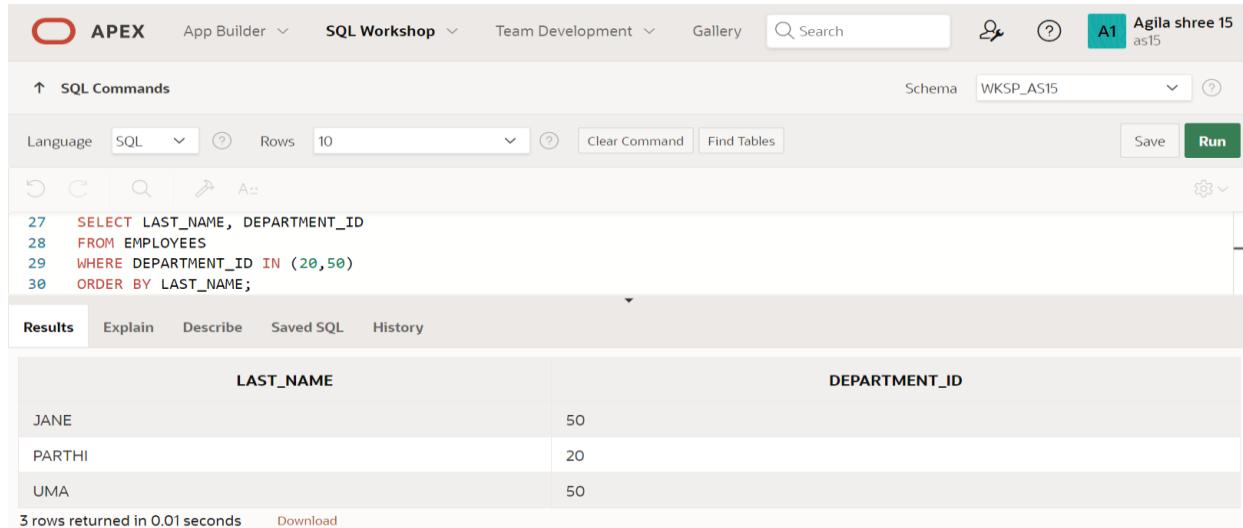
2 rows returned in 0.01 seconds Download

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,Salary from Employee where (Salary BETWEEN 5000 AND 12000) AND (dept\_id IN(20,50)) order by last\_name AS EMPLOYEE, MONTHLY SALARY;**

**OUTPUT:**



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

```
27  SELECT LAST_NAME, DEPARTMENT_ID
28  FROM EMPLOYEES
29  WHERE DEPARTMENT_ID IN (20,50)
30  ORDER BY LAST_NAME;
```

The results table has two columns: LAST\_NAME and DEPARTMENT\_ID. The data returned is:

LAST_NAME	DEPARTMENT_ID
JANE	50
PARTHI	20
UMA	50

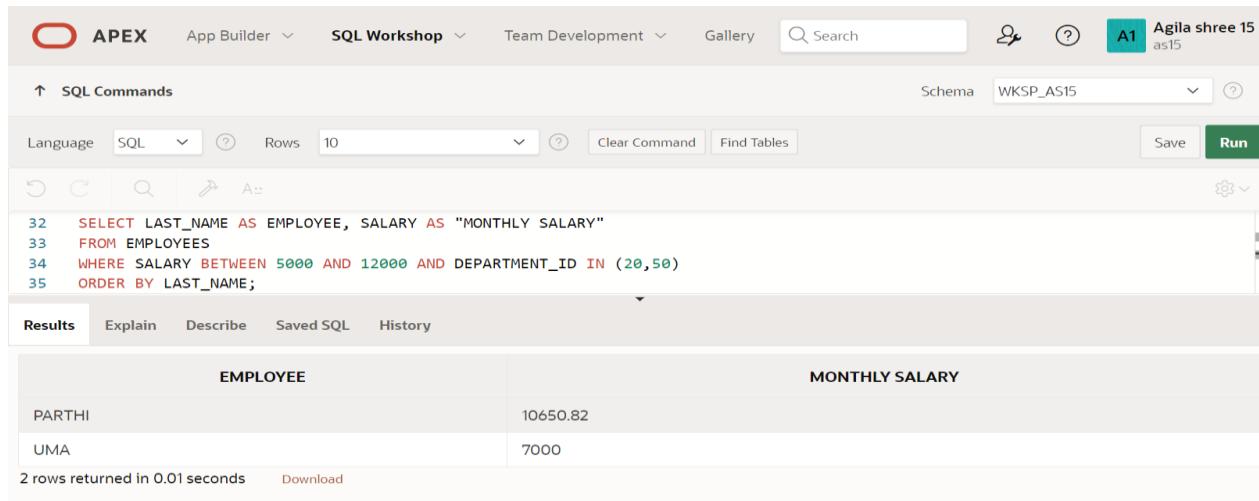
3 rows returned in 0.01 seconds [Download](#)

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 query executed is:

```
32  SELECT LAST_NAME AS EMPLOYEE, SALARY AS "MONTHLY SALARY"
33  FROM EMPLOYEES
34  WHERE SALARY BETWEEN 5000 AND 12000 AND DEPARTMENT_ID IN (20,50)
35  ORDER BY LAST_NAME;
```

The results table has two columns: EMPLOYEE and MONTHLY SALARY. The data returned is:

EMPLOYEE	MONTHLY SALARY
PARTHI	10650.82
UMA	7000

2 rows returned in 0.01 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. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', 'Gallery', a search bar, and a user profile 'A1 Agila shree 15 as15'. The main area is titled 'SQL Commands' with a schema dropdown set to 'WKSP\_AS15'. The code editor contains the following SQL query:

```
37  SELECT LAST_NAME, TO_CHAR(HIRE_DATE, 'YYYY-MM-DD') AS HIRE_DATE
38  FROM EMPLOYEES
39  WHERE HIRE_DATE LIKE '%1994';
```

The results section shows a single row of data:

LAST_NAME	HIRE_DATE
EMANUEL	1994-05-01

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

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 'A1 Agila shree 15 as15'. The main area is titled 'SQL Commands' with a schema dropdown set to 'WKSP\_AS15'. The code editor contains the following SQL query:

```
42  SELECT LAST_NAME, JOB_ID
43  FROM EMPLOYEES
44  WHERE MANAGER_ID IS NULL;
```

The results section shows two rows of data:

LAST_NAME	JOB_ID
EMANUEL	A_MANAGER
JANE	MANAGER

Below the results, it says '2 rows returned in 0.01 seconds' and has 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 null, order by)

**QUERY:**

**Select last\_name ,Salary,commission from employee where commission is not null  
order by Salary,commission DESC;**

**OUTPUT:**

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

```
46  SELECT LAST_NAME, SALARY, COMMISSION_PCT
47  FROM EMPLOYEES
48  WHERE COMMISSION_PCT IS NOT NULL
49  ORDER BY SALARY DESC, COMMISSION_PCT DESC;
```

The results table displays the following data:

LAST_NAME	SALARY	COMMISSION_PCT
JANE	23400.12	.28
JAY	13400.12	.1
EMANUEL	12000	.2

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

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 query entered is:

```
51  SELECT LAST_NAME
52  FROM EMPLOYEES
53  WHERE LAST_NAME LIKE '__A%';
```

The results table displays the following data:

LAST_NAME
UMA
EMANUEL

2 rows returned in 0.06 seconds [Download](#)

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 query entered is:

```
55  SELECT LAST_NAME
56  FROM EMPLOYEES
57  WHERE LAST_NAME LIKE '%A%' AND LAST_NAME LIKE '%E%';
```

The results table displays two rows:

LAST_NAME
EMANUEL
JANE

2 rows returned in 0.00 seconds

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 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 query entered is:

```
59  SELECT LAST_NAME,JOB_ID,SALARY
60  FROM EMPLOYEES
61  WHERE JOB_ID='SA_REP' OR JOB_ID='ST_CLERK'
62  AND salary NOT IN (2500, 3500, 7000);
```

The results table displays one row:

LAST_NAME	JOB_ID	SALARY
PARTHI	ST_CLERK	10650.82

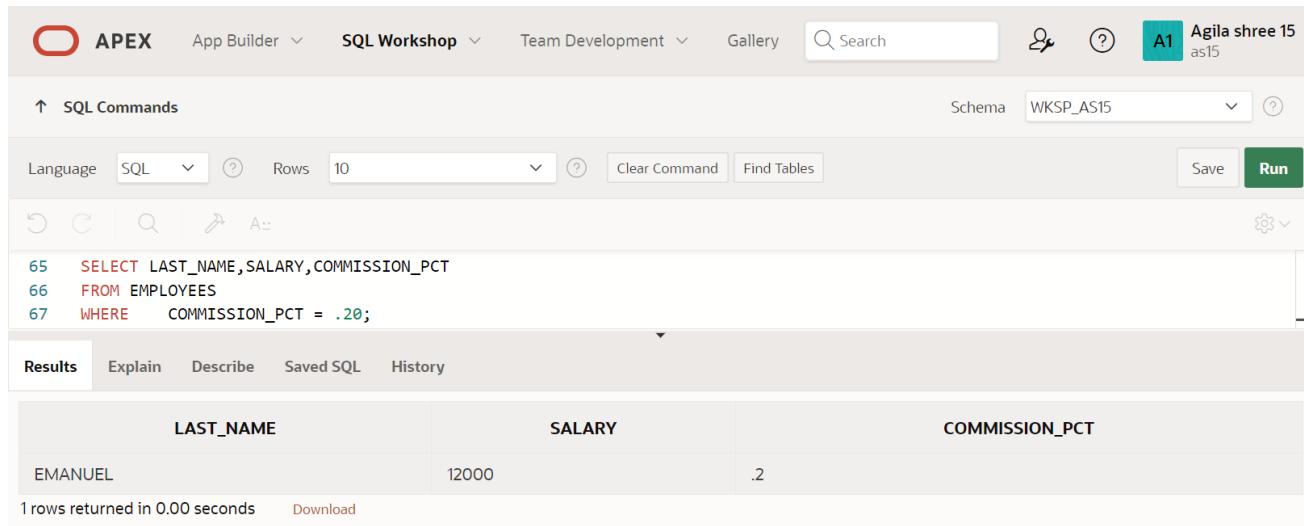
1 rows returned in 0.01 seconds

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

**QUERY:**

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

**OUTPUT:**



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, Gallery, a search bar, and user information (A1 Agila shree 15 as15). The main area is titled "SQL Commands". The language is set to SQL, rows are set to 10, and the schema is WKSP\_AS15. The command entered is:

```
65  SELECT LAST_NAME,SALARY,COMMISSION_PCT
66  FROM EMPLOYEES
67  WHERE COMMISSION_PCT = .20;
```

The results tab is selected, showing the output:

LAST_NAME	SALARY	COMMISSION_PCT
EMANUEL	12000	.2

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

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

## **RESULT:**

# SINGLE ROW FUNCTIONS

EX-NO : 6

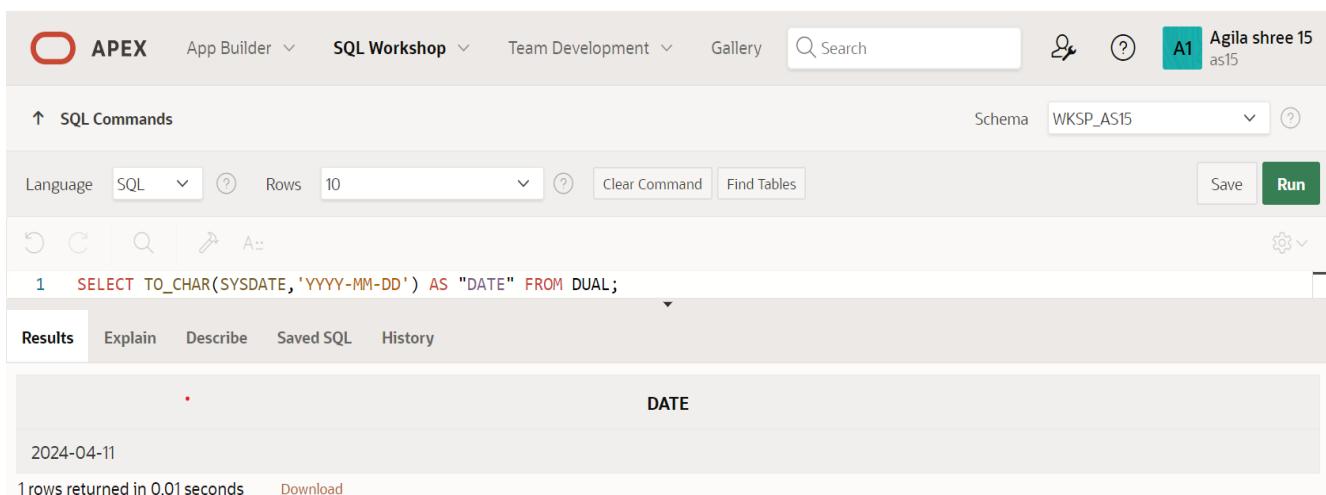
DATE:

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

## QUERY:

```
SELECT TO_CHAR(SYSDATE,'YYYY-MM-DD') AS "DATE" FROM DUAL;
```

## OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (which is selected), Team Development, and Gallery. There is also a search bar and a user profile icon for 'Agila shree 15'.

The main workspace is titled 'SQL Commands'. It features a toolbar with icons for Undo, Redo, Find, Replace, and others. Below the toolbar, there are dropdown menus for Language (set to SQL) and Rows (set to 10), along with buttons for Clear Command and Find Tables. On the right side of the toolbar are Save and Run buttons.

The SQL command entered is:

```
1  SELECT TO_CHAR(SYSDATE,'YYYY-MM-DD') AS "DATE" FROM DUAL;
```

The results section shows the output:

DATE
2024-04-11

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

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

### QUERY:

```
SELECT EMPLOYEE_ID, LAST_NAME, SALARY, SALARY+(SALARY*0.155) AS "NEW SALARY"  
FROM EMPLOYEES;
```

### OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, and Gallery. A search bar and user information (A1 Agila shree 15 as15) are also present. The main area is titled 'SQL Commands' with a schema dropdown set to 'WKSP\_AS15'. The query entered is:

```
3  SELECT EMPLOYEE_ID, LAST_NAME, SALARY, SALARY+(SALARY*0.155) AS "NEW SALARY"  
4  FROM EMPLOYEES;
```

The results tab is selected, displaying the following data:

EMPLOYEE_ID	LAST_NAME	SALARY	NEW SALARY
2	UMA	7000	8085
176	EMANUEL	12000	13860
3	PARTHI	10650.82	12301.6971
4	JANE	23400.12	27027.1386
1	JAY	13400.12	15477.1386

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

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

### QUERY:

```
SELECT EMPLOYEE_ID, LAST_NAME, SALARY, SALARY+(SALARY*15.5/100) AS "NEW SALARY", (SALARY+(SALARY*15.5/100))-SALARY AS "INCREASE"  
FROM EMPLOYEES.
```

### OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, and Gallery. A search bar and user information (A1 Agila shree 15 as15) are also present. The main area is titled 'SQL Commands' with a schema dropdown set to 'WKSP\_AS15'. The query entered is:

```
6  SELECT EMPLOYEE_ID, LAST_NAME, SALARY, SALARY+(SALARY*15.5/100) AS "NEW SALARY", (SALARY+(SALARY*15.5/100))-SALARY AS "INCREASE"  
7  FROM EMPLOYEES;
```

The results tab is selected, displaying the following data:

EMPLOYEE_ID	LAST_NAME	SALARY	NEW SALARY	INCREASE
2	UMA	7000	8085	1085
176	EMANUEL	12000	13860	1860
3	PARTHI	10650.82	12301.6971	1650.8771
4	JANE	23400.12	27027.1386	3627.0186
1	JAY	13400.12	15477.1386	2077.0186

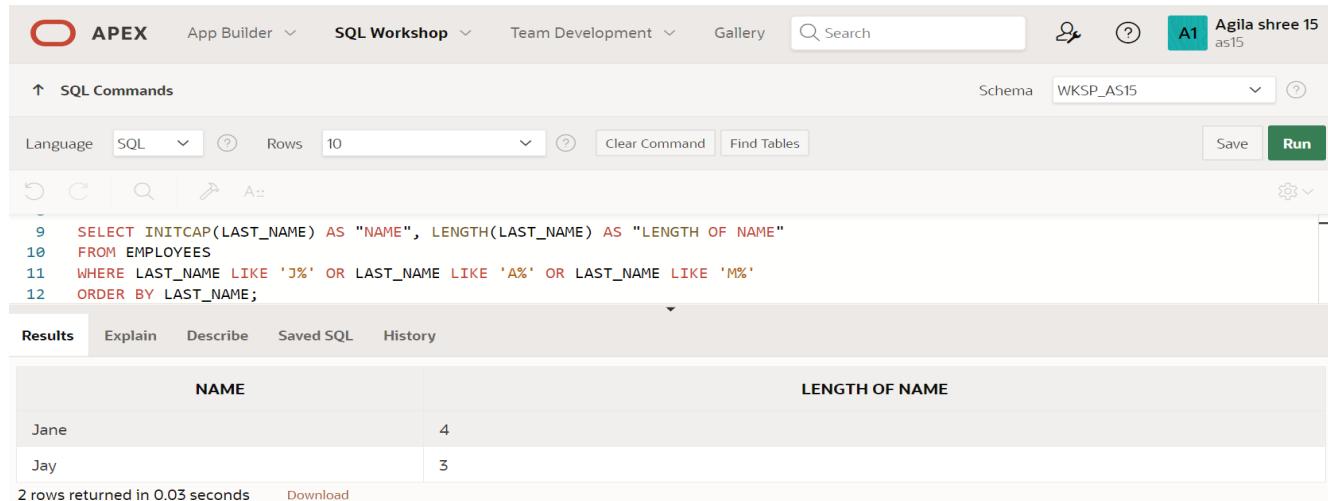
5 rows returned in 0.01 seconds [Download](#)

4. Write a query that displays the last name (with the first letter uppercase and all other letters lowercase) and the length of the last name for all 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) AS "NAME",
LENGTH(LAST_NAME) AS "LENGTH OF NAME"
FROM EMPLOYEES
WHERE LAST_NAME LIKE 'J%' OR
LAST_NAME LIKE 'A%' OR
LAST_NAME LIKE 'M%'
ORDER BY LAST_NAME;
```

**OUTPUT:**



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (which is selected), 'Team Development', 'Gallery', a search bar, and a user profile 'A1 Agila shree 15 as15'. Below the toolbar, the schema is set to 'WKSP\_AS15'. The main area contains the SQL command from step 4, which returns two rows: 'Jane' with a length of 4 and 'Jay' with a length of 3. The results are displayed in a tabular format with columns 'NAME' and 'LENGTH OF NAME'.

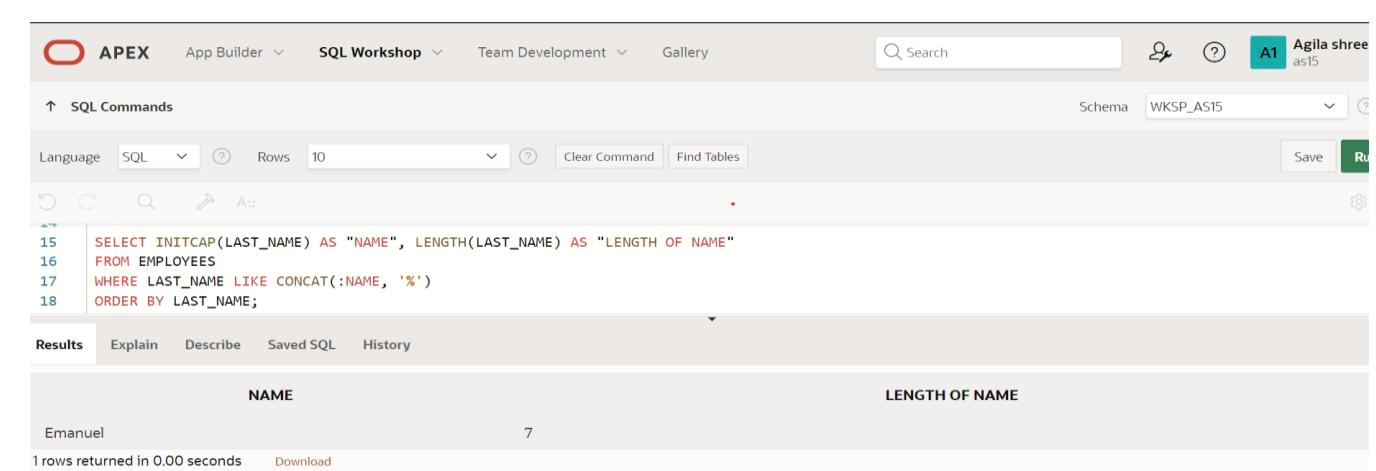
NAME	LENGTH OF NAME
Jane	4
Jay	3

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) AS "NAME", LENGTH(LAST_NAME) AS
"LENGTH OF NAME"
FROM EMPLOYEES
WHERE LAST_NAME LIKE CONCAT(:NAME, '%')
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', a search bar, and a user profile 'A1 Agila shree 15 as15'. Below the toolbar, the schema is set to 'WKSP\_AS15'. The main area contains the SQL command from step 5, which returns one row: 'Emanuel' with a length of 7. The results are displayed in a tabular format with columns 'NAME' and 'LENGTH OF NAME'.

NAME	LENGTH OF NAME
Emanuel	7

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

### QUERY:

```
SELECT LAST_NAME, ROUND(MONTHS_BETWEEN(SYSDATE,HIRE_DATE),0)  
MONTHS_WORKED FROM EMPLOYEES  
ORDER BY 2;
```

### OUTPUT:

The screenshot shows the Oracle 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 'Agila shree 15' (A1). The main area has a search bar and various toolbar buttons. The schema is set to 'WKSP\_AS15'. The SQL command entered is:

```
24  SELECT LAST_NAME, ROUND(MONTHS_BETWEEN(SYSDATE,HIRE_DATE),0) MONTHS_WORKED FROM EMPLOYEES  
25  ORDER BY 2;
```

The results tab is selected, displaying the following data:

LAST_NAME	MONTHS_WORKED
UMA	296
JAY	299
PARTHI	312
JANE	313
EMANUEL	359

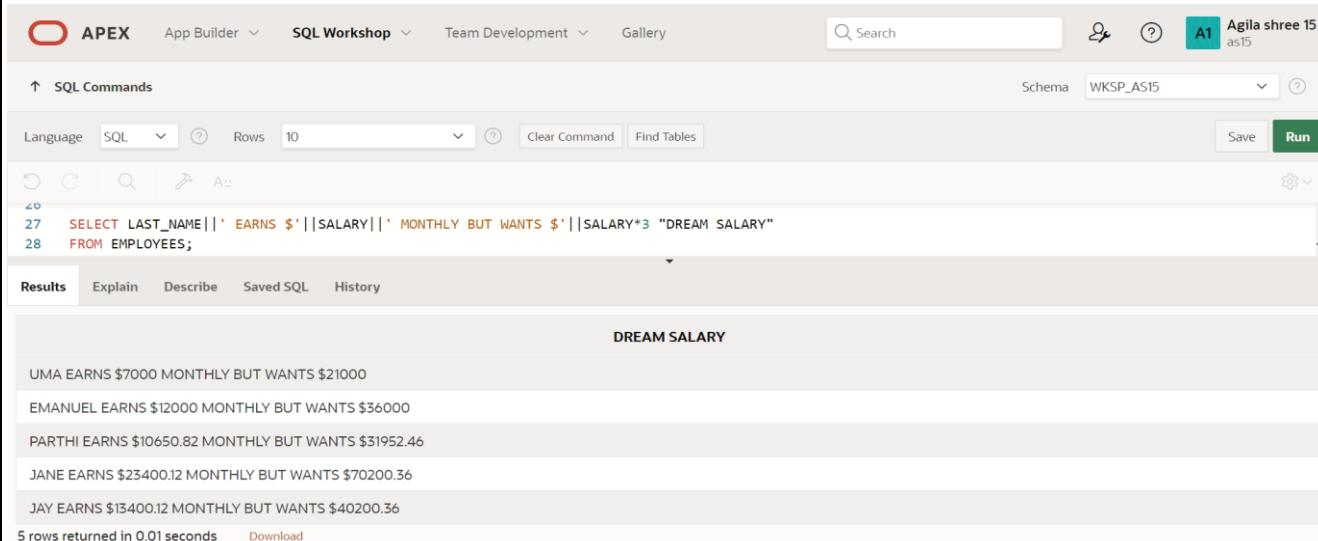
At the bottom, it says '5 rows returned in 0.00 seconds' and has a 'Download' link.

7. Create a report that produces the following for each employee: earns monthly but wants . Label the column Dream Salaries.

### QUERY:

```
SELECT LAST_NAME||' EARNS $'||SALARY||' MONTHLY BUT WANTS  
$'||SALARY*3 "DREAM SALARY"  
FROM EMPLOYEES;
```

### OUTPUT:



The screenshot shows the Oracle 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 (A1 Agila shree 15 as15) and a search bar. The main area has tabs for SQL Commands, Results, Explain, Describe, Saved SQL, and History. The SQL Commands tab contains the following code:

```
27  SELECT LAST_NAME||' EARNS $'||SALARY||' MONTHLY BUT WANTS $'||SALARY*3 "DREAM SALARY"  
28  FROM EMPLOYEES;
```

The Results tab displays the output:

DREAM SALARY	
UMA	EARNS \$7000 MONTHLY BUT WANTS \$21000
EMANUEL	EARNS \$12000 MONTHLY BUT WANTS \$36000
PARTHI	EARNS \$10650.82 MONTHLY BUT WANTS \$31952.46
JANE	EARNS \$23400.12 MONTHLY BUT WANTS \$70200.36
JAY	EARNS \$13400.12 MONTHLY BUT WANTS \$40200.36

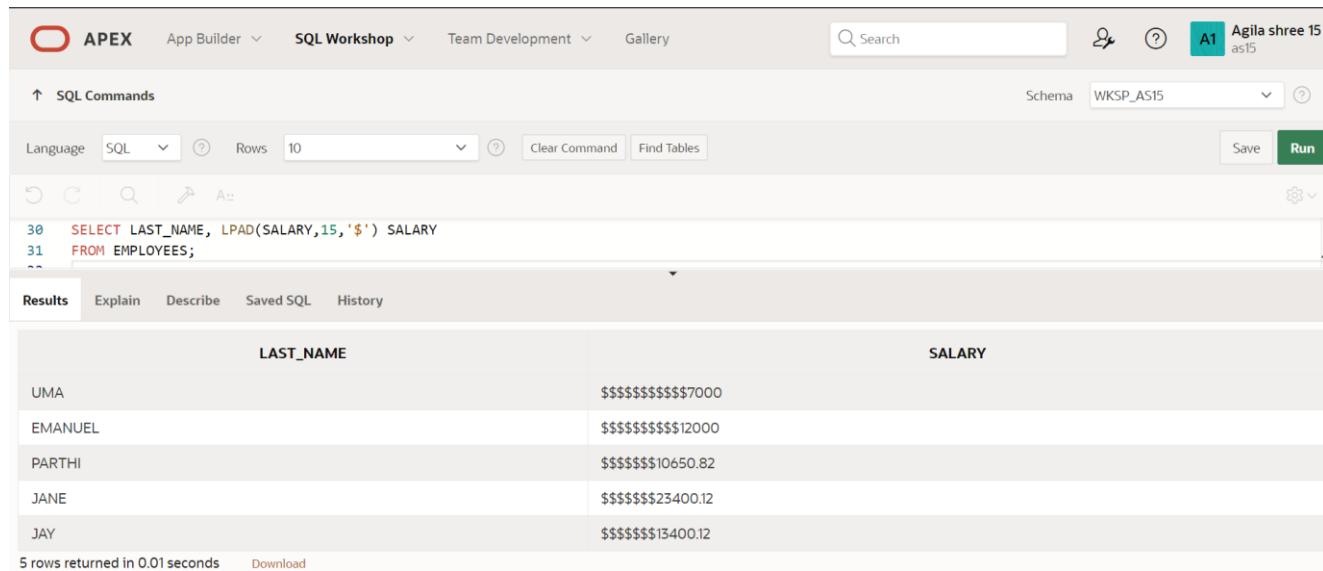
Below the results, it says "5 rows returned in 0.01 seconds" and has a "Download" link.

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

### QUERY:

```
SELECT LAST_NAME, LPAD(SALARY,15,'$') SALARY  
FROM EMPLOYEES;
```

### OUTPUT:



The screenshot shows the Oracle 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 (A1 Agila shree 15 as15) and a search bar. The main area has tabs for SQL Commands, Results, Explain, Describe, Saved SQL, and History. The SQL Commands tab contains the following code:

```
30  SELECT LAST_NAME, LPAD(SALARY,15,'$') SALARY  
31  FROM EMPLOYEES;
```

The Results tab displays the output:

LAST_NAME	SALARY
UMA	\$\$\$\$\$\$\$\$\$\$7000
EMANUEL	\$\$\$\$\$\$\$\$\$\$12000
PARTHI	\$\$\$\$\$\$10650.82
JANE	\$\$\$\$\$\$23400.12
JAY	\$\$\$\$\$\$13400.12

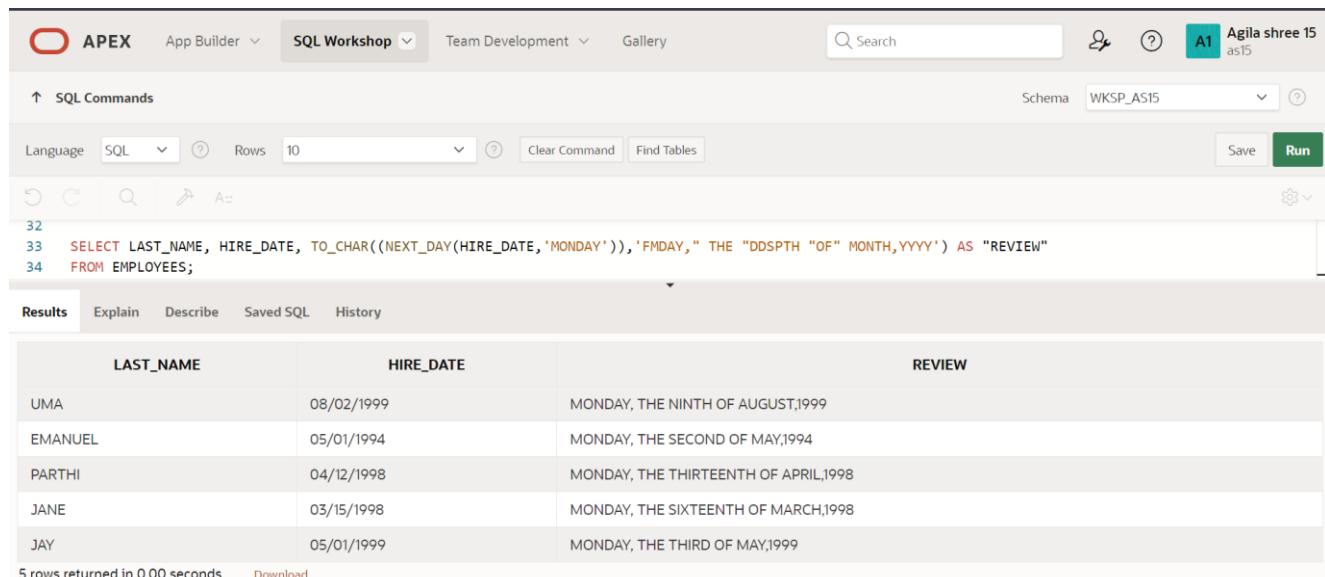
Below the results, it says "5 rows returned in 0.01 seconds" and has a "Download" link.

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 "DDSPHTH "OF"  
MONTH,YYYY') AS "REVIEW" FROM EMPLOYEES;
```

### 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 information (A1 Agila shree 15 as15) are also present. The SQL Commands tab is selected, showing the following code:

```
32  
33  SELECT LAST_NAME, HIRE_DATE, TO_CHAR((NEXT_DAY(HIRE_DATE,'MONDAY')),'FMDAY," THE "DDSPHTH "OF"  
34  MONTH,YYYY') AS "REVIEW"  
FROM EMPLOYEES;
```

The Results tab displays the output:

LAST_NAME	HIRE_DATE	REVIEW
UMA	08/02/1999	MONDAY, THE NINTH OF AUGUST,1999
EMANUEL	05/01/1994	MONDAY, THE SECOND OF MAY,1994
PARTHI	04/12/1998	MONDAY, THE THIRTEENTH OF APRIL,1998
JANE	03/15/1998	MONDAY, THE SIXTEENTH OF MARCH,1998
JAY	05/01/1999	MONDAY, THE THIRD OF MAY,1999

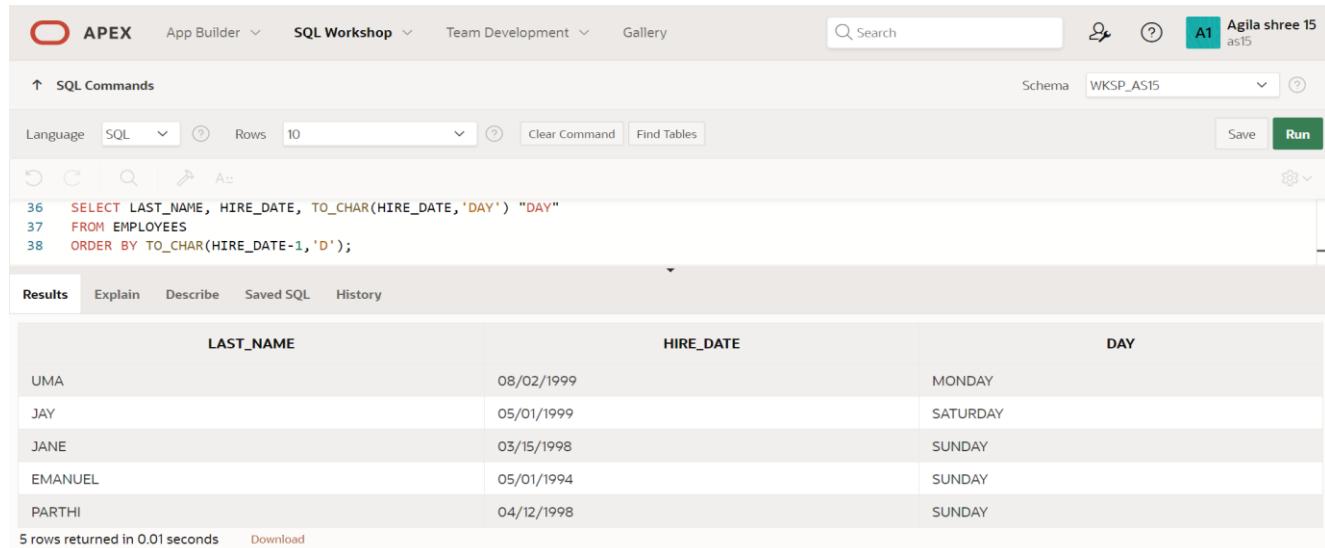
5 rows returned in 0.00 seconds    Download

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

### QUERY:

```
SELECT LAST_NAME, HIRE_DATE, TO_CHAR(HIRE_DATE,'DAY') "DAY"  
FROM EMPLOYEES  
ORDER BY TO_CHAR(HIRE_DATE-1,'D');
```

### OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. A search bar and user information (A1 Agila shree 15 as15) are also present. The SQL Commands tab is selected, showing the following code:

```
36  SELECT LAST_NAME, HIRE_DATE, TO_CHAR(HIRE_DATE,'DAY') "DAY"  
37  FROM EMPLOYEES  
38  ORDER BY TO_CHAR(HIRE_DATE-1,'D');
```

The Results tab displays the output:

LAST_NAME	HIRE_DATE	DAY
UMA	08/02/1999	MONDAY
JAY	05/01/1999	SATURDAY
JANE	03/15/1998	SUNDAY
EMANUEL	05/01/1994	SUNDAY
PARTHI	04/12/1998	SUNDAY

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

EX\_NO:7

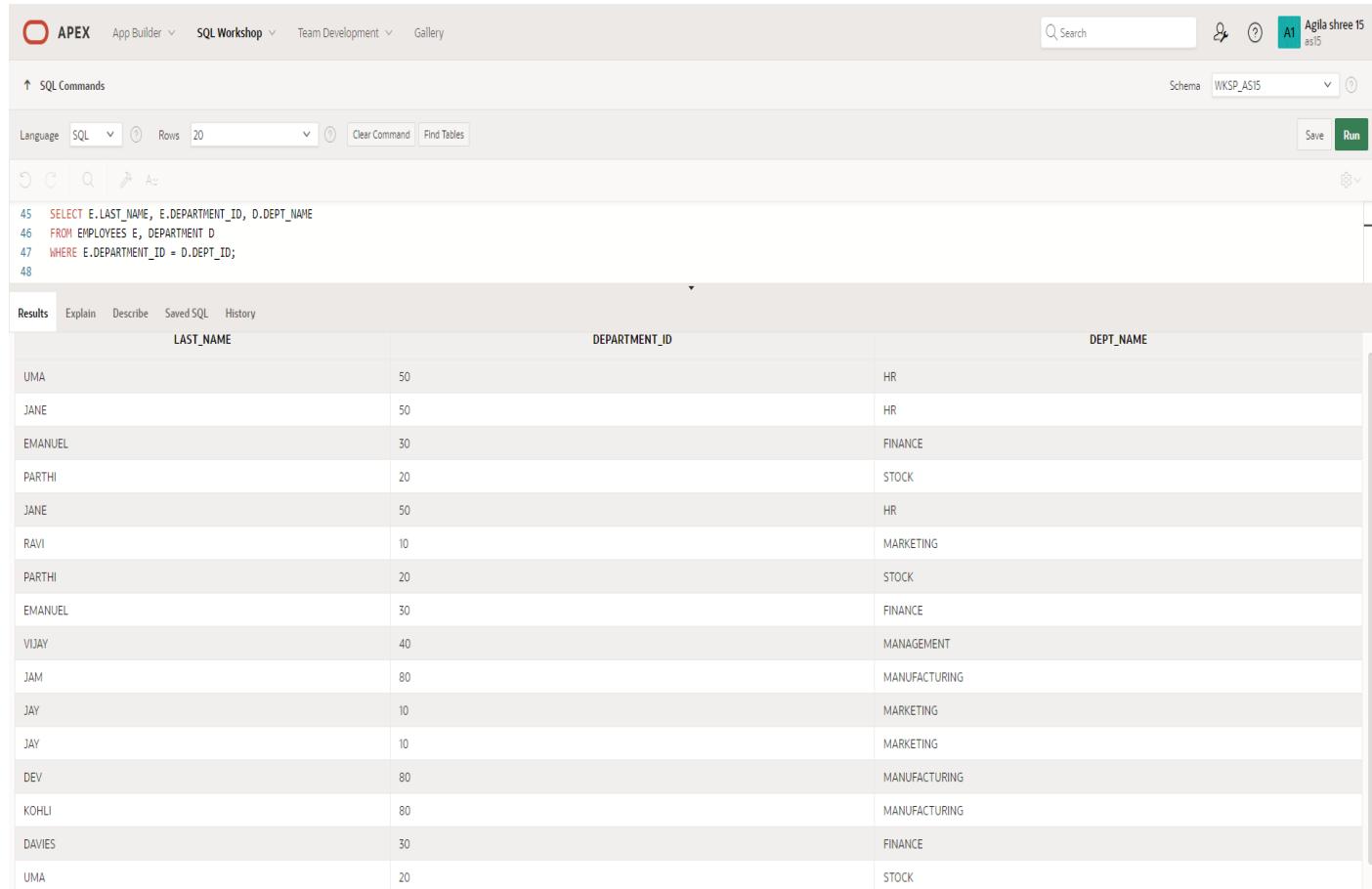
DATE:

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

**QUERY:**

```
SELECT E.LAST_NAME, E.DEPARTMENT_ID, D.DEPT_NAME  
FROM EMPLOYEES E, DEPARTMENT D  
WHERE E.DEPARTMENT_ID = D.DEPT_ID;
```

**OUTPUT:**



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. The main workspace has tabs for SQL Commands, SQL (selected), Rows (set to 20), Clear Command, Find Tables, Schema (WKSP\_A515), and Run. The code editor contains the following SQL query:

```
45: SELECT E.LAST_NAME, E.DEPARTMENT_ID, D.DEPT_NAME  
46: FROM EMPLOYEES E, DEPARTMENT D  
47: WHERE E.DEPARTMENT_ID = D.DEPT_ID;  
48:
```

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

LAST_NAME	DEPARTMENT_ID	DEPT_NAME
UMA	50	HR
JANE	50	HR
EMANUEL	30	FINANCE
PARTHI	20	STOCK
JANE	50	HR
RAVI	10	MARKETING
PARTHI	20	STOCK
EMANUEL	30	FINANCE
VIJAY	40	MANAGEMENT
JAM	80	MANUFACTURING
JAY	10	MARKETING
JAY	10	MARKETING
DEV	80	MANUFACTURING
KOHLI	80	MANUFACTURING
DAVIES	30	FINANCE
UMA	20	STOCK

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 EMPLOYEES, DEPARTMENT  
WHERE EMPLOYEES.DEPARTMENT_ID = DEPARTMENT.DEPT_ID  
AND EMPLOYEES.DEPARTMENT_ID = 80;
```

### OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery'. A search bar and a user icon 'A1 Agila shree 15' are also present. The main area is titled 'SQL Commands' with tabs for 'Language' (set to 'SQL'), 'Rows' (set to 10), 'Clear Command', and 'Find Tables'. Below this is a toolbar with icons for search, refresh, and run. The code editor contains the following SQL query:

```
49  SELECT DISTINCT JOB_ID, LOCATION_ID  
50  FROM EMPLOYEES, DEPARTMENT  
51  WHERE EMPLOYEES.DEPARTMENT_ID = DEPARTMENT.DEPT_ID  
52  AND EMPLOYEES.DEPARTMENT_ID = 80;  
53
```

The results pane shows a table with two columns: 'JOB\_ID' and 'LOCATION\_ID'. The data is as follows:

JOB_ID	LOCATION_ID
DESIGNER	6
SUPERVISOR	6
ENGINEER	6

Below the table, it says '3 rows returned in 0.06 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.DEPT_NAME, D.LOCATION_ID, L.CITY  
FROM EMPLOYEES E, DEPARTMENT D, LOCATION L  
WHERE DEPARTMENT_ID = DEPT_ID  
AND D.LOCATION_ID = L.LOCATION_ID  
AND COMMISSION_PCT IS NOT NULL;
```

### OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery'. A search bar and a user icon 'A1 Agila shree 15' are also present. The main area is titled 'SQL Commands' with tabs for 'Language' (set to 'SQL'), 'Rows' (set to 10), 'Clear Command', and 'Find Tables'. Below this is a toolbar with icons for search, refresh, and run. The code editor contains the following SQL query:

```
54  SELECT E.LAST_NAME, D.DEPT_NAME, D.LOCATION_ID, L.CITY  
55  FROM EMPLOYEES E, DEPARTMENT D, LOCATION L  
56  WHERE DEPARTMENT_ID = DEPT_ID  
57  AND D.LOCATION_ID = L.LOCATION_ID  
58  AND COMMISSION_PCT IS NOT NULL;  
59
```

The results pane shows a table with four columns: 'LAST\_NAME', 'DEPT\_NAME', 'LOCATION\_ID', and 'CITY'. The data is as follows:

LAST_NAME	DEPT_NAME	LOCATION_ID	CITY
JANE	HR	5	LONDON
EMANUEL	FINANCE	3	VALHALLA
JANE	HR	5	LONDON
RAVI	MARKETING	1	CHENNAI
EMANUEL	FINANCE	3	VALHALLA
VILAY	MANAGEMENT	4	DC
JAM	MANUFACTURING	6	TORONTO
JAY	MARKETING	1	CHENNAI
JAY	MARKETING	1	CHENNAI
DEV	MANUFACTURING	6	TORONTO

Below the table, it says 'More than 10 rows available. Increase rows selector to view more rows.' and '10 rows returned in 0.06 seconds' with 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, DEPT_NAME  
FROM EMPLOYEES, DEPARTMENT  
WHERE DEPARTMENT_ID = DEPT_ID  
AND LAST_NAME LIKE '%a%';
```

### OUTPUT:

```
60  SELECT LAST_NAME, DEPT_NAME  
61  FROM EMPLOYEES, DEPARTMENT  
62  WHERE DEPARTMENT_ID = DEPT_ID  
63  AND LAST_NAME LIKE '%a%';      -- SHOULD BE '%a' BUT SINCE I ENTERED THE DATAS IN CAPS, O/P COMES TO BE NO DATA FOUND  
64
```

LAST_NAME	DEPT_NAME
UMA	HR
JANE	HR
EMANUEL	FINANCE
PARTHI	STOCK
JANE	HR
RAVI	MARKETING
PARTHI	STOCK
EMANUEL	FINANCE
VIJAY	MANAGEMENT
JAM	MANUFACTURING

More than 10 rows available. Increase rows selector to view more rows.  
10 rows returned in 0.01 seconds [Download](#)

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

### QUERY:

```
SELECT LAST_NAME, JOB_ID, DEPARTMENT_ID, DEPT_NAME  
FROM EMPLOYEES JOIN DEPARTMENT D  
ON (DEPARTMENT_ID = DEPT_ID)  
JOIN LOCATION L  
ON (D.LOCATION_ID = L.LOCATION_ID)  
WHERE LOWER(L.CITY) = 'toronto';
```

### OUTPUT:

```
65  SELECT LAST_NAME, JOB_ID, DEPARTMENT_ID, DEPT_NAME  
66  FROM EMPLOYEES JOIN DEPARTMENT D  
67  ON (DEPARTMENT_ID = DEPT_ID)  
68  JOIN LOCATION L  
69  ON (D.LOCATION_ID = L.LOCATION_ID)  
70  WHERE LOWER(L.CITY) = 'toronto';
```

LAST_NAME	JOB_ID	DEPARTMENT_ID	DEPT_NAME
JAM	DESIGNER	80	MANUFACTURING
DEV	ENGINEER	80	MANUFACTURING
KOHLI	SUPERVISOR	80	MANUFACTURING

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

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.EMPLOYEE_ID "EMP#",  
M.LAST_NAME "MANAGER", M.EMPLOYEE_ID "MGR#"  
FROM EMPLOYEES W JOIN EMPLOYEES M  
ON (W.MANAGER_ID = M.EMPLOYEE_ID);
```

## OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, and Gallery. The right side shows a user profile for 'Agila shree 15' and a schema dropdown set to 'WKSP\_A51'. The main area has tabs for SQL Commands, Results, Explain, Describe, Saved SQL, and History. The SQL Commands tab contains the executed query:

```
72: SELECT W.LAST_NAME "EMPLOYEE", W.EMPLOYEE_ID "EMP#",  
73: M.LAST_NAME "MANAGER", M.EMPLOYEE_ID "MGR#"  
74: FROM EMPLOYEES W JOIN EMPLOYEES M  
75: ON (W.MANAGER_ID = M.EMPLOYEE_ID);
```

The Results tab displays the query results in a grid:

EMPLOYEE	EMP#	MANAGER	MGR#
PARTHI	103	RAVI	101
JAY	100	RAVI	101
UMA	102	RAVI	101
DAVIES	110	EMANUEL	105
JANE	104	VIJAY	106
RAVI	101	VIJAY	106
EMANUEL	105	VIJAY	106
JAM	107	VIJAY	106
KOHLI	109	VIJAY	106
DEV	108	JAM	107

Below the table, it says '10 rows returned in 0.01 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.EMPLOYEE_ID "EMP#",  
M.LAST_NAME "MANAGER", M.EMPLOYEE_ID "MGR#"  
FROM EMPLOYEES W  
LEFT OUTER JOIN EMPLOYEES M  
ON (W.MANAGER_ID = M.EMPLOYEE_ID)  
ORDER BY W.EMPLOYEE_ID;
```

### OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, and Gallery. The SQL Workshop tab is selected. The schema is set to WKSP\_AS15. The query editor contains the provided SQL code. The results tab is selected, displaying the output in a grid format. The output shows 10 rows of data, with the last row indicating that more than 10 rows are available. The columns in the results grid are labeled: EMPLOYEE, EMP#, MANAGER, and MGR#.

EMPLOYEE	EMP#	MANAGER	MGR#
JAY	1	-	-
UMA	2	-	-
PARTHI	3	-	-
JANE	4	-	-
JAY	100	RAVI	101
RAVI	101	VIJAY	106
UMA	102	RAVI	101
PARTHI	103	RAVI	101
JANE	104	VIJAY	106
EMANUEL	105	VIJAY	106

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.LAST_NAME EMPLOYEE, E.DEPARTMENT_ID DEPARTMENT,  
C.LAST_NAME COLLEAGUE  
FROM EMPLOYEES E JOIN EMPLOYEES C  
ON (E.DEPARTMENT_ID = C.DEPARTMENT_ID)  
WHERE E.EMPLOYEE_ID <> C.EMPLOYEE_ID  
ORDER BY E.DEPARTMENT_ID, E.LAST_NAME;
```

### OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, and a search bar. The schema is set to WKSP\_ASIS. The SQL command window contains the query from above. The results window shows the output in a tabular format with columns: EMPLOYEE, DEPARTMENT, and COLLEAGUE. The data is as follows:

EMPLOYEE	DEPARTMENT	COLLEAGUE
JAY	10	RAVI
JAY	10	JAY
JAY	10	JAY
JAY	10	RAVI
RAVI	10	JAY
RAVI	10	JAY
PARTHI	20	UMA
PARTHI	20	UMA
PARTHI	20	PARTHI
PARTHI	20	PARTHI

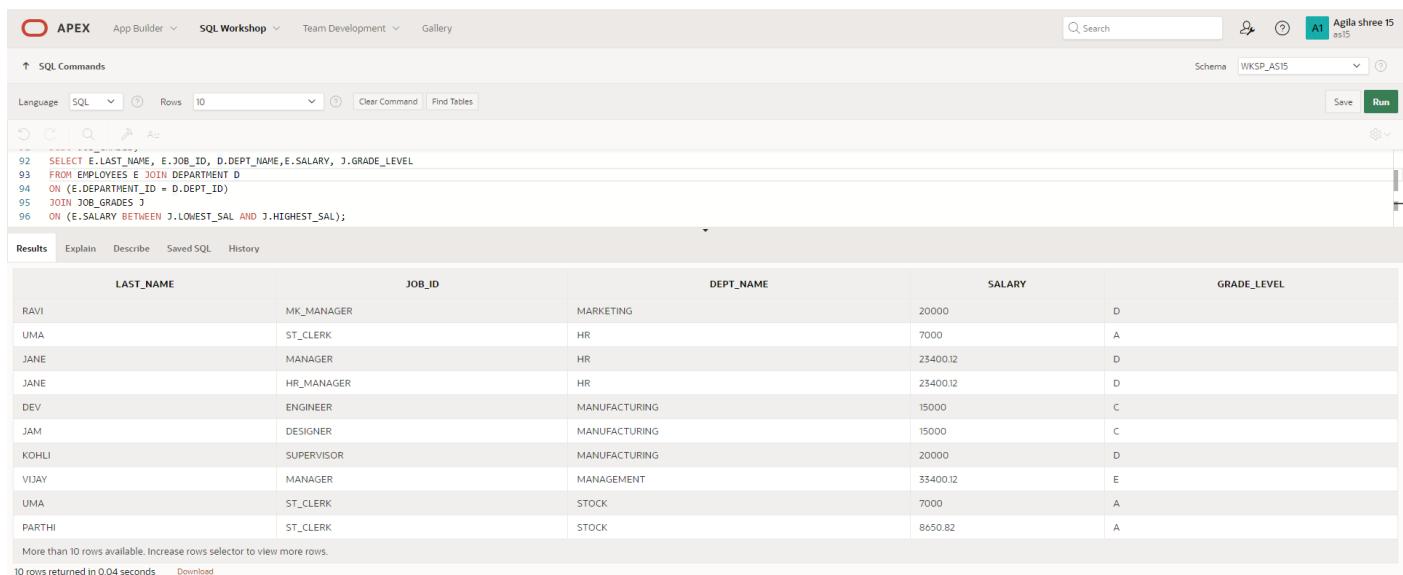
Below the results, a message indicates "More than 10 rows available. Increase rows selector to view more rows." and "10 rows returned in 0.01 seconds".

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

### QUERY:

```
SELECT E.LAST_NAME, E.JOB_ID, D.DEPT_NAME, E.SALARY, J.GRADE_LEVEL  
  
FROM EMPLOYEES E JOIN DEPARTMENT D  
  
ON (E.DEPARTMENT_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 query from step 9 is executed, and the results are displayed in a grid. The columns are LAST\_NAME, JOB\_ID, DEPT\_NAME, SALARY, and GRADE\_LEVEL. The data includes rows for RAVI, UMA, JANE, KOHLI, VIJAY, and PARTHI, among others, with their respective job roles and department assignments.

LAST_NAME	JOB_ID	DEPT_NAME	SALARY	GRADE_LEVEL
RAVI	MK_MANAGER	MARKETING	20000	D
UMA	ST_CLERK	HR	7000	A
JANE	MANAGER	HR	23400.12	D
JANE	HR_MANAGER	HR	23400.12	D
DEV	ENGINEER	MANUFACTURING	15000	C
JAM	DESIGNER	MANUFACTURING	15000	C
KOHLI	SUPERVISOR	MANUFACTURING	20000	D
VIJAY	MANAGER	MANAGEMENT	33400.12	E
UMA	ST_CLERK	STOCK	7000	A
PARTHI	ST_CLERK	STOCK	8650.82	A

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 EMPLOYEES E JOIN EMPLOYEES DAVIES
```

```
ON (DAVIES.LAST_NAME = 'DAVIES')
```

```
WHERE DAVIES.HIRE_DATE < E.HIRE_DATE;
```

### OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The query from step 10 is executed, and the results are displayed in a grid. The columns are LAST\_NAME and HIRE\_DATE. The data shows two rows: RAVI (hire date 01/07/2004) and DEV (hire date 10/12/2004), both of whom were hired after the employee Davies.

LAST_NAME	HIRE_DATE
RAVI	01/07/2004
DEV	10/12/2004

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:

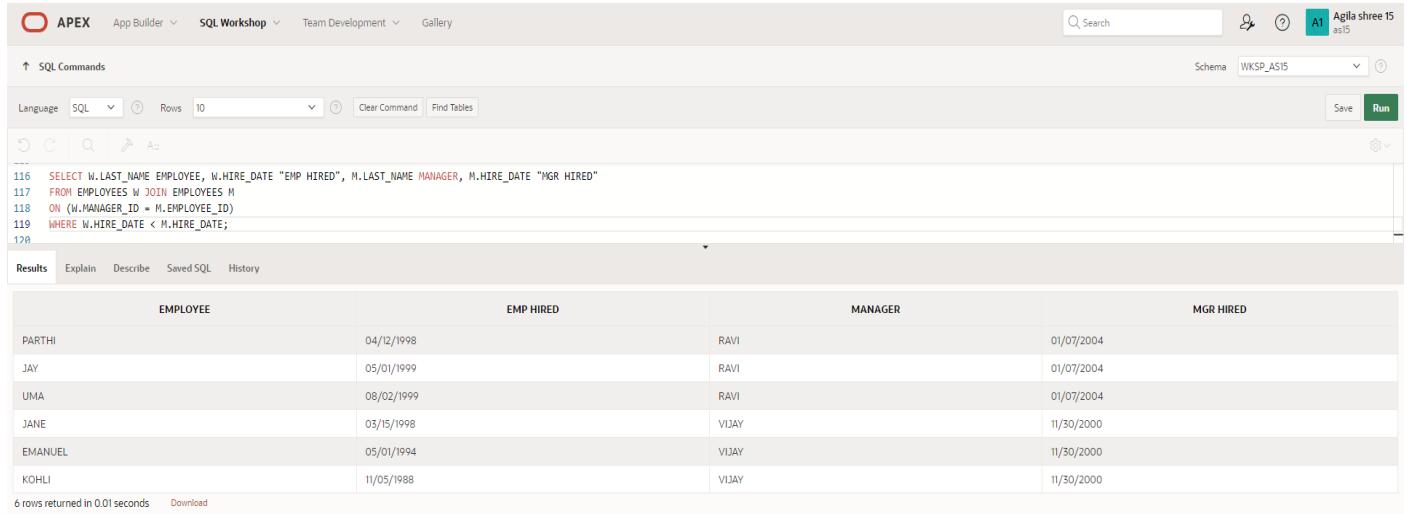
```
SELECT W.LAST_NAME EMPLOYEE, W.HIRE_DATE "EMP HIRED", M.LAST_NAME MANAGER, M.HIRE_DATE "MGR HIRED"
```

```
FROM EMPLOYEES W JOIN EMPLOYEES M
```

```
ON (W.MANAGER_ID = M.EMPLOYEE_ID)
```

```
WHERE 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, Team Development, and Gallery. The right side shows a user profile for 'Agila shree 15' and a schema dropdown set to 'WKSP\_AS15'. The main area has tabs for SQL Commands, Language (set to SQL), Rows (set to 10), Clear Command, Find Tables, Save, and Run. The SQL editor contains the query code from the previous steps. The results tab is selected, displaying a table with four columns: EMPLOYEE, EMP HIRED, MANAGER, and MGR HIRED. The data rows are:

EMPLOYEE	EMP HIRED	MANAGER	MGR HIRED
PARTHI	04/12/1998	RAVI	01/07/2004
JAY	05/01/1999	RAVI	01/07/2004
UMA	08/02/1999	RAVI	01/07/2004
JANE	03/15/1998	VIJAY	11/30/2000
EMANUEL	05/01/1994	VIJAY	11/30/2000
KOHLI	11/05/1988	VIJAY	11/30/2000

At the bottom left, it says '6 rows returned in 0.01 seconds' and there is a 'Download' link.

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) AS "MAXIMUM", ROUND(MIN(SALARY),0) AS "MINIMUM", ROUND(SUM(SALARY),0) AS "SUM",  
ROUND(AVG(SALARY),0) AS "AVERAGE" FROM EMPLOYEES;
```

**OUTPUT:**

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, and Gallery. On the right, there's a search bar, a help icon, and a session identifier A1 Agila shree 15. The main area is titled 'SQL Commands' with a language dropdown set to SQL, a rows dropdown set to 10, and buttons for Clear Command and Find Tables. Below this is a toolbar with icons for Undo, Redo, Search, and Run. The SQL editor contains the following code:

```
6: SELECT ROUND(MAX(SALARY),0) AS "MAXIMUM", ROUND(MIN(SALARY),0) AS "MINIMUM", ROUND(SUM(SALARY),0) AS "SUM",  
7: ROUND(AVG(SALARY),0) AS "AVERAGE" FROM EMPLOYEES;  
8:
```

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

	MAXIMUM	MINIMUM	SUM	AVERAGE
	33400	5500	299802	16656

Below the table, it says '1 rows returned in 0.00 seconds' and has a 'Download' link.

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

### QUERY:

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

### OUTPUT:

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

```
9 SELECT JOB_ID,ROUND(MAX(SALARY),0) AS "MAXIMUM", ROUND(MIN(SALARY),0) AS "MINIMUM", ROUND(SUM(SALARY),0) AS "SUM",  
10 ROUND(AVG(SALARY),0) AS "AVERAGE" FROM EMPLOYEES GROUP BY JOB_ID;  
11
```

The results table has columns: JOB\_ID, MAXIMUM, MINIMUM, SUM, and AVERAGE. The data is as follows:

JOB_ID	MAXIMUM	MINIMUM	SUM	AVERAGE
A_MANAGER	12000	12000	12000	12000
FI_MANAGER	22000	22000	22000	22000
DEVELOPER	20000	20000	20000	20000
ST_CLERK	10651	7000	33302	8325
MANAGER	35400	23400	56800	28400
SUPERVISOR	20000	20000	20000	20000
MK_MANAGER	20000	20000	20000	20000
ENGINEER	15000	15000	15000	15000
TESTER	5500	5500	5500	5500
SL_REP	13400	13400	26800	13400

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(JOB_ID) FROM EMPLOYEES  
GROUP BY JOB_ID;
```

### OUTPUT:

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

```
12  
13 SELECT JOB_ID,COUNT(JOB_ID) FROM EMPLOYEES  
14 GROUP BY JOB_ID;
```

The results table has columns: JOB\_ID and COUNT(JOB\_ID). The data is as follows:

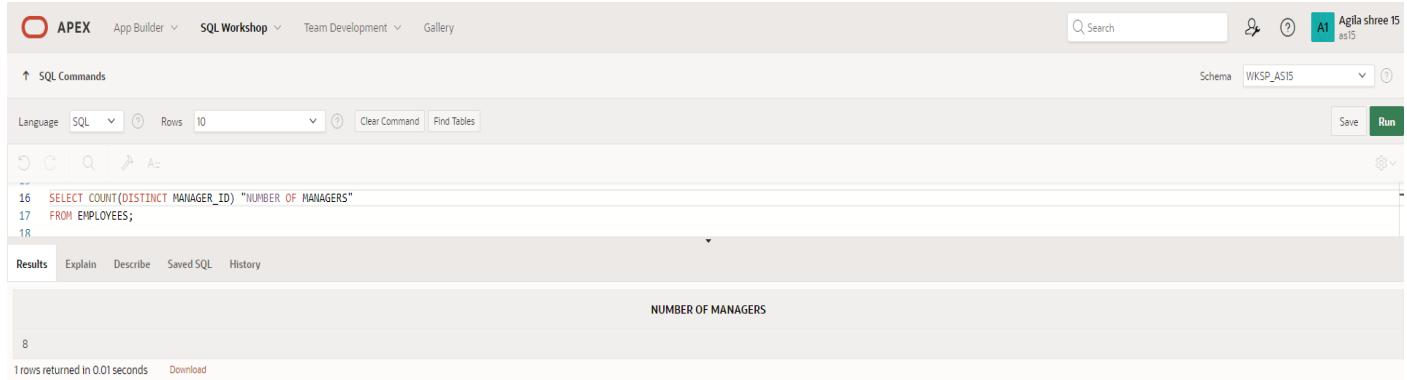
JOB_ID	COUNT(JOB_ID)
A_MANAGER	1
FI_MANAGER	1
DEVELOPER	1
ST_CLERK	4
MANAGER	2
SUPERVISOR	1
MK_MANAGER	1
ENGINEER	1
TESTER	1
SL_REP	2

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:



The screenshot shows the Oracle SQL Workshop interface. The query is executed and returns a single row with the value 8, labeled 'NUMBER OF MANAGERS'.

NUMBER OF MANAGERS
8

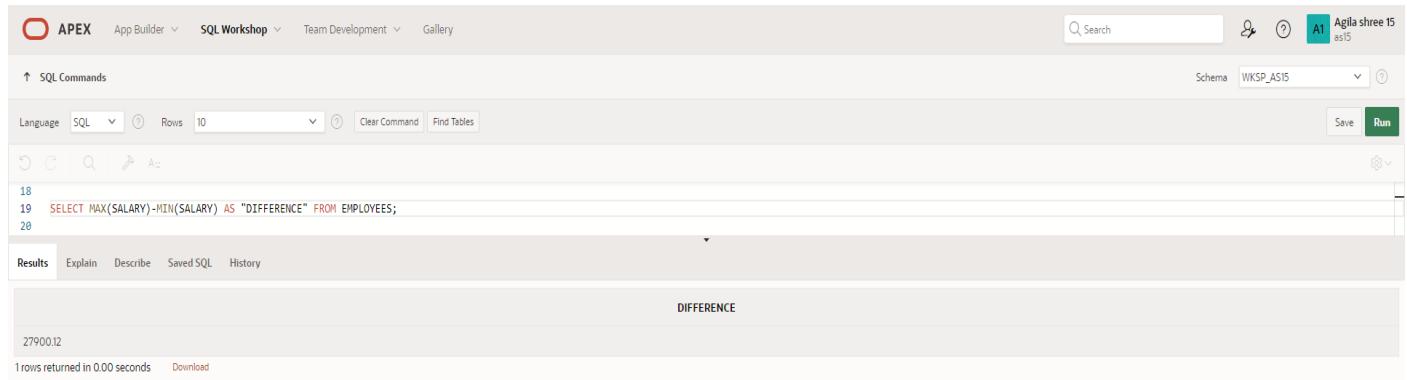
1 rows returned in 0.01 seconds [Download](#)

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

### QUERY:

```
SELECT MAX(SALARY)-MIN(SALARY) AS "DIFFERENCE" FROM EMPLOYEES;
```

### OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The query is executed and returns a single row with the value 27900.12, labeled 'DIFFERENCE'.

DIFFERENCE
27900.12

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

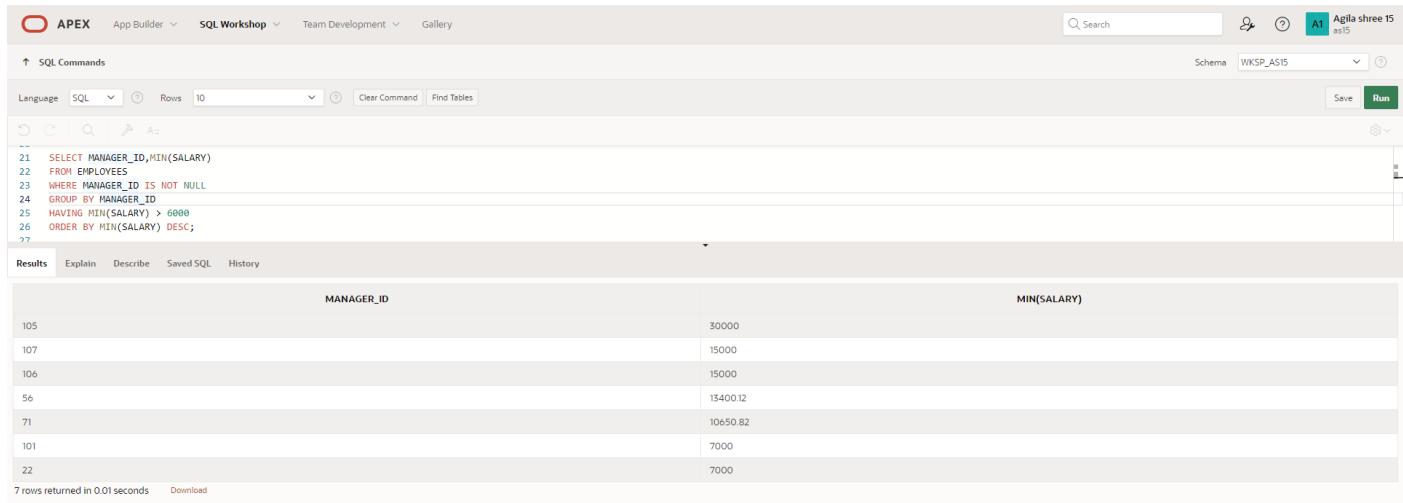
9.Create a report to display the manager number and the salary of the lowest-paid employee for that manager. Exclude anyone whose manager is not known. Exclude any groups where the minimum salary is \$6,000 or less. Sort the output in descending order of salary.

### 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 SQL Workshop interface. The query is displayed in the SQL Commands pane:

```
--  
21 SELECT MANAGER_ID,MIN(SALARY)  
22 FROM EMPLOYEES  
23 WHERE MANAGER_ID IS NOT NULL  
24 GROUP BY MANAGER_ID  
25 HAVING MIN(SALARY) > 6000  
26 ORDER BY MIN(SALARY) DESC;  
?7
```

The Results pane displays the output:

MANAGER_ID	MIN(SALARY)
105	30000
107	15000
106	15000
56	15400.12
71	10650.82
101	7000
22	7000

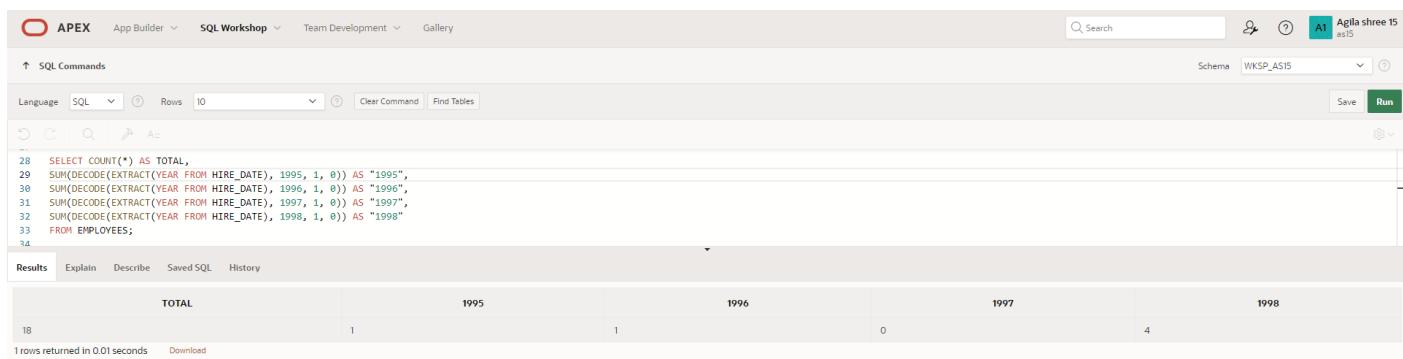
7 rows returned in 0.01 seconds

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(EXTRACT(YEAR FROM HIRE_DATE), 1995, 1, 0)) AS "1995",  
  
SUM (DECODE(EXTRACT(YEAR FROM HIRE_DATE), 1996, 1, 0)) AS "1996",  
  
SUM (DECODE(EXTRACT(YEAR FROM HIRE_DATE), 1997, 1, 0)) AS "1997",  
  
SUM (DECODE(EXTRACT(YEAR FROM HIRE_DATE), 1998, 1, 0)) AS "1998"  
  
FROM EMPLOYEES.
```

### OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The query is displayed in the SQL Commands pane:

```
--  
28 SELECT COUNT(*) AS TOTAL,  
29 SUM(DECODE(EXTRACT(YEAR FROM HIRE_DATE), 1995, 1, 0)) AS "1995",  
30 SUM(DECODE(EXTRACT(YEAR FROM HIRE_DATE), 1996, 1, 0)) AS "1996",  
31 SUM(DECODE(EXTRACT(YEAR FROM HIRE_DATE), 1997, 1, 0)) AS "1997",  
32 SUM(DECODE(EXTRACT(YEAR FROM HIRE_DATE), 1998, 1, 0)) AS "1998"  
33 FROM EMPLOYEES;  
?4
```

The Results pane displays the output:

TOTAL	1995	1996	1997	1998
18	1	1	0	4

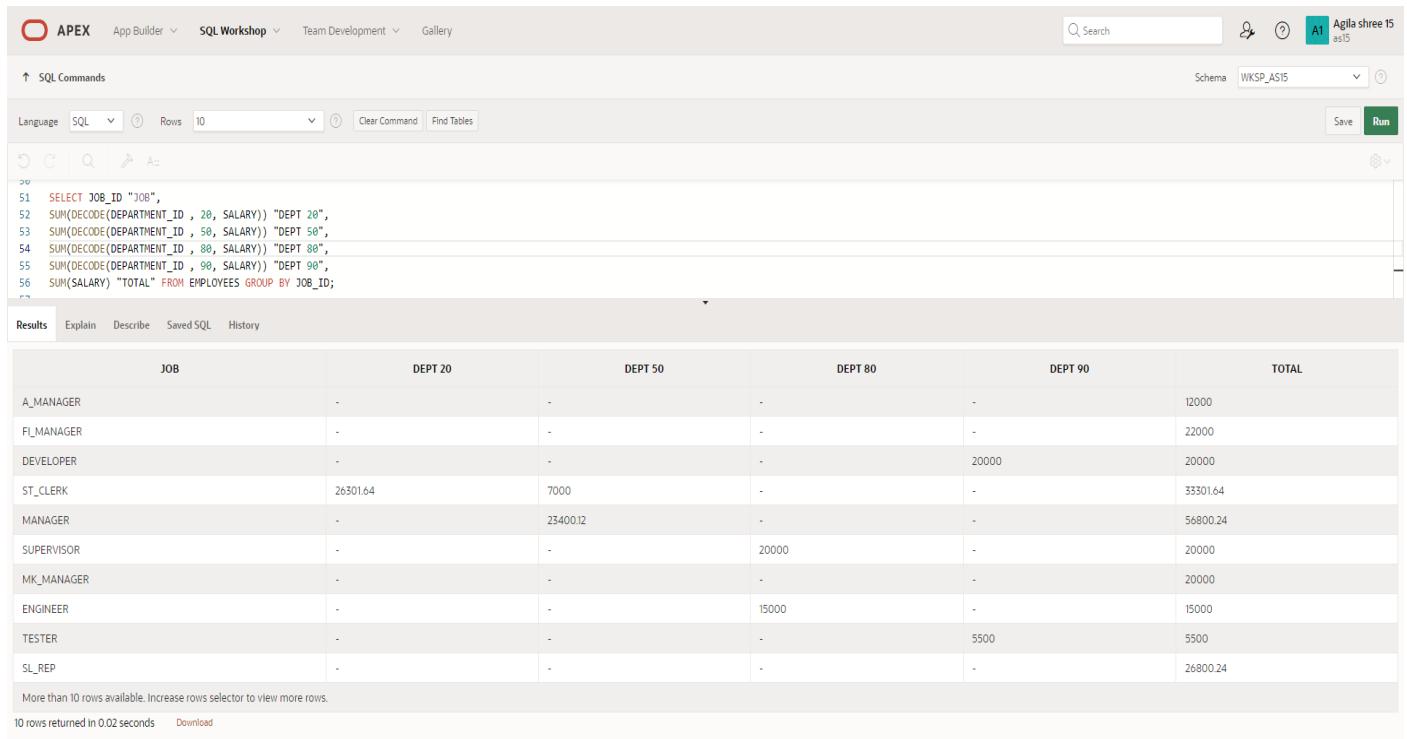
1 rows returned in 0.01 seconds

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

### QUERY:

```
SELECT JOB_ID "JOB",
       SUM(DECODE(DEPARTMENT_ID , 20, SALARY)) "DEPT 20",
       SUM(DECODE(DEPARTMENT_ID , 50, SALARY)) "DEPT 50",
       SUM(DECODE(DEPARTMENT_ID , 80, SALARY)) "DEPT 80",
       SUM(DECODE(DEPARTMENT_ID , 90, SALARY)) "DEPT 90",
       SUM(SALARY) "TOTAL" FROM EMPLOYEES GROUP BY JOB_ID;
```

### OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, and Gallery. The SQL Workshop tab is selected. The schema is set to WKSP\_AS15. The SQL command window contains the query code. The results tab is selected, displaying the output as a matrix:

JOB	DEPT 20	DEPT 50	DEPT 80	DEPT 90	TOTAL
A_MANAGER	-	-	-	-	12000
FI_MANAGER	-	-	-	-	22000
DEVELOPER	-	-	-	20000	20000
ST_CLERK	26301.64	7000	-	-	33301.64
MANAGER	-	23400.12	-	-	56800.24
SUPERVISOR	-	-	20000	-	20000
MK_MANAGER	-	-	-	-	20000
ENGINEER	-	-	15000	-	15000
TESTER	-	-	-	5500	5500
SL_REP	-	-	-	-	26800.24

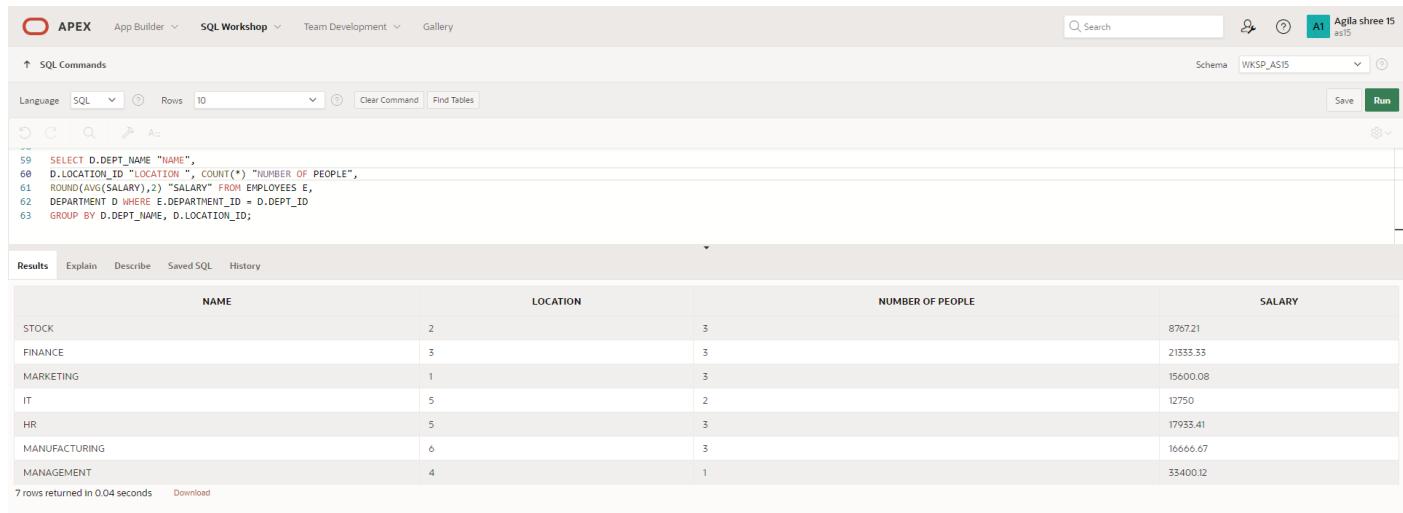
Below the results, a message indicates "More than 10 rows available. Increase rows selector to view more rows." and "10 rows returned in 0.02 seconds".

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.DEPARTMENT_NAME "NAME",
       D.LOCATION_ID "LOCATION ", COUNT(*) "NUMBER OF PEOPLE",
       ROUND(AVG(SALARY),2) "SALARY" FROM EMPLOYEES E,
       DEPARTMENT D WHERE E.DEPARTMENT_ID = D.DEPARTMENT_ID
       GROUP BY D.DEPARTMENT_NAME, D.LOCATION_ID;
```

### OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The SQL command is pasted into the editor, and the results are displayed in a grid format below. The results show the department name, location, number of people, and average salary for each department.

NAME	LOCATION	NUMBER OF PEOPLE	SALARY
STOCK	2	3	8767.21
FINANCE	3	3	21333.33
MARKETING	1	3	15600.08
IT	5	2	12750
HR	5	3	17933.41
MANUFACTURING	6	3	16666.67
MANAGEMENT	4	1	33400.12

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

**RESULT:**

## SUB-QUERIES

EX.NO:9

DATE:

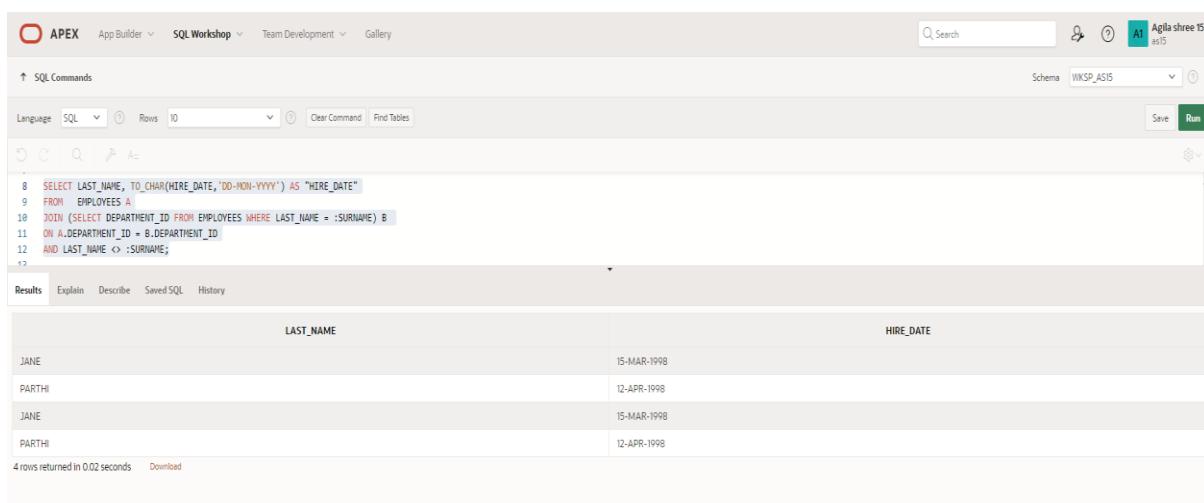
Find the Solution for the following:

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

### **QUERY:**

```
SELECT LAST_NAME, TO_CHAR(HIRE_DATE,'DD-MON-YYYY') AS "HIRE_DATE"
FROM EMPLOYEES A
JOIN (SELECT DEPARTMENT_ID FROM EMPLOYEES WHERE LAST_NAME = :SURNAME) B
ON A.DEPARTMENT_ID = B.DEPARTMENT_ID
AND LAST_NAME <> :SURNAME;
```

### **OUTPUT:**



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

```
8  SELECT LAST_NAME, TO_CHAR(HIRE_DATE,'DD-MON-YYYY') AS "HIRE_DATE"
9  FROM EMPLOYEES A
10 JOIN (SELECT DEPARTMENT_ID FROM EMPLOYEES WHERE LAST_NAME = :SURNAME) B
11 ON A.DEPARTMENT_ID = B.DEPARTMENT_ID
12 AND LAST_NAME <> :SURNAME;
13
```

The results window displays the output:

LAST_NAME	HIRE_DATE
JANE	15-MAR-1998
PARTHI	12-APR-1998
JANE	15-MAR-1998
PARTHI	12-APR-1998

4 rows returned in 0.02 seconds

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

## QUERY:

```
SELECT EMPLOYEE_ID, LAST_NAME, SALARY FROM EMPLOYEES  
WHERE SALARY > (SELECT AVG(SALARY) FROM EMPLOYEES)  
ORDER BY SALARY
```

## OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery'. The top right corner shows 'A1 1815 Agila shree 15'. The main area has tabs for 'SQL Commands' and 'Results'. The 'SQL Commands' tab contains the following code:

```
13  
14  SELECT EMPLOYEE_ID, LAST_NAME, SALARY FROM EMPLOYEES  
15 WHERE SALARY > (SELECT AVG(SALARY) FROM EMPLOYEES)  
16 ORDER BY SALARY;
```

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

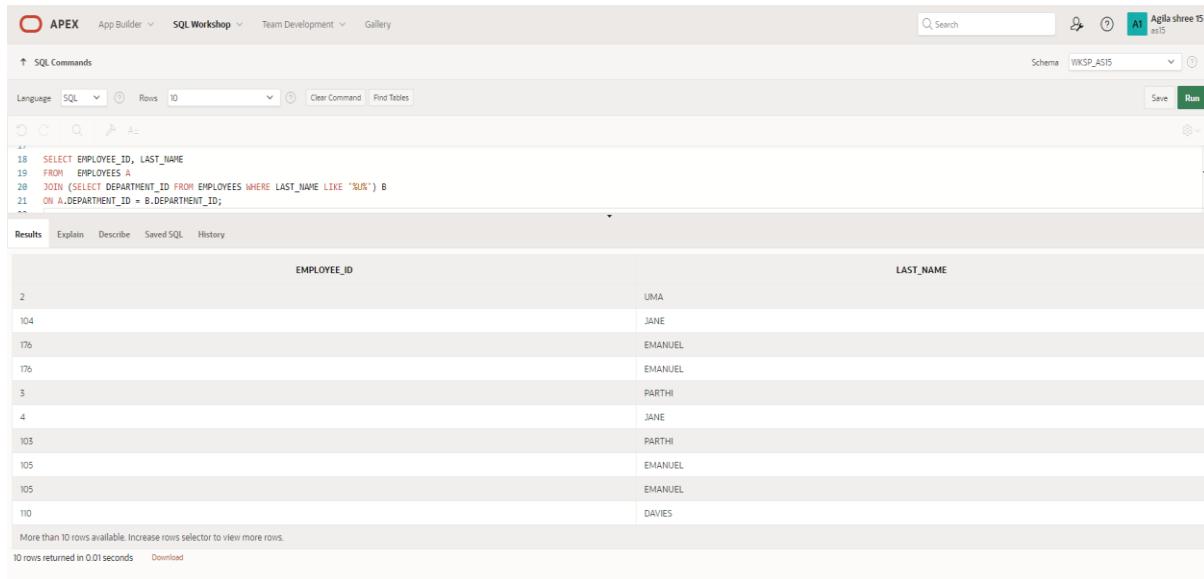
EMPLOYEE_ID	LAST_NAME	SALARY
109	KOHLI	20000
111	JOBSS	20000
101	RAVI	20000
105	EMANUEL	22000
104	JANE	23400.12
110	DAVIES	30000
106	VIJAY	33400.12

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 A  
JOIN (SELECT DEPARTMENT_ID FROM EMPLOYEES WHERE LAST_NAME LIKE '%U%') B  
ON A.DEPARTMENT_ID = B.DEPARTMENT_ID;
```

## OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, and Gallery. The right side of the header shows the user 'Agila shree 15' and a schema dropdown set to 'WKSP\_A515'. The main area has tabs for SQL Commands, Results, Explain, Describe, Saved SQL, and History. The SQL Commands tab shows the executed query:

```
--  
18  SELECT EMPLOYEE_ID, LAST_NAME  
19  FROM EMPLOYEES A  
20  JOIN (SELECT DEPARTMENT_ID FROM EMPLOYEES WHERE LAST_NAME LIKE '%U%') B  
21  ON A.DEPARTMENT_ID = B.DEPARTMENT_ID;  
--
```

The Results tab displays the output in a table:

EMPLOYEE_ID	LAST_NAME
2	UMA
104	JANE
176	EMANUEL
176	EMANUEL
3	PARTHI
4	JANE
103	PARTHI
105	EMANUEL
105	EMANUEL
110	DAVIES

Below the table, a message says 'More than 10 rows available. Increase rows selector to view more rows.' and '10 rows returned in 0.01 seconds'. There is also a 'Download' link.

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 A  
JOIN (SELECT DEPT_ID FROM DEPARTMENT WHERE LOCATION_ID=1700) B  
ON A.DEPARTMENT_ID=B.DEPT_ID;
```

### OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The query is displayed in the command window:

```
29  SELECT LAST_NAME, DEPARTMENT_ID, JOB_ID  
30  FROM EMPLOYEES A  
31  JOIN (SELECT DEPT_ID FROM DEPARTMENT WHERE LOCATION_ID=1700) B  
32  ON A.DEPARTMENT_ID=B.DEPT_ID;
```

The results are shown in a grid:

LAST_NAME	DEPARTMENT_ID	JOB_ID
UMA	50	ST_CLERK
JANE	50	HR_MANAGER
JANE	50	MANAGER
RAVI	90	TESTER
JOBs	90	DEVELOPER

5 rows returned in 0.01 seconds

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

### QUERY:

```
SELECT LAST_NAME, SALARY FROM EMPLOYEES  
WHERE MANAGER_ID IN (SELECT EMPLOYEE_ID FROM EMPLOYEES WHERE MANAGER_ID IS NULL);
```

### OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The query is displayed in the command window:

```
35  SELECT LAST_NAME, SALARY FROM EMPLOYEES  
36  WHERE MANAGER_ID IN (SELECT EMPLOYEE_ID FROM EMPLOYEES WHERE MANAGER_ID IS NULL);  
37
```

The results are shown in a grid:

LAST_NAME	SALARY
VIJAY	33400.12
SIDDHU	40000

2 rows returned in 0.01 seconds

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

### QUERY:

```
SELECT DEPARTMENT_ID, LAST_NAME, JOB_ID FROM EMPLOYEES  
WHERE DEPARTMENT_ID = (SELECT DEPT_ID FROM DEPARTMENT WHERE DEPT_NAME='EXECUTIVE');
```

### OUTPUT:

The screenshot shows the Oracle SQL Workshop interface with the following details:

- Toolbar: APEX, App Builder, SQL Workshop, Team Development, Gallery.
- Search Bar: Search, Schema: WKSP\_A515.
- Query Editor:

```
42  
43  SELECT DEPARTMENT_ID, LAST_NAME, JOB_ID FROM EMPLOYEES  
44  WHERE DEPARTMENT_ID=(SELECT DEPT_ID FROM DEPARTMENT WHERE DEPT_NAME='EXECUTIVE');
```
- Results Tab: Shows the output of the query.
- Output Data:

DEPARTMENT_ID	LAST_NAME	JOB_ID
60	SIDDHU	COO
60	HARSH	CEO
- Message: 2 rows returned in 0.01 seconds.

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

### QUERY:

```
SELECT EMPLOYEE_ID, LAST_NAME, SALARY FROM EMPLOYEES  
WHERE SALARY > (SELECT AVG(SALARY) FROM EMPLOYEES)  
AND DEPARTMENT_ID IN (SELECT DEPARTMENT_ID FROM EMPLOYEES WHERE LAST_NAME LIKE '%U%');
```

### OUTPUT:

The screenshot shows the Oracle SQL Workshop interface with the following details:

- Toolbar: APEX, App Builder, SQL Workshop, Team Development, Gallery.
- Search Bar: Search, Schema: WKSP\_A515.
- Query Editor:

```
46  
47  SELECT EMPLOYEE_ID, LAST_NAME, SALARY FROM EMPLOYEES  
48  WHERE SALARY > (SELECT AVG(SALARY) FROM EMPLOYEES)  
49  AND DEPARTMENT_ID IN (SELECT DEPARTMENT_ID FROM EMPLOYEES WHERE LAST_NAME LIKE '%U%');
```
- Results Tab: Shows the output of the query.
- Output Data:

EMPLOYEE_ID	LAST_NAME	SALARY
104	JANE	23400.12
4	JANE	23400.12
105	EMANUEL	22000
110	DAVIES	30000
114	SIDDHU	40000
115	HARSH	45000
- Message: 6 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

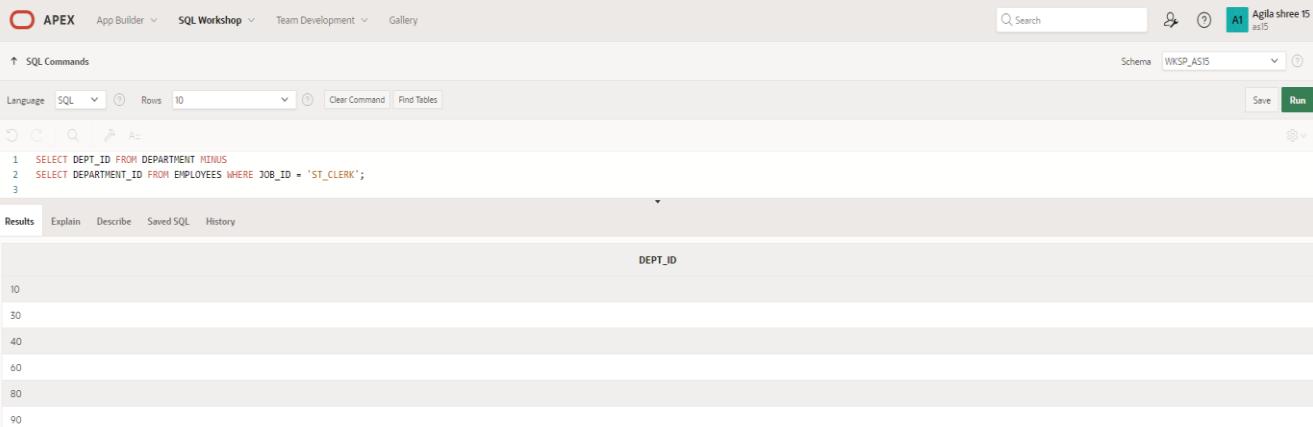
DATE:

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

## QUERY:

```
SELECT DEPT_ID FROM DEPARTMENT MINUS  
SELECT DEPARTMENT_ID FROM EMPLOYEES WHERE JOB_ID = 'ST_CLERK';
```

## OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side of the header shows the session name "Agila shree 15" and workspace "WSP\_AS15". The main area has tabs for Search, Help, and Run. Below that are buttons for Schema, Save, and Run. The SQL Commands tab is selected, showing the following code:

```
1 SELECT DEPT_ID FROM DEPARTMENT MINUS  
2 SELECT DEPARTMENT_ID FROM EMPLOYEES WHERE JOB_ID = 'ST_CLERK';  
3
```

The Results tab is active, displaying the output of the query. The output shows a single column named "DEPT\_ID" with values 10, 30, 40, 60, 80, and 90.

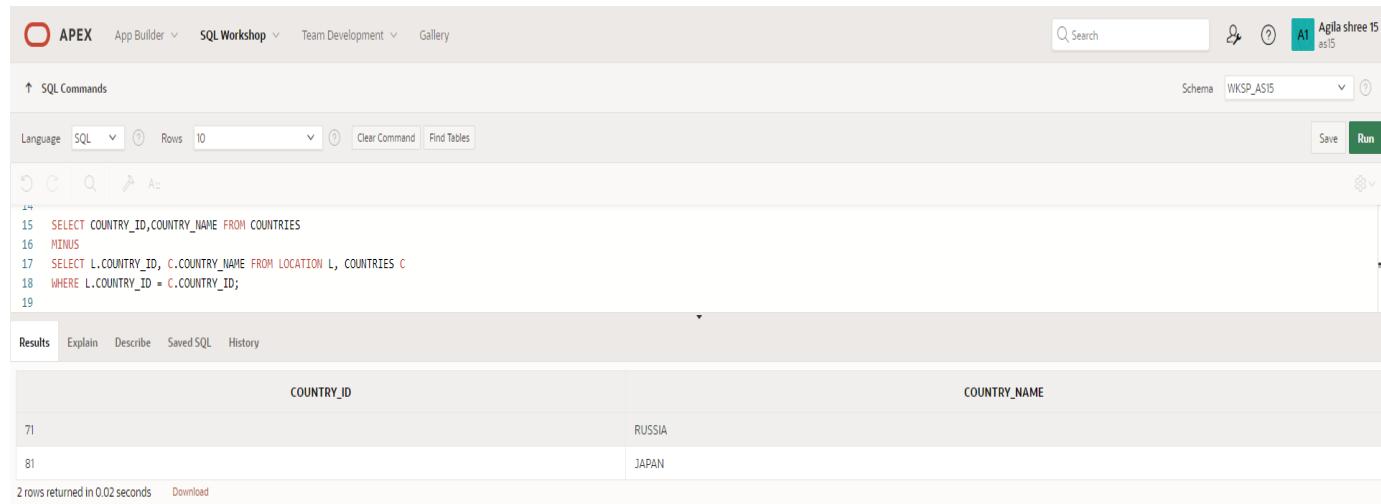
DEPT_ID
10
30
40
60
80
90

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

### QUERY:

```
SELECT COUNTRY_ID,COUNTRY_NAME FROM COUNTRIES  
MINUS  
SELECT L.COUNTRY_ID,C.COUNTRY_NAME FROM LOCATION L, COUNTRIES C  
WHERE L.COUNTRY_ID = C.COUNTRY_ID;
```

### OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there are search, schema dropdown (set to WKSP\_AS15), and user information (Agila shree 15). Below the navigation is a toolbar with buttons for SQL (selected), Rows (set to 10), Clear Command, Find Tables, Save, and Run. The main area contains the SQL code and its results. The code is as follows:

```
15: SELECT COUNTRY_ID,COUNTRY_NAME FROM COUNTRIES  
16: MINUS  
17: SELECT L.COUNTRY_ID, C.COUNTRY_NAME FROM LOCATION L, COUNTRIES C  
18: WHERE L.COUNTRY_ID = C.COUNTRY_ID;  
19:
```

The Results tab is selected, displaying a table with two rows:

COUNTRY_ID	COUNTRY_NAME
71	RUSSIA
81	JAPAN

Below the table, it says "2 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 DISTINCT JOB_ID, DEPARTMENT_ID FROM EMPLOYEES WHERE  
DEPARTMENT_ID = 10  
UNION ALL  
SELECT DISTINCT JOB_ID, DEPARTMENT_ID FROM EMPLOYEES WHERE  
DEPARTMENT_ID = 50  
UNION ALL  
SELECT DISTINCT JOB_ID, DEPARTMENT_ID FROM EMPLOYEES WHERE  
DEPARTMENT_ID = 20;
```

### OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows a search bar, schema dropdown (WKSP\_AS15), and a toolbar with Save and Run buttons. The main area has tabs for SQL Commands, Results, Explain, Describe, Saved SQL, and History. The SQL Commands tab displays the following code:

```
22 SELECT DISTINCT JOB_ID, DEPARTMENT_ID FROM EMPLOYEES WHERE DEPARTMENT_ID = 10  
23 UNION ALL  
24 SELECT DISTINCT JOB_ID, DEPARTMENT_ID FROM EMPLOYEES WHERE DEPARTMENT_ID = 50  
25 UNION ALL  
26 SELECT DISTINCT JOB_ID, DEPARTMENT_ID FROM EMPLOYEES WHERE DEPARTMENT_ID = 20;  
27
```

The Results tab shows the output of the query:

JOB_ID	DEPARTMENT_ID
SL_REP	10
MK_MANAGER	10
ST_CLERK	50
HR_MANAGER	50
MANAGER	50
ST_CLERK	20

Below the table, it says "6 rows returned in 0.02 seconds" and there is a "Download" link.

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

### QUERY:

```
SELECT EMPLOYEE_ID, JOB_ID FROM EMPLOYEES INTERSECT  
SELECT EMPLOYEE_ID, JOB_ID FROM JOB_HISTORY;
```

### OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows a user profile (A1 Agila shree 15) and a search bar. The main area has tabs for SQL Commands, Results (selected), Explain, Describe, Saved SQL, and History. The SQL Commands tab shows the following code:

```
33  SELECT EMPLOYEE_ID, JOB_ID FROM EMPLOYEES INTERSECT  
34  SELECT EMPLOYEE_ID, JOB_ID FROM JOB_HISTORY;
```

The Results tab displays the output of the query:

EMPLOYEE_ID	JOB_ID
100	SL REP
103	ST CLERK

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

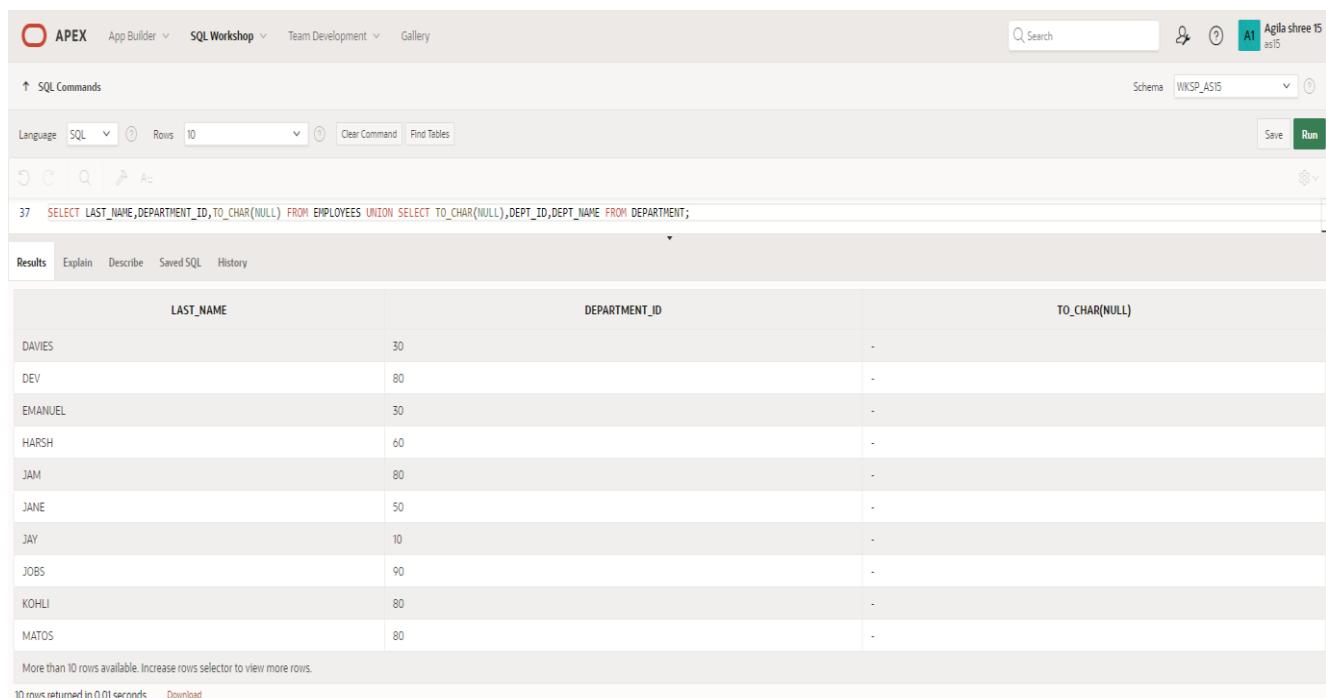
5. The HR department needs a report with the following specifications:

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

### QUERY:

```
SELECT LAST_NAME,DEPARTMENT_ID,TO_CHAR(NULL) FROM EMPLOYEES  
UNION SELECT TO_CHAR(NULL),DEPT_ID,DEPT_NAME FROM DEPARTMENT;
```

### OUTPUT:



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

```
37  SELECT LAST_NAME,DEPARTMENT_ID,TO_CHAR(NULL) FROM EMPLOYEES UNION SELECT TO_CHAR(NULL),DEPT_ID,DEPT_NAME FROM DEPARTMENT;
```

The results are displayed in a grid:

LAST_NAME	DEPARTMENT_ID	TO_CHAR(NULL)
DAVIES	30	-
DEV	80	-
EMANUEL	30	-
HARSH	60	-
JAM	80	-
JANE	50	-
JAY	10	-
JOBS	90	-
KOHLI	80	-
MATOS	80	-

More than 10 rows available. Increase rows selector to view more rows.  
10 rows returned in 0.01 seconds [Download](#)

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

**RESULT:**

# CREATING VIEWS

EX-NO : 11

DATE:

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

## QUERY:

```
CREATE OR REPLACE VIEW EMPLOYEES_VU (EMPLOYEE_ID, EMPLOYEE,
DEPARTMENT_ID) AS SELECT
EMPLOYEE_ID, FIRST_NAME || '' || LAST_NAME, DEPARTMENT_ID FROM
EMPLOYEES;
```

## OUTPUT:



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

```
1 CREATE OR REPLACE VIEW EMPLOYEES_VU (EMPLOYEE_ID, EMPLOYEE, DEPARTMENT_ID) AS SELECT
2 EMPLOYEE_ID, FIRST_NAME || '' || LAST_NAME, DEPARTMENT_ID FROM
3 EMPLOYEES;
4
```

The code is highlighted in red, indicating it is syntax-highlighted. Below the code, the results show:

View created.  
0.01 seconds

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

### QUERY:

```
SELECT*FROM EMPLOYEES_VU;
```

### OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The query `SELECT*FROM EMPLOYEES_VU;` has been run, and the results are displayed in a grid. The columns are labeled `EMPLOYEE_ID`, `EMPLOYEE`, and `DEPARTMENT_ID`. The data includes rows for employees like TARA UMA, MARY JANE, SANA EMANUEL, etc., across various department IDs (50, 30, 20, 10, 40). A note at the bottom indicates that more than 10 rows are available.

EMPLOYEE_ID	EMPLOYEE	DEPARTMENT_ID
2	TARA UMA	50
104	MARY JANE	50
176	SANA EMANUEL	30
3	SANJ PARTHI	20
4	MARY JANE	50
101	ASHA RAVI	10
103	SANJ PARTHI	20
105	SANA EMANUEL	30
106	JOE VIJAY	40
112	RITU RAVI	90

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. The query `SELECT VIEW_NAME,TEXT FROM USER_VIEWS;` has been run, and the results are displayed in a grid. The columns are labeled `VIEW_NAME` and `TEXT`. The data includes three rows: `DEPT50` with the text of a query selecting employees from department 50, `EMPLOYEES_VU` with the text of a query selecting employees by first and last names and department ID, and `SALARY_VU` with a complex query involving multiple tables (E, D, J) to select salary ranges.

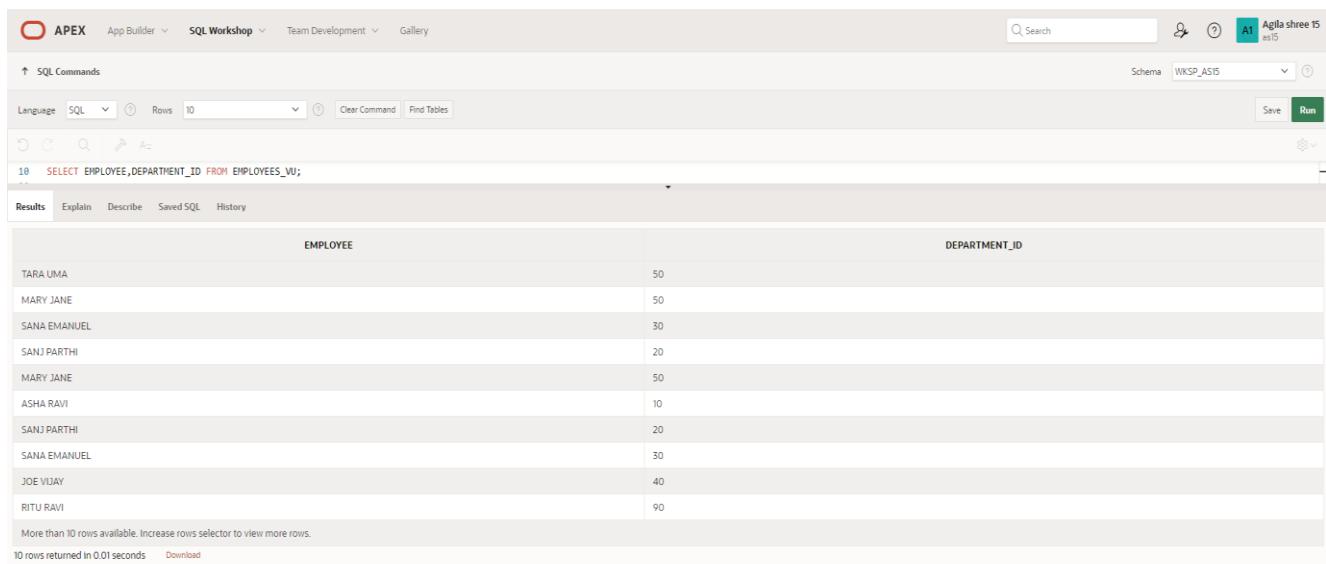
VIEW_NAME	TEXT
DEPT50	SELECT EMPLOYEE_ID, LAST_NAME, DEPARTMENT_ID FROM EMPLOYEES WHERE DEPARTMENT_ID = 50
EMPLOYEES_VU	SELECT EMPLOYEE_ID, FIRST_NAME   ' '   LAST_NAME, DEPARTMENT_ID FROM EMPLOYEES
SALARY_VU	SELECT E.LAST_NAME "EMPLOYEE", D.DEPARTMENT_NAME "DEPARTMENT", E.SALARY "SALARY", J.GRADE_LEVEL "GRADES" FROM EMPLOYEES E, DEPARTMENT D, JOB_GRADES J WHERE E.DEPARTMENT_ID = D.DEPARTMENT_ID AND E.SALARY BETWEEN J.LOWEST_SAL AND J.HIGHEST_SAL

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

## QUERY:

```
SELECT EMPLOYEE,DEPARTMENT_ID FROM EMPLOYEES_VU;
```

## OUTPUT:



EMPLOYEE	DEPARTMENT_ID
TARA UMA	50
MARY JANE	50
SANA EMANUEL	30
SANJ PARTHI	20
MARY JANE	50
ASHA RAVI	10
SANJ PARTHI	20
SANA EMANUEL	30
JOE VIJAY	40
RITU RAVI	90

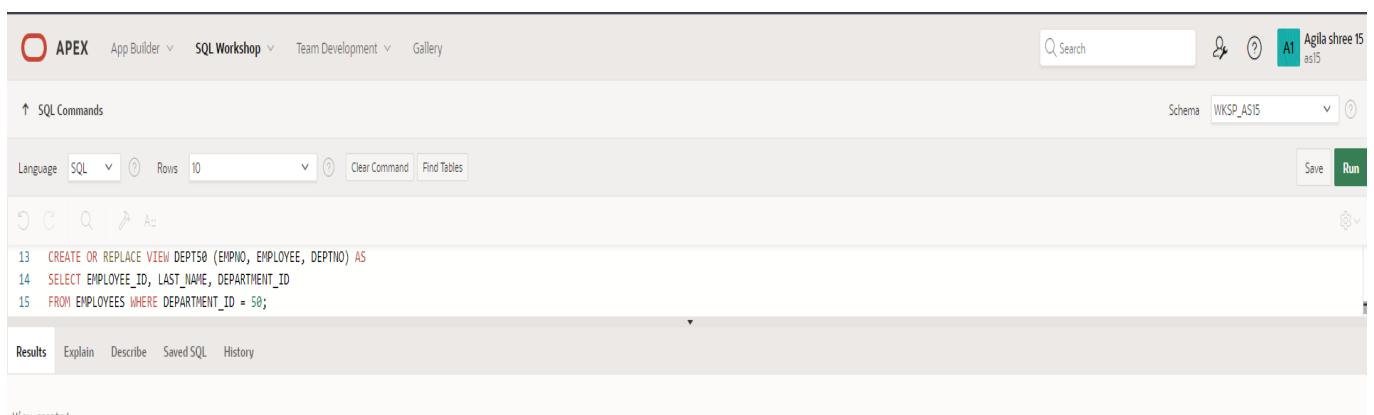
More than 10 rows available. Increase rows selector to view more rows.  
10 rows returned in 0.01 seconds Download

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 OR REPLACE VIEW DEPT50 (EMPNO, EMPLOYEE, DEPTNO) AS  
SELECT EMPLOYEE_ID, LAST_NAME, DEPARTMENT_ID  
FROM EMPLOYEES WHERE DEPARTMENT_ID = 50;
```

## OUTPUT:



View created.

## 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 (selected), Team Development, and Gallery. The right side shows a user profile for 'Agila shree 15' and a schema dropdown set to 'WKSP\_ASIS'. The main area displays the results of the DESCRIBE command for the DEPT50 view. The results tab is selected, showing the following table structure:

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
DEPT50	EMPNO	NUMBER	-	6	0	-	-	-	-
	EMPLOYEE	VARCHAR2	25	-	-	-	-	-	-
	DEPTNO	NUMBER	-	4	0	-	✓	-	-

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 'Agila shree 15' and a schema dropdown set to 'WKSP\_ASIS'. The main area displays the results of the SELECT\* command for the DEPT50 view. The results tab is selected, showing the following data:

EMPNO	EMPLOYEE	DEPTNO
2	UMA	50
104	JANE	50
4	JANE	50

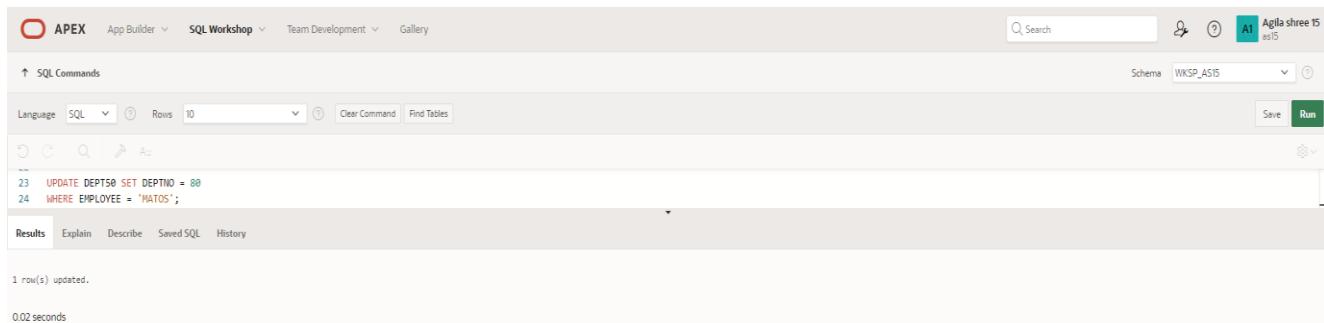
At the bottom left, it says '3 rows returned in 0.01 seconds'.

7. Attempt to reassign Matos to department 80.

### QUERY:

```
UPDATE DEPT50 SET DEPTNO = 80  
WHERE EMPLOYEE = 'MATOS';
```

### OUTPUT:



A screenshot of the Oracle SQL Workshop interface. The top navigation bar shows 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. The right side of the header includes a search bar, user profile 'Agila shree 15 as15', and a 'Run' button. The main area has tabs for 'SQL Commands', 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History'. The 'Results' tab is selected. The SQL command entered is:  
--  
23 UPDATE DEPT50 SET DEPTNO = 80  
24 WHERE EMPLOYEE = 'MATOS';  
The output shows '1 row(s) updated.' and a execution time of '0.02 seconds'.

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.DEPARTMENT_NAME "DEPARTMENT",  
E.SALARY "SALARY", J.GRADE_LEVEL "GRADES"  
FROM EMPLOYEES E, DEPARTMENT D, JOB_GRADES J  
WHERE E.DEPARTMENT_ID = D.DEPARTMENT_ID  
AND E.SALARY BETWEEN J.LOWEST_SAL AND J.HIGHEST_SAL;
```

### OUTPUT:



A screenshot of the Oracle SQL Workshop interface, similar to the previous one. The top navigation bar shows 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. The right side of the header includes a search bar, user profile 'Agila shree 15 as15', and a 'Run' button. The main area has tabs for 'SQL Commands', 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History'. The 'Results' tab is selected. The SQL command entered is:  
25  
26 CREATE OR REPLACE VIEW SALARY\_VU AS  
27 SELECT E.LAST\_NAME "EMPLOYEE", D.DEPARTMENT\_NAME "DEPARTMENT",  
28 E.SALARY "SALARY", J.GRADE\_LEVEL "GRADES"  
29 FROM EMPLOYEES E, DEPARTMENT D, JOB\_GRADES J  
30 WHERE E.DEPARTMENT\_ID = D.DEPARTMENT\_ID  
31 AND E.SALARY BETWEEN J.LOWEST\_SAL AND J.HIGHEST\_SAL;  
The output shows 'View created.'

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?

Ans:

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?

Ans:

- 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?

Ans:

- 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.

Ans:

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.

Ans:

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.

Ans:

```
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.

Ans:

Table Created.

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

Ans:

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	

Ans:

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

## **PRIMARY KEY, FOREIGN KEY, AND CHECK CONSTRAINTS**

1. What is the purpose of a

- PRIMARY KEY
- FOREIGN KEY
- CHECK CONSTRAINT

Ans:

**a. PRIMARY KEY**

Uniquely identify each row in table.

**b. FOREIGN KEY**

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

**c. CHECK CONSTRAINT**

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

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

Ans:

animal\_id NUMBER(6) - **PRIMARY KEY**

name VARCHAR2(25)

license\_tag\_number NUMBER(10) - **UNIQUE**

admit\_date DATE -**NOT NULL**

adoption\_id NUMBER(5),

vaccination\_date DATE -**NOT NULL**

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

Ans:

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

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

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

Ans:

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

```
SELECT * FROM animals;
```

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

Ans:

**COLUMN LEVEL STATEMENT:**

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

**TABLE LEVEL STATEMENT:**

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

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

**a. ON DELETE CASCADE**

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

**b. ON DELETE SET NULL**

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

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

Ans:

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

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

**RESULT :**

# EXERCISE 13

## Creating Views

1. What are three uses for a view from a DBA's perspective?
  - Restrict access and display selective columns
  - Reduce complexity of queries from other internal systems. So, providing a way to view same data in a different manner.
  - Let the app code rely on views and allow the internal implementation of tables to be modified later.

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

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

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

Results	Explain	Describe	Saved SQL	History
ID	Song Title	ARTIST		
47	Hurrah for Today	The Jubilant Trio		
49	Lets Celebrate	The Celebrants		

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

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

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

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

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

6. It is company policy that only upper-level management be allowed access to individual employee salaries.

The department managers, however, need to know the minimum, maximum, and average salaries, grouped by department. Use the Oracle database to prepare a view that displays the needed information for department managers.

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

# DML Operations and Views

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

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

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

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

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

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

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

```
SELECT * FROM view_copy_d_songs;
```

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

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

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

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

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

```
SELECT * FROM read_copy_d_cds;
```

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

# Managing Views

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

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

```
SELECT * FROM view_copy_d_songs;
```

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

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

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

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

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

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

```
SELECT empm.last_name, empm.salary, dptmx.department_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;
```

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

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

# **Indexes and Synonyms**

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

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

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

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

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

3. When will an index be created automatically?

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

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

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

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

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

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

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

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

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

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

```
CREATE INDEX d_ptr_last_name_idx  
ON d_partners(LOWER(last_name));
```

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

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

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

10. Drop the synonym that you created in question

**DROP SYNONYM dj\_tracks2;**

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

**RESULT:**

# OTHER DATABASE OBJECTS

EX\_NO:14

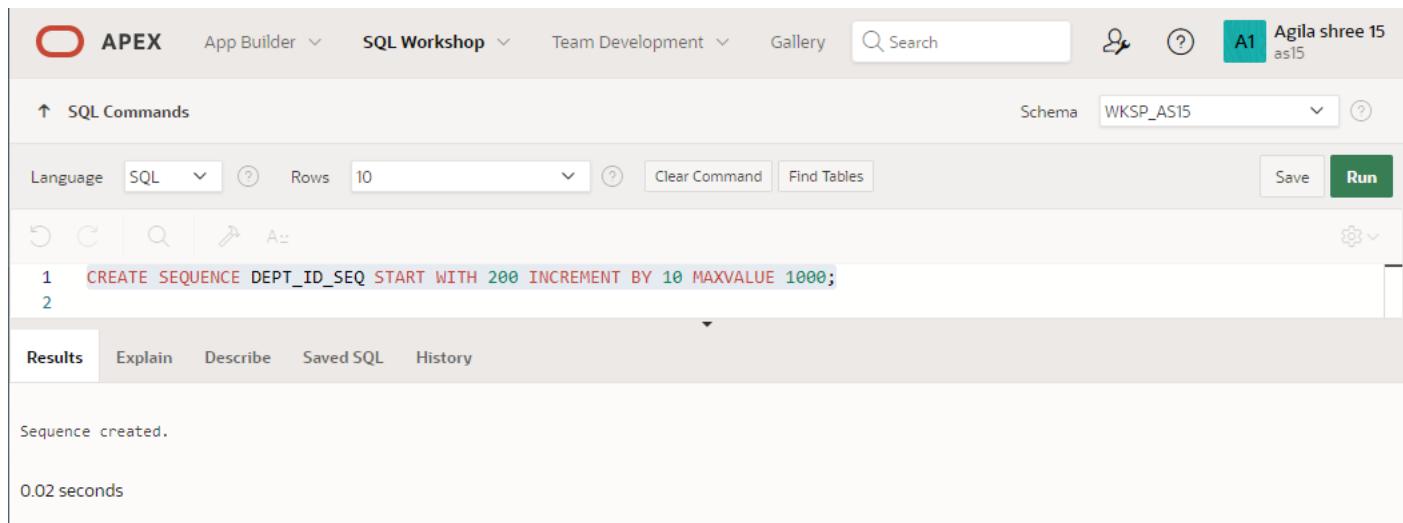
DATE:

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

QUERY:

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

OUTPUT:



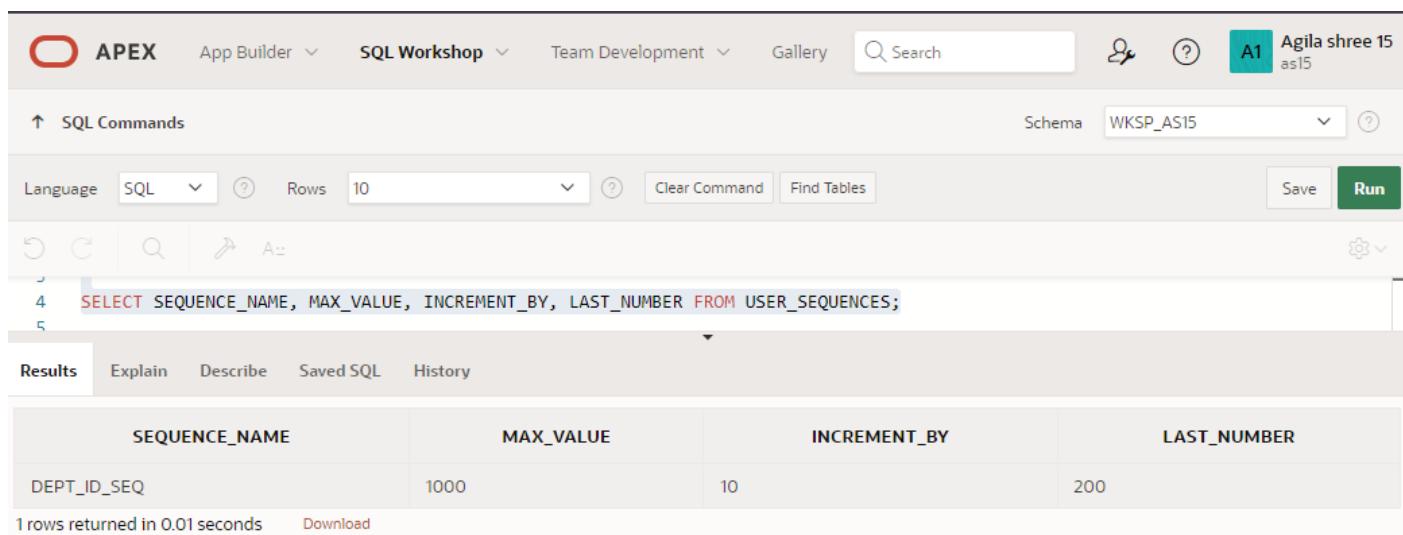
The screenshot shows the Oracle SQL Workshop interface. In the top navigation bar, 'APEX' is selected. The main area is titled 'SQL Commands'. A red box highlights the command: 'CREATE SEQUENCE DEPT\_ID\_SEQ START WITH 200 INCREMENT BY 10 MAXVALUE 1000;'. Below the command, the output shows 'Sequence 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. In the top navigation bar, 'APEX' is selected. The main area is titled 'SQL Commands'. A red box highlights the command: 'SELECT sequence\_name, max\_value, increment\_by, last\_number FROM user\_sequences;'. Below the command, the output shows a table with one row of data:

SEQUENCE_NAME	MAX_VALUE	INCREMENT_BY	LAST_NUMBER
DEPT_ID_SEQ	1000	10	200

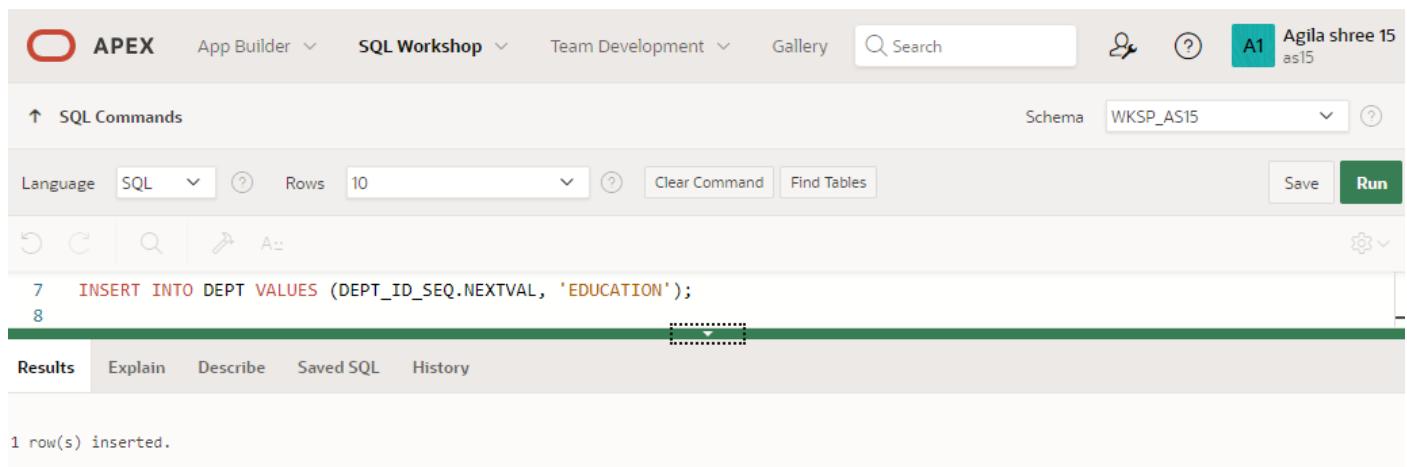
At the bottom, it says '1 rows returned in 0.01 seconds'.

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');
```

**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 'A1 Agila shree 15 as15'. Below the navigation is a toolbar with icons for undo, redo, search, and run. The main area is titled 'SQL Commands' with a schema dropdown set to 'WKSP\_AS15'. The SQL editor contains the following code:

```
7  INSERT INTO DEPT VALUES (DEPT_ID_SEQ.NEXTVAL, 'EDUCATION');
8
```

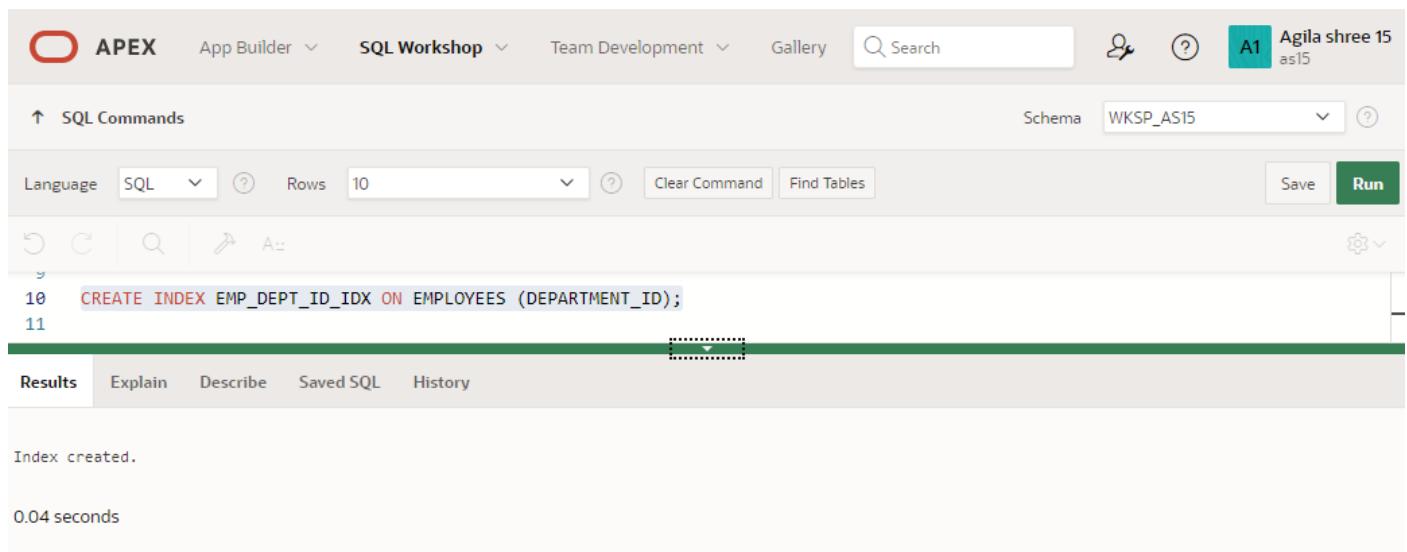
The 'Results' tab is selected, showing the output: '1 row(s) inserted.'

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', 'Gallery', a search bar, and a user profile 'A1 Agila shree 15 as15'. Below the navigation is a toolbar with icons for undo, redo, search, and run. The main area is titled 'SQL Commands' with a schema dropdown set to 'WKSP\_AS15'. The SQL editor contains the following code:

```
10 CREATE INDEX EMP_DEPT_ID_IDX ON EMPLOYEES (DEPARTMENT_ID);
11
```

The 'Results' tab is selected, showing the output: 'Index created.' and '0.04 seconds'.

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

**QUERY:**

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

**OUTPUT:**

The screenshot shows the Oracle SQL Workshop interface. The query `SELECT INDEX\_NAME, TABLE\_NAME,UNIQUENESS FROM USER\_INDEXES WHERE TABLE\_NAME='EMPLOYEES';` has been run and returned one row. The results table has columns: INDEX\_NAME, TABLE\_NAME, and UNIQUENESS. The single row shows `EMP\_DEPT\_ID\_IDX` as the index name, `EMPLOYEES` as the table name, and `NONUNIQUE` as the uniqueness type. The results were returned in 0.07 seconds.

INDEX_NAME	TABLE_NAME	UNIQUENESS
EMP_DEPT_ID_IDX	EMPLOYEES	NONUNIQUE

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

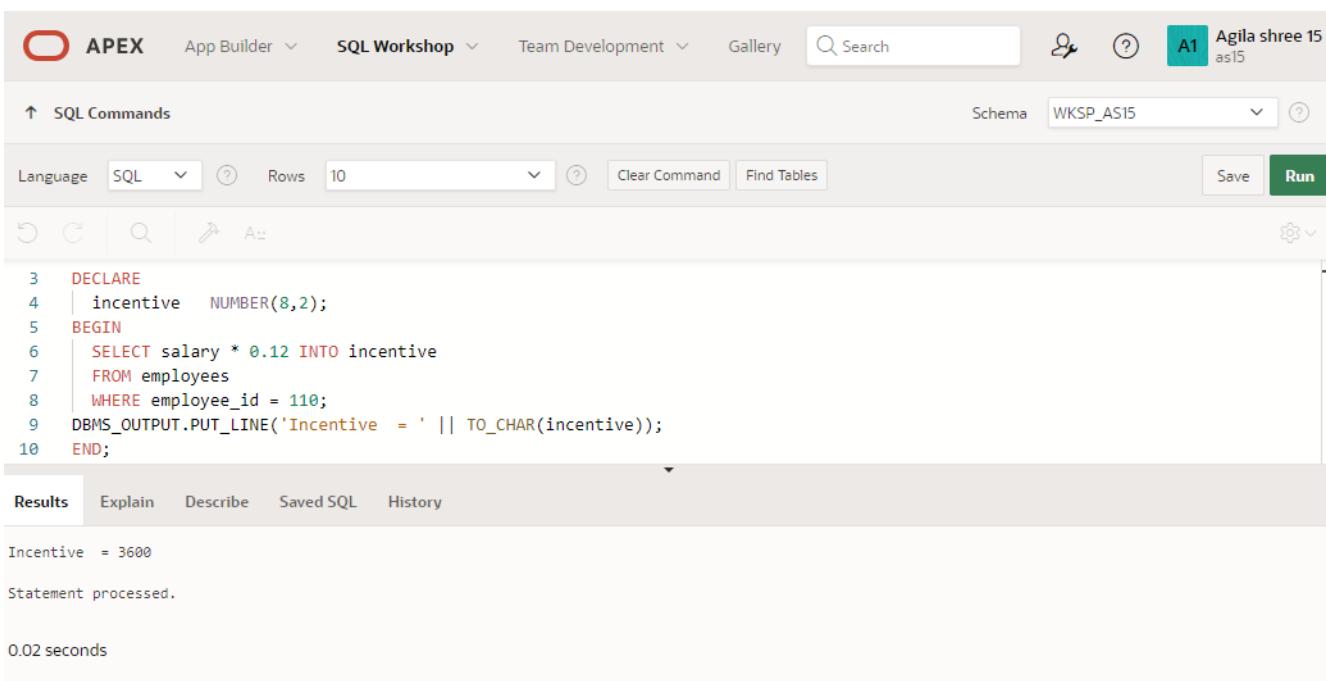
DATE:

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 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 schema WKSP\_AS15, as indicated by the A1 button. The main workspace is titled 'SQL Commands' and contains the following PL/SQL code:

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

The code is executed, and the results panel displays the output:

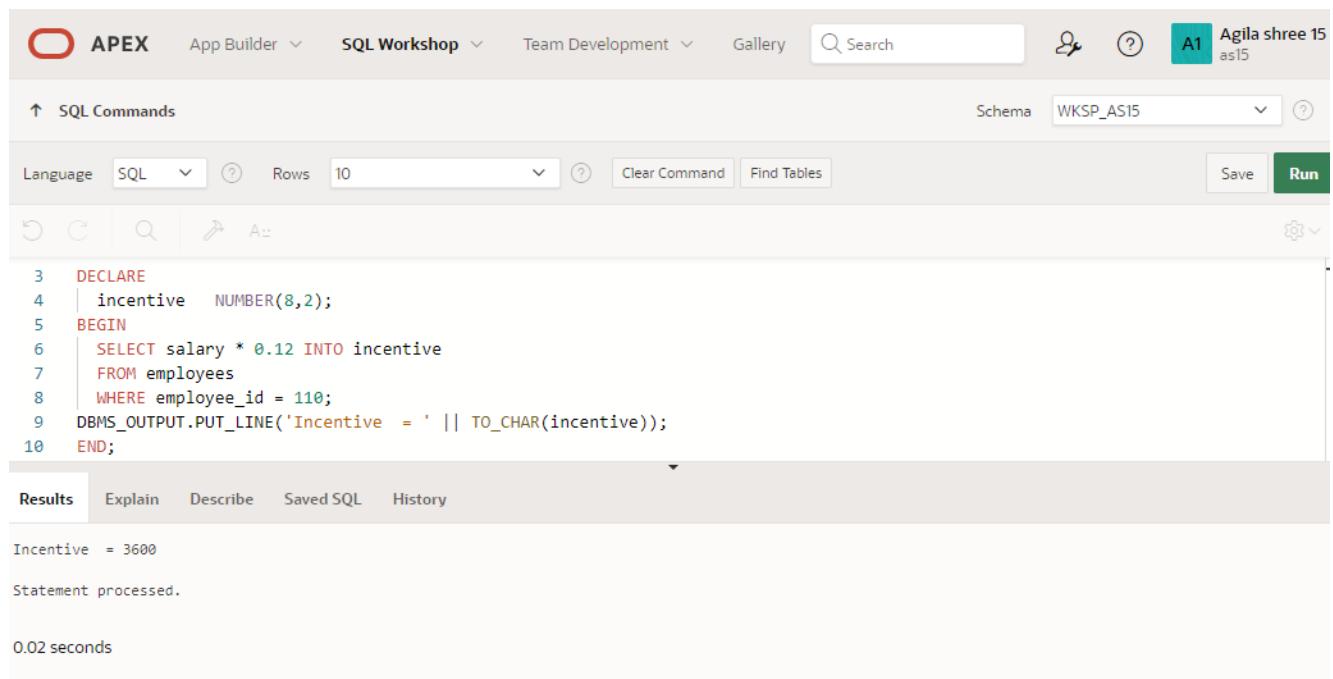
```
Incentive = 3600
Statement processed.
0.02 seconds
```

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

### QUERY:

```
DECLARE
    "WELCOME" varchar2(10) := 'welcome'; -- identifier with quotation
BEGIN
    DBMS_Output.Put_Line(WELCOME); --reference to the identifier without quotation
END;
```

### 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 A1 Agila shree 15. The main workspace is titled 'SQL Commands' and shows the following PL/SQL code:

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

The code is executed, and the results panel displays the output:

```
Incentive  = 3600
Statement processed.

0.02 seconds
```

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

**QUERY:**

DECLARE

```
v_employee_id NUMBER := 122;  
v_new_salary NUMBER;  
  
BEGIN  
    SELECT salary INTO v_new_salary  
    FROM employees  
    WHERE employee_id = v_employee_id;  
    v_new_salary := v_new_salary * 1.1;  
    UPDATE employees  
    SET salary = v_new_salary  
    WHERE employee_id = v_employee_id;  
    COMMIT;
```

DBMS\_OUTPUT.PUT\_LINE('Salary of employee ' || v\_employee\_id || ' has been adjusted.');

EXCEPTION

```
WHEN NO_DATA_FOUND THEN  
    DBMS_OUTPUT.PUT_LINE('Employee with ID ' || v_employee_id || ' not found.');//  
WHEN OTHERS THEN  
    DBMS_OUTPUT.PUT_LINE('An error occurred: ' || SQLERRM);  
    ROLLBACK;
```

END;

**OUTPUT:**

The screenshot shows the Oracle SQL Workshop interface. In the top navigation bar, the tabs are "APEX", "App Builder", "SQL Workshop", "Team Development", and "Gallery". A search bar is present, along with user information "A1 Agila shree 15 as15". The main area is titled "SQL Commands" and contains the following PL/SQL code:

```

25  DECLARE
26      v_employee_id NUMBER := 122;
27      v_new_salary NUMBER;
28  BEGIN
29      -- Retrieve current salary
30      SELECT salary INTO v_new_salary
31      FROM employees
32      WHERE employee_id = v_employee_id;
33
34      -- Adjust the salary by 10%
35      v_new_salary := v_new_salary * 1.1;
36
37      -- Update the salary
38      UPDATE employees
39          SET salary = v_new_salary
40          WHERE employee_id = v_employee_id;
41
42      -- Commit the transaction
43      COMMIT;
44
45      -- Display success message
46      DBMS_OUTPUT.PUT_LINE('Salary of employee ' || v_employee_id || ' has been adjusted.');
47  EXCEPTION
48      WHEN NO_DATA_FOUND THEN
49          DBMS_OUTPUT.PUT_LINE('Employee with ID ' || v_employee_id || ' not found.');
50      WHEN OTHERS THEN
51          DBMS_OUTPUT.PUT_LINE('An error occurred: ' || SQLERRM);
52          ROLLBACK;
53  END;

```

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

```

Employee with ID 122 not found.
1 row(s) updated.

```

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;
/
DECLARE
    PROCEDURE pri_not_m (
        m BOOLEAN
    ) IS
    BEGIN
        pri_bool ('m', m);
        pri_bool ('NOT m', NOT m);
    END pri_not_m;
BEGIN
    DBMS_OUTPUT.PUT_LINE('----- FOR m TRUE -----');
    pri_not_m (TRUE);
    DBMS_OUTPUT.PUT_LINE('----- FOR m FALSE -----');
    pri_not_m (FALSE);

```

END;  
OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, and Gallery. The SQL Workshop tab is selected. The schema dropdown is set to WRGP\_ADS. The main area displays a PL/SQL procedure named pr1\_bool. The code uses DBMS\_OUTPUT.PUT\_LINE to print boolean values based on input parameters. The results section shows the output for m = TRUE, m = FALSE, m = NULL, and m = .NULL.. The bottom status bar indicates 1 statement processed.

```
60 CREATE OR REPLACE PROCEDURE pr1_bool(
61   boo_name  VARCHAR2,
62   boo_val   BOOLEAN
63 ) IS
64 BEGIN
65   IF boo_val IS NULL THEN
66     DBMS_OUTPUT.PUT_LINE( boo_name || ' = NULL');
67   ELSIF boo_val = TRUE THEN
68     DBMS_OUTPUT.PUT_LINE( boo_name || ' = TRUE');
69   ELSE
70     DBMS_OUTPUT.PUT_LINE( boo_name || ' = FALSE');
71   END IF;
72 END;
73 /
74
75 DECLARE
76   PROCEDURE pri_not_m (
77     m  BOOLEAN
78   ) IS
79   BEGIN
80     pri_bool ('m', m);
81     pri_bool ('NOT m', NOT m);
82   END pri_not_m;
83
84 BEGIN
85   DBMS_OUTPUT.PUT_LINE('----- FOR m TRUE -----');
86   pri_not_m (TRUE);
87   DBMS_OUTPUT.PUT_LINE('----- FOR m FALSE -----');
88   pri_not_m (FALSE);
89   DBMS_OUTPUT.PUT_LINE('----- FOR m NULL -----');
90   pri_not_m (NULL);
91 END;
```

Results Explain Describe Saved SQL History

```
m = TRUE ----- FOR m TRUE -----
m = FALSE ----- FOR m FALSE -----
m = FALSE
NOT m = TRUE
----- FOR m NULL -----
m = NULL
NOT m = NULL

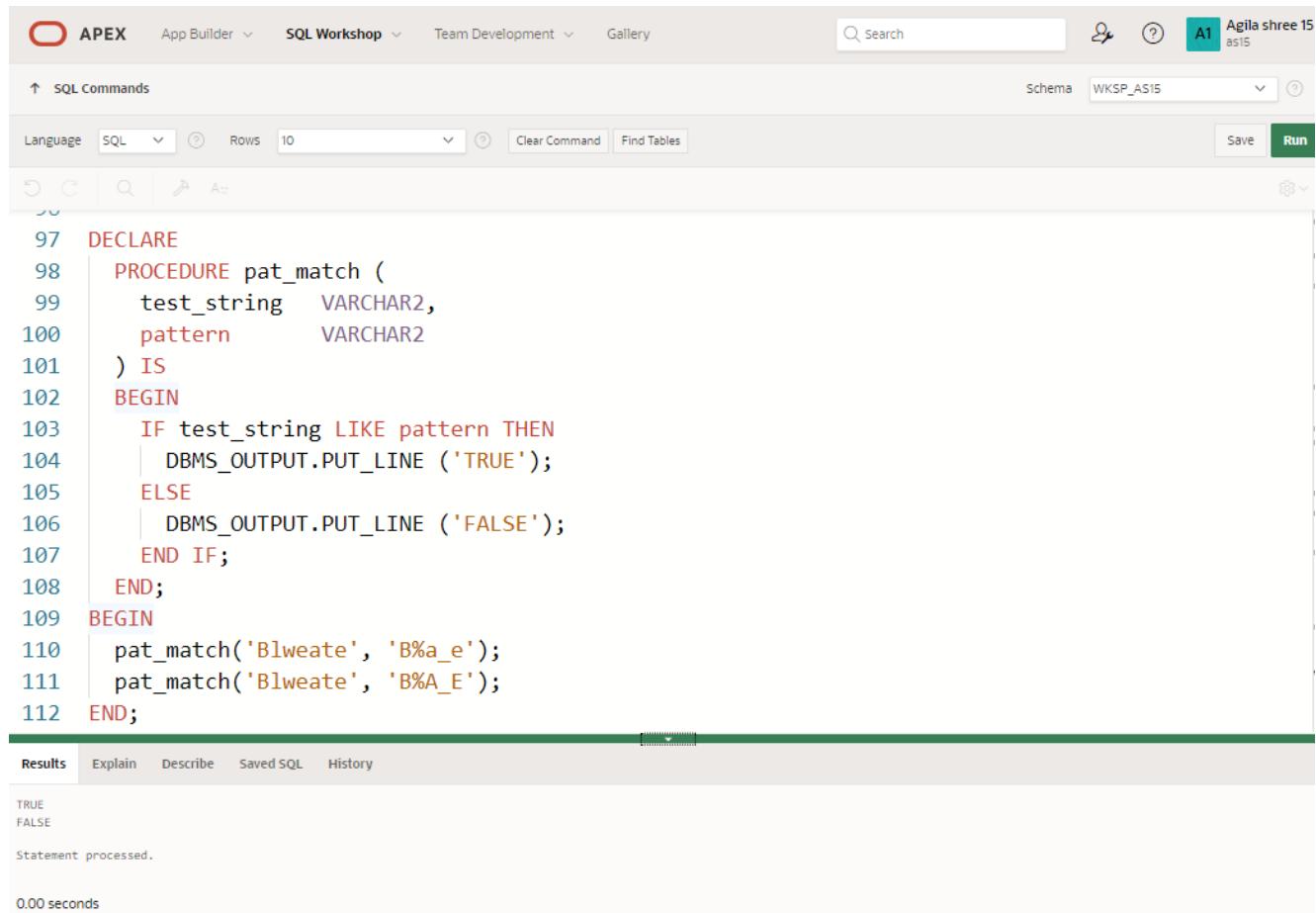
statement processed.
```

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

### QUERY:

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

### OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows a user profile for 'Agila shree 15' (as15). The main area is titled 'SQL Commands' with a schema dropdown set to 'WKSP\_AS15'. The code editor displays the PL/SQL block provided above. The code is numbered from 97 to 112. The output pane at the bottom shows the results of the execution: 'TRUE' and 'FALSE' on separate lines, followed by the message 'Statement processed.' and a timestamp of '0.00 seconds'.

```
97  DECLARE
98  PROCEDURE pat_match (
99    test_string  VARCHAR2,
100   pattern      VARCHAR2
101  ) IS
102 BEGIN
103   IF test_string LIKE pattern THEN
104     DBMS_OUTPUT.PUT_LINE ('TRUE');
105   ELSE
106     DBMS_OUTPUT.PUT_LINE ('FALSE');
107   END IF;
108 END;
109 BEGIN
110   pat_match('Blweate', 'B%a_e');
111   pat_match('Blweate', 'B%A_E');
112 END;
```

Results	Explain	Describe	Saved SQL	History
TRUE FALSE Statement processed. 0.00 seconds				

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

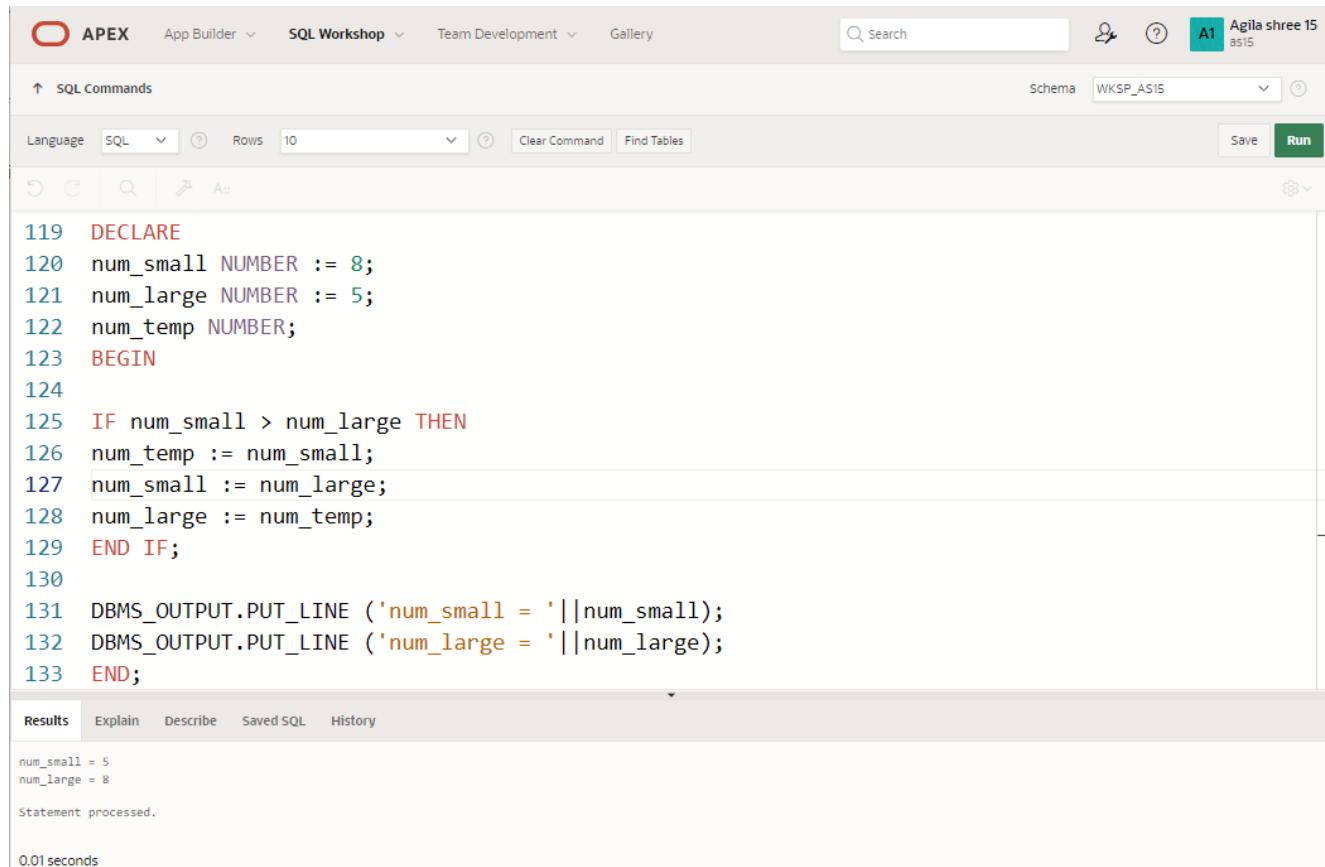
## QUERY:

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

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

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

## OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows a user profile for 'Agila shree 15' (as15). The main area displays a PL/SQL block with line numbers 119 through 133. Lines 119-133 show the declaration of variables and the execution of the PL/SQL block. The bottom pane shows the results of the execution, displaying the output of the DBMS\_OUTPUT.PUT\_LINE statements: 'num\_small = 5' and 'num\_large = 8'. It also indicates that the statement was processed in 0.01 seconds.

```
119  DECLARE
120  num_small NUMBER := 8;
121  num_large NUMBER := 5;
122  num_temp NUMBER;
123  BEGIN
124
125  IF num_small > num_large THEN
126  num_temp := num_small;
127  num_small := num_large;
128  num_large := num_temp;
129  END IF;
130
131  DBMS_OUTPUT.PUT_LINE ('num_small ='||num_small);
132  DBMS_OUTPUT.PUT_LINE ('num_large ='||num_large);
133  END;
```

Results Explain Describe Saved SQL History

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

0.01 seconds
```

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

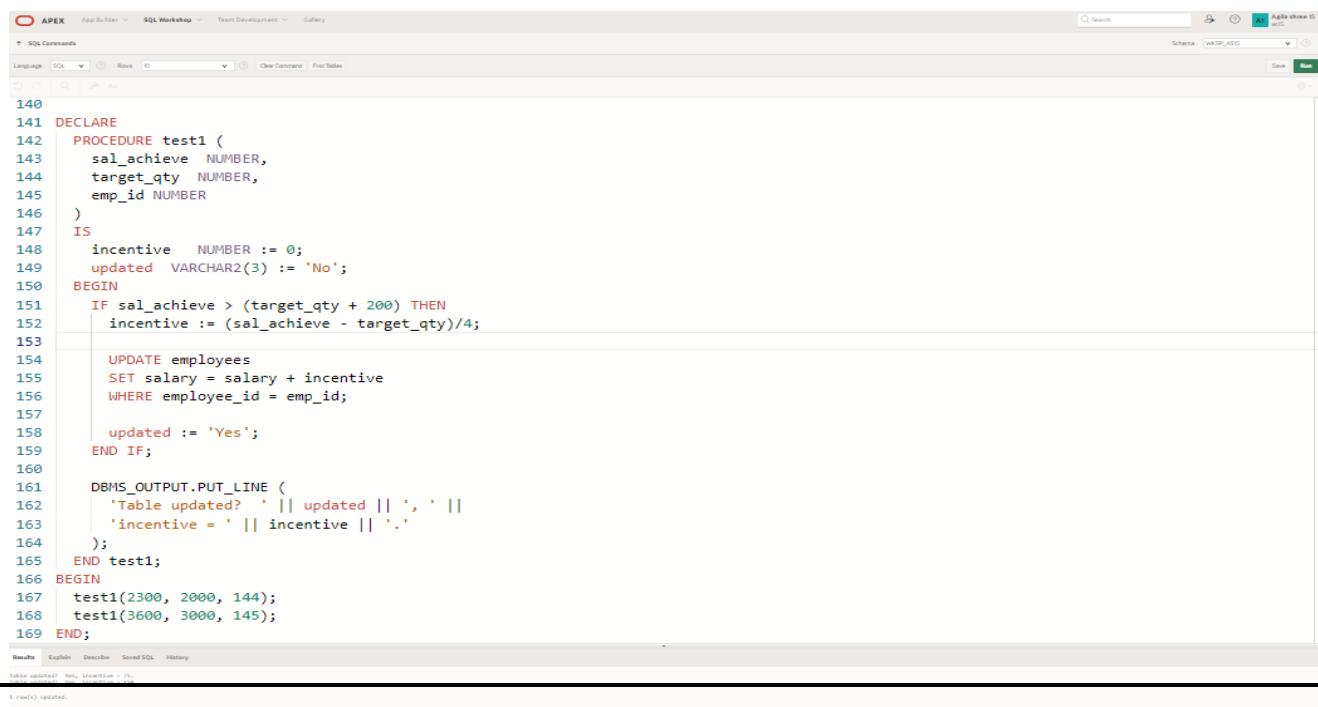
## QUERY:

```
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:



```
140
141  DECLARE
142    PROCEDURE test1 (
143      sal_achieve NUMBER,
144      target_qty NUMBER,
145      emp_id NUMBER
146    )
147    IS
148      incentive NUMBER := 0;
149      updated VARCHAR2(3) := 'No';
150    BEGIN
151      IF sal_achieve > (target_qty + 200) THEN
152        incentive := (sal_achieve - target_qty)/4;

153        UPDATE employees
154        SET salary = salary + incentive
155        WHERE employee_id = emp_id;

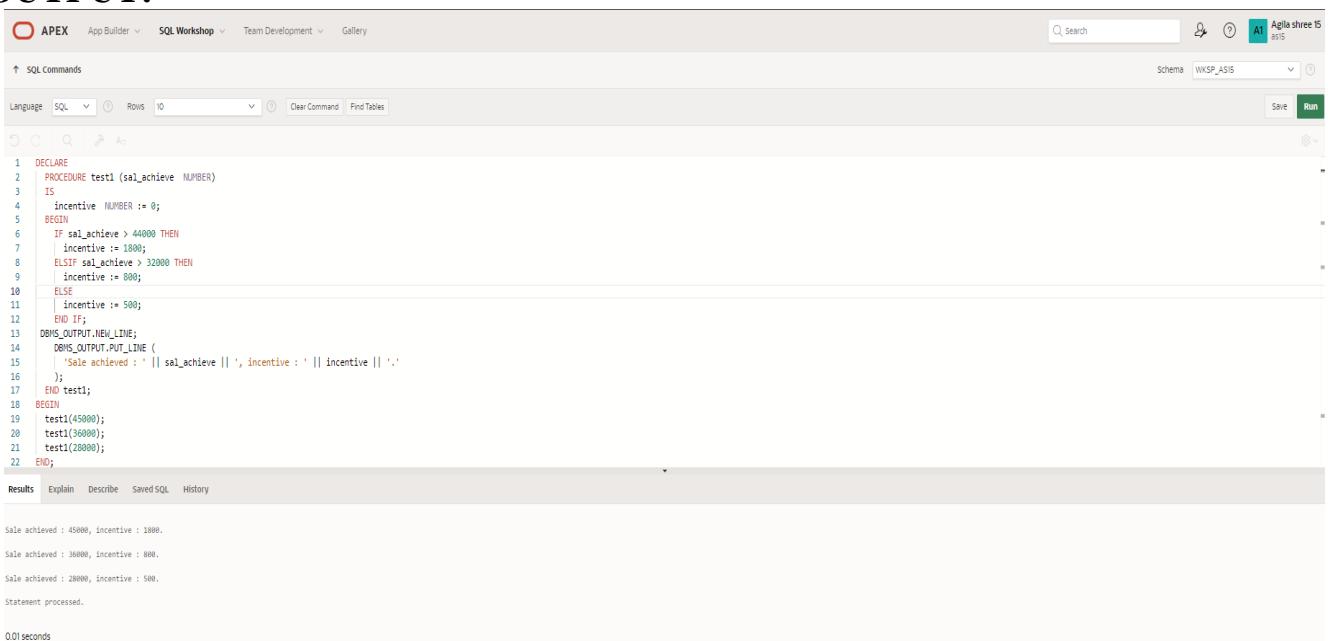
156        updated := 'Yes';
157      END IF;
158
159      DBMS_OUTPUT.PUT_LINE (
160        'Table updated? ' || updated || ', '
161        'incentive = ' || incentive || '.'
162      );
163    END test1;
164  BEGIN
165    test1(2300, 2000, 144);
166    test1(3600, 3000, 145);
167  END;
```

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

### QUERY:

```
DECLARE
  PROCEDURE test1 (sal_achieve NUMBER)
  IS
    incentive NUMBER := 0;
  BEGIN
    IF sal_achieve > 44000 THEN
      incentive := 1800;
    ELSIF sal_achieve > 32000 THEN
      incentive := 800;
    ELSE
      incentive := 500;
    END IF;
    DBMS_OUTPUT.NEW_LINE;
    DBMS_OUTPUT.PUT_LINE (
      'Sale achieved : ' || sal_achieve || ', incentive : ' || incentive || '');
  END test1;
BEGIN
  test1(45000);
  test1(36000);
  test1(28000);
END;
```

### OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, and Gallery. The SQL Workshop tab is selected. The schema dropdown is set to WKSP\_A515. The main area displays the PL/SQL code for the test1 procedure and its execution. The results pane at the bottom shows the output of the procedure for three different sales amounts: 45000, 36000, and 28000, with their respective incentives of 1800, 800, and 500.

```
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 (
15     'Sale achieved : ' || sal_achieve || ', incentive : ' || incentive || '');
16 );
17 END test1;
18 BEGIN
19   test1(45000);
20   test1(36000);
21   test1(28000);
22 END;
```

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

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

### QUERY:

DECLARE

```
v_emp_count NUMBER;
v_vacancies NUMBER := 45;
```

BEGIN

```
-- Count the number of employees in department 50
```

```
SELECT COUNT(*)
```

```
INTO v_emp_count
```

```
FROM employees
```

```
WHERE department_id = 50;
```

```
-- Display the number of employees in department 50
```

```
DBMS_OUTPUT.PUT_LINE('Number of employees in department 50: ' || v_emp_count);
```

```
-- Check if there are any vacancies
```

```
IF v_emp_count < v_vacancies THEN
```

```
    DBMS_OUTPUT.PUT_LINE('There are vacancies in department 50.');
```

```
ELSE
```

```
    DBMS_OUTPUT.PUT_LINE('There are no vacancies in department 50.');
```

```
END IF;
```

```
END;
```

### OUTPUT:

```
27 DECLARE
28     v_emp_count NUMBER;
29     v_vacancies NUMBER := 45;
30 BEGIN
31     -- Count the number of employees in department 50
32     SELECT COUNT(*)
33     INTO v_emp_count
34     FROM employees
35     WHERE department_id = 50;
36
37     -- Display the number of employees in department 50
38     DBMS_OUTPUT.PUT_LINE('Number of employees in department 50: ' || v_emp_count);
39
40     -- Check if there are any vacancies
41     IF v_emp_count < v_vacancies THEN
42         DBMS_OUTPUT.PUT_LINE('There are vacancies in department 50.');
43     ELSE
44         DBMS_OUTPUT.PUT_LINE('There are no vacancies in department 50.');
45     END IF;
46 END;
47 /
48
```

Results Explain Describe Saved SQL History

```
Number of employees in department 50: 3
There are vacancies in department 50.

Statement processed.
```

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

```
v_dept_id employees.department_id%TYPE := 50; -- Change this to the desired department ID
```

```
v_dept_count NUMBER;
```

```
v_vacancies NUMBER := 45; -- Change this to the number of vacancies in the department BEGIN
```

```
SELECT COUNT(*)
```

```
INTO v_dept_count
```

```
FROM employees
```

```
WHERE department_id = v_dept_id;
```

```
DBMS_OUTPUT.PUT_LINE('Number of employees in department ' || v_dept_id || ':' || v_dept_count);
```

```
IF v_dept_count < v_vacancies THEN
```

```
    DBMS_OUTPUT.PUT_LINE('There are vacancies in department ' || v_dept_id || '.');
```

```
    DBMS_OUTPUT.PUT_LINE('Number of vacancies: ' || (v_vacancies - v_dept_count));
```

```
ELSE
```

```
    DBMS_OUTPUT.PUT_LINE('There are no vacancies in department ' || v_dept_id || '.');
```

```
END IF;
```

```
END;
```

## OUTPUT:

```
27 DECLARE
28     v_emp_count NUMBER;
29     v_vacancies NUMBER := 45;
30 BEGIN
31     -- Count the number of employees in department 50
32     SELECT COUNT(*)
33     INTO v_emp_count
34     FROM employees
35     WHERE department_id = 50;
36
37     -- Display the number of employees in department 50
38     DBMS_OUTPUT.PUT_LINE('Number of employees in department 50: ' || v_emp_count);
39
40     -- Check if there are any vacancies
41     IF v_emp_count < v_vacancies THEN
42         DBMS_OUTPUT.PUT_LINE('There are vacancies in department 50.');
43     ELSE
44         DBMS_OUTPUT.PUT_LINE('There are no vacancies in department 50.');
45     END IF;
46
47
48
```

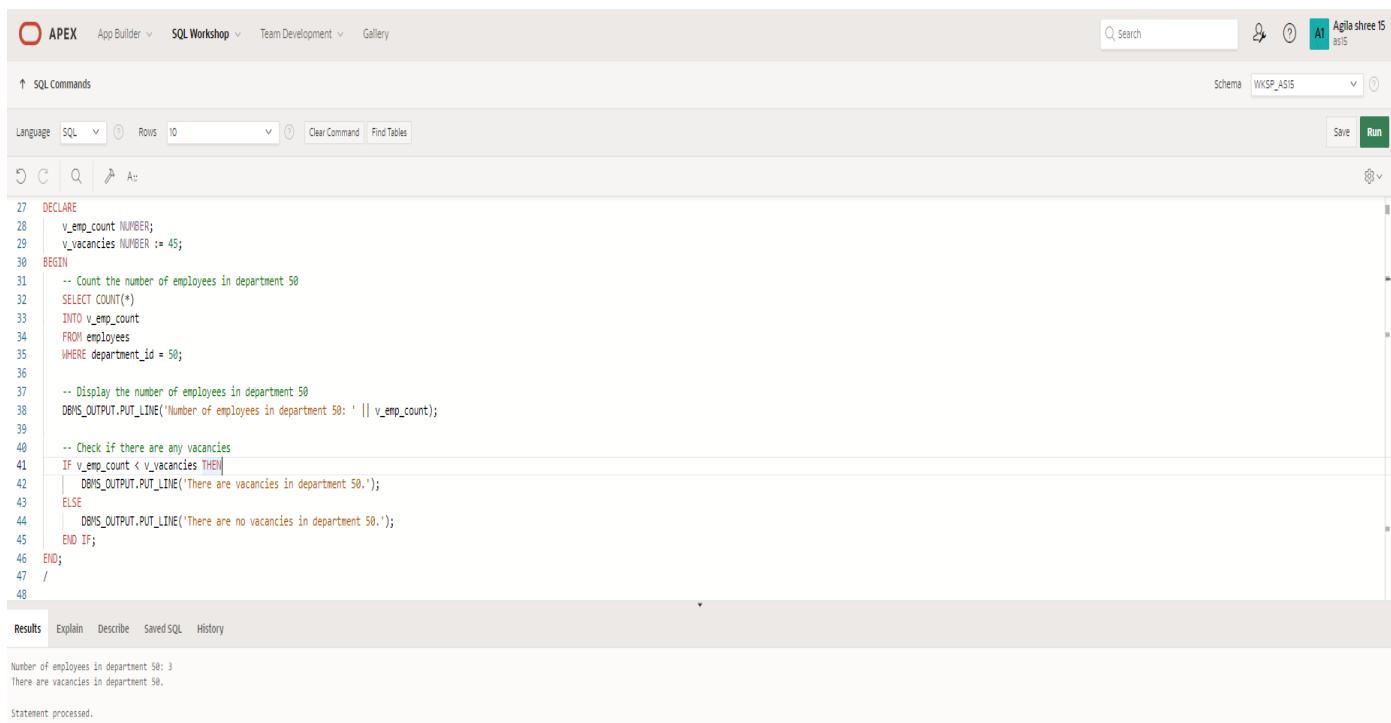
Number of employees in department 50: 3  
There are vacancies in department 50.  
Statement processed.

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

## QUERY:

```
DECLARE
  CURSOR employee_cursor IS
    SELECT employee_id, first_name || ' ' || last_name AS full_name, job_id, hire_date,
salary
      FROM employees;
BEGIN
  -- Loop through the cursor and display employee information
  FOR employee_rec IN employee_cursor LOOP
    DBMS_OUTPUT.PUT_LINE('Employee ID: ' || employee_rec.employee_id);
    DBMS_OUTPUT.PUT_LINE('Employee Name: ' || employee_rec.full_name);
    DBMS_OUTPUT.PUT_LINE('Job Title: ' || employee_rec.job_id);
    DBMS_OUTPUT.PUT_LINE('Hire Date: ' || TO_CHAR(employee_rec.hire_date,
'DD-MON-YYYY'));
    DBMS_OUTPUT.PUT_LINE('Salary: ' || employee_rec.salary);
    DBMS_OUTPUT.PUT_LINE('-----');
  END LOOP;
END;
```

## OUTPUT:



The screenshot shows the Oracle SQL Workshop interface with the following details:

- Toolbar:** Includes APEX, App Builder, SQL Workshop, Team Development, and Gallery.
- Search Bar:** A search field with placeholder text "Search".
- Schema:** Set to "WKSP\_ASIS".
- Run Button:** A green "Run" button.
- Code Area:** Displays the PL/SQL code from the previous section. Lines 27-48 are visible, showing the declaration of variables, opening a cursor, performing a query to count employees in department 50, displaying the result, and checking if there are any vacancies.
- Results Area:** Shows the output of the executed code:

```
Number of employees in department 50: 3
There are vacancies in department 50.

Statement processed.
```

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

### QUERY:

DECLARE

```
CURSOR employee_cursor IS
  SELECT e.employee_id, e.first_name || ' ' || e.last_name AS full_name, d.dept_name
  FROM employees e
  INNER JOIN department d ON e.department_id = d.dept_id;
```

BEGIN

```
-- Loop through the cursor and display employee information
```

```
FOR employee_rec IN employee_cursor LOOP
```

```
  DBMS_OUTPUT.PUT_LINE('Employee ID: ' || employee_rec.employee_id);
  DBMS_OUTPUT.PUT_LINE('Employee Name: ' || employee_rec.full_name);
  DBMS_OUTPUT.PUT_LINE('Department Name: ' || employee_rec.dept_name);
  DBMS_OUTPUT.PUT_LINE('-----');
```

```
END LOOP;
```

END;

### OUTPUT:

```
99
100  DECLARE
101    CURSOR employee_cursor IS
102      SELECT e.employee_id, e.first_name || ' ' || e.last_name AS full_name, d.dept_name
103      FROM employees e
104      INNER JOIN department d ON e.department_id = d.dept_id;
105
106    BEGIN
107      -- Loop through the cursor and display employee information
108      FOR employee_rec IN employee_cursor LOOP
109        DBMS_OUTPUT.PUT_LINE('Employee ID: ' || employee_rec.employee_id);
110        DBMS_OUTPUT.PUT_LINE('Employee Name: ' || employee_rec.full_name);
111        DBMS_OUTPUT.PUT_LINE('Department Name: ' || employee_rec.dept_name);
112        DBMS_OUTPUT.PUT_LINE('-----');
113    END LOOP;
114
115  END;
```

Results Explain Describe Saved SQL History

Employee Name	Job Title	Hire Date	Salary
STEVE JOBS	DEVELOPER	13-JUN-1995	3000
VJ SIDHU	COO	12-JUL-1999	4000
JOHN MATOS	ASST	17-FEB-2000	1000
TARA UMA	ST_CLERK	02-AUG-1999	700
KHAN HARSH	CEO	23-FEB-1998	45000

Statement processed.

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 job_id, MIN(salary) AS min_salary
      FROM employees
      GROUP BY job_id;
```

BEGIN

  -- Loop through the cursor and display job information

  FOR job\_rec IN job\_cursor LOOP

```
    DBMS_OUTPUT.PUT_LINE('Job ID: ' || job_rec.job_id);
```

```
    DBMS_OUTPUT.PUT_LINE('Minimum Salary: ' || job_rec.min_salary);
```

```
    DBMS_OUTPUT.PUT_LINE('-----');
```

  END LOOP;

END;

## OUTPUT:

```
118  DECLARE
119    CURSOR job_cursor IS
120      SELECT job_id, MIN(salary) AS min_salary
121        FROM employees
122       GROUP BY job_id;
123
124  BEGIN
125    -- Loop through the cursor and display job information
126    FOR job_rec IN job_cursor LOOP
127      DBMS_OUTPUT.PUT_LINE('Job ID: ' || job_rec.job_id);
128      DBMS_OUTPUT.PUT_LINE('Minimum Salary: ' || job_rec.min_salary);
129      DBMS_OUTPUT.PUT_LINE('-----');
130    END LOOP;
131  END;
```

Results

Job ID	Title	MinSalary
A_MANAGER	ADVISOR	10000
CE_O	CEO	45000
ASST	ASST	10000
FL_MANAGER	FL_MANAGER	22000
DEVELOPER	DEVELOPER	20000
ST_CLERK	ST_CLERK	7000
MANAGER	MANAGER	23480.12
SUPERVISOR	SUPERVISOR	20000
MK_MANAGER	MK_MANAGER	20000
ENGINEER	ENGINEER	15000
TESTER	TESTER	5500
SL REP	SL REP	10400.12

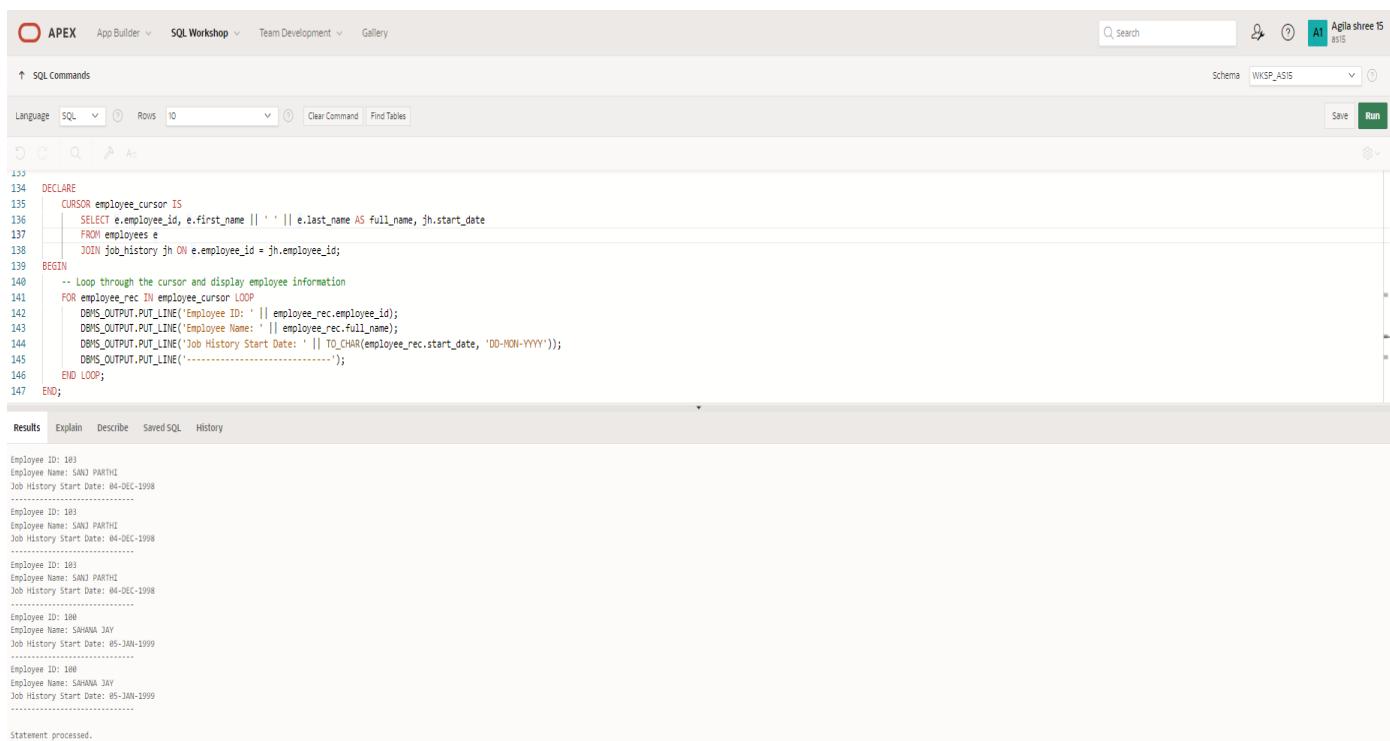
14. Write a PL/SQL program to display the employee IDs, names, and job history start dates of all employees.

## QUERY:

DECLARE

```
CURSOR employee_cursor IS
  SELECT e.employee_id, e.first_name || ' ' || e.last_name AS full_name, jh.start_date
  FROM employees e
  JOIN job_history jh ON e.employee_id = jh.employee_id;
BEGIN
  -- Loop through the cursor and display employee information
  FOR employee_rec IN employee_cursor LOOP
    DBMS_OUTPUT.PUT_LINE('Employee ID: ' || employee_rec.employee_id);
    DBMS_OUTPUT.PUT_LINE('Employee Name: ' || employee_rec.full_name);
    DBMS_OUTPUT.PUT_LINE('Job History Start Date: ' ||
      TO_CHAR(employee_rec.start_date, 'DD-MON-YYYY'));
    DBMS_OUTPUT.PUT_LINE('-----');
  END LOOP;
END;
```

## OUTPUT:



The screenshot shows the Oracle SQL Workshop interface with the following details:

- Toolbar:** APEX, App Builder, SQL Workshop, Team Development, Gallery.
- Search Bar:** Search, AI Agila shree 15.
- Schema:** WKSP\_A515.
- Code Area:** The code block above is pasted here. Lines 133-147 are visible, showing the declaration of the cursor, the select statement, the begin block, the loop, and the end loop.
- Results Area:** The results section displays the output for each employee record. It includes the Employee ID, Employee Name, and Job History Start Date. The output is as follows:

Employee ID	Employee Name	Job History Start Date
101	SANJU PARTHI	04-DEC-1998
102	SANJU PARTHI	04-DEC-1998
103	SANJU PARTHI	04-DEC-1998
104	SAHANA JAY	05-JAN-1999
105	SAHANA JAY	05-JAN-1999

Below the results, a message states "Statement processed."

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

## **QUERY:**

## DECLARE

```
CURSOR employee_cursor IS
  SELECT e.employee_id, e.first_name || ' ' || e.last_name AS full_name, jh.end_date
  FROM employees e
  JOIN job_history jh ON e.employee_id = jh.employee_id;
```

BEGIN

-- Loop through the cursor and display employee information

FOR employee\_rec IN employee\_cursor LOOP

```
DBMS_OUTPUT.PUT_LINE('Employee ID: ' || employee_rec.employee_id);
DBMS_OUTPUT.PUT_LINE('Employee Name: ' || employee_rec.full_name);
```

-- Check if the end date is NULL (meaning the employee is currently in the job)  
IF employee\_rec.end\_date IS NULL THEN

```
DBMS_OUTPUT.PUT_LINE('Job History End Date: (Still Employed)');
```

ELSE

```
DBMS_OUTPUT.PUT_LINE('Job History End Date: ' ||
```

TO\_CHAR(employee\_rec.end\_date, 'DD-MON-YYYY'));

END IF;

```
DBMS_OUTPUT.PUT_LINE('-----');
```

END LOOP;

END;

## **OUTPUT:**

The screenshot shows the Agila shire 15 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 a workspace named 'WKSP\_ASIS'. The main area has tabs for 'Language' (selected), 'SQL', 'Rows 10', 'Clear Command', and 'Find Tables'. A 'Save' and 'Run' button is at the bottom right. The code editor displays the following PL/SQL block:

```
133  
134  DECLARE  
135    CURSOR employee_cursor IS  
136      SELECT e.employee_id, e.first_name || ' ' || e.last_name AS full_name, jh.start_date  
137      FROM employees e  
138      JOIN job_history jh ON e.employee_id = jh.employee_id;  
139  BEGIN  
140    -- Loop through the cursor and display employee information  
141    FOR employee_rec IN employee_cursor LOOP  
142      DBMS_OUTPUT.PUT_LINE('Employee ID: ' || employee_rec.employee_id);  
143      DBMS_OUTPUT.PUT_LINE('Employee Name: ' || employee_rec.full_name);  
144      DBMS_OUTPUT.PUT_LINE('Job History Start Date: ' || TO_CHAR(employee_rec.start_date, 'DD-MON-YYYY'));  
145      DBMS_OUTPUT.PUT_LINE('-----');  
146    END LOOP;  
147  END;
```

The results tab is selected, showing the output of the cursor loop:

Employee ID	Employee Name	Job History Start Date
183	SANJ PARTHI	84-DEC-1998
183	SARAHNA JAY	85-JAN-1999
183	SARAHNA JAY	85-JAN-1999
183	SARAHNA JAY	85-JAN-1999

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

**RESULT :**

# PROCEDURES AND FUNCTIONS

EX. NO: 17

DATE:

1.) Factorial of a number using function.

## QUERY:

DECLARE

    fac NUMBER := 1;

    n NUMBER := :1;

BEGIN

    WHILE n > 0 LOOP

        fac := n \* fac;

        n := n - 1;

    END LOOP;

    DBMS\_OUTPUT.PUT\_LINE(fac);

END;

## OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows a user profile for 'Agila shree 15'. The main area is titled 'SQL Commands' with tabs for Language (SQL selected), Rows (10), and Clear Command. The schema is set to 'WKSP\_ASIS'. The code area contains a PL/SQL block to calculate the factorial of 5, using a function and a procedure. The results tab at the bottom shows the output: 'Factorial of 5 is 120' and 'Statement processed.'

```
1 CREATE OR REPLACE FUNCTION factorial(n IN NUMBER) RETURN NUMBER IS
2     result NUMBER := 1;
3 BEGIN
4     FOR i IN 1..n LOOP
5         result := result * i;
6     END LOOP;
7     RETURN result;
8 END;
9 /
10 DECLARE
11     num NUMBER := 5;
12     fact NUMBER;
13 BEGIN
14     fact := factorial(num);
15     DBMS_OUTPUT.PUT_LINE('Factorial of ' || num || ' is ' || fact);
16 END;
```

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 SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery'. The main area is titled 'SQL Commands' with tabs for 'Language' (set to 'SQL'), 'Rows' (set to '10'), 'Clear Command', and 'Find Tables'. On the right, there are buttons for 'Save', 'Run', and a search bar. The code area contains a PL/SQL block:

```
56  DECLARE
57    v_book_id NUMBER := 1; -- Example book ID
58    v_title VARCHAR2(100);
59    v_author VARCHAR2(100);
60    v_year_published NUMBER;
61  BEGIN
62    -- Call the procedure
63    get_book_info(v_book_id, v_title, v_author, v_year_published);
64
65    -- Display the retrieved information
66    IF v_title IS NOT NULL THEN
67      DBMS_OUTPUT.PUT_LINE('Title: ' || v_title);
68      DBMS_OUTPUT.PUT_LINE('Author: ' || v_author);
69      DBMS_OUTPUT.PUT_LINE('Year Published: ' || v_year_published);
70    ELSE
71      DBMS_OUTPUT.PUT_LINE('Book information could not be retrieved.');
72    END IF;
73  END;
```

The results section shows the output of the query:

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

Statement processed.
```

Execution time: 0.02 seconds

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

## RESULT:

# TRIGGER

EX\_NO: 18

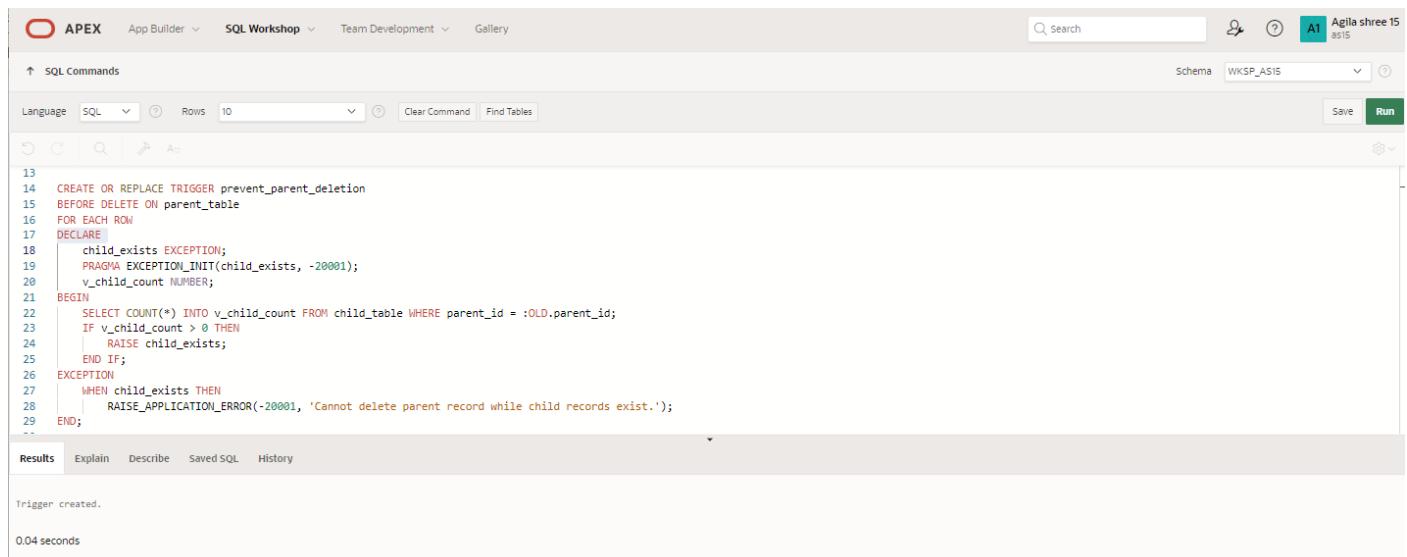
DATE:

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

## QUERY:

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

## OUTPUT:



The screenshot shows the Oracle SQL Workshop interface with the following details:

- Toolbar:** APEX, App Builder, SQL Workshop, Team Development, Gallery.
- Search Bar:** Search (placeholder: Agila shree 15).
- Schema:** WKSP\_ASIS.
- Language:** SQL.
- Results Panel:** Shows the SQL code for the trigger creation.
- Status Bar:** Trigger created. 0.04 seconds.

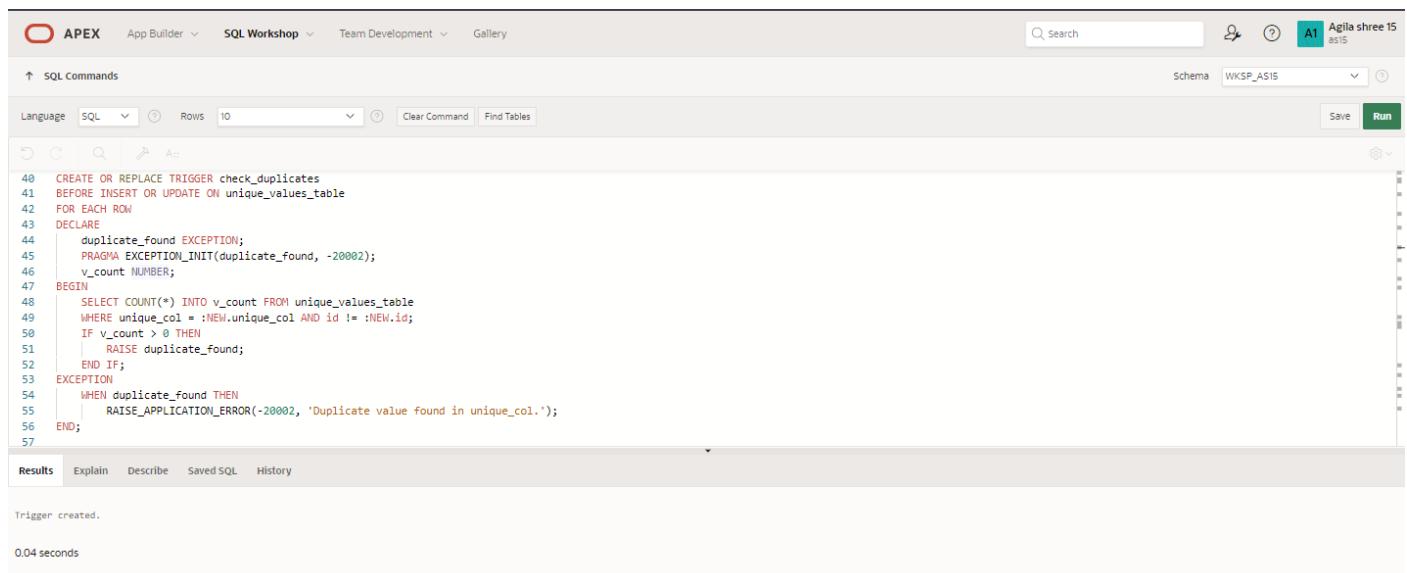
```
13
14 CREATE OR REPLACE TRIGGER prevent_parent_deletion
15 BEFORE DELETE ON parent_table
16 FOR EACH ROW
17 DECLARE
18     child_exists EXCEPTION;
19     PRAGMA EXCEPTION_INIT(child_exists, -20001);
20     v_child_count NUMBER;
21 BEGIN
22     SELECT COUNT(*) INTO v_child_count FROM child_table WHERE parent_id = :OLD.parent_id;
23     IF v_child_count > 0 THEN
24         RAISE child_exists;
25     END IF;
26 EXCEPTION
27     WHEN child_exists THEN
28         RAISE_APPLICATION_ERROR(-20001, 'Cannot delete parent record while child records exist.');
29 END;
```

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

### QUERY:

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

### OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. In the top navigation bar, 'APEX' is selected. Below the toolbar, the schema 'WKSP\_ASIS' is chosen. The main area displays the PL/SQL code for the trigger. The code is highlighted in green and blue, indicating syntax. The output pane at the bottom shows the message 'Trigger created.' and a execution time of '0.04 seconds'.

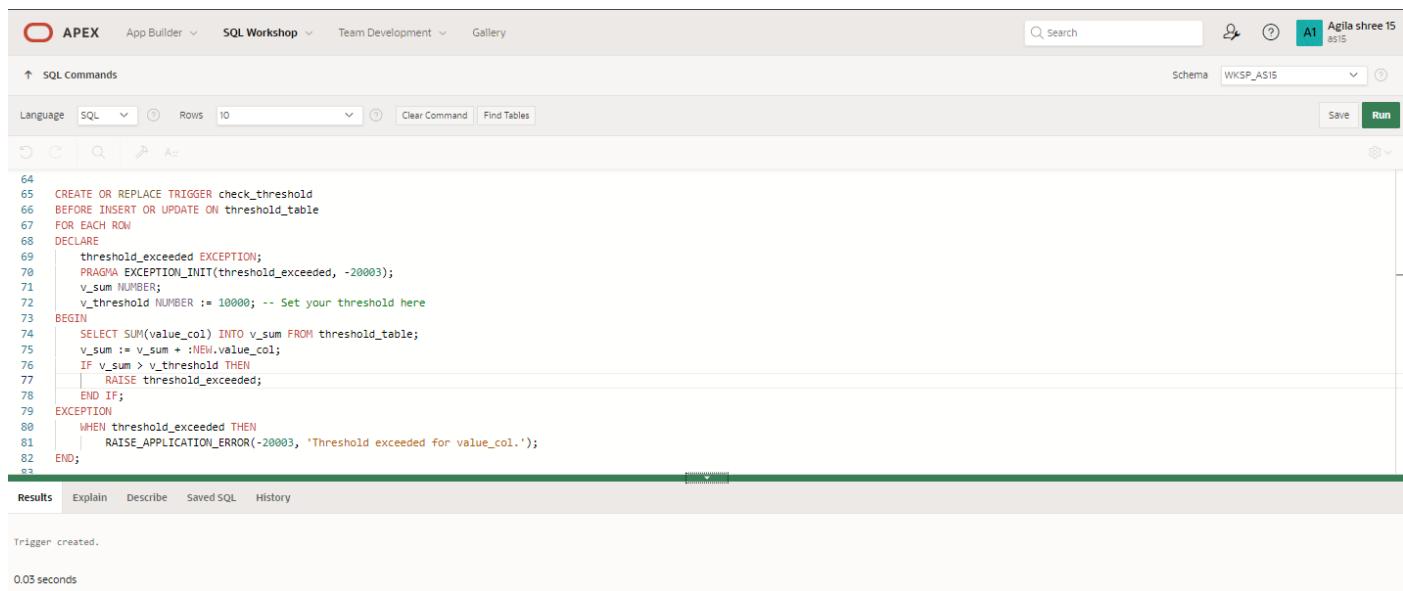
```
40 CREATE OR REPLACE TRIGGER check_duplicates
41 BEFORE INSERT OR UPDATE ON unique_values_table
42 FOR EACH ROW
43 DECLARE
44     duplicate_found EXCEPTION;
45     PRAGMA EXCEPTION_INIT(duplicate_found, -20002);
46     v_count NUMBER;
47 BEGIN
48     SELECT COUNT(*) INTO v_count FROM unique_values_table
49     WHERE unique_col = :NEW.unique_col AND id != :NEW.id;
50     IF v_count > 0 THEN
51         RAISE duplicate_found;
52     END IF;
53 EXCEPTION
54     WHEN duplicate_found THEN
55         RAISE_APPLICATION_ERROR(-20002, 'Duplicate value found in unique_col.');
56 END;
57
```

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

#### QUERY:

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

#### OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side of the header shows the user 'Agila shree 15' and the schema 'WKSP\_AS15'. The main area is titled 'SQL Commands' with tabs for Language (SQL selected), Rows (10), and Schema (WKSP\_AS15). Below the tabs are icons for Undo, Redo, Search, Find Tables, Clear Command, and Run. The SQL editor contains the PL/SQL code for the trigger. The code is numbered from 64 to 83. Lines 64-82 show the trigger definition, and line 83 shows the successful creation message: 'Trigger created.' The bottom status bar indicates '0.03 seconds'.

```
64
65 CREATE OR REPLACE TRIGGER check_threshold
66 BEFORE INSERT OR UPDATE ON threshold_table
67 FOR EACH ROW
68 DECLARE
69     threshold_exceeded EXCEPTION;
70     PRAGMA EXCEPTION_INIT(threshold_exceeded, -20003);
71     v_sum NUMBER;
72     v_threshold NUMBER := 10000; -- Set your threshold here
73 BEGIN
74     SELECT SUM(value_col) INTO v_sum FROM threshold_table;
75     v_sum := v_sum + :NEW.value_col;
76     IF v_sum > v_threshold THEN
77         RAISE threshold_exceeded;
78     END IF;
79 EXCEPTION
80     WHEN threshold_exceeded THEN
81         RAISE_APPLICATION_ERROR(-20003, 'Threshold exceeded for value_col.');
82 END;
83
```

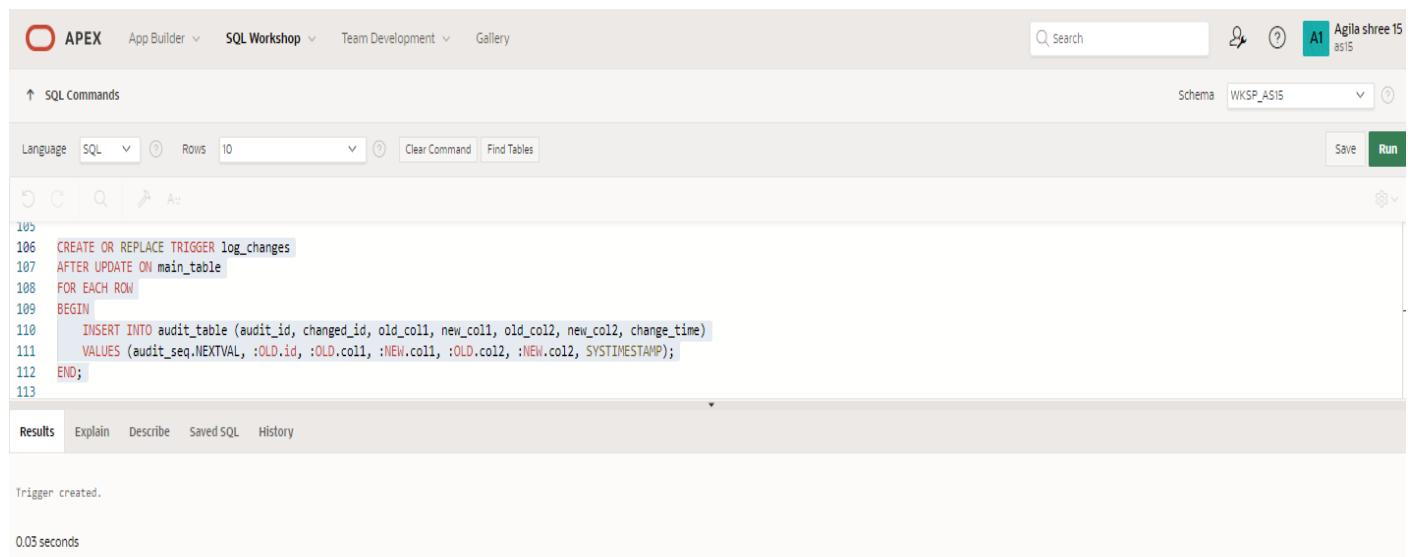
Trigger created.  
0.03 seconds

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

### QUERY:

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

### OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows a user profile for Agila shree 15. The main area is titled 'SQL Commands' with tabs for Language (SQL selected), Rows (10), Clear Command, Find Tables, Save, and Run. The code editor displays the PL/SQL trigger creation script. The output pane at the bottom shows the message 'Trigger created.' and a execution time of '0.05 seconds'.

```
105
106 CREATE OR REPLACE TRIGGER log_changes
107 AFTER UPDATE ON main_table
108 FOR EACH ROW
109 BEGIN
110     INSERT INTO audit_table (audit_id, changed_id, old_col1, new_col1, old_col2, new_col2, change_time)
111     VALUES (audit_seq.NEXTVAL, :OLD.id, :OLD.col1, :NEW.col1, :OLD.col2, :NEW.col2, SYSTIMESTAMP);
112 END;
113
```

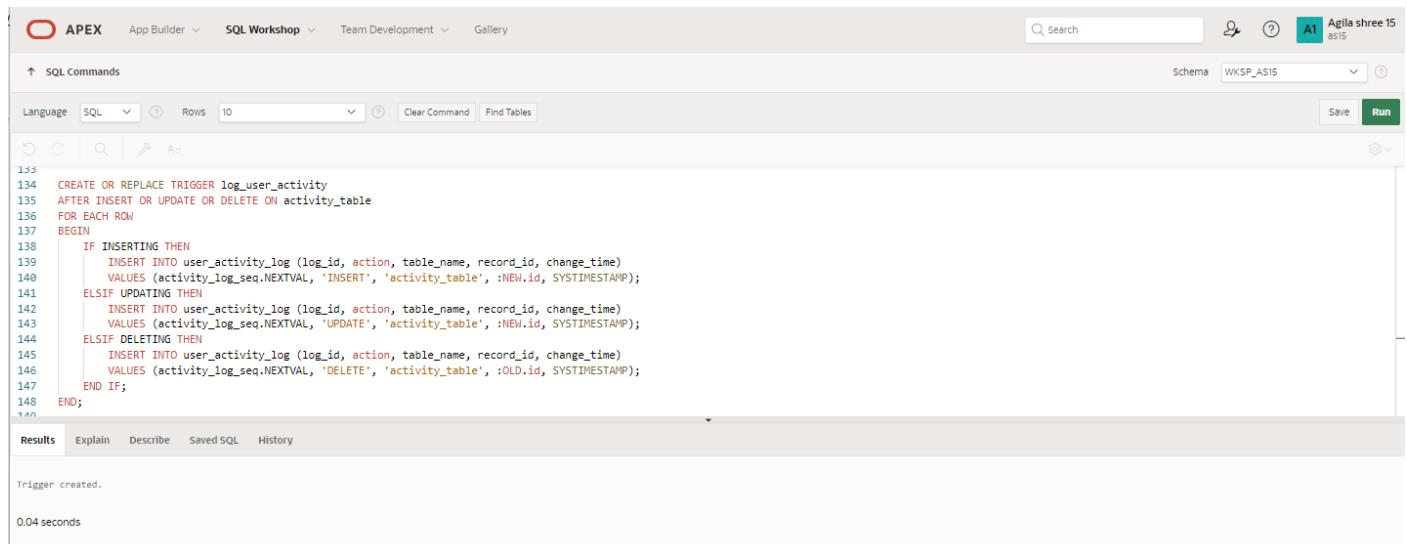
Trigger created.  
0.05 seconds

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

### QUERY:

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

### OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, and Gallery. The SQL Workshop tab is selected. The main area displays the PL/SQL code for the trigger. The code is highlighted in red and black, indicating syntax. The output pane at the bottom shows the message "Trigger created." and "0.04 seconds".

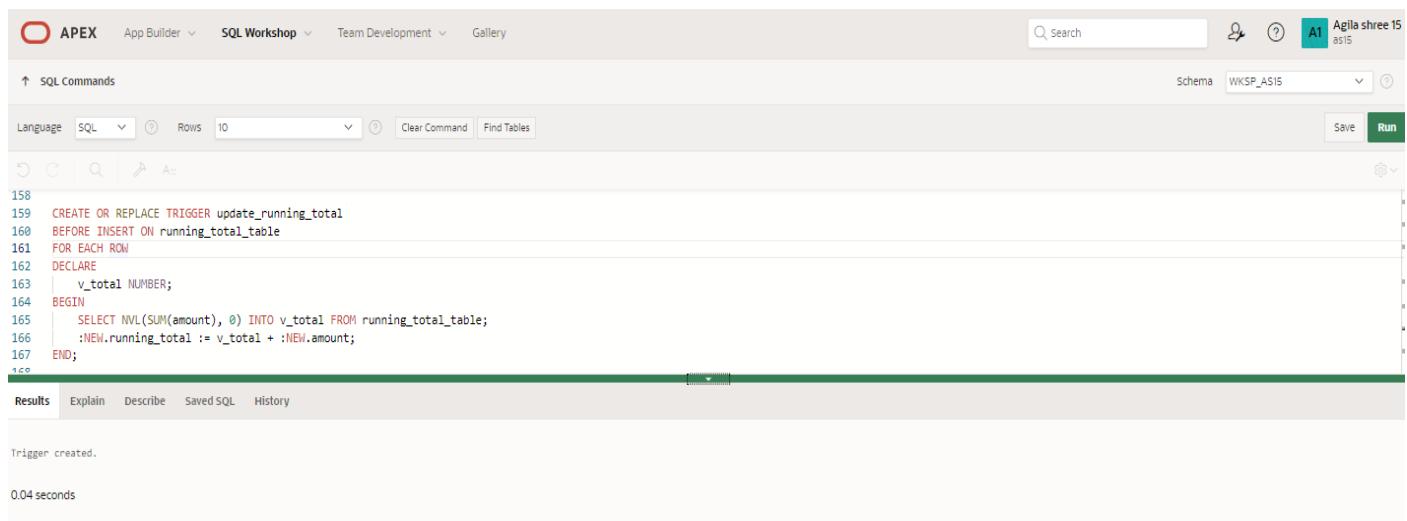
```
134 CREATE OR REPLACE TRIGGER log_user_activity
135   AFTER INSERT OR UPDATE OR DELETE ON activity_table
136   FOR EACH ROW
137 BEGIN
138   IF INSERTING THEN
139     INSERT INTO user_activity_log (log_id, action, table_name, record_id, change_time)
140     VALUES (activity_log_seq.NEXTVAL, 'INSERT', 'activity_table', :NEW.id, SYSTIMESTAMP);
141   ELSIF UPDATING THEN
142     INSERT INTO user_activity_log (log_id, action, table_name, record_id, change_time)
143     VALUES (activity_log_seq.NEXTVAL, 'UPDATE', 'activity_table', :NEW.id, SYSTIMESTAMP);
144   ELSIF DELETING THEN
145     INSERT INTO user_activity_log (log_id, action, table_name, record_id, change_time)
146     VALUES (activity_log_seq.NEXTVAL, 'DELETE', 'activity_table', :OLD.id, SYSTIMESTAMP);
147   END IF;
148 END;
```

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

### QUERY:

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

### OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. In the top navigation bar, 'APEX' is selected. The main area displays the SQL command for creating the trigger, which is identical to the one provided in the question. The code is numbered from 158 to 167. Below the code, the 'Results' tab is active, showing the message 'Trigger created.' and a execution time of '0.04 seconds'. The schema is set to 'WKSP\_AS15'.

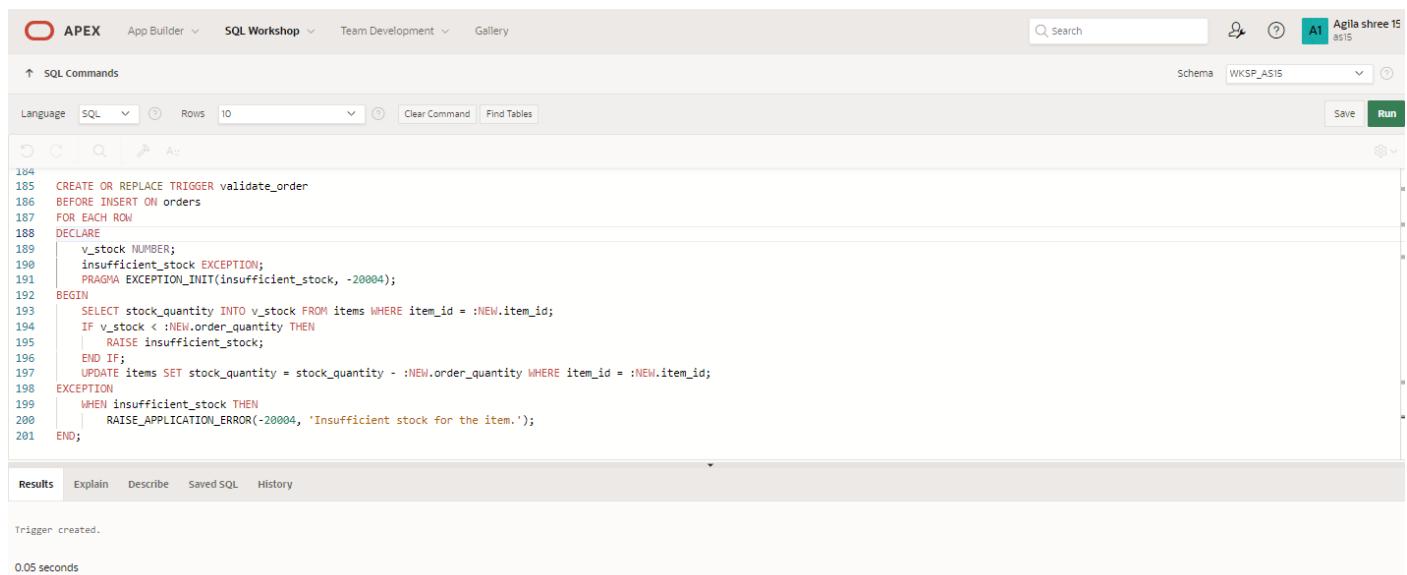
```
158
159  CREATE OR REPLACE TRIGGER update_running_total
160  BEFORE INSERT ON running_total_table
161  FOR EACH ROW
162  DECLARE
163      v_total NUMBER;
164  BEGIN
165      SELECT NVL(SUM(amount), 0) INTO v_total FROM running_total_table;
166      :NEW.running_total := v_total + :NEW.amount;
167  END;
168
```

7.) Write a code in PL/SQL to create a trigger that validates the availability of items before allowing an order to be placed, considering stock levels and pending orders

### QUERY:

```
CREATE OR REPLACE TRIGGER validate_order
BEFORE INSERT ON orders
FOR EACH ROW
DECLARE
    v_stock NUMBER;
    insufficient_stock EXCEPTION;
    PRAGMA EXCEPTION_INIT(insufficient_stock, -20004);
BEGIN
    SELECT stock_quantity INTO v_stock FROM items WHERE item_id = :NEW.item_id;
    IF v_stock < :NEW.order_quantity THEN
        RAISE insufficient_stock;
    END IF;
    UPDATE items SET stock_quantity = stock_quantity - :NEW.order_quantity WHERE
item_id = :NEW.item_id;
EXCEPTION
    WHEN insufficient_stock THEN
        RAISE_APPLICATION_ERROR(-20004, 'Insufficient stock for the item.');
END;
```

### OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. 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 of the header, there's a search bar, a refresh icon, and a user profile labeled 'A1 Agila shree 15 asis'. The main workspace is titled 'SQL Commands'. It contains the PL/SQL code for the 'validate\_order' trigger. The code is numbered from 184 to 201. The 'Results' tab is selected at the bottom, showing the message 'Trigger created.' and a execution time of '0.05 seconds'.

```
184 CREATE OR REPLACE TRIGGER validate_order
185 BEFORE INSERT ON orders
186 FOR EACH ROW
187 DECLARE
188     v_stock NUMBER;
189     insufficient_stock EXCEPTION;
190     PRAGMA EXCEPTION_INIT(insufficient_stock, -20004);
191 BEGIN
192     SELECT stock_quantity INTO v_stock FROM items WHERE item_id = :NEW.item_id;
193     IF v_stock < :NEW.order_quantity THEN
194         RAISE insufficient_stock;
195     END IF;
196     UPDATE items SET stock_quantity = stock_quantity - :NEW.order_quantity WHERE item_id = :NEW.item_id;
197 EXCEPTION
198     WHEN insufficient_stock THEN
199         RAISE_APPLICATION_ERROR(-20004, 'Insufficient stock for the item.');
200 END;
```

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

## **RESULT:**

# MONGO DB

EX\_NO: 19

DATE:

1.) Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which prepared dish except 'American' and 'Chinees' or restaurant's name begins with letter 'Wil'.

## QUERY:

```
db.restaurants.find(
{
  $or: [
    { name: /^Wil/ },
    { cuisine: { $nin: ['American', 'Chinese'] } }
  ]
},
{
  restaurant_id: 1,
  name: 1,
  borough: 1,
  cuisine: 1
}
);
```

## OUTPUT:

```
[mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000
Agila_Shree_015> db.restaurants.find({ $or: [ { name: /^Wil/ }, { cuisine: { $nin: ['American', 'Chinese'] } } ] }, { restaurant_id: 1, name: 1, borough: 1, cuisine: 1 });
[
  {
    _id: ObjectId('564c2d949eb21ad392f1d6de'),
    borough: 'Manhattan',
    cuisine: 'Other',
    name: '',
    restaurant_id: '50017887'
  },
  {
    _id: ObjectId('564c2d949eb21ad392f1d6ed'),
    borough: 'Brooklyn',
    cuisine: 'Other',
    name: '',
    restaurant_id: '50017910'
  },
  {
    _id: ObjectId('564c2d949eb21ad392f1d6ed'),
    borough: 'Manhattan',
    cuisine: 'Other',
    name: '',
    restaurant_id: '50017912'
  },
  {
    _id: ObjectId('564c2d949eb21ad392f1d6f5'),
    borough: 'Brooklyn',
    cuisine: 'Other',
    name: '',
    restaurant_id: '50017925'
  }
]
```

2.) Write a MongoDB query to find the restaurant Id, name, and grades for those restaurants which achieved a grade of "A" and scored 11 on an ISODate "2014-08- 11T00:00:00Z" among many of survey dates.

QUERY:

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

OUTPUT:

```
[mongosh mongoDB://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000] Agila_Shree_015> db.restaurants.find( { grades: { $elemMatch: { grade: "A", score: 11, date: ISODate("2014-08-11T00:00:00Z") } } }, { restaurant_id: 1, name: 1, grades: 1 } ); Agila_Shree_015>
```

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("2014-08-01T00:00:00Z")  
  },  
  {  
    restaurant_id: 1,  
    name: 1,  
    grades: 1  
  }  
);
```

### OUTPUT:

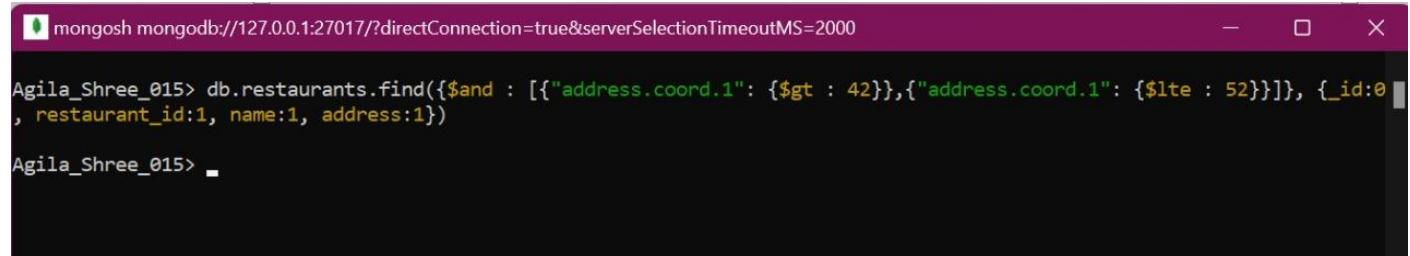
```
[mongosh mongodbs://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000]  
  
Agila_Shree_015> db.restaurants.find(  
...   {  
...     "grades.1.grade": "A",  
...     "grades.1.score": 9,  
...     "grades.1.date": ISODate("2014-08-01T00:00:00Z")  
...   },  
...   {  
...     restaurant_id: 1,  
...     name: 1,  
...     grades: 1  
...   }  
... );  
  
Agila_Shree_015> ■
```

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:



A screenshot of a terminal window titled "mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000". The window shows a command being typed in: "Agila\_Shree\_015> db.restaurants.find({\$and : [{"address.coord.1": {\$gt : 42}}, {"address.coord.1": {\$lte : 52}}]}, {"\_id:0, restaurant\_id:1, name:1, address:1})". The command is partially typed, with the final closing brace and a new line below it.

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:

```
Agila_Shree_015> db.restaurants.find({$and : [{"address.coord.1": {$gt : 42}}, {"address.coord.1": {$lte : 52}}]}, { _id:0, restaurant_id:1, name:1, address:1})  
Agila_Shree_015> db.restaurants.find({}, { _id: 0 }).sort({ name: 1 });  
[  
  {  
    address: {  
      building: '154',  
      coord: [ -73.9189064, 40.8654529 ],  
      street: 'Post Ave',  
      zipcode: '10034'  
    },  
    borough: 'Manhattan',  
    cuisine: 'Other',  
    grades: [],  
    name: '',  
    restaurant_id: '50017887'  
  },  
  {  
    address: {  
      building: '508',  
      coord: [ -73.999813, 40.683876 ],  
      street: 'Henry St',  
      zipcode: '11231'  
    },  
    Agila_Shree_015:  
    cuisine: 'Other',  
    grades: [],  
    name: '',  
    restaurant_id: '50017910'  
  },  
  {  
    address: {  
      building: '15',  
      coord: [ -73.9966882, 40.7139264 ],  
      street: 'Division St',  
      zipcode: '10002'  
    },  
    borough: 'Manhattan',  
  }
```

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:**

```
Agila_Shree_015> db.restaurants.find({}, { _id: 0 }).sort({ name: -1 })
[
  {
    address: {
      building: '154',
      coord: [ -73.9189064, 40.8654529 ],
      street: 'Post Ave',
      zipcode: '10034'
    },
    borough: 'Manhattan',
    cuisine: 'Other',
    grades: [],
    name: '',
    restaurant_id: '50017887'
  },
  {
    address: {
      building: '508',
      coord: [ -73.999813, 40.683876 ],
      street: 'Henry St',
      zipcode: '11231'
    },
    borough: 'Brooklyn',
    cuisine: 'Other',
    grades: [],
    name: '',
    restaurant_id: '50017910'
  },
  {
    address: {
      building: '15',
      coord: [ -73.9966882, 40.7139264 ],
      street: 'Division St',
      zipcode: '10002'
    },
    borough: 'Manhattan',
    cuisine: 'Other',
```

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:

```
Agila_Shree_015> db.restaurants.find({}, { _id: 0 }).sort({ cuisine: 1, borough: -1 })
[
  {
    address: {
      building: '154',
      coord: [ -73.9189064, 40.8654529 ],
      street: 'Post Ave',
      zipcode: '10034'
    },
    borough: 'Manhattan',
    cuisine: 'Other',
    grades: [],
    name: '',
    restaurant_id: '50017887'
  },
  {
    address: {
      building: '15',
      coord: [ -73.9966882, 40.7139264 ],
      street: 'Division St',
      zipcode: '10002'
    },
    borough: 'Manhattan',
    cuisine: 'Other',
    grades: [],
    name: '',
    restaurant_id: '50017912'
  },
  {
    address: {
      building: '508',
      coord: [ -73.999813, 40.683876 ],
      street: 'Henry St',
      zipcode: '11231'
    },
    borough: 'Brooklyn',
    cuisine: 'Other',
    grades: [],
    name: '',
    restaurant_id: '50017910'
  }
]
```

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:

```
Agila_Shree_015> db.restaurants.find({ "address.street": { $exists: true, $ne: "" } })
[
  {
    _id: ObjectId('564c2d949eb21ad392f1d6de'),
    address: {
      building: '154',
      coord: [ -73.9189064, 40.8654529 ],
      street: 'Post Ave',
      zipcode: '10034'
    },
    borough: 'Manhattan',
    cuisine: 'Other',
    grades: [],
    name: '',
    restaurant_id: '50017887'
  },
  {
    _id: ObjectId('564c2d949eb21ad392f1d6ec'),
    address: {
      building: '508',
      coord: [ -73.999813, 40.683876 ],
      street: 'Henry St',
      zipcode: '11231'
    },
    borough: 'Brooklyn',
    cuisine: 'Other',
    grades: [],
    name: '',
    restaurant_id: '50017910'
  },
  {
    _id: ObjectId('564c2d949eb21ad392f1d6ed'),
    address: {
      building: '15',
      coord: [ -73.9966882, 40.7139264 ],
      street: 'Division St',
      zipcode: '10002'
    },
    borough: 'Manhattan',
    cuisine: 'American',
    grades: [
      {
        date: '2014-01-01T00:00:00Z',
        grade: 'A',
        score: 100
      }
    ],
    name: 'The Spotted Pig',
    restaurant_id: '50017910'
  }
]
```

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:

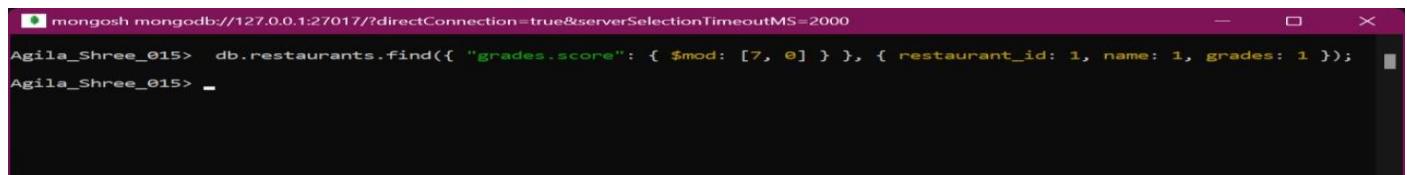
```
Agila_Shree_015> db.restaurants.find({ "address.coord": { $elemMatch: { $type: "double" } } })
[
  {
    _id: ObjectId('564c2d949eb21ad392f1d6de'),
    address: {
      building: '154',
      coord: [ -73.9189064, 40.8654529 ],
      street: 'Post Ave',
      zipcode: '10034'
    },
    borough: 'Manhattan',
    cuisine: 'Other',
    grades: [],
    name: '',
    restaurant_id: '50017887'
  },
  {
    _id: ObjectId('564c2d949eb21ad392f1d6ec'),
    address: {
      building: '508',
      coord: [ -73.999813, 40.683876 ],
      street: 'Henry St',
      zipcode: '11231'
    },
    borough: 'Brooklyn',
    cuisine: 'Other',
    grades: [],
    name: '',
    restaurant_id: '50017910'
  },
  {
    _id: ObjectId('564c2d949eb21ad392f1d6ed'),
    address: {
      building: '15',
      coord: [ -73.9966882, 40.7139264 ],
      street: 'Division St',
      zipcode: '10002'
    }
  }
]
```

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:



A screenshot of a terminal window titled 'mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000'. The window displays the command 'db.restaurants.find({ "grades.score": { \$mod: [7, 0] } }, { restaurant\_id: 1, name: 1, grades: 1 })' and its execution results. The results show three documents from the 'restaurants' collection, each containing the fields '\_id', 'address', 'borough', 'cuisine', 'grades', 'name', and 'restaurant\_id'. The 'grades' field is present in each document, even though it was not explicitly selected in the query projection.

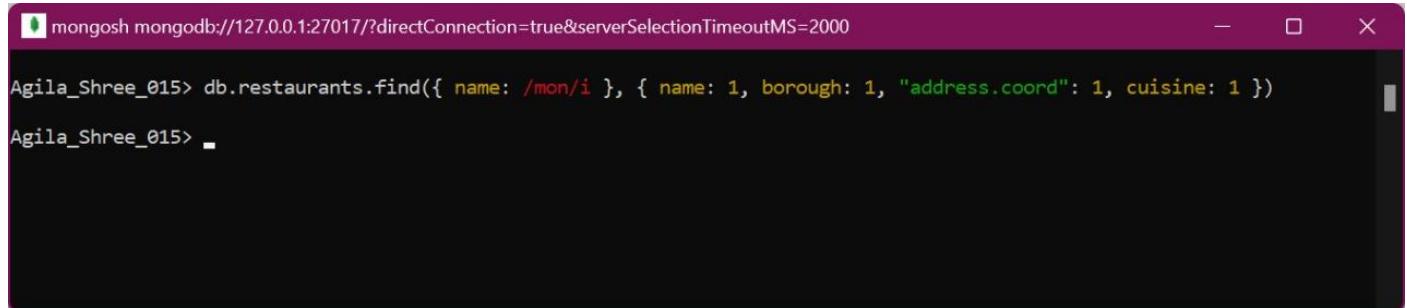
```
mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000
Agila_Shree_015> db.restaurants.find({ "grades.score": { $mod: [7, 0] } }, { restaurant_id: 1, name: 1, grades: 1 });
Agila_Shree_015> -
```

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:



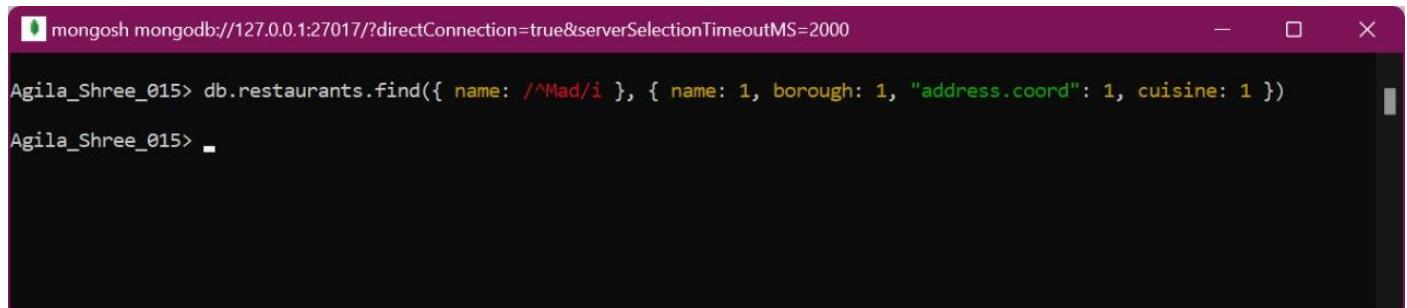
```
mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000
Agila_Shree_015> db.restaurants.find({ name: /mon/i }, { name: 1, borough: 1, "address.coord": 1, cuisine: 1 })
Agila_Shree_015> 
```

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:



```
mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000
Agila_Shree_015> db.restaurants.find({ name: /^Mad/i }, { name: 1, borough: 1, "address.coord": 1, cuisine: 1 })
Agila_Shree_015> 
```

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:



```
mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000
Agila_Shree_015> db.restaurants.find({ "grades": { $elemMatch: { "score": { $lt: 5 } } } })
Agila_Shree_015> 
```

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:



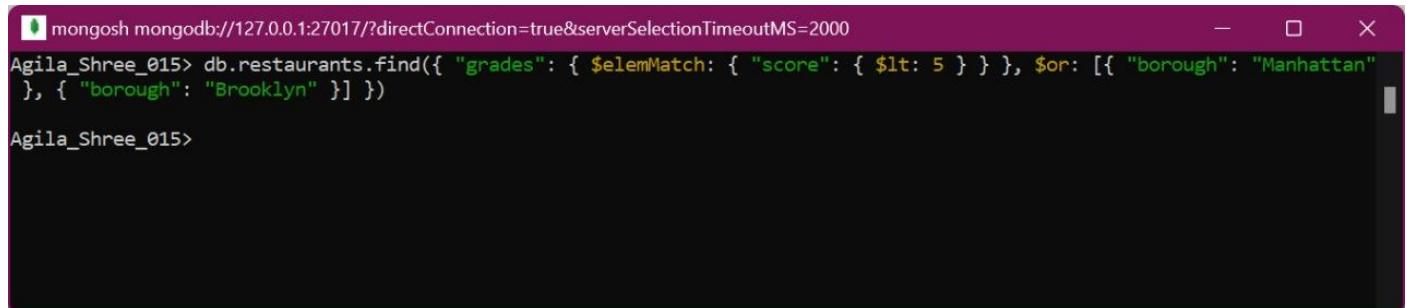
```
mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000
Agila_Shree_015> db.restaurants.find({ "grades": { $elemMatch: { "score": { $lt: 5 } } }, "borough": "Manhattan" })
Agila_Shree_015>
```

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:



```
mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000
Agila_Shree_015> db.restaurants.find({ "grades": { $elemMatch: { "score": { $lt: 5 } } }, $or: [{ "borough": "Manhattan" }, { "borough": "Brooklyn" }] })
Agila_Shree_015>
```

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:



```
mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000
Agila_Shree_015> db.restaurants.find({ "grades": { $elemMatch: { "score": { $lt: 5 } } }, $or: [{ "borough": "Manhattan" }, { "borough": "Brooklyn" }], "cuisine": { $ne: "American" } })
Agila_Shree_015>
```

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:

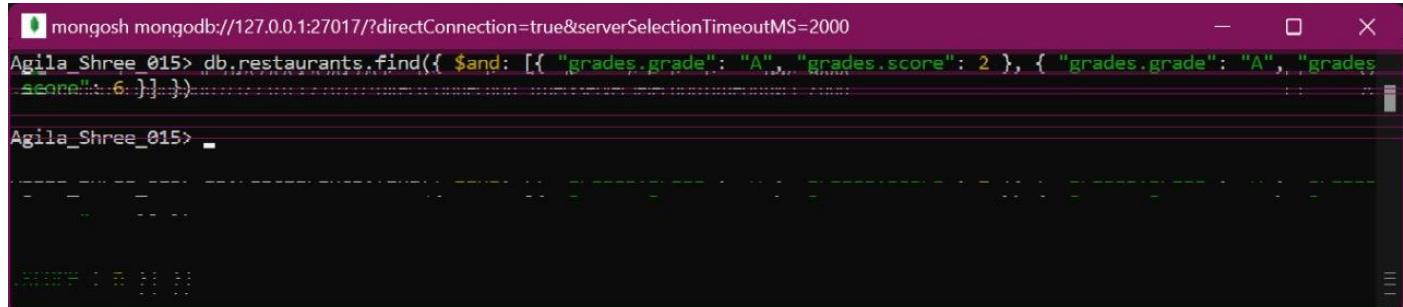


18. Write a MongoDB query to find the restaurants that have a grade with a score of 2 and a grade with a score of 6.

QUERY:

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

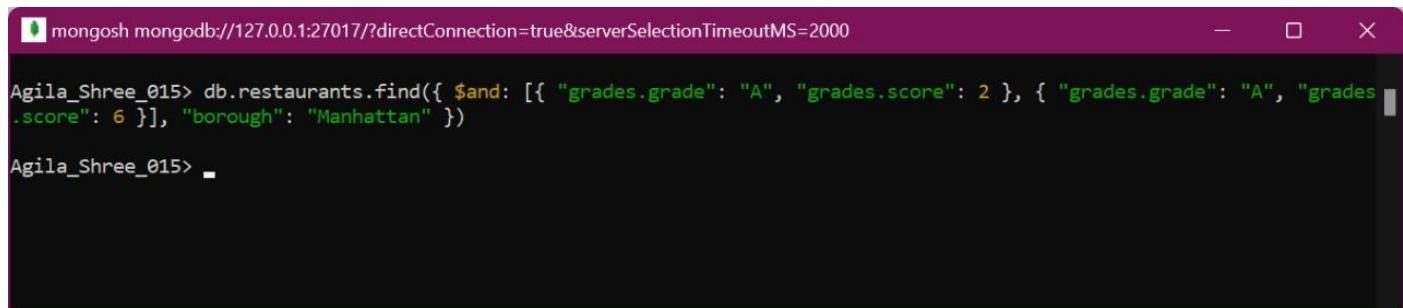
OUTPUT:



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" })
```

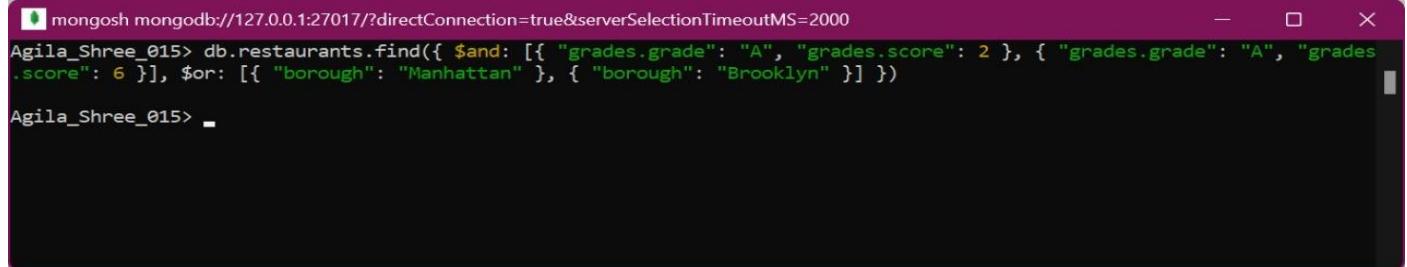


20. Write a MongoDB query to find the restaurants that have a grade with a score of 2 and a grade with a score of 6 and are located in the borough of Manhattan or Brooklyn.

QUERY:

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

OUTPUT:



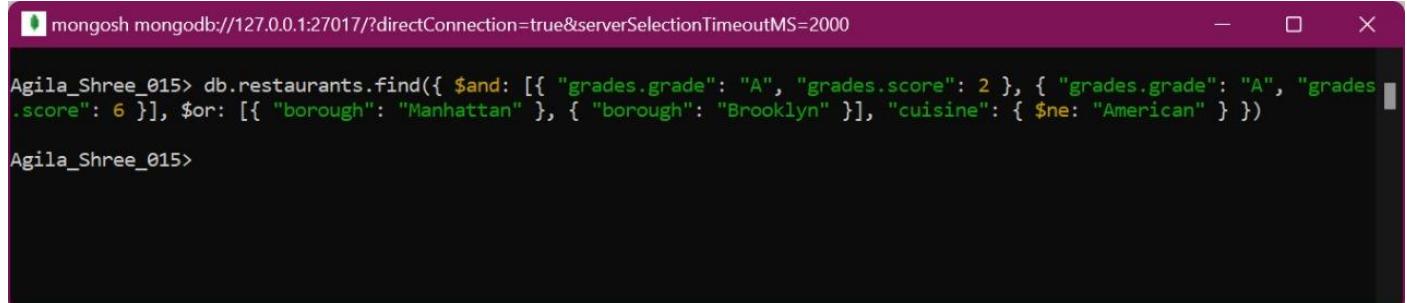
```
mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000
Agila_Shree_015> db.restaurants.find({ $and: [{ "grades.grade": "A", "grades.score": 2 }, { "grades.grade": "A", "grades.score": 6 }], $or: [{ "borough": "Manhattan" }, { "borough": "Brooklyn" }] })
Agila_Shree_015> -
```

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:



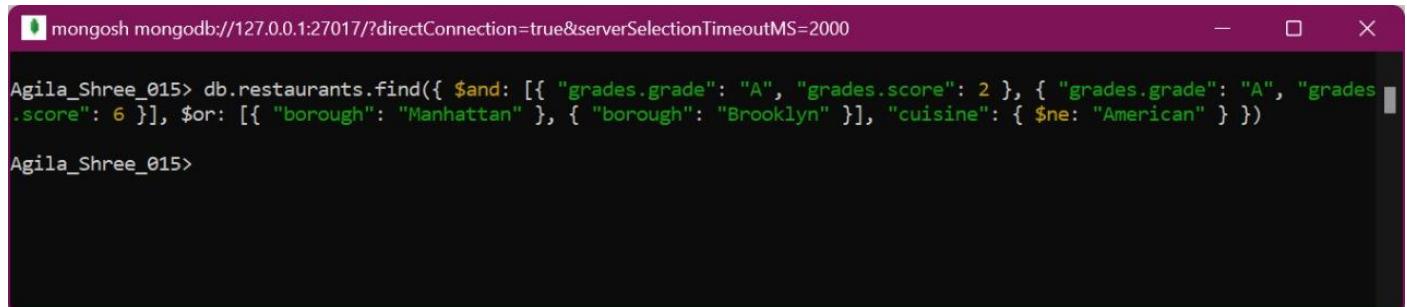
```
mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000
Agila_Shree_015> db.restaurants.find({ $and: [{ "grades.grade": "A", "grades.score": 2 }, { "grades.grade": "A", "grades.score": 6 }], $or: [{ "borough": "Manhattan" }, { "borough": "Brooklyn" }], "cuisine": { $ne: "American" } })
Agila_Shree_015>
```

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:



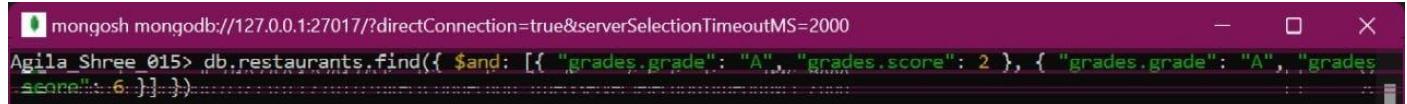
A screenshot of a MongoDB shell window titled 'mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000'. The command entered is: db.restaurants.find({ \$and: [{ "grades.grade": "A", "grades.score": 2 }, { "grades.grade": "A", "grades.score": 6 }], \$or: [{ "borough": "Manhattan" }, { "borough": "Brooklyn" }], "cuisine": { \$ne: "American" } }). The output shows the results of the query.

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:



A screenshot of a MongoDB shell window titled 'mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000'. The command entered is: db.restaurants.find({ \$and: [{ "grades.grade": "A", "grades.score": 2 }, { "grades.grade": "A", "grades.score": 6 }]}). The output shows the results of the query.

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:

```
agila_shree_15> db.movies.find({ year: 1893 })
[
  {
    _id: ObjectId('573a1390f29313caabcd4135'),
    plot: 'Three men hammer on an anvil and pass a bottle of beer around.',
    genres: [ 'Short' ],
    runtime: 1,
    cast: [ 'Charles Kayser', 'John Ott' ],
    num_mflix_comments: 1,
    title: 'Blacksmith Scene',
    fullplot: 'A stationary camera looks at a large anvil with a blacksmith behind it and one on either side. The smith in the middle draws a heated metal rod from the fire, places it on the anvil, and all three begin a rhythmic hammering. After several blows, the metal goes back in the fire. One smith pulls out a bottle of beer, and they each take a swig. Then, out comes the glowing metal and the hammering resumes.',
    countries: [ 'USA' ],
    released: ISODate('1893-05-09T00:00:00.000Z'),
    directors: [ 'William K.L. Dickson' ],
    rated: 'UNRATED',
    awards: { wins: 1, nominations: 0, text: '1 win.' },
    lastupdated: '2015-08-26 00:03:50.133000000',
    year: 1893,
    imbd: { rating: 6.2, votes: 1189, id: 5 },
    type: 'movie',
    tomatoes: {
      viewer: { rating: 3, numReviews: 184, meter: 32 },
      lastUpdated: ISODate('2015-06-28T18:34:09.000Z')
    }
  }
]
agila_shree_15>
```

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:

```
agila_shree_15> db.movies.find({ runtime: { $gt: 120 } })
[
  {
    _id: ObjectId('573a1390f22313caabcd5967'),
    plot: 'An intrepid reporter and his loyal friend battle a bizarre secret society of criminals known as The Vampires.',
    genres: [ 'Action', 'Adventure', 'Crime' ],
    runtime: 399,
    rated: 'NOT RATED',
    cast: [ 'Musidora', 'édouardMathè', 'Marcel Lèvesque', 'Jean Aymè' ],
    poster: 'https://m.media-amazon.com/images/M/MV5BMTC1NTY3NDIzNl5BMl5BanBnXkFtZTgwNTIyODg5MTE@._V1_SY1000_SX677_AL_.jpg',
    title: 'Les vampires',
    fullplot: 'An intrepid reporter and his loyal friend battle a bizarre secret society of criminals known as The Vampires.',
    languages: [ 'French' ],
    released: ISODate('1916-11-23T00:00:00.000Z'),
    directors: [ 'Louis Feuillade' ],
    writers: [ 'Louis Feuillade' ],
    awards: { wins: 0, nominations: 1, text: '1 nomination.' },
    lastupdated: '2015-09-02 00:24:27.333000000',
    year: 1915,
    imdb: { rating: 6.8, votes: 2878, id: 6206 },
    countries: [ 'France' ],
    type: 'movie',
    tomatoes: {
      viewer: { rating: 3.8, numReviews: 2118, meter: 82 },
      dvd: ISODate('2000-05-16T00:00:00.000Z'),
      critic: { rating: 8.8, numReviews: 13, meter: 100 },
      lastUpdated: ISODate('2015-09-15T17:02:33.000Z'),
      rotten: 0,
      fresh: 13
    }
  }
]
agila_shree_15>
```

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

QUERY:

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

OUTPUT:

```
agila_shree_15> db.movies.find({ genres: 'Short' })
[{"_id": ObjectId('573a1390f29313caabcd4135'),  
 "plot": "Three men hammer on an anvil and pass a bottle of beer around.",  
 "genres": [ "Short" ],  
 "runtime": 1,  
 "cast": [ "Charles Kayser", "John Ott" ],  
 "num_mflix_comments": 1,  
 "title": "Blacksmith Scene",  
 "fullplot": "A stationary camera looks at a large anvil with a blacksmith behind it and one on either side. The smith in the middle draws a heated metal rod from the fire, places it on the anvil, and all three begin a rhythmic hammering. After several blows, the metal goes back in the fire. One smith pulls out a bottle of beer, and they each take a swig. Then, out comes the glowing metal and the hammering resumes.",  
 "countries": [ "USA" ],  
 "released": ISODate('1893-05-09T00:00:00.000Z'),  
 "directors": [ "William K.L. Dickson" ],  
 "rated": "UNRATED",  
 "awards": { wins: 1, nominations: 0, text: '1 win.' },  
 "lastupdated": '2015-08-26 00:03:50.133000000',  
 "year": 1893,  
 "imdb": { rating: 6.2, votes: 1189, id: 5 },  
 "type": "movie",  
 "tomatoes": {  
     "viewer": { rating: 3, numReviews: 184, meter: 32 },  
     "lastUpdated": ISODate('2015-06-28T18:34:09.000Z')  
 }
```

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:

```
agila_shree_15> db.movies.find({ directors: 'William K.L. Dickson' })
[
  {
    _id: ObjectId('573a1390f29313caabcd4135'),
    plot: 'Three men hammer on an anvil and pass a bottle of beer around.',
    genres: [ 'Short' ],
    runtime: 1,
    cast: [ 'Charles Kayser', 'John Ott' ],
    num_mflix_comments: 1,
    title: 'Blacksmith Scene',
    fullplot: 'A stationary camera looks at a large anvil with a blacksmith behind it and one on either side. The smith in the middle draws a heated metal rod from the fire, places it on the anvil, and all three begin a rhythmic hammering. After several blows, the metal goes back in the fire. One smith pulls out a bottle of beer, and they each take a swig. Then, out comes the glowing metal and the hammering resumes.',
    countries: [ 'USA' ],
    released: ISODate('1893-05-09T00:00:00.000Z'),
    directors: [ 'William K.L. Dickson' ],
    rated: 'UNRATED',
    awards: { wins: 1, nominations: 0, text: '1 win.' },
    lastupdated: '2015-08-26 00:03:50.133000000',
    year: 1893,
    imdb: { rating: 6.2, votes: 1189, id: 5 },
    type: 'movie',
    tomatoes: {
      viewer: { rating: 3, numReviews: 184, meter: 32 },
      lastUpdated: ISODate('2015-06-28T18:34:09.000Z')
    }
  }
]
agila_shree_15>
```

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:

```
agila_shree_15> db.movies.find({ countries: 'USA' })
[
  {
    _id: ObjectId('573a1390f29313caabcd4135'),
    plot: 'Three men hammer on an anvil and pass a bottle of beer around.',
    genres: [ 'Short' ],
    runtime: 1,
    cast: [ 'Charles Kayser', 'John Ott' ],
    num_mflix_comments: 1,
    title: 'Blacksmith Scene',
    fullplot: 'A stationary camera looks at a large anvil with a blacksmith behind it and one on either side. The smith in the middle draws a heated metal rod from the fire, places it on the anvil, and all three begin a rhythmic hammering. After several blows, the metal goes back in the fire. One smith pulls out a bottle of beer, and they each take a swig. Then, out comes the glowing metal and the hammering resumes.',
    countries: [ 'USA' ],
    released: ISODate('1893-05-09T00:00:00.000Z'),
    directors: [ 'William K.L. Dickson' ],
    rated: 'UNRATED',
    awards: { wins: 1, nominations: 0, text: '1 win.' },
    lastupdated: '2015-08-26 00:03:50.133000000',
    year: 1893,
    imbd: { rating: 6.2, votes: 1189, id: 5 },
    type: 'movie',
    tomatoes: {
      viewer: { rating: 3, numReviews: 184, meter: 32 },
      lastUpdated: ISODate('2015-06-28T18:34:09.000Z')
    }
  }
]
agila_shree_15>
```

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

QUERY:

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

OUTPUT:

```
agila_shree_15> db.movies.find({ rated: 'UNRATED' })
[
  {
    _id: ObjectId('573a1390f29313caabcd4135'),
    plot: 'Three men hammer on an anvil and pass a bottle of beer around.',
    genres: [ 'Short' ],
    runtime: 1,
    cast: [ 'Charles Kayser', 'John Ott' ],
    num_mflix_comments: 1,
    title: 'Blacksmith Scene',
    fullplot: 'A stationary camera looks at a large anvil with a blacksmith behind it and one on either side. The smith in the middle draws a heated metal rod from the fire, places it on the anvil, and all three begin a rhythmic hammering. After several blows, the metal goes back in the fire. One smith pulls out a bottle of beer, and they each take a swig. Then, out comes the glowing metal and the hammering resumes.',
    countries: [ 'USA' ],
    released: ISODate('1893-05-09T00:00:00.000Z'),
    directors: [ 'William K.L. Dickson' ],
    rated: 'UNRATED',
    awards: { wins: 1, nominations: 0, text: '1 win.' },
    lastupdated: '2015-08-26 00:03:50.133000000',
    year: 1893,
    imdb: { rating: 6.2, votes: 1189, id: 5 },
    type: 'movie',
    tomatoes: {
      viewer: { rating: 3, numReviews: 184, meter: 32 },
      lastUpdated: ISODate('2015-06-28T18:34:09.000Z')
    }
  }
]
agila_shree_15>
```

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:

```
agila_shree_15> db.movies.find({ 'imdb.votes': { $gt: 1000 } })
[
  {
    _id: ObjectId('573a1390f29313caabcd4135'),
    plot: 'Three men hammer on an anvil and pass a bottle of beer around.',
    genres: [ 'Short' ],
    runtime: 1,
    cast: [ 'Charles Kayser', 'John Ott' ],
    num_mflix_comments: 1,
    title: 'Blacksmith Scene',
    fullplot: 'A stationary camera looks at a large anvil with a blacksmith behind it and one on either side. The smith in the middle draws a heated metal rod from the fire, places it on the anvil, and all three begin a rhythmic hammering. After several blows, the metal goes back in the fire. One smith pulls out a bottle of beer, and they each take a swig. Then, out comes the glowing metal and the hammering resumes.',
    countries: [ 'USA' ],
    released: ISODate('1893-05-09T00:00:00.000Z'),
    directors: [ 'William K.L. Dickson' ],
    rated: 'UNRATED',
    awards: { wins: 1, nominations: 0, text: '1 win.' },
    lastupdated: '2015-08-26 00:03:50.133000000',
    year: 1893,
    imdb: { rating: 6.2, votes: 1189, id: 5 },
    type: 'movie',
    tomatoes: {
      viewer: { rating: 3, numReviews: 184, meter: 32 },
      lastUpdated: ISODate('2015-06-28T18:34:09.000Z')
    }
  },
  {
    _id: ObjectId('573a1390f22313caabcd5967'),
    plot: 'An intrepid reporter and his loyal friend battle a bizarre secret society of criminals known as The Vampires.',
    genres: [ 'Action', 'Adventure', 'Crime' ],
    runtime: 399,
    rated: 'NOT RATED',
    cast: [ 'Musidora', 'édouardMathè', 'Marcel Lèvesque', 'Jean Aymè' ],
    poster: 'https://m.media-amazon.com/images/M/MV5BMTc1NTY3NDIzNl5BMl5BanBnXkFtZTgwNTIyODg5MTE@._V1_SY1000_SX677_AL_.jpg',
    title: 'Les vampires',
    fullplot: 'An intrepid reporter and his loyal friend battle a bizarre secret society of criminals known as The Vampires.',
    languages: [ 'French' ],
    released: ISODate('1916-11-23T00:00:00.000Z'),
    directors: [ 'Louis Feuillade' ],
    writers: [ 'Louis Feuillade' ],
    awards: { wins: 0, nominations: 1, text: '1 nomination.' },
    lastupdated: '2015-09-02 00:24:27.333000000',
    year: 1915,
    imdb: { rating: 6.8, votes: 2878, id: 6206 },
    countries: [ 'France' ],
    type: 'movie',
    tomatoes: {
      viewer: { rating: 3.8, numReviews: 2118, meter: 82 },
      dvd: ISODate('2000-05-16T00:00:00.000Z'),
      critic: { rating: 8.8, numReviews: 13, meter: 100 },
      lastUpdated: ISODate('2015-09-15T17:02:33.000Z'),
      rotten: 0,
      fresh: 13
    }
  }
]
```

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:

```
agila_shree_15> db.movies.find({ 'imdb.rating': { $gt: 7 } })
[
  {
    _id: ObjectId('573a1391f29313caabcd7a34'),
    plot: 'A kept woman runs into her one-time fiance and finds herself torn between love and comfort.',
    genres: [ 'Drama', 'Romance' ],
    runtime: 78,
    rated: 'TV-PG',
    cast: [
      'Edna Purviance',
      'Clarence Geldart',
      'Carl Miller',
      'Lydia Knott'
    ],
    num_mflix_comments: 3,
    poster: 'https://m.media-amazon.com/images/M/MV5BZjJiMTU2NGQtNWRkNi00ZjExLWExMTUtMmNkNTU0NzRlMTA3XkEyXkFqcGdeQXVyNjUwNzk3NDc@._V1_SV1000_SX677_AL_.jpg',
    title: 'A Woman of Paris: A Drama of Fate',
    fullplot: 'Marie St. Clair believes she has been jilted by her artist fiance Jean when he fails to meet her at the railway station. She goes off to Paris alone. A year later, mistress of wealthy Pierre Revel, she meets Jean again. Misinterpreting events she bounces back and forth between apparent security and true love.',
    countries: [ 'USA' ],
    released: ISODate('1923-11-04T00:00:00.000Z'),
    directors: [ 'Charles Chaplin' ],
    writers: [ 'Charles Chaplin' ],
    awards: { wins: 1, nominations: 0, text: '1 win.' },
    lastupdated: '2015-09-02 00:22:09.303000000',
    year: 1923,
    imdb: { rating: 7.1, votes: 3179, id: 14624 },
    type: 'movie',
    tomatoes: {
      viewer: { rating: 3.7, numReviews: 886, meter: 78 },
      dvd: ISODate('2004-03-02T00:00:00.000Z'),
      critic: { rating: 7.4, numReviews: 11, meter: 91 },
      lastUpdated: ISODate('2015-08-23T18:34:44.000Z'),
      rotten: 1,
      production: 'Criterion Collection',
      fresh: 10
    }
  }
]
agila_shree_15>
```

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:

```
agila_shree_15> db.movies.find({ 'tomatoes.viewer.rating': { $gt: 4 } })
[ {
  _id: ObjectId('573a1391f29313caabcd8945'),
  plot: 'A married farmer falls under the spell of a slatternly woman from the city, who tries to convince him to drown his wife.',
  genres: [ 'Drama', 'Romance' ],
  runtime: 94,
  rated: 'NOT RATED',
  cast: [
    "George O'Brien",
    'Janet Gaynor',
    'Margaret Livingston',
    'Bodil Rosing'
  ],
  num_mflix_comments: 1,
  poster: 'https://m.media-amazon.com/images/M/MV5BNDVkyMwM2IzMzRiMy00NWQ4LTlhMjMtNDI1ZDYyOGVmMzJjXkEyXkFqcGdeQXVvNTgzMzU5MDI@._V1_SX677_AL_.jpg',
  title: 'Sunrise',
  fullplot: 'In this fable-morality subtitled "A Song of Two Humans", the "evil" temptress is a city woman who bewitches farmer Anses and tries to convince him to murder his neglected wife, Indre.',
  countries: [ 'USA' ],
  released: ISODate('1927-11-04T00:00:00.000Z'),
  directors: [ 'F.W. Murnau' ],
  writers: [
    'Carl Mayer (scenario)',
    'Hermann Sudermann (from an original theme by)',
    'Katherine Hilliker (titles)',
    'H.H. Caldwell (titles)'
  ],
  awards: {
    wins: 5,
    nominations: 1,
    text: 'Won 3 Oscars. Another 2 wins & 1 nomination.'
  },
  lastupdated: '2015-09-12 00:26:13.493000000',
  year: 1927,
  imdb: { rating: 8.4, votes: 24480, id: 18455 },
  type: 'movie',
  tomatoes: {
    viewer: { rating: 4.4, numReviews: 9134, meter: 92 },
    dvd: ISODate('2008-12-09T00:00:00.000Z'),
    critic: { rating: 8.9, numReviews: 48, meter: 98 },
    lastUpdated: ISODate('2015-09-10T19:15:02.000Z'),
    consensus: 'Boasting masterful cinematography to match its well-acted, wonderfully romantic storyline, Sunrise is perhaps the final -- and arguably definitive -- statement of the silent era.',
    rotten: 1,
    production: 'Fox Films',
    fresh: 47
  }
},
```

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

QUERY:

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

OUTPUT:

```
agila_shree_15> db.movies.find({ "awards.wins": { $gt: 0 } })
[ {
  _id: ObjectId('573a1390f29313caabcd4135'),
  plot: 'Three men hammer on an anvil and pass a bottle of beer around.',
  genres: [ 'Short' ],
  runtime: 1,
  cast: [ 'Charles Kayser', 'John Ott' ],
  num_mflix_comments: 1,
  title: 'Blacksmith Scene',
  fullplot: 'A stationary camera looks at a large anvil with a blacksmith behind it and one on either side. The smith in the middle draws a heated metal rod from the fire, places it on the anvil, and all three begin a rhythmic hammering. After several blows, the metal goes back in the fire. One smith pulls out a bottle of beer, and they each take a swig. Then, out comes the glowing metal and the hammering resumes.',
  countries: [ 'USA' ],
  released: ISODate('1893-05-09T00:00:00.000Z'),
  directors: [ 'William K.L. Dickson' ],
  rated: 'UNRATED',
  awards: { wins: 1, nominations: 0, text: '1 win.' },
  lastupdated: '2015-08-26 00:03:50.133000000',
  year: 1893,
  imdb: { rating: 6.2, votes: 1189, id: 5 },
  type: 'movie',
  tomatoes: {
    viewer: { rating: 3, numReviews: 184, meter: 32 },
    lastUpdated: ISODate('2015-06-28T18:34:09.000Z')
  }
},
{
  _id: ObjectId('573a1391f29313caabcd7a34'),
  plot: 'A kept woman runs into her one-time fiance and finds herself torn between love and comfort.',
  genres: [ 'Drama', 'Romance' ],
  runtime: 78,
  rated: 'TV-PG',
  cast: [
    'Edna Purviance',
    'Clarence Geldart',
    'Carl Miller',
    'Lydia Knott'
  ],
  num_mflix_comments: 3,
  poster: 'https://m.media-amazon.com/images/M/MV5BZjJiMTU2NGQtNWRkNi00ZjExLWExMTUtMmNkNTU0NzRjMTA3XkEyXkFqcGdeQXVyNjIwMzknMDc0LVI SY1000 SX677 AL_.jpg',
  title: 'A Woman of Fate',
  fullplot: 'Marie St. Clair believes she has been jilted by her artist fiance Jean when he fails to meet her at the railway station. She goes off to Paris alone. A year later, mistress of wealthy Pierre Revel, she meets Jean again. Misinterpreting events she bounces back and forth between apparent security and true love.',
  countries: [ 'USA' ],
  released: ISODate('1923-11-04T00:00:00.000Z'),
  directors: [ 'Charles Chaplin' ],
  writers: [ 'Charles Chaplin' ],
  awards: { wins: 1, nominations: 0, text: '1 win.' },
  lastupdated: '2015-09-02 00:22:09.303000000',
  year: 1923,
  imdb: { rating: 7.1, votes: 3179, id: 14624 },
  type: 'movie',
  tomatoes: {
    viewer: { rating: 3.7, numReviews: 886, meter: 78 },
    dvd: ISODate('2004-03-02T00:00:00.000Z'),
    critic: { rating: 7.4, numReviews: 11, meter: 91 },
    lastUpdated: ISODate('2015-08-23T18:34:44.000Z'),
    rotten: 1,
    production: 'Criterion Collection',
    fresh: 10
  }
},
{
  _id: ObjectId('573a1391f29313caabcd8945'),
  plot: 'A married farmer falls under the spell of a slatternly woman from the city, who tries to convince him to drown his wife.',
  genres: [ 'Drama', 'Romance' ],
  runtime: 94,
  rated: 'NOT RATED',
  cast: [
    'George O'Brien',
    'Janet Gaynor',
    'Margaret Livingston',
    'Bodie Rising'
  ],
  num_mflix_comments: 1,
  poster: 'https://m.media-amazon.com/images/M/MV5BN0VkyYmM2ItNzRiMy00NAQ4LTlhMjMtNDI1ZDYyOGVmMzJjXkEyXkFqcGdeQXVyNTIwMzQ0LVI SY1000 SX677 AL_.jpg',
  title: 'Sunrise',
  fullplot: 'In this fable-morality subtitled "A Song of Two Humans", the "evil" temptress is a city woman who bewitches farmer Arnes and tries to convince him to murder his neglected wife, Indre.',
  countries: [ 'USA' ],
  released: ISODate('1927-11-04T00:00:00.000Z'),
  directors: [ 'F.W. Murnau' ],
  writers: [
    'Carl Mayer (scenario)',
    'Hermann Sudermann (from an original theme by)',
    'Katherine Hilliker (titles)',
    'H.H. Caldwell (titles)'
  ],
  awards: {
    wins: 5,
    nominations: 1,
    text: 'Won 3 Oscars. Another 2 wins & 1 nomination.'
  },
  lastupdated: '2015-09-12 00:26:13.493000000',
  year: 1927,
  imdb: { rating: 8.4, votes: 24480, id: 18455 },
  type: 'movie',
  tomatoes: {
    viewer: { rating: 4.4, numReviews: 9134, meter: 92 },
    dvd: ISODate('2008-12-09T00:00:00.000Z'),
    critic: { rating: 8.9, numReviews: 48, meter: 98 },
    lastUpdated: ISODate('2015-09-10T19:15:02.000Z'),
    consensus: 'Boasting masterful cinematography to match its well-acted, wonderfully romantic storyline, Sunrise is perhaps the final -- and arguably definitive -- statement of the silent era.',
    rotten: 1,
    production: 'Fox Films',
    fresh: 47
  }
}
]
```

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 })
```

```
agila_shree_15> 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 })
[
  {
    _id: ObjectId('573a1390f22313caabcd5967'),
    genres: [ 'Action', 'Adventure', 'Crime' ],
    runtime: 399,
    cast: [ 'Musidora', 'éduardMathè', 'Marcel Lèvesque', 'Jean Aymè' ],
    title: 'Les vampires',
    languages: [ 'French' ],
    released: ISODate('1916-11-23T00:00:00.000Z'),
    directors: [ 'Louis Feuillade' ],
    writers: [ 'Louis Feuillade' ],
    awards: { wins: 0, nominations: 1, text: '1 nomination.' },
    year: 1915,
    countries: [ 'France' ]
  },
  {
    _id: ObjectId('573a1391f29313caabcd8945'),
    genres: [ 'Drama', 'Romance' ],
    runtime: 94,
    cast: [
      "George O'Brien",
      'Janet Gaynor',
      'Margaret Livingston',
      'BodilRosing'
    ],
    title: 'Sunrise',
    countries: [ 'USA' ],
    released: ISODate('1927-11-04T00:00:00.000Z'),
    directors: [ 'F.W. Murnau' ],
    writers: [
      'Carl Mayer (scenario)',
      'Hermann Sudermann (from an original theme by)',
      'Katherine Hilliker (titles)',
      'H.H. Caldwell (titles)'
    ],
    awards: {
      wins: 5,
      nominations: 1,
      text: 'Won 3 Oscars. Another 2 wins & 1 nomination.'
    },
    year: 1927
  }
]
```

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:

```
agila_shree_15> 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 } )
[
  {
    _id: ObjectId('573a1390f29313caabcd4135'),
    genres: [ 'Short' ],
    runtime: 1,
    cast: [ 'Charles Kayser', 'John Ott' ],
    title: 'Blacksmith Scene',
    countries: [ 'USA' ],
    released: ISODate('1893-05-09T00:00:00.000Z'),
    directors: [ 'William K.L. Dickson' ],
    awards: { wins: 1, nominations: 0, text: '1 win.' },
    year: 1893
  }
]
agila_shree_15>
```

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:

```
agila_shree_15> db.movies.find( { released: ISODate("1893-05-09T00:00:00.000Z") }, { title: 1, languages: 1, released: 1, directors: 1, writers: 1, countries: 1 } )
[
  {
    _id: ObjectId('573a1390f29313caabcd4135'),
    title: 'Blacksmith Scene',
    countries: [ 'USA' ],
    released: ISODate('1893-05-09T00:00:00.000Z'),
    directors: [ 'William K.L. Dickson' ]
  }
]
agila_shree_15> ■
```

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:

```
agila_shree_15> db.movies.find( { title: /scene/i }, { title: 1, languages: 1, released: 1, directors: 1, writers: 1, countries: 1 } )
[
  {
    _id: ObjectId('573a1390f29313caabcd4135'),
    title: 'Blacksmith Scene',
    countries: [ 'USA' ],
    released: ISODate('1893-05-09T00:00:00.000Z'),
    directors: [ 'William K.L. Dickson' ]
  }
]
agila_shree_15> ■
```

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

RESULT: