

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),  
    NAME varchar2(25)  
);
```

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 DEPT table:

```
1 CREATE TABLE DEPT(  
2     ID number(7),  
3     NAME varchar2(25)  
4 );
```

The 'Results' tab is active, showing the output: "Table created." Below it, the time taken is listed as "0.04 seconds".

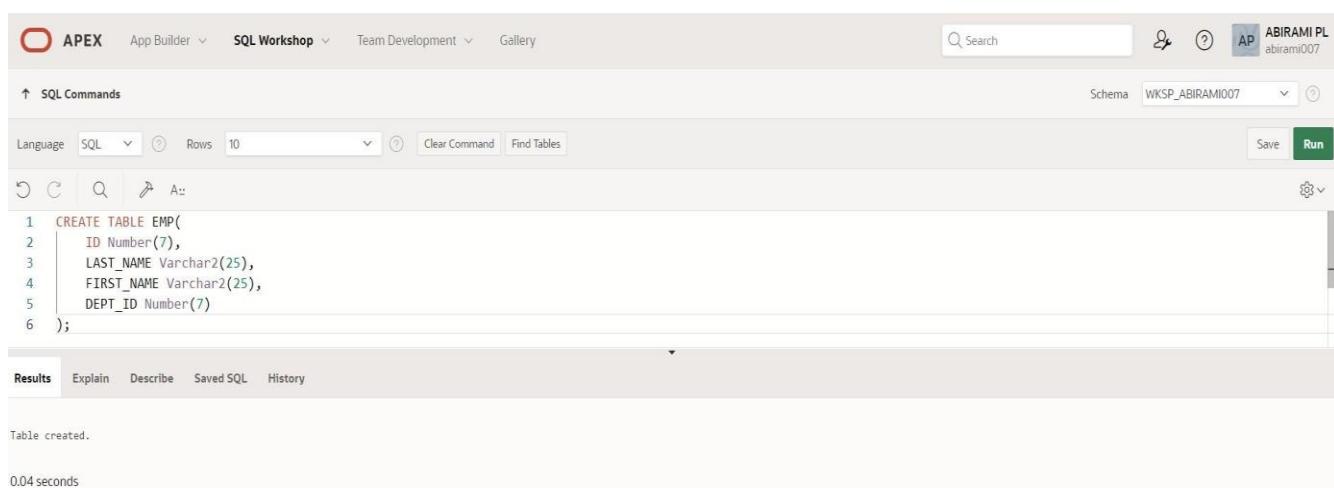
2. Create the EMP table based on the following instance chart. Confirm that the table is created.

Column name	ID	LAST_NAME	FIRST_NAME	DEPT_ID
Key Type				
Nulls/Unique				
FK table				
FK column				
Data Type	Number	Varchar	Varchar	Number
Length	7	25	25	7

QUERY:

```
CREATE TABLE EMP(
    ID Number(7),
    LAST_NAME Varchar2(25),
    FIRST_NAME Varchar2(25),
    DEPT_ID Number(7)
);
```

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there are tabs for Search, Help, and a user profile (ABIRAMI PL abirami007). The main area is titled "SQL Commands". It shows the SQL code for creating the EMP table. Below the code, the "Results" tab is selected, displaying the message "Table created." and "0.04 seconds".

```
1 CREATE TABLE EMP(
2     ID Number(7),
3     LAST_NAME Varchar2(25),
4     FIRST_NAME Varchar2(25),
5     DEPT_ID Number(7)
6 );
```

Table created.
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 (selected), Team Development, and Gallery. On the right, there's a user profile for 'ABIRAMI PL' and a schema dropdown set to 'WKSP_ABIRAMI007'. The main area is titled 'SQL Commands' with a sub-section 'Language: SQL'. Below the command input field, there are buttons for Save and Run. The command entered is 'ALTER TABLE EMP MODIFY (LAST_NAME VARCHAR(50));'. The results section shows the output: 'Table altered.' and a execution time of '0.07 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(Employee_id Number(6),first_name  
varchar(20),last_name varchar(25),salary Number(8,2),Dept_id Number(4));
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there's a user profile for 'ABIRAMI PL' and a schema dropdown set to 'WKSP_ABIRAMI007'. The main area is titled 'SQL Commands' with a sub-section 'Language: SQL'. Below the command input field, there are buttons for Save and Run. The command entered is 'CREATE TABLE EMPLOYEES2(Employee_id Number(6),first_name varchar(20),last_name varchar(25),salary Number(8,2),Dept_id Number(4));'. The results section shows the output: 'Table created.' and a execution time of '0.04 seconds'.

5. Drop the EMP table.

QUERY:

```
DROP TABLE EMP;
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. In the top navigation bar, 'APEX' is selected. The main area is titled 'SQL Commands'. A single command is entered in the text field: 'DROP TABLE EMP;'. Below the command, the output shows the message 'Table dropped.' and a execution time of '0.09 seconds'. The results tab is active.

```
1  DROP TABLE EMP;
```

Table dropped.
0.09 seconds

6. Rename the EMPLOYEES2 table as EMP.

QUERY:

```
RENAME EMPLOYEES2 TO EMP;
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. In the top navigation bar, 'APEX' is selected. The main area is titled 'SQL Commands'. A single command is entered in the text field: 'RENAME EMPLOYEES2 TO EMP;'. Below the command, the output shows the message 'Statement processed.' and a execution time of '0.05 seconds'. The results tab is active.

```
1  RENAME EMPLOYEES2 TO EMP;
```

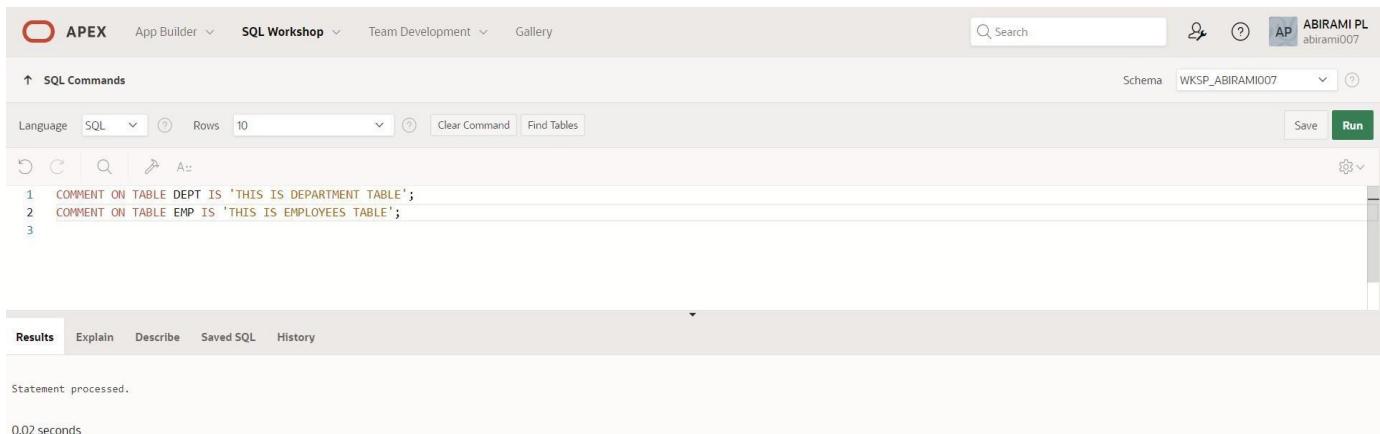
Statement processed.
0.05 seconds

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. In the top navigation bar, 'APEX' is selected. The main area is titled 'SQL Commands'. Two SQL statements are present in the command pane:

```
1 COMMENT ON TABLE DEPT IS 'THIS IS DEPARTMENT TABLE';
2 COMMENT ON TABLE EMP IS 'THIS IS EMPLOYEES TABLE';
3
```

Below the command pane, the 'Results' tab is selected. The output shows:

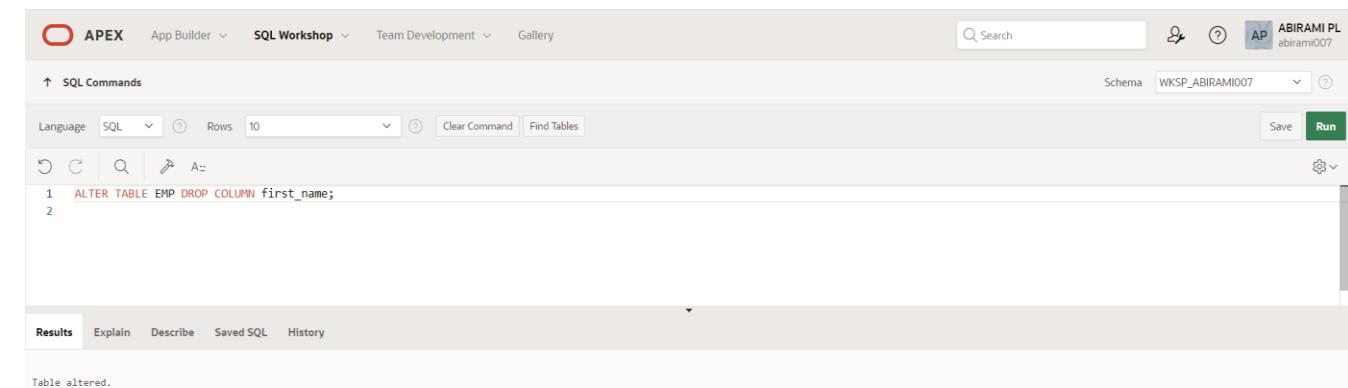
```
Statement processed.
0.02 seconds
```

8. Drop the First_name column from the EMP table and confirm it.

QUERY:

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

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. In the top navigation bar, 'APEX' is selected. The main area is titled 'SQL Commands'. One SQL statement is present in the command pane:

```
1 ALTER TABLE EMP DROP COLUMN first_name;
2
```

Below the command pane, the 'Results' tab is selected. The output shows:

```
Table altered.
```

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, Team Development, and Gallery. The SQL Workshop tab is selected. The schema dropdown shows WKSP_ABIRAMI007. The main area displays the SQL command for creating the MY_EMPLOYEE table, which was copied from the previous section. The command is:

```
1 CREATE TABLE MY_EMPLOYEE(ID NUMBER(4) NOT NULL,  
2 LAST_NAME VARCHAR(25),FIRST_NAME VARCHAR(25),  
3 USERID VARCHAR(25), SALARY NUMBER(9,2));  
4  
5
```

Below the command, the Results tab is active, showing the output: "Table created." and "0.04 seconds".

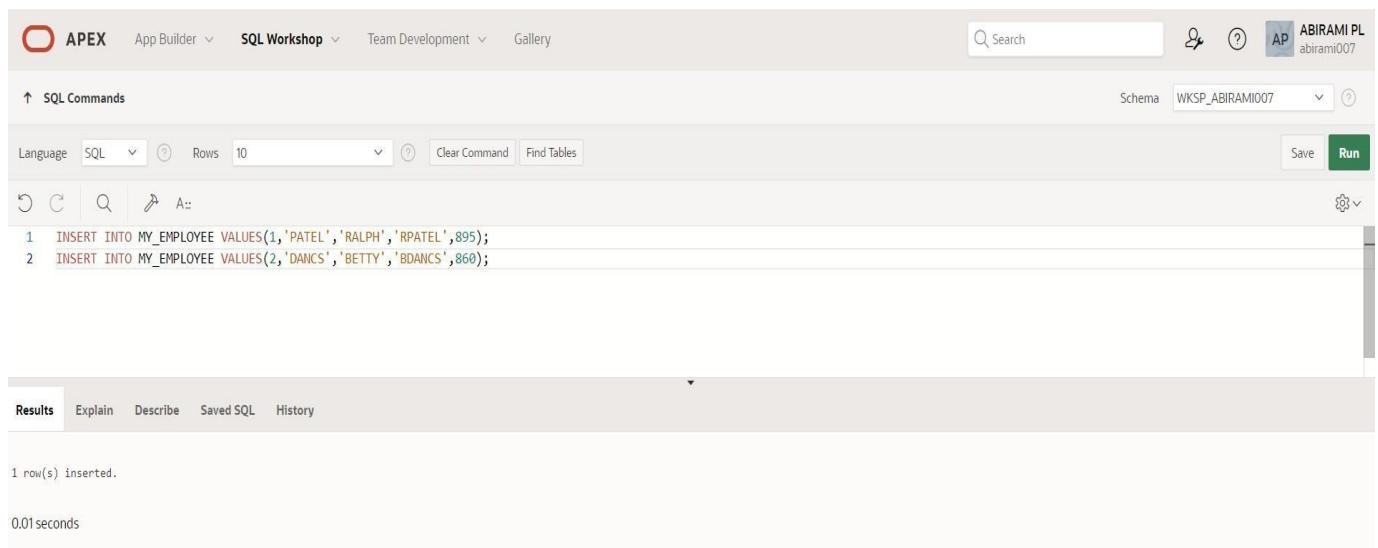
2. Add the first and second rows data to MY_EMPLOYEE table from the following sample data.

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 links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. The user is logged in as ABIRAMI PL abirami007. The main area is titled "SQL Commands". The command input field contains the following SQL code:

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

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

1 row(s) inserted.
0.01 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 (selected), Team Development, and Gallery. The right side shows a user profile for ABIRAMI PL abirami007. The main area has tabs for SQL Commands, Results, Explain, Describe, Saved SQL, and History. The SQL Commands tab shows the query: `SELECT * FROM MY_EMPLOYEE;`. The Results tab displays the following data:

ID	LAST_NAME	FIRST_NAME	USERID	SALARY
2	DANCS	BETTY	BDANCS	860
1	PATEL	RALPH	RPATEL	895

2 rows returned in 0.04 seconds. There is a 'Download' link at the bottom.

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(ID,Last_Name,First_Name,Userid,Salary)
```

```
VALUES(3,'Biri','Ben','bbiri',1100);
```

```
Insert INTO MY_EMPLOYEE(ID,Last_Name,First_Name,Userid,Salary)
```

```
VALUES(4,'Newman','chad','cnewman',750);
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows a user profile for ABIRAMI PL abirami007. The main area has tabs for SQL Commands, Results, Explain, Describe, Saved SQL, and History. The SQL Commands tab shows the following insert statements:

```
1 Insert INTO MY_EMPLOYEE(ID,Last_Name,First_Name,Userid,Salary)
2 VALUES(3,'Biri','Ben','bbiri',1100);
3 Insert INTO MY_EMPLOYEE(ID,Last_Name,First_Name,Userid,Salary)
4 VALUES(4,'Newman','chad','cnewman',750);
```

The Results tab shows the message: 1 row(s) inserted. Execution time: 0.01 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, and Gallery. On the right, there's a search bar, user profile, and schema information (WKSP_ABIRAMI007). The main area is titled 'SQL Commands' with a language dropdown set to SQL. Below it, a command line shows the query 'SELECT * FROM MY_EMPLOYEE;'. The results tab is selected, displaying a table with four rows of employee data. The columns are ID, LAST_NAME, FIRST_NAME, USERID, and SALARY. The data is as follows:

ID	LAST_NAME	FIRST_NAME	USERID	SALARY
2	DANCS	BETTY	BDANCS	860
1	PATEL	RALPH	RPATEL	895
4	Newman	chad	Cnewman	750
3	Biri	Ben	bbiri	1100

Below the table, a message indicates '4 rows returned in 0.01 seconds' and a 'Download' link.

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, similar to the previous one but with a different query. The top navigation bar and schema information are the same. The main area shows the command 'UPDATE MY_EMPLOYEE SET LAST_NAME='DREXLER' WHERE ID=3;'. The results tab is selected, showing a message '1 row(s) updated.' and a time stamp '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. In the top navigation bar, 'APEX' is selected. The main area is titled 'SQL Commands'. A single line of SQL code is entered: '1 UPDATE MY_EMPLOYEE SET SALARY=1000 WHERE SALARY < 900;'. Below the command, the results tab is selected, showing the message '3 row(s) updated.' and a execution time of '0.01seconds'.

8. Delete Betty Dancs from MY_EMPLOYEE table.

QUERY:

```
DELETE FROM MY_EMPLOYEE WHERE LAST_NAME='DANCS';
```

OUTPUT:

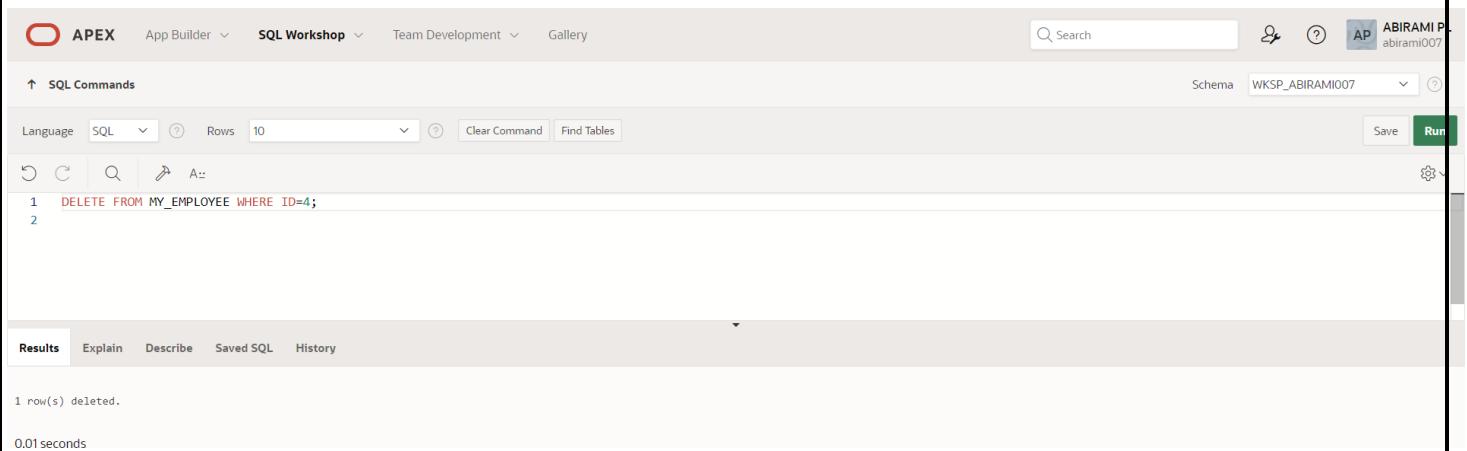
The screenshot shows the Oracle SQL Workshop interface. In the top navigation bar, 'APEX' is selected. The main area is titled 'SQL Commands'. A single line of SQL code is entered: '2 DELETE FROM MY_EMPLOYEE WHERE LAST_NAME='DANCS';'. Below the command, the results tab is selected, showing the message '1 row(s) deleted.' and a execution time of '0.01seconds'.

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. In the SQL Commands pane, the following command is entered:

```
1  DELETE FROM MY_EMPLOYEE WHERE ID=4;
2
```

In the Results pane, the output is:

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

Execution time: 0.01 seconds

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

RESULT :

INCLUDING CONSTRAINTS

EX-NO : 3

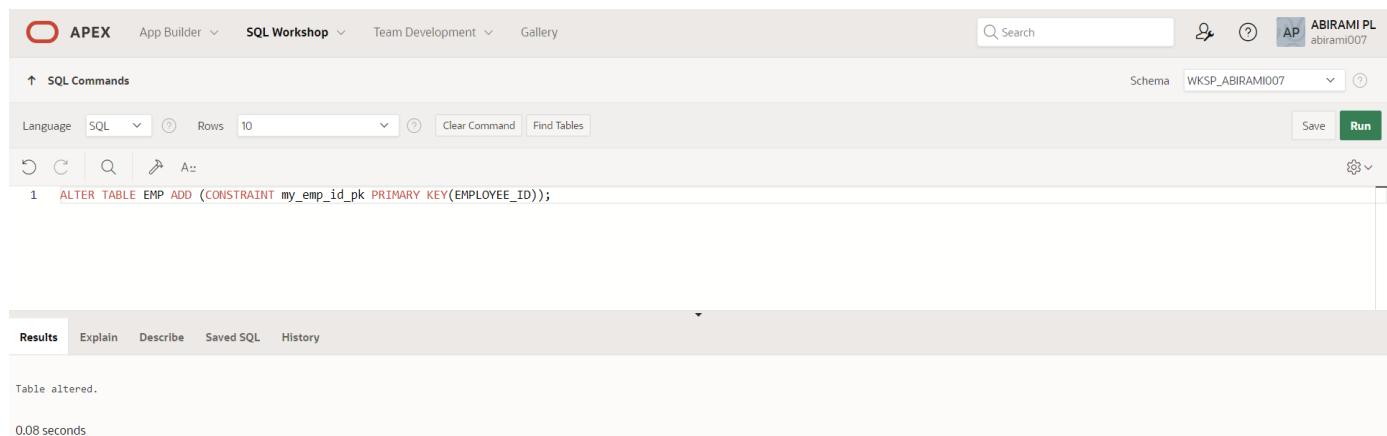
DATE:

1. Add a table-level PRIMARY KEY constraint to the EMP table on the ID column. The constraint should be named at creation. Name the constraint my_emp_id_pk.

QUERY:

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

OUTPUT:



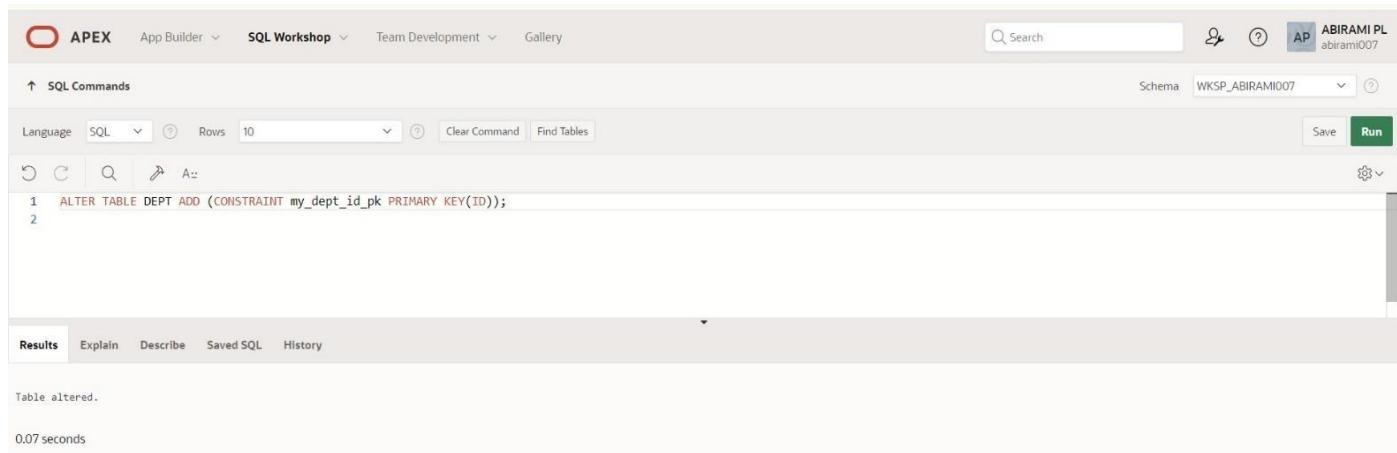
The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there's a search bar, user information (ABIRAMI PL abirami007), and a schema dropdown set to WKSP_ABIRAMI007. Below the toolbar, the SQL Commands tab is active, with Language set to SQL, Rows to 10, and a Run button. The main area displays the SQL command: `1 ALTER TABLE EMP ADD (CONSTRAINT my_emp_id_pk PRIMARY KEY(EMPLOYEE_ID));`. The results section shows the output: `Table altered.` and `0.08 seconds`.

2. Create a PRIMARY KEY constraint to the DEPT table using the ID column. The constraint should be named at creation. Name the constraint my_dept_id_pk.

QUERY:

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

OUTPUT:



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

```
1 ALTER TABLE DEPT ADD (CONSTRAINT my_dept_id_pk PRIMARY KEY(ID));  
2
```

Below the command, the results section displays the output:

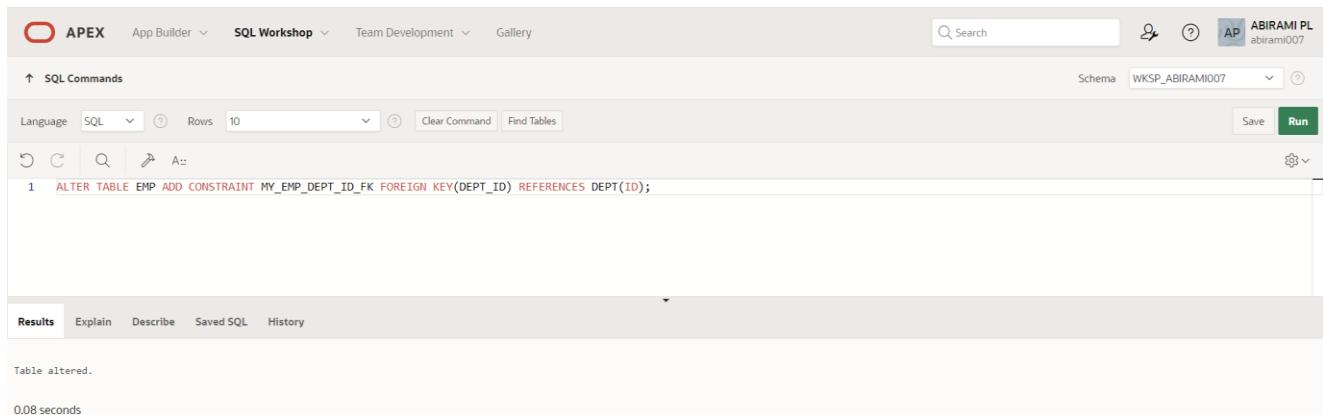
```
Table altered.  
0.07 seconds
```

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 DEPT(ID);
```

OUTPUT:



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

```
1 ALTER TABLE EMP ADD CONSTRAINT MY_EMP_DEPT_ID_FK FOREIGN KEY(DEPT_ID) REFERENCES DEPT(ID);
```

Below the command, the results section displays the output:

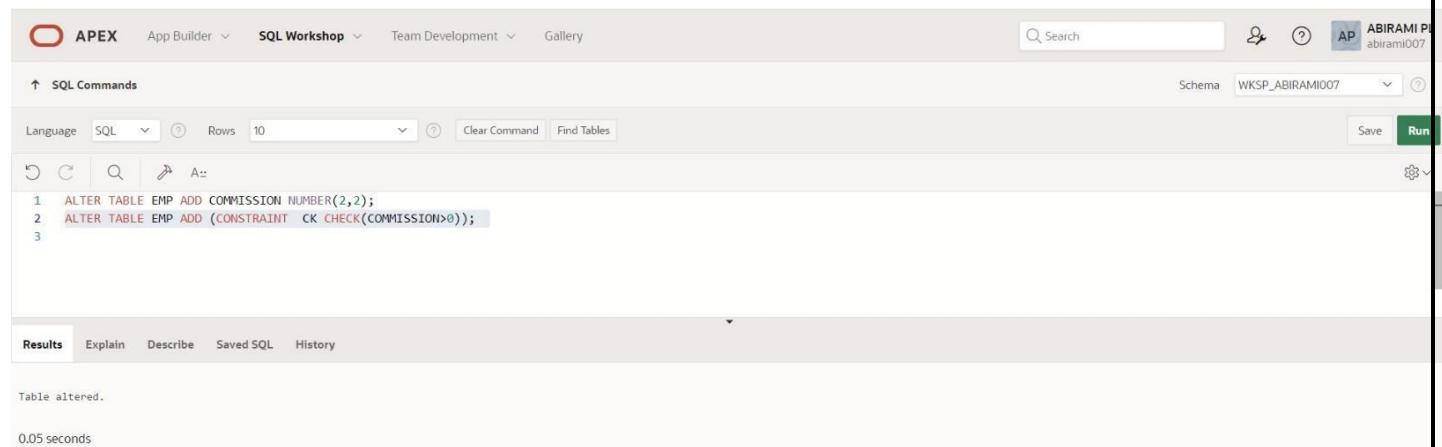
```
Table altered.  
0.08 seconds
```

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);
ALTER TABLE EMP ADD (CONSTRAINT CK CHECK(COMMISSION>0));
```

OUTPUT:



```
1 ALTER TABLE EMP ADD COMMISSION NUMBER(2,2);
2 ALTER TABLE EMP ADD (CONSTRAINT CK CHECK(COMMISSION>0));
3
```

Table altered.
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.

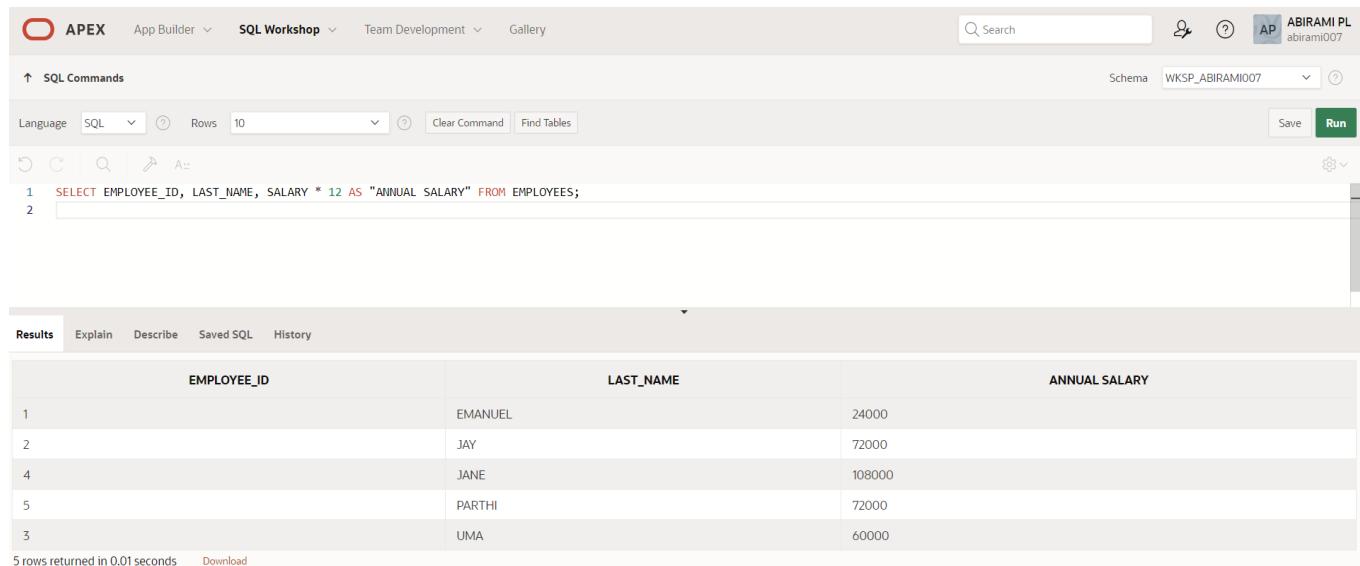
Identify the Errors

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

QUERY:

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

OUTPUT:



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

```
1 SELECT EMPLOYEE_ID, LAST_NAME, SALARY * 12 AS "ANNUAL SALARY" FROM EMPLOYEES;  
2
```

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

EMPLOYEE_ID	LAST_NAME	ANNUAL SALARY
1	EMANUEL	24000
2	JAY	72000
4	JANE	108000
5	PARTHI	72000
3	UMA	60000

Below the table, it says "5 rows returned in 0.01 seconds".

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 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. The right side shows a user profile for 'ABIRAMI PL abirami007'. The main area has tabs for 'SQL Commands' and 'Describe'. The SQL command 'DESC DEPARTMENT;' is entered in the editor. Below the editor is a table describing the 'DEPARTMENT' table:

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
DEPARTMENT	DEPT_ID	NUMBER	-	7	0	-	✓	-	-
	DEPT_NAME	VARCHAR2	20	-	-	-	✓	-	-
	MANAGER_ID	NUMBER	-	7	0	-	✓	-	-
	LOCATION_ID	NUMBER	-	20	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 with the same top navigation and user profile as the previous screenshot. The SQL command is now a select query:

```
1 SELECT EMPLOYEE_ID, LAST_NAME, JOB_ID, HIRE_DATE
2 FROM EMPLOYEES;
3
4
```

Below the editor is a table showing the results of the query:

EMPLOYEE_ID	LAST_NAME	JOB_ID	HIRE_DATE
1	EMANUEL	101	02/02/1998
2	JAY	102	02/02/1999
4	JANE	105	02/02/1999
5	PARTHI	104	02/02/1999
3	UMA	102	02/02/1999

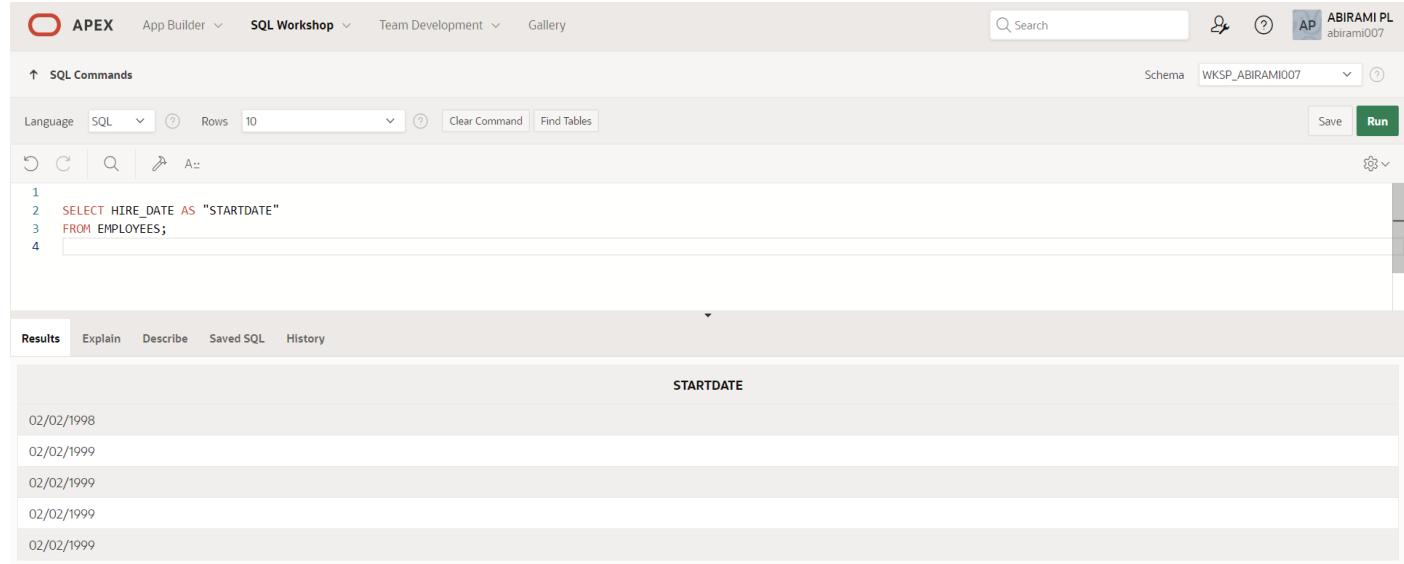
At the bottom, it says '5 rows returned in 0.01 seconds' and 'Download'.

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 APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there's a search bar, user profile, and schema information (WKSP_ABIRAMI007). The main area is titled 'SQL Commands' with a sub-section 'Results'. The SQL code entered is:

```
1 SELECT HIRE_DATE AS "STARTDATE"  
2 FROM EMPLOYEES;  
3  
4
```

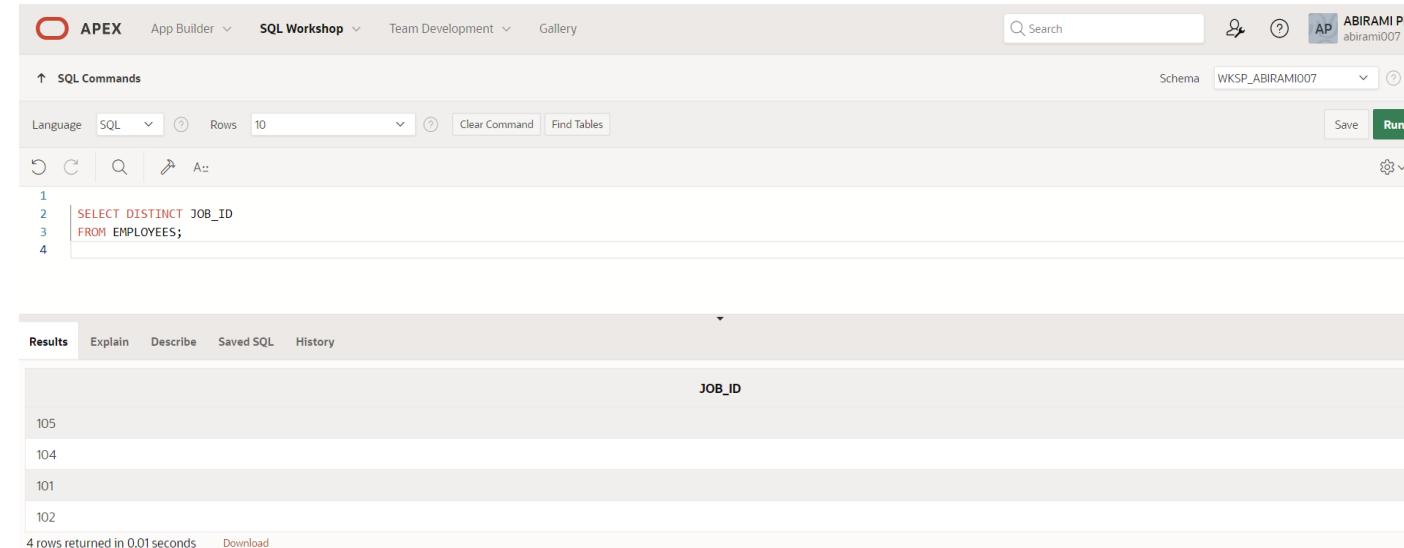
The results section shows a single column named 'STARTDATE' with five rows of data: 02/02/1998, 02/02/1999, 02/02/1999, 02/02/1999, and 02/02/1999. Below the results, it says '5 rows returned in 0.00 seconds' and has a 'Download' link.

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 APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there's a search bar, user profile, and schema information (WKSP_ABIRAMI007). The main area is titled 'SQL Commands' with a sub-section 'Results'. The SQL code entered is:

```
1 SELECT DISTINCT JOB_ID  
2 FROM EMPLOYEES;  
3  
4
```

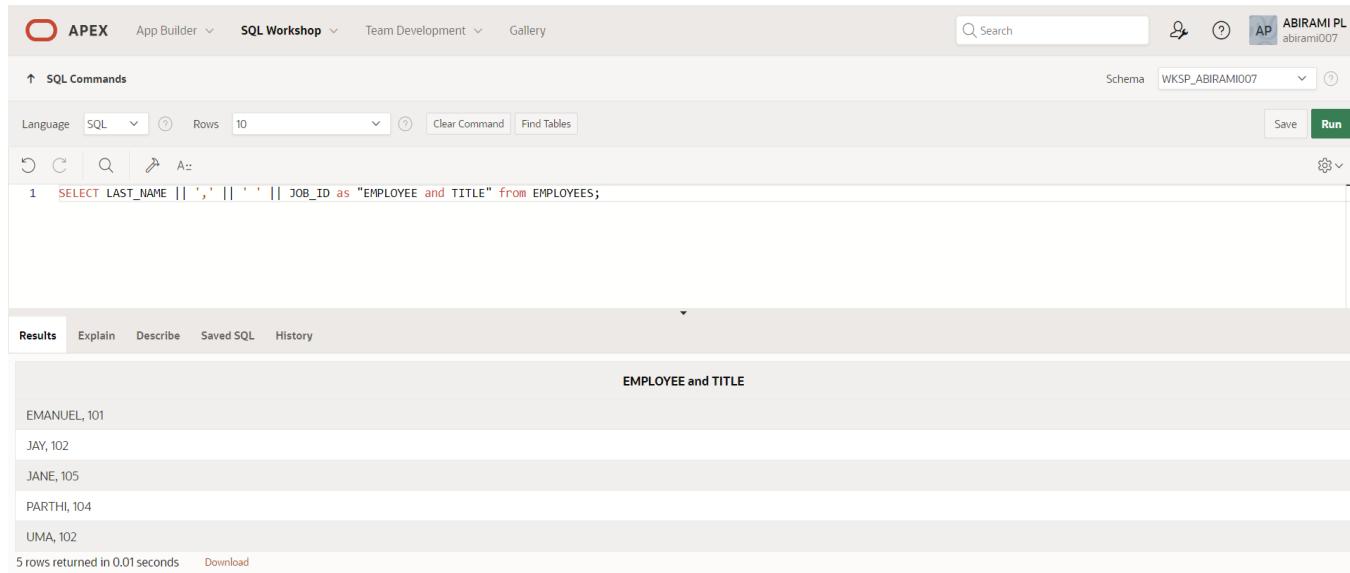
The results section shows a single column named 'JOB_ID' with four distinct values: 105, 104, 101, and 102. Below the results, it says '4 rows returned in 0.01 seconds' and has a 'Download' link.

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 APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows a user profile for ABIRAMI PL abirami007. The main area is titled 'SQL Commands' with a schema dropdown set to WKSP_ABIRAMI007. The SQL editor contains the following query:

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

The results tab is selected, displaying the output:

EMPLOYEE and TITLE
EMANUEL, 101
JAY, 102
JANE, 105
PARTHI, 104
UMA, 102

Below the results, it says '5 rows returned in 0.01 seconds' and has 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. In the top navigation bar, 'APEX' is selected. Below it, tabs for 'SQL Workshop' and 'Team Development' are visible. On the right, a user profile for 'ABIRAMI PL abirami007' is shown. The main area is titled 'SQL Commands'. A code editor window displays the query:

```
1 SELECT
2  EMPLOYEE_ID||','||FIRST_NAME||',||LAST_NAME||',||EMAIL||',
3  ||PHONE_NUMBER||',||HIRE_DATE||',||JOB_ID||',||SALARY||',||COMMISSION_PCT||',
4  ||MANAGER_ID||',||DEPARTMENT_ID AS "THE_OUTPUT"
5 FROM EMPLOYEES;
6
```

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

THE_OUTPUT
1, SAM , EMANUEL , 12@gmail.com , 1235 , 02/02/1998 , 101 , 2000 , 120 , 9 , 87
2, ALEX , JAY , 987@gmail.com , 1675 , 02/02/1999 , 102 , 6000 , 187 , 4 , 89
4, JENNIE , JANE , 98@gmail.com , 1575 , 02/02/1999 , 105 , 9000 , 67 , 4 , 79
5, MEENA , PARTHI , 6798@gmail.com , 1585 , 02/02/1999 , 104 , 6000 , 67 , 4 , 79
3, JENNY , UMA , 123@gmail.com , 12345 , 02/02/1999 , 102 , 5000 , 1200 , 2 , 30

At the bottom left, it says '5 rows returned in 0.01 seconds'. There is also a 'Download' link.

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

RESULT :

RESTRICTING AND SORTING DATA

EX-NO : 5

DATE:

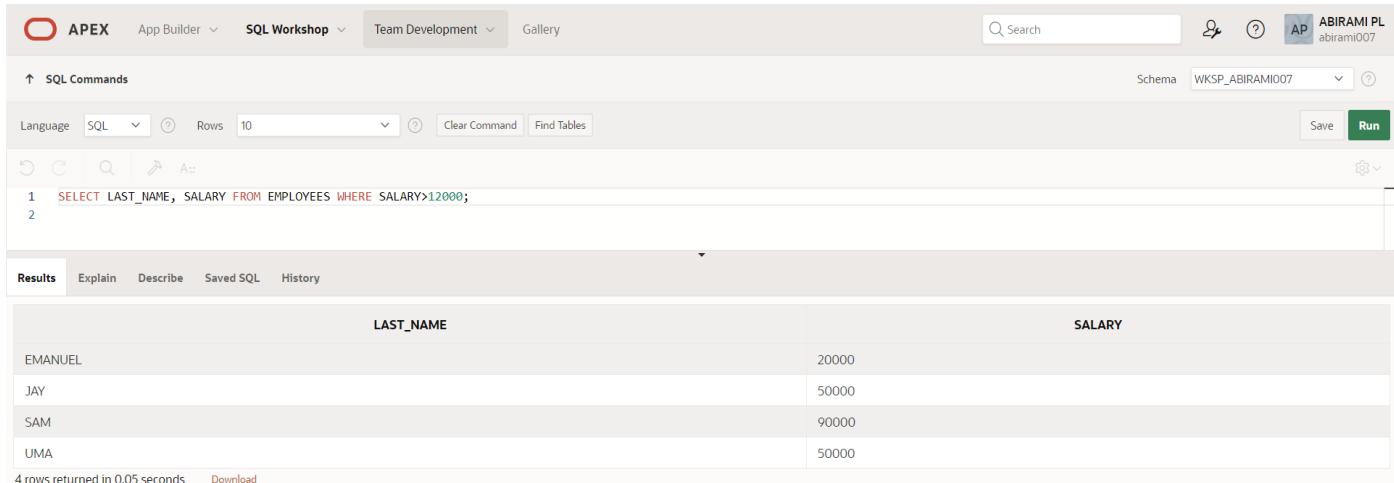
Find the Solution for the following :

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

QUERY:

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

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 ABIRAMI PL abirami007. The main area has tabs for SQL Commands, Explain, Describe, Saved SQL, and History. The SQL Commands tab shows the query: `SELECT LAST_NAME, SALARY FROM EMPLOYEES WHERE SALARY>12000;`. The results table has columns LAST_NAME and SALARY, displaying four rows: EMANUEL (20000), JAY (50000), SAM (90000), and UMA (50000). A note at the bottom says "4 rows returned in 0.05 seconds".

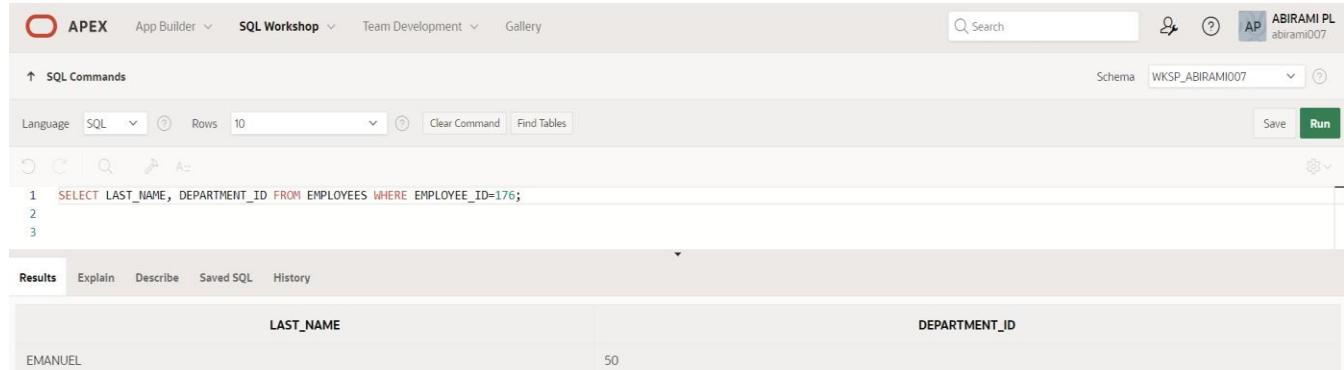
LAST_NAME	SALARY
EMANUEL	20000
JAY	50000
SAM	90000
UMA	50000

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

QUERY:

```
SELECT LAST_NAME, DEPARTMENT_ID FROM EMPLOYEES WHERE  
EMPLOYEE_ID=176;
```

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. On the right, there's a search bar, a user icon for 'ABIRAMI PL abirami007', and a 'Run' button. The main area has tabs for 'SQL Commands' and 'Results'. In the 'SQL Commands' tab, the query 'SELECT LAST_NAME, DEPARTMENT_ID FROM EMPLOYEES WHERE EMPLOYEE_ID=176;' is entered. The 'Results' tab is selected, displaying the output in a grid format:

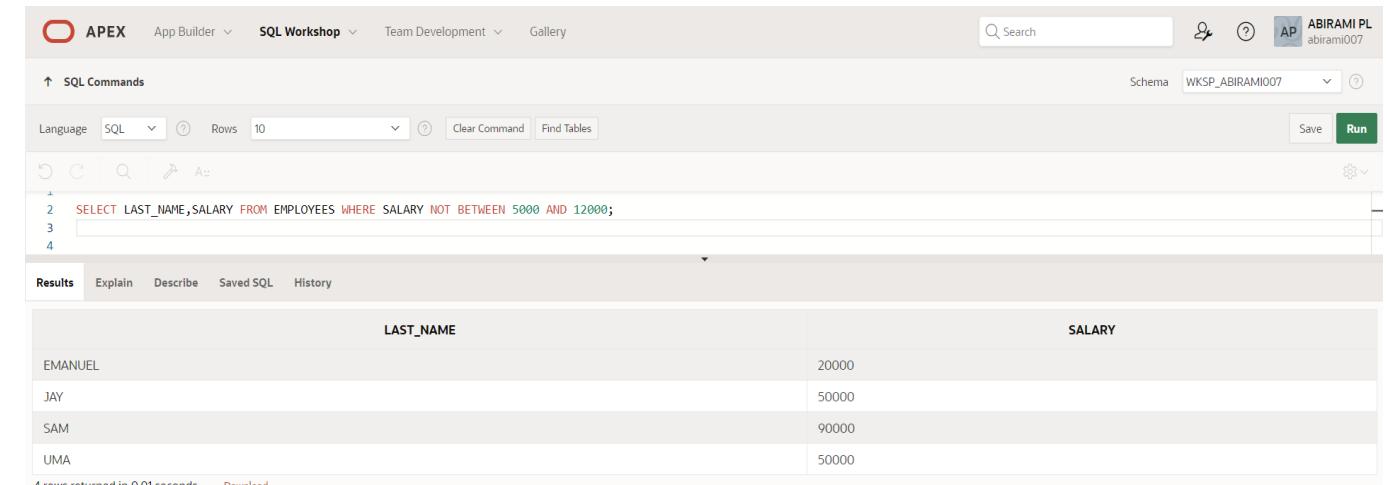
LAST_NAME	DEPARTMENT_ID
EMANUEL	50

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

QUERY:

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

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. On the right, there's a search bar, a user icon for 'ABIRAMI PL abirami007', and a 'Run' button. The main area has tabs for 'SQL Commands' and 'Results'. In the 'SQL Commands' tab, the query 'SELECT LAST_NAME,SALARY FROM EMPLOYEES WHERE SALARY NOT BETWEEN 5000 AND 12000;' is entered. The 'Results' tab is selected, displaying the output in a grid format:

LAST_NAME	SALARY
EMANUEL	20000
JAY	50000
SAM	90000
UMA	50000

4 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.

QUERY:

```
SELECT LAST_NAME, JOB_ID, TO_CHAR(HIRE_DATE, 'YYYY-MM-DD') AS HIRE_DATE  
FROM EMPLOYEES WHERE HIRE_DATE BETWEEN TO_DATE('1998-02-20','YYYY-MM-DD')  
AND TO_DATE('1998-05-01','YYYY-MM-DD') ORDER BY HIRE_DATE ;
```

OUTPUT:

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

```
1 SELECT LAST_NAME, JOB_ID, TO_CHAR(HIRE_DATE, 'YYYY-MM-DD') AS HIRE_DATE FROM EMPLOYEES WHERE HIRE_DATE  
2 BETWEEN TO_DATE('1998-02-20','YYYY-MM-DD') AND TO_DATE('1998-05-01','YYYY-MM-DD') ORDER BY HIRE_DATE ;
```

The Results tab displays the output:

LAST_NAME	JOB_ID	HIRE_DATE
JAY	102	1998-02-22
JANE	105	1998-03-06

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

QUERY:

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

OUTPUT:

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

```
1 SELECT LAST_NAME, DEPARTMENT_ID FROM EMPLOYEES WHERE DEPARTMENT_ID IN (20,50) ORDER BY LAST_NAME;  
2  
3
```

The Results tab displays the output:

LAST_NAME	DEPARTMENT_ID
EMANUEL	50
PARTHI	50

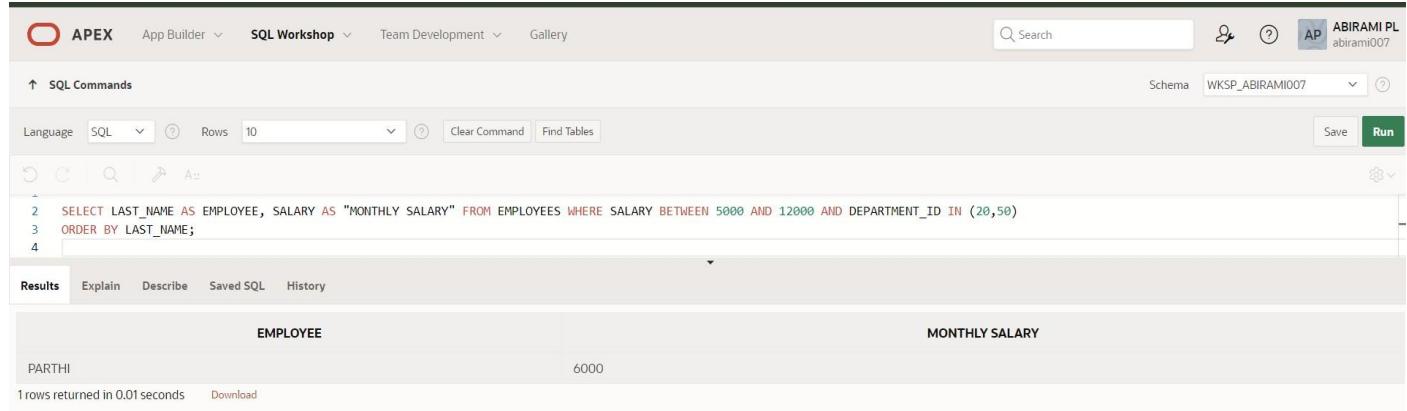
2 rows returned in 0.01 seconds [Download](#)

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. (Hint: between, in)

QUERY:

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

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there's a search bar, a user icon for 'ABIRAMI PL abirami007', and a 'Run' button. The main area is titled 'SQL Commands'. It shows the following code:

```
2 SELECT LAST_NAME AS EMPLOYEE, SALARY AS "MONTHLY SALARY" FROM EMPLOYEES WHERE SALARY BETWEEN 5000 AND 12000 AND DEPARTMENT_ID IN (20,50)  
3 ORDER BY LAST_NAME;  
4
```

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

EMPLOYEE	MONTHLY SALARY
PARTHI	6000

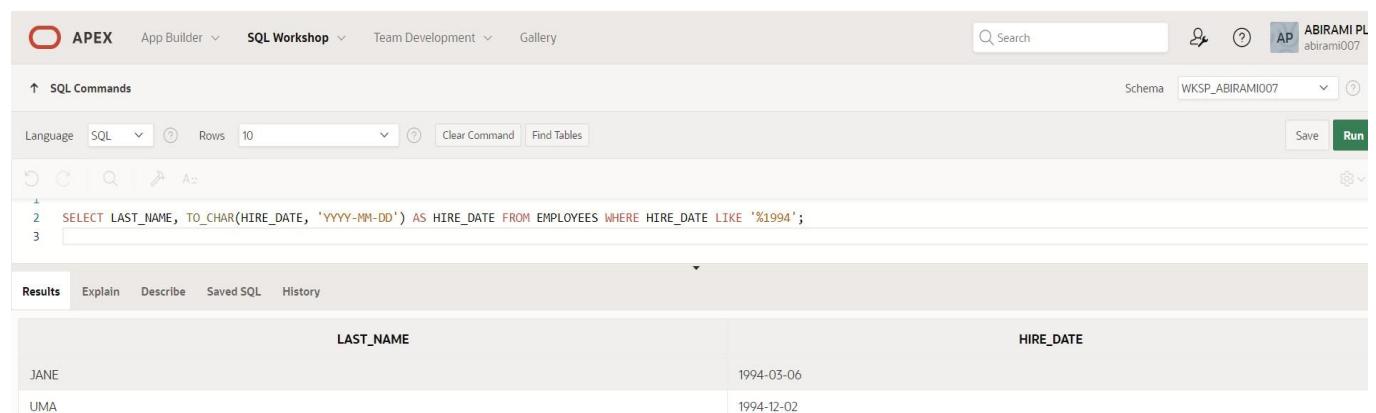
1 rows returned in 0.01 seconds. There is a 'Download' link at the bottom.

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

QUERY:

```
SELECT LAST_NAME, TO_CHAR(HIRE_DATE, 'YYYY-MM-DD') AS HIRE_DATE FROM EMPLOYEES 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, and Gallery. On the right, there's a search bar, a user icon for 'ABIRAMI PL abirami007', and a 'Run' button. The main area is titled 'SQL Commands'. It shows the following code:

```
2 SELECT LAST_NAME, TO_CHAR(HIRE_DATE, 'YYYY-MM-DD') AS HIRE_DATE FROM EMPLOYEES WHERE HIRE_DATE LIKE '%1994';  
3
```

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

LAST_NAME	HIRE_DATE
JANE	1994-03-06
UMA	1994-12-02

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

QUERY:

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

OUTPUT:

LAST_NAME	JOB_ID
SAM	102
UMA	102

9. Display the last name, salary, and commission for all employees who earn commissions. Sort data in descending order of salary and commissions. (Hint: is not null,order by)

QUERY:

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

OUTPUT:

LAST_NAME	SALARY	COMMISSION_PCT
SAM	90000	265
JANE	9000	67
PARTHI	6000	67
JAY	50000	187
UMA	50000	1200
EMANUEL	20000	120

6 rows returned in 0.01 seconds Download

10. Display the last name of all employees where the third letter of the name is *a*.

QUERY:

```
SELECT LAST_NAME FROM EMPLOYEES WHERE LAST_NAME LIKE '_A%';
```

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 ('ABIRAMI PL abirami007'). Below the navigation is a toolbar with icons forundo, redo, search, and run. The main area is titled 'SQL Commands' and contains the following code:

```
1
2  SELECT LAST_NAME FROM EMPLOYEES WHERE LAST_NAME LIKE '_A%';
3
4
5
```

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

LAST_NAME
EMANUEL
UMA

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

11. Display the last name of all employees who have an *a* and an *e* in their last name.

QUERY:

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

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', 'Gallery', a search bar, and user information ('ABIRAMI PL abirami007'). Below the navigation is a toolbar with icons forundo, redo, search, and run. The main area is titled 'SQL Commands' and contains the following code:

```
1
2  SELECT LAST_NAME FROM EMPLOYEES WHERE LAST_NAME LIKE '%A%' AND LAST_NAME LIKE '%E%';
3
```

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

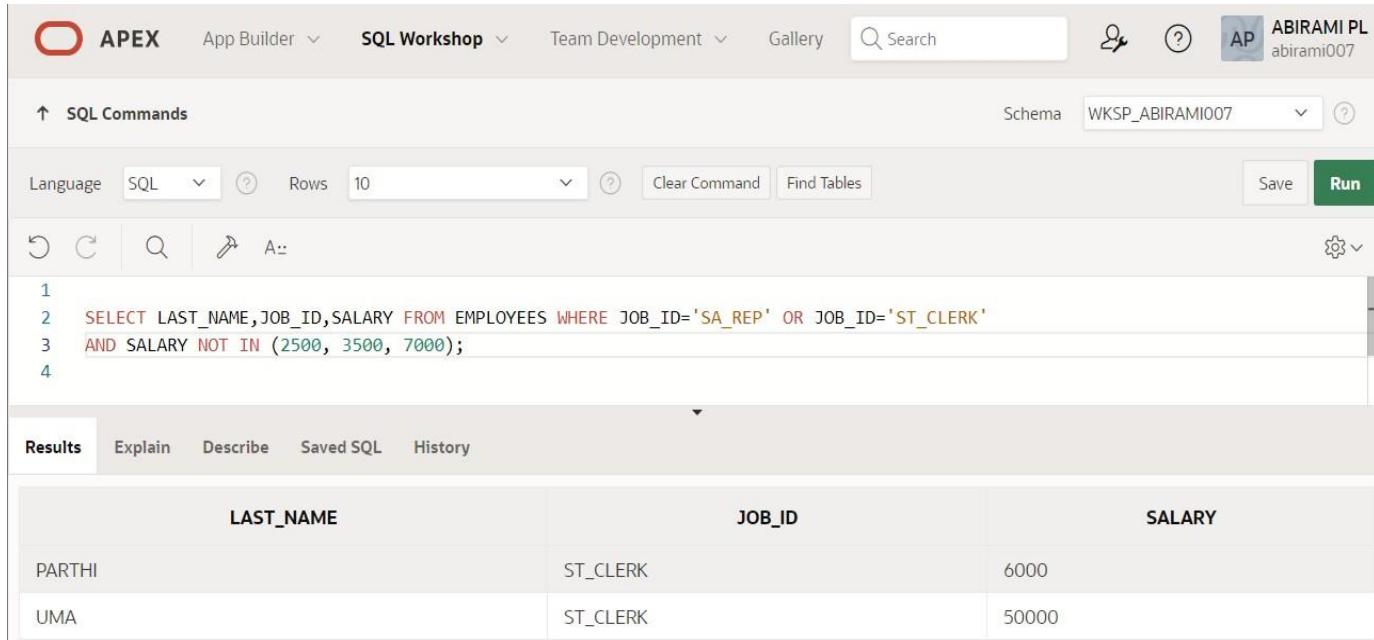
LAST_NAME
EMANUEL
JANE

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.

QUERY:

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

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 'ABIRAMI PL abirami007'. The main area is titled 'SQL Commands' with a schema dropdown set to 'WKSP_ABIRAMI007'. The SQL editor contains the following code:

```
1  
2 SELECT LAST_NAME,JOB_ID,SALARY FROM EMPLOYEES WHERE JOB_ID='SA_REP' OR JOB_ID='ST_CLERK'  
3 AND SALARY NOT IN (2500, 3500, 7000);  
4
```

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

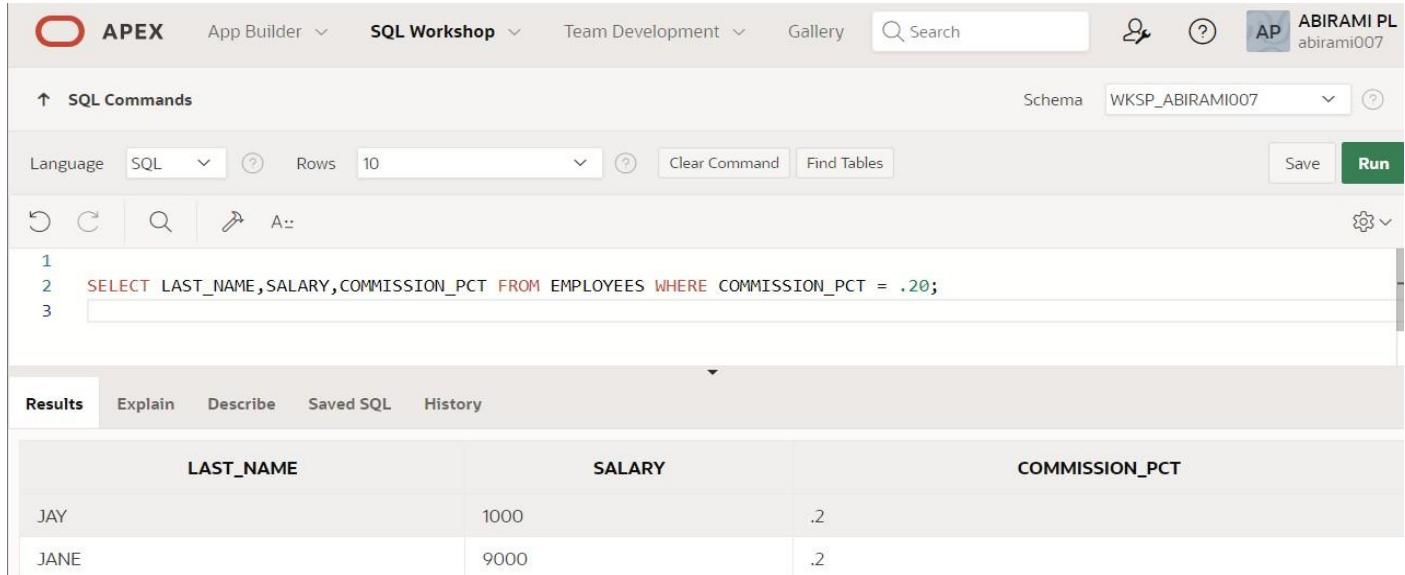
LAST_NAME	JOB_ID	SALARY
PARTHI	ST_CLERK	6000
UMA	ST_CLERK	50000

13. Display the last name, salary, and commission for all employees whose commission amount is 20%.

QUERY:

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

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 'ABIRAMI PL abirami007'. The main area is titled 'SQL Commands' with a schema dropdown set to 'WKSP_ABIRAMI007'. The SQL editor contains the following code:

```
1  
2 SELECT LAST_NAME,SALARY,COMMISSION_PCT FROM EMPLOYEES WHERE COMMISSION_PCT = .20;  
3
```

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

LAST_NAME	SALARY	COMMISSION_PCT
JAY	1000	.2
JANE	9000	.2

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 APEX, App Builder, SQL Workshop, Team Development, Gallery, a search bar, and user information (ABIRAMI PL abirami007). The main area is titled 'SQL Commands' with a schema dropdown set to WKSP_ABIRAMI007. The command input field contains the following SQL code:

```
1 SELECT TO_CHAR(SYSDATE,'YYYY-MM-DD') AS "DATE" FROM DUAL;
2
3 
```

The results section shows the output:

DATE
2024-04-25

Below the results, it says '1 rows returned in 0.02 seconds' and has a 'Download' link.

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

QUERY:

```
SELECT EMPLOYEE_ID, LAST_NAME, SALARY, SALARY+(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, Gallery, a search bar, and user information (ABIRAMI PL, abirami007). The SQL Commands tab is selected, showing the following SQL code:

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

Below the code, the Results tab is selected, displaying the query results in a grid format:

EMPLOYEE_ID	LAST_NAME	SALARY	NEW SALARY
176	EMANUEL	20000	23100
2	JAY	1000	1155
4	SAM	90000	103950
172	JANE	9000	10395
5	PARTHI	6000	6930
3	UMA	50000	57750

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' (selected), 'Team Development', 'Gallery', a search bar, and a user profile for 'ABIRAMI PL abirami007'. Below the toolbar, the schema is set to 'WKSP_ABIRAMI007'. The main area is titled 'SQL Commands' with a sub-section '1'. The query entered is:

```
1  
2 SELECT EMPLOYEE_ID, LAST_NAME, SALARY, SALARY+(SALARY*15.5/100) AS "NEW SALARY", (SALARY+(SALARY*15.5/100))-SALARY  
3 AS "INCREASE" FROM EMPLOYEES;  
4
```

Below the command area, there are tabs for 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History'. The 'Results' tab is selected, displaying the query's output as a table:

EMPLOYEE_ID	LAST_NAME	SALARY	NEW SALARY	INCREASE
176	EMANUEL	20000	23100	3100
2	JAY	1000	1155	155
4	SAM	90000	103950	13950
172	JANE	9000	10395	1395
5	PARTHI	6000	6930	930
3	UMA	50000	57750	7750

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', 'Team Development', 'Gallery', a search bar, and a user profile 'ABIRAMI PL abirami007'. Below the navigation is a toolbar with icons forundo, redo, search, and refresh, followed by 'SQL Commands', 'Schema (WKSP_ABIRAMI007)', 'Save', and a 'Run' button. The main area contains the SQL code and its results. The SQL code is:

```
2 SELECT INITCAP(LAST_NAME) AS "NAME", LENGTH(LAST_NAME) AS "LENGTH OF NAME" FROM
3 EMPLOYEES WHERE LAST_NAME LIKE 'J%' OR LAST_NAME LIKE 'A%' OR LAST_NAME LIKE 'M%'
4 ORDER BY LAST_NAME;
5
6
```

The results section shows a table with two rows:

NAME	LENGTH OF NAME
Jane	4
Jay	3

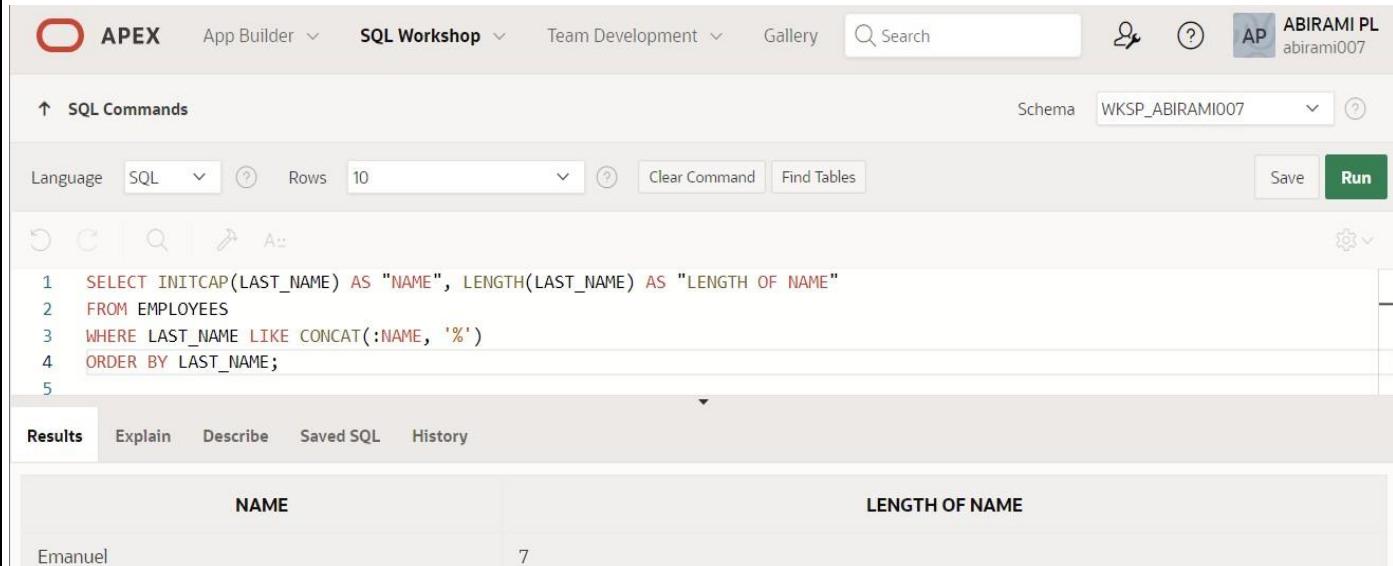
Below the table, it says '2 rows returned in 0.06 seconds' and has a 'Download' link.

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 for ABIRAMI PL abirami007. The SQL Commands tab is active. The schema is set to WKSP_ABIRAMI007. The SQL editor contains the following code:

```
1 SELECT INITCAP(LAST_NAME) AS "NAME", LENGTH(LAST_NAME) AS "LENGTH OF NAME"  
2 FROM EMPLOYEES  
3 WHERE LAST_NAME LIKE CONCAT(:NAME, '%')  
4 ORDER BY LAST_NAME;  
5
```

The Results tab is selected, displaying the output:

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 links for APEX, App Builder, SQL Workshop (which is currently selected), Team Development, and Gallery. There is also a search bar and user profile information for 'ABIRAMI PL abirami007'.

In the main workspace, under the 'SQL Commands' tab, the following SQL code is displayed:

```
1
2  SELECT LAST_NAME, ROUND(MONTHS_BETWEEN(SYSDATE,HIRE_DATE),0) MONTHS_WORKED
3  FROM EMPLOYEES ORDER BY 2;
4
```

The 'Run' button is highlighted in green at the bottom right of the command input area.

Below the command input, there are tabs for Results, Explain, Describe, Saved SQL, and History. The 'Results' tab is selected, showing the output of the query:

LAST_NAME	MONTHS_WORKED
PARTHI	303
JAY	314
EMANUEL	315
SAM	315
UMA	353
JANE	362

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, Team Development, Gallery, a search bar, and user information for ABIRAMI PL abirami007. Below the navigation is a toolbar with various icons and dropdowns for Language (SQL), Rows (10), Clear Command, Find Tables, Save, and Run. The main area displays the SQL command entered:

```
1 SELECT LAST_NAME||' EARNS $'||SALARY||' MONTHLY BUT WANTS  
2 $'||SALARY*3 "DREAM SALARY" FROM EMPLOYEES;
```

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

DREAM SALARY
EMANUEL EARNS \$20000 MONTHLY BUT WANTS \$60000
JAY EARNS \$1000 MONTHLY BUT WANTS \$3000
SAM EARNS \$90000 MONTHLY BUT WANTS \$270000
JANE EARNS \$9000 MONTHLY BUT WANTS \$27000
PARTHI EARNS \$6000 MONTHLY BUT WANTS \$18000
UMA EARNS \$50000 MONTHLY BUT WANTS \$150000

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, Gallery, a search bar, and user information (ABIRAMI PL, abirami007). Below the navigation is a toolbar with icons for Undo, Redo, Find, Sort, and Refresh, followed by a dropdown menu. The main area has tabs for SQL Commands, Explain, Describe, Saved SQL, and History. The SQL Commands tab is active, showing the query: `SELECT LAST_NAME, LPAD(SALARY,15,'$') SALARY FROM EMPLOYEES;`. The Results tab is selected, displaying the output in a grid format:

LAST_NAME	SALARY
EMANUEL	\$\$\$\$\$\$\$\$\$\$20000
JAY	\$\$\$\$\$\$\$\$\$\$1000
SAM	\$\$\$\$\$\$\$\$\$\$90000
JANE	\$\$\$\$\$\$\$\$\$\$9000
PARTHI	\$\$\$\$\$\$\$\$\$\$6000
UMA	\$\$\$\$\$\$\$\$\$\$50000

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

QUERY:

```
SELECT LAST_NAME, HIRE_DATE,  
TO_CHAR((NEXT_DAY(HIRE_DATE,'MONDAY')),'FMDAY," THE "DDSPTH "OF"  
MONTH,YYYY') AS "REVIEW" 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 (ABIRAMI PL, abirami007). The main area is titled 'SQL Commands' with a schema dropdown set to WKSP_ABIRAMI007. The SQL editor contains the following code:

```
1 SELECT LAST_NAME, HIRE_DATE, TO_CHAR((NEXT_DAY(HIRE_DATE, 'MONDAY')),'FMDAY," THE "DDSPTH "OF" MONTH,YYYY') AS "REVIEW"  
2 FROM EMPLOYEES;  
3
```

The 'Run' button is highlighted in green. Below the editor, the 'Results' tab is selected, showing the output of the query:

LAST_NAME	HIRE_DATE	REVIEW
EMANUEL	02/02/1998	MONDAY, THE NINTH OF FEBRUARY,1998
JAY	02/22/1998	MONDAY, THE TWENTY-THIRD OF FEBRUARY,1998
SAM	02/02/1998	MONDAY, THE NINTH OF FEBRUARY,1998
JANE	03/06/1994	MONDAY, THE SEVENTH OF MARCH,1994
PARTHI	02/02/1999	MONDAY, THE EIGHTH OF FEBRUARY,1999
UMA	12/02/1994	MONDAY, THE FIFTH OF DECEMBER,1994

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

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', 'Gallery', a search bar, and user information 'ABIRAMI PL abirami007'. Below the toolbar, the 'SQL Commands' tab is active, with 'Schema' set to 'WKSP_ABIRAMI007'. The command input area contains the following SQL code:

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

The results section displays the output of the query:

LAST_NAME	HIRE_DATE	DAY
EMANUEL	02/02/1998	MONDAY
SAM	02/02/1998	MONDAY
PARTHI	02/02/1999	TUESDAY
UMA	12/02/1994	FRIDAY
JANE	03/06/1994	SUNDAY
JAY	02/22/1998	SUNDAY

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 :

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, DEPT D  
WHERE E.DEPARTMENT_ID = D.DEPT_ID;
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows a user profile for ABIRAMI PL (abirami007). The main area has tabs for SQL Commands, Results, Explain, Describe, Saved SQL, and History. The SQL Commands tab contains the following query:

```
1 SELECT E.LAST_NAME, E.DEPARTMENT_ID, D.DEPT_NAME  
2 FROM EMPLOYEES E, DEPT D  
3 WHERE E.DEPARTMENT_ID = D.DEPT_ID;
```

The Results tab displays the output of the query:

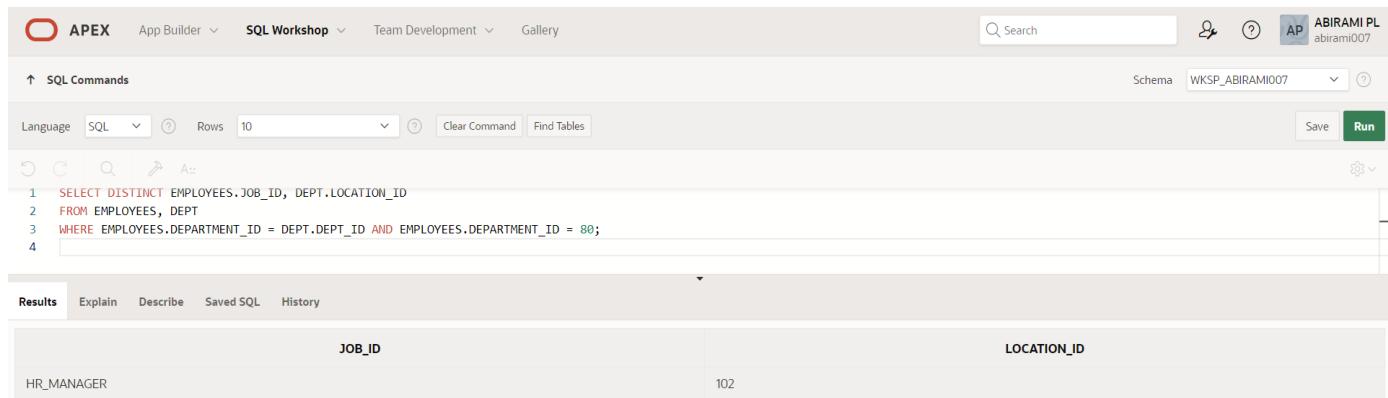
LAST_NAME	DEPARTMENT_ID	DEPT_NAME
EMANUEL	50	MARKETING
JAY	39	HR
JANE	39	HR
PARTHI	50	MARKETING

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 EMPLOYEES.JOB_ID, DEPT.LOCATION_ID  
FROM EMPLOYEES, DEPT  
WHERE EMPLOYEES.DEPARTMENT_ID = DEPT.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 (selected), Team Development, and Gallery. On the right, there's a search bar, a user icon for 'ABIRAMI PL abirami007', and a 'Run' button. Below the toolbar, the schema is set to 'WKSP_ABIRAMI007'. The main area has tabs for SQL Commands, SQL (selected), and Clear Command. The SQL pane contains the query:

```
1 SELECT DISTINCT EMPLOYEES.JOB_ID, DEPT.LOCATION_ID  
2 FROM EMPLOYEES, DEPT  
3 WHERE EMPLOYEES.DEPARTMENT_ID = DEPT.DEPT_ID AND EMPLOYEES.DEPARTMENT_ID = 80;  
4
```

The Results tab is selected, displaying the output:

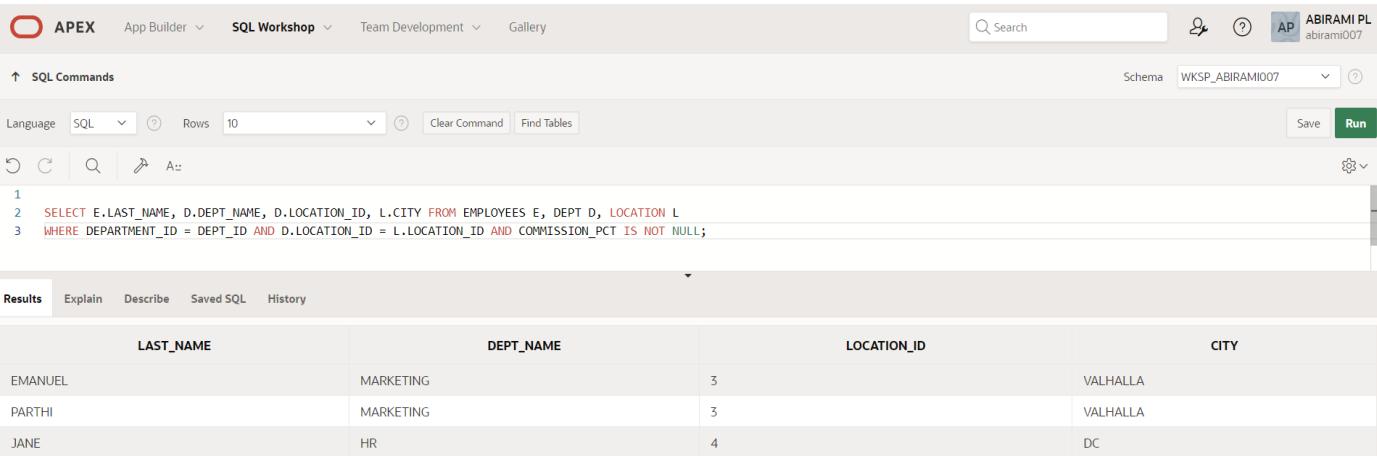
JOB_ID	LOCATION_ID
HR_MANAGER	102

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

QUERY:

```
SELECT E.LAST_NAME, D.DEPARTMENT_NAME, D.LOCATION_ID, L.CITY FROM  
EMPLOYEES E, DEPT 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 (selected), Team Development, and Gallery. On the right, there's a search bar, a user icon for 'ABIRAMI PL abirami007', and a 'Run' button. Below the toolbar, the schema is set to 'WKSP_ABIRAMI007'. The main area is titled 'SQL Commands' with tabs for Language (set to SQL), Rows (set to 10), and other options like Clear Command and Find Tables. The SQL editor contains the three-line query from the question. The results tab is selected, showing a table with four columns: LAST_NAME, DEPT_NAME, LOCATION_ID, and CITY. The data returned is as follows:

LAST_NAME	DEPT_NAME	LOCATION_ID	CITY
EMANUEL	MARKETING	3	VALHALLA
PARTHI	MARKETING	3	VALHALLA
JANE	HR	4	DC

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, DEPT WHERE  
DEPARTMENT_ID = DEPT_ID AND LAST_NAME LIKE '%a%';
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows a user profile for ABIRAMI PL abirami007. The main area is titled "SQL Commands". The language is set to SQL, and the results are displayed over 10 rows. The query entered is:

```
1 SELECT LAST_NAME, DEPT_NAME FROM EMPLOYEES, DEPT WHERE DEPARTMENT_ID = DEPT_ID AND LAST_NAME LIKE '%a%';  
2  
3
```

The results section shows the output of the query:

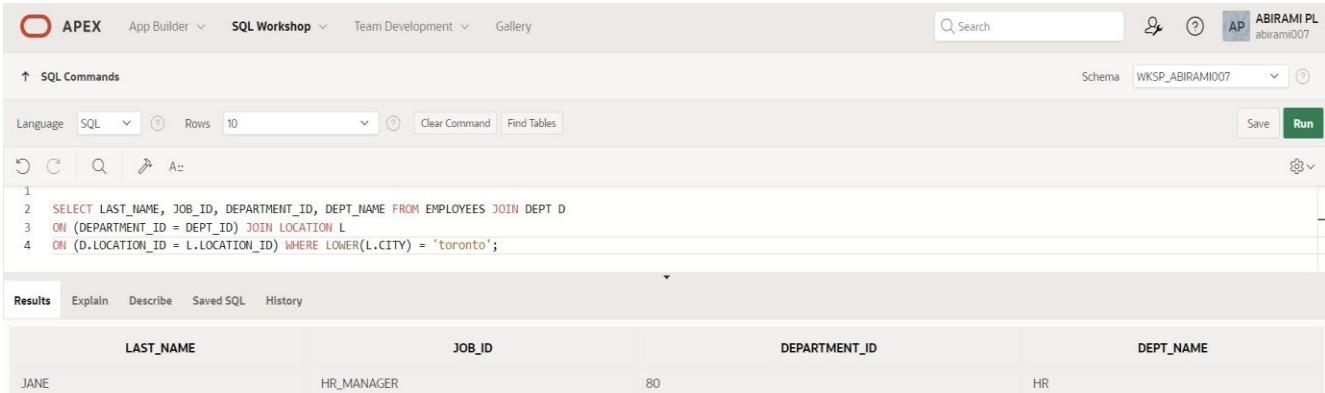
LAST_NAME	DEPT_NAME
EMANUEL	MARKETING
JAY	HR
JANE	HR
PARTHI	MARKETING

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 DEPT D
ON (DEPARTMENT_ID = DEPT_ID) JOIN LOCATION L
ON (D.LOCATION_ID = L.LOCATION_ID) WHERE LOWER(L.CITY) = 'toronto';
```

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, and Gallery. The right side shows a user profile for ABIRAMI PL abirami007 and a schema dropdown for WKSP_ABIRAMI007. The main area has tabs for SQL Commands, Results, Explain, Describe, Saved SQL, and History. The SQL Commands tab contains the following SQL code:

```
1
2  SELECT LAST_NAME, JOB_ID, DEPARTMENT_ID, DEPT_NAME FROM EMPLOYEES JOIN DEPT D
3  ON (DEPARTMENT_ID = DEPT_ID) JOIN LOCATION L
4  ON (D.LOCATION_ID = L.LOCATION_ID) WHERE LOWER(L.CITY) = 'toronto';
```

The Results tab displays the query results in a table:

LAST_NAME	JOB_ID	DEPARTMENT_ID	DEPT_NAME
JANE	HR_MANAGER	80	HR

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 (selected), Team Development, and Gallery. The right side shows a user profile for ABIRAMI PL (abirami007). The main area has tabs for SQL Commands, Results, Explain, Describe, Saved SQL, and History. The SQL Commands tab contains the following code:

```
1 SELECT W.LAST_NAME "EMPLOYEE", W.EMPLOYEE_ID "EMP#", M.LAST_NAME "MANAGER", M.EMPLOYEE_ID "MGR#"  
2 FROM EMPLOYEES W JOIN EMPLOYEES M ON (W.MANAGER_ID = M.EMPLOYEE_ID);  
3  
4
```

The Results tab displays the query results in a grid:

EMPLOYEE	EMP#	MANAGER	MGR#
JAY	2	SAM	4
JANE	172	SAM	4
PARTHI	5	SAM	4

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

7. Modify lab4_6.sql to display all employees including King, who has no manager. Order the results by the employee number.

QUERY:

```
SELECT W.LAST_NAME "EMPLOYEE", W.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 (selected), Team Development, and Gallery. The right side shows the user ABIRAMI PL and session details. The main area has tabs for SQL Commands, Results, Explain, Describe, Saved SQL, and History. The SQL Commands tab contains the executed query:

```
1  
2 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)  
3 ORDER BY W.EMPLOYEE_ID;  
4
```

The Results tab displays the query output as a table:

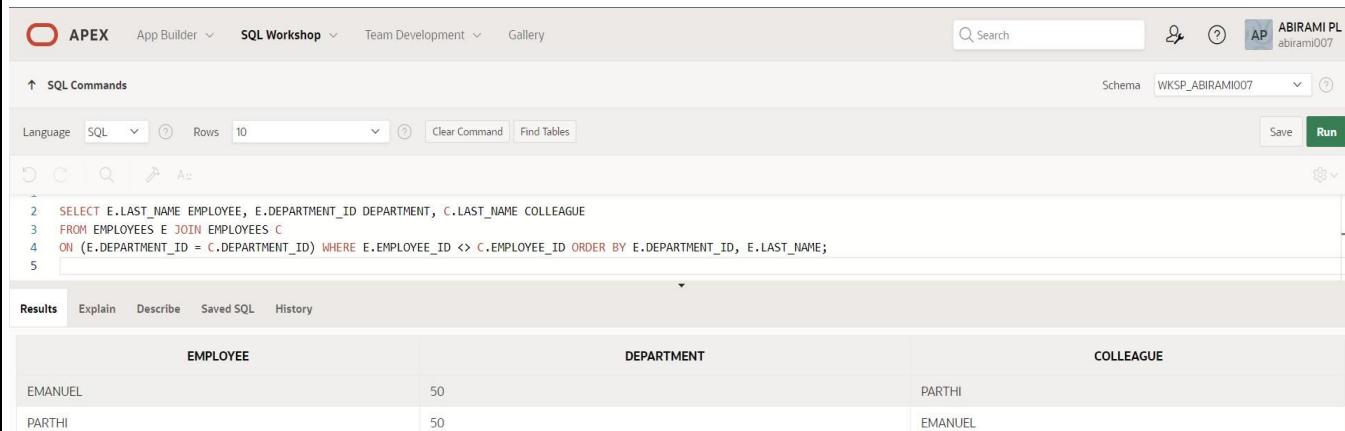
EMPLOYEE	EMP#	MANAGER	MGR#
JAY	2	SAM	4
UMA	3	-	-
SAM	4	-	-
PARTHI	5	SAM	4
JANE	172	SAM	4
EMANUEL	176	-	-

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 (selected), Team Development, and Gallery. The right side shows the user ABIRAMI PL and session ID abirami007. The main area has tabs for SQL Commands, Results, Explain, Describe, Saved SQL, and History. The SQL Commands tab contains the executed query:

```
2 SELECT E.LAST_NAME EMPLOYEE, E.DEPARTMENT_ID DEPARTMENT, C.LAST_NAME COLLEAGUE
3 FROM EMPLOYEES E JOIN EMPLOYEES C
4 ON (E.DEPARTMENT_ID = C.DEPARTMENT_ID) WHERE E.EMPLOYEE_ID <> C.EMPLOYEE_ID ORDER BY E.DEPARTMENT_ID, E.LAST_NAME;
5
```

The Results tab displays the output:

EMPLOYEE	DEPARTMENT	COLLEAGUE
EMANUEL	50	PARTHI
PARTHI	50	EMANUEL

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:

```
DESC JOB_GRADES;
```

```
SELECT E.LAST_NAME, E.JOB_ID, D.DEPT_NAME, E.SALARY, J.GRADE_LEVEL
FROM EMPLOYEES E JOIN DEPT 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 two panels of the Oracle SQL Workshop interface. The top panel displays the description of the JOB_GRADES table, and the bottom panel shows the execution results of a query.

Top Panel (Describe View):

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
JOB_GRADES	GRADE_LEVEL	VARCHAR2	2	-	-	-	✓	-	-
	LOWEST_SAL	NUMBER	22	-	-	-	✓	-	-
	HIGHEST_SAL	NUMBER	22	-	-	-	✓	-	-

Bottom Panel (Query Results):

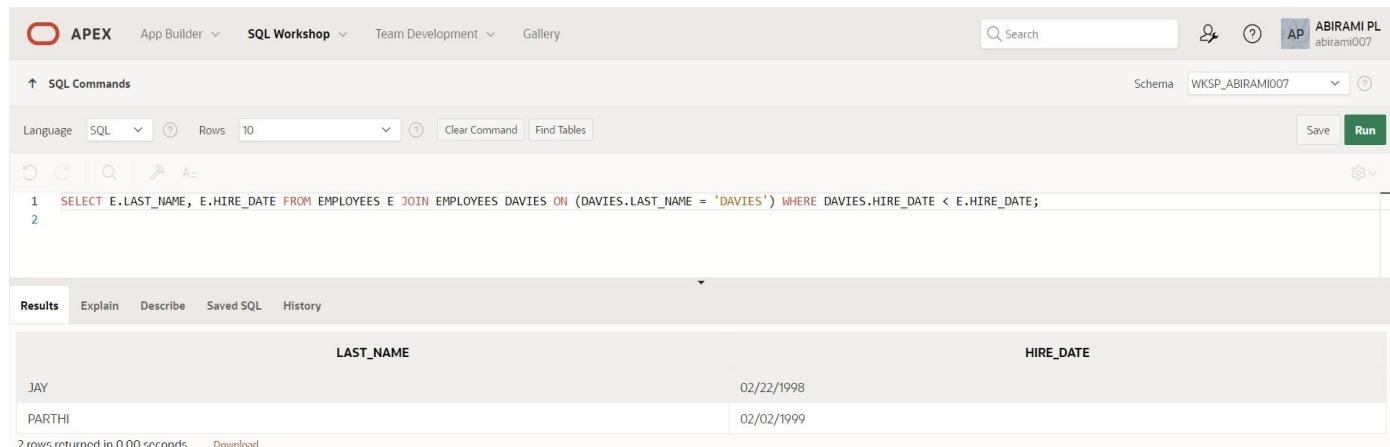
LAST_NAME	JOB_ID	DEPT_NAME	SALARY	GRADE_LEVEL
PARTHI	ST_CLERK	MARKETING	6000	A
JANE	HR_MANAGER	HR	9000	A
EMANUEL	FL_MANAGER	MARKETING	20000	D

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 top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there's a search bar, a user icon for ABIRAMI PL abirami007, and a schema dropdown set to WKSP_ABIRAMI007. Below the toolbar, the SQL Commands tab is selected, showing the following SQL code:

```
1 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;  
2
```

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

LAST_NAME	HIRE_DATE
JAY	02/22/1998
PARTHI	02/02/1999

Below the table, it says "2 rows returned in 0.00 seconds." and has a "Download" link.

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 'ABIRAMI PL abirami007' and a schema dropdown 'WKSP_ABIRAMI007'. The main area has tabs for 'SQL Commands' and 'Results'. Under 'SQL Commands', the query is pasted:

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

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

EMPLOYEE	EMP HIRED	MANAGER	MGR HIRED
JANE	03/06/1994	DAVIES	02/02/1998

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

RESULT :

AGGREGATING DATA USING GROUP FUNCTIONS

EX-NO : 8

DATE:

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

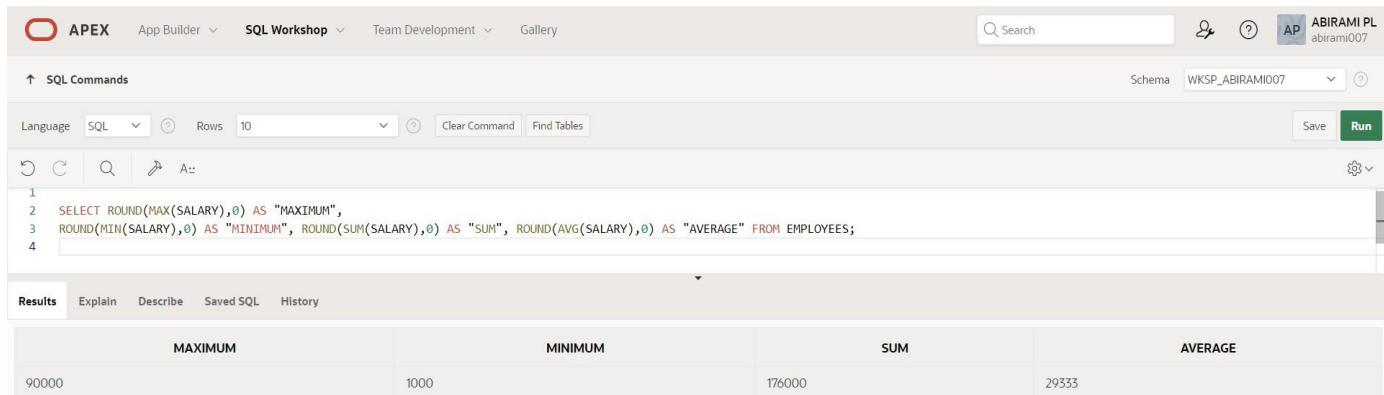
FALSE

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. The right side shows a user profile for ABIRAMI PL abirami007. The main area has tabs for SQL Commands and Results. In the SQL Commands tab, the query is pasted and executed. The Results tab displays the output in a grid format.

MAXIMUM	MINIMUM	SUM	AVERAGE
90000	1000	176000	29333

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. In the SQL Commands pane, two lines of SQL code are entered:

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

The Results pane displays the output of the query:

JOB_ID	MAXIMUM	MINIMUM	SUM	AVERAGE
FI_MANAGER	20000	20000	20000	20000
ST_CLERK	6000	50000	56000	28000
SL_REP	1000	1000	1000	1000
DESIGNER	90000	90000	90000	90000
HR_MANAGER	9000	9000	9000	9000

Below the table, it says "5 rows returned in 0.01 seconds".

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

QUERY:

```
SELECT JOB_ID,COUNT(JOB_ID) 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 (selected), Team Development, and Gallery. The right side shows a user profile for ABIRAMI PL abirami007. The main area is titled "SQL Commands". The command entered is:

```
1 | SELECT JOB_ID,COUNT(JOB_ID) FROM EMPLOYEES GROUP BY JOB_ID;  
2 |
```

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

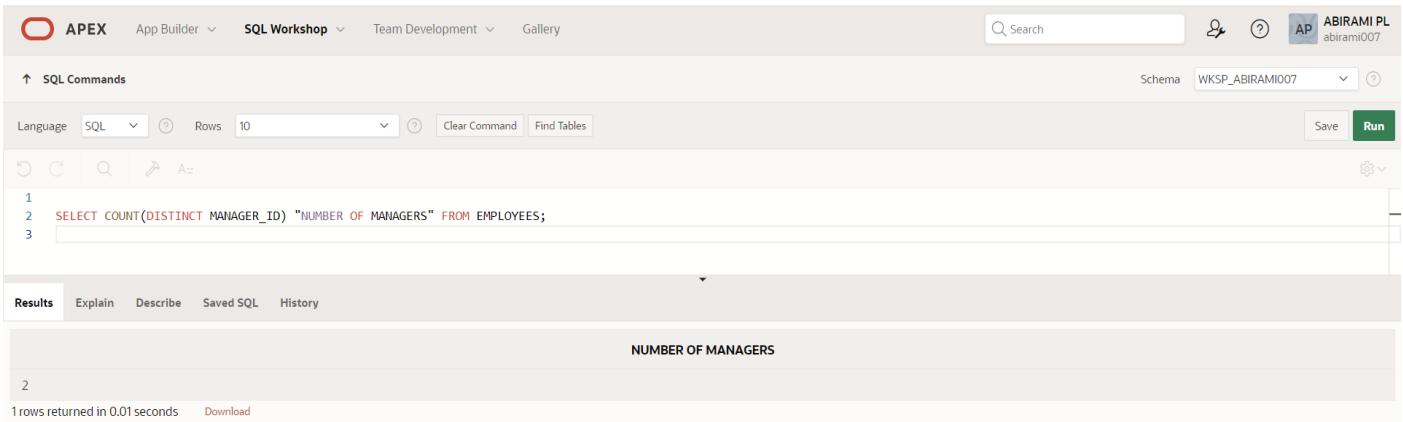
JOB_ID	COUNT(JOB_ID)
FI_MANAGER	1
ST_CLERK	2
SL REP	1
DESIGNER	1
HR_MANAGER	1

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

```
1  
2  SELECT COUNT(DISTINCT MANAGER_ID) "NUMBER OF MANAGERS"  
3   FROM EMPLOYEES;
```

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

NUMBER OF MANAGERS
2

Below the table, it says "1 rows returned in 0.01 seconds".

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

QUERY:

```
SELECT MAX(SALARY)-MIN(SALARY) AS "DIFFERENCE" 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 for ABIRAMI PL abirami007. The main area is titled 'SQL Commands' with tabs for Language (selected), SQL, Clear Command, and Find Tables. The schema is set to WKSP_ABIRAMI007. The query entered is:

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

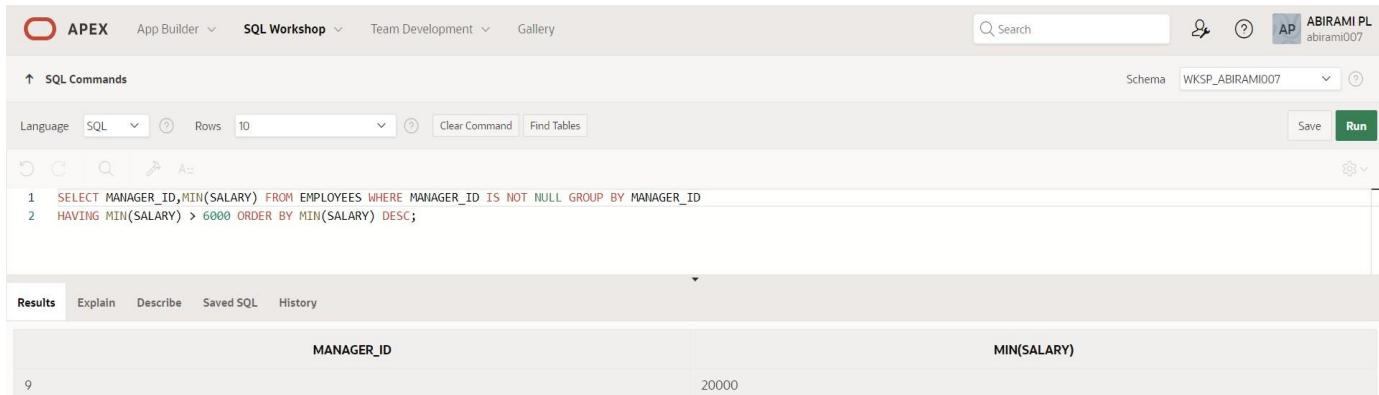
The results tab is selected, showing a single row with the heading 'DIFFERENCE' and the value '89000'.

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 top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. On the right, there's a search bar, a user icon for 'ABIRAMI PL abirami007', and a 'Save' button. The main area is titled 'SQL Commands'. It shows the following SQL code:

```
1 SELECT MANAGER_ID,MIN(SALARY) FROM EMPLOYEES WHERE MANAGER_ID IS NOT NULL GROUP BY MANAGER_ID  
2 HAVING MIN(SALARY) > 6000 ORDER BY MIN(SALARY) DESC;
```

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

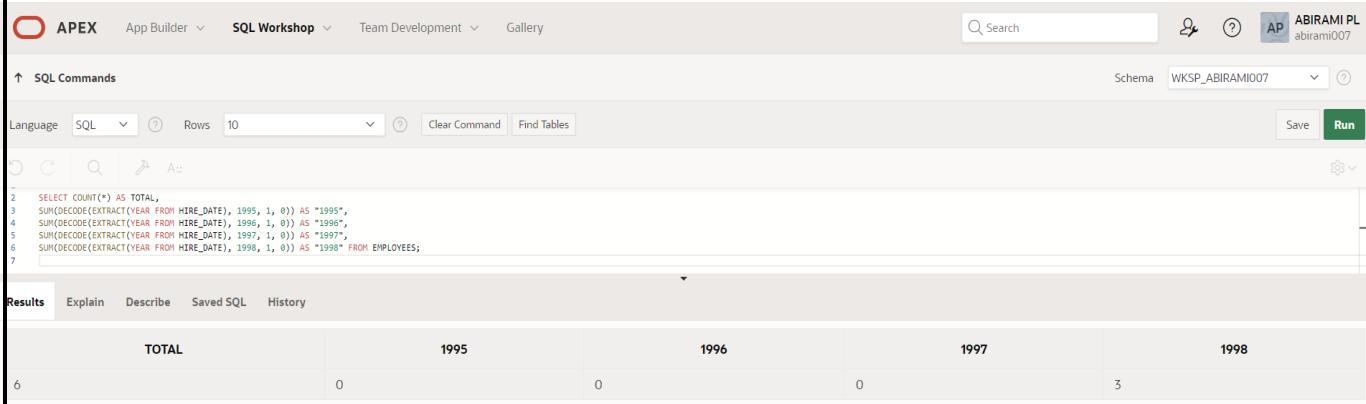
MANAGER_ID	MIN(SALARY)
9	20000

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 top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there's a search bar, a user icon for ABIRAMI PL abirami007, and a schema dropdown set to WKSP_ABIRAMI007. Below the toolbar, the SQL Commands tab is active, showing the query code. The code is as follows:

```
2  SELECT COUNT(*) AS TOTAL,  
3    SUM(DECODE(EXTRACT(YEAR FROM HIRE_DATE), 1995, 1, 0)) AS "1995",  
4    SUM(DECODE(EXTRACT(YEAR FROM HIRE_DATE), 1996, 1, 0)) AS "1996",  
5    SUM(DECODE(EXTRACT(YEAR FROM HIRE_DATE), 1997, 1, 0)) AS "1997",  
6    SUM(DECODE(EXTRACT(YEAR FROM HIRE_DATE), 1998, 1, 0)) AS "1998" FROM EMPLOYEES;  
7
```

Below the code, the Results tab is selected, displaying the query results in a grid format:

TOTAL	1995	1996	1997	1998
6	0	0	0	3

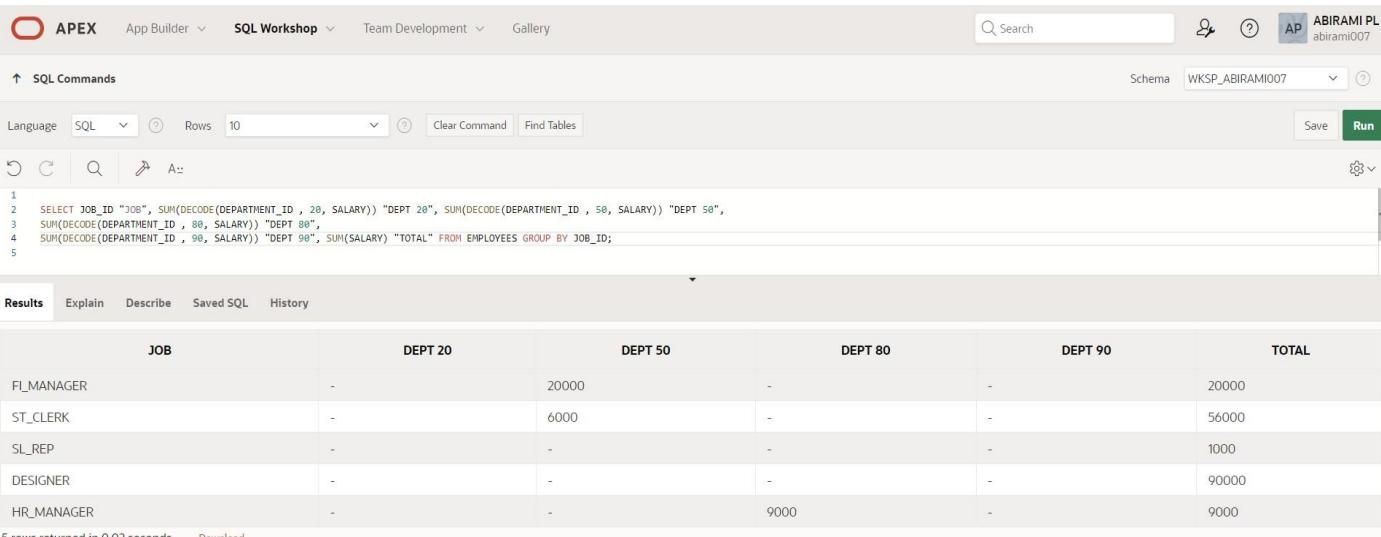
A status message at the bottom indicates the query was executed in 0.00 seconds.

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' (selected), 'Team Development', and 'Gallery'. The right side shows a user profile for 'ABIRAMI PL abirami007'. The main area has tabs for 'SQL Commands' and 'Results'. In the 'SQL Commands' tab, the query is pasted. In the 'Results' tab, the output is displayed in a grid format:

JOB	DEPT 20	DEPT 50	DEPT 80	DEPT 90	TOTAL
FI_MANAGER	-	20000	-	-	20000
ST_CLERK	-	6000	-	-	56000
SL REP	-	-	-	-	1000
DESIGNER	-	-	-	-	90000
HR_MANAGER	-	-	9000	-	9000

Below the table, it says '5 rows returned in 0.02 seconds' and there is a 'Download' link.

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

QUERY:

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

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 is a search bar, user information (ABIRAMI PL, abirami007), and a schema dropdown set to WKSP_ABIRAMI007. Below the toolbar, the SQL Commands section shows the following code:

```
1  
2  SELECT D.DEPT_NAME "NAME", D.LOCATION_ID "LOCATION ", COUNT(*) "NUMBER OF PEOPLE", ROUND(AVG(SALARY),2) "SALARY"  
3  FROM EMPLOYEES E, DEPT D WHERE E.DEPARTMENT_ID = D.DEPT_ID GROUP BY D.DEPT_NAME, D.LOCATION_ID;  
4
```

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

NAME	LOCATION	NUMBER OF PEOPLE	SALARY
HR	105	1	1000
HR	6	1	9000
MARKETING	3	2	13000

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

RESULT :

SUB-QUERIES

EX-NO : 9

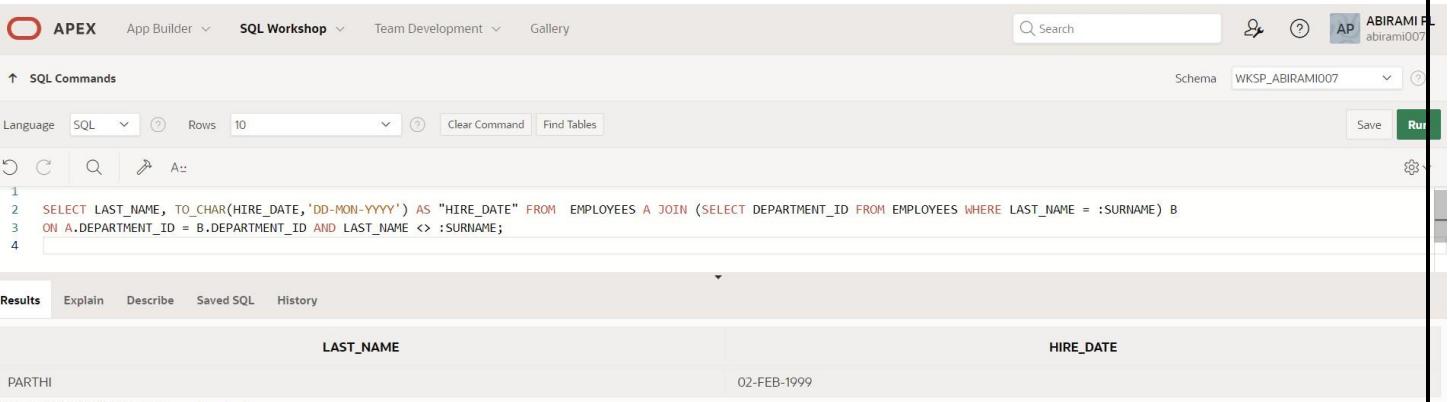
DATE:

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

QUERY:

```
SELECT LAST_NAME, 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 top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. On the right, there's a search bar, a help icon, and a session identifier 'ABIRAMI abirami007'. Below the toolbar, the schema is set to 'WKSP_ABIRAMI007'. The main area is titled 'SQL Commands' with options for 'Language' (set to 'SQL'), 'Rows' (set to 10), and 'Clear Command'. The SQL editor contains the following code:

```
1
2  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
3  ON A.DEPARTMENT_ID = B.DEPARTMENT_ID AND LAST_NAME <> :SURNAME;
4
```

The results section shows a single row returned by the query:

LAST_NAME	HIRE_DATE
PARTHI	02-FEB-1999

A note at the bottom indicates the query was returned in 0.01 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 (selected), Team Development, and Gallery. The right side shows a user profile for ABIRAMI P. abirami007. The main area has tabs for SQL Commands and Results. In the SQL Commands tab, the query is entered and run. In the Results tab, the output is displayed as a table with columns: EMPLOYEE_ID, LAST_NAME, and SALARY. Two rows are shown: one for employee 3 (LAST_NAME UMA, SALARY 50000) and one for employee 4 (LAST_NAME DAVIES, SALARY 90000).

	EMPLOYEE_ID	LAST_NAME	SALARY
3		UMA	50000
4		DAVIES	90000

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' (selected), 'Team Development', and 'Gallery'. On the right, there's a search bar, user information for 'ABIRAMI PL abirami007', and a 'Run' button. Below the toolbar, the 'SQL Commands' tab is active, showing the following code in the command window:

```
1 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;
2
3
```

Below the code, the 'Results' tab is selected, displaying the query results in a grid:

EMPLOYEE_ID	LAST_NAME
176	EMANUEL
5	PARTHI
3	UMA

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 DEPT WHERE LOCATION_ID=1700) B ON
A.DEPARTMENT_ID=B.DEPT_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's a search bar, a help icon, and a session identifier "ABIRAMI PL abirami007". 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 Undo, Redo, Search, and other utilities. The SQL editor contains the following code:

```
1
2  SELECT LAST_NAME, DEPARTMENT_ID, JOB_ID FROM EMPLOYEES A
3  JOIN (SELECT DEPT_ID FROM DEPT WHERE LOCATION_ID=1700) B ON A.DEPARTMENT_ID=B.DEPT_ID;
4
```

Below the editor, there are tabs for Results, Explain, Describe, Saved SQL, and History. The Results tab is selected and displays the query results in a table:

LAST_NAME	DEPARTMENT_ID	JOB_ID
JAY	39	SL REP

At the bottom left, it says "1 rows returned in 0.01 seconds". There are also "Download" and "Run" buttons at the bottom right of the results panel.

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 top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows a user profile for ABIRAMI P abirami007. The main area has a search bar and a schema dropdown set to WKSP_ABIRAMI007. The SQL Commands section shows the query being run, with lines 1 through 4 visible. The Results tab is selected, displaying a table with three rows: JAY, JANE, and PARTHI, each with their corresponding salaries. The bottom status bar indicates 5 rows returned in 0.01 seconds and provides a download link.

LAST_NAME	SALARY
JAY	1000
JANE	9000
PARTHI	6000

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

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. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows a user profile for ABIRAMI PL (abirami007). The main area has a search bar and a schema dropdown set to WKSP_ABIRAMI007. Below is a toolbar with Language (SQL selected), Rows (10), Clear Command, Find Tables, Save, and Run buttons. The SQL editor contains two lines of code: a comment '1' and the SELECT statement. The results tab is active, displaying a single row of data from the query execution.

DEPARTMENT_ID	LAST_NAME	JOB_ID
39	JAY	SL REP

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. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows a user profile for ABIRAMI PL (abirami007). The main area has tabs for SQL Commands, Results (selected), Explain, Describe, Saved SQL, and History. The SQL Commands tab shows the query code. The Results tab displays the output table with three columns: EMPLOYEE_ID, LAST_NAME, and SALARY. One row is shown, corresponding to the query result.

EMPLOYEE_ID	LAST_NAME	SALARY
3	UMA	50000

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

RESULT :

USING THE SET OPERATORS

EX-NO : 10

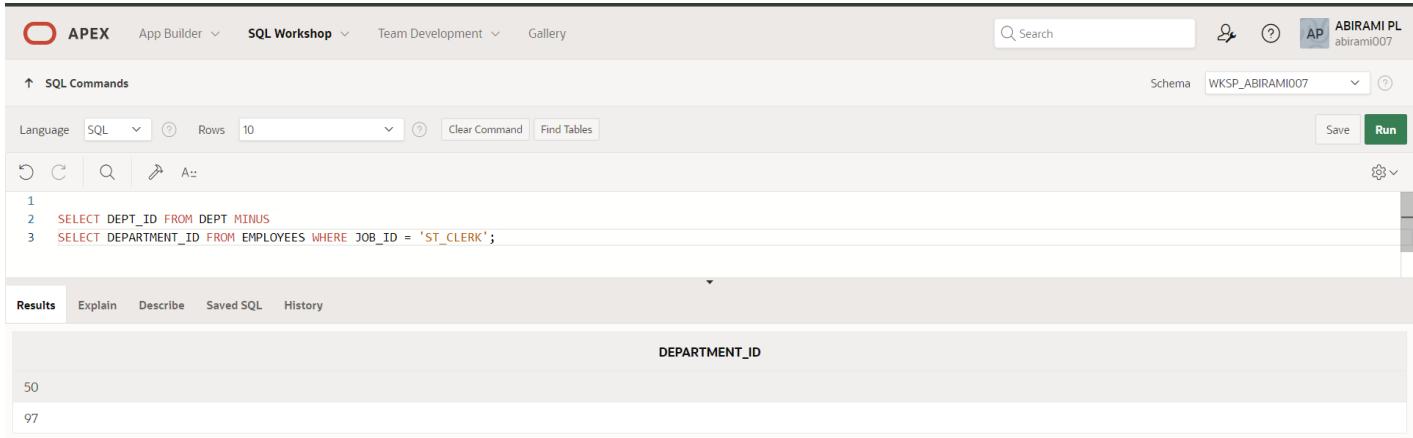
DATE:

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. On the right, there's a search bar, a user icon for ABIRAMI PL abirami007, and a Run button. The main area is titled 'SQL Commands' with options for Language (SQL selected), Rows (10), Clear Command, and Find Tables. Below this, the SQL code is displayed:

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

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

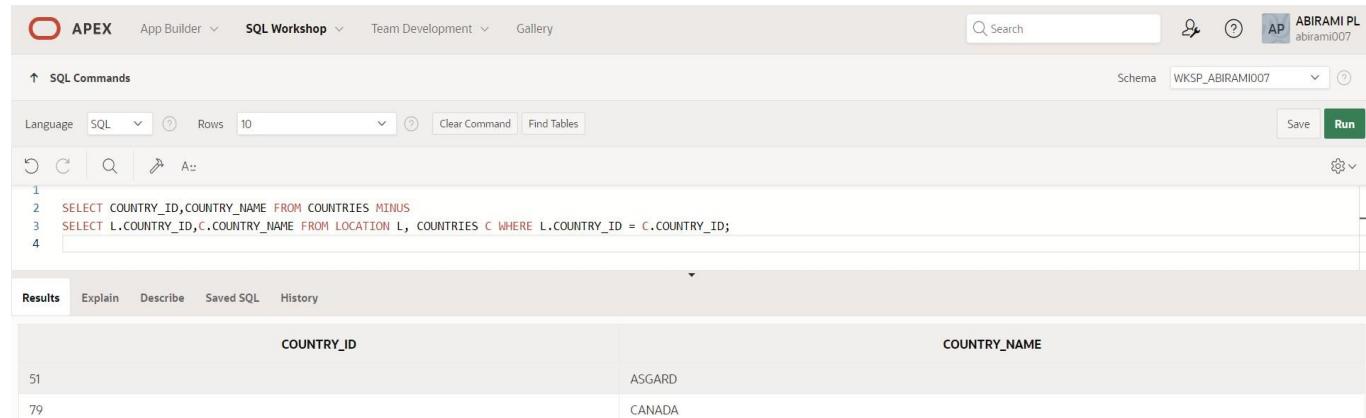
DEPARTMENT_ID
50
97

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 APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows a user profile for ABIRAMI PL (abirami007). The main area has tabs for SQL Commands, Results, Explain, Describe, Saved SQL, and History. The SQL Commands tab displays the following code:

```
1  
2 SELECT COUNTRY_ID,COUNTRY_NAME FROM COUNTRIES MINUS  
3 SELECT L.COUNTRY_ID,C.COUNTRY_NAME FROM LOCATION L, COUNTRIES C WHERE L.COUNTRY_ID = C.COUNTRY_ID;  
4
```

The Results tab shows the output of the query:

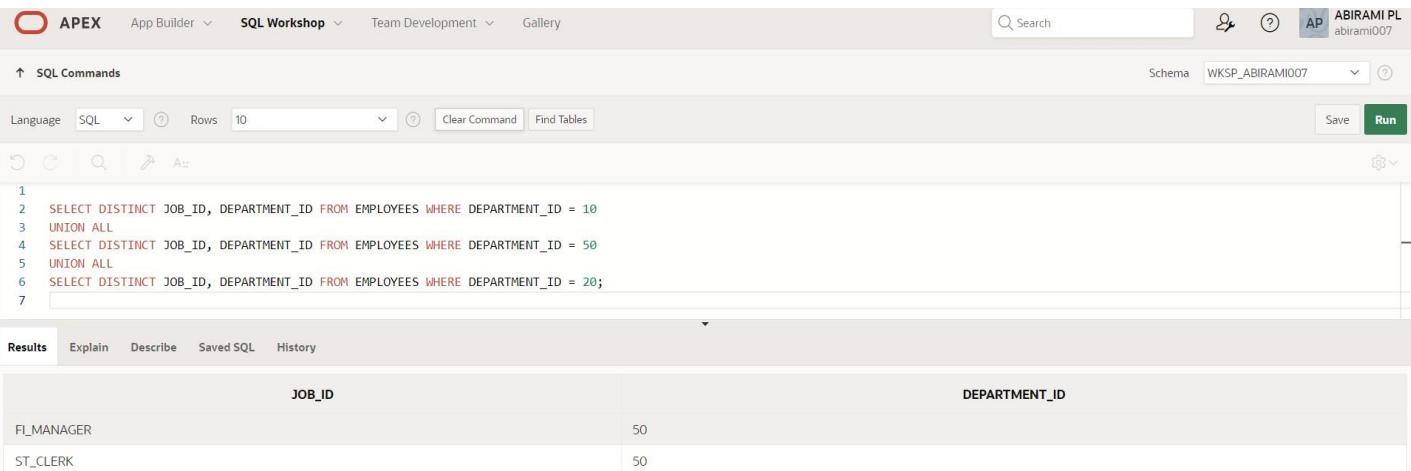
COUNTRY_ID	COUNTRY_NAME
51	ASGARD
79	CANADA

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 user profile for 'ABIRAMI PL' (abirami007). The main area has tabs for SQL Commands, Results (selected), Explain, Describe, Saved SQL, and History. The SQL Commands tab displays the six-line query. The Results tab shows the output in a table:

JOB_ID	DEPARTMENT_ID
FI_MANAGER	50
ST_CLERK	50

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 for ABIRAMI PL (abirami007). The main area is titled 'SQL Commands' with tabs for Language (SQL selected), Rows (10), Clear Command, Find Tables, Save, and Run.

The SQL editor contains the following code:

```
1 SELECT EMPLOYEE_ID, JOB_ID FROM EMPLOYEES INTERSECT  
2 SELECT EMPLOYEE_ID, JOB_ID FROM JOB_HISTORY;  
3
```

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

EMPLOYEE_ID	JOB_ID
176	FI_MANAGER
2	SL REP
4	DESIGNER
172	HR_MANAGER
5	ST_CLERK
3	ST_CLERK

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 top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows a user profile for ABIRAMI PL abirami007. The main area is titled 'SQL Commands' with tabs for Language (set to SQL), Rows (set to 10), Clear Command, and Find Tables. The SQL editor contains the following code:

```
1 SELECT LAST_NAME,DEPARTMENT_ID,TO_CHAR(NULL) FROM EMPLOYEES UNION  
2 SELECT DEPT_NAME,DEPT_ID,TO_CHAR(NULL) FROM DEPT;  
3
```

The results tab is selected, displaying the output of the query. The columns are LAST_NAME, DEPARTMENT_ID, and TO_CHAR(NULL). The data is as follows:

LAST_NAME	DEPARTMENT_ID	TO_CHAR(NULL)
EMANUEL	50	-
JAY	39	-
DAVIES	5	-
JANE	80	-
PARTHI	50	-
UMA	97	-

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. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there's a search bar, a user icon, and a session identifier 'ABIRAMI PL abirami007'. Below the toolbar, the schema is set to 'WKSP_ABIRAMI007'. The main area is titled 'SQL Commands' with options for Language (SQL selected), Rows (10), Clear Command, Find Tables, Save, and Run. The code editor contains the following SQL command:

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

Below the code editor, the 'Results' tab is active, showing the message 'View created.' and a execution time of '0.03 seconds'.

2. Display the contents of the EMPLOYEES_VU view.

QUERY:

SELECT*FROM EMPLOYEES_VU;

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. At the top, there are tabs for APEX, App Builder, SQL Workshop (which is selected), Team Development, and Gallery. On the right, there are buttons for Search, Save, Run, and a schema dropdown set to WKSP_ABIRAMI007. The main area has tabs for SQL Commands, Results (which is selected), Explain, Describe, Saved SQL, and History. In the SQL Commands tab, the query `SELECT*FROM EMPLOYEES_VU;` is entered. The Results tab displays the following data:

EMPLOYEE_ID	EMPLOYEE	DEPARTMENT_ID
176	SAM EMANUEL	50
2	ALEX JAY	39
4	MAX DAVIES	5
172	JENNIE JANE	80
5	MEENA PARTHI	50
3	JENNY UMA	97

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 top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, and Gallery. The right side shows a user profile for ABIRAMI PL (abirami007). The main area is titled "SQL Commands". The command entered is:

```
1
2  SELECT VIEW_NAME,TEXT FROM USER_VIEWS;
3
```

The results tab is selected, displaying the output:

VIEW_NAME	TEXT
EMPLOYEES_VU	SELECT EMPLOYEE_ID,FIRST_NAME '' LAST_NAME,DEPARTMENT_ID FROM EMPLOYEES

4. Using your EMPLOYEES_VU view, enter a query to display all employees names and department.

QUERY:

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

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. At the top, there's a navigation bar with 'APEX' logo, 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. To the right are 'Search' input, a user icon 'abirami007', and a 'Run' button. Below the navigation is a toolbar with icons for 'SQL Commands', 'Language' set to 'SQL', 'Rows' set to '10', and buttons for 'Clear Command' and 'Find Tables'. The main area contains the SQL command entered:

```
1 SELECT EMPLOYEE,DEPARTMENT_ID FROM EMPLOYEES_VU;
2
3
```

Below the command is a results grid. The 'Results' tab is selected. The grid has two columns: 'EMPLOYEE' and 'DEPARTMENT_ID'. The data is as follows:

EMPLOYEE	DEPARTMENT_ID
SAM EMANUEL	50
ALEX JAY	39
MAX DAVIES	5
JENNIE JANE	80
MEENA PARTHI	50
JENNY UMA	97

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:

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 ABIRAMI PL (abirami007). 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 Undo, Redo, Search, and Run. The code editor displays the following SQL command:

```
1 CREATE OR REPLACE VIEW DEPT50 (EMPNO, EMPLOYEE, DEPTNO) AS SELECT EMPLOYEE_ID, LAST_NAME, DEPARTMENT_ID  
2 FROM EMPLOYEES WHERE DEPARTMENT_ID = 50;  
3  
4
```

Below the code editor is a results panel with tabs for Results (selected), Explain, Describe, Saved SQL, and History. The results section displays the message: "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. In the top navigation bar, the 'SQL Workshop' tab is selected. The main area displays the following SQL commands:

```
1 DESCRIBE DEPT50;  
2 SELECT*FROM DEPT50;  
3
```

Below the commands, the 'Describe' tab is selected in the results pane. The results show the structure of the DEPT50 view:

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
DEPT50	EMPNO	NUMBER	-	7	0	-	✓	-	-
	EMPLOYEE	VARCHAR2	20	-	-	-	✓	-	-
	DEPTNO	VARCHAR2	10	-	-	-	✓	-	-

The screenshot shows the Oracle SQL Workshop interface. In the top navigation bar, the 'SQL Workshop' tab is selected. The main area displays the same SQL commands as the previous screenshot:

```
1 DESCRIBE DEPT50;  
2 SELECT*FROM DEPT50;  
3
```

Below the commands, the 'Results' tab is selected in the results pane. The results show the data contained in the DEPT50 view:

EMPNO	EMPLOYEE	DEPTNO
176	EMANUEL	50
5	PARTHI	50

7. Attempt to reassign Matos to department 80.

QUERY:

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

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there's a search bar, a user icon, and the session information "ABIRAMI PL abirami007". The main area is titled "SQL Commands" and contains the following content:

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

```
1 UPDATE DEPT50 SET DEPTNO = 80 WHERE EMPLOYEE = 'MATOS';  
2
```

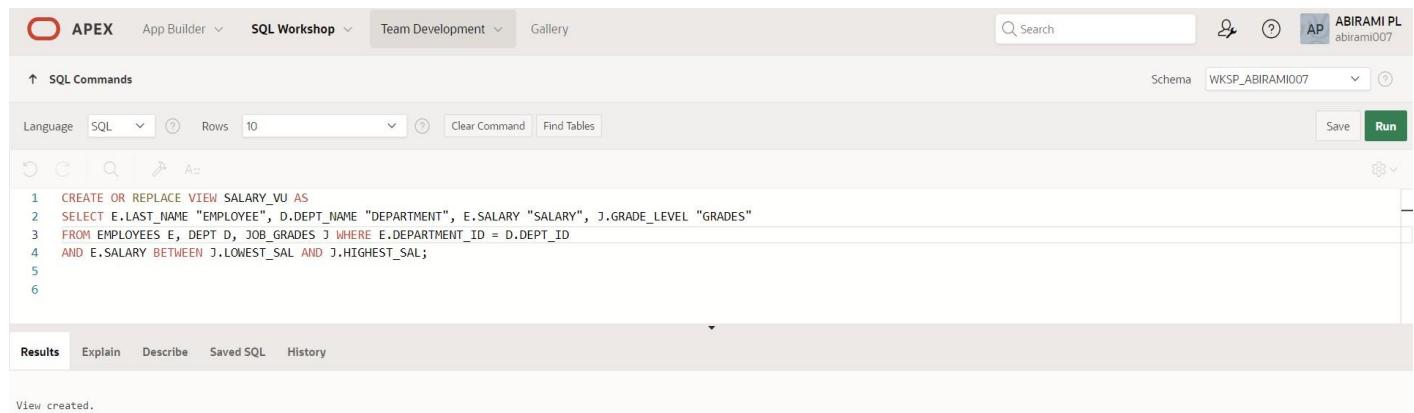
Below the command input area, there are tabs for Results, Explain, Describe, Saved SQL, and History. The Results tab is selected, and the output message "1 row(s) updated." is displayed.

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:



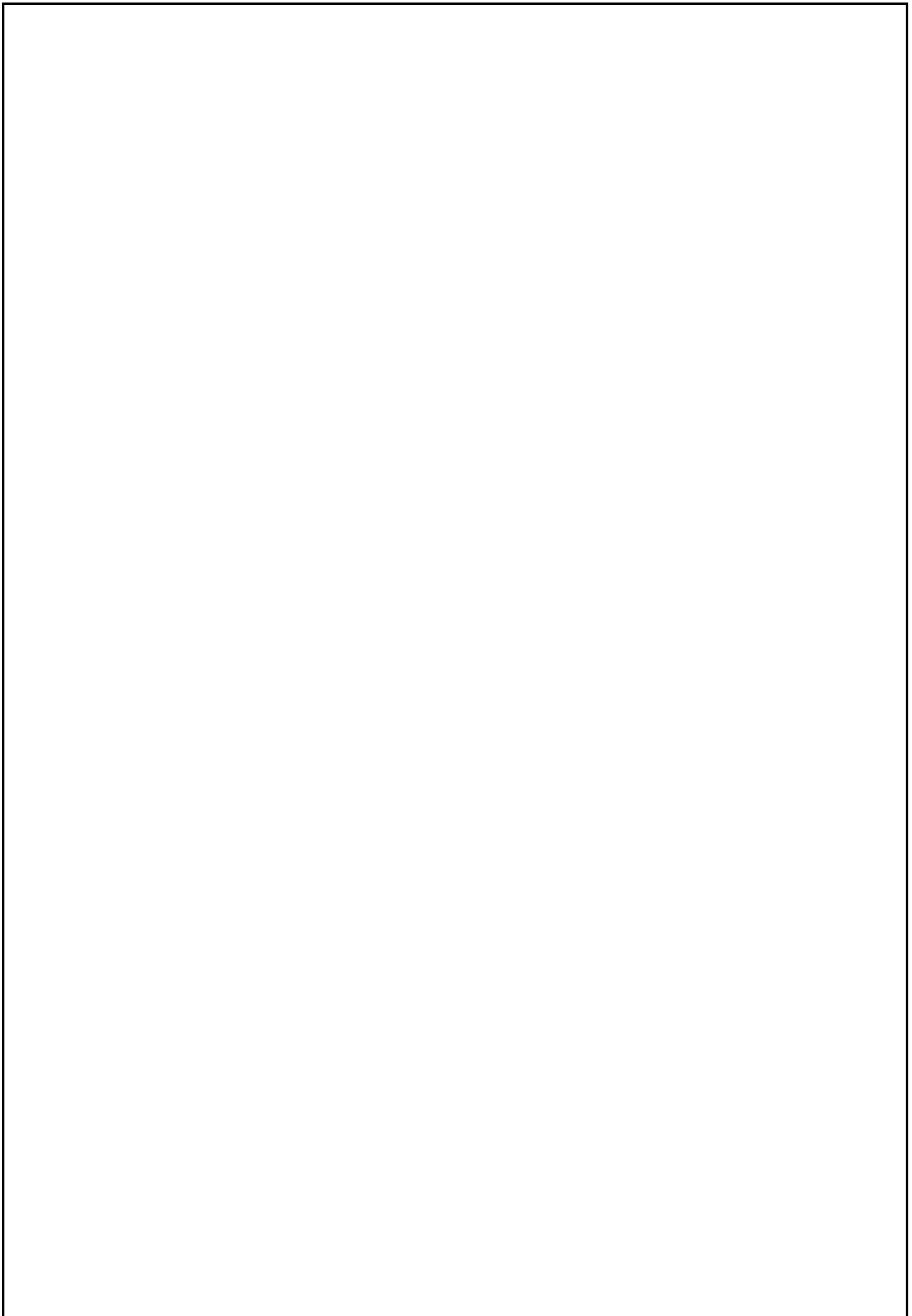
The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there's a search bar, user information (ABIRAMI PL abirami007), and a Run button. The main area is titled 'SQL Commands' with tabs for Language (set to SQL), Rows (set to 10), and various icons. Below this is a code editor containing the SQL query for creating the SALARY_VU view. The code is as follows:

```
1 CREATE OR REPLACE VIEW SALARY_VU AS
2 SELECT E.LAST_NAME "EMPLOYEE", D.DEPARTMENT_NAME "DEPARTMENT",
3 E.SALARY "SALARY", J.GRADE_LEVEL "GRADES"
4 FROM EMPLOYEES E, DEPARTMENT D, JOB_GRADES J WHERE E.DEPARTMENT_ID = D.DEPARTMENT_ID
5 AND E.SALARY BETWEEN J.LOWEST_SAL AND J.HIGHEST_SAL;
6
```

At the bottom, there are tabs for Results, Explain, Describe, Saved SQL, and History. The Results tab shows the message "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

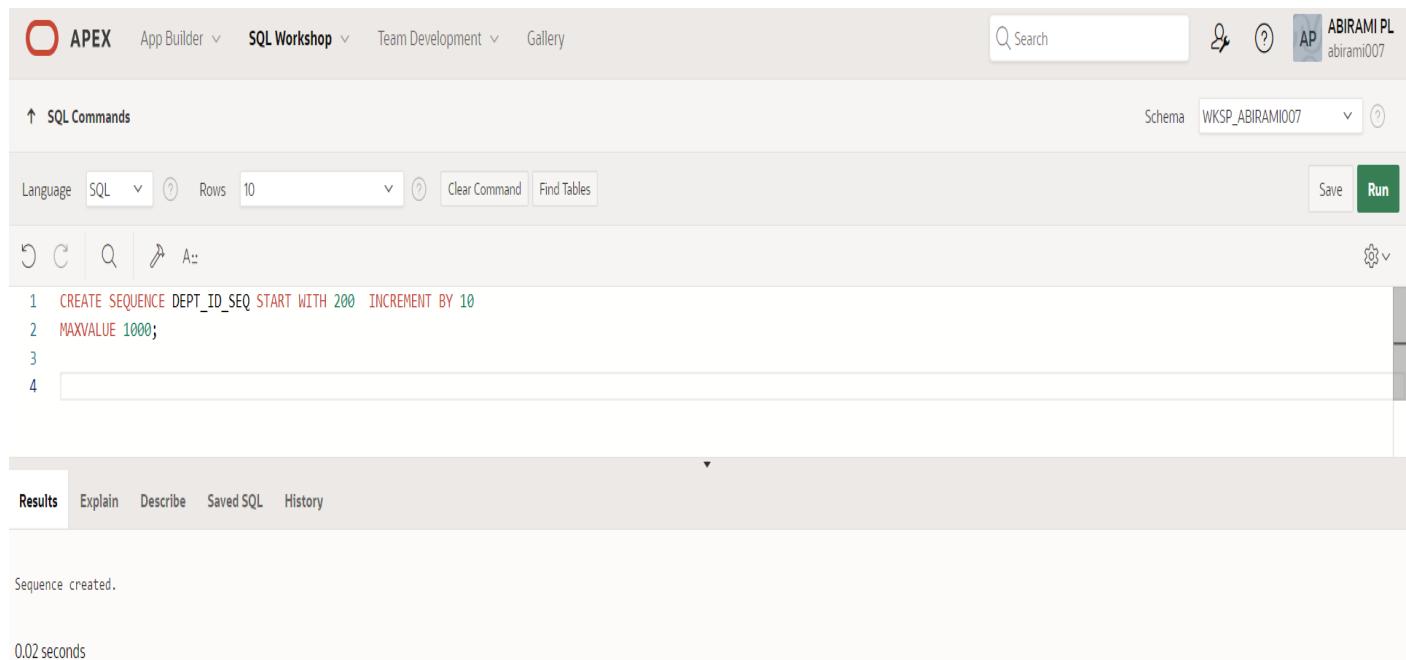
DATE:

-
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. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there's a search bar, a user icon (AP abirami007), and a schema dropdown set to WKSP_ABIRAMI007. The main workspace is titled 'SQL Commands' and contains the following SQL code:

```
1 CREATE SEQUENCE DEPT_ID_SEQ START WITH 200 INCREMENT BY 10  
2 MAXVALUE 1000;  
3  
4
```

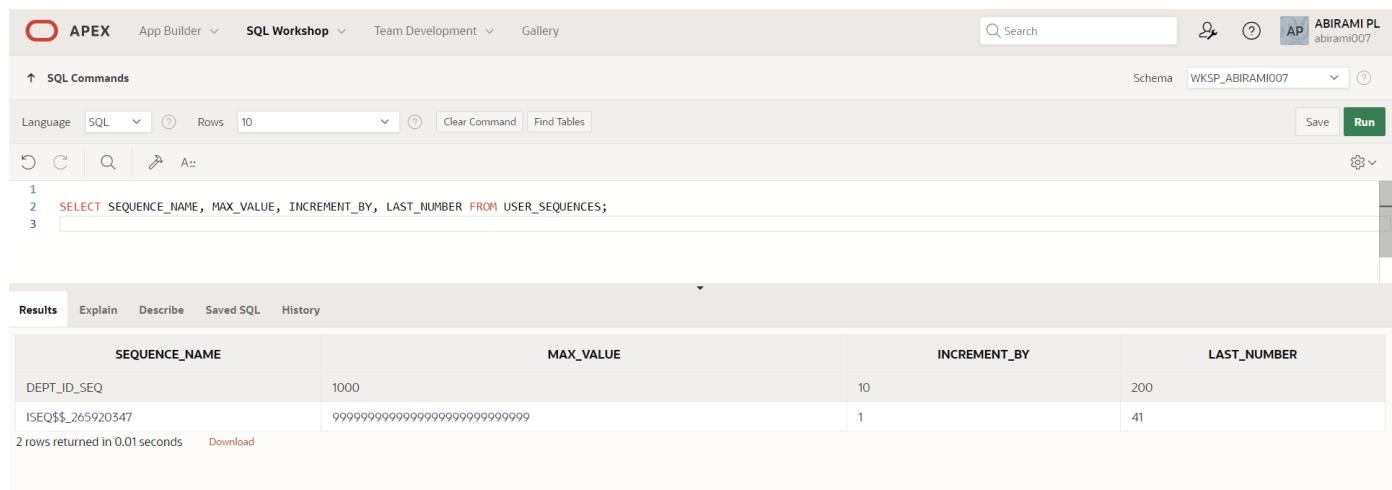
Below the code, the 'Results' tab is selected, showing the message "Sequence created." and a execution time of "0.02 seconds". Other tabs available include Explain, Describe, Saved SQL, and History.

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. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows the user ABIRAMI PL and session ID abirami007. The main area is titled "SQL Commands" with tabs for Language (SQL selected), Rows (10), Clear Command, Find Tables, Save, and Run. The SQL editor contains the following code:

```
1 SELECT SEQUENCE_NAME, MAX_VALUE, INCREMENT_BY, LAST_NUMBER FROM USER_SEQUENCES;  
2  
3
```

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

SEQUENCE_NAME	MAX_VALUE	INCREMENT_BY	LAST_NUMBER
DEPT_ID_SEQ	1000	10	200
ISEQ\$\$_265920347	99999999999999999999999999999999	1	41

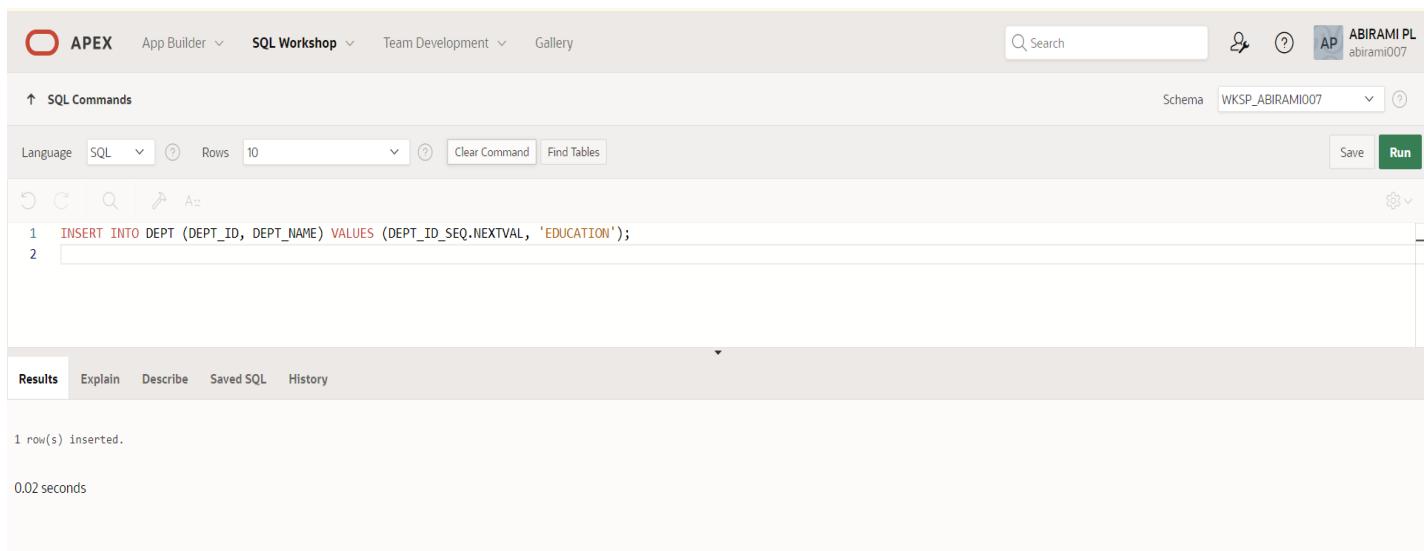
Below the table, it says "2 rows returned in 0.01 seconds" and has a "Download" link.

3. Write a script to insert two rows into the DEPT table. Name your script lab12_3.sql. Be sure to use the sequence that you created for the ID column. Add two departments named Education and Administration. Confirm your additions. Run the commands in your script.

QUERY:

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

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's a search bar, a user icon for 'ABIRAMI PL abirami007', and a schema dropdown set to 'WKSP_ABIRAMI007'. The main workspace is titled 'SQL Commands' and contains the following SQL code:

```
1 INSERT INTO DEPT (DEPT_ID, DEPT_NAME) VALUES (DEPT_ID_SEQ.NEXTVAL, 'EDUCATION');
2 
```

Below the code, the results section shows the output:

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

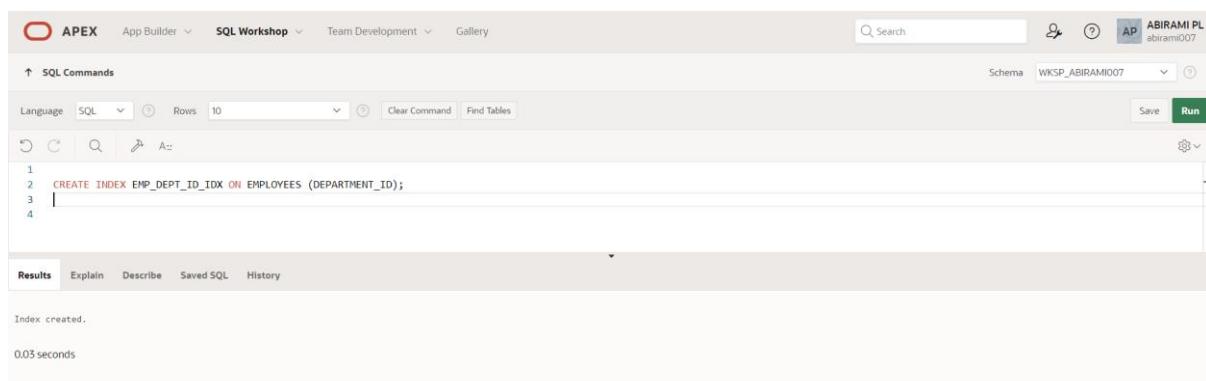
Execution time: 0.02 seconds

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

QUERY:

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

OUTPUT:



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

```
1 CREATE INDEX EMP_DEPT_ID_IDX ON EMPLOYEES (DEPARTMENT_ID);
```

In the results section, the output is:

```
Index created.
```

Execution time is listed as 0.03 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 SQL command entered is:

```
1  
2 SELECT INDEX_NAME, TABLE_NAME,UNIQUENESS FROM USER_INDEXES WHERE TABLE_NAME='EMPLOYEES';  
3  
4
```

The results section displays the following table:

INDEX_NAME	TABLE_NAME	UNIQUENESS
EMP_DEPT_ID_IDX	EMPLOYEES	NONUNIQUE

1 rows returned in 0.08 seconds Download

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

RESULT :

CONTROLLING USER ACCESS

EX_NO:15

DATE:

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

The CREATE SESSION system privilege

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

The CREATE TABLE privilege

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

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

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

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

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

The ALTER USER statement

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

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

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

7. Query all the rows in your DEPARTMENTS table.

SELECT * FROM departments;

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

Team 1 executes this INSERT statement. INSERT INTO departments(department_id, department_name) VALUES (500, 'Education'); COMMIT;

Team 2 executes this INSERT statement. `INSERT INTO departments(department_id, department_name) VALUES (510, 'Administration'); COMMIT;`

9. Query the `USER_TABLES` data dictionary to see information about the tables that you own.
`SELECT table_name FROM user_tables;`

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

Team 1 revokes the privilege.

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

Team 2 revokes the privilege.

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

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

Team 1 executes this `INSERT` statement.

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

Team 2 executes this `INSERT` statement.

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

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

RESULT:

PL/SQL CONTROL STRUCTURES

EX-NO : 16

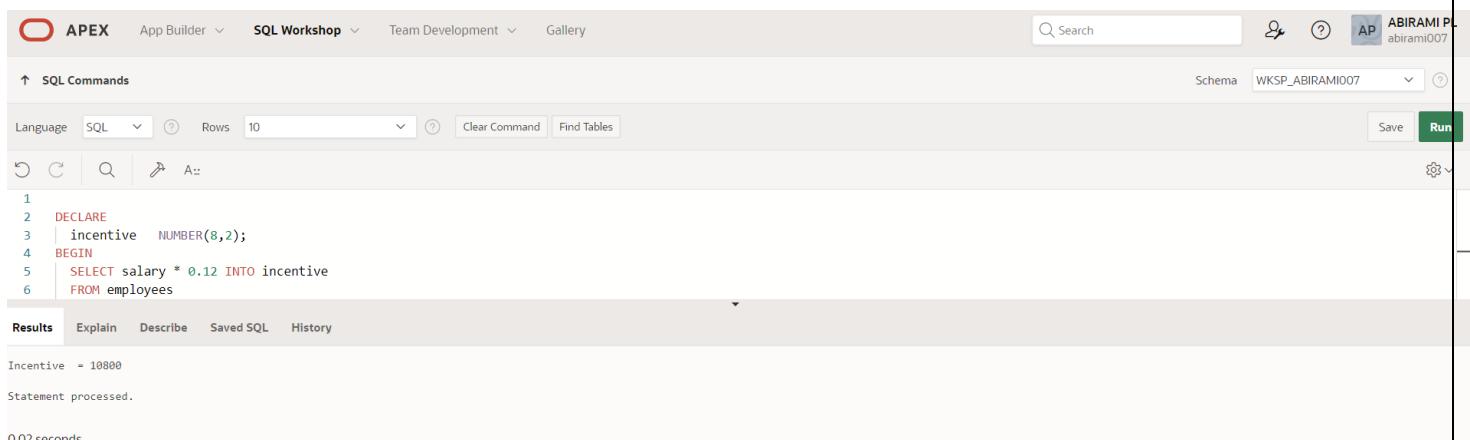
DATE:

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

QUERY:

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

OUTPUT:



The screenshot shows the Oracle 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 ABIRAMI P abirami007. The main area is titled 'SQL Commands' with tabs for Language (set to SQL), Rows (set to 10), and various command buttons like Clear Command and Find Tables. The code editor contains the following PL/SQL block:

```
1  DECLARE
2      incentive NUMBER(8,2);
3  BEGIN
4      SELECT salary * 0.12 INTO incentive
5      FROM employees
6  END;
```

The results tab at the bottom shows the output of the query:

```
Incentive = 10800
Statement processed.

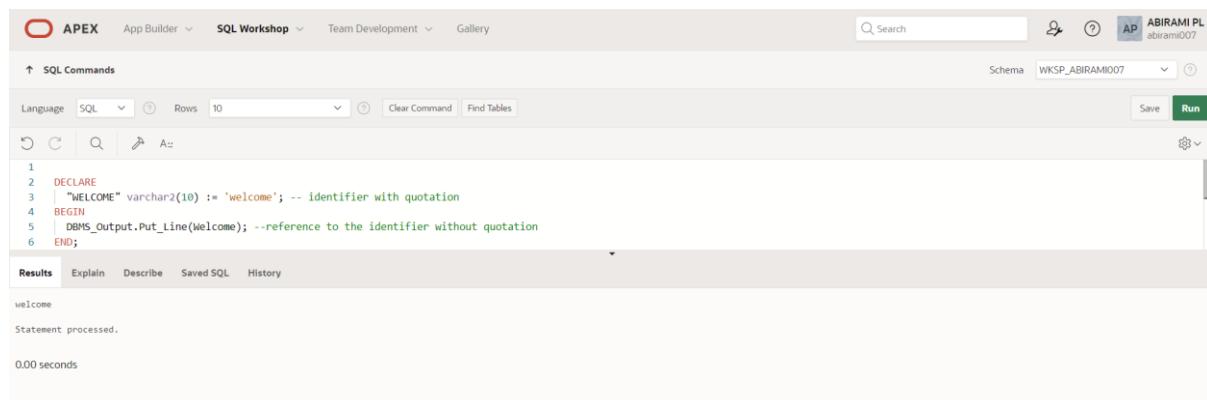
0.02 seconds
```

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', and 'Gallery'. On the right, there's a user icon, a search bar, and a schema dropdown set to 'WKSP_ABIRAMI007'. The main area is titled 'SQL Commands' with a 'Run' button. Below it, a code editor window displays the following PL/SQL block:

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

Below the code editor, the 'Results' tab is selected. The output pane shows the results of the execution:

```
Welcome
Statement processed.
0.00 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:

```
1 DECLARE
2     v_employee_id NUMBER := 122;
3     v_new_salary NUMBER;
4 BEGIN
5     SELECT salary INTO v_new_salary
6     FROM employees
```

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

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

QUERY:

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

```

APEX App Builder SQL Workshop Team Development Gallery
SQL Commands Schema: WKSP_ABIRAMI007
Language: SQL Rows: 10 Save Run
1 CREATE OR REPLACE PROCEDURE pri_bool(
2   boo_name VARCHAR2,
3   boo_val BOOLEAN
4 ) IS
5 BEGIN
6   IF boo_val IS NULL THEN
----- FOR m TRUE -----
m = TRUE
NOT m = FALSE
----- FOR m FALSE -----
m = FALSE
NOT m = TRUE
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:

```

APEX App Builder SQL Workshop Team Development Gallery
SQL Commands Schema: WKSP_ABIRAMI007
Language: SQL Rows: 10 Save Run
1
2 DECLARE
3   PROCEDURE pat_match (
4     test_string  VARCHAR2,
5     pattern      VARCHAR2
6   ) IS
----- TRUE -----
TRUE
----- FALSE -----
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.

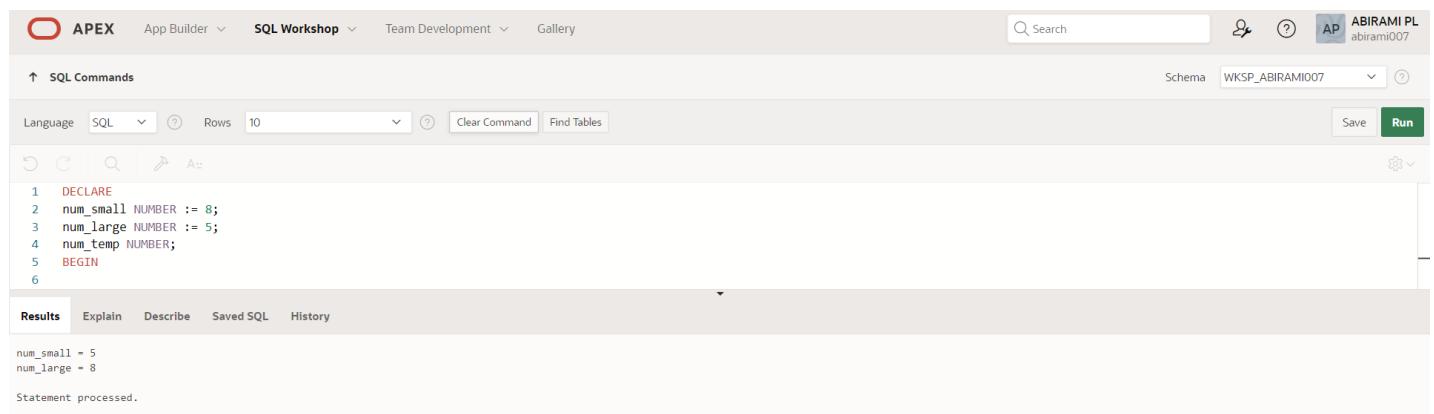
QUERY:

```
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 ABIRAMI PL abirami007. 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 Undo, Redo, Search, and Run. The code editor contains a PL/SQL block with numbered lines 1 through 6. The results tab at the bottom shows the output of the executed code: 'num_small = 5' and 'num_large = 8'. A message 'Statement processed.' is also visible.

```
1  DECLARE
2  num_small NUMBER := 8;
3  num_large NUMBER := 5;
4  num_temp NUMBER;
5  BEGIN
6

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

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

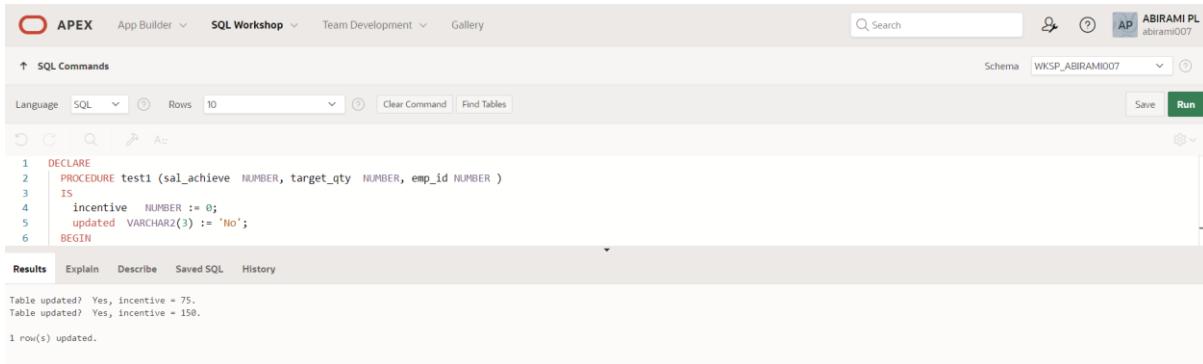
QUERY:

```
DECLARE
  PROCEDURE test1 (sal_achieve NUMBER, target_qty NUMBER, emp_id
NUMBER )
  IS
    incentive NUMBER := 0;
    updated VARCHAR2(3) := 'No';
  BEGIN
    IF sal_achieve > (target_qty + 200) THEN
      incentive := (sal_achieve - target_qty)/4;

      UPDATE employees
      SET salary = salary + incentive
      WHERE employee_id = emp_id;

      updated := 'Yes';
    END IF;
    DBMS_OUTPUT.PUT_LINE (
      'Table updated? ' || updated || ',' ||
      'incentive = ' || incentive || '
    );
  END test1;
BEGIN
  test1(2300, 2000, 144);
  test1(3600, 3000, 145);
END;
```

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, and Gallery. The SQL Workshop tab is selected. The schema dropdown shows WKS_ABIRAMI007. 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 DBMS_OUTPUT.PUT_LINE statements and the confirmation of one row being updated.

```
1  DECLARE
2  2    PROCEDURE test1 (sal_achieve NUMBER, target_qty NUMBER, emp_id NUMBER )
3  3    IS
4  4      incentive NUMBER := 0;
5  5      updated VARCHAR2(3) := 'No';
6  6    BEGIN
7
8      UPDATE employees
9      SET salary = salary + incentive
10     WHERE employee_id = emp_id;
11
12      updated := 'Yes';
13    END IF;
14    DBMS_OUTPUT.PUT_LINE (
15      'Table updated? ' || updated || ',' ||
16      'incentive = ' || incentive || '
17    );
18  END test1;
19
20 BEGIN
21   test1(2300, 2000, 144);
22   test1(3600, 3000, 145);
23 END;
```

Results

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

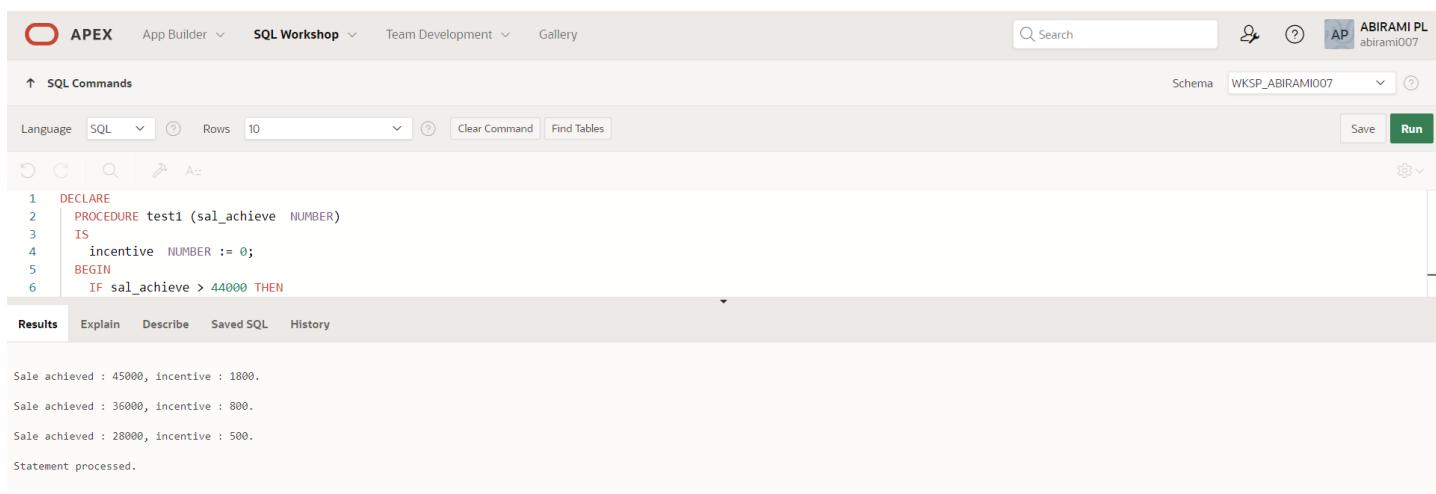
1 row(s) updated.
```

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

QUERY:

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

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows the user's profile: ABIRAMI PL abirami007. The main area has tabs for SQL Commands, Results, Explain, Describe, Saved SQL, and History. The SQL Commands tab is active, displaying the PL/SQL code from the previous block. The Results tab shows the output of the procedure execution:

```
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.');
```

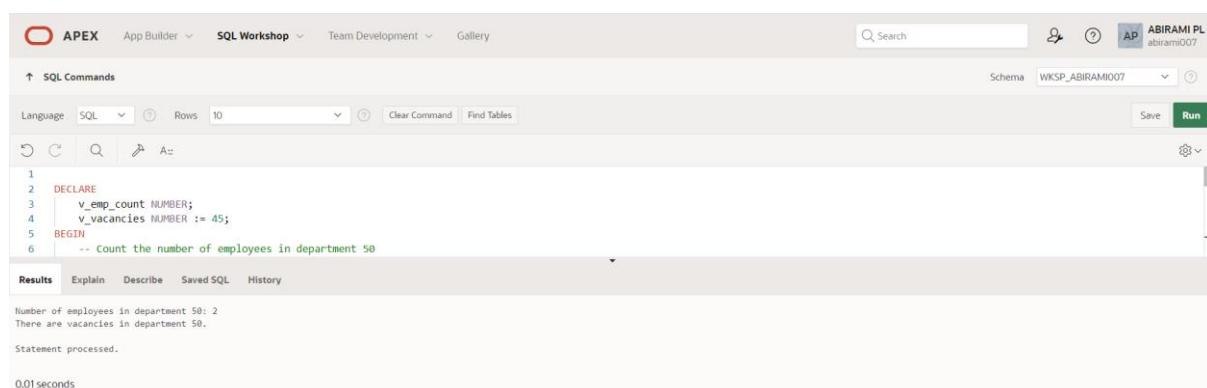
```
ELSE
```

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

```
END IF;
```

END;

OUTPUT:



```
1  DECLARE
2      v_emp_count NUMBER;
3      v_vacancies NUMBER := 45;
4
5  BEGIN
6      -- Count the number of employees in department 50
```

Number of employees in department 50: 2
There are vacancies in department 50.
Statement processed.
0.01 seconds

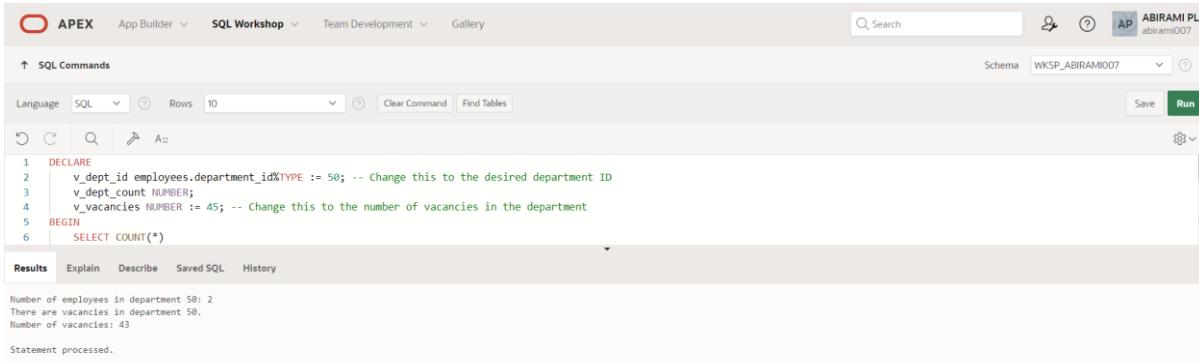
10. Write a PL/SQL program to count number of employees in a specific department and check whether this department have any vacancies or not. If any vacancies, how many vacancies are in that department.

QUERY:

DECLARE

```
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:



```
1 DECLARE
2     v_dept_id employees.department_id%TYPE := 50; -- Change this to the desired department ID
3     v_dept_count NUMBER;
4     v_vacancies NUMBER := 45; -- Change this to the number of vacancies in the department
5 BEGIN
6     SELECT COUNT(*)
```

Number of employees in department 50: 2
There are no vacancies in department 50.
Number of vacancies: 43

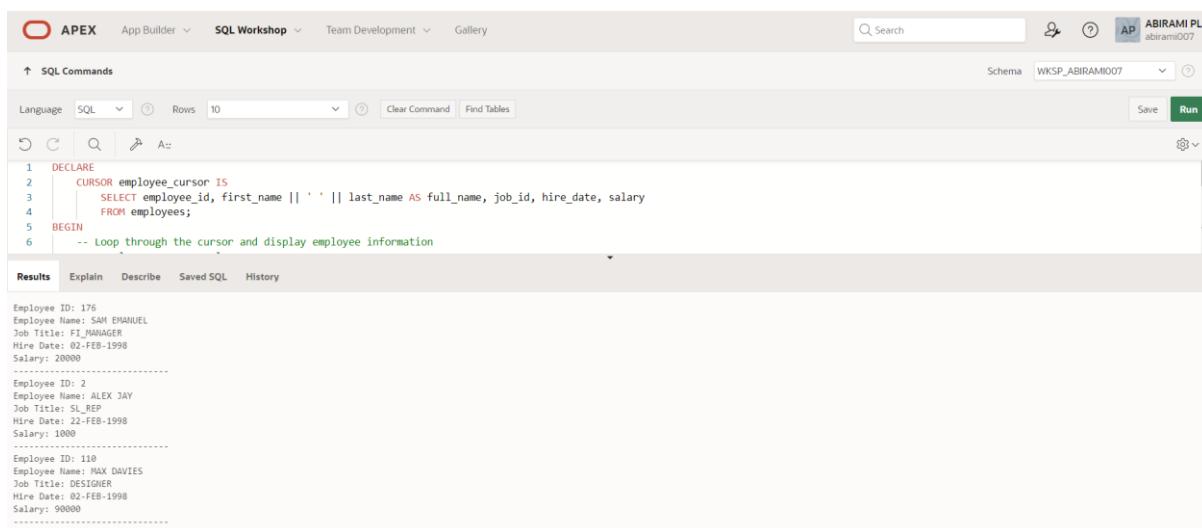
Statement processed.

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. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows the schema as WKSP_ABIRAMI007. The main area has tabs for SQL Commands, Results (selected), Explain, Describe, Saved SQL, and History. The SQL Commands tab contains the PL/SQL code provided above. The Results tab displays the output for each employee in the loop:

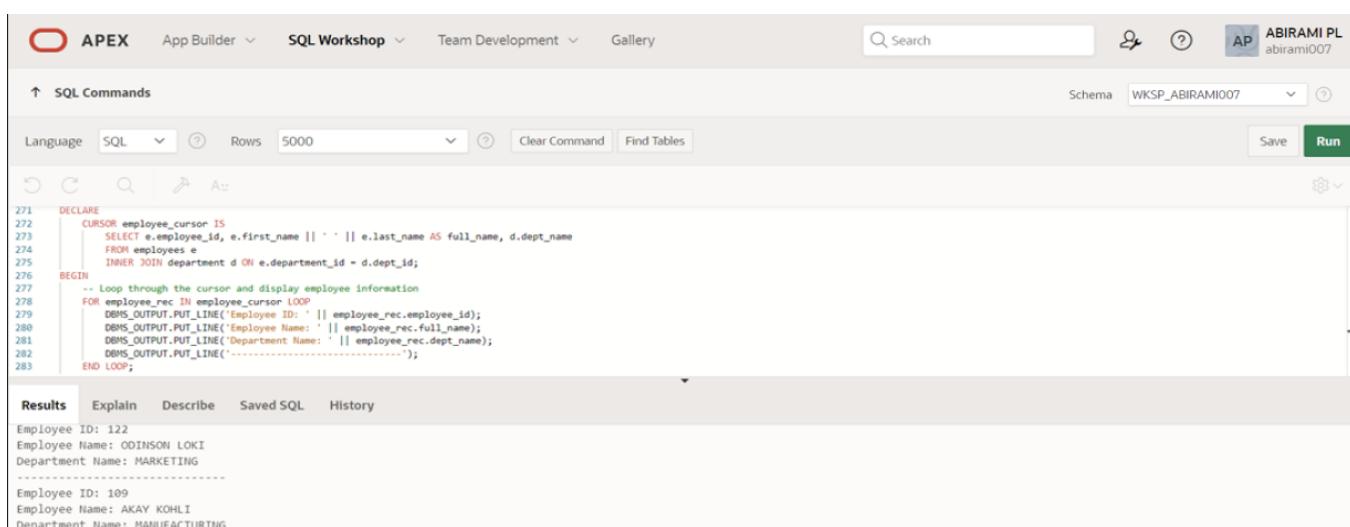
```
Employee ID: 176
Employee Name: SAM ERIC
Job Title: IT_PROG
Hire Date: 01-JAN-1997
Salary: 3000
-----
Employee ID: 102
Employee Name: ALLEN
Job Title: ST_CLERK
Hire Date: 20-JUN-1997
Salary: 1600
-----
Employee ID: 110
Employee Name: ADAM
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 111
Employee Name: TAYLOR
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 112
Employee Name: MARK
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 113
Employee Name: PATRICK
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 114
Employee Name: HUNTER
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 115
Employee Name: MARGARET
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 116
Employee Name: LYNNE
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 117
Employee Name: DUSTY
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 118
Employee Name: STEPHEN
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 119
Employee Name: KATHY
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 120
Employee Name: NANCY
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 121
Employee Name: SHERILYN
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 122
Employee Name: GENE
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 123
Employee Name: BRIAN
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 124
Employee Name: ANTHONY
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 125
Employee Name: GENE
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 126
Employee Name: ANTHONY
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 127
Employee Name: GENE
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 128
Employee Name: ANTHONY
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 129
Employee Name: GENE
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 130
Employee Name: ANTHONY
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 131
Employee Name: GENE
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 132
Employee Name: ANTHONY
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 133
Employee Name: GENE
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 134
Employee Name: ANTHONY
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 135
Employee Name: GENE
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 136
Employee Name: ANTHONY
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 137
Employee Name: GENE
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 138
Employee Name: ANTHONY
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 139
Employee Name: GENE
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 140
Employee Name: ANTHONY
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 141
Employee Name: GENE
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 142
Employee Name: ANTHONY
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 143
Employee Name: GENE
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 144
Employee Name: ANTHONY
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 145
Employee Name: GENE
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 146
Employee Name: ANTHONY
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 147
Employee Name: GENE
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 148
Employee Name: ANTHONY
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 149
Employee Name: GENE
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 150
Employee Name: ANTHONY
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 151
Employee Name: GENE
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 152
Employee Name: ANTHONY
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 153
Employee Name: GENE
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 154
Employee Name: ANTHONY
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 155
Employee Name: GENE
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 156
Employee Name: ANTHONY
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 157
Employee Name: GENE
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 158
Employee Name: ANTHONY
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 159
Employee Name: GENE
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 160
Employee Name: ANTHONY
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 161
Employee Name: GENE
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 162
Employee Name: ANTHONY
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 163
Employee Name: GENE
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 164
Employee Name: ANTHONY
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 165
Employee Name: GENE
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 166
Employee Name: ANTHONY
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 167
Employee Name: GENE
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 168
Employee Name: ANTHONY
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 169
Employee Name: GENE
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 170
Employee Name: ANTHONY
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 171
Employee Name: GENE
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 172
Employee Name: ANTHONY
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 173
Employee Name: GENE
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 174
Employee Name: ANTHONY
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
-----
Employee ID: 175
Employee Name: GENE
Job Title: SALESREP
Hire Date: 01-JUL-1997
Salary: 1200
```

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:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows the user ABIRAMI PL and schema WKSP_ABIRAMI007. The main area is titled 'SQL Commands' with a 'Run' button. The code area contains a PL/SQL block with lines numbered 271 to 283. The results section displays the output of the program, which lists two employees: ODINSON LOKI and AKAY KOHLI, along with their respective department names.

```
271  DECLARE
272    CURSOR employee_cursor IS
273      SELECT e.employee_id, e.first_name || ' ' || e.last_name AS full_name, d.dept_name
274        FROM employees e
275       INNER JOIN department d ON e.department_id = d.dept_id;
276
277  BEGIN
278    -- Loop through the cursor and display employee information
279    FOR employee_rec IN employee_cursor LOOP
280      DBMS_OUTPUT.PUT_LINE('Employee ID: ' || employee_rec.employee_id);
281      DBMS_OUTPUT.PUT_LINE('Employee Name: ' || employee_rec.full_name);
282      DBMS_OUTPUT.PUT_LINE('Department Name: ' || employee_rec.dept_name);
283      DBMS_OUTPUT.PUT_LINE('-----');
284    END LOOP;

```

Results
Employee ID: 122 Employee Name: ODINSON LOKI Department Name: MARKETING ----- Employee ID: 109 Employee Name: AKAY KOHLI Department Name: MANUFACTURING

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:

The screenshot shows the APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. The SQL Workshop tab is active. On the right, there's a search bar, user profile icons, and a schema dropdown set to WKSP_ABIRAMI007. The main workspace displays an SQL command to find the minimum salary for each job ID:

```
1 DECLARE
2   CURSOR job_cursor IS
3     SELECT job_id, MIN(salary) AS min_salary
4       FROM employees
5      GROUP BY job_id;
6 BEGIN
```

The results pane shows the output of the query:

Job ID	Minimum Salary
FI_MANAGER	20000
ST_CLERK	50000
SL REP	1000
DESIGNER	90000
HR_MANAGER	9000

At the bottom, a message indicates "Statement processed."

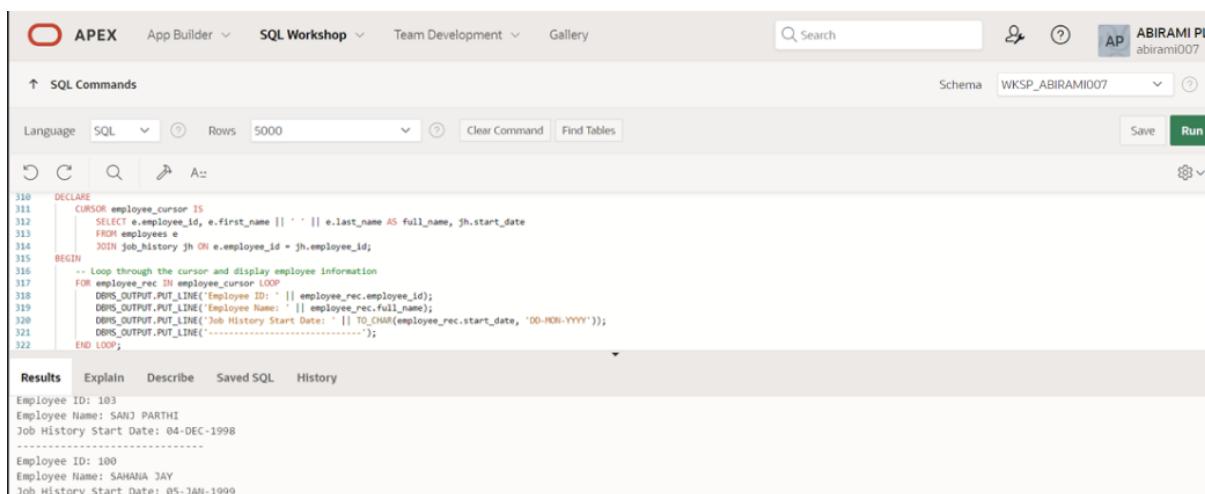
14. Write a PL/SQL program to display the employee IDs, names, and job history start dates of all employees.

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. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, and Gallery. The right side shows the user ABIRAMI PL and schema WKSP_ABIRAMI007. The main area is titled 'SQL Commands' and contains the following PL/SQL code:

```
310  DECLARE
311      CURSOR employee_cursor IS
312          SELECT e.employee_id, e.first_name || ' ' || e.last_name AS full_name, jh.start_date
313          FROM employees e
314          JOIN job_history jh ON e.employee_id = jh.employee_id;
315
316  BEGIN
317      -- Loop through the cursor and display employee information
318      FOR employee_rec IN employee_cursor LOOP
319          DBMS_OUTPUT.PUT_LINE('Employee ID: ' || employee_rec.employee_id);
320          DBMS_OUTPUT.PUT_LINE('Employee Name: ' || employee_rec.full_name);
321          DBMS_OUTPUT.PUT_LINE('Job History Start Date: ' || TO_CHAR(employee_rec.start_date, 'DD-MON-YYYY'));
322          DBMS_OUTPUT.PUT_LINE('-----');
323      END LOOP;
324  END;
```

The results pane at the bottom displays the output of the PL/SQL block:

Results	Explain	Describe	Saved SQL	History
Employee ID: 103 Employee Name: SANJ PARTHI Job History Start Date: 04-DEC-1998 ----- Employee ID: 100 Employee Name: SAHANA JAY Job History Start Date: 05-JAN-1999				

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:

APEX App Builder SQL Workshop Team Development Gallery Search AP ABIRAMI PL abirami007

SQL Commands Schema WKSP_ABIRAMI007 Save Run

Language SQL Rows 5000 Clear Command Find Tables

```

330 DECLARE
331   CURSOR employee_cursor IS
332     SELECT e.employee_id, e.first_name || ' ' || e.last_name AS full_name, jh.end_date
333     FROM employees e
334     JOIN job_history jh ON e.employee_id = jh.employee_id;
335 BEGIN
336   -- Loop through the cursor and display employee information
337   FOR employee_rec IN employee_cursor LOOP
338     DBMS_OUTPUT.PUT_LINE('Employee ID: ' || employee_rec.employee_id);
339     DBMS_OUTPUT.PUT_LINE('Employee Name: ' || employee_rec.full_name);
340   END LOOP;
341 END;

```

Results Explain Describe Saved SQL History

```

Employee ID: 103
Employee Name: SANJ PARTHI
Job History End Date: 30-DEC-1999
-----
Employee ID: 100
Employee Name: SAHANA JAY
Job History End Date: 01-MAY-2000
-----
```

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

RESULT :

PROCEDURES AND FUNCTIONS

EX_NO: 17

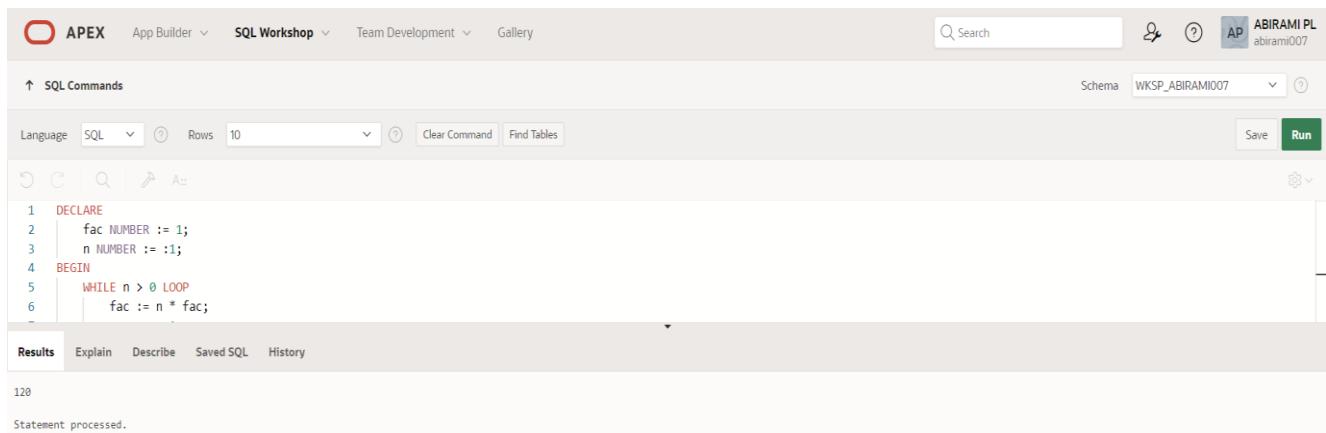
DATE:

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'. On the right, there's a search bar, a user icon for 'ABIRAMI PL abirami007', and a 'Run' button. The main workspace is titled 'SQL Commands'. It shows the following PL/SQL code:

```
1 DECLARE
2     fac NUMBER := 1;
3     n NUMBER := :1;
4 BEGIN
5     WHILE n > 0 LOOP
6         fac := n * fac;

```

Below the code, the 'Results' tab is selected, showing the output: 'Statement processed.' and '120'.

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);

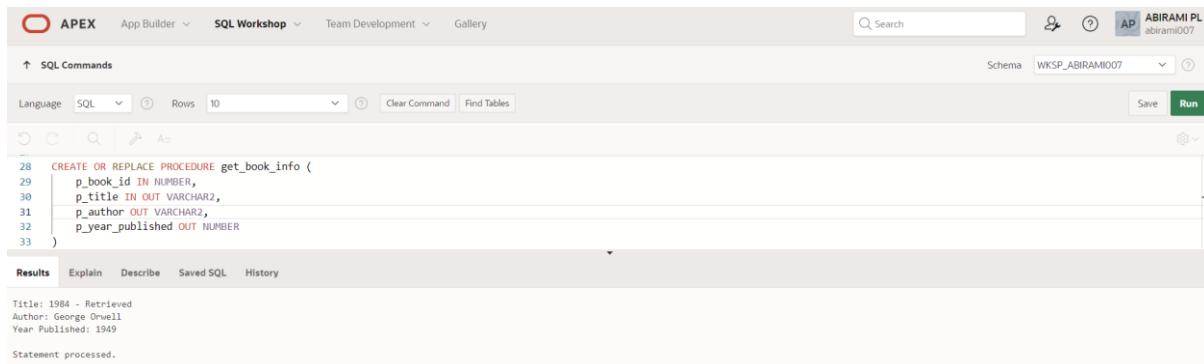
    DBMS_OUTPUT.PUT_LINE('Title: ' || v_title);
```

```

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. In the SQL Commands tab, a PL/SQL procedure named 'get_book_info' is defined. The procedure takes a parameter 'p_book_id' (IN NUMBER) and returns four variables: 'p_title' (IN OUT VARCHAR2), 'p_author' (OUT VARCHAR2), 'p_year_published' (OUT NUMBER), and 'v_author' (OUT NUMBER). After executing the procedure with a specific book ID, the results are displayed in the Results tab, showing the title, author, year published, and retrieved status.

```

28 CREATE OR REPLACE PROCEDURE get_book_info (
29   p_book_id IN NUMBER,
30   p_title IN OUT VARCHAR2,
31   p_author OUT VARCHAR2,
32   p_year_published OUT NUMBER
33 )

```

Evaluation Procedure	Marks awarded
Query(5)	
Execution (5)	

Evaluation Procedure	Marks awarded
Query(5)	
Execution (5)	

Viva(5)	
Total (15)	
Faculty Signature	

RESULT:

TRIGGER

EX_NO: 18

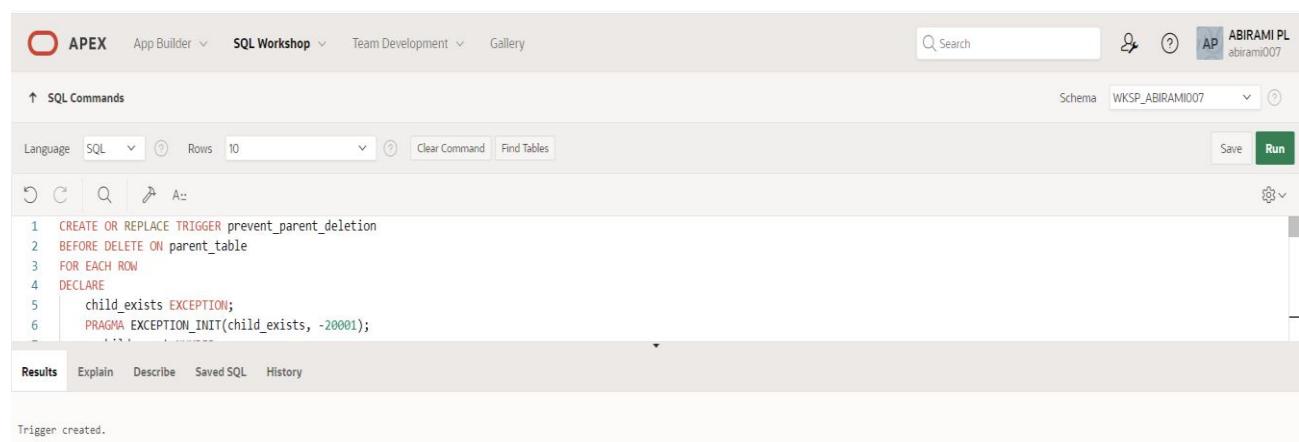
DATE:

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

QUERY:

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

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows the schema WKS_ABIRAMI007 and user ABIRAMI PL. The main area is titled 'SQL Commands' with a 'Results' tab selected. The SQL command entered is:

```
1 CREATE OR REPLACE TRIGGER prevent_parent_deletion
2 BEFORE DELETE ON parent_table
3 FOR EACH ROW
4 DECLARE
5     child_exists EXCEPTION;
6     PRAGMA EXCEPTION_INIT(child_exists, -20001);
```

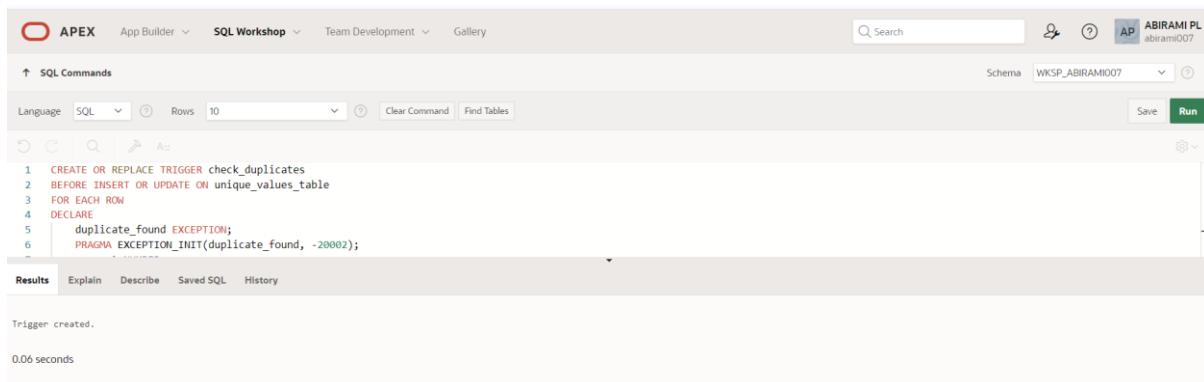
The status bar at the bottom indicates 'Trigger created.'

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. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there are tabs for Schema (WKSP_ABIRAMI007), Save, and Run. The main area displays the SQL command for creating the trigger:

```
1 CREATE OR REPLACE TRIGGER check_duplicates
2 BEFORE INSERT OR UPDATE ON unique_values_table
3 FOR EACH ROW
4 DECLARE
5     duplicate_found EXCEPTION;
6     PRAGMA EXCEPTION_INIT(duplicate_found, -20002);
```

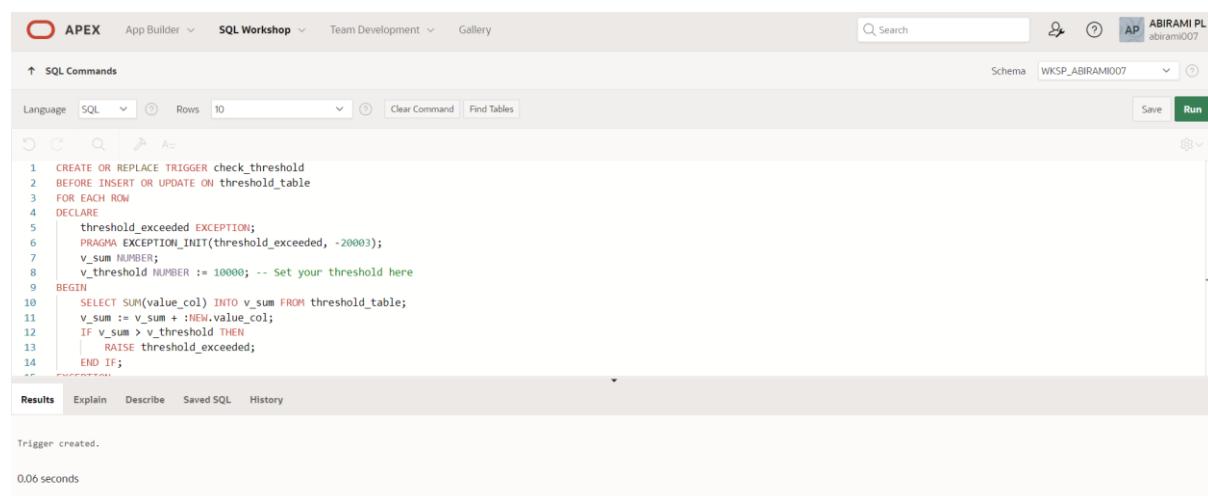
Below the command, the Results tab is selected, showing the output: "Trigger created." and "0.06 seconds".

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

QUERY:

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

OUTPUT:



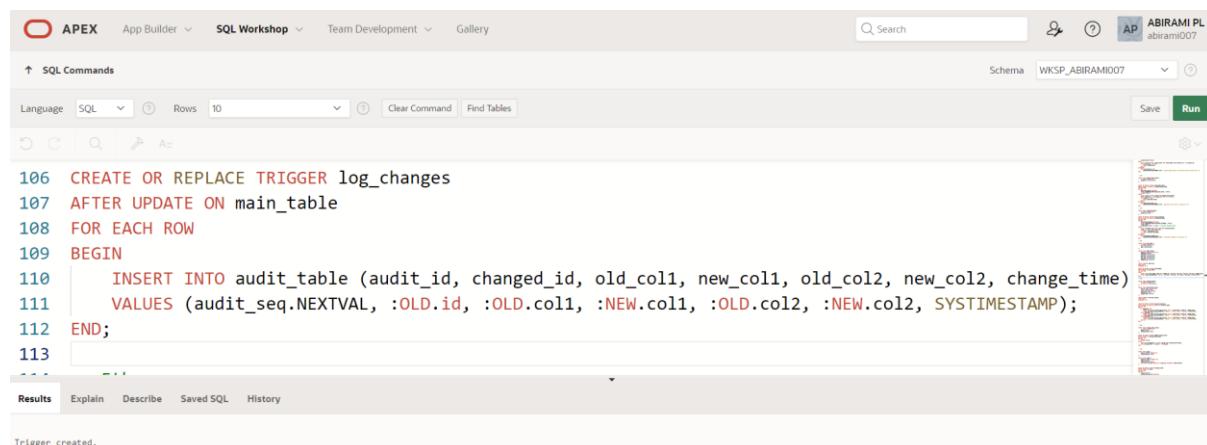
The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. The right side of the header shows the user 'ABIRAMI PL' and the schema 'WKSP_ABIRAMI007'. Below the header is a toolbar with 'Search', 'Save', 'Run', and other icons. The main workspace is titled 'SQL Commands'. It contains the PL/SQL code for the trigger 'check_threshold'. The code defines a trigger that fires before inserting or updating rows in the 'threshold_table'. It declares an exception 'threshold_exceeded', initializes it with code '-20003', and sets a threshold of 10000 for the column 'value_col'. If the sum of the current row's value and the existing total exceeds the threshold, it raises the exception. An exception block handles this raise, raising an application error with the message 'Threshold exceeded for value_col.'. The status bar at the bottom indicates 'Trigger created.' and '0.06 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 the schema WKS_ABIRAMI007 and user ABIRAMI PL. The main area is titled 'SQL Commands' and contains the PL/SQL code for the trigger. The code is numbered from 106 to 113. The 'Run' button is visible at the bottom right of the code editor. Below the code, the 'Results' tab is selected, displaying the message 'Trigger created.'

```
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
```

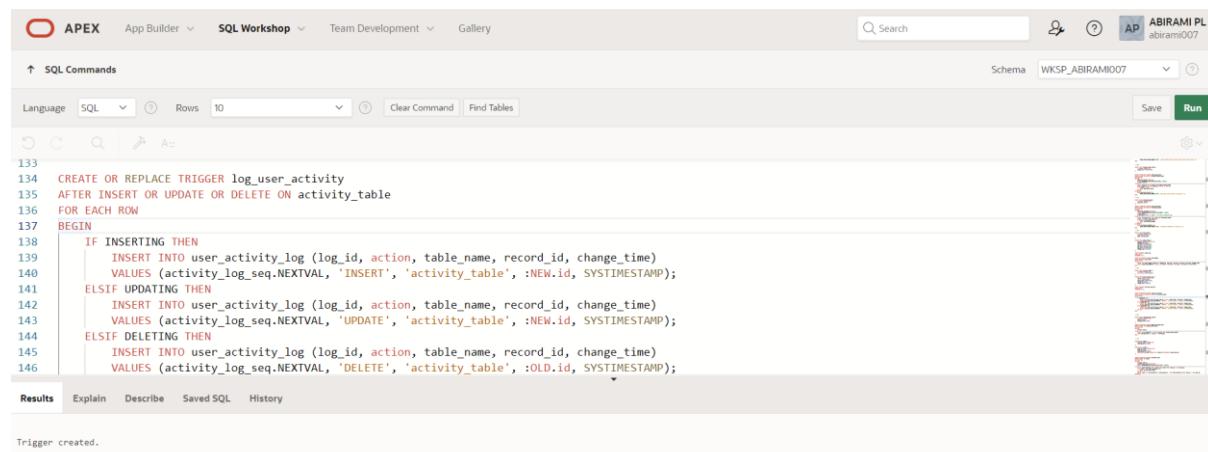
Trigger created.

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' (selected), 'Team Development', and 'Gallery'. The right side of the header shows the schema 'WKSP_ABIRAMI007' and a user icon. Below the header is a toolbar with 'Search', 'Save', and 'Run' buttons. The main area is a SQL editor window. The code listed is the PL/SQL trigger definition provided in the previous section. At the bottom of the editor, the status bar displays 'Trigger created.'

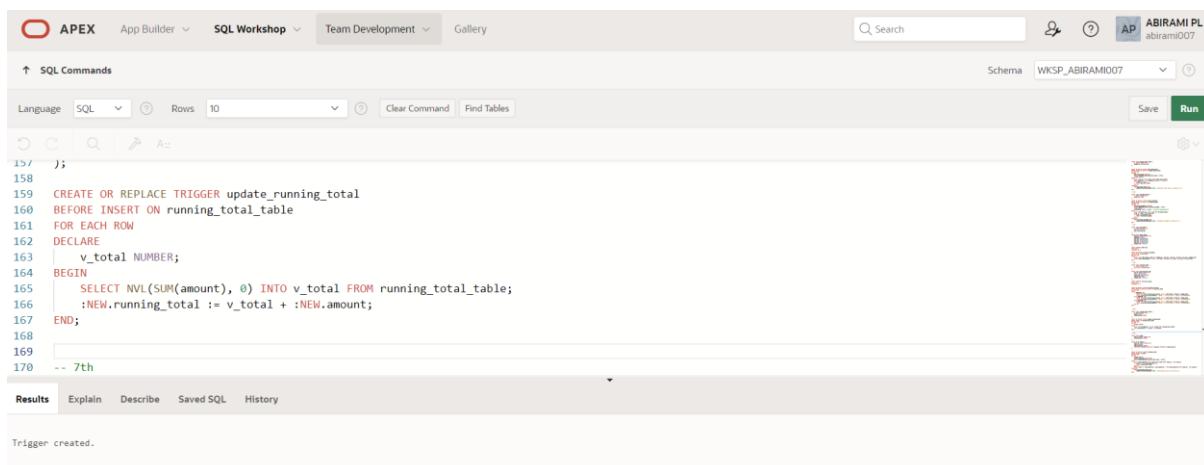
```
133
134 CREATE OR REPLACE TRIGGER log_user_activity
135 AFTER INSERT OR UPDATE OR DELETE ON activity_table
136 FOR EACH ROW
137 BEGIN
138     IF INSERTING THEN
139         INSERT INTO user_activity_log (log_id, action, table_name, record_id, change_time)
140         VALUES (activity_log_seq.NEXTVAL, '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);
```

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. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. The right side shows the user 'ABIRAMI PL' and schema 'WKSP_ABIRAMI007'. The main area is titled 'SQL Commands' with a language dropdown set to 'SQL'. The code area contains the PL/SQL trigger definition. The bottom pane shows the results of the command execution.

```
157 );
158 CREATE OR REPLACE TRIGGER update_running_total
159   BEFORE INSERT ON running_total_table
160   FOR EACH ROW
161   DECLARE
162     |   v_total NUMBER;
163   BEGIN
164     |   SELECT NVL(SUM(amount), 0) INTO v_total FROM running_total_table;
165     |   :NEW.running_total := v_total + :NEW.amount;
166   END;
167
168
169 -- 7th
170
```

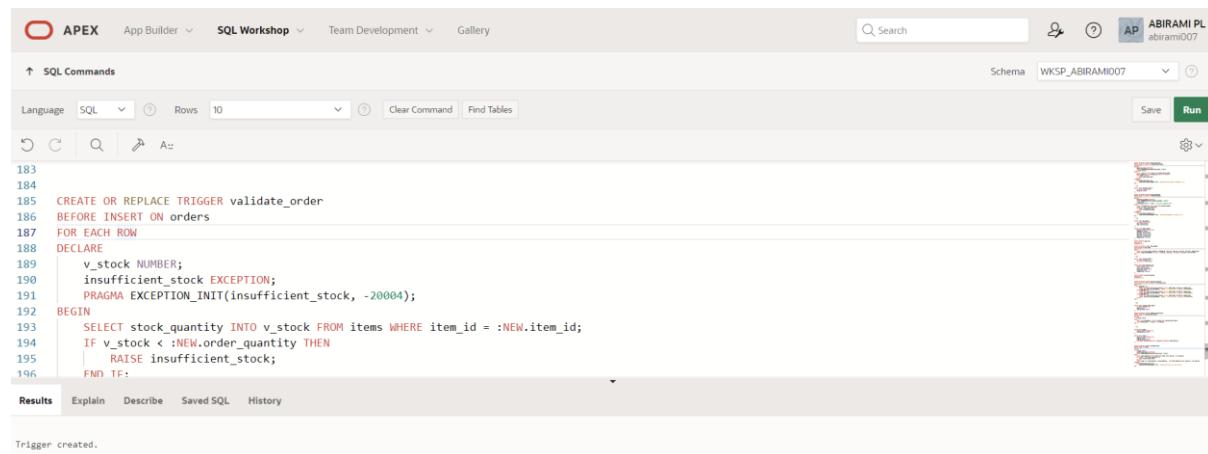
Results tab: Trigger created.

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

QUERY:

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

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows a schema tree for 'WKSP_ABIRAMI007'. The main workspace displays the PL/SQL code for the 'validate_order' trigger. The code is as follows:

```
183
184
185 CREATE OR REPLACE TRIGGER validate_order
186 BEFORE INSERT ON orders
187 FOR EACH ROW
188 DECLARE
189     v_stock NUMBER;
190     insufficient_stock EXCEPTION;
191     PRAGMA EXCEPTION_INIT(insufficient_stock, -20004);
192 BEGIN
193     SELECT stock_quantity INTO v_stock FROM items WHERE item_id = :NEW.item_id;
194     IF v_stock < :NEW.order_quantity THEN
195         RAISE insufficient_stock;
196     END IF;

```

The status bar at the bottom indicates "Trigger created."

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:

```
abirami@~/> 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('564c2d949eb21ad392f1d6ec'),
  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:

```
] abirami_07>  
abirami_07> db.restaurants.find( { grades: { $elemMatch: { grade: "A", score: 11, date: ISODate("2014-08-11T00:00:00Z") } } }, { restaurant_id: 1, name: 1, grades: 1 } );  
abirami_07>
```

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:

```
abirami_07> 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  
...   }  
... );  
abirami_07>
```

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:

```
abirami_07> db.restaurants.find({$and : [{"address.coord.1": {$gt : 42}}, {"address.coord.1": {$lte : 52}}]}, {_id:0, restaurant_id:1, name:1, address:1})  
abirami_07>
```

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:

```
abirami_07> 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',  
    grades: [],  
    name: '',  
    restaurant_id: '50017912'  
  },  
  {  
    address: {  
      building: '4704',  
      coord: [ -74.013391, 40.64943 ],  
      street: '3Rd Ave',  
      zipcode: '11220'  
    },  
    borough: 'Brooklyn',  
    cuisine: 'Other',  
    grades: [],  
    name: '',  
    restaurant_id: '50017925'  
  }]  
abirami_07>
```

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:

```
abirami_07> 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',
    grades: [],
    name: '',
    restaurant_id: '50017912'
  },
  {
    address: {
      building: '4704',
      coord: [ -74.013391, 40.64943 ],
      street: '3Rd Ave',
      zipcode: '11220'
    },
    borough: 'Brooklyn',
    cuisine: 'Other',
    grades: [],
    name: '',
    restaurant_id: '50017925'
  }
]
```

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:

```
abirami_07> 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'
  },
  {
    address: {
      building: '4704',
      coord: [ -74.013391, 40.64943 ],
      street: '3Rd Ave',
      zipcode: '11220'
    },
    borough: 'Brooklyn',
    cuisine: 'Other',
    grades: [],
    name: '',
    restaurant_id: '50017925'
  }
]
abirami_07>
```

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:

```
abirami_07> 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: 'Other',
    grades: [],
    name: '',
    restaurant_id: '50017912'
  },
  {
    _id: ObjectId('564c2d949eb21ad392f1d6f5'),
    address: {
      building: '4704',
      coord: [ -74.013391, 40.64943 ],
      street: '3Rd Ave',
      zipcode: '11220'
    },
    borough: 'Brooklyn',
    cuisine: 'Other',
    grades: [],
    name: '',
    restaurant_id: '50017925'
  }
]
abirami_07>
```

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:

```
abirami_07> 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'
    },
    borough: 'Manhattan',
    cuisine: 'Other',
    grades: [],
    name: '',
    restaurant_id: '50017912'
  },
  {
    _id: ObjectId('564c2d949eb21ad392f1d6f5'),
    address: {
      building: '4704',
      coord: [ -74.013391, 40.64943 ],
      street: '3Rd Ave',
      zipcode: '11220'
    },
    borough: 'Brooklyn',
    cuisine: 'Other',
    grades: [],
    name: '',
    restaurant_id: '50017925'
  }
]
abirami_07>
```

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:

```
] abirami_07> db.restaurants.find({ "grades.score": { $mod: [7, 0] } }, { restaurant_id: 1, name: 1, grades: 1 })  
abirami_07>
```

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:

```
abirami_07> db.restaurants.find({ name: /mon/i }, { name: 1, borough: 1, "address.coord": 1, cuisine: 1 })  
abirami_07>
```

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:

```
abirami_07> db.restaurants.find({ name: /^Mad/i }, { name: 1, borough: 1, "address.coord": 1, cuisine: 1 })  
abirami_07>
```

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:

```
abirami_07> db.restaurants.find({ "grades": { $elemMatch: { "score": { $lt: 5 } } } })  
abirami_07>
```

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:

```
abirami_07> db.restaurants.find({ "grades": { $elemMatch: { "score": { $lt: 5 } } }, "borough": "Manhattan" })  
abirami_07>
```

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:

```
abirami_07> db.restaurants.find({ "grades": { $elemMatch: { "score": { $lt: 5 } } }, $or: [{ "borough": "Manhattan" }, { "borough": "Brooklyn" }] })  
abirami_07>
```

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:

```
abirami_07> db.restaurants.find({ "grades": { $elemMatch: { "score": { $lt: 5 } } }, $or: [{ "borough": "Manhattan" }, { "borough": "Brooklyn" }], "cuisine": { $ne: "American" } })  
abirami_07>
```

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:

```
abirami_07> db.restaurants.find({ "grades": { $elemMatch: { "score": { $lt: 5 } } }, $or: [{ "borough": "Manhattan" }, { "borough": "Brooklyn" }], "cuisine": { $nin: ["American", "Chinese"] } })  
abirami_07>
```

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:

```
abirami_07> db.restaurants.find({ $and: [{ "grades.grade": "A", "grades.score": 2 }, { "grades.grade": "A", "grades.score": 6 }] })  
abirami_07>
```

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

QUERY:

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

OUTPUT:

```
abirami_07> db.restaurants.find({ $and: [{ "grades.grade": "A", "grades.score": 2 }, { "grades.grade": "A", "grades.score": 6 }], "borough": "Manhattan" })  
abirami_07> |
```

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:

```
abirami_07> db.restaurants.find({ $and: [{ "grades.grade": "A", "grades.score": 2 }, { "grades.grade": "A", "grades.score": 6 }], $or: [{ "borough": "Manhattan" }, { "borough": "Brooklyn" }] })  
abirami_07> |
```

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:

```
abirami_07> db.restaurants.find({ $and: [{ "grades.grade": "A", "grades.score": 2 }, { "grades.grade": "A", "grades.score": 6 }], $or: [{ "borough": "Manhattan" }, { "borough": "Brooklyn" }], "cuisine": { $ne: "American" } })  
abirami_07>
```

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:

```
abirami_07> 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"] } })  
abirami_07> |
```

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:

```
abirami_07> db.restaurants.find({ $or: [{ "grades.score": 2 }, { "grades.score": 6 }] })  
abirami_07>
```

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:

```
abirami_07> 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,
    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')
    }
  }
]
abirami_07>
```

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:

```
abirami_07> db.movies.find({ runtime: { $gt: 120 } })
[ {
  _id: ObjectId('665448ceab180f3fb9cdcdf6'),
  id: ObjectId('573a1390f22313caabced5967'),
  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', 'éduardMathé', 'Marcel Lévesque', 'Jean Aymé' ],
  poster: 'https://m.media-amazon.com/images/M/MV5BMHc1NTY3NDIzNl5BMl5BanBnXkFtZTgwNTIyODg5MTE@._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
  }
}
```

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

QUERY:

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

OUTPUT:

```

abirami_07> 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')
    }
  }
]
abirami_07>

```

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:

```

abirami_07> 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')
    }
  }
]
abirami_07>

```

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:

```
abirami_07> 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 Kaysen', 'John Ott'],
    num_flix_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, it cuts to the glowing metal and the hammering resumes.',
    countries: ['USA'],
    released: ISODate('1893-05-09T00:00:00.000Z'),
    directors: ['William R.L. Dickson'],
    rated: 'UNRATED',
    awards: { wins: 1, nominations: 0, text: '1 win.' },
    lastupdated: '2015-08-26 00:00:00.133000000',
    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('665448eab180f34b9cdcdf9'),
    id: ObjectId('573a1391e29313caacbd7a34'),
    plot: 'A kept woman runs into her one-time fiancé 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_flix_comments: 3,
    poster: 'https://s.media-amazon.com/images/M/MV5BZjJjMTU2NGQtNWRNN100ZjExLwExHTUTheNkNTU0NzRlMTA3XkEyXkFqcGdeQXVyNjUeNzK3MDc@._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'],
    awards: { wins: 1, nominations: 0, text: '1 win.' },
    lastupdated: '2015-08-02 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('66544905ab180f34b9cdcf8'),
    id: ObjectId('573a1391e29313caacbd8945'),
    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 Resing'
    ],
    num_flix_comments: 1,
    poster: 'https://s.media-amazon.com/images/M/MV5BNDVkYwM21tNzRlMy68NwQ4LYlhMjMtdI2ZDyOGVmMzJjXkEyXkFqcGdeQXVyNTgzMzU5MDI@._V1_SV1000_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 neglect ed wife and son in the country.',
    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 Hillier (titles)',
      'H.H. Caldwell (titles)'
    ]
  }
]
```

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

QUERY:

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

OUTPUT:

```
abirami_07> 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')
    }
  }
]
abirami_07>
```

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:

```
abirami_07> db.movies.find({ 'imdb.votes': { $gt: 1000 } })
[ {
    _id: ObjectId('665448ceab180f34b9cdcdf6'),
    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
    }
},
{
    _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('665448eeab180f34b9cdcdf7'),
  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/MV5BZjJiMTU2NGQtNWRkNi00ZjExLWEzMTUtMmNkNTU0NzRlMTA3XkEyXkFqcGdeQXVyNjUwNzk3NDc@._V1_SY1000_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
  }
}
```

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:

```
[abirami_07] > db.movies.find({ 'imdb.rating': { $gt: 7 } })
[{"_id": ObjectId('665448eeab180f34b9cdcdf7'),
 "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/MV5BZjJiMTU2NGQtNWRkNi00ZjExLWEzMUTMmNkNTU0NzRlMTA3XeYXkfFcGdeQXVyNjUwNzk3NDc@._V1_SY1000_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
 },
 },
 {"_id": ObjectId('66544905ab180f34b9cdcdf8'),
 "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',
   'John Wengraf',
   'Lillian Hall-Davis',
   'John B. Kelly',
   'John J. McGinnis',
   'John L. Williams',
   'John Wengraf',
   'Lillian Hall-Davis',
   'John B. Kelly',
   'John J. McGinnis',
   'John L. Williams'
 ],
 "num_mflix_comments": 1,
 "poster": "https://m.media-amazon.com/images/M/MV5BZjJiMTU2NGQtNWRkNi00ZjExLWEzMUTMmNkNTU0NzRlMTA3XeYXkfFcGdeQXVyNjUwNzk3NDc@._V1_SY1000_SX677_AL.jpg",
 "title": "The Farmer's Wife",
 "fullplot": "A married farmer falls under the spell of a slatternly woman from the city, who tries to convince him to drown his wife. The woman, played by Janet Gaynor, is a widow who has come to the country to find work. She seduces the farmer, played by George O'Brien, and they have an affair. The farmer's wife, played by Lillian Hall-Davis, is suspicious and confronts her husband about his infidelity. The farmer's wife threatens to leave him if he doesn't give up the woman, so the farmer agrees to drown her. However, the woman convinces the farmer to let her go, and they end up getting married. The film ends with the couple living happily ever after. The movie is a classic example of silent film era cinema, featuring some impressive special effects for its time.",

 "countries": [ 'USA' ],
 "released": ISODate('1925-03-01T00:00:00.000Z'),
 "directors": [ 'John Ford' ],
 "writers": [ 'John Ford', 'John Wengraf' ],
 "awards": { wins: 0, nominations: 0, text: 'No awards information available.' },
 "lastupdated": '2015-09-02 00:22:09.303000000',
 "year": 1925,
 "imdb": { rating: 7.5, votes: 1000, id: 14625 },
 "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
 },
 }
]
```

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:

```
abirami_07> db.movies.find({ 'tomatoes.viewer.rating': { $gt: 4 } })
[ {
  _id: ObjectId('66544905ab180f34b9cdcdf8'),
  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/MV5BNDVkyMwM2ItNzRiMy00NWQ4LTlhMjMtNDI1ZDYyOGVmMzJjXkEyXkFqcGdeQXVyNTgzMzU5MDI@._V1_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 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,
  imbd: { 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
  }
}
]
abirami_07> |
```

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

QUERY:

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

OUTPUT:

```
abirami_07> 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('665448eeab180f34b9cdcdf7'),
    id: ObjectId('573a1391f29313caabcd7a34'),
    plot: 'A kept woman runs into her one-time fianc  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_SY1000_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' ],
}
```

11.) Find all movies with title, languages, released, directors, writers, awards, year, genres, runtime, cast, countries from the 'movies' collection in MongoDB that have at least one nomination.

QUERY:

```
db.movies.find( { 'awards.nominations': { $gt: 0 } }, { title: 1, languages: 1, released: 1, directors: 1, writers: 1, awards: 1, year: 1, genres: 1, runtime: 1, cast: 1, countries: 1 } )
```

OUTPUT:

```
abirami_07> 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('665448ceab180f34b9cdcdf6'),
    genres: [ 'Action', 'Adventure', 'Crime' ],
    runtime: 399,
    cast: [ 'Musidora', 'édouardMathè', '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('66544905ab180f34b9cdcdf8'),
    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
  }
]
abirami_07>
```

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:

```
abirami_07> 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
  }
]
abirami_07>
```

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:

```
abirami_07> 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
  }
]
abirami_07>

abirami_07> 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' ]
  }
]
abirami_07>
```

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:

```
abirami_07> 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' ]
  }
]
abirami_07> |
```

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

RESULT: