

Creating and Managing Tables

EX_NO:1

DATE:

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

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. In the SQL Commands tab, a single line of SQL code is entered: `CREATE TABLE DEPT(Dept_id number(6) not null,Dept_name varchar(20),Manager_id varchar(6),location_id number(4));`. The 'Run' button is visible at the top right of the command entry area. Below the command, the results section displays the message "Table created." and "0.04 seconds". At the bottom of the page, footer information includes the user's email (220701008@rajalakshmi.edu.in), the schema name (abisheak08), the language (en), copyright notice (Copyright © 1999, 2023, Oracle and/or its affiliates.), and the software version (Oracle APEX 23.2.4).

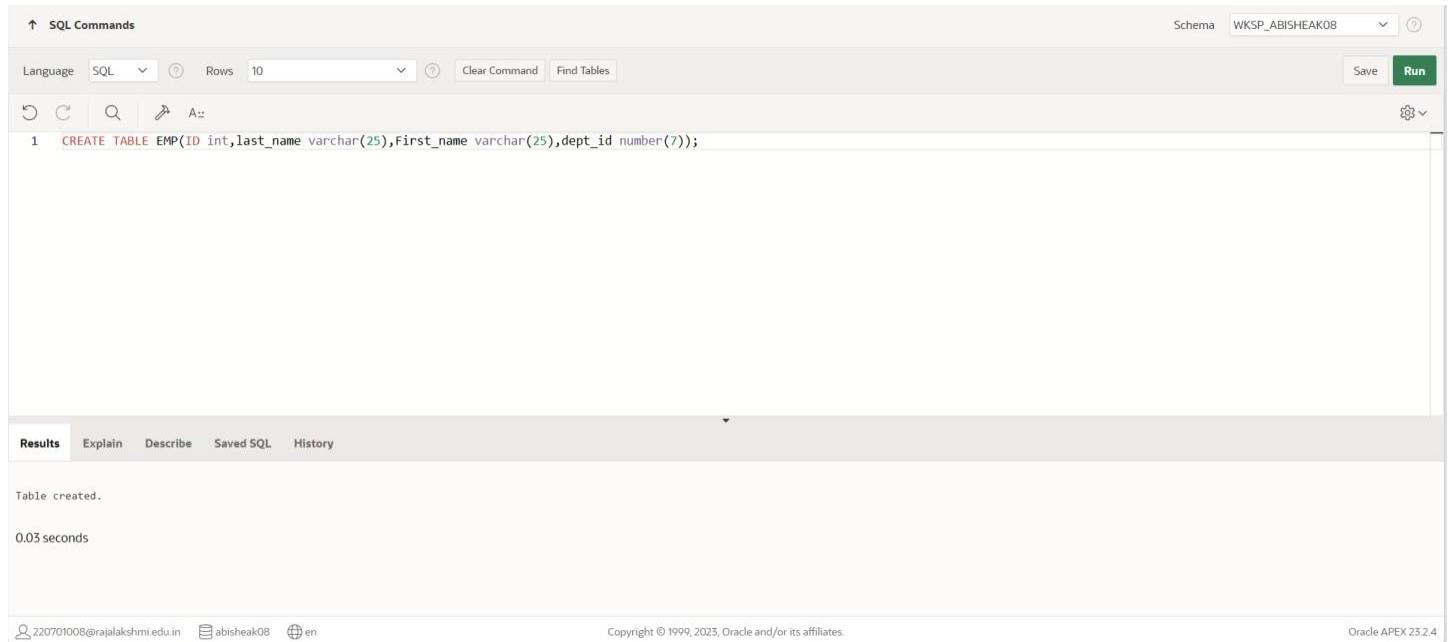
2. Create the EMP table based on the following instance chart. Confirm that the table is created.

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

QUERY:

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

OUTPUT:



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

```
1 CREATE TABLE EMI(ID int, last_name varchar(25),First_name varchar(25),dept_id number(7));
```

Below the command, the results section displays the output:

```
Table created.
```

Execution time: 0.05 seconds

At the bottom, the footer includes user information (220701008@rajalakshmi.edu.in, abishek08, en), copyright notice (Copyright © 1999, 2023, Oracle and/or its affiliates.), and the version (Oracle APEX 23.2.4).

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

QUERY:

Alter table EMI modify(Last_Name varchar2(50));

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. At the top, there are navigation links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. On the right, there is a search bar, a user profile for 'vijay PS PS india@123', and a schema dropdown set to 'WKSP_ABISHEAK08'. Below the header, the 'SQL Commands' tab is selected. The SQL editor contains the following command:

```
1 | Alter table Emp modify(Last_name varchar(50));
```

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

```
Table altered.
```

Execution time: 0.05 seconds.

At the bottom of the page, there are footer links for copyright information and the Oracle APEX version (25.2.4).

4. Create the EMPLOYEES2 table based on the structure of EMPLOYEES table. Include Only the Employee_id, First_name, Last_name, Salary and Dept_id columns. Name the columns Id, First_name, Last_name, salary and Dept_id respectively.

QUERY:

```
Create table E2(id number(7),First_name varchar2(25),Last_name varchar2(25),Salary int,Dept_id number(7));
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. In the top navigation bar, 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery' are visible. On the right, there's a search bar, a user icon for 'vijay PS PS india@123', and a schema dropdown set to 'WKSP_ABISHEAK08'. The main area is titled 'SQL Commands' with a 'Language' dropdown set to 'SQL' and 'Rows' set to '10'. Below this is a toolbar with icons for Undo, Redo, Find, Replace, and Run. The command input field contains the SQL statement: 'create table employees2(id number(6) not null,first_name varchar(20),last_name varchar(25),salary number(8,2),dept_id number(6) not null);'. The results tab is selected, showing the output: 'Table created.' and a execution time of '0.03 seconds'. The bottom status bar includes the URL '220701008@rajalakshmi.edu.in', the session ID 'abisheak08', and the language 'en'. Copyright information 'Copyright © 1999, 2023, Oracle and/or its affiliates.' and the version 'Oracle APEX 23.2.4' are also present.

5. Drop the EMP table.

QUERY:

Drop table emp;

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar and schema dropdown are identical to the previous screenshot. The main area shows the command input field with 'drop table emp;'. The results tab displays the output: 'Table dropped.' and an execution time of '0.07 seconds'. The bottom status bar and copyright information are consistent with the first screenshot.

6. Rename the EMPLOYEES2 table as EMP.

QUERY:

Rename employees2 to emp;

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. In the top navigation bar, 'APEX' is selected, along with 'SQL Workshop'. The main area contains a SQL command: '1 rename employees2 to emp;'. Below the command, the 'Results' tab is active, showing the output: 'Statement processed.' and '0.05 seconds'. The bottom status bar includes user information and the version 'Oracle APEX 23.2.4'.

```
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 'Department info';

comment on table emp is Employee info';

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. In the top navigation bar, 'APEX' is selected, along with 'SQL Workshop'. The main area contains three SQL commands: '1 comment on table dept is 'department info''; '2 comment on table emp is 'employee_ info''; and '3 |'. Below the command, the 'Results' tab is active, showing the output: 'Statement processed.' and '0.03 seconds'. The bottom status bar includes user information and the version 'Oracle APEX 23.2.4'.

```
1 comment on table dept is 'department info';
2 comment on table emp is 'employee_ info';
3 |
```

Statement processed.
0.03 seconds

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

QUERY:

```
Alter table emp drop column First_name;
```

OUTPUT:

The screenshot shows the Oracle Application Express interface. In the top navigation bar, the path is Home > SQL Workshop > SQL Commands. The main area contains the following SQL command:

```
Alter table emp drop column First_name;
```

Below the command, the results are displayed:

Table altered.
0.05 seconds

In the bottom right corner of the results pane, it says "Application Express 4.0.2.00.09".

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),last_name varchar(25),first_name varchar(25),userid varchar(25),salary number(9,2);

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. At the top, there are tabs for 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. On the right, there's a user profile for 'vijay PS PS india@125' and a 'Run' button. The main area is titled 'SQL Commands' with a 'Language' dropdown set to 'SQL'. It shows a single command: 'CREATE TABLE MY_EMPLOYEES1(ID number(4), last_name varchar(25),first_name varchar(25),salary number(9,2));'. Below the command, the 'Results' tab is selected, showing the output: 'Table created.' and '0.03 seconds'. The URL at the bottom is 'https://apex.oracle.com/pls/apex/r/apex/gallery/apps?session=12746530189742'.

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

OUTPUT:

APEX App Builder SQL Workshop Team Development Gallery

SQL Commands

Language SQL Rows 10 Clear Command Find Tables Save Run

```
1 INSERT INTO MY_EMPLOYEE VALUES(1,'Patle','Ralph','raptel',895);
2 INSERT INTO MY_EMPLOYEE VALUES(2,'Dancs','betty','bdancs',860);
```

Results Explain Describe Saved SQL History

1 row(s) inserted.
0.00 seconds

220701008@rajalakshmi.edu.in abisheak08 en Copyright © 1999, 2023, Oracle and/or its affiliates. Oracle APEX 23.2.4

3. Display the table with values.

QUERY:

Select * from MY_EMPLOYEE;

OUTPUT:

APEX App Builder SQL Workshop Team Development Gallery

SQL Commands

Language SQL Rows 10 Clear Command Find Tables Save Run

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

Results Explain Describe Saved SQL History

ID	LAST_NAME	FIRST_NAME	USER_ID	SALARY
2	Dancs	betty	bdancs	860
1	Patle	Ralph	raptel	895

2 rows returned in 0.01 seconds Download

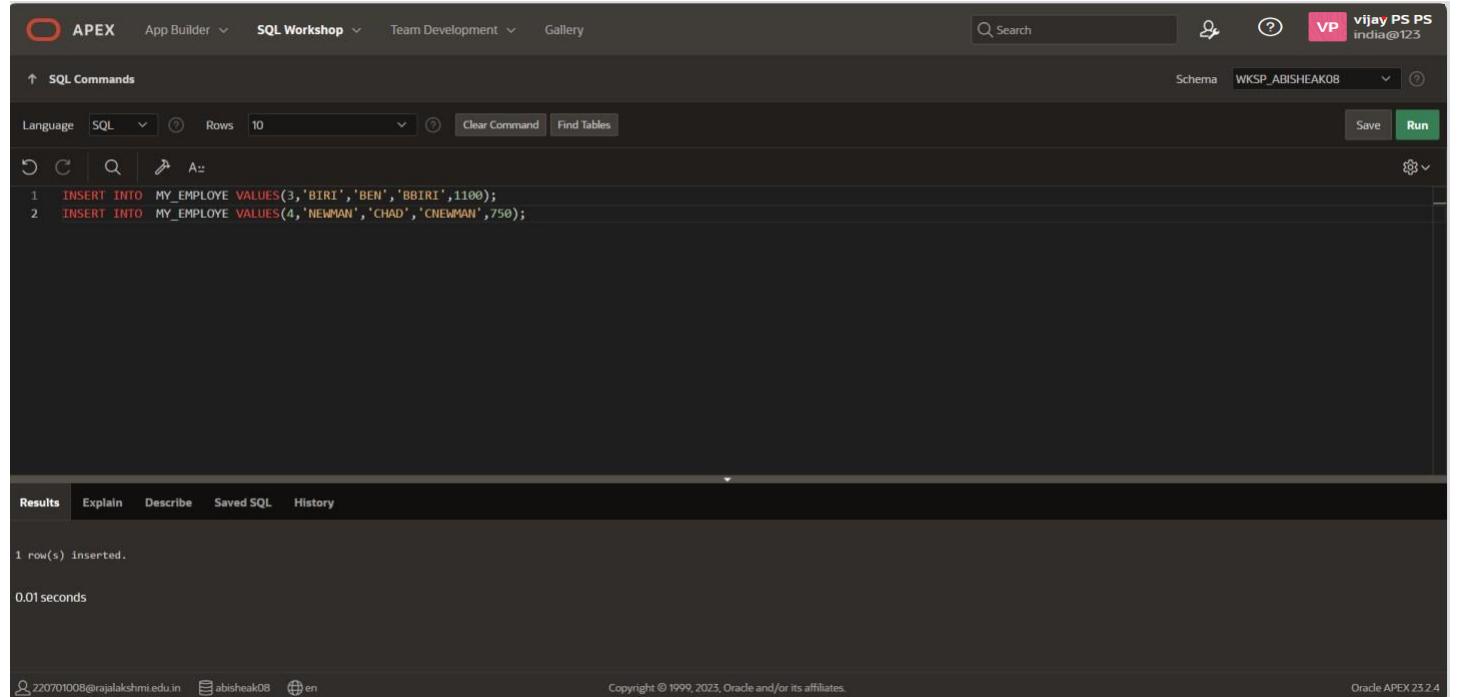
220701008@rajalakshmi.edu.in abisheak08 en Copyright © 1999, 2023, Oracle and/or its affiliates. Oracle APEX 23.2.4

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

QUERY:

Insert into MY_EMPLOYEE VALUES(2,'Dances','Betty','bdances,860);

OUTPUT:



The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. On the right, there's a user profile for 'vijay PS PS india@123'. The main area is titled 'SQL Commands' and contains the following SQL code:

```
1 INSERT INTO MY_EMPLOYEE VALUES(3,'BIRI','BEN','BBIRI',1100);
2 INSERT INTO MY_EMPLOYEE VALUES(4,'NEWMAN','CHAD','CNEWMAN',750);
```

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

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

At the bottom, the footer includes copyright information: 'Copyright © 1999, 2023, Oracle and/or its affiliates.' and 'Oracle APEX 23.2.4'.

5. Make the data additions permanent.

QUERY:

Select * from MY_EMPLOYEE;

OUTPUT:

The screenshot shows the Oracle APEX interface with the SQL Workshop tab selected. The query `SELECT * FROM MY_EMPLOYEE ORDER BY ID;` is entered in the command line. The results are displayed in a table with columns: ID, LAST_NAME, FIRST_NAME, USER_ID, and SALARY. The data is as follows:

ID	LAST_NAME	FIRST_NAME	USER_ID	SALARY
1	Patle	Ralph	raptel	895
2	Danics	betty	bdanics	860
3	BIRI	BEN	BBIRI	1100
4	NEWMAN	CHAD	CNEWMAN	750

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

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

QUERY:

UPDATE MY_EMPLOYEE SET last_name ='Drexler' where ID=3;

OUTPUT:

A screenshot of the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. On the right, there's a search bar, user information (vijay PS PS india@123), and a schema dropdown set to WKSP_ABISHEAK08. The main workspace shows a SQL command line with the following code:

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

The results pane below shows the output of the query:

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

At the bottom, the footer includes copyright information (Copyright © 1999, 2023, Oracle and/or its affiliates.) and the version (Oracle APEX 23.2.4).

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

QUERY:

UPDATE MY_EMPLOYEE SET Salary=1000 where Salary<900;

OUTPUT:

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

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

The results pane shows:

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

The footer remains the same with copyright and version information.

8. Delete Betty from MY_EMPLOYEE table.

QUERY:

Delete from MY_EMPLOYEE where first_name='Betty';

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. At the top, there are tabs for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there's a search bar, a user icon for 'vijay PS PS india@123', and a schema dropdown set to 'WKSP_ABISHEAK08'. Below the tabs, there's a toolbar with Language (SQL selected), Rows (10), Clear Command, Find Tables, Save, and Run buttons. The main area contains a SQL command: '1 DELETE FROM MY_EMPLOYEE WHERE FIRST_NAME='BETTY';'. The results section shows '0 row(s) deleted.' and a time of '0.01 seconds'. The bottom footer includes session information (220701008@rajalakshmi.edu.in, abishek08, en), copyright notice (Copyright © 1999, 2023, Oracle and/or its affiliates.), and the version 'Oracle APEX 23.2.4'.

9. Empty the fourth row of the emp table.

QUERY:

Delete from MY_EMPLOYEE where ID=4;

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. At the top, there are navigation links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. On the right, there is a search bar, user information for 'vijay PS PS india@123', and a schema dropdown set to 'WKSP_ABISHEAK08'. Below the header, the main area is titled 'SQL Commands' with a 'Language' dropdown set to 'SQL'. It shows a single command: 'DELETE FROM MY_EMPLOYEE WHERE ID=4;'. The results section shows '1 row(s) deleted.' and a execution time of '0.01 seconds'. The bottom of the page includes copyright information for Oracle and the APEX version.

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

1 row(s) deleted.

0.01 seconds
```

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

RESULT:

INCLUDING CONSTRAINTS

EX_NO:3

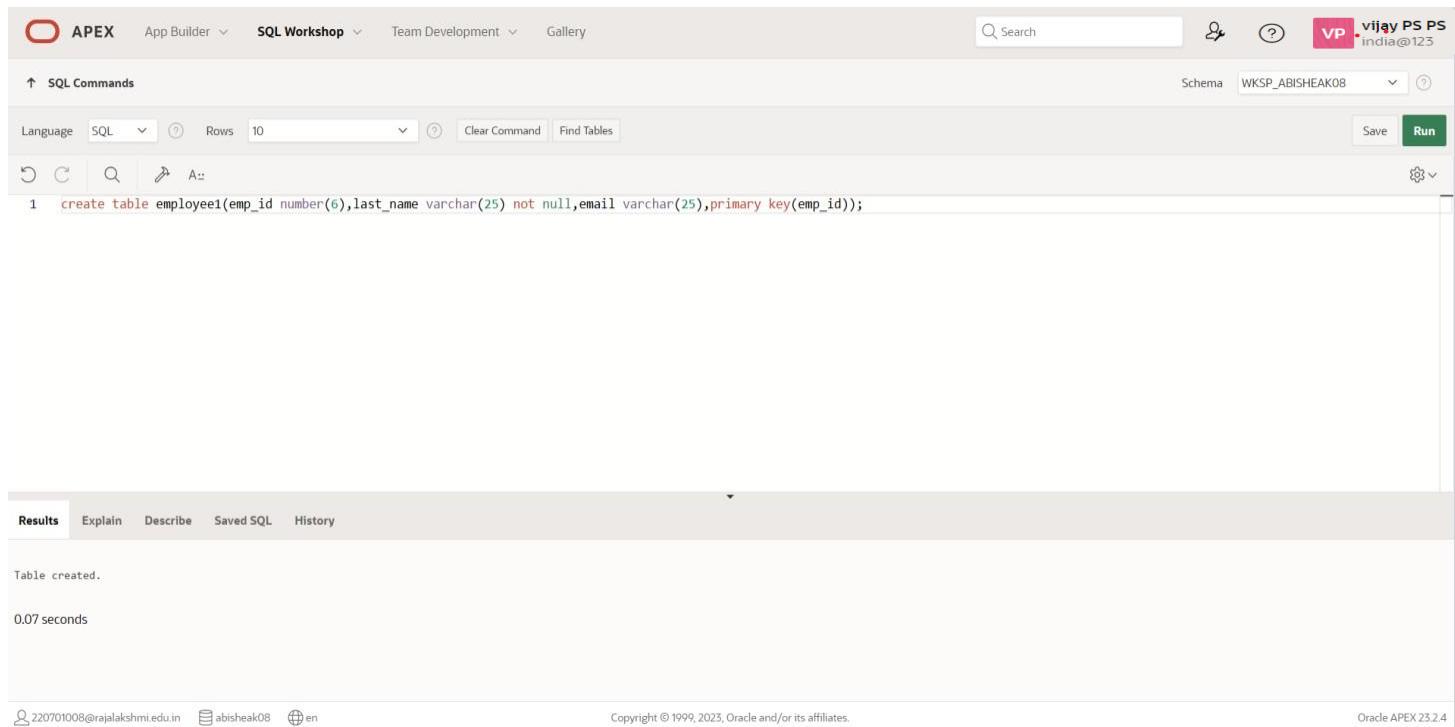
DATE:

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

QUERY:

Create table Employee1(Emp_Id number(6),last_name varchar(25) not null,email varchar(25),primary key(EMP_ID));

OUTPUT:



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

```
1 create table employee1(emp_id number(6),last_name varchar(25) not null,email varchar(25),primary key(emp_id));
```

Below the command line, the results section displays the output:

Table created.
0.07 seconds

At the bottom of the page, there are footer links: 220701008@rajalakshmi.edu.in, abisheak08, and Copyright © 1999, 2023, Oracle and/or its affiliates. Oracle APEX 23.2.4

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

QUERY:

Create table Dept(My_dept_id_pk int,Dept_name varchar(20),Manager_name varchar(25),Location_id varchar(25),Primary key(My_dept_id_pk));

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. At the top, there are navigation links for App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there is a search bar, a user icon for 'vijay PS PS india@123', and a schema dropdown set to 'WKSP_ABISHEAK08'. Below the header, the SQL Commands tab is selected. The SQL editor contains the following command:

```
1 create table dept(my_dept_id_pk int,dept_name varchar(20),manager_name varchar(25),primary key(my_dept_id_pk));|
```

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

```
Table created.
```

Execution time: 0.06 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 emp(id);

OUTPUT:

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

```
1 alter table employee1 add constraint my_employee1_emp_id_fk foreign key(emp_id) references employee1(emp_id);
```

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

```
Table altered.
```

Execution details: 0.07 seconds. The bottom right corner indicates Oracle APEX 23.2.4.

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

QUERY:

**Alter table EMP2 add Commission number (2,2)
Constraint Commission where (Commission>0);**
OUTPUT:

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

```
1 alter table employee1 add(commision number(8));
```

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

```
Table altered.
```

Execution details: 0.06 seconds. The bottom right corner indicates Oracle APEX 23.2.4.

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

RESULT:

Writing Basic SQL SELECT Statements

EX_NO:4

DATE:

1. The following statement executes successfully.

Identify the Errors

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

QUERY:

```
SELECT EMPLOYEE_ID,Last_name ,Salary*12 as Annual salary  
From employee;
```

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

QUERY:

```
Desc Department;
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there's a user profile for 'vijay PS PS india@123'. The main workspace has a search bar and a schema dropdown set to 'WKSP_ABISHEAK08'. Below the toolbar, there are buttons for Language (SQL selected), Rows (10), Clear Command, Find Tables, Save, and Run. The SQL command input field contains '1 desc department;'. The results pane shows the 'Describe' tab selected, displaying the structure of the 'DEPARTMENT' table. The table has two columns: 'DEPT_ID' (NUMBER) and 'DEPT_NAME' (VARCHAR2(20)). The 'DEPT_ID' column is primary key, nullable, and has a default value of '-'. The 'DEPT_NAME' column is nullable and has a default value of '-'. The bottom of the screen shows copyright information for Oracle and the APEX version.

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 Number,Last_name,job_id,Hire_Date,Emp_ID from Employee;
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. At the top, there are navigation links for App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there is a search bar, a user icon for 'vijay PS PS india@123', and a schema dropdown set to 'WKSP_ABISHEAK08'. Below the header, the 'SQL Commands' tab is selected, showing the following SQL code:

```
1 select employee_id, last_name, job_id, hire_date
2 from employees_table2;
```

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

EMPLOYEE_ID	LAST_NAME	JOB_ID	HIER_DATE
1	stfananthan	2669	052024

Below the table, it says '1 rows returned in 0.01 seconds' and has a 'Download' link. At the bottom of the page, there are footer links for user profile, email, and language, along with copyright information: 'Copyright © 1999, 2023, Oracle and/or its affiliates.' and 'Oracle APEX 23.2.4'.

4. Provide an alias STARTDATE for the hire date.

QUERY:

Select hire_date as Start_Date from employee;

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. At the top, there are navigation links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. On the right, there is a search bar, user profile information (vijay PS PS india@123), and a schema dropdown set to WKSP_ABISHEAK08. Below the header, the SQL Commands tab is selected, showing the following query:

```
1 select hier_date as startdate
2 from employees_table2;
```

The Results tab is selected, displaying the output:

STARTDATE
052024

Below the results, it says "1 rows returned in 0.01 seconds". The bottom of the page includes copyright information and a link to Oracle APEX 23.2.4.

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

QUERY:

Select Unique JOB CODES from employee;

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface, identical to the previous one but with a different query. The SQL Commands tab is selected, showing the following query:

```
1 select unique job_id
2 from employees_table2;
```

The Results tab is selected, displaying the output:

JOB_ID
2669
534
455

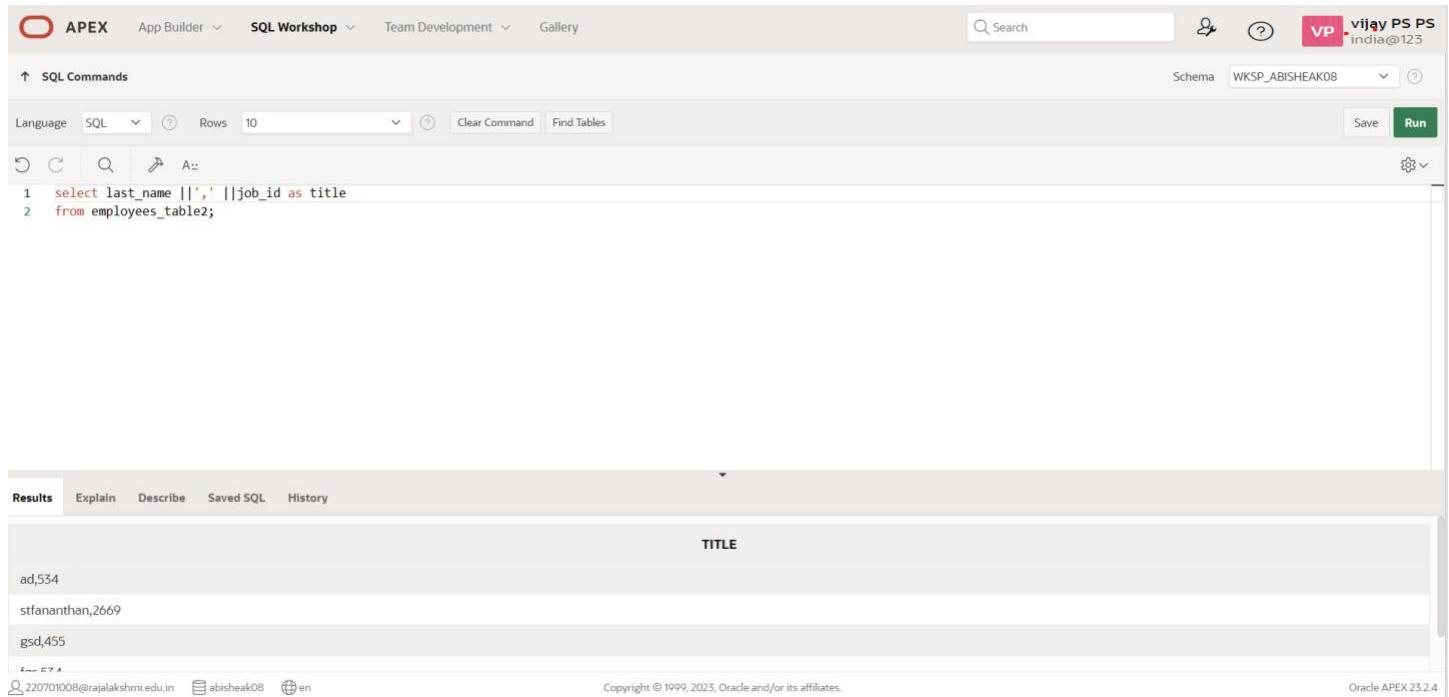
Below the results, it says "3 rows returned in 0.00 seconds". The bottom of the page includes copyright information and a link to Oracle APEX 23.2.4.

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

QUERY:

Select Last_name || ',' || JOB_ID as Title from employee;

OUTPUT:



The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there's a user icon for 'vijay PS PS india@123' and a search bar. The main area has tabs for SQL Commands, Results, Explain, Describe, Saved SQL, and History. The SQL Commands tab shows the following code:

```
1 select last_name ||','||job_id as title
2 from employees_table2;
```

The Results tab displays the output:

TITLE
ad,534
stfanthan,2669
gsd,455

At the bottom, it shows the connection details: 220701008@rajalakshmi.edu.in, Schema: WKSP_ABISHEAK08, and Oracle APEX 23.2.4.

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

QUERY:

Select Employee_ID ||','|| Last_name ||','|| Job_Id ||','|| Email ||','|| Salary ||','|| Hire_Date As "The Output" From Employee;

OUTPUT:



The screenshot shows the Oracle APEX SQL Workshop interface, similar to the previous one. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there's a user icon for 'vijay PS PS india@123' and a search bar. The main area has tabs for SQL Commands, Results, Explain, Describe, Saved SQL, and History. The SQL Commands tab shows the following code:

```
1 select employee_id||','|| last_name ||','||email||','||salary||','|| hier_date as "the_output"
2 from employees_table2;
```

The Results tab displays the output:

the_output
1,ad,ad,42,231
1,stfanthan,gs,27586,05/2024
25,gsd,gpa,455,23/44

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

RESULT:

RESTRICTING AND SORTING DATA

EX_NO:5

DATE:

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

QUERY:

Select last_name,salary from empo21 where salary>12000;

OUTPUT:

The screenshot shows the Oracle APEX interface with the SQL Workshop module selected. The query `select last_name,salary from empo21 where salary >12000;` is entered in the command line. The results page displays three rows of data:

LAST_NAME	SALARY
appu	100000
priya	100000
dhruv	20000

3 rows returned in 0.03 seconds

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

QUERY:

select last_name,department_number from empo21 where emp_id=176;

OUTPUT:

The screenshot shows the Oracle APEX interface with the SQL Workshop module selected. The query `select last_name,department_number from empo21 where emp_id=176;` is entered in the command line. The results page displays one row of data:

LAST_NAME	DEPARTMENT_NUMBER
appu	111

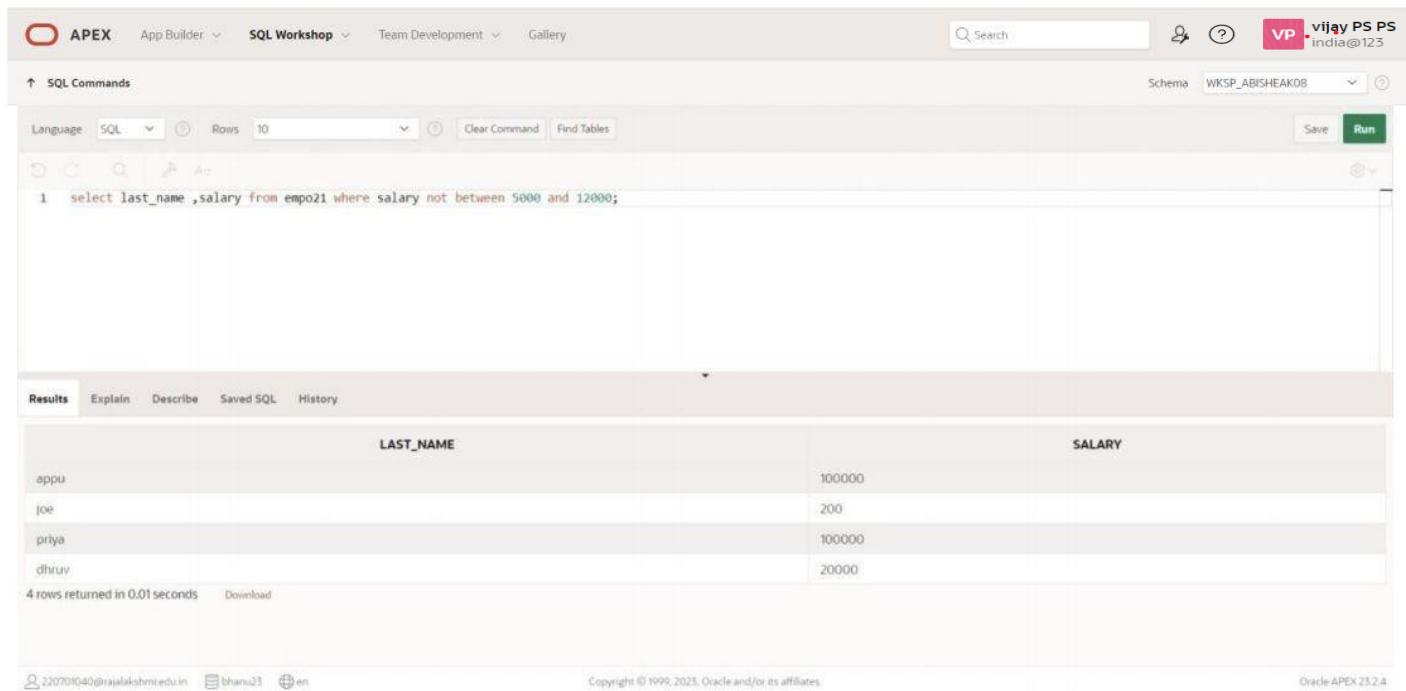
1 rows returned in 0.05 seconds

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

QUERY:

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

OUTPUT:



LAST_NAME	SALARY
appu	100000
joe	200
priya	100000
dhruv	20000

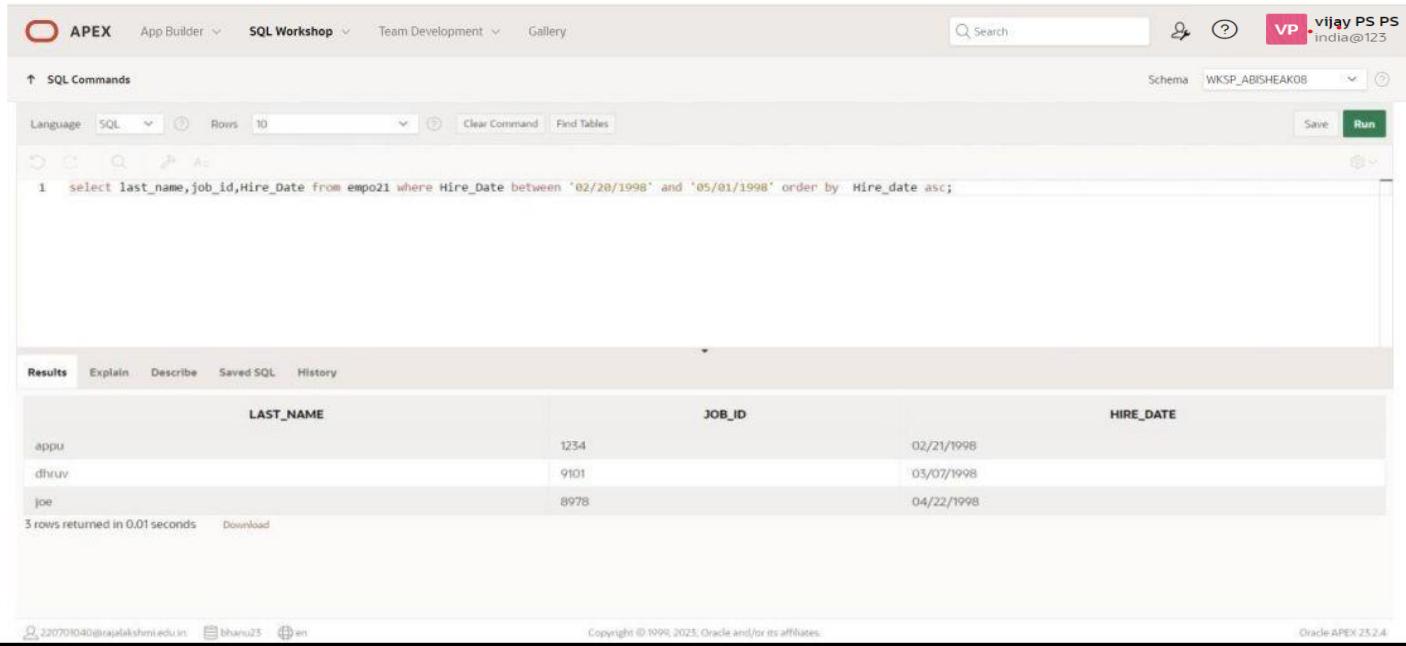
4 rows returned in 0.01 seconds [Download](#)

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.(hints: between)

QUERY:

```
Select last_name,job_id,hire_date from empo21 where hire_date between '02/20/1998' and '05/01/1998' order by hire_date asc;
```

OUTPUT:



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

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

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. On the right, there's a user profile for 'vijay PS' and a schema dropdown set to 'WKSP_ABISHEAK08'. The main area has tabs for 'SQL Commands' and 'Results'. Under 'Results', the 'Explain' tab is selected. The SQL command entered is:

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

The results table has two columns: 'LAST_NAME' and 'DEPARTMENT_NUMBER'. The data is:

LAST_NAME	DEPARTMENT_NUMBER
appu	111
dhruv	333
priya	111

Below the table, it says '3 rows returned in 0.00 seconds' and has a 'Download' link. At the bottom of the page, there are footer links for '220701040@rajalakshmi.edu.in', 'bhansu23', and 'en', along with copyright information 'Copyright © 1999, 2023, Oracle and/or its affiliates.' and 'Oracle APEX 23.2.4'.

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

QUERY:

```
Select last_name as "EMPLOYEE", salary as "MONTHLY_SALARY" from empo21 where salary between 5000 and 12000 and department in(20,50) order by Name_emp asc;
```

OUTPUT:

APEX App Builder SQL Workshop Team Development Gallery

Schema: WKSP_ABISHEAK08

SQL Commands

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

```
1 Select last_name as "EMPLOYEE", salary as "MONTHLY_SALARY" From empo21
2 where salary between 5000 and 12000 and
3 department in(26,50)
4 order by last_name asc;
```

Results

EMPLOYEE	MONTHLY_SALARY
dhruv	11000
priya	5000

2 rows returned in 0.01 seconds Download

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

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

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

OUTPUT:

APEX App Builder SQL Workshop Team Development Gallery

Schema: WKSP_ABISHEAK08

SQL Commands

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

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

Results

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

2 rows returned in 0.01 seconds Download

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

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

QUERY:

SELECT last_name, job_id FROM empo21 WHERE manager_id IS NULL;

OUTPUT:

A screenshot of the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. On the right, there's a search bar, a user icon for 'vijay PS PS india@123', and a schema dropdown set to 'WKSP_ABISHEAK08'. The main area shows a SQL command window with the following code:

```

1 SELECT last_name, job_id
2 FROM emp021
3 WHERE manager_id IS NULL;

```

The results tab displays the output of the query:

LAST_NAME	JOB_ID
joe	8978
dhiruv	9101

Below the table, it says "2 rows returned in 0.01 seconds" and has a "Download" link. At the bottom, there are footer links for 220701040@rajalakshmi.edu.in, bharu23, and en, along with copyright information: Copyright © 1999, 2023, Oracle and/or its affiliates. Oracle APEX 23.2.4.

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

QUERY:

`SELECT last_name, salary, commission_pct FROM emp021 WHERE commission_pct IS NOT NULL ORDER BY salary DESC, commission_pct DESC;` **OUTPUT:**

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

```

1 SELECT last_name, salary, commission_pct
2 FROM emp021
3 WHERE commission_pct IS NOT NULL
4 ORDER BY salary DESC, commission_pct DESC;

```

The results tab shows the output:

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

Below the table, it says "3 rows returned in 0.01 seconds" and has a "Download" link. At the bottom, there are footer links for 220701040@rajalakshmi.edu.in, bharu23, and en, along with copyright information: Copyright © 1999, 2023, Oracle and/or its affiliates. Oracle APEX 23.2.4.

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

QUERY:

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

OUTPUT:

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

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

The results section displays the output:

LAST_NAME
payaae
dhauvae

2 rows returned in 0.01 seconds

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

QUERY:

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

OUTPUT:

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

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

The results section displays the output:

LAST_NAME
priyae
dhruvae

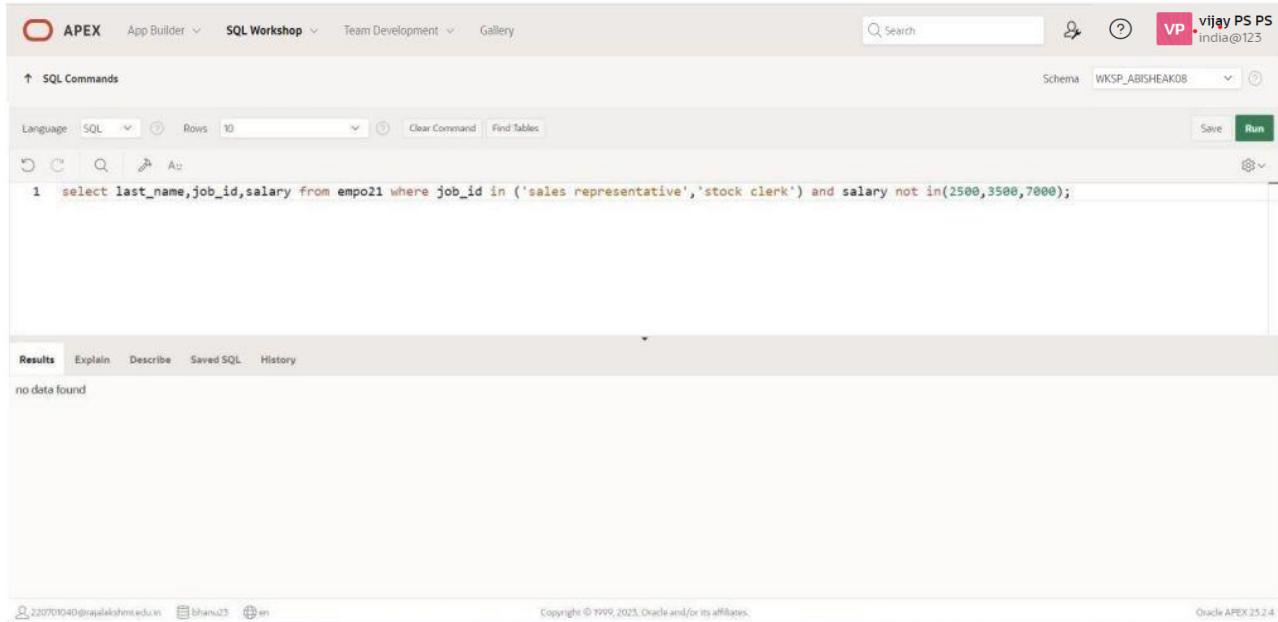
2 rows returned in 0.00 seconds

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

QUERY:

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

OUTPUT:



The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. On the right, there's a user profile for 'vijay PS PS' and a schema dropdown set to 'WKSP_ABISHEAK08'. The main area is titled 'SQL Commands' with a sub-section 'Language: SQL'. Below it, the query is written:

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

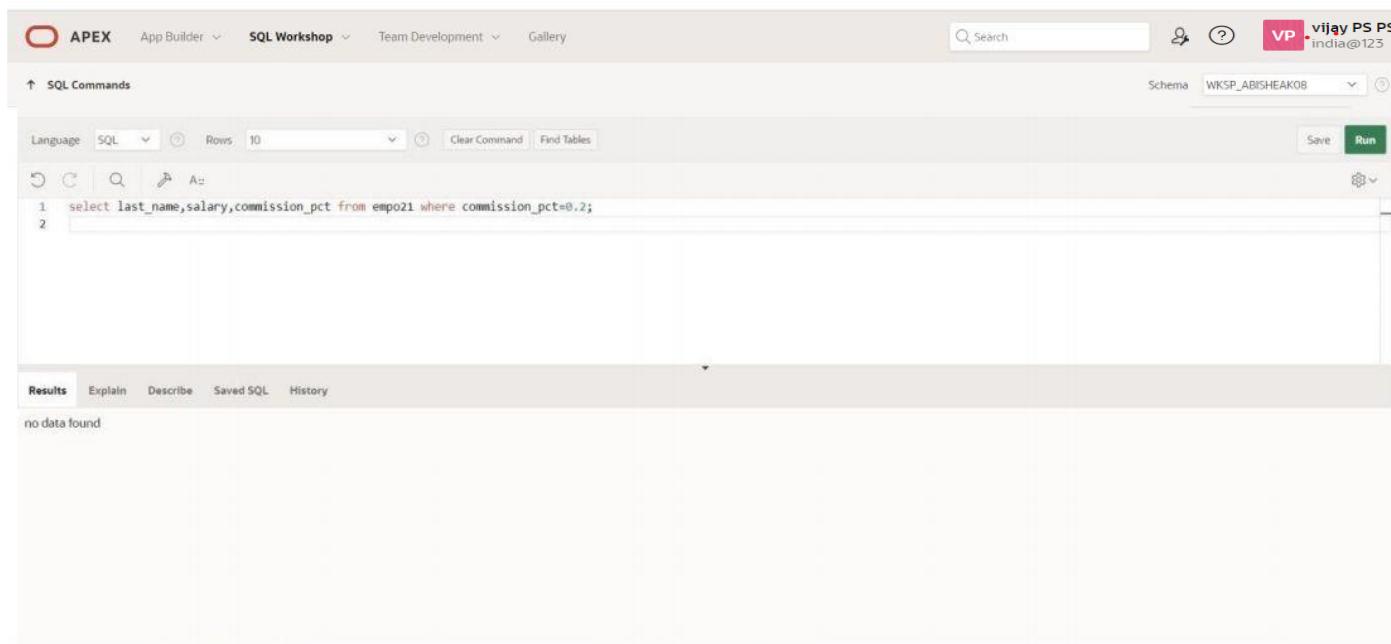
Below the query, the 'Results' tab is selected, showing the message 'no data found'.

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

QUERY:

```
select last_name,salary,commission_pct from employees where commission_pct=0.2;
```

OUTPUT:



The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. On the right, there's a user profile for 'vijay PS PS' and a schema dropdown set to 'WKSP_ABISHEAK08'. The main area is titled 'SQL Commands' with a sub-section 'Language: SQL'. Below it, the query is written:

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

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

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

RESULT:

SINGLE ROW FUNCTIONS

EX_NO:6

DATE:

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

QUERY:

```
select sysdate from dual;
```

OUTPUT:

APEX App Builder SQL Workshop Team Development Gallery

SQL Commands Schema WKSP_ABISHEAK08

Language SQL Rows 10 Clear Command Find Tables Save Run

```
1 select sysdate from dual;
```

Results Explain Describe Saved SQL History

SYSDATE

03/12/2024

1 rows returned in 0.02 seconds Download

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

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

QUERY:

OUTPUT:

APEX App Builder SQL Workshop Team Development Gallery

SQL Commands Schema WKSP_ABISHEAK08

Language SQL Rows 10 Clear Command Find Tables Save Run

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

Results Explain Describe Saved SQL History

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

4 rows returned in 0.03 seconds Download

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

3.Modify your query lab_03_02.sql to add a column that subtracts the old salary from the new salary. Label the column Increase.

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The query is executed successfully, returning 4 rows of data. The results are displayed in a table with columns: EMP_ID, LAST_NAME, SALARY, new_salary, and Increase. The data is as follows:

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

4 rows returned in 0.01 seconds

1. Write a query that displays the last name (with the first letter uppercase and all other letters lowercase) and the length of the last name for all empo21 whose name starts with the letters J, A, or M. Give each column an appropriate label. Sort the results by the empo21' last names.

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The query is executed successfully, returning 1 row of data. The results are displayed in a table with columns: INITCAP(LAST_NAME) and Length_of_last_name. The data is as follows:

INITCAP(LAST_NAME)	Length_of_last_name
Joe	3

1 rows returned in 0.01 seconds

2. Rewrite the query so that the user is prompted to enter a letter that starts the last name. For example, if the user enters H when prompted for a letter, then the output should show all empo21 whose last name starts with the letter H.

QUERY:

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

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery'. On the right, there's a user profile for 'vijay PS PS' and the schema 'WKSP_ABISHEAK08'. The main area is titled 'SQL Commands' with a 'Run' button. The code entered is:

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

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

OUTPUT:

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface with the same top navigation and user profile as the previous screenshot. The 'SQL Commands' section contains the same query as above. The 'Results' tab is selected, displaying the following data:

LAST_NAME	MONTHS_WORKED
Harsun	317
praya	318
Joe	364
appu	366

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

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

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

QUERY:

```
select last_name||' earns'||salary||' monthly but wants'||salary*3 as "DREAM_SALARIES" from empo21;
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', 'Gallery', a search bar, and a user profile for 'vijay PS PS india@123'. The main area is titled 'SQL Commands' with tabs for 'Language', 'SQL', 'Rows' (set to 10), 'Clear Command', and 'Find Tables'. A command has been entered in the text area:

```
1 select last_name||' earns'||salary||' monthly but wants'||salary*3 as "DREAM_SALARIES" from empo21;
```

Below the command, the results are displayed in a table with a single column labeled 'DREAM_SALARIES'. The output shows four rows of data:

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

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

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', 'Gallery', a search bar, and a user profile for 'vijay PS PS india@123'. The main area is titled 'SQL Commands' with tabs for 'Language', 'SQL', 'Rows' (set to 10), 'Clear Command', and 'Find Tables'. A command has been entered in the text area:

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

Below the command, the results are displayed in a table with two columns: 'LAST_NAME' and 'SALARY'. The output shows four rows of data:

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

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

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

QUERY:

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

The screenshot shows the Oracle APEX SQL Workshop interface. At the top, there are tabs for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there is a user profile for 'vijay PS PS' with the email 'india@123'. Below the tabs, there is a search bar and a 'Run' button. The main area is titled 'SQL Commands' and contains the following SQL code:

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

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

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

At the bottom left, it says '4 rows returned in 0.00 seconds' and 'Download'. At the bottom right, it says 'Copyright © 1999, 2023, Oracle and/or its affiliates.' and 'Oracle APEX 23.2.4'.

OUTPUT:

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

QUERY:

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

OUTPUT:

APEX App Builder SQL Workshop Team Development Gallery

Search Schema: WKSP_ABISHEAK08

vijay PS PS india@123

SQL Commands

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

```
1 SELECT last_name,hire_date,TO_CHAR(hire_date,'Day') as Day from emp021 order by
2 TO_CHAR(hire_date,'Day');
```

Results Explain Describe Saved SQL History

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

4 rows returned in 0.00 seconds Download

220701040@rajalakshmi.edu.in bhanu23 en Copyright © 1999, 2023, Oracle and/or its affiliates. Oracle APEX 23.2.4

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

RESULT:

DISPLAYING DATA FROM MULTIPLE TABLES

EX_NO:7

DATE:

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (which is selected), 'Team Development', and 'Gallery'. On the right, there's a user profile for 'vijay PS PS India@123'. The main area has tabs for 'SQL Commands' and 'Results'. The SQL command entered is: `select e.last_name,e.department_number,d.dept_id from empo21 e,dept23 d where e.department_number=d.dept_id;`. The results section displays the following data:

LAST_NAME	DEPARTMENT_NUMBER	DEPT_ID
Joe	80	80

1 rows returned in 0.01 seconds. The bottom of the screen shows copyright information: Copyright © 1999, 2023, Oracle and/or its affiliates. and Oracle APEX 23.2.4.

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. On the right, there's a user profile for 'vijay PS PS India@123'. The main area has tabs for 'SQL Commands' and 'Results'. The SQL command entered is: `select distinct job_id,loc_id from empo21 e,dept23 d where e.department_number=d.dept_id and e.department_number=80;`. The results section displays the following data:

JOB_ID	LOC_ID
8978	49

1 rows returned in 0.03 seconds. The bottom of the screen shows copyright information: Copyright © 1999, 2023, Oracle and/or its affiliates. and Oracle APEX 23.2.4.

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. On the right, there's a user profile for 'vijay PS PS India@123'. The main area has tabs for 'SQL Commands' and 'Results'. Under 'SQL Commands', the query is:

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

Under 'Results', the output is:

JOB_ID	LOC_ID
8978	49

1 rows returned in 0.03 seconds. The bottom status bar shows the URL '220701040@rajalakshmi.edu.in', a session identifier 'bhansu25', and the copyright notice 'Copyright © 1999-2023, Oracle and/or its affiliates'. It also indicates 'Oracle APEX 23.2.4'.

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. On the right, there's a user profile for 'vijay PS PS India@123'. The main area has tabs for 'SQL Commands' and 'Results'. Under 'SQL Commands', the query is:

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

Under 'Results', the output is:

JOB_ID	LOC_ID
8978	49

1 rows returned in 0.03 seconds. The bottom status bar shows the URL '220701040@rajalakshmi.edu.in', a session identifier 'bhansu25', and the copyright notice 'Copyright © 1999-2023, Oracle and/or its affiliates'. It also indicates 'Oracle APEX 23.2.4'.

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

QUERY:

```
Select e.last_name,e.department_number,e.job_id,d.dept_name from empo21 e join dept d  
on(e.department_number=d.dept_id) join location on (d.location_id=location.location_id) where  
lower(location.city)='toronto';
```

OUTPUT:

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

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

The results table has two columns: JOB_ID and LOC_ID. One row is returned, with values 8978 and 49 respectively. The status bar at the bottom indicates "1 rows returned in 0.03 seconds".

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

QUERY:

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

OUTPUT:

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

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

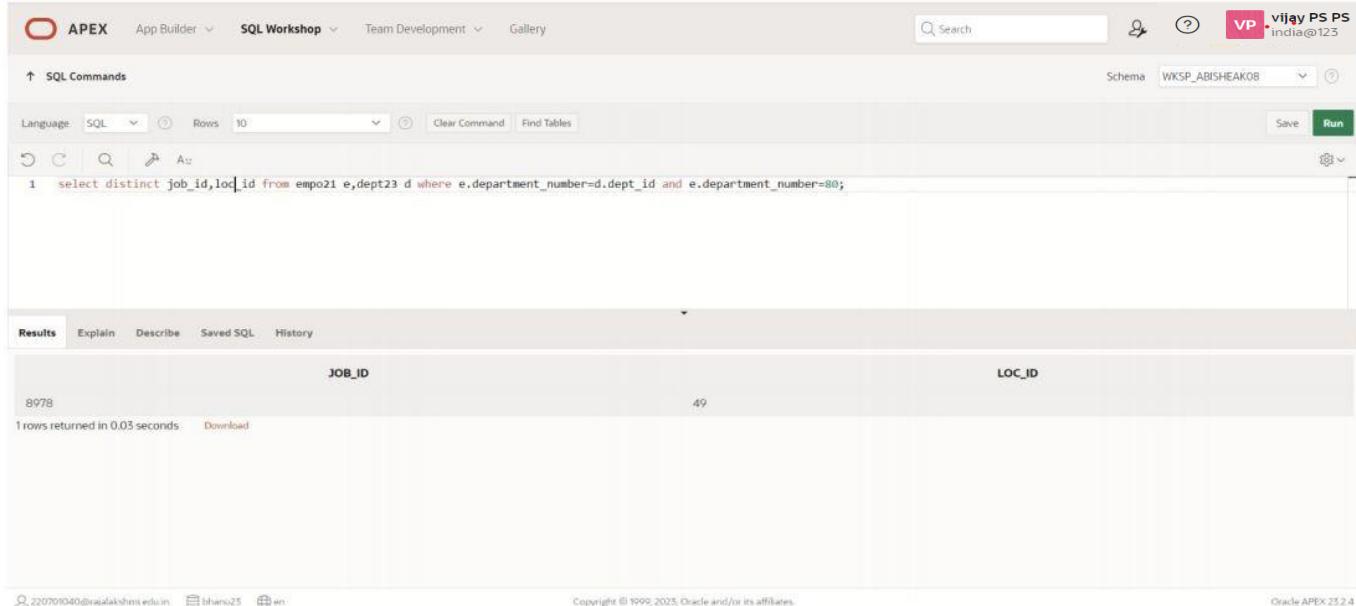
The results table has two columns: JOB_ID and LOC_ID. One row is returned, with values 8978 and 49 respectively. The status bar at the bottom indicates "1 rows returned in 0.03 seconds".

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

QUERY:

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

OUTPUT:



The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows a user profile for 'vijay PS PS' with the email 'india@123'. The main area is titled 'SQL Commands' with a schema dropdown set to 'WKSP_ABISHEAKOB'. The SQL editor contains the following query:

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

The results section shows the output of the query:

JOB_ID	LOC_ID
8978	49

Below the table, it says '1 rows returned in 0.03 seconds' and there is a 'Download' link. The bottom of the page includes standard footer links and the text 'Copyright © 1999-2025, Oracle and/or its affiliates.' and 'Oracle APEX 25.2.4'.

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

QUERY:

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

OUTPUT:

A screenshot of the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. On the right, there's a search bar, a user profile for 'vijay PS PS india@123', and a schema dropdown set to 'WKSP_ABISHEAK08'. The main workspace shows a SQL command line with the following query:

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

The results section displays the output of the query:

JOB_ID	LOC_ID
8978	49

1 rows returned in 0.03 seconds. Below the results, there are tabs for Explain, Describe, Saved SQL, and History. At the bottom, there are footer links for copyright information and the Oracle APEX version.

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

QUERY:

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

OUTPUT:

A screenshot of the Oracle APEX SQL Workshop interface, similar to the previous one but with a different query. The top navigation bar and user profile are identical. The main workspace shows the same SQL command as above:

```
1 SELECT e.last_name, e.job_id, d.dept_name, e.salary, j.grade_level
2 FROM empo21 e JOIN dept23 d
3 ON (e.dept_id = d.dept_id)
4 JOIN job_grade j
5 ON (e.salary BETWEEN j.lowest_sal AND j.highest_sal);
```

The results section shows 'no data found'.

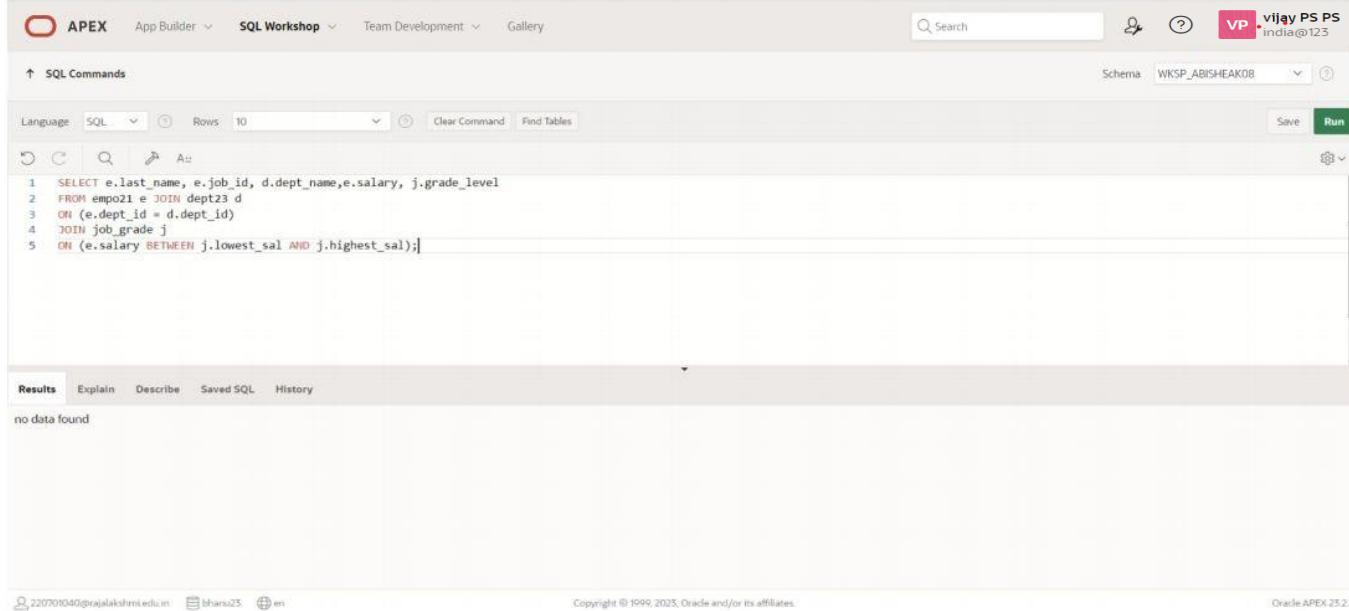
At the bottom, there are footer links for copyright information and the Oracle APEX version.

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

QUERY:

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

OUTPUT:



The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there's a user profile for 'vijay PS PS' and the schema 'WKSP_ABISHEAKOB'. The main area has tabs for SQL Commands, Results, Explain, Describe, Saved SQL, and History. The SQL Commands tab shows the following query:

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

The Results tab indicates "no data found".

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. In the top navigation bar, 'APEX' is selected, followed by 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery'. On the right, there's a user profile for 'vijay PS PS' with the email 'vijay_ps_ps@india@125'. The schema dropdown is set to 'WKSP_BHANU25'. Below the navigation, the 'SQL Commands' tab is active, showing the following SQL code:

```

1 SELECT e.last_name, e.job_id, d.dept_name, e.salary, j.grade_level
2 FROM emp021 e JOIN dept03 d
3 ON (e.dept_id = d.dept_id)
4 JOIN job_grade j
5 ON (e.salary BETWEEN j.lowest_sal AND j.highest_sal);

```

Below the code, the 'Results' tab is selected, showing the message 'no data found'. At the bottom of the page, the footer includes the copyright notice 'Copyright © 1999-2025, Oracle and/or its affiliates.' and the version 'Oracle APEX 23.2.4'.

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

RESULT

AGGREGATING DATA USING GROUP FUNCTIONS

EX_NO : 8

DATE:

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

TRUE

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

FALSE

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

FALSE

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', 'Gallery', 'Search', and a user profile for 'vijay PS PS India@123'. The main area has tabs for 'SQL Commands' and 'Results'. The SQL command entered is:

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

The results section displays the following table:

Maximum	Minimum	sum	Average
90000	60000	300000	75000

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

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. On the right, there is a search bar, a help icon, and a user profile for 'vijay PS PS' with the email 'india@123'. Below the navigation is a toolbar with options for Language (SQL), Rows (10), Clear Command, Find Tables, Save, and Run.

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

The results section displays a table with the following data:

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

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

At the bottom, the footer includes the URL '220701040@rajalakshmi.edu.in', session information 'bhansu25', and 'en', copyright notice 'Copyright © 1999-2023, Oracle and/or its affiliates.', and the version 'Oracle APEX 25.2.4'.

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar and user profile are identical to the previous screenshot. The toolbar includes Language (SQL), Rows (10), Clear Command, Find Tables, Save, and Run.

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

The results section displays a table with the following data:

JOB_ID	COUNT(*)
47	1

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

At the bottom, the footer includes the URL '220701040@rajalakshmi.edu.in', session information 'bhansu25', and 'en', copyright notice 'Copyright © 1999-2023, Oracle and/or its affiliates.', and the version 'Oracle APEX 25.2.4'.

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX interface with the SQL Workshop tab selected. The query `select count(distinct manager_id) "Number of managers" from empb;` is run, resulting in the output `3`. The schema is set to `WKSP_ABISHEAK08`.

DIFFERENCE
3

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

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX interface with the SQL Workshop tab selected. The query `select max(salary)-min(salary) difference from empb;` is run, resulting in the output `5000`. The schema is set to `WKSP_ABISHEAK08`.

Difference
5000

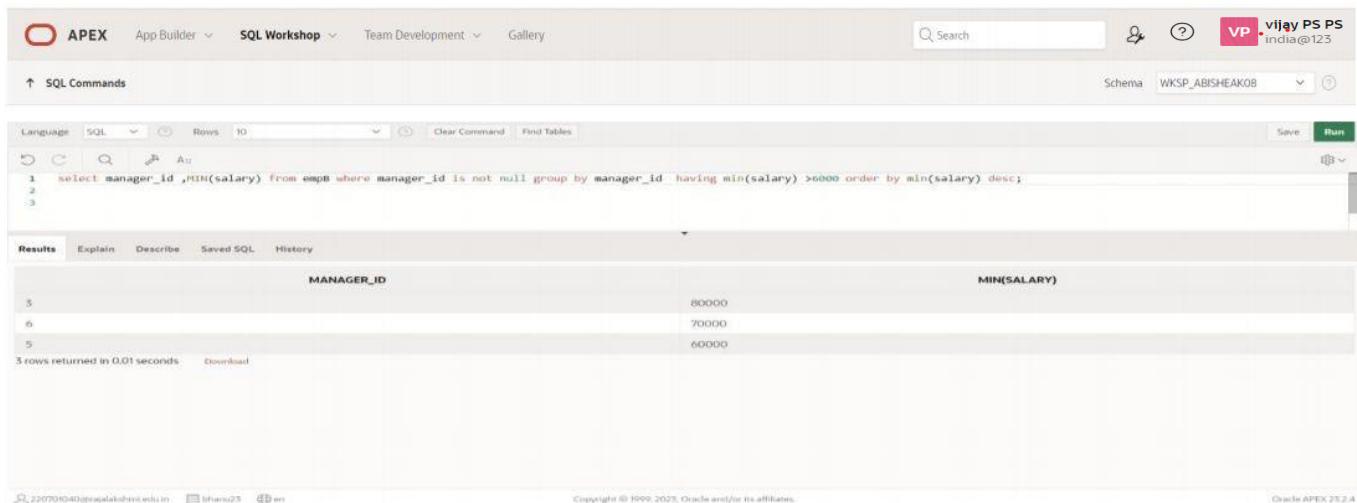
Copyright © 1999, 2023, Oracle and/or its affiliates.

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

QUERY:

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

OUTPUT:



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

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

The results table has two columns: MANAGER_ID and MIN(SALARY). The data is:

MANAGER_ID	MIN(SALARY)
5	80000
6	70000
5	60000

3 rows returned in 0.01 seconds.

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery'. On the right, there's a search bar, a help icon, and a user profile for 'vijay PS PS' at 'india@123'. Below the navigation is a toolbar with 'Language: SQL', 'Rows: 10', 'Clear Command', and 'Find Tables'. The main area contains a code editor with the following SQL query:

```

1 | SELECT count(*) total ,sum(decode (to_char(hire_date,'YYYY'),1995,1,0))"1995",
2 | sum(decode(to_char(hire_date , 'YYYY'),1996,1,0))"1996",sum(decode(to_char(hire_date,'YYYY'),1997,1,0))"1997",sum(decode(to_char(hire_date,'YYYY'),1998,1,0)) "1998" from empB;
3 |
4 |

```

Below the code editor is a results grid with columns 'TOTAL', '1995', '1996', '1997', and '1998'. The data row shows values: TOTAL=4, 1995=1, 1996=0, 1997=1, 1998=0. A note says '1 rows returned in 0.01 seconds'. The bottom of the page includes copyright information and a link to Oracle APEX 25.2.4.

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar and user profile are identical to the previous screenshot. The code editor contains the following SQL query:

```

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

```

Below the code editor is a results grid with columns 'job', 'Dept20', 'dept50', 'dept80', 'dept90', and 'TOTAL'. The data row shows values: job=44, Dept20=60000, dept50=90000, dept80=80000, dept90=70000, TOTAL=200000. A note says '4 rows returned in 0.01 seconds'. The bottom of the page includes copyright information and a link to Oracle APEX 25.2.4.

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. At the top, there are tabs for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there's a search bar, user information (vijay PS PS, vijay PS PS, india@123), and a schema dropdown (WKSP_ABISHEAK08). Below the tabs, there are buttons for Language (SQL), Rows (10), Clear Command, Find Tables, Save, and Run.

The SQL command entered is:

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

The Results section displays the output:

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

Below the table, it says "4 rows returned in 0.07 seconds" and has a "Download" link.

At the bottom, there are links for 220701040@rajalakshmi.edu.in, bharus25, and en. Copyright information (Copyright © 1999, 2023, Oracle and/or its affiliates) and the text "Oracle APEX 25.2.4".

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

Faculty Signature	
-------------------	--

RESULT:

SUB QUERIES

EX_NO:9

DATE:

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. On the right, there's a user profile for 'vijay PS PS India@123'. The main workspace has a toolbar with 'Language' (set to SQL), 'Rows' (set to 10), and other options like 'Clear Command' and 'Find Tables'. Below the toolbar is a code editor window containing the following SQL query:

```
1 select last_name,hire_date from emp021 where dept_id=(select dept_id from emp021 where last_name='Janu') and last_name not in('Janu');
2 
```

Below the code editor is a results grid. The first row shows the column headers 'LAST_NAME' and 'HIRE_DATE'. The second row contains two entries: 'Jones' with hire date '05/01/1998' and 'Williams' with hire date '02/19/1994'. A note at the bottom left says '2 rows returned in 0.00 seconds'. The bottom of the page includes footer links for '2207010-40@irejalekshmi.edu.in', 'bhavna25', 'en', 'Copyright © 1999, 2023, Oracle and/or its affiliates.', and 'Oracle APEX 23.2.2'.

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX interface with the SQL Workshop tab selected. The schema is set to WKSP_ABISHEAK08. The query entered is:

```
1 select emp_id, last_name, salary from empo21 where salary > (select avg(salary) from empo21) order by salary;
```

The results section displays the following data:

EMP_ID	LAST_NAME	SALARY
2	Janu	60000
5	Brown	80000

2 rows returned in 0.01 seconds

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX interface with the SQL Workshop tab selected. The schema is set to WKSP_ABISHEAK08. The query entered is:

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

The results section displays the following data:

EMPLOYEE_ID	LAST_NAME
2	Janu
4	Jones
5	Williams

3 rows returned in 0.03 seconds

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

QUERY:

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

OUTPUT:

APEX App Builder SQL Workshop Team Development Gallery

SQL Commands

Schema: WKSP_ABISHEAK08

Language: SQL Rows: 10 Save Run

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

Results Explain Describe Saved SQL History

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

5 rows returned in 0.06 seconds Download

<https://apex.oracle.com/pls/apex/r/apex/workspace/home?session=10064742540...> Copyright © 1999, 2023, Oracle and/or its affiliates. Oracle APEX 23.2.4

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

QUERY:

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

OUTPUT:

APEX App Builder SQL Workshop Team Development Gallery

SQL Commands

Schema: WKSP_ABISHEAK08

Language: SQL Rows: 10 Save Run

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

Results Explain Describe Saved SQL History

LAST_NAME	SALARY
davies	50000
Williams	20000
Jones	48000

3 rows returned in 0.01 seconds Download

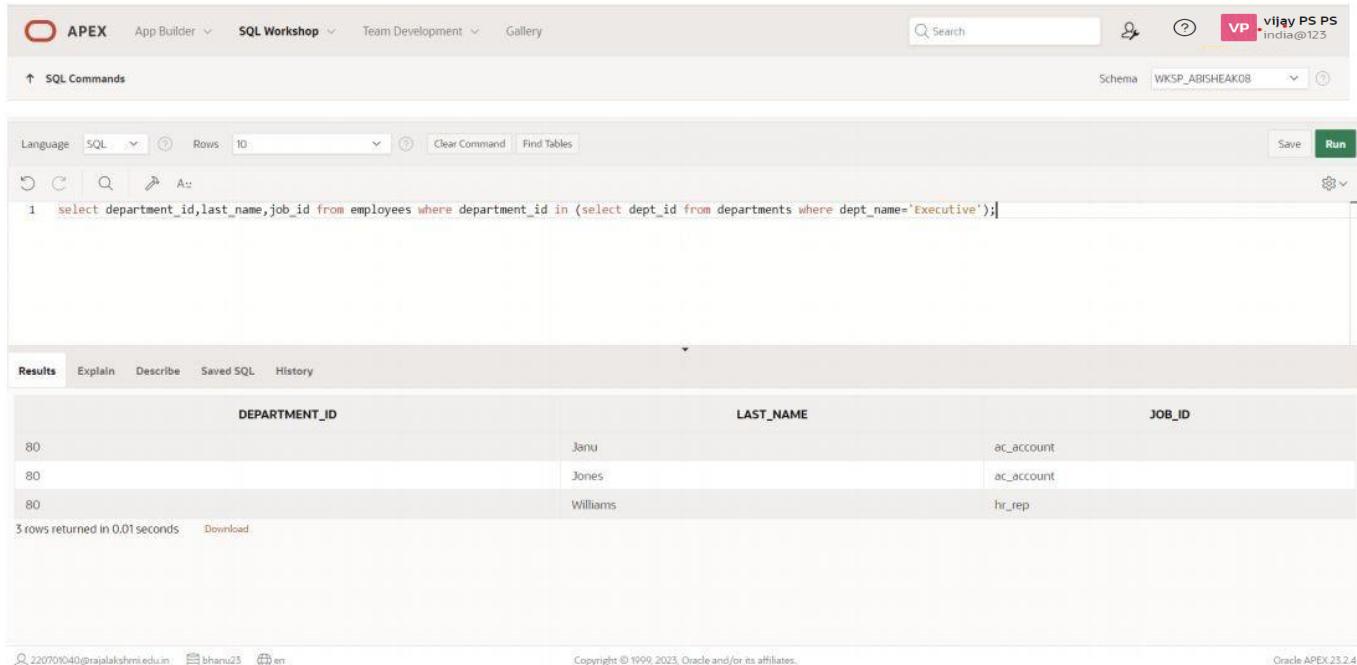
220701040@rajalakshmi.edu.in bhanu25 en Copyright © 1999, 2023, Oracle and/or its affiliates. Oracle APEX 23.2.4

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

QUERY:

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

OUTPUT:



The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. The user's name, "vijay PS PS India@125", is displayed in the top right corner. The SQL Commands tab is selected. The SQL editor contains the following query:

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

The results section displays the following data:

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

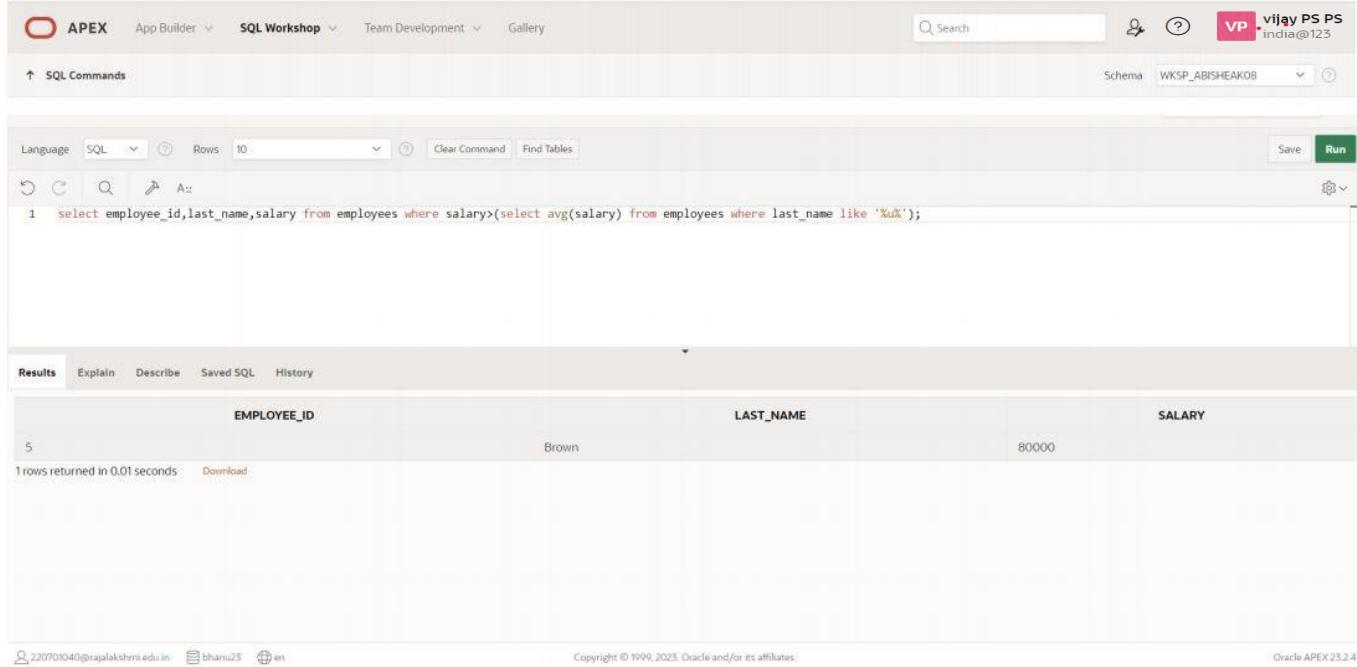
3 rows returned in 0.01 seconds. There is a "Download" link below the results.

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

QUERY:

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

OUTPUT:



The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. The user's name, "vijay PS PS India@125", is displayed in the top right corner. The SQL Commands tab is selected. The SQL editor contains the following query:

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

The results section displays the following data:

EMPLOYEE_ID	LAST_NAME	SALARY
5	Brown	80000

1 rows returned in 0.01 seconds. There is a "Download" link below the results.

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

RESULT:

USING THE SET OPERATORS

EX_NO:10

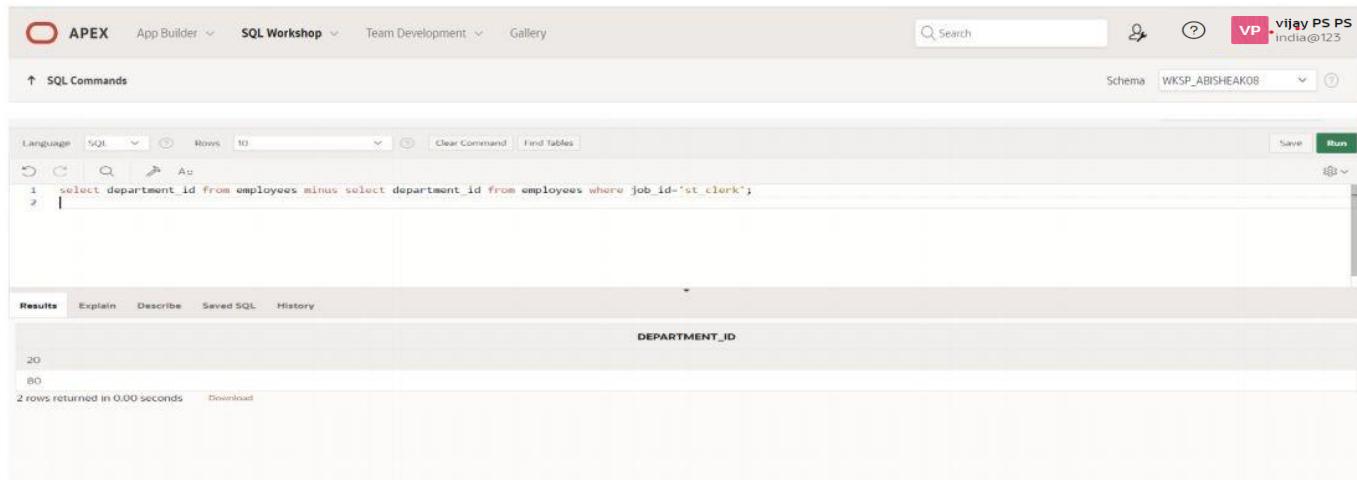
DATE:

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

QUERY:

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

OUTPUT:



The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. On the right, there's a user profile for 'vijay PS PS' and a schema dropdown set to 'WKSP_ABISHEAK08'. The main workspace is titled 'SQL Commands' and contains the following code:

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

The results section shows a single row of data:

DEPARTMENT_ID
20

Below the table, it says '2 rows returned in 0.00 seconds'.

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. The top right shows a user profile for 'vijay PS PS' and the schema 'WKSP_ABISHEAK08'. The main area has tabs for 'SQL Commands' and 'Results'. The SQL command entered is:

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

The results section shows a single row:

COUNTRY_ID	STATE_PROVINCE
34	TN

1 rows returned in 0.01 seconds. There is a 'Download' link below the results.

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. The top right shows a user profile for 'vijay PS PS' and the schema 'WKSP_ABISHEAK08'. The main area has tabs for 'SQL Commands' and 'Results'. The SQL command entered is:

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

The results section shows two rows:

JOB_ID	DEPARTMENT_ID
sales_rep	20
st_clerk	50

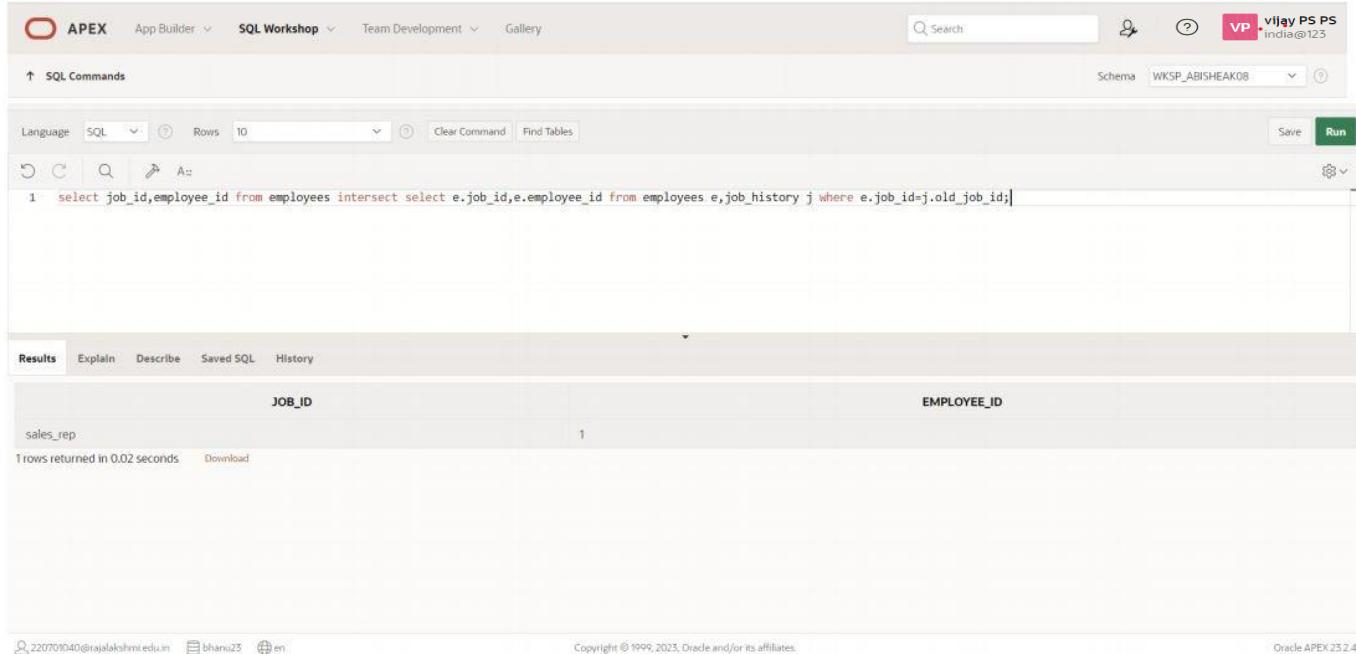
2 rows returned in 0.00 seconds. There is a 'Download' link below the results.

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

QUERY:

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

OUTPUT:



The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. On the right, there's a user profile for 'Vijay PS' and the schema 'WKSP_ABISHEAK08'. The main area has tabs for 'SQL Commands' and 'Results'. Under 'SQL Commands', the query is pasted. Under 'Results', the output shows one row with 'JOB_ID' as 'sales_rep' and 'EMPLOYEE_ID' as '1'.

JOB_ID	EMPLOYEE_ID
sales_rep	1

1 rows returned in 0.02 seconds. Download

5.)The HR department needs a report with the following specifications: - Last name and department ID of all the employees from the EMPLOYEES table, regardless of whether or not they belong to a department. - Department ID and department name of all the departments from the DEPARTMENTS table, regardless of whether or not they have employees working in them Write a compound query to accomplish this.

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. At the top, there are tabs for APEX, App Builder, SQL Workshop, Team Development, and Gallery. On the right, there's a search bar, user information for 'vijay PS PS' (india@123), and a schema dropdown set to 'WKSP_ABISHEAKOB'. Below the tabs, there's a toolbar with Language (SQL), Rows (10), Clear Command, Find Tables, Save, and Run buttons.

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

The results section displays a table with two columns: 'Name' and 'DEPARTMENT_ID'. The data is as follows:

Name	DEPARTMENT_ID
Janu	80
davies	20
Jones	80
Brown	50
Williams	80
Executive	80
IT	25
CS	46
IT	80

Below the table, it says '9 rows returned in 0.01 seconds' and has a 'Download' link. At the bottom, there are footer links for '220701040@rajalokhm.edu.in', 'shans23', and 'an', along with copyright information 'Copyright © 1999, 2025, Oracle and/or its affiliates.' and 'Oracle APEX 23.2.4'.

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

RESULT:

CREATING VIEWS

EX_NO:11

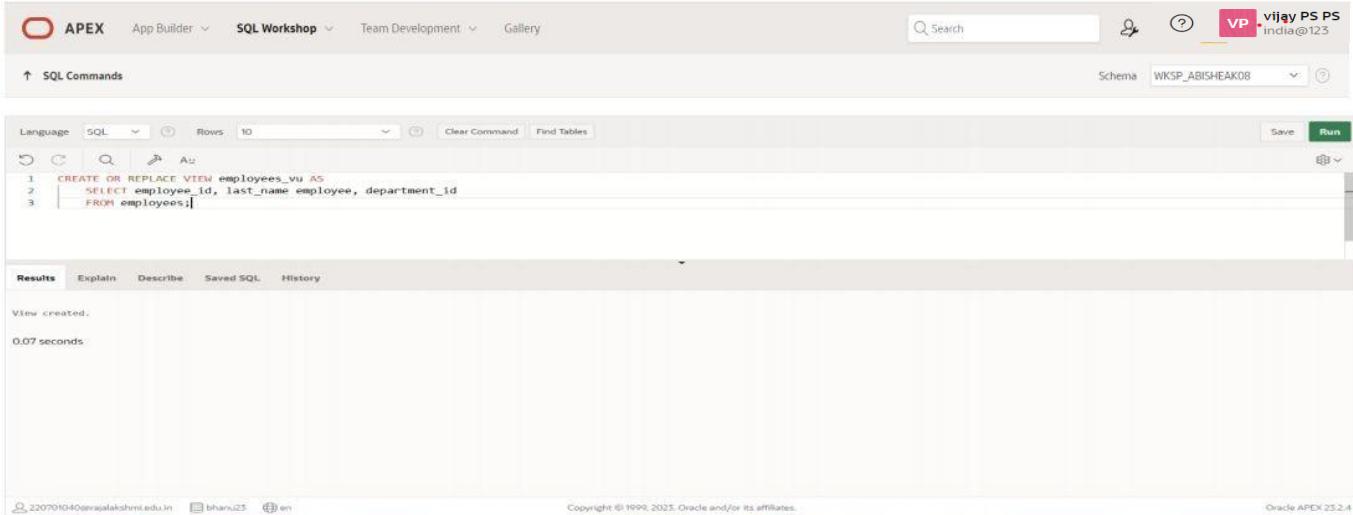
DATE:

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

QUERY:

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

OUTPUT:



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

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

Below the code, the results show:

```
View created.
0.07 seconds
```

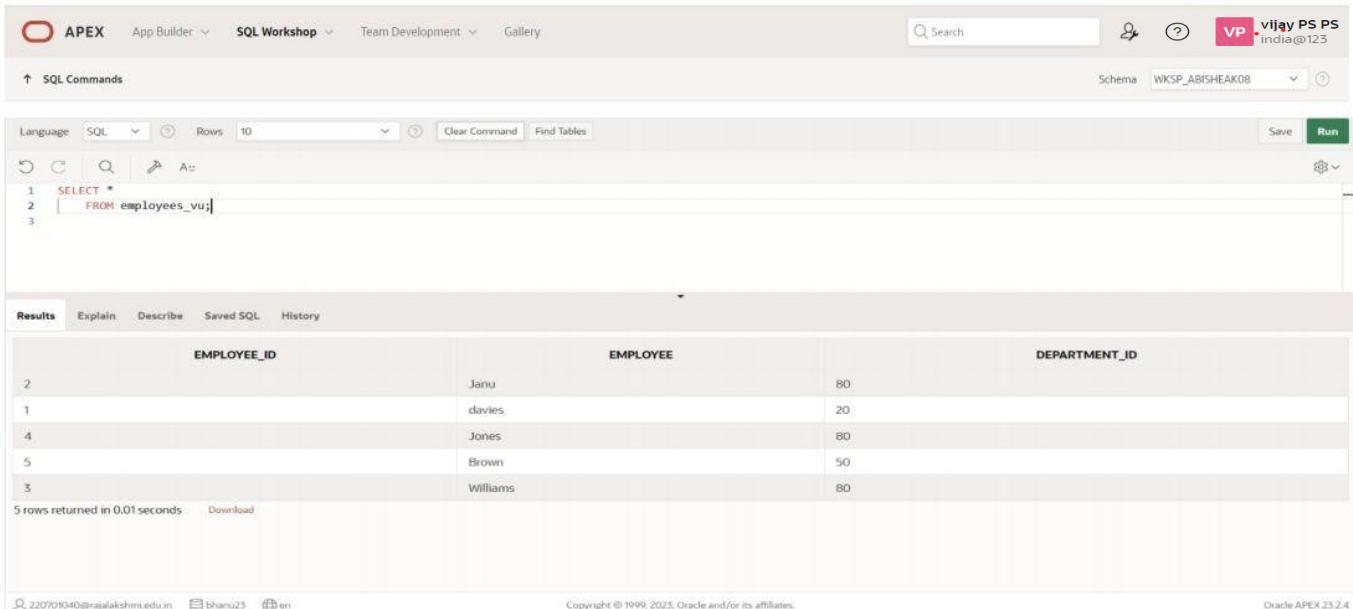
At the bottom, the URL is 220701040@rajalakshmi.edu.in, the session ID is bhanu23, and the page number is 1 of 1. Copyright information and Oracle APEX 23.2.4 are also visible.

2.) Display the contents of the EMPLOYEES_VU view.

QUERY:

```
select * from employees_vu;
```

OUTPUT:



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

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

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

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

At the bottom, the URL is 220701040@rajalakshmi.edu.in, the session ID is bhanu23, and the page number is 1 of 1. Copyright information and Oracle APEX 23.2.4 are also visible.

3.)Select the view name and text from the USER_VIEWS data dictionary views

QUERY:

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

OUTPUT:

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

```
1 | SELECT view_name, text
2 | FROM user_views;
3 |
```

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

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

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

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

QUERY:

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

OUTPUT:

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

```
1 | SELECT employee, department_id
2 | FROM employees_vu;
3 |
```

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

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

5 rows returned in 0.01 seconds Copyright © 1999, 2023, Oracle and/or its affiliates. Oracle APEX 23.2.4

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

QUERY:

```
CREATE VIEW dept50 AS SELECT employee_id empno, last_name employee, department_id
deptno FROM employees WHERE department_id = 50 WITH CHECK OPTION CONSTRAINT
emp_dept_50;
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there is a search bar, user profile information for 'vijay PS PS india@123', and a schema dropdown set to 'WKSP_ABISHEAK08'. The main workspace is titled 'SQL Commands' and contains the SQL code for creating the DEPT50 view. Below the code, the 'Results' tab is selected, showing the message 'View created.' and a execution time of '0.04 seconds'. The bottom of the screen displays standard Oracle APEX footer information.

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

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

QUERY:

```
Describe dept50;
```

OUTPUT:

APEX App Builder SQL Workshop Team Development Gallery

SQL Commands

Language: SQL Rows: 10 Clear Command Find Tables

1 Describe dept50

Results Explain Describe Saved SQL History

Object Type: VIEW Object: DEPT50

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
DEPT50	EMPNO	NUMBER	22	-	0	-	-	-	-
	EMPLOYEE	VARCHAR2	20	-	1	-	✓	-	-
	DEPTNO	NUMBER	22	-	0	1	✓	-	-

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

7.) Attempt to reassign Matos to department 80

QUERY:

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

OUTPUT:

APEX App Builder SQL Workshop Team Development Gallery

SQL Commands

Language: SQL Rows: 10 Clear Command Find Tables

```
1 UPDATE dept50
2 SET deptno=80
3 WHERE employee='Matos';
4
5
```

Results Explain Describe Saved SQL History

0 rows(s) updated.

0.05 seconds

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

8.) Create a view called SALARY_VU based on the employee last names, department names, salaries, and salary grades for all employees. Use the Employees, DEPARTMENTS and JOB_GRADE tables. Label the column Employee, Department, salary, and Grade respectively.

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. At the top, there are tabs for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there is a search bar, a help icon, and a user profile for 'vijay PS PS India@123'. Below the tabs, it says 'SQL Commands' and shows a schema dropdown set to 'WKSP_ABISHEAK08'. The main area contains a code editor with the following SQL command:

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

Below the code editor, there are buttons for Save and Run. Underneath the editor, there is a results section with tabs for Results, Explain, Describe, Saved SQL, and History. The results tab shows the message 'View created.' and '0.04 seconds'. At the bottom of the page, there is footer information: 'Copyright © 1999-2023, Oracle and/or its affiliates.', 'Oracle APEX 23.2.4', and a URL 'https://220701040@replikshmi.edu.in'.

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

Total (15)	
Faculty Signature	

RESULT:

EXERCISE 12

PRACTICE QUESTIONS

Intro to Constraints; NOT NULL and UNIQUE Constraints

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

Table Created.

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

```
DESCRIBE f_global_locations;
```

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

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

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

PRIMARY KEY, FOREIGN KEY, and CHECK Constraints

1. What is the purpose of a

- PRIMARY KEY
- FOREIGN KEY
- CHECK CONSTRAINT

a. **PRIMARY KEY**

Uniquely identify each row in table.

b. **FOREIGN KEY**

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

c. **CHECK CONSTRAINT**

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

2. Using the column information for the animals table below, name constraints where applicable at the table level, otherwise name them at the column level. Define the primary key (animal_id). The license_tag_number must be unique. The admit_date and vaccination_date columns cannot contain null values.

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

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

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

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

ANIMAL_ID	NAME	LICENSE_TAG_NUMBE R	ADMIT_DATE	ADOPTION_I D	VACCINATION_DAT E
101	Spot	35540	10-Oct-2004	205	12-Oct-2004

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

```
SELECT * FROM animals;
```

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

COLUMN LEVEL STATEMENT:

```
ALTER TABLE animals
```

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

TABLE LEVEL STATEMENT:

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

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

a. ON DELETE CASCADE

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

b. ON DELETE SET NULL

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

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

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

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

PRACTICE PROBLEM

Managing Constraints

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

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

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

- ADD
- DROP
- ENABLE
- DISABLE

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

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

3. Create a FOREIGN KEY constraint in the copy_d_events table. Name the foreign key copy_d_events_fk. This key references the copy_d_clients table client_number column. What is the syntax you used to create the FOREIGN KEY constraint in the copy_d_events table?

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

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

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

a. The constraint name for the primary key in the copy_d_clients table is_____.

COPY_D_CLT_CLIENT_NUMBER_PK

5. Drop the PRIMARY KEY constraint on the copy_d_clients table. Explain your results.

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

6. Add the following event to the copy_d_events table. Explain your results.

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

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

RESULT: ORA-02291: integrity constraint (HKUMAR.COPY_D_EVE_CLIENT_NUMBER_FK) violated - parent key not found

7. Create an ALTER TABLE query to disable the primary key in the copy_d_clients table. Then add the values from #6 to the copy_d_events table. Explain your results.

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

8. Repeat question 6: Insert the new values in the copy_d_events table. Explain your results.

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

1 row(s) inserted.

9. Enable the primary-key constraint in the copy_d_clients table. Explain your results.

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

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

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

1 row(s) deleted.

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

Table altered.

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

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

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

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

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

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

EXERCISE 13

Creating Views

1. What are three uses for a view from a DBA's perspective?
 - **Restrict access and display selective columns**
 - **Reduce complexity of queries from other internal systems. So, providing a way to view same data in a different manner.**
 - **Let the app code rely on views and allow the internal implementation of tables to be modified later.**
2. Create a simple view called view_d_songs that contains the ID, title and artist from the DJs on Demand table for each "New Age" type code. In the subquery, use the alias "Song Title" for the title column.

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

3. SELECT * FROM view_d_songs. What was returned?

Results	Explain	Describe	Saved SQL	History
ID	Song Title	ARTIST		
47	Hurrah for Today	The Jubilant Trio		
49	Lets Celebrate	The Celebrants		
2 rows returned in 0.00 seconds		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

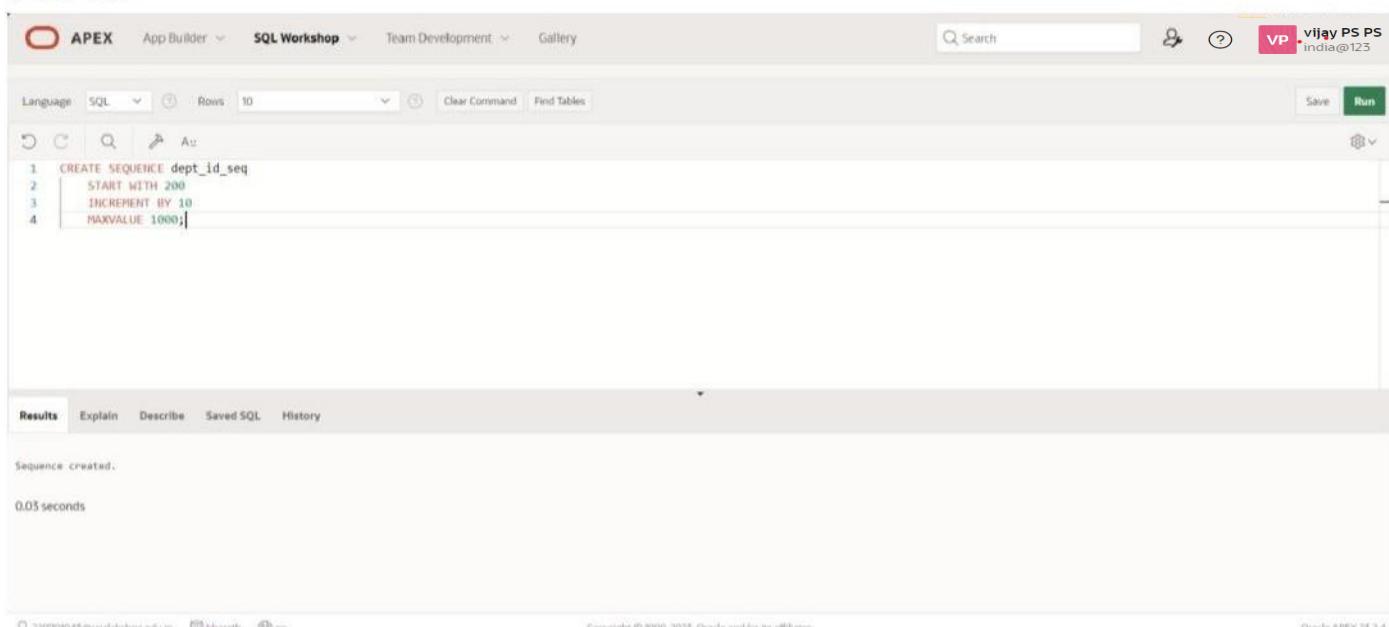
DATE:

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 APEX SQL Workshop interface. In the top navigation bar, 'APEX' and 'SQL Workshop' are selected. The main area contains the following SQL code:

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

Below the code, the results pane shows the output:

```
Sequence created.
```

Execution time: 0.03 seconds.

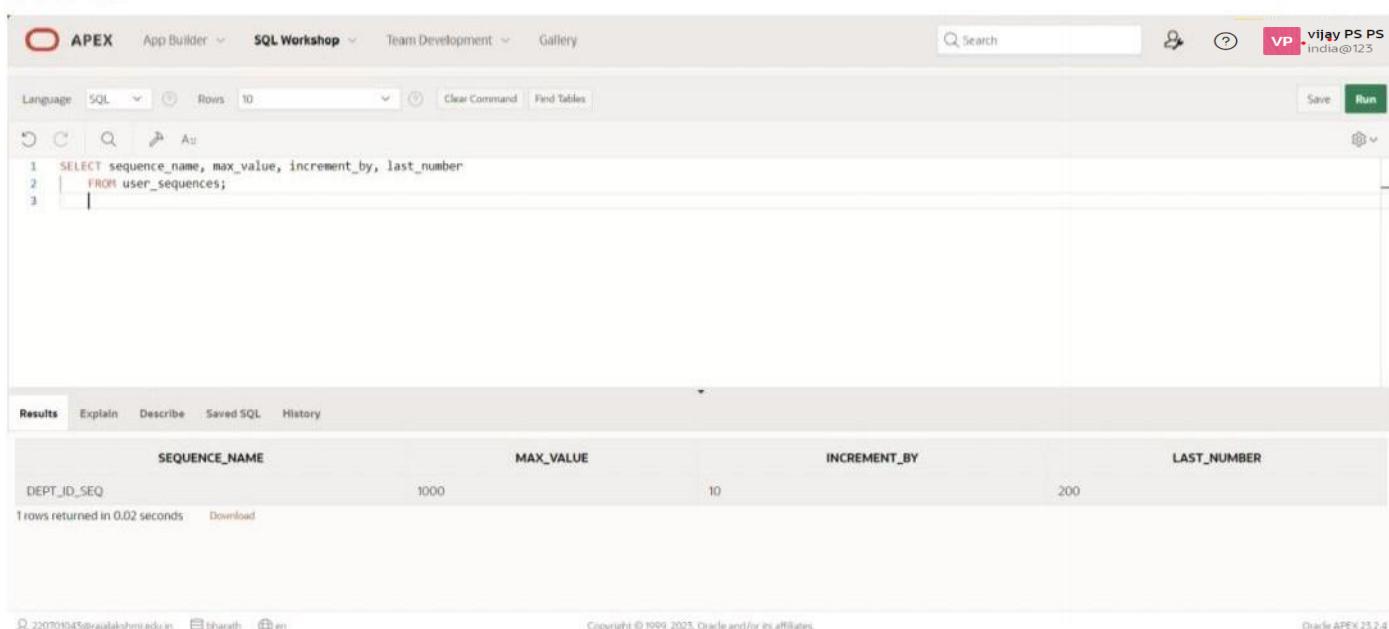
At the bottom, the copyright notice reads "Copyright © 1999, 2023, Oracle and/or its affiliates." and the version "Oracle APEX 25.2.4".

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

QUERY:

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

OUTPUT:



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

```
1 SELECT sequence_name, max_value, increment_by, last_number
2   FROM user_sequences;
3 |
```

Below the code, the results pane shows the output:

SEQUENCE_NAME	MAX_VALUE	INCREMENT_BY	LAST_NUMBER
DEPT_ID_SEQ	1000	10	200

Execution time: 0.02 seconds.

At the bottom, the copyright notice reads "Copyright © 1999, 2023, Oracle and/or its affiliates." and the version "Oracle APEX 25.2.4".

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. On the right, there's a user profile for 'vijay PS PS india@123'. The main area displays the results of a script run titled 'exercise14'. The summary table shows 1 row inserted. Below this, three detailed rows show the execution of the insert statement: one successful (labeled 'Successful') and two with errors (labeled 'With Errors'). The bottom of the screen shows the Oracle APEX copyright notice and version 25.2.4.

Number	Elapsed	Statement	Feedback	Rows
1	0.05	INSERT INTO dept VALUES (dept_id_seq.nextval, 'Education')	1 row(s) inserted.	1
				row(s) 1 - 1 of 1

Statements Processed: 1
Successful: 1
With Errors: 0

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. On the right, there's a user profile for 'vijay PS PS india@123'. The main area shows the execution of an SQL command to create an index. The output panel displays the message 'Index created.' and the execution time '0.05 seconds'. The bottom of the screen shows the Oracle APEX copyright notice and version 25.2.4.

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

Index created.
0.05 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 APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there's a search bar, user profile information for 'vijay PS PS' (india@123), and buttons for Save and Run.

The main workspace displays the SQL command:

```
1 SELECT index_name, table_name, uniqueness
2   FROM user_indexes
3  WHERE table_name = 'EMPLOYEES';
4
```

The results section shows the output of the query:

INDEX_NAME	TABLE_NAME	UNIQUENESS
EMP_DEPT_ID_IDX	EMPLOYEES	NONUNIQUE

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

The bottom of the page includes footer links for 220701045@rajalakshmi.edu.in, bharath, and sri, along with copyright information: Copyright © 1999, 2023, Oracle and/or its affiliates. and Oracle APEX 25.2.4.

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:

DATE:

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. The right side shows a user profile for 'vijay PS PS' with the email 'india@123'. The main workspace has tabs for 'Language' (set to 'SQL'), 'Rows' (set to 10), 'Clear Command', and 'Find Tables'. Below these are icons for 'Copy', 'Run', 'Save', and 'Run'. The code area contains a numbered PL/SQL block:

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

The results tab shows the output of the query:

```
Incentive = 8400
Statement processed.
0.00 seconds
```

At the bottom, the footer includes user information (220701043@rajalakshmi.edu.in, bharath, en), copyright notice (Copyright © 1999-2023, Oracle and/or its affiliates.), and system details (19 May 2024, Oracle APEX 25.2.4).

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. On the right, there's a user profile for 'vijay PS PS' with the email 'india@123'. The main area has tabs for 'Language' (set to 'SQL'), 'Rows' (set to 10), 'Clear Command', and 'Find Tables'. Below these are standard edit tools like Undo, Redo, Find, Replace, and Paste. The code editor contains the following PL/SQL block:

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

An error message is displayed in a yellow box:

```
Error at line 4/25: ORA-06550: line 4, column 25:
PLS-00201: identifier 'Welcome' must be declared
ORA-06512: at 'SYS.IMPV_DBMS_SQL_APEX_230200', line 801
ORA-06550: line 4, column 3:
PL/SQL: Statement ignored
```

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

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

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

This screenshot shows the same Oracle APEX SQL Workshop interface. The top navigation bar and user profile are identical. The code editor contains the same PL/SQL block as the previous screenshot:

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

An error message is displayed in a yellow box:

```
Error at line 4/25: ORA-06550: line 4, column 25:
PLS-00201: identifier 'Welcome' must be declared
ORA-06512: at 'SYS.IMPV_DBMS_SQL_APEX_230200', line 801
ORA-06550: line 4, column 3:
PL/SQL: Statement ignored
```

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

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

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

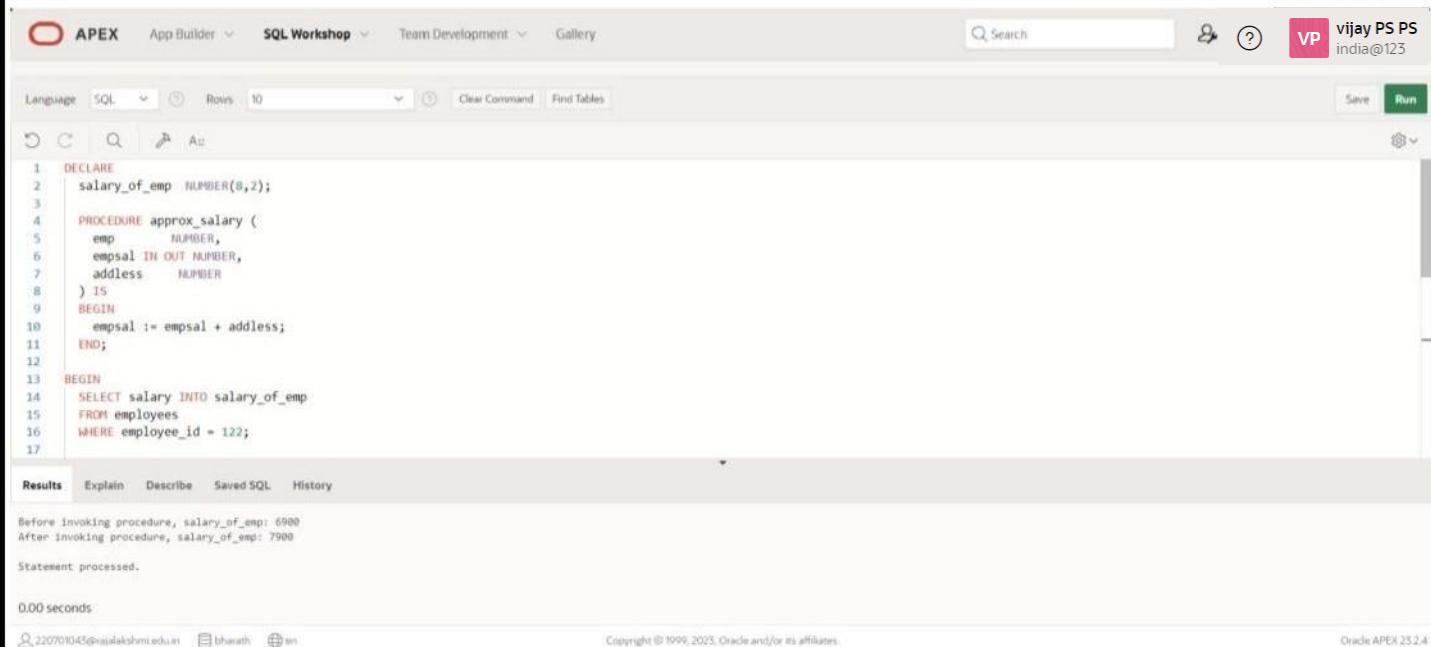
3.) Write a PL/SQL block to adjust the salary of the employee whose ID 122.

QUERY:

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

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

OUTPUT:



The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. On the right, there's a user profile for 'vijay PS PS' with the email 'india@123'. The main workspace displays the PL/SQL code from the 'QUERY:' section. Below the code, the 'Results' tab is selected, showing the output of the executed block. The output includes the initial value of salary_of_emp (6900), the result after invoking the procedure (7900), and a 'Statement processed.' message. At the bottom, it shows the execution time (0.00 seconds) and copyright information for Oracle APEX 25.2.4.

```
1  DECLARE
2      salary_of_emp  NUMBER(8,2);
3
4  PROCEDURE approx_salary (
5      emp      NUMBER,
6      empsal IN OUT NUMBER,
7      addless  NUMBER
8  ) IS
9  BEGIN
10     empsal := empsal + addless;
11
12
13 BEGIN
14     SELECT salary INTO salary_of_emp
15     FROM employees
16     WHERE employee_id = 122;
17
Results Explain Describe Saved SQL History
```

Before invoking procedure, salary_of_emp: 6900
After invoking procedure, salary_of_emp: 7900
Statement processed.
0.00 seconds

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder...', 'SQL Workshop' (selected), 'Team Development...', 'Gallery', 'Search', and a user profile 'vijay PS PS india@123'. The main workspace displays the PL/SQL code for the 'pri_bool' procedure. The code is as follows:

```
1 CREATE OR REPLACE PROCEDURE pri_bool(
2   boo_name VARCHAR2,
3   boo_val BOOLEAN
4 ) IS
5 BEGIN
6   IF boo_val IS NULL THEN
7     DBMS_OUTPUT.PUT_LINE( boo_name || '=' || 'NULL');
8   ELSIF boo_val = TRUE THEN
9     DBMS_OUTPUT.PUT_LINE( boo_name || '=' || 'TRUE');
10  ELSE
11    DBMS_OUTPUT.PUT_LINE( boo_name || '=' || 'FALSE');
12  END IF;
13 END;
14 /
```

Below the code, there are tabs for 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History'. The 'Results' tab is selected, showing the message 'Procedure created.' and a timestamp '0.03 seconds'. The bottom of the screen includes copyright information: 'Copyright © 1999, 2025, Oracle and/or its affiliates.' and 'Oracle APEX 23.2.4'.

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there's a user profile for 'vijay PS' and the email 'india@123'. The main workspace displays the PL/SQL code from the 'QUERY:' section. Below the code, the 'Results' tab is selected, showing the output of the procedure execution. The output consists of two rows: 'TRUE' and 'FALSE', corresponding to the two calls to the 'pat_match' procedure. The bottom status bar indicates the statement was processed in 0.00 seconds.

```
1  DECLARE
2  2    PROCEDURE pat_match (
3  3      test_string  VARCHAR2,
4  4      pattern      VARCHAR2
5  5    ) IS
6  BEGIN
7  7      IF test_string LIKE pattern THEN
8  8          DBMS_OUTPUT.PUT_LINE ('TRUE');
9  9      ELSE
10 10          DBMS_OUTPUT.PUT_LINE ('FALSE');
11 11      END IF;
12 12  END;
13 13  BEGIN
14 14      pat_match('Blweate', 'B%a_e');
15 15      pat_match('Blweate', 'B%A_E');
16 16  END;
17 17  /
```

Results Explain Describe Saved SQL History

TRUE
FALSE

Statement processed.

0.00 seconds

220701045@rajalakshmi.edu.in bharathi en Copyright © 1999-2023, Oracle and/or its affiliates. Oracle APEX 23.2.4

- 6.) Write a PL/SQL program to arrange the number of two variable in such a way that the small number will store in num_small variable and large number will store in num_large variable

QUERY:

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

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

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

OUTPUT:

The screenshot shows the Oracle APEX interface with the SQL Workshop tab selected. In the SQL editor, the following PL/SQL block is written:

```
2 num_small NUMBER := 8;
3 num_large NUMBER := 5;
4 num_temp NUMBER;
5 BEGIN
6
7 IF num_small > num_large THEN
8 num_temp := num_small;
9 num_small := num_large;
10 num_large := num_temp;
11 END IF;
12
13 DBMS_OUTPUT.PUT_LINE ('num_small = '||num_small);
14 DBMS_OUTPUT.PUT_LINE ('num_large = '||num_large);
15 END;
16 /
17
```

In the Results tab, the output of the block is shown:

```
num_small = 5
num_large = 8

Statement processed.
```

At the bottom, it shows "0.00 seconds" and the copyright notice "Copyright © 1999, 2023, Oracle and/or its affiliates".

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

QUERY:

```
DECLARE
  PROCEDURE test1 (
    sal_achieve NUMBER,
    target_qty NUMBER,
    emp_id NUMBER
  )
  IS
    incentive NUMBER := 0;
    updated VARCHAR2(3) := 'No';
  BEGIN
    IF sal_achieve > (target_qty + 200) THEN
      incentive := (sal_achieve - target_qty)/4;
      UPDATE employees
      SET salary = salary + incentive
      WHERE employee_id = emp_id;
      updated := 'Yes';
    END IF;
    DBMS_OUTPUT.PUT_LINE (
      'Table updated? ' || updated || ',' ||
      'incentive = ' || incentive || '
    );
  END test1;
BEGIN
  test1(2300, 2000, 144);
  test1(3600, 3000, 145);
END;
/
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. On the right, there are user profile and run buttons. The main workspace displays the PL/SQL code for the 'test1' procedure. The code uses DBMS_OUTPUT.PUT_LINE to print the status of the update and the calculated incentive. The results pane at the bottom shows the output: 'Table updated? Yes, incentive = 75.' and 'Table updated? Yes, incentive = 150.'. It also indicates 1 row(s) updated and a execution time of 0.02 seconds.

```
1  DECLARE
2  2    PROCEDURE test1 (
3  3      sal_achieve NUMBER,
4  4      target_qty NUMBER,
5  5      emp_id NUMBER
6  6    )
7  7    IS
8  8      incentive NUMBER := 0;
9  9      updated VARCHAR2(3) := 'No';
10 10    BEGIN
11 11      IF sal_achieve > (target_qty + 200) THEN
12 12        incentive := (sal_achieve - target_qty)/4;
13 13        UPDATE employees
14 14        SET salary = salary + incentive
15 15        WHERE employee_id = emp_id;
16 16    END IF;
17 17    DBMS_OUTPUT.PUT_LINE (
18 18      'Table updated? ' || updated || ',' ||
19 19      'incentive = ' || incentive || '
20 20    );
21 21  END test1;
22 22 BEGIN
23 23   test1(2300, 2000, 144);
24 24   test1(3600, 3000, 145);
25 25 END;
26 26 /
```

Results Explain Describe Saved SQL History

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

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

QUERY:

```

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

```

The screenshot shows the Oracle APEX interface with the SQL Workshop tab selected. The code area contains the PL/SQL procedure defined above. The results pane shows the output of the procedure's execution for each salary input. The bottom of the results pane displays a message indicating the statement was processed.

```

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

```

Results

```

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

```

0.00 seconds

220701045@cajalakshmi.edu.in bharath eri Copyright © 1999, 2023, Oracle and/or its affiliates. Oracle APEX 23.2.4

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

QUERY:

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

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

OUTPUT:

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

```
1 SET SERVEROUTPUT ON
2 DECLARE
3     tot_emp NUMBER;
4 BEGIN
5     SELECT Count(*)
6         INTO tot_emp
7     FROM employees e
8         join departments d
9             ON e.department_id = d.department_id
10    WHERE e.department_id = 50;
11
12    dbms_output.Put_line ('The employees are in the department 50: '
13                           ||To_char(tot_emp));
14
15    IF tot_emp >= 45 THEN
```

The results pane shows the output of the query:

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

0.00 seconds

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

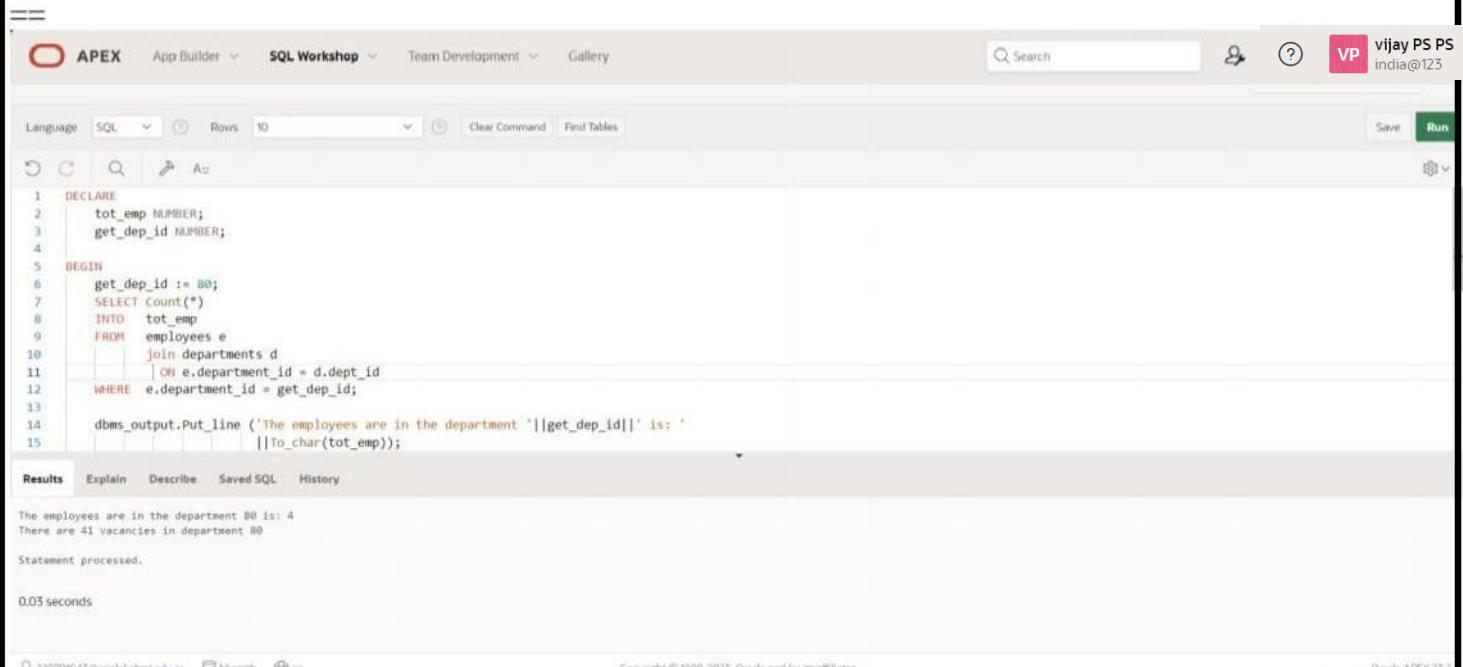
QUERY:

```
DECLARE
    tot_emp NUMBER;
    get_dep_id NUMBER;

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

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

    IF tot_emp >= 45 THEN
        dbms_output.Put_line ('There are no vacancies in the department'||get_dep_id);
    ELSE
        dbms_output.Put_line ('There are '||to_char(45-tot_emp)|||' vacancies in department'|||
get_dep_id );
    END IF;
END;
/
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. On the right, there's a user profile for 'vijay PS PS india@123'. The main workspace has tabs for Language (set to SQL), Rows (set to 10), and Clear Command. Below these are standard database navigation icons (refresh, search, etc.). The SQL command area contains the provided PL/SQL code. The results tab shows the output of the execution:

```
1 DECLARE
2     tot_emp NUMBER;
3     get_dep_id NUMBER;
4
5 BEGIN
6     get_dep_id := 80;
7     SELECT Count(*)
8     INTO tot_emp
9     FROM employees e
10    join departments d
11       | ON e.department_id = d.dept_id
12      WHERE e.department_id = get_dep_id;
13
14    dbms_output.Put_line ('The employees are in the department'||get_dep_id||' is: '
15                           ||To_char(tot_emp));
```

The output pane displays the results of the query:

```
The employees are in the department 80 is: 4
There are 41 vacancies in department 80

Statement processed.

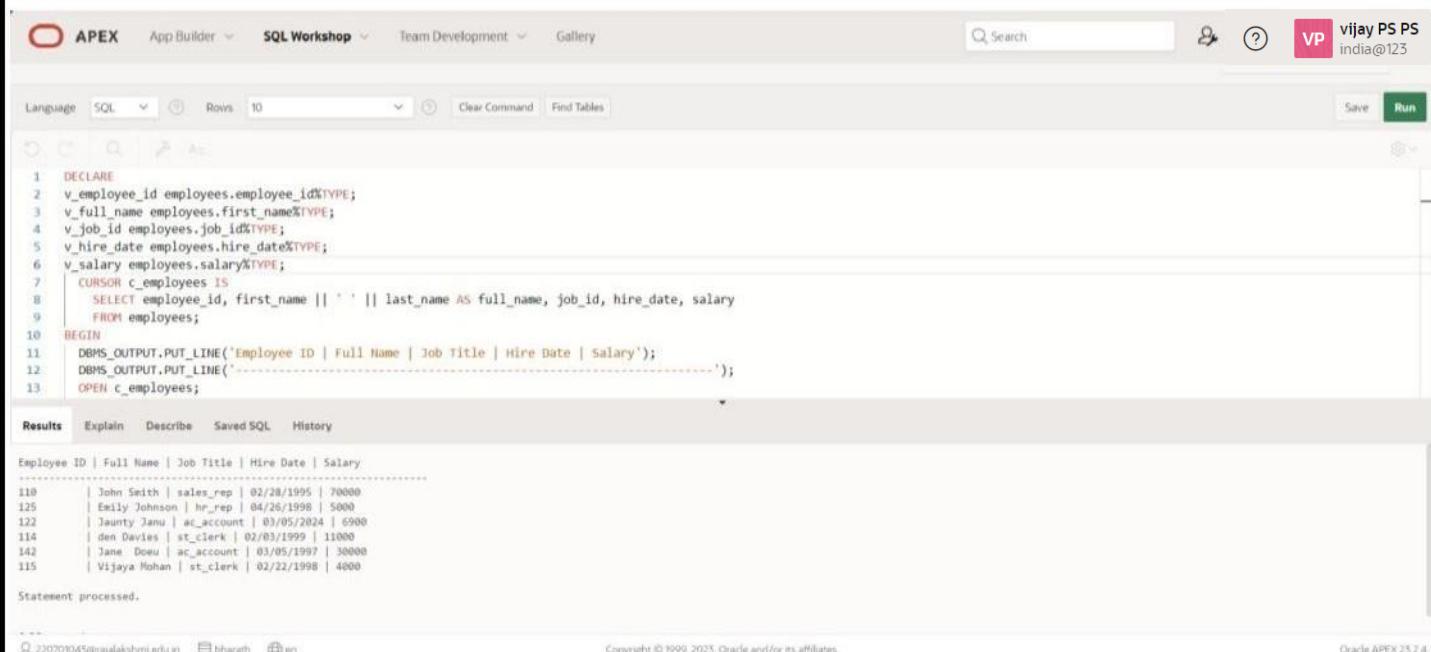
0.03 seconds
```

At the bottom, footer information includes the session ID (220701043@rajalakshmi.edu.in), the user (bharath), the language (en), copyright notice (Copyright © 1999, 2025, Oracle and/or its affiliates.), and the APEX version (Oracle APEX 23.2).

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

QUERY:

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

OUTPUT:

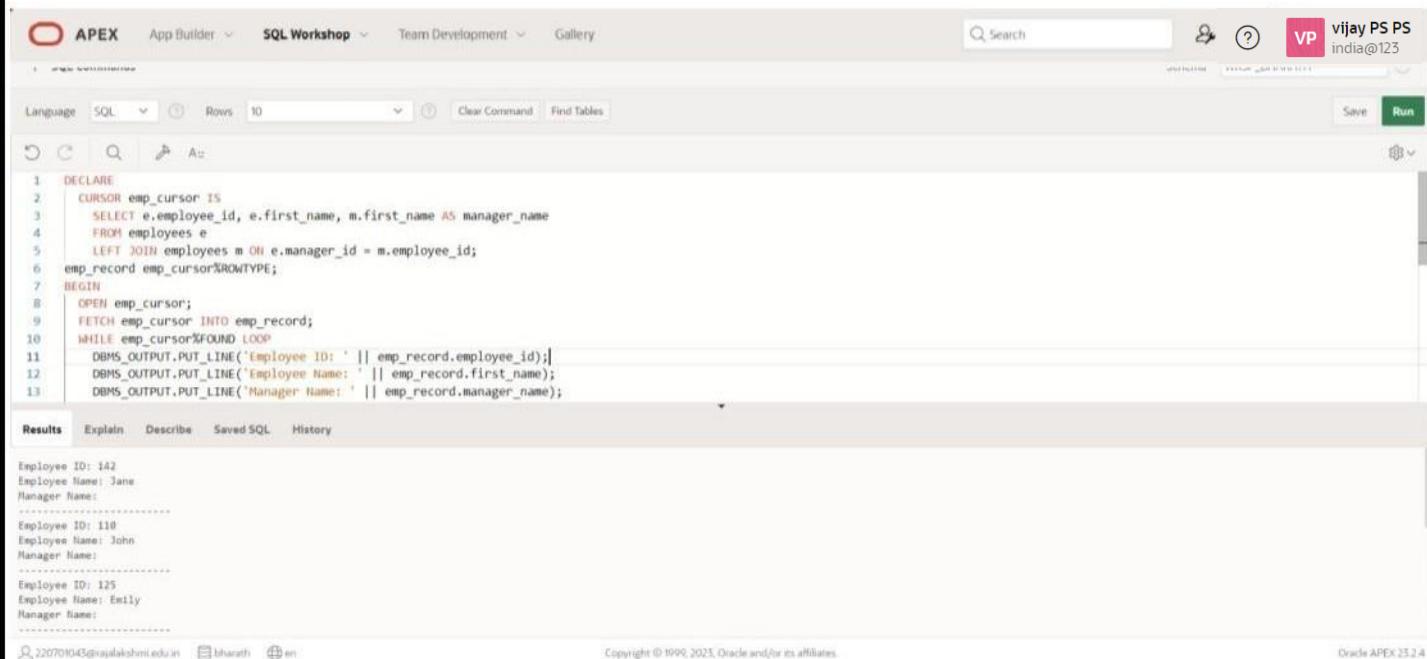
```
Employee ID | Full Name | Job Title | Hire Date | Salary
-----|-----|-----|-----|-----
110 | John Smith | sales_rep | 02/28/1995 | 70000
125 | Emily Johnson | hr_rep | 04/26/1998 | 5000
122 | Jaunty Janu | ac_account | 03/05/2024 | 6500
114 | den Davies | st_clerk | 02/03/1999 | 11000
142 | Jane Doe | ac_account | 03/05/1997 | 30000
115 | Vijaya Mohan | st_clerk | 02/22/1998 | 4000
```

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

QUERY:

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

OUTPUT:



```
Employee ID: 142
Employee Name: Jane
Manager Name:
-----
Employee ID: 118
Employee Name: John
Manager Name:
-----
Employee ID: 125
Employee Name: Emily
Manager Name:
```

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

QUERY:

```

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

```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. In the code editor, a PL/SQL block is written to query the job_grade and employees tables to find the lowest salary for each job. The output window displays the results for four jobs: sales_rep, hr_rep, ac_account, and st_clerk, all with a minimum salary of 2500.

```

1  DECLARE
2    CURSOR job_cursor IS
3      SELECT e.job_id, j.lowest_sal
4        FROM job_grade j,employees e;
5    job_record job_cursor%ROWTYPE;
6    BEGIN
7      OPEN job_cursor;
8      FETCH job_cursor INTO job_record;
9      WHILE job_cursor%FOUND LOOP
10        DBMS_OUTPUT.PUT_LINE('Job ID: ' || job_record.job_id);
11        DBMS_OUTPUT.PUT_LINE('Minimum Salary: ' || job_record.lowest_sal);
12        DBMS_OUTPUT.PUT_LINE('-----');
13        FETCH job_cursor INTO job_record;

```

Job ID	Minimum Salary
sales_rep	2500
hr_rep	2500
ac_account	2500
st_clerk	2500

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

QUERY:

```

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

```

OUTPUT:

The screenshot shows the Oracle APEX interface with the SQL Workshop tab selected. The code area contains the PL/SQL block provided above. The results tab displays the output of the block, which includes two rows of employee data:

Employee ID	Last Name	Job Id	Start Date
125	Johnson	hr_rep	22-apr-1999
125	Johnson	hr_rep	22-apr-1999

Statement processed.

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

QUERY:

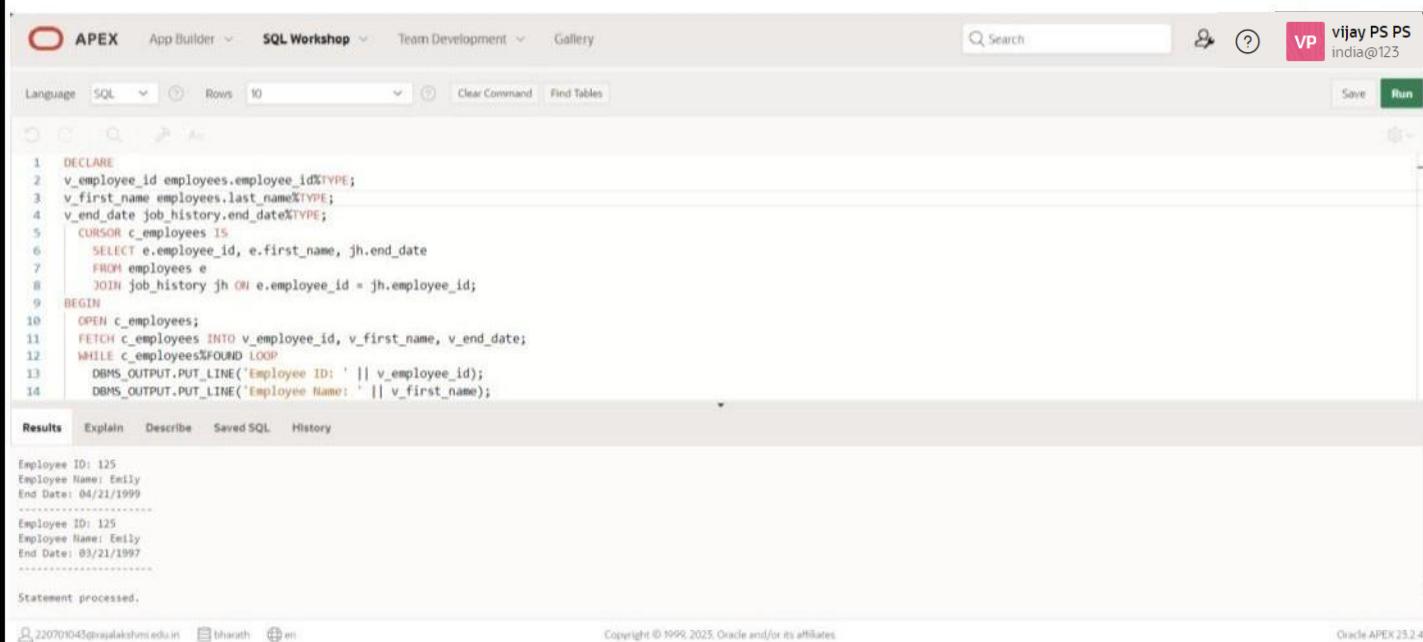
```
DECLARE
```

```

v_employee_id employees.employee_id%TYPE;
v_first_name employees.last_name%TYPE;
v_end_date job_history.end_date%TYPE;
CURSOR c_employees IS
  SELECT e.employee_id, e.first_name, jh.end_date
  FROM employees e
  JOIN job_history jh ON e.employee_id = jh.employee_id;
BEGIN
  OPEN c_employees;
  FETCH c_employees INTO v_employee_id, v_first_name, v_end_date;
  WHILE c_employees%FOUND LOOP
    DBMS_OUTPUT.PUT_LINE('Employee ID: ' || v_employee_id);
    DBMS_OUTPUT.PUT_LINE('Employee Name: ' || v_first_name);
    DBMS_OUTPUT.PUT_LINE('End Date: ' || v_end_date);
    DBMS_OUTPUT.PUT_LINE('-----');
    FETCH c_employees INTO v_employee_id, v_first_name, v_end_date;
  END LOOP;
  CLOSE c_employees;
END;

```

OUTPUT:



The screenshot shows the Oracle APEX interface with the SQL Workshop tab selected. The code area contains the PL/SQL block provided above. The results pane displays the output of the DBMS_OUTPUT.PUT_LINE statements for two employees: one with ID 125 and another with ID 126.

```

1  DECLARE
2    v_employee_id employees.employee_id%TYPE;
3    v_first_name employees.last_name%TYPE;
4    v_end_date job_history.end_date%TYPE;
5    CURSOR c_employees IS
6      SELECT e.employee_id, e.first_name, jh.end_date
7      FROM employees e
8      JOIN job_history jh ON e.employee_id = jh.employee_id;
9    BEGIN
10      OPEN c_employees;
11      FETCH c_employees INTO v_employee_id, v_first_name, v_end_date;
12      WHILE c_employees%FOUND LOOP
13        DBMS_OUTPUT.PUT_LINE('Employee ID: ' || v_employee_id);
14        DBMS_OUTPUT.PUT_LINE('Employee Name: ' || v_first_name);

```

Results:

```

Employee ID: 125
Employee Name: Emily
End Date: 04/21/1999
-----
Employee ID: 126
Employee Name: Emilly
End Date: 03/21/1997
-----

Statement processed.

```

At the bottom left, the URL is 220701043@rajalakshmi.edu.in and the session ID is bhuath. At the bottom right, the copyright notice is Copyright © 1999-2025, Oracle and/or its affiliates, and the version is Oracle APEX 23.2.4.

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

RESULT:

PROCEDURES AND FUNCTIONS

EX_NO: 17

DATE:

1.) Factorial of a number using function.

QUERY:

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

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. The right side shows a user profile for 'vijay PS PS india@123'. The main area is titled 'SQL Commands' with a schema dropdown set to 'WKSP_ABISHEAK08'. The code editor contains the provided PL/SQL block. The results tab shows the output: '479001600' and 'Statement processed.' Below that, it says '0.00 seconds'.

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

Results Explain Describe Saved SQL History

479001600
Statement processed.
0.00 seconds

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

QUERY:

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

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

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

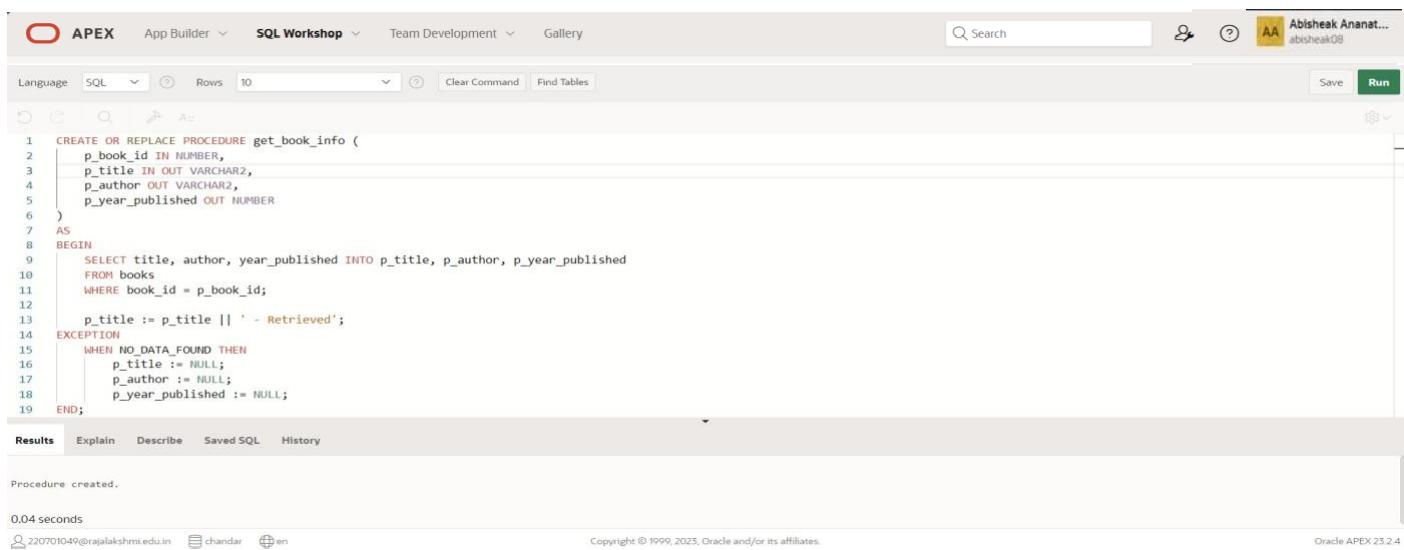
```

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

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

```

OUTPUT:



The screenshot shows the Oracle APEX interface with the SQL Workshop tab selected. A user named Abishek Ananat... is logged in. The code area contains the PL/SQL procedure definition. The results tab shows the message "Procedure created." and the execution time "0.04 seconds".

```

1 CREATE OR REPLACE PROCEDURE get_book_info (
2     p_book_id IN NUMBER,
3     p_title IN OUT VARCHAR2,
4     p_author OUT VARCHAR2,
5     p_year_published OUT NUMBER
6 )
7 AS
8 BEGIN
9     SELECT title, author, year_published INTO p_title, p_author, p_year_published
10    FROM books
11   WHERE book_id = p_book_id;
12
13   p_title := p_title || ' - Retrieved';
14 EXCEPTION
15   WHEN NO_DATA_FOUND THEN
16       p_title := NULL;
17       p_author := NULL;
18       p_year_published := NULL;
19 END;

```

Results Explain Describe Saved SQL History

Procedure created.

0.04 seconds

Copyright © 1999, 2025, Oracle and/or its affiliates. Oracle APEX 23.2.4

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

RESULT:

TRIGGER

EX_NO: 18

DATE:

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

QUERY:

```
CREATE OR REPLACE TRIGGER prevent_parent_deletion
```

```
BEFORE DELETE ON parent_table
```

```
FOR EACH ROW
```

```
DECLARE
```

```
    child_exists EXCEPTION;
```

```
    PRAGMA EXCEPTION_INIT(child_exists, -20001);
```

```
    v_child_count NUMBER;
```

```
BEGIN
```

```
    SELECT COUNT(*) INTO v_child_count FROM child_table WHERE parent_id =  
    :OLD.parent_id;
```

```
    IF v_child_count > 0 THEN
```

```
        RAISE child_exists;
```

```
    END IF;
```

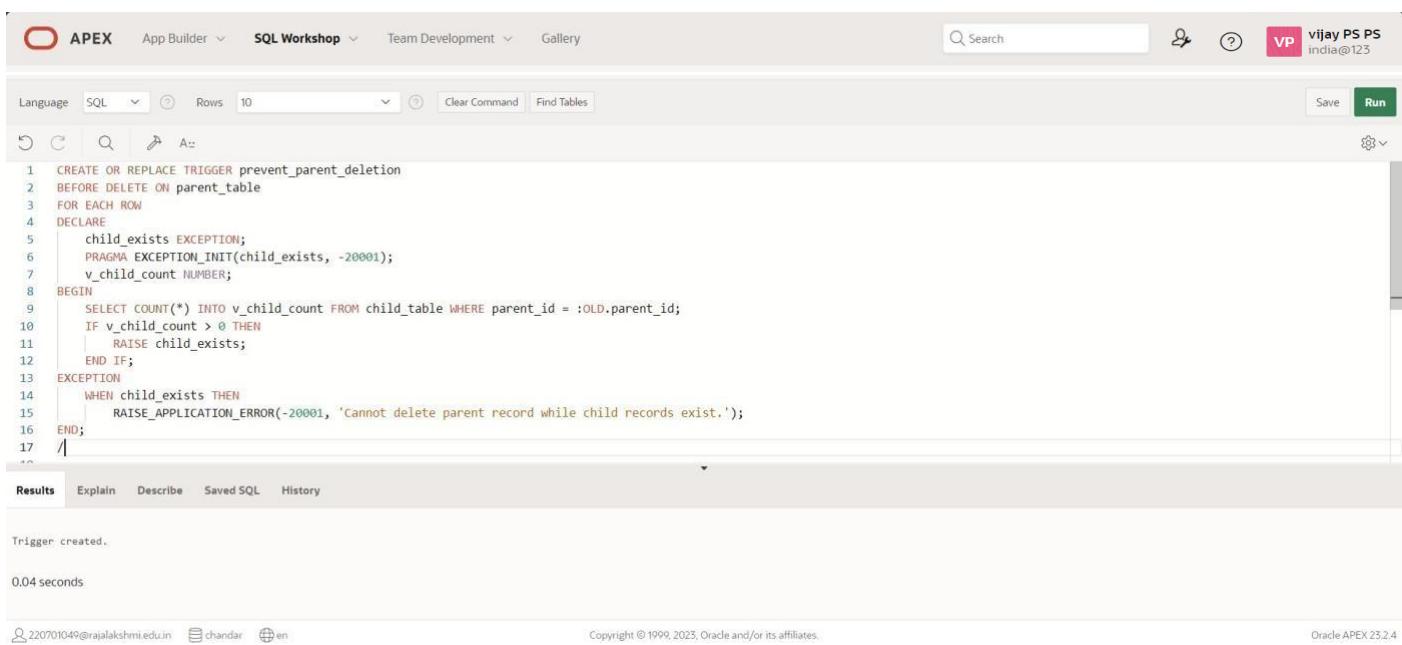
```
EXCEPTION
```

```
    WHEN child_exists THEN
```

```
        RAISE_APPLICATION_ERROR(-20001, 'Cannot delete parent record while child records  
        exist.');
```

```
END;
```

OUTPUT:



The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The top right corner shows a user profile for 'vijay PS' with the email 'india@123'. The main workspace has a toolbar with various icons. The SQL editor area contains the PL/SQL code for the trigger. The bottom section shows the results of the trigger creation, indicating it was successful.

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

Results Explain Describe Saved SQL History

Trigger created.

0.04 seconds

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

Oracle APEX 23.2.4

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

QUERY:

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

OUTPUT:

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

```

1 CREATE OR REPLACE TRIGGER check_duplicates
2 BEFORE INSERT OR UPDATE ON unique_values_table
3 FOR EACH ROW
4 DECLARE
5     duplicate_found EXCEPTION;
6     PRAGMA EXCEPTION_INIT(duplicate_found, -20002);
7     v_count NUMBER;
8 BEGIN
9     SELECT COUNT(*) INTO v_count FROM unique_values_table
10    WHERE unique_col = :NEW.unique_col AND id != :NEW.id;
11    IF v_count > 0 THEN
12        RAISE duplicate_found;
13    END IF;
14 EXCEPTION
15    WHEN duplicate_found THEN
16        RAISE_APPLICATION_ERROR(-20002, 'Duplicate value found in unique_col.');
17 END;

```

Below the code, the 'Results' tab is selected, showing the message 'Trigger created.' and a execution time of '0.04 seconds'. The bottom right corner indicates the version 'Oracle APEX 23.2.4'.

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

QUERY:

```

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

```

EXCEPTION

```
WHEN threshold_exceeded THEN
    RAISE_APPLICATION_ERROR(-20003, 'Threshold exceeded for value_col.');
END;
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. On the right, there's a search bar, a user icon, and a message from 'vijay PS'.

The main area displays the SQL code for creating a trigger:

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

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

```
Trigger created.
```

Execution time: 0.05 seconds

User information at the bottom: 220701049@rajalakshmi.edu.in, chandar, en

Copyright © 1999, 2025, Oracle and/or its affiliates. Oracle APEX 23.2.4

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

QUERY:

```
CREATE OR REPLACE TRIGGER log_changes
```

AFTER UPDATE ON main_table

FOR EACH ROW

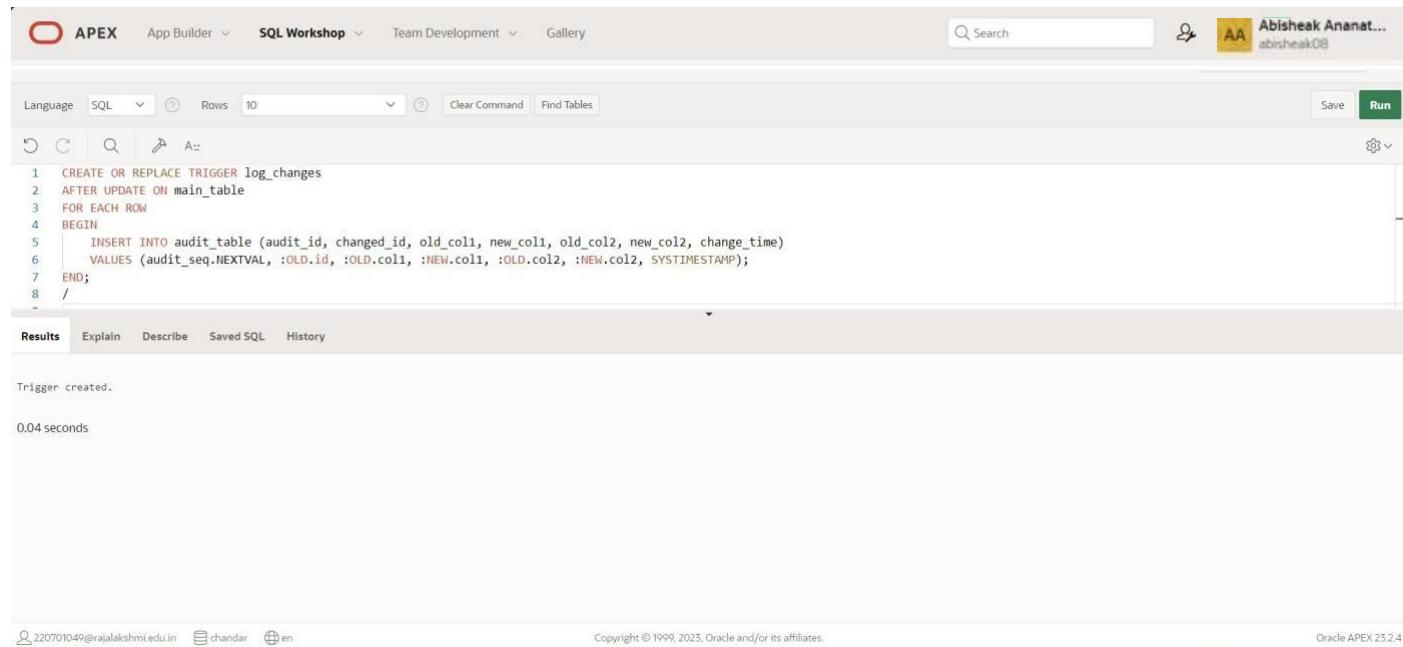
BEGIN

```
    INSERT INTO audit_table (audit_id, changed_id, old_col1, new_col1, old_col2, new_col2,
change_time)
```

```
        VALUES (audit_seq.NEXTVAL, :OLD.id, :OLD.col1, :NEW.col1, :OLD.col2, :NEW.col2,
SYSTIMESTAMP);
```

END;

OUTPUT:



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

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

The code is highlighted in red and blue, indicating syntax errors. Below the code, the 'Results' tab is selected, showing the output: "Trigger created." and "0.04 seconds". The bottom of the screen shows user information and copyright details.

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. In the top navigation bar, 'APEX' is selected. Below it, tabs for 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery' are visible. On the right, a user profile for 'vijay PS PS' with the email 'india@123' is shown. The main workspace is titled 'SQL' and has settings for 'Rows: 10'. The toolbar includes icons for refresh, search, and run. The code area contains the following PL/SQL trigger definition:

```

1 CREATE OR REPLACE TRIGGER log_user_activity
2 AFTER INSERT OR UPDATE OR DELETE ON activity_table
3 FOR EACH ROW
4 BEGIN
5   IF INSERTING THEN
6     INSERT INTO user_activity_log (log_id, action, table_name, record_id, change_time)
7     VALUES (activity_log_seq.NEXTVAL, 'INSERT', 'activity_table', :NEW.id, SYSTIMESTAMP);
8   ELSIF UPDATING THEN
9     INSERT INTO user_activity_log (log_id, action, table_name, record_id, change_time)
10    VALUES (activity_log_seq.NEXTVAL, 'UPDATE', 'activity_table', :NEW.id, SYSTIMESTAMP);
11   ELSIF DELETING THEN
12     INSERT INTO user_activity_log (log_id, action, table_name, record_id, change_time)
13    VALUES (activity_log_seq.NEXTVAL, 'DELETE', 'activity_table', :OLD.id, SYSTIMESTAMP);
14   END IF;
15 END;
16 /

```

Below the code, there are tabs for 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History'. The 'Results' tab is selected, showing the message 'Trigger created.' and a execution time of '0.05 seconds'. At the bottom, the URL '220701049@rajalakshmi.edu.in', the session ID 'chandar', and the language 'en' are displayed, along with copyright information and the version 'Oracle APEX 23.2.4'.

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

QUERY:

```

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

```

END;

OUTPUT:

The screenshot shows the Oracle APEX interface with the SQL Workshop tab selected. The code area contains the following SQL code:

```
1 CREATE OR REPLACE TRIGGER update_running_total
2 BEFORE INSERT ON running_total_table
3 FOR EACH ROW
4 DECLARE
5     v_total NUMBER;
6 BEGIN
7     SELECT NVL(SUM(amount), 0) INTO v_total FROM running_total_table;
8     :NEW.running_total := v_total + :NEW.amount;
9 END;
10 /
```

The results section shows the message "Trigger created." and a execution time of "0.04 seconds".

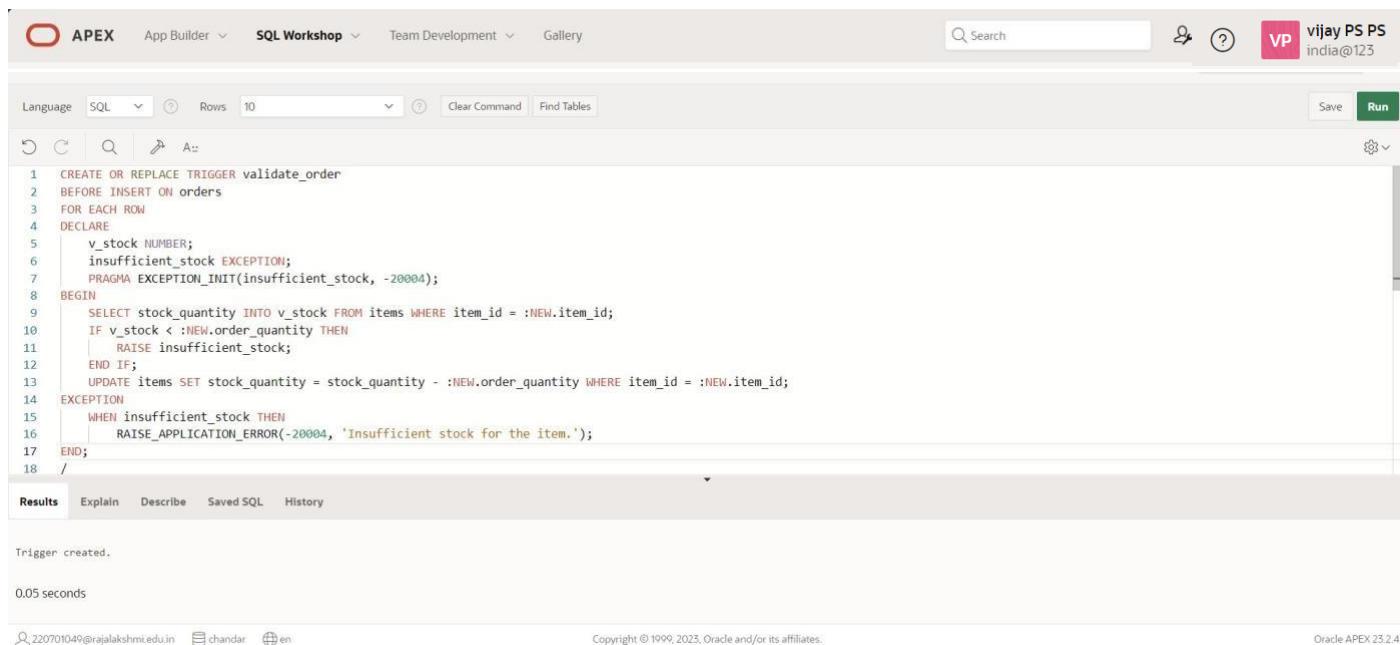
At the bottom, the footer includes user information (220701049@rajalakshmi.edu.in, chandar, en), copyright notice (Copyright © 1999,2023, Oracle and/or its affiliates.), and the version (Oracle APEX 23.2.4).

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

QUERY:

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

OUTPUT:



The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there's a user profile for 'vijay PS' and a search bar. The main workspace is titled 'SQL' and shows the trigger creation code. Below the code, the 'Results' tab is selected, displaying the message 'Trigger created.' and a execution time of '0.05 seconds'. The bottom footer includes copyright information for Oracle and the APEX version.

```
1 CREATE OR REPLACE TRIGGER validate_order
2 BEFORE INSERT ON orders
3 FOR EACH ROW
4 DECLARE
5     v_stock NUMBER;
6     insufficient_stock EXCEPTION;
7     PRAGMA EXCEPTION_INIT(insufficient_stock, -20004);
8 BEGIN
9     SELECT stock_quantity INTO v_stock FROM items WHERE item_id = :NEW.item_id;
10    IF v_stock < :NEW.order_quantity THEN
11        RAISE insufficient_stock;
12    END IF;
13    UPDATE items SET stock_quantity = stock_quantity - :NEW.order_quantity WHERE item_id = :NEW.item_id;
14 EXCEPTION
15    WHEN insufficient_stock THEN
16        RAISE_APPLICATION_ERROR(-20004, 'Insufficient stock for the item.');
17 END;
18 /
```

Trigger created.
0.05 seconds

220701049@rajalakshmi.edu.in chandar en Copyright © 1999, 2023, Oracle and/or its affiliates. Oracle APEX 23.2.4

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

RESULT:

TRIGGER

EX_NO: 18

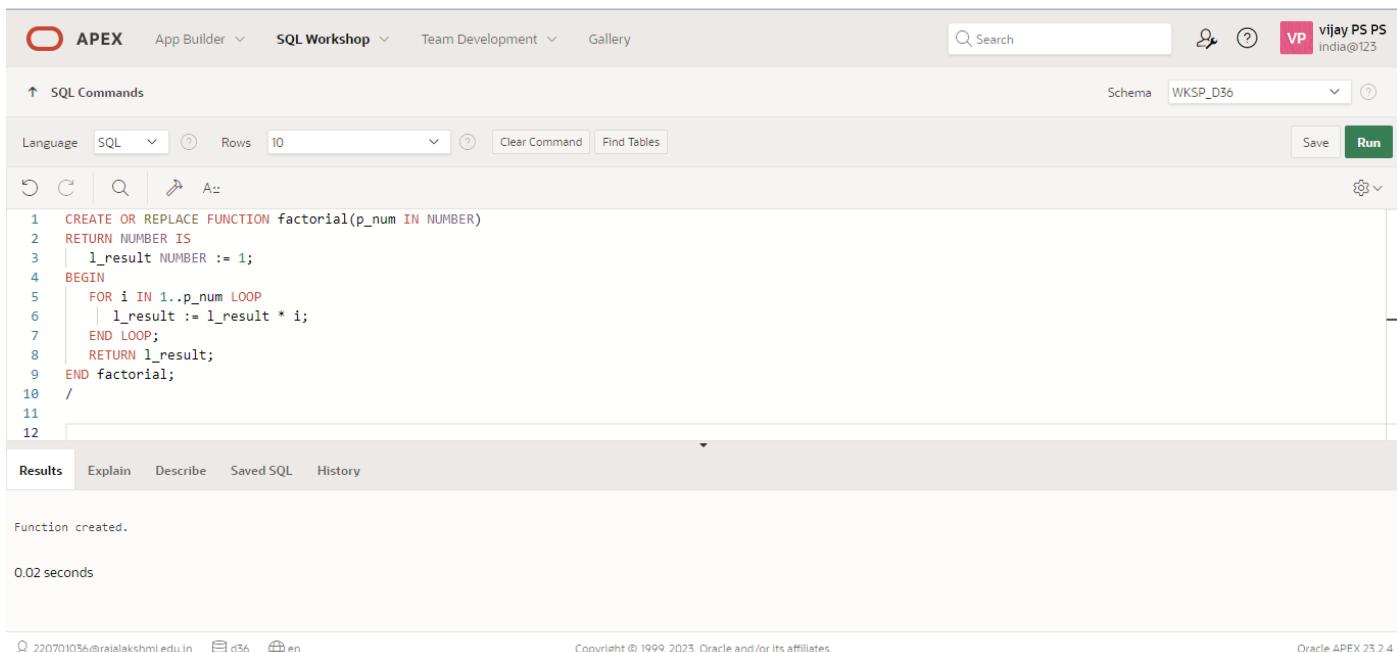
DATE:

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

QUERY:

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

OUTPUT:



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

```
1 CREATE OR REPLACE FUNCTION factorial(p_num IN NUMBER)
2 RETURN NUMBER IS
3     l_result NUMBER := 1;
4 BEGIN
5     FOR i IN 1..p_num LOOP
6         l_result := l_result * i;
7     END LOOP;
8     RETURN l_result;
9 END factorial;
10 /
11
12
```

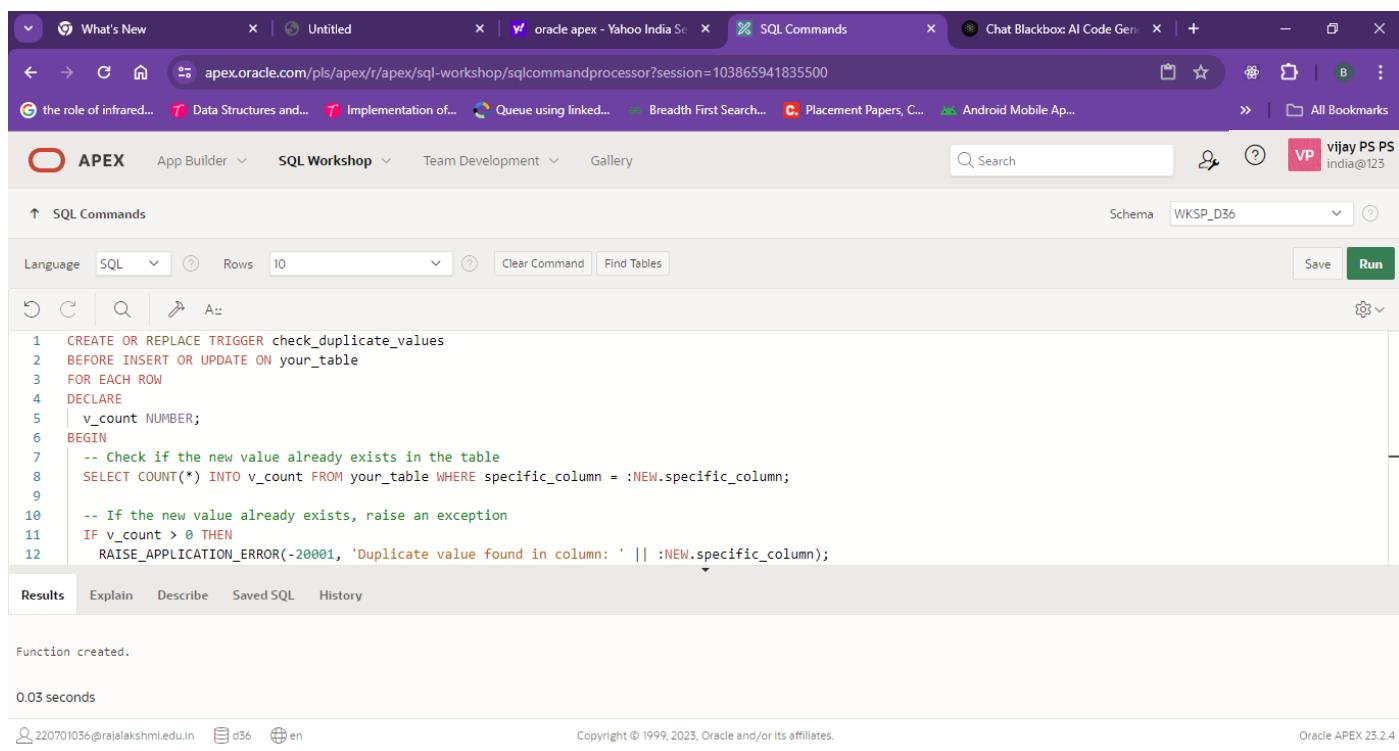
Below the code, the 'Results' tab is active, showing the message "Function created." and "0.02 seconds". At the bottom of the page, the footer includes the URL "220701036@rajalakshmi.edu.in", the page number "d36", and the language "en". The copyright notice "Copyright © 1999, 2023, Oracle and/or Its affiliates." and the version "Oracle APEX 23.2.4" are also present.

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

QUERY:

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

OUTPUT:



The screenshot shows the Oracle Apex SQL Workshop interface. The top navigation bar includes tabs for 'What's New', 'Untitled', 'oracle apex - Yahoo India Site Search', 'SQL Commands', and 'Chat Blackbox: AI Code Gen'. Below the navigation is a toolbar with various icons. The main workspace is titled 'APEX' and contains a 'SQL Workshop' tab. A search bar and a schema dropdown ('WKSP_D36') are also present. The SQL editor area contains the PL/SQL code for the trigger. The code is numbered from 1 to 12. Lines 1 through 11 are visible, while line 12 is partially cut off at the bottom. The 'Results' tab is selected at the bottom, showing the output: 'Function created.' and '0.03 seconds'. The status bar at the bottom right indicates 'Oracle APEX 23.2.4'.

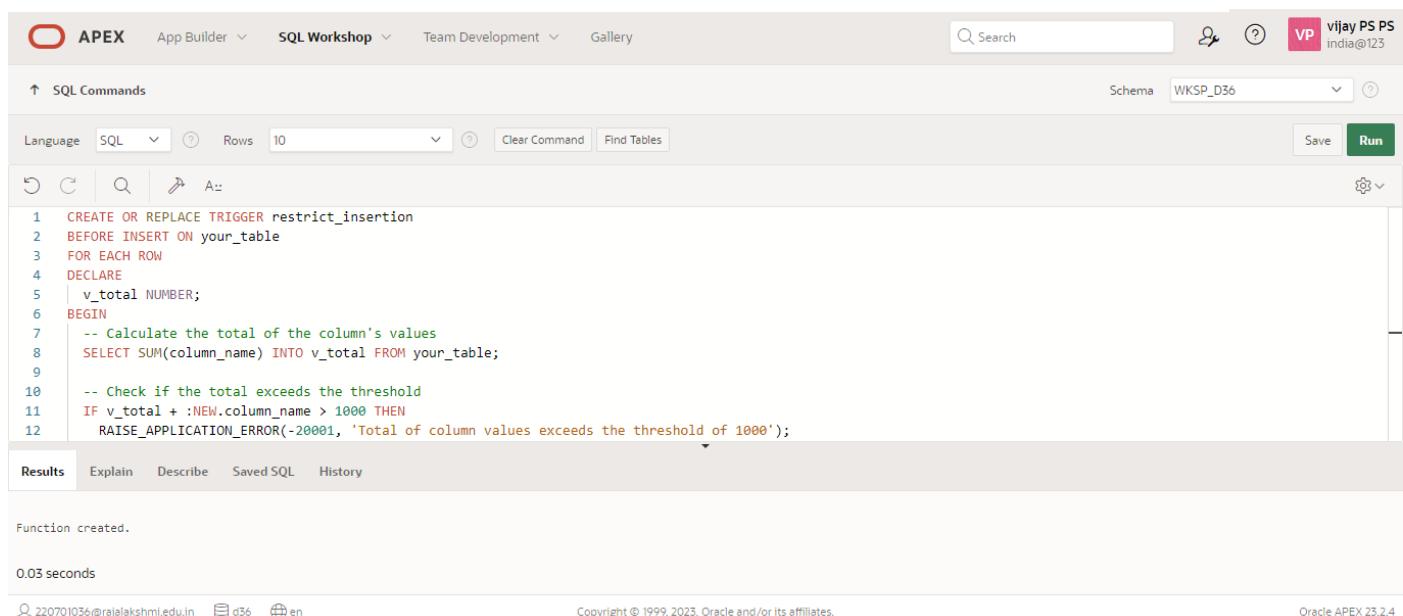
```
1 CREATE OR REPLACE TRIGGER check_duplicate_values
2 BEFORE INSERT OR UPDATE ON your_table
3 FOR EACH ROW
4 DECLARE
5     v_count NUMBER;
6 BEGIN
7     -- Check if the new value already exists in the table
8     SELECT COUNT(*) INTO v_count FROM your_table WHERE specific_column = :NEW.specific_column;
9
10    -- If the new value already exists, raise an exception
11    IF v_count > 0 THEN
12        RAISE_APPLICATION_ERROR(-20001, 'Duplicate value found in column: ' || :NEW.specific_column);
```

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

QUERY:

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

OUTPUT:



The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. A search bar and user information (vijay PS PS India@123) are also present. The main workspace is titled 'SQL Commands' and contains the PL/SQL code for creating a trigger named 'check_threshold'. The code defines a trigger that fires before inserting or updating rows in the 'threshold_table'. It declares variables 'v_sum' and 'v_threshold' (set to 10000), calculates the current sum of values in the table plus the new value, and raises an exception if it exceeds the threshold. The 'Run' button at the bottom right is highlighted.

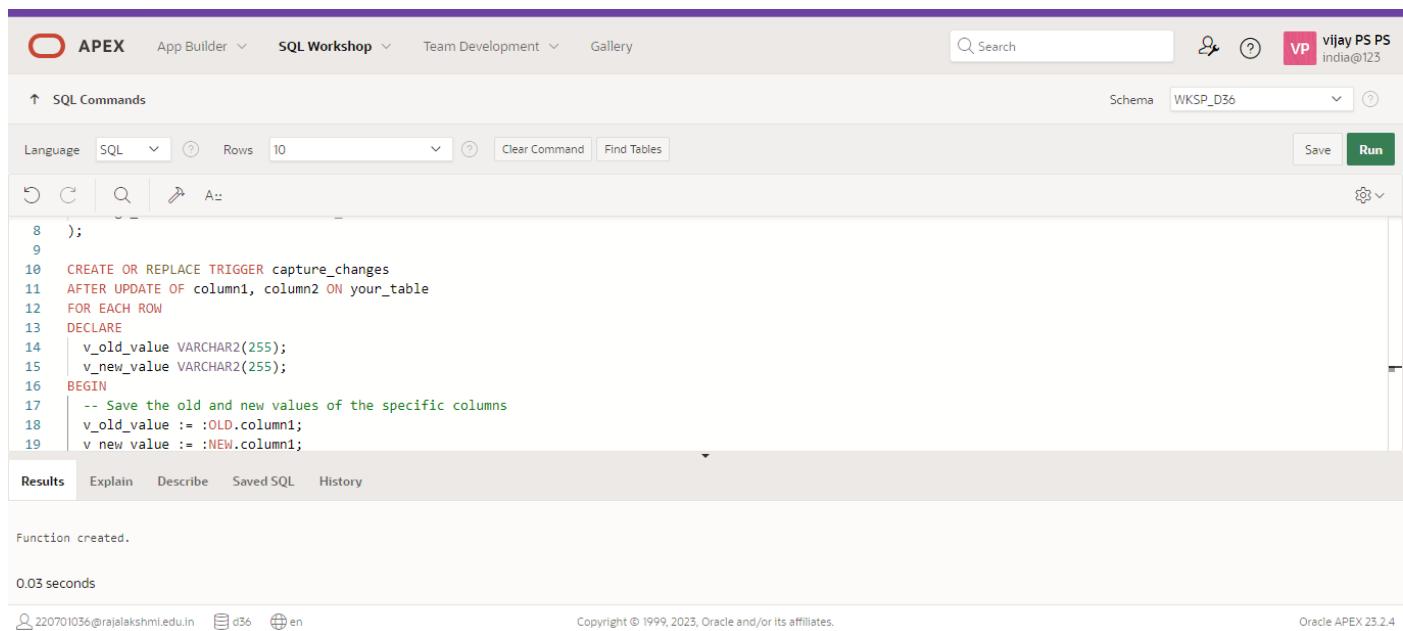
```
1 CREATE OR REPLACE TRIGGER check_threshold
2 BEFORE INSERT OR UPDATE ON threshold_table
3 FOR EACH ROW
4 DECLARE
5     v_sum NUMBER;
6     v_threshold NUMBER := 10000; -- Set your threshold here
7 BEGIN
8     SELECT SUM(value_col) INTO v_sum FROM threshold_table;
9     v_sum := v_sum + :NEW.value_col;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

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

QUERY:

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

OUTPUT:



The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. On the right, there is a search bar, a user profile for 'vijay PS PS india@123', and a schema dropdown set to 'WKSP_D36'. The main workspace is titled 'SQL Commands' and contains a SQL editor with the following code:

```
8 );
9
10 CREATE OR REPLACE TRIGGER capture_changes
11 AFTER UPDATE OF column1, column2 ON your_table
12 FOR EACH ROW
13 DECLARE
14     v_old_value VARCHAR2(255);
15     v_new_value VARCHAR2(255);
16 BEGIN
17     -- Save the old and new values of the specific columns
18     v_old_value := :OLD.column1;
19     v_new_value := :NEW.column1;
```

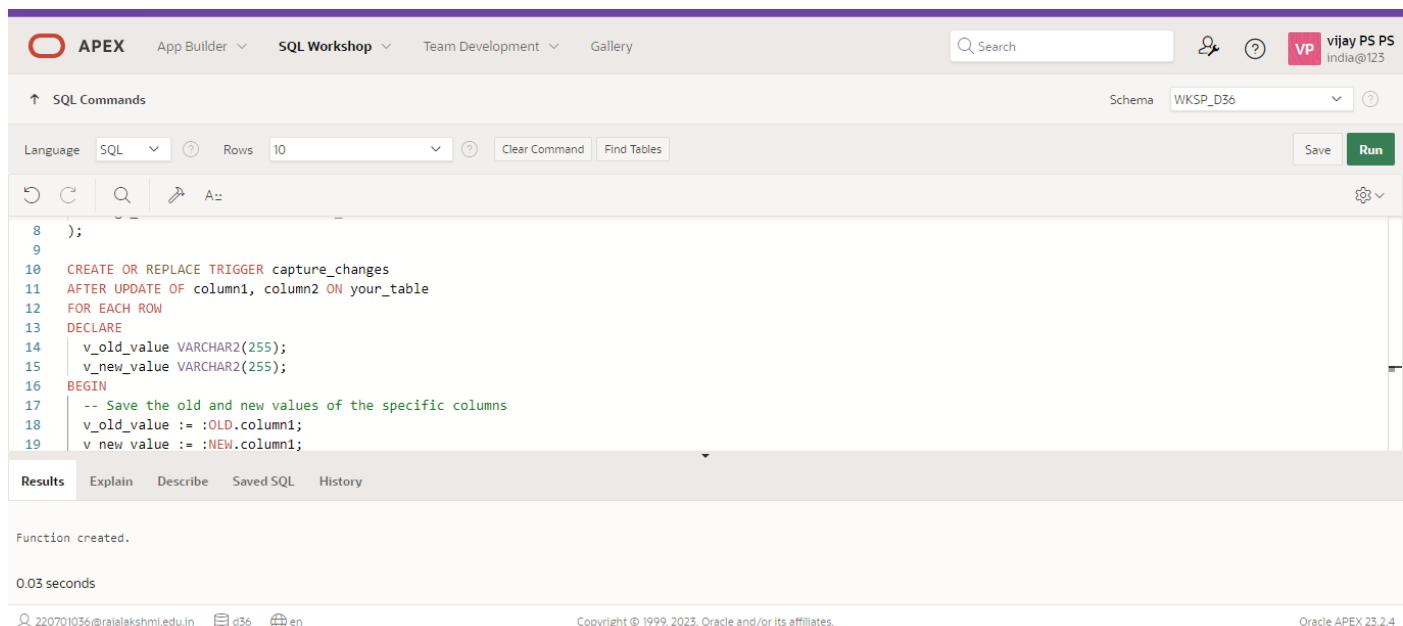
Below the code editor, there are tabs for 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History'. The 'Results' tab is selected, displaying the message 'Function created.' and '0.03 seconds'. At the bottom, the footer includes the URL '220701036@rajalakshmi.edu.in', the session ID 'd36', and the language 'en', along with copyright information 'Copyright © 1999, 2023, Oracle and/or its affiliates.' and the version 'Oracle APEX 23.2.4'.

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

QUERY:

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

OUTPUT:



The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. On the right, there's a search bar, a user icon for 'vijay PS PS india@123', and a schema dropdown set to 'WKSP_D36'. The main workspace is titled 'SQL Commands' and contains the PL/SQL code for creating a trigger. The code is as follows:

```
8  );
9
10 CREATE OR REPLACE TRIGGER capture_changes
11 AFTER UPDATE OF column1, column2 ON your_table
12 FOR EACH ROW
13 DECLARE
14   v_old_value VARCHAR2(255);
15   v_new_value VARCHAR2(255);
16 BEGIN
17   -- Save the old and new values of the specific columns
18   v_old_value := :OLD.column1;
19   v new value := :NEW.column1;
```

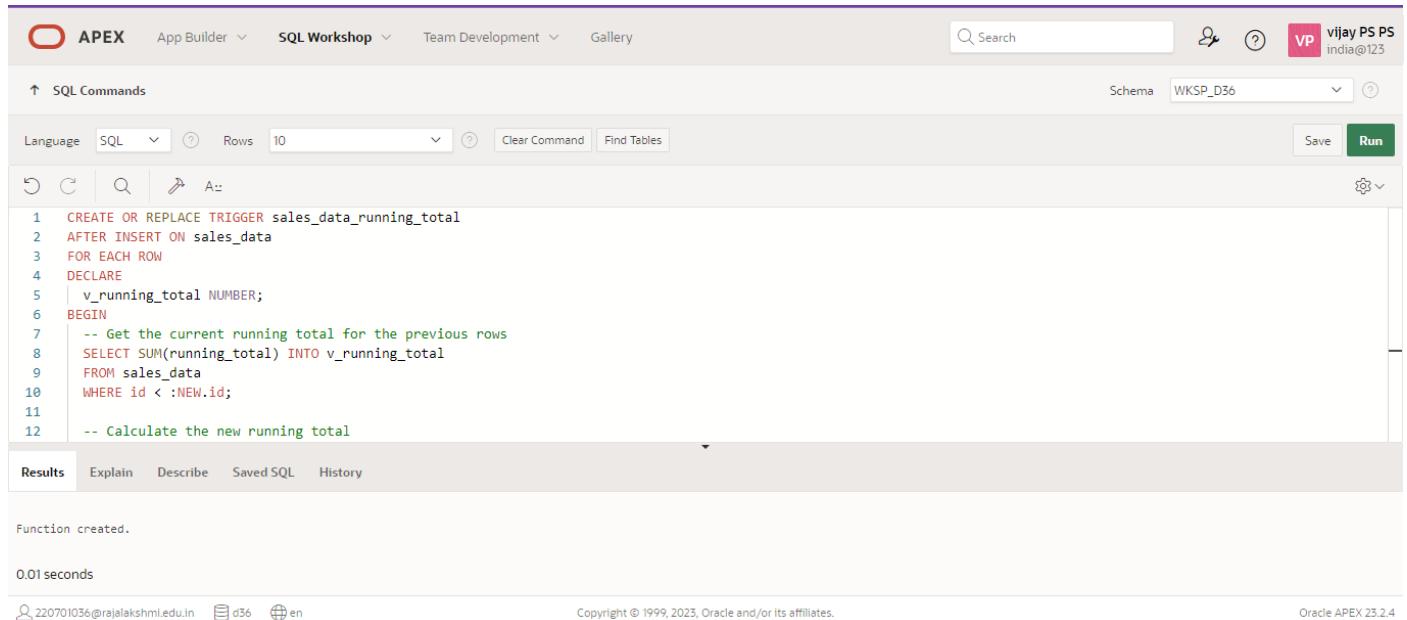
The 'Results' tab at the bottom shows the output: 'Function created.' and '0.03 seconds'. The bottom footer includes copyright information for Oracle and the APEX version.

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

QUERY:

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

OUTPUT:



The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. On the right, there is a search bar, a user profile for 'vijay PS PS india@123', and a 'Run' button. The main workspace is titled 'SQL Commands' and contains the following SQL code:

```
1 CREATE OR REPLACE TRIGGER sales_data_running_total
2 AFTER INSERT ON sales_data
3 FOR EACH ROW
4 DECLARE
5     v_running_total NUMBER;
6 BEGIN
7     -- Get the current running total for the previous rows
8     SELECT SUM(running_total) INTO v_running_total
9     FROM sales_data
10    WHERE id < :NEW.id;
11
12    -- Calculate the new running total

```

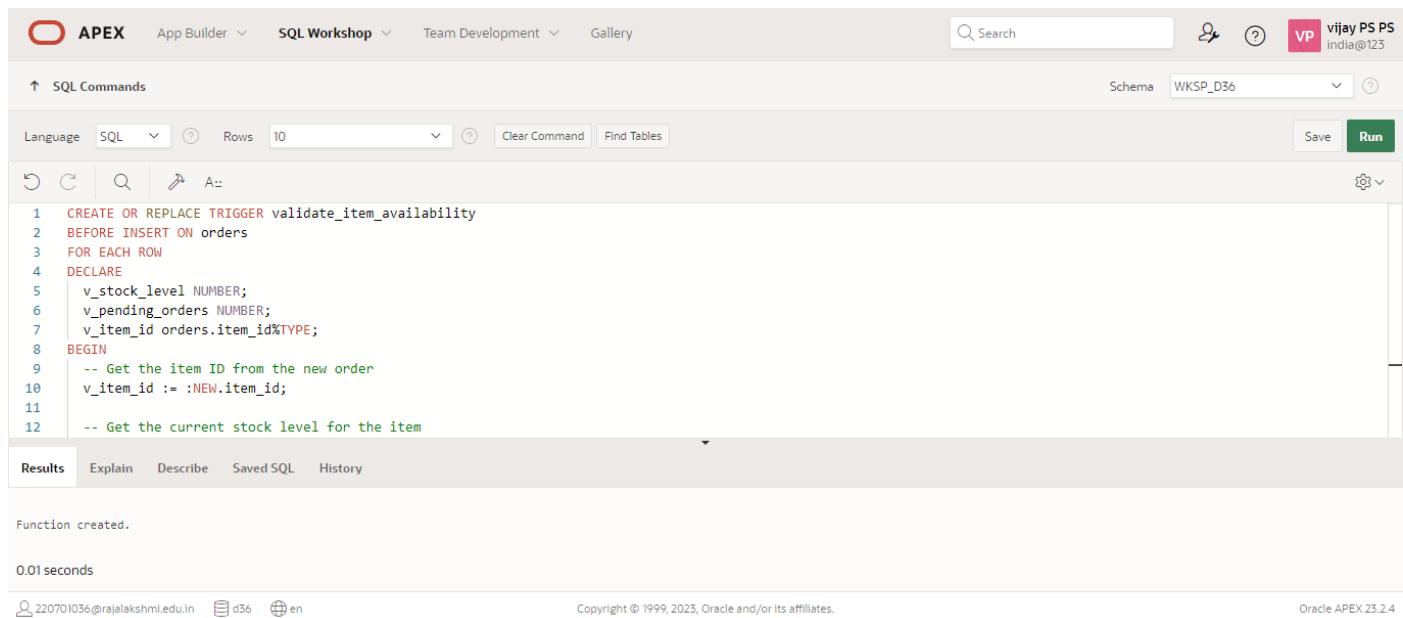
Below the code, there are tabs for 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History'. The 'Results' tab is selected, showing the message 'Function created.' and a execution time of '0.01 seconds'. At the bottom, the footer includes user information (220701036@rajalakshmi.edu.in, d56, en), copyright notice (Copyright © 1999, 2023, Oracle and/or its affiliates.), and the version 'Oracle APEX 23.2.4'.

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

QUERY:

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

OUTPUT:



The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for App Builder, SQL Workshop, Team Development, and Gallery. On the right, there's a user profile for 'vijay PS PS' and a schema dropdown set to 'WKSP_D36'. The main workspace is titled 'SQL Commands' and contains the PL/SQL code for the 'validate_order' trigger. Below the code, the 'Results' tab is selected, displaying the message 'Function created.' and a execution time of '0.01 seconds'. The bottom footer includes copyright information for Oracle and the APEX version.

```
1 CREATE OR REPLACE TRIGGER validate_order
2 BEFORE INSERT ON orders
3 FOR EACH ROW
4 DECLARE
5     v_stock_level NUMBER;
6     v_pending_orders NUMBER;
7     v_item_id orders.item_id%TYPE;
8 BEGIN
9     -- Get the item ID from the new order
10    v_item_id := :NEW.item_id;
11
12    -- Get the current stock level for the item
```

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:

The screenshot shows the myCompiler interface. In the top bar, there are buttons for English, Recent, Login, and Sign up. Below the bar, there is a search input field labeled "Enter a title...". Underneath it, there are two buttons: "MongoDB" and "Run". To the right of these buttons is a "Save" button. The main area contains a code editor with the following MongoDB query:

```
1 db.restaurants.find({  
2   $or: [  
3     { cuisine: { $not: { $in: ["American", "Chinese"] } } },  
4     { name: /^Wil/ }  
5   ],  
6 }, {  
7   restaurant_id: 1,  
8   name: 1,  
9   borough: 1,  
10  cuisine: 1  
11});
```

To the right of the code editor is an "Output" window. It shows the command "mycompiler_mongodb> ...". Below it, the response "mycompiler_mongodb>" is shown, followed by "[Execution complete with exit code 0]". At the bottom of the output window, there is a small advertisement for FigJam AI.

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:

The screenshot shows the myCompiler interface. In the top bar, there are buttons for English, Recent, Login, and Sign up. Below the bar, there is a search input field labeled "Enter a title...". Underneath it, there are two buttons: "MongoDB" and "Run". To the right of these buttons is a "Save" button. The main area contains a code editor with the following MongoDB query:

```
1 db.restaurants.find({  
2   "grades": {  
3     $elemMatch: {  
4       "date": ISODate("2014-08-11T00:00:00Z"),  
5       "grade": "A",  
6       "score": 11  
7     }  
8   },  
9 }, {  
10   _id: 0,  
11   restaurant_id: 1,  
12   name: 1,  
13   grades: 1  
14});
```

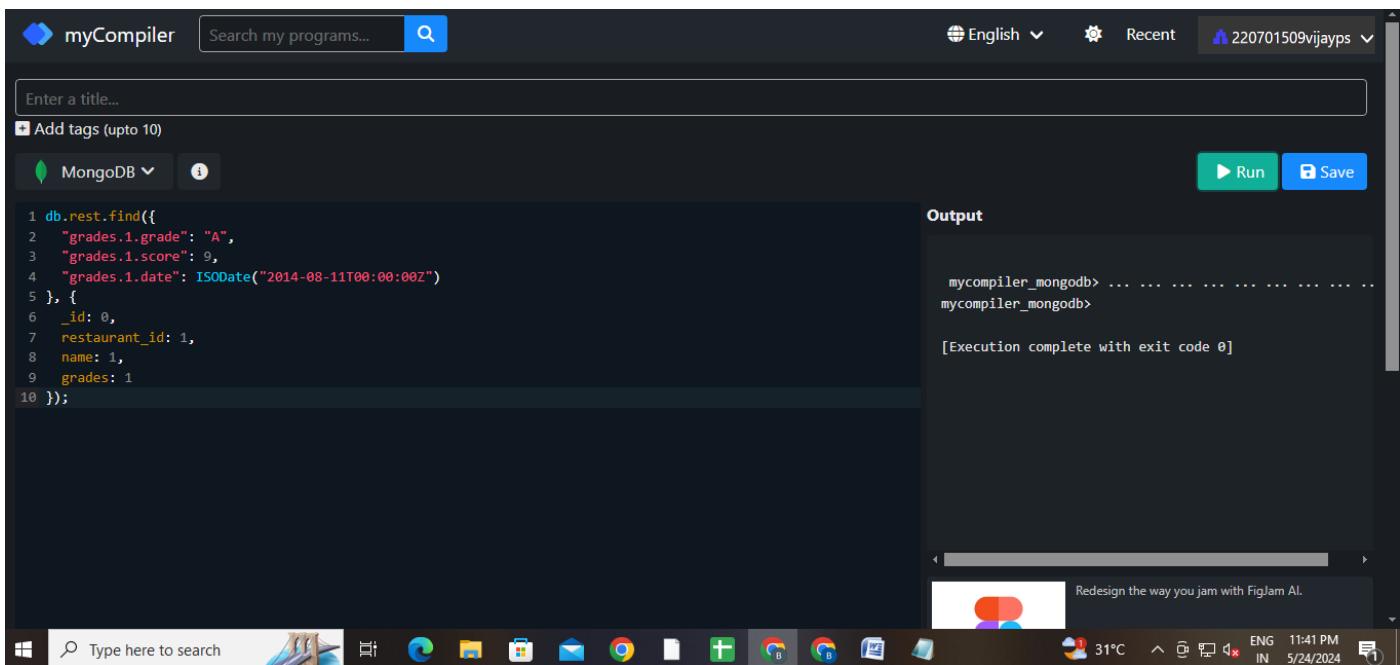
To the right of the code editor is an "Output" window. It shows the command "mycompiler_mongodb> ...". Below it, the response "mycompiler_mongodb>" is shown, followed by "[Execution complete with exit code 0]". At the bottom of the output window, there is a small advertisement for FigJam AI.

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

OUTPUT:



The screenshot shows the myCompiler application interface. In the top bar, there's a search bar for programs and a language selection dropdown set to English. On the right, there are buttons for Recent and a user profile (220701509vijayps). The main area has tabs for MongoDB and another one that is partially visible. A code editor window contains the following MongoDB query:

```
1 db.rest.find({  
2   "grades.1.grade": "A",  
3   "grades.1.score": 9,  
4   "grades.1.date": ISODate("2014-08-11T00:00:00Z")  
5 }, {  
6   _id: 0,  
7   restaurant_id: 1,  
8   name: 1,  
9   grades: 1  
10});
```

To the right of the code editor is an 'Output' window showing the command prompt and the results of the execution:

```
mycompiler_mongodb> ...  
mycompiler_mongodb>  
[Execution complete with exit code 0]
```

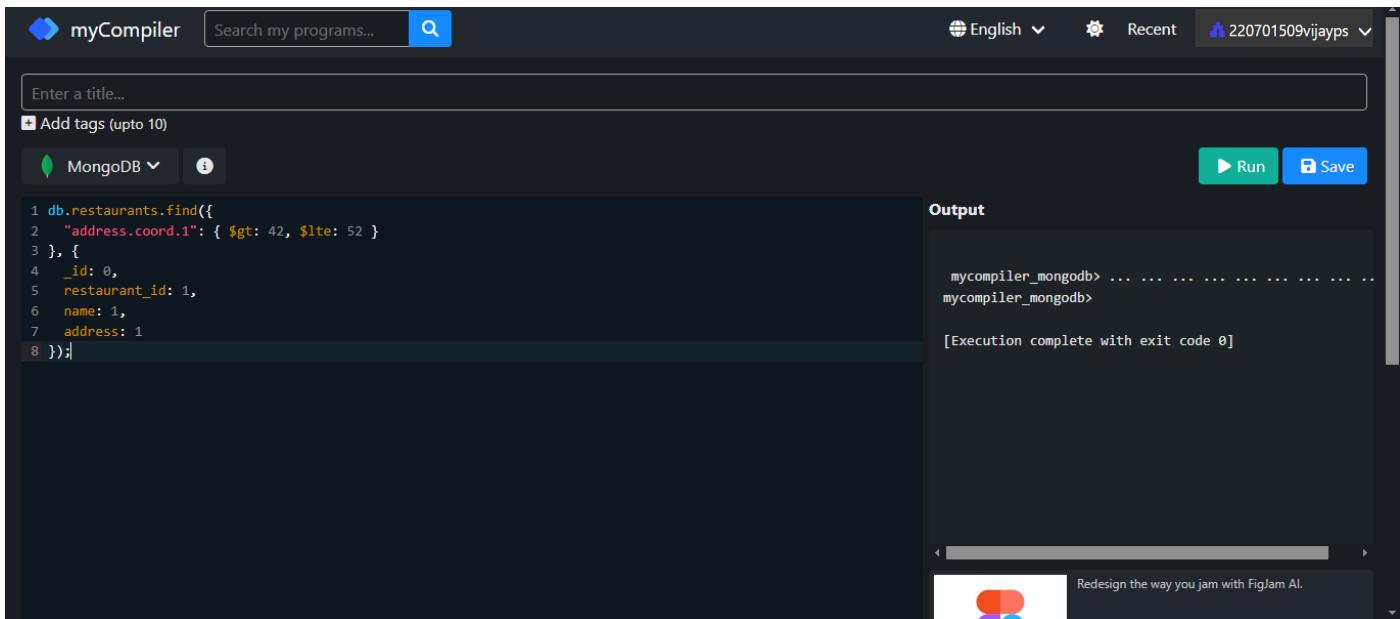
The bottom of the screen shows a Windows taskbar with various icons and system status information.

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:



The screenshot shows the myCompiler application interface. In the top bar, there's a search bar for programs and a language selection dropdown set to English. On the right, there are buttons for Recent and a user profile (220701509vijayps). The main area has tabs for MongoDB and another one that is partially visible. A code editor window contains the following MongoDB query:

```
1 db.restaurants.find({  
2   "address.coord.1": { $gt: 42, $lte: 52 }  
3 }, {  
4   _id: 0,  
5   restaurant_id: 1,  
6   name: 1,  
7   address: 1  
8});
```

To the right of the code editor is an 'Output' window showing the command prompt and the results of the execution:

```
mycompiler_mongodb> ...  
mycompiler_mongodb>  
[Execution complete with exit code 0]
```

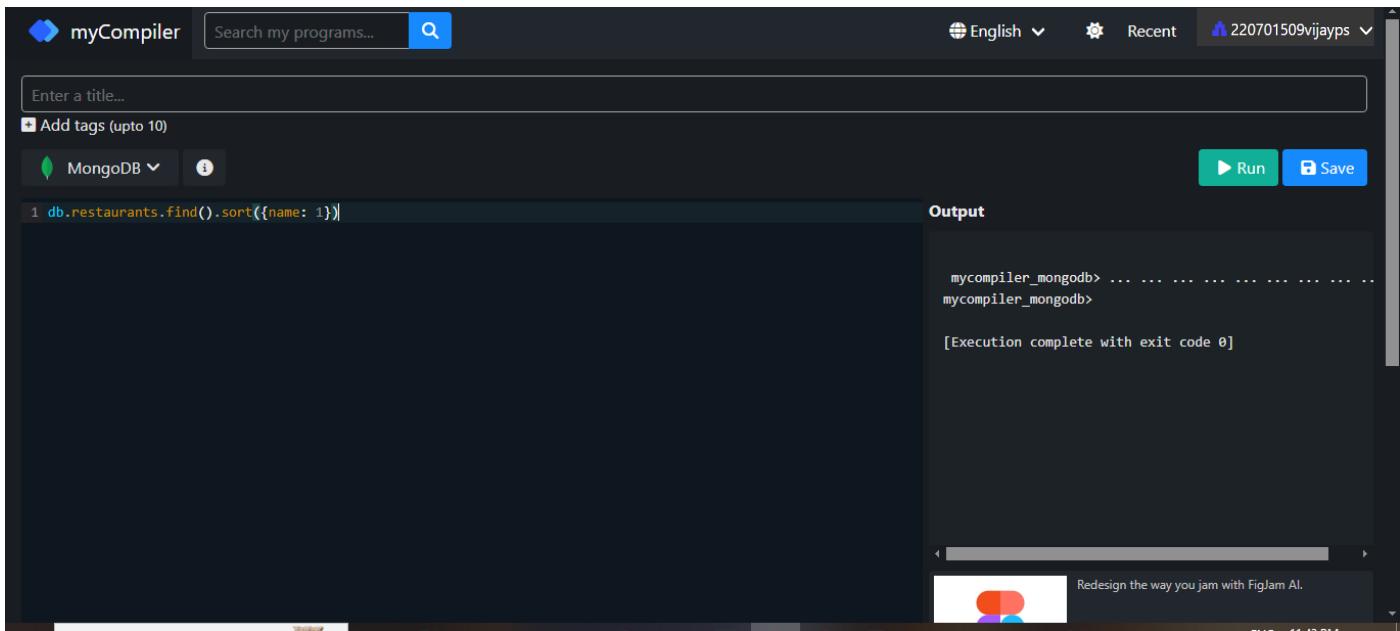
The bottom of the screen shows a Windows taskbar with various icons and system status information.

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:



The screenshot shows the myCompiler application interface. In the top bar, there is a search bar for 'my programs...', a language selector for 'English', and a user profile for '220701509vijayps'. Below the search bar, there is a text input field with placeholder 'Enter a title...' and a checkbox for 'Add tags (upto 10)'. On the left, there is a MongoDB connection status icon and a dropdown menu. On the right, there are 'Run' and 'Save' buttons. The main area contains a code editor with the following MongoDB query:

```
1 db.restaurants.find().sort({name: 1})
```

To the right of the code editor is the 'Output' panel, which displays the results of the query execution:

```
mycompiler_mongodb> ...
mycompiler_mongodb>
[Execution complete with exit code 0]
```

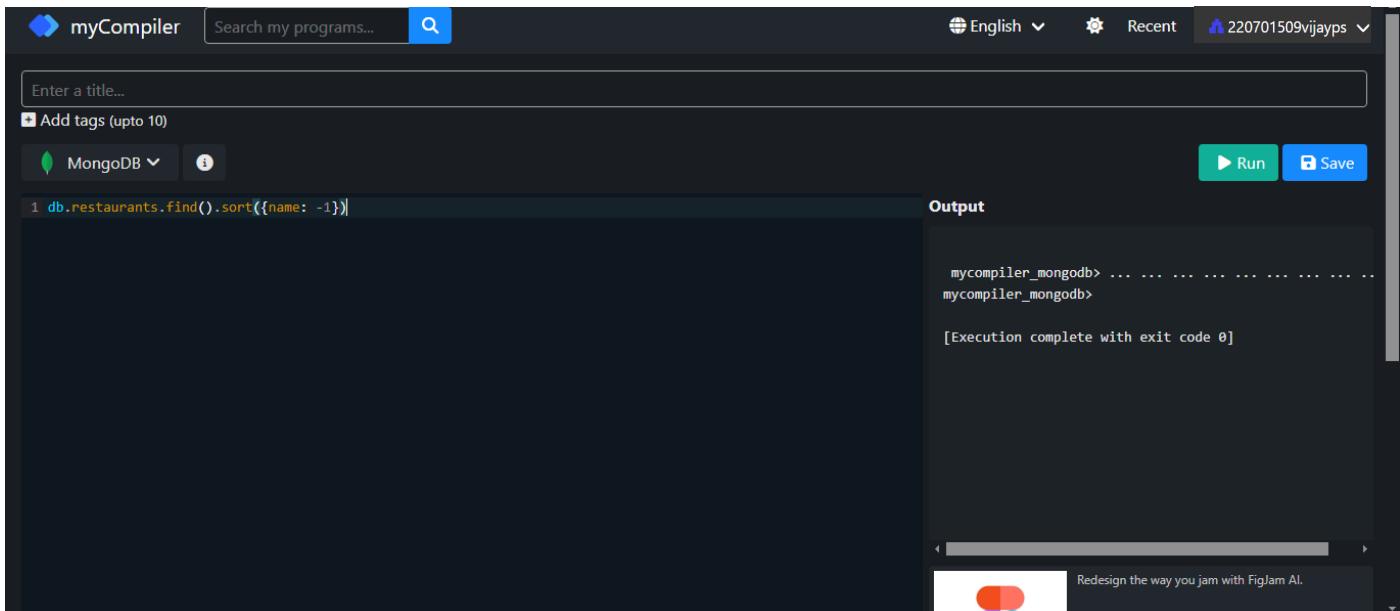
At the bottom of the interface, there is a small advertisement for FigJam AI.

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:



The screenshot shows the myCompiler application interface. The layout is identical to the previous screenshot, with the same top bar, search bar, language selector, and user profile. The code editor contains the same MongoDB query as the previous screenshot:

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

The 'Output' panel shows the results of the query execution:

```
mycompiler_mongodb> ...
mycompiler_mongodb>
[Execution complete with exit code 0]
```

At the bottom of the interface, there is a small advertisement for FigJam AI.

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:

The screenshot shows the myCompiler interface. In the top bar, there are buttons for English, Recent, and a user profile (220701509vijayps). The main area has a search bar 'Search my programs...' and a 'Run' button. A title input field says 'Enter a title...' and a tags input field says '+ Add tags (upto 10)'. Below this, a MongoDB dropdown is set to 'MongoDB' and a help icon is shown. The code input field contains the following MongoDB query:

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

To the right, an 'Output' panel shows the results of the query execution:

```
mycompiler_mongodb> ...
mycompiler_mongodb>
[Execution complete with exit code 0]
```

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:

The screenshot shows the myCompiler interface. In the top bar, there are buttons for English, Recent, and a user profile (220701509vijayps). The main area has a search bar 'Search my programs...' and a 'Run' button. A title input field says 'Enter a title...' and a tags input field says '+ Add tags (upto 10)'. Below this, a MongoDB dropdown is set to 'MongoDB' and a help icon is shown. The code input field contains the following MongoDB query:

```
1 db.restaurants.find({ "address.street": { $exists: true } }).count() == db.restaurants.count()
```

To the right, an 'Output' panel shows the results of the query execution:

```
mycompiler_mongodb> ...
mycompiler_mongodb>
[Execution complete with exit code 0]
```

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:

The screenshot shows the myCompiler MongoDB interface. In the code editor, a query is written:

```
1 db.restaurants.find({ "address.coord.0": { $type: "double" }, "address.coord.1": { $type: "double" } })
```

The output window shows the results of the query execution:

```
mycompiler_mongodb> ...
mycompiler_mongodb>
[Execution complete with exit code 0]
```

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:

The screenshot shows the myCompiler MongoDB interface. In the code editor, a query is written:

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

The output window shows the results of the query execution:

```
mycompiler_mongodb> ...
mycompiler_mongodb>
[Execution complete with exit code 0]
```

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:

The screenshot shows the myCompiler MongoDB interface. In the code editor, a query is written:

```
1 db.restaurants.find({ name: { $regex: /mon/i } }, { _id: 0, name: 1, borough: 1, address: 1, cuisine: 1 })
```

The output window shows the results of the query:

```
mycompiler_mongodb> ... ... ... ...
mycompiler_mongodb>
[Execution complete with exit code 0]
```

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:

The screenshot shows the myCompiler MongoDB interface. In the code editor, a query is written:

```
1 db.restaurants.find({ name: { $regex: /^Mad/i } }, { _id: 0, name: 1, borough: 1, address: 1, cuisine: 1 })
```

The output window shows the results of the query:

```
mycompiler_mongodb> ... ... ...
mycompiler_mongodb>
[Execution complete with exit code 0]
```

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:

The screenshot shows the myCompiler interface. In the top left, there's a logo and the text "myCompiler". A search bar with the placeholder "Search my programs..." and a magnifying glass icon is next to it. On the right, there are language selection ("English"), recent projects ("Recent"), and user account ("220701509vijayps") buttons. Below the search bar is a text input field with placeholder "Enter a title...". Underneath it is a checkbox labeled "Add tags (upto 10)". The main workspace has tabs for "MongoDB" and "Save" (with a help icon). A code editor window contains the following MongoDB query:

```
1 db.restaurants.find({ "grades.score": { $lt: 5 } }, { _id: 0, name: 1, borough: 1, address: 1, cuisine: 1 })
```

To the right of the code editor is an "Output" panel. It shows the command "mycompiler_mongodb> ..." followed by the results of the query. The output text is:

```
mycompiler_mongodb> ...
mycompiler_mongodb>
[Execution complete with exit code 0]
```

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:

The screenshot shows the myCompiler interface. The layout is identical to the previous one, with the "MongoDB" tab selected in the workspace. The code editor window contains the same MongoDB query as the previous screenshot:

```
1 db.restaurants.find({ "grades.score": { $lt: 5 }, borough: "Manhattan" }, { _id: 0, name: 1, borough: 1, address: 1, cuisine: 1 })
```

The "Output" panel shows the command "mycompiler_mongodb> ..." followed by the results of the query. The output text is:

```
mycompiler_mongodb> ...
mycompiler_mongodb>
[Execution complete with exit code 0]
```

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:

The screenshot shows the myCompiler interface. In the top bar, there's a search bar with "Search my programs..." and a magnifying glass icon. On the right, there are buttons for "English", "Recent", and a user profile "220701509vijayaps". Below the search bar is a text input field with placeholder "Enter a title...". Underneath it is a "Add tags (upto 10)" button. A "MongoDB" dropdown menu is open, showing "MongoDB" and an info icon. To the right are "Run" and "Save" buttons. The main area contains a code editor with the following MongoDB query:

```
1, {borough: "Brooklyn"}], "grades.score": { $lt: 5 } }, { _id: 0, name: 1, borough: 1, address: 1, cuisine: 1 }) Output
```

Below the code editor is the output window, which shows the command being run and the result:

```
mycompiler_mongodb> ... . . . . .  
mycompiler_mongodb>  
[Execution complete with exit code 0]
```

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:

The screenshot shows the myCompiler interface. The setup is identical to the previous one, with the MongoDB dropdown open and the "Run" and "Save" buttons visible. The code editor contains the same MongoDB query as the previous screenshot:

```
1, {grades.score": { $lt: 5 }, cuisine: { $ne: "American" } }, { _id: 0, name: 1, borough: 1, address: 1, cuisine: 1 }) Output
```

The output window shows the command being run and the result:

```
mycompiler_mongodb> ... . . . . .  
mycompiler_mongodb>  
[Execution complete with exit code 0]
```

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:

The screenshot shows the myCompiler interface. At the top, there's a search bar with "Search my programs..." and a magnifying glass icon. To the right are language selection ("English"), recent projects ("Recent"), and user information ("220701509vijayps"). Below the search bar is a text input field with placeholder "Enter a title...". Underneath it is a button "Add tags (upto 10)". On the left, there's a "MongoDB" dropdown and an "Info" button. On the right, there are "Run" and "Save" buttons. The main area is titled "Output" and contains the following text:
1\$lt: 5 }, cuisine: { \$nin: ["American", "Chinese"] } }, { _id: 0, name: 1, borough: 1, address: 1, cuisine: 1 } }
mycompiler_mongodb> ...
mycompiler_mongodb>
[Execution complete with exit code 0]
A small advertisement for FigJam AI is visible at the bottom right.

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:

The screenshot shows the myCompiler interface. The setup is identical to the previous one, with a search bar, language selection, and user info. The text input field is empty. The "MongoDB" dropdown and "Info" button are present. The "Run" and "Save" buttons are on the right. The "Output" section shows the query and its execution results:
1\$restaurants.find({ "grades.score": { \$in: [2, 6] } }, { _id: 0, name: 1, borough: 1, address: 1, cuisine: 1 })
mycompiler_mongodb> ...
mycompiler_mongodb>
[Execution complete with exit code 0]
A small advertisement for FigJam AI is visible at the bottom right.

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:

```
Enter a title...
+ Add tags (upto 10)
MongoDB i Run Save
1grades.score": { $in: [2, 6] }, borough: "Manhattan" }, { _id: 0, name: 1, borough: 1, address: 1, cuisine: 1 }) Output
```

```
mycompiler_mongodb> ...
mycompiler_mongodb>

[Execution complete with exit code 0]
```



Redesign the way you jam with FigJam AI.
ADS VIA CARBON

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:

```
Enter a title...
+ Add tags (upto 10)
MongoDB i Run Save
1rough: "Brooklyn"]}, "grades.score": { $in: [2, 6] } }, { _id: 0, name: 1, borough: 1, address: 1, cuisine: 1 }) Output
```

```
mycompiler_mongodb> ...
mycompiler_mongodb>

[Execution complete with exit code 0]
```



Redesign the way you jam with FigJam AI.
ADS VIA CARBON

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:

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:

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:

Enter a title...

Add tags (upto 10)

MongoDB 

 Run  Save

```
1 db.restaurants.find({ "grades.score": { $in: [2, 6] } }, { _id: 0, name: 1, borough: 1, address: 1, cuisine: 1 })
```

Output

```
mycompiler_mongodb> ... ... ... ... ...
mycompiler_mongodb>
[Execution complete with exit code 0]
```


 Redesign the way you jam with FigJam AI.

ADS VIA CARBON

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

RESULT:

MONGO DB

EX_NO: 20

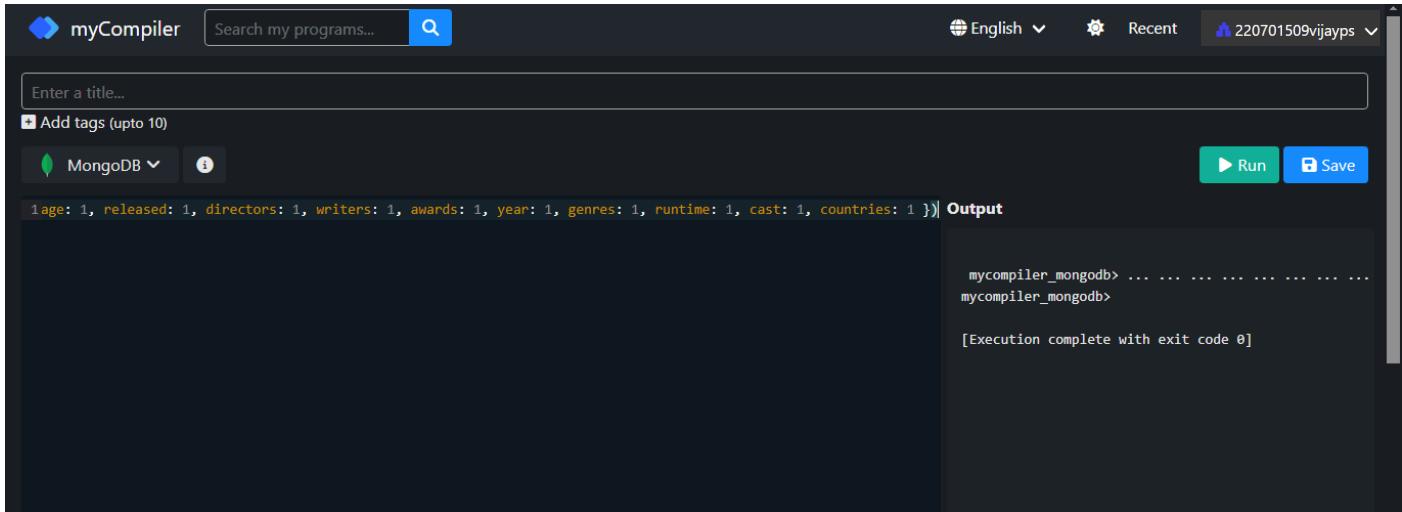
DATE:

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

QUERY:

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

OUTPUT:



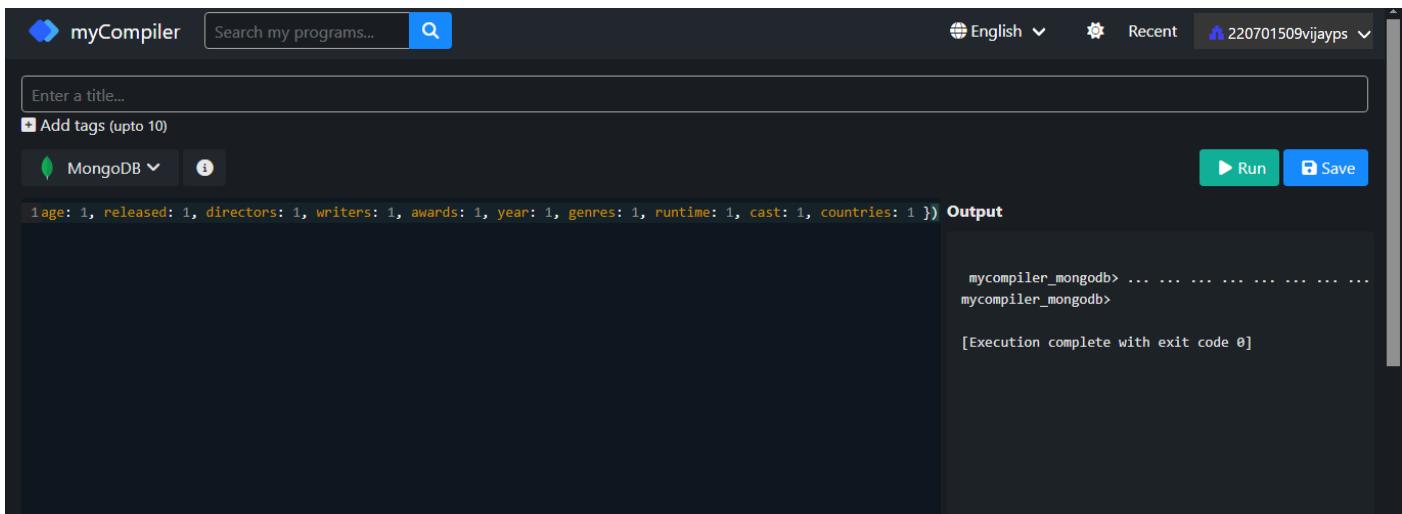
The screenshot shows the myCompiler interface with a dark theme. At the top, there's a search bar labeled "Search my programs..." and a "Run" button. Below the search bar, there's a text input field with placeholder text "Enter a title...". Underneath the input field, there's a checkbox labeled "Add tags (upto 10)". On the left side, there's a dropdown menu set to "MongoDB" and a help icon. On the right side, there are two buttons: a green "Run" button and a blue "Save" button. The main area is titled "Output" and contains the following text:
`1 age: 1, released: 1, directors: 1, writers: 1, awards: 1, year: 1, genres: 1, runtime: 1, cast: 1, countries: 1 }`
mycompiler_mongodb>
mycompiler_mongodb>
[Execution complete with exit code 0]

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:



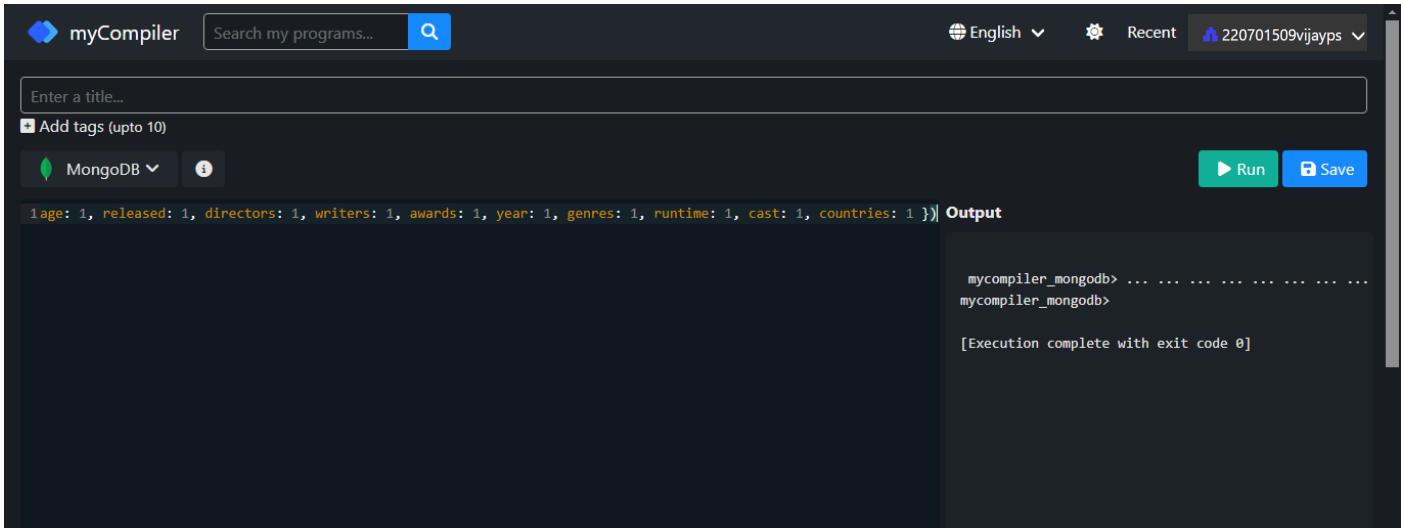
The screenshot shows the myCompiler interface with a dark theme. At the top, there's a search bar labeled "Search my programs..." and a "Run" button. Below the search bar, there's a text input field with placeholder text "Enter a title...". Underneath the input field, there's a checkbox labeled "Add tags (upto 10)". On the left side, there's a dropdown menu set to "MongoDB" and a help icon. On the right side, there are two buttons: a green "Run" button and a blue "Save" button. The main area is titled "Output" and contains the following text:
`1 age: 1, released: 1, directors: 1, writers: 1, awards: 1, year: 1, genres: 1, runtime: 1, cast: 1, countries: 1 }`
mycompiler_mongodb>
mycompiler_mongodb>
[Execution complete with exit code 0]

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

QUERY:

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

OUTPUT:



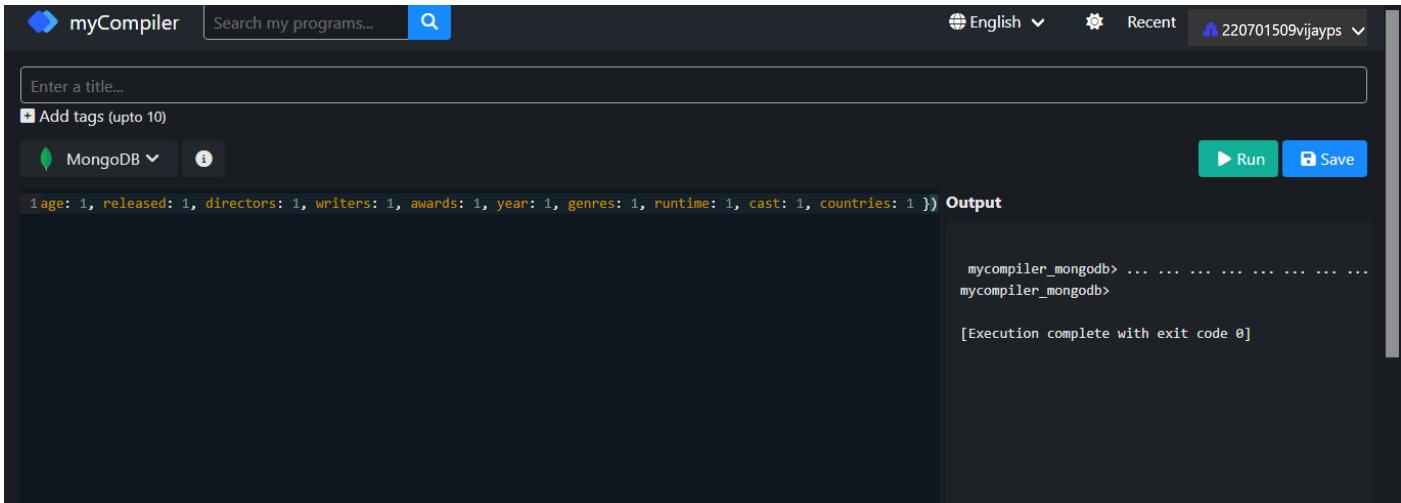
The screenshot shows the myCompiler interface. At the top, there's a search bar for programs and a language selection dropdown set to English. Below the search bar is a text input field labeled 'Enter a title...' and a checkbox for 'Add tags (upto 10)'. A MongoDB connection dropdown is set to 'MongoDB'. On the right side, there are 'Run' and 'Save' buttons. The main area is titled 'Output' and contains the command: '1 age: 1, released: 1, directors: 1, writers: 1, awards: 1, year: 1, genres: 1, runtime: 1, cast: 1, countries: 1 }'. Below this, the mongo shell output shows: 'mycompiler_mongodb> ...' followed by several dots, and '[Execution complete with exit code 0]'. The interface has a dark theme with light-colored text and buttons.

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:



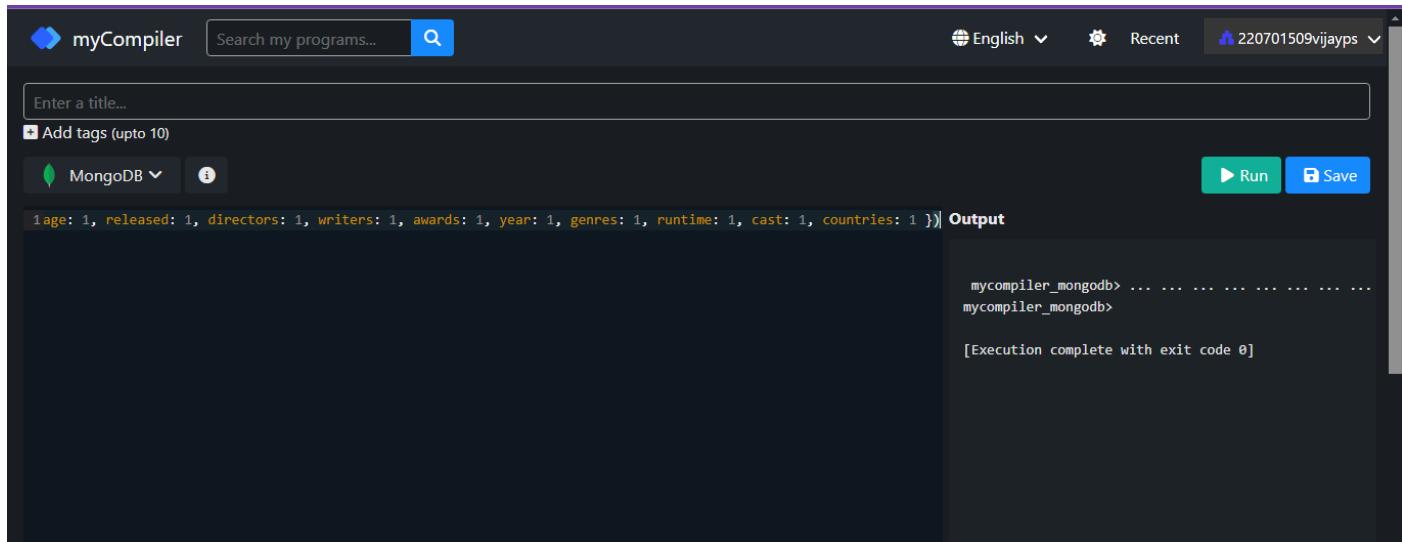
The screenshot shows the myCompiler interface. It has the same layout as the previous one, with a search bar, language selection, and MongoDB connection dropdown. The 'Run' and 'Save' buttons are present. The 'Output' section shows the command: '1 age: 1, released: 1, directors: 1, writers: 1, awards: 1, year: 1, genres: 1, runtime: 1, cast: 1, countries: 1 }'. The mongo shell output below it shows: 'mycompiler_mongodb> ...' followed by several dots, and '[Execution complete with exit code 0]'. The interface maintains its dark theme.

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:



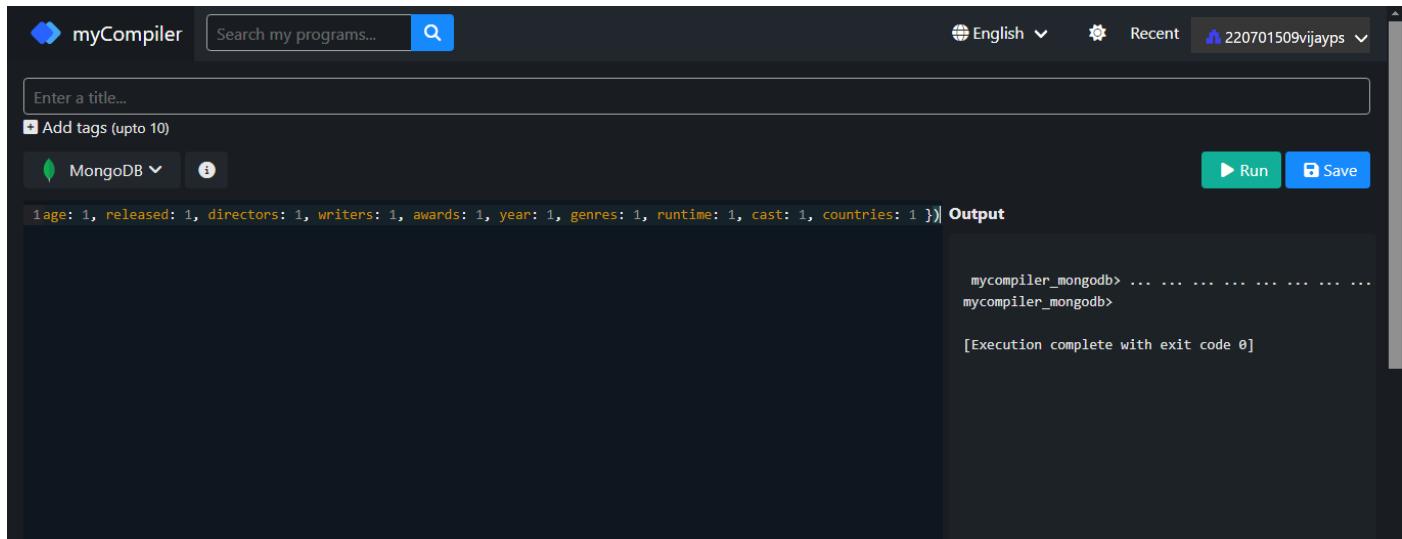
The screenshot shows the myCompiler interface. At the top, there's a search bar for programs and a language selector set to English. Below the search bar is a text input field labeled "Enter a title...". Underneath it is a button "Add tags (upto 10)". On the left, there are two dropdown menus: "MongoDB" and "Run/Save" buttons. The main area contains the MongoDB query: "db.movies.find({ countries: 'USA' })". To the right of the query is a "Output" section showing the results of the command. The output starts with "mycompiler_mongodb>" followed by "mycompiler_mongodb> [Execution complete with exit code 0]".

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

QUERY:

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

OUTPUT:



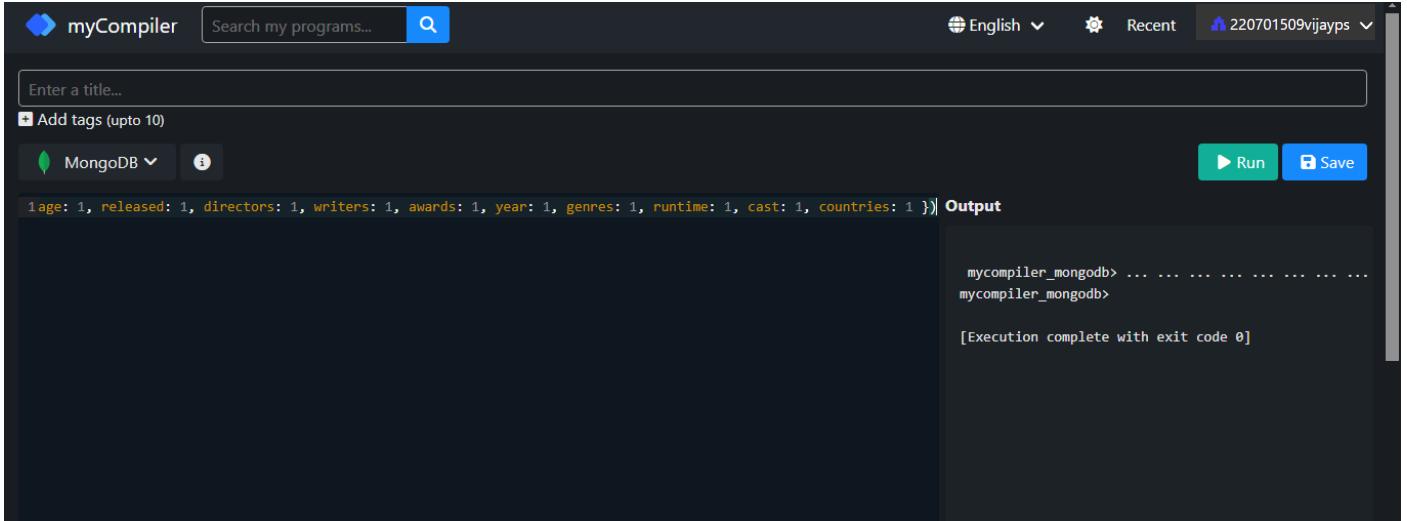
The screenshot shows the myCompiler interface. The setup is identical to the previous one: English language, "MongoDB" selected, and "Run/Save" buttons. The query "db.movies.find({ rated: 'UNRATED' })" is entered. The output shows "mycompiler_mongodb>" followed by "mycompiler_mongodb> [Execution complete with exit code 0]".

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:



The screenshot shows the myCompiler interface with the MongoDB extension selected. The query `db.movies.find({ 'imdb.votes': { $gt: 1000 } })` is entered in the code editor. The output window displays the command and its execution results:

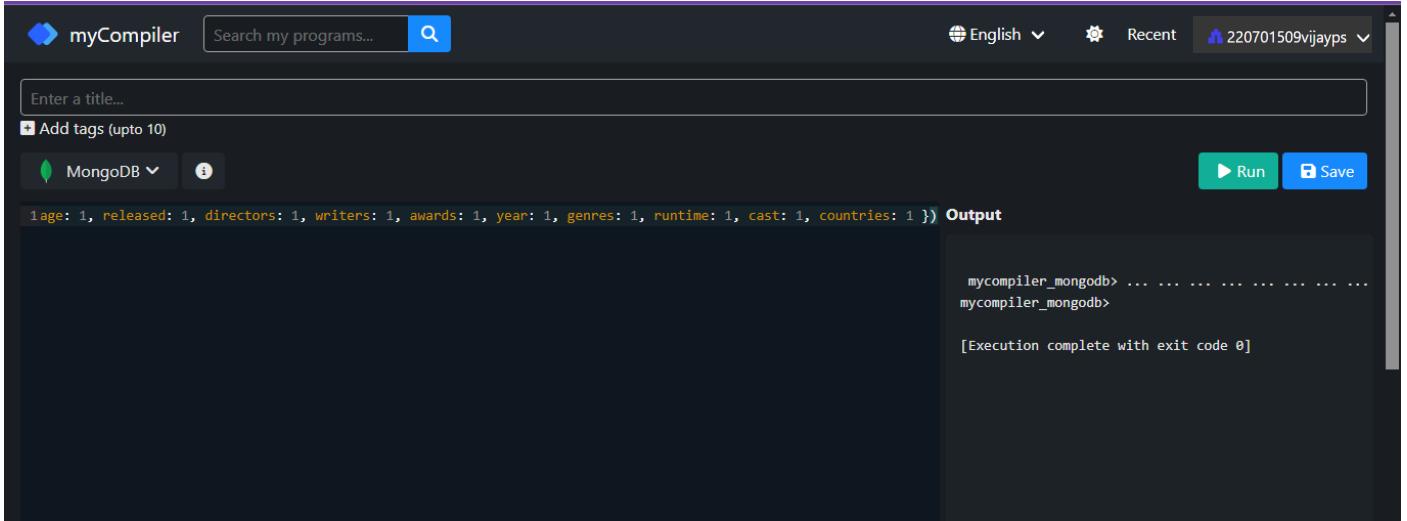
```
mycompiler_mongodb> ... ...
mycompiler_mongodb>
[Execution complete with exit code 0]
```

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:



The screenshot shows the myCompiler interface with the MongoDB extension selected. The query `db.movies.find({ 'imdb.rating': { $gt: 7 } })` is entered in the code editor. The output window displays the command and its execution results:

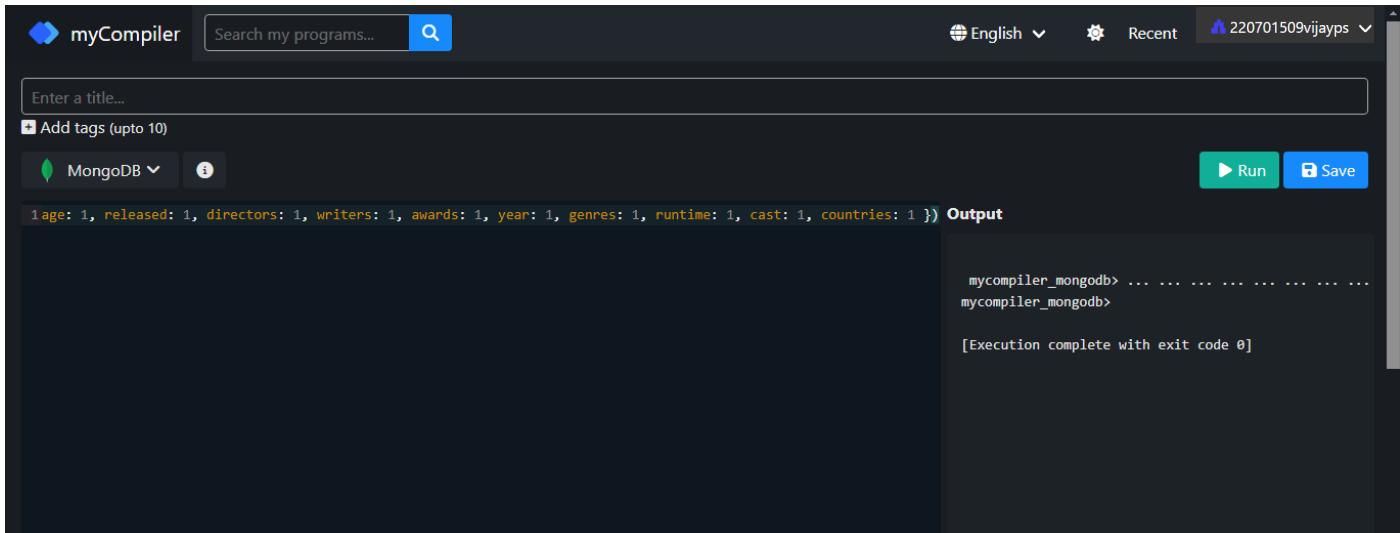
```
mycompiler_mongodb> ... ...
mycompiler_mongodb>
[Execution complete with exit code 0]
```

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:



The screenshot shows the myCompiler interface. At the top, there's a search bar for programs and a language selector set to English. Below the search bar is a text input field labeled "Enter a title..." and a checkbox for "Add tags (upto 10)". On the left, there are buttons for MongoDB, Run, and Save. The main area contains a code editor with the following MongoDB query:

```
1age: 1, released: 1, directors: 1, writers: 1, awards: 1, year: 1, genres: 1, runtime: 1, cast: 1, countries: 1 })
```

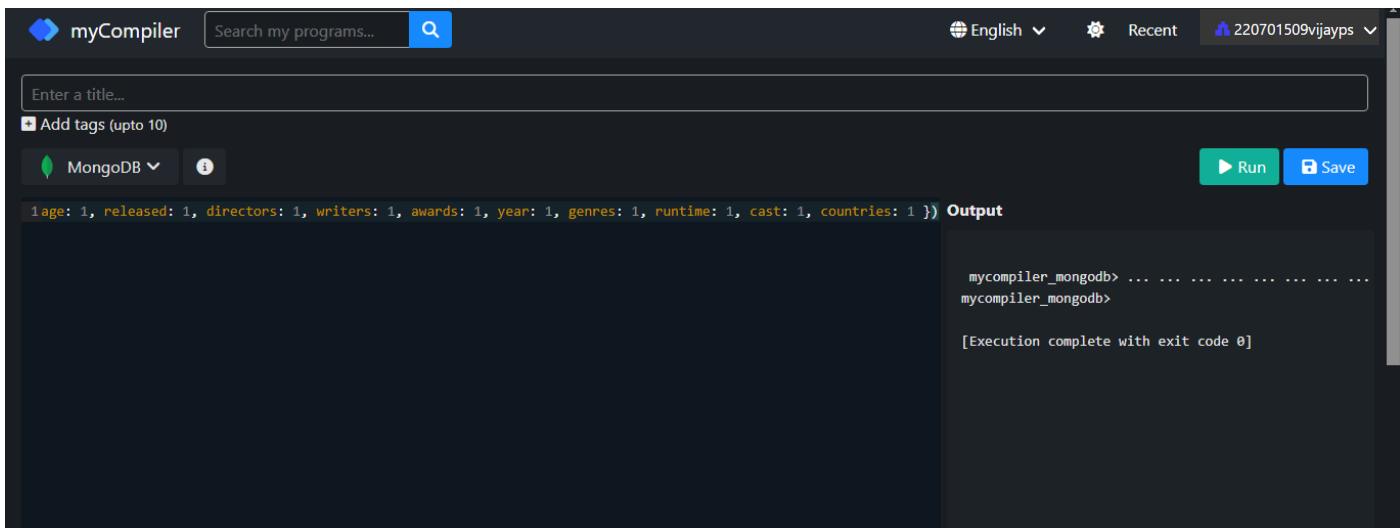
To the right of the code editor is an "Output" panel. It displays the command "mycompiler_mongodb> ...", the response "mycompiler_mongodb>", and the message "[Execution complete with exit code 0]".

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

QUERY:

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

OUTPUT:



The screenshot shows the myCompiler interface. At the top, there's a search bar for programs and a language selector set to English. Below the search bar is a text input field labeled "Enter a title..." and a checkbox for "Add tags (upto 10)". On the left, there are buttons for MongoDB, Run, and Save. The main area contains a code editor with the following MongoDB query:

```
1age: 1, released: 1, directors: 1, writers: 1, awards: 1, year: 1, genres: 1, runtime: 1, cast: 1, countries: 1 })
```

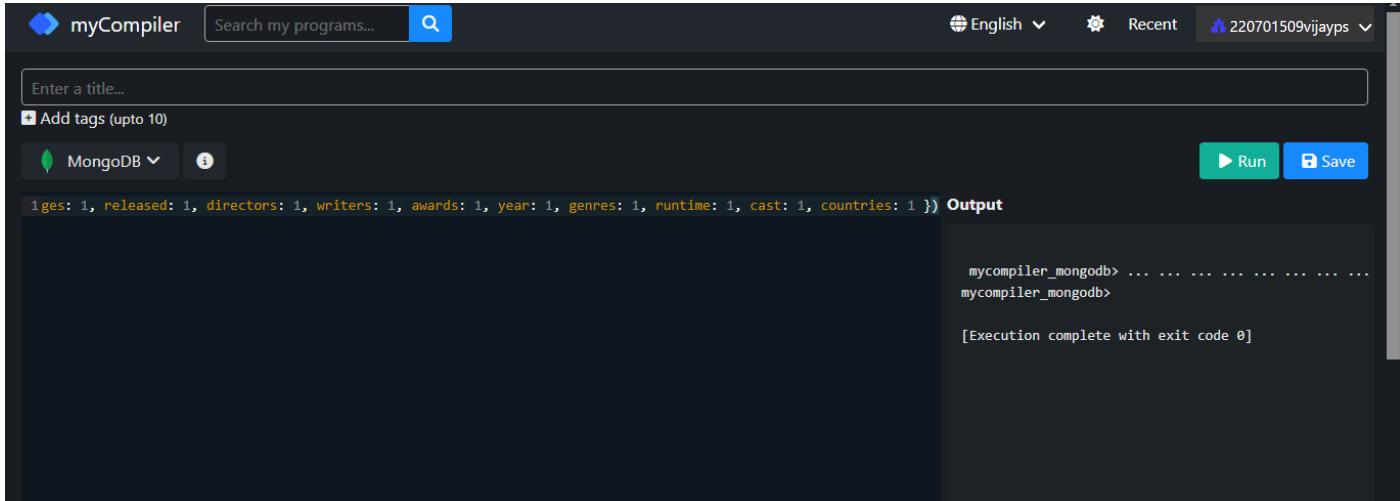
To the right of the code editor is an "Output" panel. It displays the command "mycompiler_mongodb> ...", the response "mycompiler_mongodb>", and the message "[Execution complete with exit code 0]".

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:



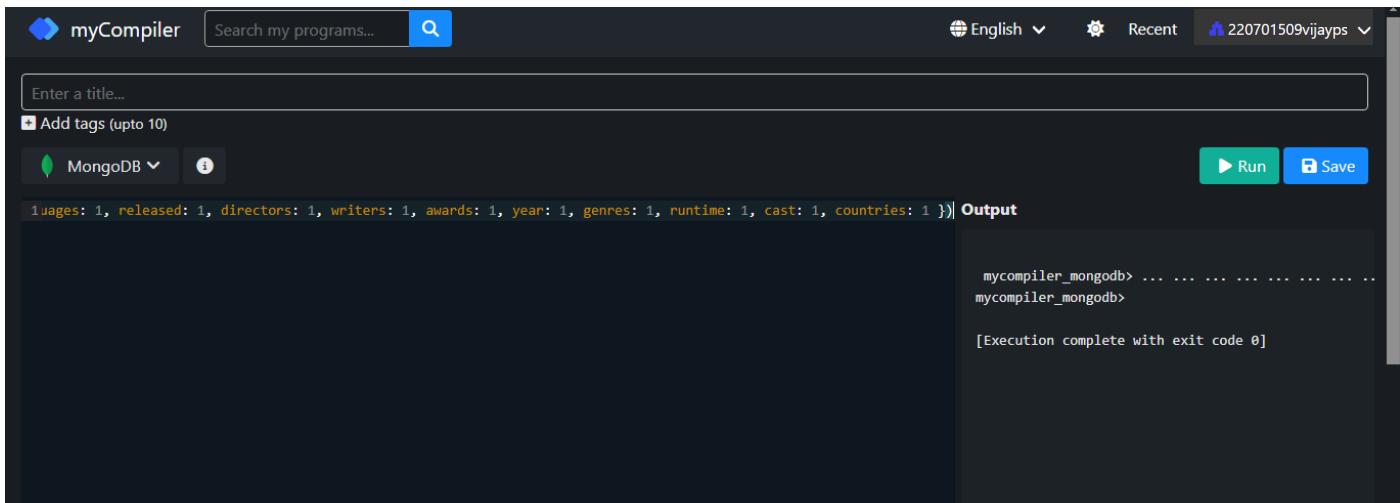
The screenshot shows the myCompiler interface. In the top bar, there are tabs for 'myCompiler', 'Search my programs...', and a magnifying glass icon. On the right, there are language selection dropdowns ('English'), a settings gear icon, and a user profile ('Recent' and '220701509vijayaps'). Below the top bar, there's a search bar with placeholder 'Enter a title...' and a checkbox for 'Add tags (upto 10)'. Underneath the search bar are two buttons: 'MongoDB' with a dropdown arrow and an info icon, and 'Run' with a play icon. To the right of these buttons is a 'Save' button with a disk icon. The main area is titled 'Output' and contains the MongoDB command: '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 })'. Below the command, the terminal output shows the command being run: 'mycompiler_mongodb> 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 })'. It also shows the message '[Execution complete with exit code 0]'. The background of the interface is dark grey.

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:



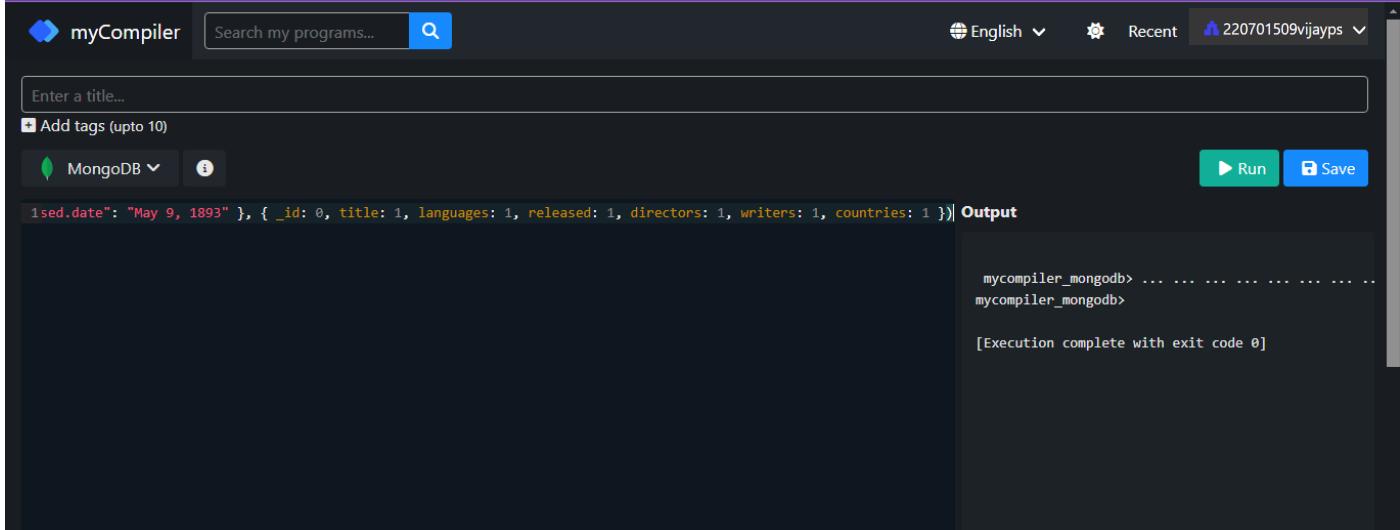
The screenshot shows the myCompiler interface, identical to the previous one but with a different query. The 'Output' section now displays the results of the filtered query: 'mycompiler_mongodb> db.movies.find({ "cast": "Charles Kayser" }, { "title": 1, "languages": 1, "released": 1, "directors": 1, "writers": 1, "awards": 1, "year": 1, "genres": 1, "runtime": 1, "cast": 1, "countries": 1 })'. It also shows the message '[Execution complete with exit code 0]'. The background is dark grey.

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:



The screenshot shows the myCompiler interface. In the top bar, there are tabs for "myCompiler", "Search my programs...", and a magnifying glass icon. On the right, there are language selection ("English"), recent projects ("Recent"), and user information ("220701509vijayps"). Below the top bar, there's a search bar labeled "Enter a title..." and a "Add tags (upto 10)" button. A dropdown menu shows "MongoDB". On the left, there's a "Run" and "Save" button. The main area contains a code editor with the following MongoDB query:

```
1sed.date": "May 9, 1893" }, { _id: 0, title: 1, languages: 1, released: 1, directors: 1, writers: 1, countries: 1 })| Output
```

To the right of the code editor is the "Output" panel, which displays the command prompt:

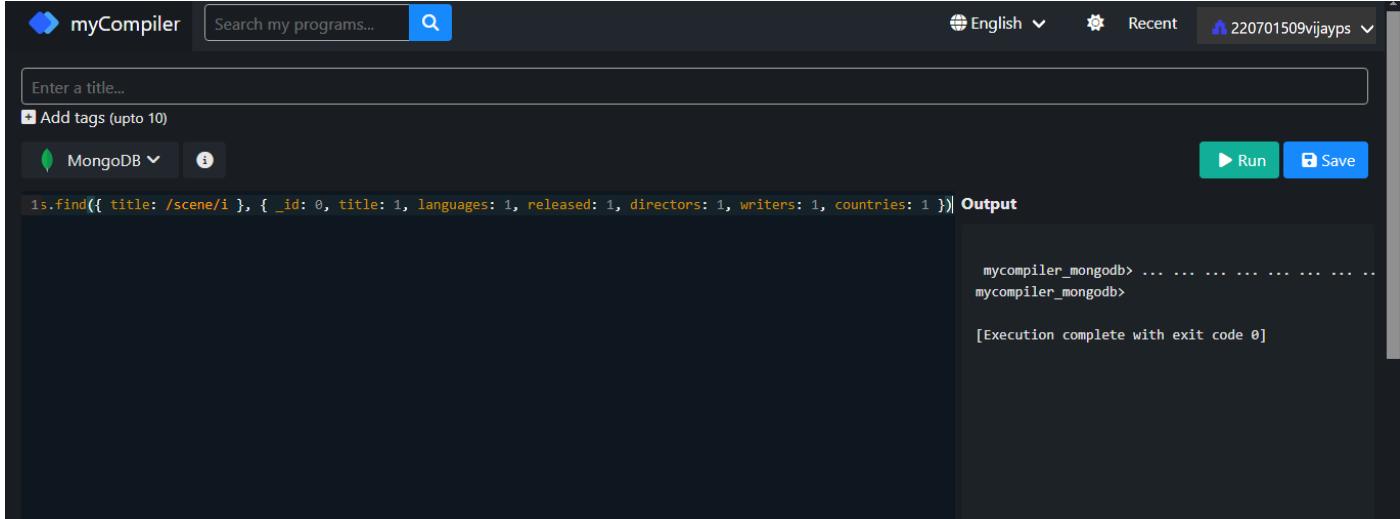
```
mycompiler_mongodb> ... . . . . . . . . .  
mycompiler_mongodb>  
[Execution complete with exit code 0]
```

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:



The screenshot shows the myCompiler interface. The layout is identical to the previous screenshot, with tabs for "myCompiler", "Search my programs...", and a magnifying glass icon at the top. The "MongoDB" tab is selected. On the left, there's a "Run" and "Save" button. The main area contains the same MongoDB query as the previous screenshot:

```
15.find({ title: /scene/i }, { _id: 0, title: 1, languages: 1, released: 1, directors: 1, writers: 1, countries: 1 })| Output
```

The "Output" panel shows the command prompt:

```
mycompiler_mongodb> ... . . . . . . . . .  
mycompiler_mongodb>  
[Execution complete with exit code 0]
```

Evaluation Procedure	Marks awarded

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

RESULT: