

Creating and Managing Tables

EX_NO:1

DATE:17/2/24

- 1.Create the DEPARTMENTS 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 departments(id number(7),name varchar2(25));
```

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 the SQL command: 'Create table dept(id number(7),name varchar2(25));'. Below the command, the results show 'Table created.' and a execution time of '0.02 seconds'. The bottom right corner indicates 'Oracle APEX 23.2.4'.

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

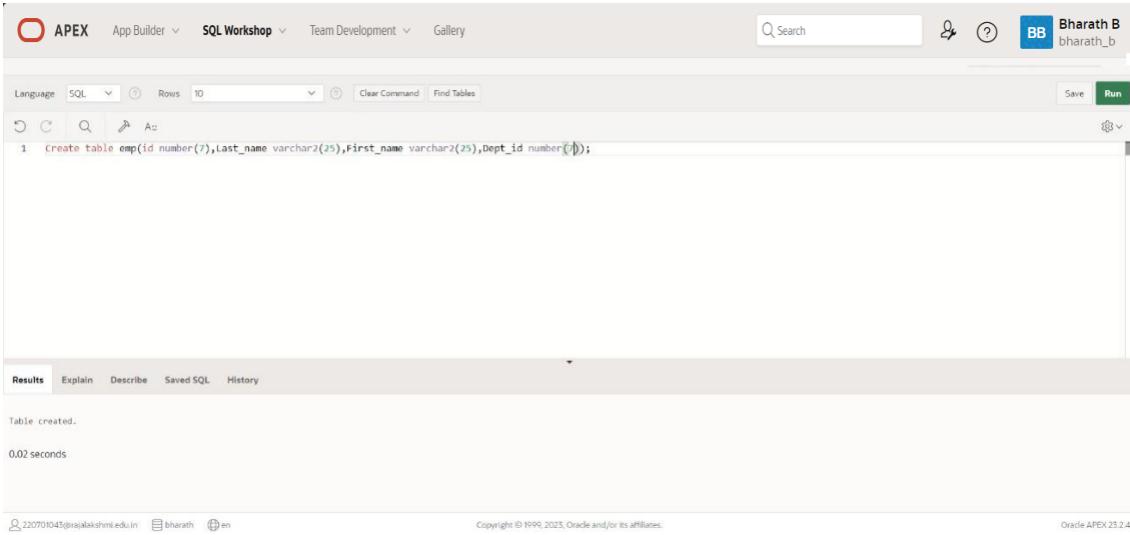
Column name	ID	LAST_NAME	FIRST_NAME	DEPARTMENT_ID
Key Type				
Nulls/Unique				
FK table				
FK column				

Data Type	Number	Varchar2	Varchar2	Number
Length	7	25	25	7

QUERY:

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

OUTPUT:



The screenshot shows the Oracle APEX SQL Workshop interface. In the top navigation bar, 'APEX' is selected, followed by 'App Builder', 'SQL Workshop' (which is currently active), 'Team Development', and 'Gallery'. On the right side, there is a user profile for 'Bharath B' (bharath_b). The main area has a toolbar with icons for Undo, Redo, Search, and Run. Below the toolbar, the SQL command is entered:

```
1 Create table emp(id number(7),Last_name varchar2(25),First_name varchar2(25),Dept_id number(7));
```

After running the command, the results show:

```
Table created.
```

Execution time: 0.02 seconds.

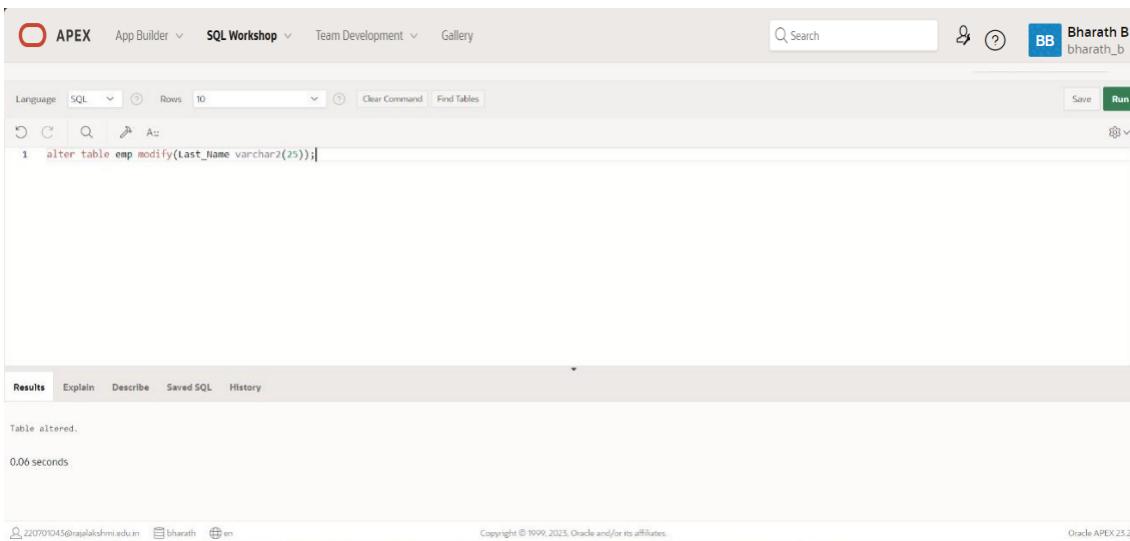
At the bottom, the session information includes the user (220701045@rajalakshmi.edu.in), the workspace (bharath), and the language (en). The copyright notice is 'Copyright © 1999, 2023, Oracle and/or its affiliates.' and the version is 'Oracle APEX 23.2.4'.

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

QUERY:

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

OUTPUT:



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

```
1 alter table emp modify(Last_Name varchar2(25));
```

After running the command, the results show:

```
Table altered.
```

Execution time: 0.06 seconds.

At the bottom, the session information includes the user (220701045@rajalakshmi.edu.in), the workspace (bharath), and the language (en). The copyright notice is 'Copyright © 1999, 2023, Oracle and/or its affiliates.' and the version is 'Oracle APEX 23.2.4'.

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

QUERY:

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

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. A search bar and user profile 'Bharath B bharath_b' are also present. The main workspace has tabs for Language (SQL selected), Rows (10), Clear Command, and Find Tables. The SQL editor contains the command: 'create table employees2(id number(7),first_name varchar2(25),Last_name varchar2(25),Salary int,Department_id number(7));'. Below the editor, the Results tab is active, showing the output: 'Table created.' and '0.03 seconds'. The bottom status bar displays the session ID '220701045@rajalakshmi.edu.in', the user 'bharath', and the locale 'en'. Copyright information and the version 'Oracle APEX 23.2.4' are also visible.

5.Drop the EMP table.

QUERY:

Drop table emp;

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. A search bar and user profile 'Bharath B bharath_b' are also present. The main workspace has tabs for Language (SQL selected), Rows (10), Clear Command, and Find Tables. The SQL editor contains the command: 'drop table emp;'. Below the editor, the Results tab is active, showing the output: 'Table dropped.' and '0.06 seconds'. The bottom status bar displays the session ID '220701045@rajalakshmi.edu.in', the user 'bharath', and the locale 'en'. Copyright information and the version 'Oracle APEX 23.2.4' are also visible.

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. The main area contains a SQL command: '1. Rename employees2 to emp;'. Below the command, the results show 'Statement processed.' and a duration of '0.06 seconds'. The bottom status bar indicates the session ID '230707045@gratulakshmi.edu.in' and the user 'Bharath B bharath_b'.

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

QUERY:

comment on table departments 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. The main area contains a SQL command: '1. comment on table dept is 'Department Info'';. Below the command, the results show 'Statement processed.' and a duration of '0.05 seconds'. The bottom status bar indicates the session ID '230707045@gratulakshmi.edu.in' and the user 'Bharath B bharath_b'.

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 APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. The main workspace has a search bar and a user profile for 'Bharath B'. Below the workspace are buttons for 'Save' and 'Run'. The SQL command entered is:

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

Under the 'Results' tab, the output shows:

```
Table altered.  
0.05 seconds
```

At the bottom, there are footer links for 'Copyright © 1999, 2023, Oracle and/or its affiliates.' and 'Oracle APEX 23.2.4'.

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

RESULT:

MANIPULATING DATA

EX_NO:2

DATE:22/2/24

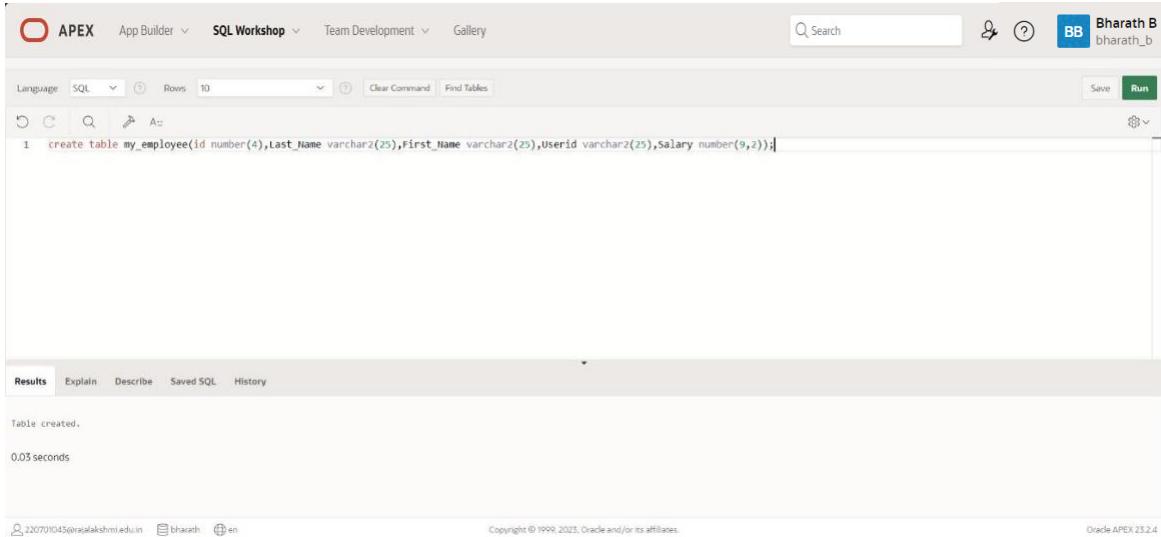
1.Create MY_EMPLOYEE table with the following structure

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 varchar2(25),First_Name varchar2(25),Userid varchar2(25),Salary number(9,2));

OUTPUT:



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

```
1 create table my_employee(id number(4),Last_Name varchar2(25),First_Name varchar2(25),Userid varchar2(25),Salary number(9,2));
```

After running the command, the results show:

Table created.

0.03 seconds

At the bottom, the footer includes:

220701045@rajalakshmi.edu.in Bharath 20 Copyright © 1999-2023, Oracle and/or its affiliates. Oracle APEX 23.2.4

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

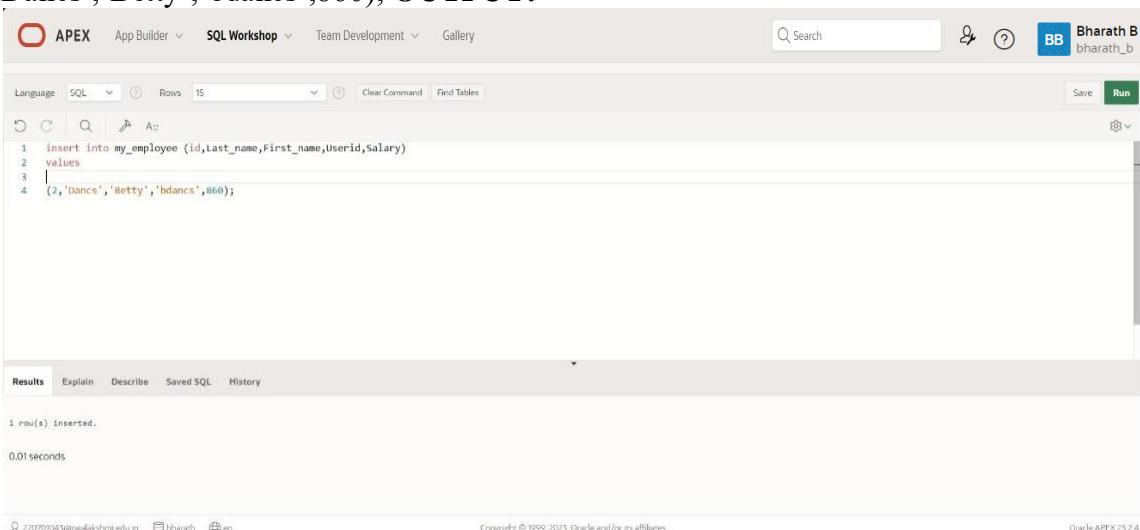
ID	Last_name	First_name	Userid	salary
1	Patel	Ralph	rpatel	895
2	Dancs	Betty	bdancs	860
3	Biri	Ben	bbiri	1100
4	Newman	Chad	Cnewman	750
5	Ropebur	Audrey	aropebur	1550

QUERY:

Insert into my_employee values(1,'Patel','Ralph','rpatel',895);

Insert into my_employee

values(1,'Dancs','Betty','bdancs',860); **OUTPUT:**



```
1 insert into my_employee (id,Last_name,First_name,Userid,salary)
2 values
3 |
4 (2,'dancs','Betty','bdancs',860);

1 row(s) inserted.

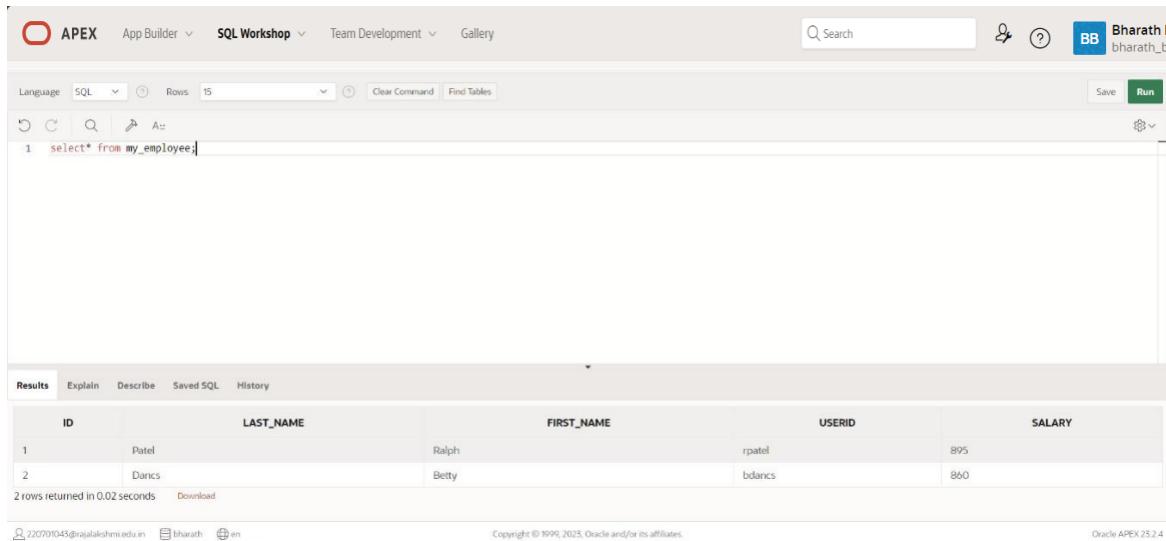
0.01 seconds
```

3. Display the table with values.

QUERY:

Select*from my_employee;

OUTPUT:



```
1 select* from my_employee;
```

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

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(1,'Patel','Ralph','rpatel',1100);
Insert into my_employee values(1,'Dancs','Betty','bdancs',750);
```

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 user icon, and a profile for 'Bharath B bharath_b'. Below the navigation is a toolbar with 'Language' (set to SQL), 'Rows' (set to 10), 'Clear Command', and 'Find Tables'. The main area contains the SQL command:

```
1 Insert into my_employee values(4,'Newman','chad','Cnewman',750);
```

Below the command, the results section shows:

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

Execution time: 0.01 seconds.

At the bottom, the URL is 220701045@rajalakshmi.edu.in, the session ID is bharath, and the page is en. Copyright information and Oracle APEX 23.2.4 are also visible.

5.Make the data additions permanent.

QUERY:

```
Select*from my_employee;
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface, similar to the previous one. The top navigation bar and user profile are the same. The main area contains the SQL command:

```
1 select*from my_employee;
```

Below the command, the results section displays the data from the my_employee table:

ID	LAST_NAME	FIRST_NAME	USERID	SALARY
1	Patel	Ralph	rpatel	1000
3	Drexler	Ben	bbri	1100
4	Newman	Chad	Cnewman	1000

3 rows returned in 0.04 seconds. There is a 'Download' link below the table.

At the bottom, the URL is 220701045@rajalakshmi.edu.in, the session ID is bharath, and the page is en. Copyright information and Oracle APEX 23.2.4 are also visible.

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

QUERY:

```
Update my_employee set last_name='Drexler' where id=3;
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (which is selected), Team Development, and Gallery. On the right, there's a search bar, a user icon for 'Bharath B bharath_b', and a 'Run' button. The main workspace has tabs for Language (SQL), Rows (10), Clear Command, and Find Tables. Below these are icons for Undo, Redo, Search, and Paste. The SQL command entered is: `1 update my_employee set last_name='Drexler' where id=3;`. The results section shows the output: `1 row(s) updated.` and `0.01 seconds`. At the bottom, it displays the user's email (220701043@rajalakshmi.edu.in), session ID (bharath), and locale (en). Copyright information for Oracle APEX 23.2.4 is also present.

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

QUERY:

```
update my_employee set salary=1000 where salary<900;
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface, identical to the previous one but with a different SQL command. The top navigation bar, workspace tools, and results section are the same. The SQL command entered is: `1 update my_employee set salary=1000 where salary<900;`. The results section shows the output: `3 row(s) updated.` and `0.01 seconds`. The bottom of the screen shows the user's email, session ID, locale, and copyright information for Oracle APEX 23.2.4.

8.Delete Betty dancs from MY _EMPLOYEE table.

QUERY:

```
delete from my_employee where last_name='Dancs';
```

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, and a button for 'Bharath B bharath_b'. The main workspace has tabs for Language (SQL selected), Rows (10), Clear Command, and Find Tables. Below these are icons for Undo, Redo, Search, and Sort. The SQL command entered is: `1 delete from my_employee where last_name='Dancs';`. The results section shows the output: `1 row(s) deleted.` and `0.01 seconds`. The bottom footer includes copyright information for Oracle and the APEX version.

9.Empty the fourth row of the emp table.

QUERY:

```
Delete from emp where id=4;
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. On the right, there's a search bar, a user icon, and a button for 'Bharath B bharath_b'. The main workspace has tabs for Language (SQL selected), Rows (10), Clear Command, and Find Tables. Below these are icons for Undo, Redo, Search, and Sort. The SQL command entered is: `1 delete from emp where id=4;`. The results section shows the output: `0 row(s) deleted.` and `0.03 seconds`. The bottom footer includes copyright information for Oracle and the APEX version.

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

RESULT:

INCLUDING CONSTRAINTS

EX_NO:3

DATE:29/2/24

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_employee_id_pk.

QUERY:

```
alter table emp add constraint my_employee_id_pk primary key(id);
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. The right side shows a user profile for 'Bharath B bharath_b'. The main workspace has tabs for 'Language' (set to 'SQL'), 'Rows' (set to 10), 'Clear Command', and 'Find Tables'. Below these are icons for 'Run', 'Save', and 'Help'. The SQL command entered is: `1 alter table emp add constraint my_employee_id_pk primary key(id);`. The results tab shows the output: `Table altered.` and `0.07 seconds`. The bottom footer includes copyright information for Oracle and the APEX version.

2.Create a PRIMAY KEY constraint to the DEPARTMENTS table using the ID colum. The constraint should be named at creation. Name the constraint my_department_id_pk.

QUERY:

```
Alter table emp add constraint my_department_id_pk primary key(id);
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. The right side shows a user profile for 'Bharath B bharath_b'. The main workspace has tabs for 'Language' (set to 'SQL'), 'Rows' (set to 10), 'Clear Command', and 'Find Tables'. Below these are icons for 'Run', 'Save', and 'Help'. The SQL command entered is: `1 alter table dept add constraint my_dept_id_pk primary key(id);`. The results tab shows the output: `Table altered.` and `0.07 seconds`. The bottom footer includes copyright information for Oracle and the APEX version.

3.Add a column DEPARTMENT_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_department_id_fk.

QUERY:

```
alter table emp add constraint my_emp_department_id_fk foreign key(department_id) references 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. On the right, there's a search bar, a help icon, and a user profile for 'Bharath B bharath_b'. The main workspace has tabs for Language (SQL selected), Rows (10), Clear Command, and Find Tables. Below these are standard edit and search tools. The SQL editor contains the command: `1 alter table emp add constraint my_emp_dept_id_fk foreign key(dept_id) references emp(id);`. The results tab shows the output: `Table altered.`, `0.06 seconds`, and the timestamp `220701043@rajalakshmi.edu.in bharath en`. The bottom footer includes copyright information for Oracle and the APEX version `Oracle APEX 23.2.4`.

```
1 alter table emp add constraint my_emp_dept_id_fk foreign key(dept_id) references emp(id);
```

Table altered.
0.06 seconds

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

QUERY:

```
Alter table emp add constraint number(2,2) check(commission);
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface, identical to the previous one but with a different query. The top navigation bar, workspace tools, and results tab are the same. The SQL editor contains the command: `1 alter table emp add commission number(2,2) check(commission>0);`. The results tab shows the output: `Table altered.`, `0.06 seconds`, and the timestamp `220701043@rajalakshmi.edu.in bharath en`. The bottom footer includes copyright information for Oracle and the APEX version `Oracle APEX 23.2.4`.

```
1 alter table emp add commission number(2,2) check(commission>0);
```

Table altered.
0.06 seconds

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

RESULT:

Writing Basic SQL SELECT Statements

EX_NO:4

DATE:2/3/24

- 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, sal*12 as "ANNUAL SALARY" from employees;
```

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

QUERY:

```
Desc departments;
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there are icons for Search, Refresh, Help, and a user profile for 'Bharath B bharath_b'. The main workspace has a toolbar with various icons. The SQL editor contains the command 'desc dept;'. Below the editor, the results tab is selected, showing the structure of the 'DEPT' table. The table has two columns: 'ID' (NUMBER type) and 'NAME' (VARCHAR2 type). The 'ID' column is defined as Primary Key (PK) and Nullable (N). The 'NAME' column is nullable (N). At the bottom of the results pane, there are links for Results, Explain, Saved SQL, and History, along with a 'Describe' button. The status bar at the bottom displays the URL '220701045@rajalakshmi.edu.in', the session name 'bharath', and the language 'en'. It also includes copyright information 'Copyright © 1999, 2023, Oracle and/or its affiliates.' and the version 'Oracle APEX 23.2.4'.

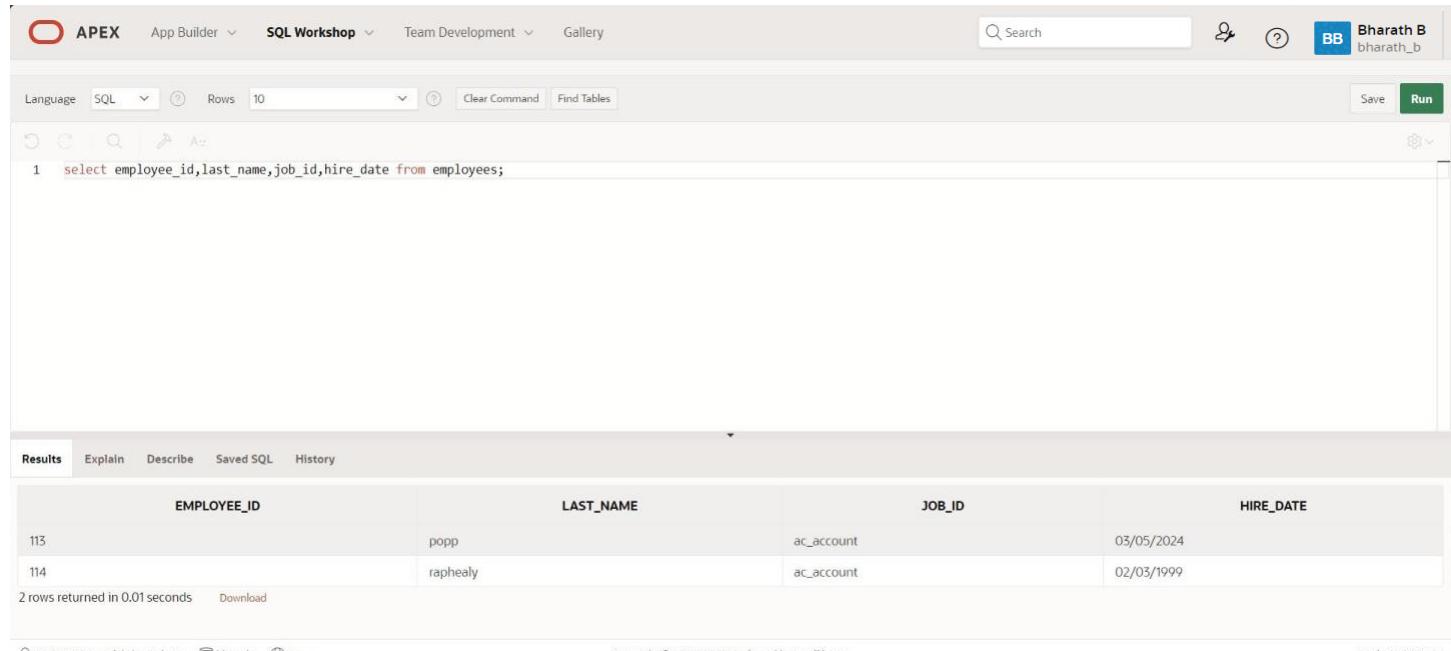
Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
DEPT	ID	NUMBER	-	7	0	1	-	-	-
	NAME	VARCHAR2	25	-	-	-	✓	-	-

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

QUERY:

Select employee_id, last_name, job_id, hire_date from employees;

OUTPUT:



The screenshot shows the Oracle 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 connection named 'Bharath B bharath_b'. Below the toolbar, the 'Language' dropdown is set to 'SQL', and the 'Rows' dropdown is set to '10'. There are buttons for 'Save' and 'Run'. The main workspace contains the SQL command: 'select employee_id, last_name, job_id, hire_date from employees;'. The results section shows the following data:

EMPLOYEE_ID	LAST_NAME	JOB_ID	HIRE_DATE
113	popp	ac_account	03/05/2024
114	raphealy	ac_account	02/03/1999

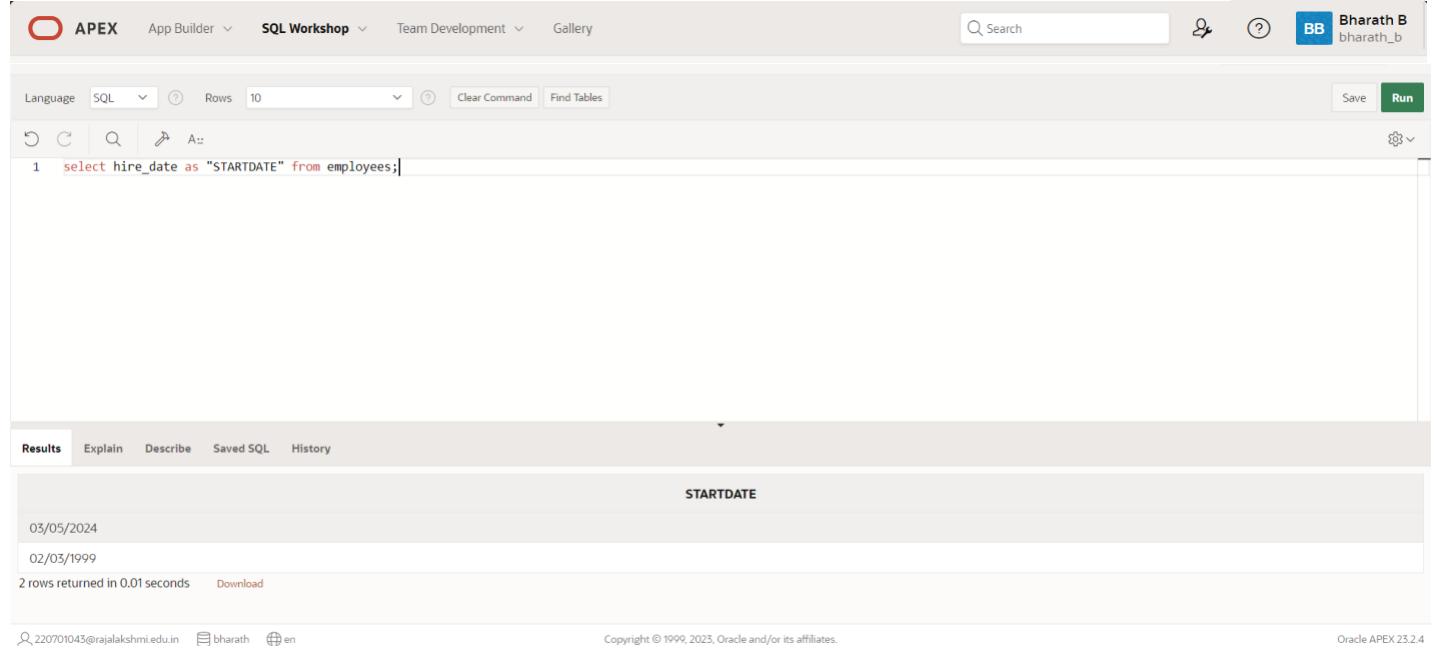
Below the table, it says '2 rows returned in 0.01 seconds' and has a 'Download' link. At the bottom, it shows the user information '220701043@rajalakshmi.edu.in bharath en' and the copyright notice 'Copyright © 1999,2023, Oracle and/or its affiliates.' followed by 'Oracle APEX 25.2.4'.

4. Provide an alias STARTDATE for the hire date.

QUERY:

select hire_date as "STARTDATE" from employees;

OUTPUT:



The screenshot shows the Oracle APEX SQL Workshop interface. The 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 connection named 'Bharath B bharath_b'. Below the toolbar, the 'Language' dropdown is set to 'SQL', and the 'Rows' dropdown is set to '10'. There are buttons for 'Save' and 'Run'. The main workspace contains the SQL command: 'select hire_date as "STARTDATE" from employees;'. The results section shows the following data:

STARTDATE
03/05/2024
02/03/1999

Below the table, it says '2 rows returned in 0.01 seconds' and has a 'Download' link. At the bottom, it shows the user information '220701043@rajalakshmi.edu.in bharath en' and the copyright notice 'Copyright © 1999,2023, Oracle and/or its affiliates.' followed by 'Oracle APEX 25.2.4'.

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

QUERY:

```
select distinct job_id from employees;
```

OUTPUT:

The screenshot shows the Oracle APEX interface with the SQL Workshop module selected. The query `select distinct job_id from employees;` is entered in the command line. The results show one row with the value `ac_account`.

JOB_ID
ac_account

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

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

QUERY:

```
Select last_name||', '||job_id as "EMPLOYEE_AND_TITLE" from employees;
```

OUTPUT:

The screenshot shows the Oracle APEX interface with the SQL Workshop module selected. The query `Select last_name||', '||job_id as "EMPLOYEE_AND_TITLE" from employees;` is entered in the command line. The results show two rows with values `popp,ac_account` and `raphealy,ac_account`.

EMPLOYEE_AND_TITLE
popp,ac_account
raphealy,ac_account

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

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

QUERY:

Select

```
employee_id||','||first_name||','||last_name||','||email||','||phone_no||','||hire_date||','||job_id||','||salary||','||commission_pct||','||manager_id||','||department_id as THE_OUTPUT from employees;
```

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 for 'Bharath B bharath_b', and a 'Run' button. Below the toolbar, the SQL editor shows the query:

```
1 select employee_id||','||first_name||','||last_name||','||email||','||phone_no||','||hire_date||','||job_id||','||salary||','||commission_pct||','||manager_id||','||department_id as THE_OUTPUT from employees;
```

The results tab is selected, displaying the output:

THE_OUTPUT
113,louis,popp,lpopp,5151244567,03/05/2024,ac_account,6900,,205,100
114,den,radeheal,drapheal,5151274561,02/03/1999,ac_account,11000,,100,30

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

At the bottom, it shows the URL <https://220701043@rajalakshmi.edu.in>, the user [bharath](#), and the language [en](#). Copyright information and the Oracle APEX version (25.2.4) are also present.

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

RESULT:

RESTRICTING AND SORTING DATA

EX_NO:5

DATE: 07-03-2024

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

QUERY:

Select last_name from employees where salary>12000;

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. On the right, there's a search bar, a help icon, and a user profile for 'Bharath B bharath_b'. The main workspace has tabs for 'Language' (set to 'SQL'), 'Rows' (set to 10), 'Clear Command', and 'Find Tables'. Below this is a code editor with the following SQL command:

```
1 select last_name,salary from employees where salary>12000;
```

Below the code editor is a results panel with tabs for 'Results' (selected), 'Explain', 'Describe', 'Saved SQL', and 'History'. The results section displays the message 'no data found'.

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

QUERY:

Select last_name,department_id from employees where employee_id=176;

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface, identical to the previous one but with a different query. The top navigation bar and workspace are the same. The code editor contains the following SQL command:

```
1 select last_name,department_id from employees where employee_id=176;
```

The results panel shows 'no data found'.

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 employees where salary not between 5000 and 12000;
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there's a search bar, a help icon, and a user profile for 'Bharath B' (bharath_b). The main workspace has tabs for Language (SQL selected), Rows (10), Clear Command, and Find Tables. Below is a code editor with the following SQL command:

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

Under the Results tab, it says "no data found". At the bottom, there are footer links for 220701043@rajalakshmi.edu.in, bharath, en, Copyright © 1999, 2023, Oracle and/or its affiliates, and Oracle APEX 23.2.4.

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

QUERY:

```
Select last_name,job_id,hire_date from employees where hire_date between  
'February,20,1998' and 'May,1,1998';
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there's a search bar, a help icon, and a user profile for 'Bharath B' (bharath_b). The main workspace has tabs for Language (SQL selected), Rows (10), Clear Command, and Find Tables. Below is a code editor with the following SQL command:

```
1 select last_name,job_id,hire_date from employees where hire_date between 'February,20,1998' and 'May,1,1998';
```

Under the Results tab, the output is displayed in a table:

LAST_NAME	JOB_ID	HIRE_DATE
Mohan	ac_account	02/22/1998

At the bottom, it says "1 rows returned in 0.03 seconds" and provides a "Download" link. The footer includes links for 220701043@rajalakshmi.edu.in, bharath, en, Copyright © 1999, 2023, Oracle and/or its affiliates, and Oracle APEX 23.2.4.

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

QUERY:

```
select last_name,department_id from employees where department_id in(20,50) order by last_name;
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. The right side shows a user profile for 'Bharath B' (bharath_b). The main area has tabs for 'Language' (set to 'SQL'), 'Rows' (set to 10), 'Clear Command', and 'Find Tables'. Below these are icons for 'Run', 'Save', and 'Run' (green button). The SQL command entered is:

```
1 select last_name,department_id from employees where department_id in(20,50) order by last_name;
```

The results section shows a single row:

LAST_NAME	DEPARTMENT_ID
raphealy	20

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

At the bottom, the footer includes the URL '220701043@rajalakshmi.edu.in', the name 'bharath', and language 'en'. It also states '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 employees where (salary between 5000 and 12000) and (department_id 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'. The right side shows a user profile for 'Bharath B' (bharath_b). The main area has tabs for 'Language' (set to 'SQL'), 'Rows' (set to 10), 'Clear Command', and 'Find Tables'. Below these are icons for 'Run', 'Save', and 'Run' (green button). The SQL command entered is:

```
1 select last_name as "EMPLOYEE",salary as "MONTHLY SALARY" from employees where (salary between 5000 and 12000) and (department_id in(20,50)) order by last_name asc;
```

The results section shows a single row:

EMPLOYEE	MONTHLY SALARY
raphealy	11000

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

At the bottom, the footer includes the URL '220701043@rajalakshmi.edu.in', the name 'bharath', and language 'en'. It also states 'Copyright © 1999, 2023, Oracle and/or its affiliates.' and '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 employees where hire_date like '1994%';
```

OUTPUT:

A screenshot of the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The search bar contains the placeholder "Search". On the right, there's a user profile for "Bharath B" (bharath_b) with a blue icon. The main workspace shows a SQL command: "1 select last_name,hire_date from employees where hire_date like '1994';". Below the command, the results tab is selected, showing the message "no data found". The bottom status bar displays the session ID "220701043@rajalakshmi.edu.in", the user "bharath", and the language "en". It also includes copyright information for Oracle and the APEX version "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 employees where manager_id is null;
```

OUTPUT:

A screenshot of the Oracle APEX SQL Workshop interface, identical to the previous one but with a different query. The workspace shows the command: "1 select last_name,job_id from employees where manager_id is null;". The results tab indicates "no data found". The bottom status bar shows the session ID "220701043@rajalakshmi.edu.in", the user "bharath", and the language "en". Copyright information and the APEX version "Oracle APEX 23.2.4" are also present.

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

QUERY:

```
select last_name,salary,commission_pct from employees where commission_pct is not null  
order by salary desc;
```

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 help icon, and a user profile for 'Bharath B bharath_b'. The main workspace has tabs for Language (SQL), Rows (10), Clear Command, and Find Tables. Below these are icons for Refresh, Undo, Redo, Search, and Run. The SQL command entered is: `1 select last_name,salary,commission_pct from employees where commission_pct is not null order by salary desc;`. The results tab is selected, showing the message 'no data found'. At the bottom, there are footer links for 220701043@irajalakshmi.edu.in, bharath, en, Copyright © 1999, 2023, Oracle and/or its affiliates, and 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 from employees where last_name like '__a%';
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. On the right, there's a search bar, a help icon, and a user profile for 'Bharath B bharath_b'. The main workspace has tabs for Language (SQL), Rows (10), Clear Command, and Find Tables. Below these are icons for Refresh, Undo, Redo, Search, and Run. The SQL command entered is: `1 select last_name from employees where last_name like '__a%';`. The results tab is selected, showing the message 'no data found'. At the bottom, there are footer links for 220701043@irajalakshmi.edu.in, bharath, en, Copyright © 1999, 2023, Oracle and/or its affiliates, and Oracle APEX 23.2.4.

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 employees where last_name like '%a%' and last_name like '%e%';
```

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 'Bharath B' (bharath_b), and a 'Run' button. The main workspace shows a SQL command window with the following content:

```
1 select last_name from employees where last_name like '%a%' and last_name like '%e%';
```

The results pane shows a single row of data:

LAST_NAME
raphealy

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

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

OUTPUT:

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

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

The results pane shows 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:

APEX App Builder SQL Workshop Team Development Gallery

Language SQL Rows 10 Clear Command Find Tables Save Run

Results Explain Describe Saved SQL History

no data found

220701043@rajalakshmi.edu.in bharath en Copyright © 1999, 2023, Oracle and/or its affiliates. Oracle APEX 23.2.4

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

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

RESULT:

SINGLE ROW FUNCTIONS

EX_NO:6

DATE:09-03-2024

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

QUERY:

```
select sysdate from dual;
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. The right side shows a user profile for 'Bharath B' (bharath_b). The main workspace has a toolbar with icons for refresh, search, and run. Below the toolbar, the SQL command is entered: 'select sysdate from dual;'. The results tab is selected, showing the output: '03/10/2024'. Below the results, it says '1 rows returned in 0.01 seconds' and there is a 'Download' link. At the bottom, the footer includes the URL '220701043@rajalakshmi.edu.in', the name 'bharath', 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. The HR department needs a report to display the employee number, last name, salary, and increased by 15.5% (expressed as a whole number) for each employee. Label the column New Salary.

QUERY:

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

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 'Bharath B' (bharath_b). The main workspace has a toolbar with icons for refresh, search, and run. Below the toolbar, the SQL command is entered: 'select employee_id, last_name, salary, salary+(15.5/100*salary) "new_salary" from employees;'. The results tab is selected, showing a table with columns: 'EMPLOYEE_ID', 'LAST_NAME', 'SALARY', and 'new_salary'. The data rows are: (113, popp, 6900, 7969.5), (114, raphealy, 11000, 12705), and (2, Mohan, 4000, 4620). Below the results, it says '3 rows returned in 0.01 seconds' and there is a 'Download' link. At the bottom, the footer includes the URL '220701043@rajalakshmi.edu.in', the name 'bharath', 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.

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 employees;
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The query has been run, and the results are displayed in a table. The table has columns: EMPLOYEE_ID, LAST_NAME, SALARY, new_salary, and Increase. The data shows three rows of employees with their names, salaries, and calculated values.

EMPLOYEE_ID	LAST_NAME	SALARY	new_salary	Increase
113	popp	6900	7969.5	1069.5
114	raphealy	11000	12705	1705
2	Mohan	4000	4620	620

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

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The query has been run, and the results are displayed in a table. The table has columns: INITCAP(LAST_NAME) and Length_of_last_name. The data shows one row of an employee named Mohan, with the last name processed according to the specified rules.

INITCAP(LAST_NAME)	Length_of_last_name
Mohan	5

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

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

QUERY:

```
select initcap(last_name) "Name",length(last_name) "Length of last name" from employees  
where last_name like 'H%' order by last_name;
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. On the right, there's a search bar, a help icon, and a session header for 'Bharath B bharath_b'. Below the navigation is a toolbar with icons for refresh, search, and run. The main area has tabs for 'Language', 'SQL', 'Rows', 'Clear Command', and 'Find Tables'. A code editor window contains the following SQL query:

```
1 select initcap(last_name) "Name",length(last_name) "Length of last name" from employees where last_name like 'H%' order by last_name;
```

Below the code editor is a results grid with two columns: 'Name' and 'Length of last name'. A single row is displayed for the employee 'Hari', with a value of 4 in the 'Length of last name' column. At the bottom left, it says '1 rows returned in 0.03 seconds'. At the bottom right, it shows 'Copyright © 1999, 2023, Oracle and/or its affiliates.' and 'Oracle APEX 23.2.4'.

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface, identical to the previous one but with a different query. The code editor now contains:

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

The results grid shows the 'LAST_NAME' and 'MONTHS_WORKED' columns. Three rows are listed: 'popp' with 0 months, 'raphealy' with 306 months, and 'Mohan' with 317 months. The bottom status bar indicates '3 rows returned in 0.01 seconds'.

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

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

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

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 'Bharath B bharath_b'. The main area has tabs for 'Language', 'SQL' (selected), 'Rows' (set to 10), 'Clear Command', and 'Find Tables'. Below these are icons for refresh, search, and run. The SQL command entered is:

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

The results tab is selected, showing the output:

DREAM_SALARIES	
popp	earns 6900 monthly but wants 20700
raphealy	earns 11000 monthly but wants 33000
Mohan	earns 4000 monthly but wants 12000

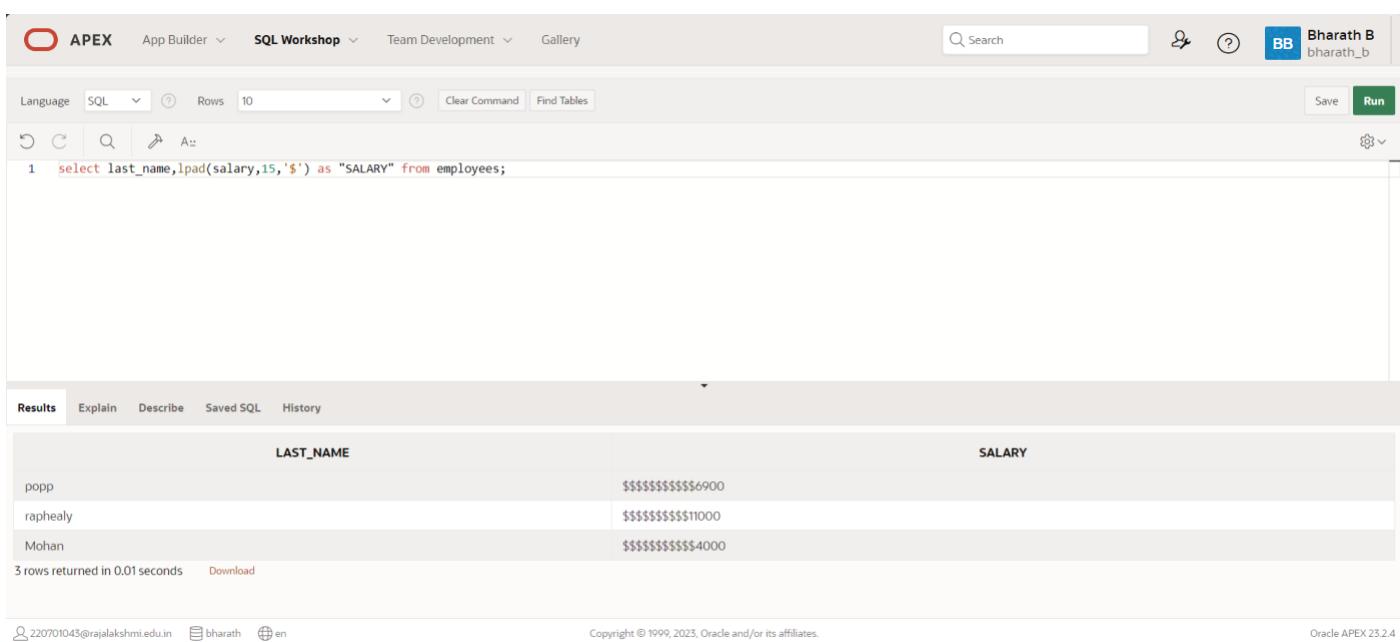
3 rows returned in 0.00 seconds. There are download and refresh buttons at the bottom.

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

QUERY:

select last_name, lpad(salary,15,'\$') as "SALARY" from employees;

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 'Bharath B bharath_b'. The main area has tabs for 'Language', 'SQL' (selected), 'Rows' (set to 10), 'Clear Command', and 'Find Tables'. Below these are icons for refresh, search, and run. The SQL command entered is:

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

The results tab is selected, showing the output:

LAST_NAME	SALARY
popp	\$\$\$\$\$\$\$\$\$\$6900
raphealy	\$\$\$\$\$\$\$\$\$\$11000
Mohan	\$\$\$\$\$\$\$\$\$\$4000

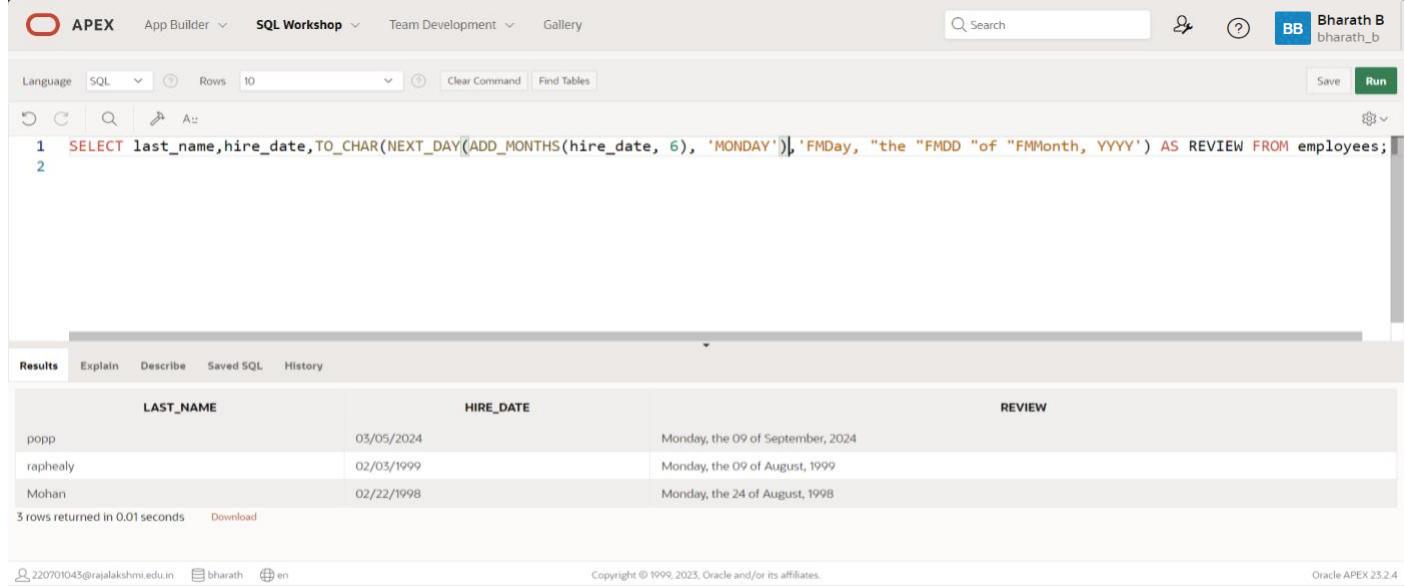
3 rows returned in 0.01 seconds. There are download and refresh buttons at the bottom.

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

QUERY:

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

OUTPUT:



The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there is a user profile for 'Bharath B bharath_b'. The main area has tabs for Language (SQL selected), Rows (10), Clear Command, Find Tables, Save, and Run. The SQL command entered is:

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

The Results tab displays the output:

LAST_NAME	HIRE_DATE	REVIEW
popp	03/05/2024	Monday, the 09 of September, 2024
raphealy	02/03/1999	Monday, the 09 of August, 1999
Mohan	02/22/1998	Monday, the 24 of August, 1998

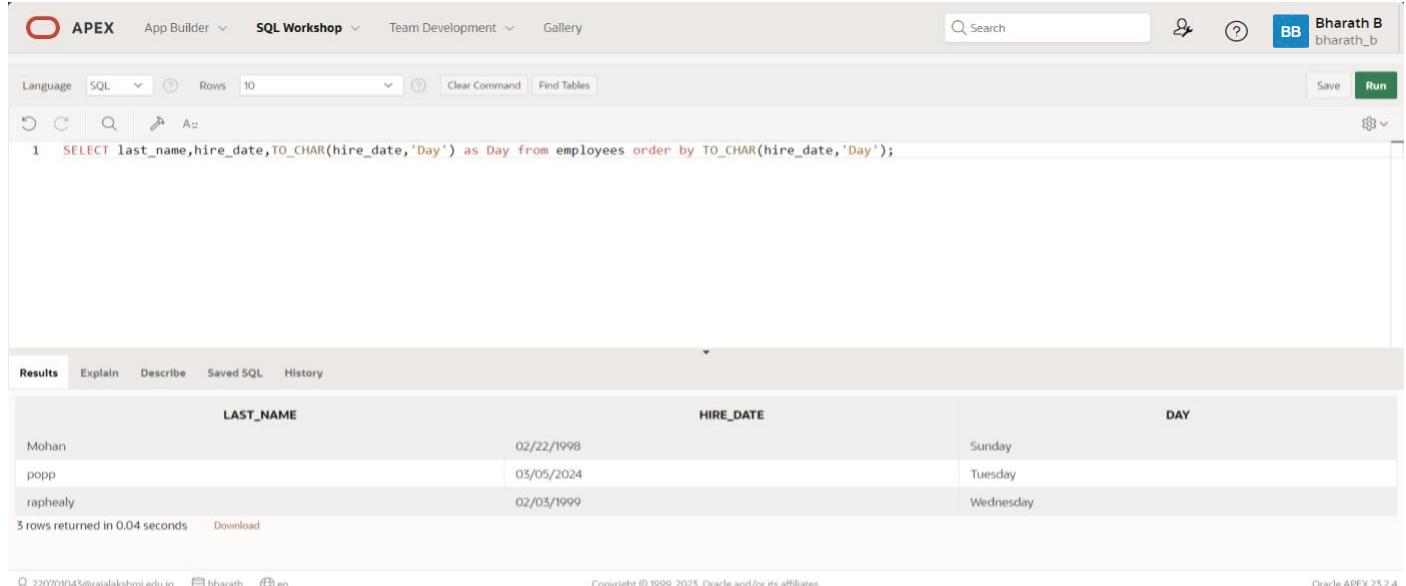
3 rows returned in 0.01 seconds. The bottom status bar shows the session ID 220701045@rajalakshmi.edu.in, the user bharath, and the locale en.

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

QUERY:

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

OUTPUT:



The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there is a user profile for 'Bharath B bharath_b'. The main area has tabs for Language (SQL selected), Rows (10), Clear Command, Find Tables, Save, and Run. The SQL command entered is:

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

The Results tab displays the output:

LAST_NAME	HIRE_DATE	DAY
Mohan	02/22/1998	Sunday
popp	03/05/2024	Tuesday
raphealy	02/03/1999	Wednesday

3 rows returned in 0.04 seconds. The bottom status bar shows the session ID 220701045@rajalakshmi.edu.in, the user bharath, and the locale en.

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

RESULT:

DISPLAYING DATA FROM MULTIPLE TABLES

EX_NO:7

DATE:

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

QUERY:

```
Select e.last_name,e.department_id,d.department_id from employees e,departments d  
where e.department_id=d.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 search bar, a user icon, and a session header 'Bharath B bharath_b'. Below the toolbar, the SQL command is displayed:

```
1 select e.last_name,e.department_id,d.Dept_name from employees e,departments d ;
```

The results section shows the output of the query:

LAST_NAME	DEPARTMENT_ID	DEPT_NAME
Janu	100	Public Relations
HARI	20	Public Relations
Mohan	150	Public Relations
Janu	100	finance
HARI	20	finance
Mohan	150	finance

Below the table, it says '6 rows returned in 0.01 seconds'.

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 employees e,departments d where e.department_id=d.department_id  
and e.department_id=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 search bar, a user icon, and a session header 'Bharath B bharath_b'. Below the toolbar, the SQL command is displayed:

```
1 select distinct job_id,LOCATION_ID FROM EMPLOYEES,DEPARTMENTS where employees.department_id=departments.dept_id and employees.department_id=80;
```

The results section shows the output of the query:

JOB_ID	LOCATION_ID
ac_account	4598

Below the table, it says '1 rows returned in 0.02 seconds'.

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_id,d.dept_name,d.loc_id,l.city from employees e,departments d,location l  
where e.department_id=d.department_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'. The right side shows a user profile for 'Bharath B' (bharath_b). The main workspace has a search bar and a toolbar with icons for refresh, search, and run. The SQL command entered is:

```
1 select e.last_name,d.dept_name,d.location_id,l.city from employees e,departments d,location l where e.department_id=d.department_id and d.location_id=l.location_id and e.commission_pct is not null;
```

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

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

QUERY:

```
Select employees.last_name,departments.dept_name from employees,departments  
where employees.department_id=departments.department_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'. The right side shows a user profile for 'Bharath B' (bharath_b). The main workspace has a search bar and a toolbar with icons for refresh, search, and run. The SQL command entered is:

```
1 select last_name,dept_name from employees,departments where employees.department_id=departments.department_id and last_name like '%a%';
```

The results tab is selected, displaying the output:

LAST_NAME	DEPT_NAME
Janu	Public Relations

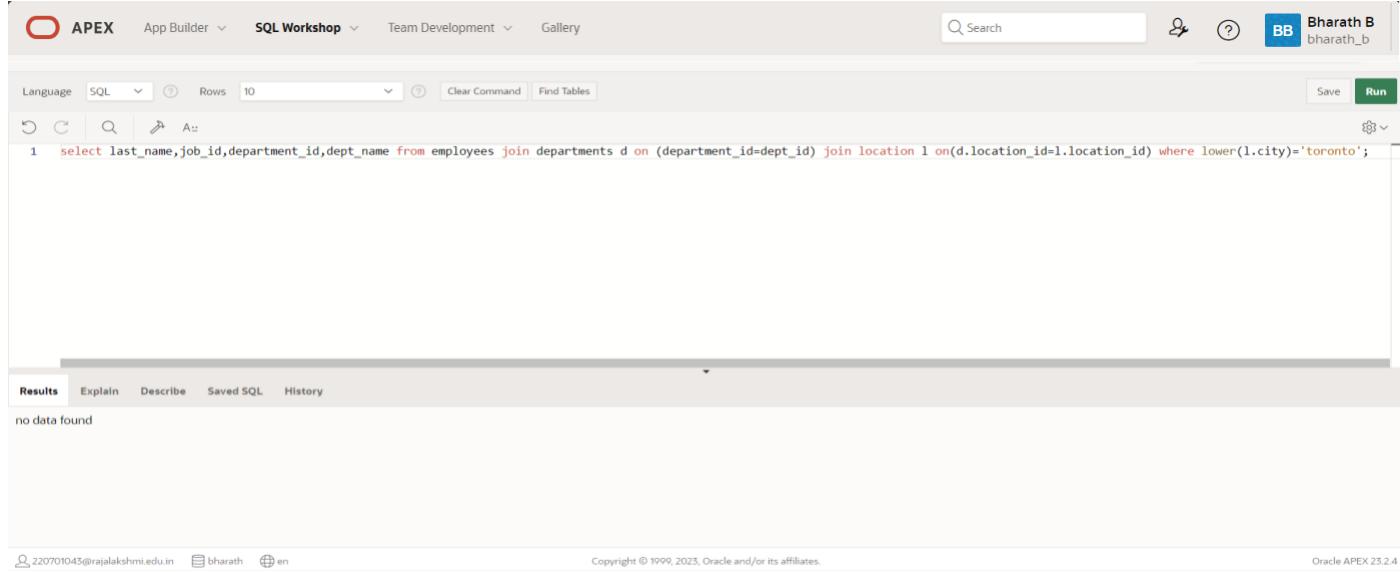
1 rows returned in 0.01 seconds [Download](#)

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

QUERY:

```
Select e.last_name,e.department_id,e.job_id,d.dept_name from employees e join departments d  
on(e.department_id=d.department_id) join location on (d.location_id=location.location_id) where  
lower(location.city)=’toronto’;
```

OUTPUT:



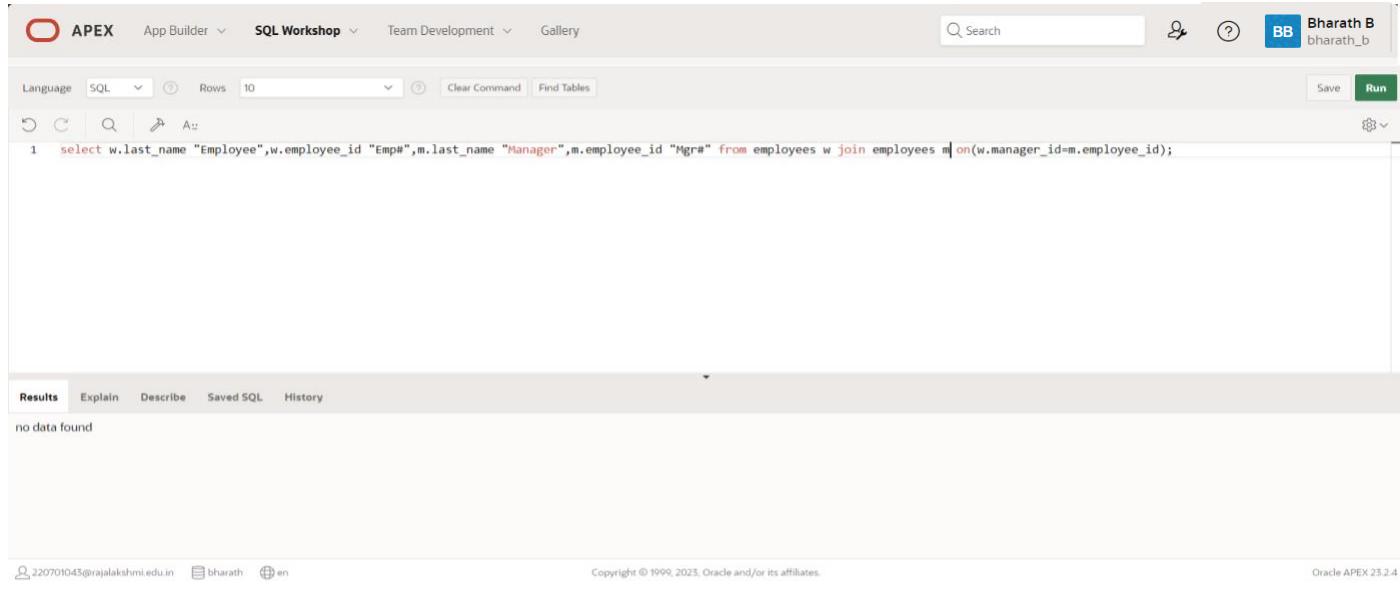
The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there's a user profile for 'Bharath B' (bharath_b). The main workspace has tabs for Language (SQL selected), Rows (10), Clear Command, and Find Tables. Below these are standard database navigation icons (refresh, search, etc.). The SQL editor contains the query provided above. The results tab is active, showing the message 'no data found'. At the bottom, the footer includes the user's email (220701045@rajalakshmi.edu.in), session ID (bharath), and language (en). It also credits Oracle and mentions 'Oracle APEX 23.2.4'.

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

QUERY:

```
Select w.last_name “Employee”,w.employee_id “emp#”,m.last_name ‘manager’,m.employee_id “Mgr#”  
from employees m on (w.manager_id=m.employee_id);
```

OUTPUT:



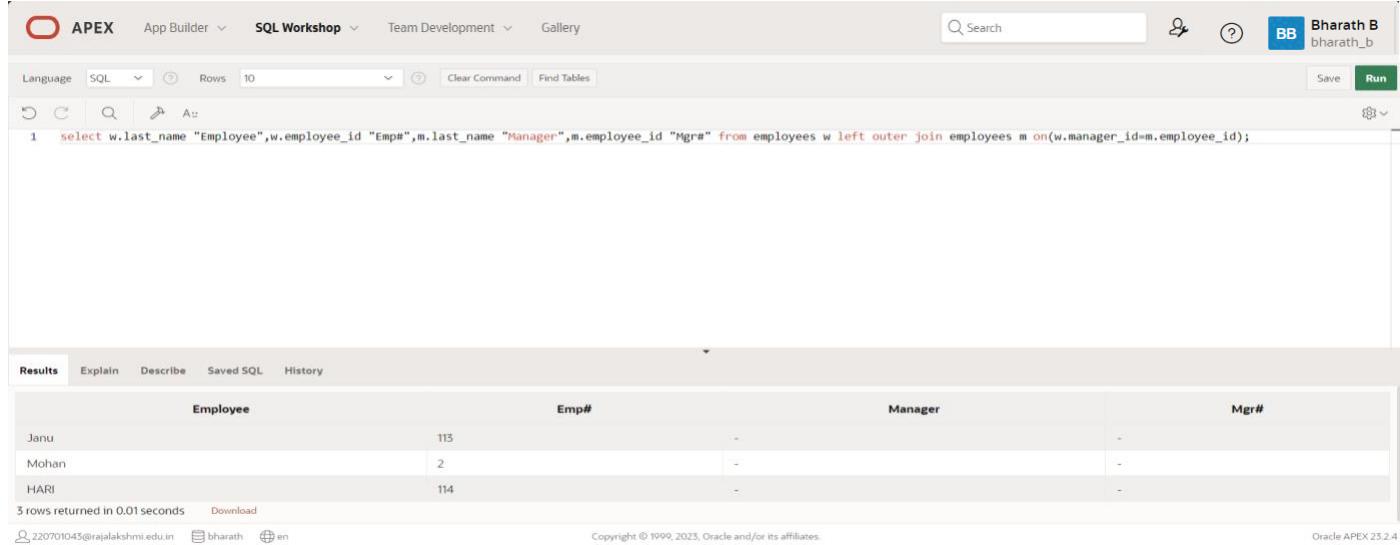
The screenshot shows the Oracle APEX SQL Workshop interface, identical to the previous one but with a different query. The top navigation bar, user profile, and workspace layout are the same. The SQL editor now contains the query provided above. The results tab is active, showing the message 'no data found'. The footer information (email, session ID, language, and Oracle APEX version) is also present at the bottom.

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

QUERY:

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

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 'Bharath B bharath_b'. The main area has tabs for 'Language', 'SQL', 'Rows', '10', 'Clear Command', and 'Find Tables'. Below these are icons for 'Search', 'Run', and other utilities. The SQL command entered is:

```
1 select w.last_name "Employee",w.employee_id "Emp#",m.last_name "Manager",m.employee_id "Mgr#" from employees w left outer join employees m on(w.manager_id=m.employee_id);
```

The results section shows a table with four columns: Employee, Emp#, Manager, and Mgr#. The data returned is:

Employee	Emp#	Manager	Mgr#
Janu	113	-	-
Mohan	2	-	-
HARI	114	-	-

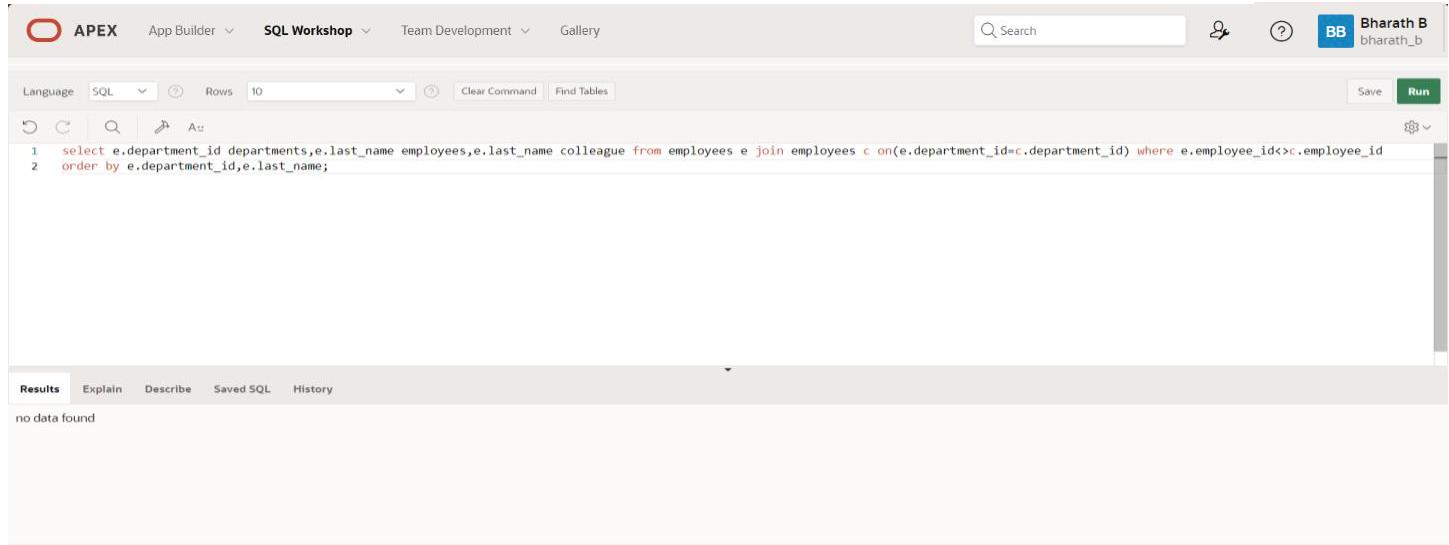
Below the table, it says '3 rows returned in 0.01 seconds' and has 'Download' and 'History' buttons. At the bottom, it shows the URL '220701045@rajalakshmi.edu.in', the session ID 'bhарат', and the language 'en'. Copyright information 'Copyright © 1999, 2023, Oracle and/or its affiliates.' and the version 'Oracle APEX 23.2.4' are also present.

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_id departments,e.last_name colleague from employees e join employees c on (e.department_id=c.department_id) where e.employee_id <> c.employee_id order by e.department_id,e.last_name,c.last_name;
```

OUTPUT:



The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. On the right, there's a user profile for 'Bharath B bharath_b'. The main area has tabs for 'Language', 'SQL', 'Rows', '10', 'Clear Command', and 'Find Tables'. Below these are icons for 'Search', 'Run', and other utilities. The SQL command entered is:

```
1 select e.department_id departments,e.last_name employees,e.last_name colleague from employees e join employees c on(e.department_id=c.department_id) where e.employee_id <> c.employee_id
2 order by e.department_id,e.last_name;
```

The results section shows the message 'no data found'.

At the bottom, it shows the URL '220701045@rajalakshmi.edu.in', the session ID 'bhарат', 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.

9. Show the structure of the JOB_GRADES table. Create a query that displays the name, job,

department name, salary, and grade for all employees

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. On the right, there's a search bar, a user icon for 'Bharath B bharath_b', and 'Save' and 'Run' buttons. The main area has tabs for 'Language' (set to 'SQL'), 'Rows' (set to 10), 'Clear Command', and 'Find Tables'. Below this is a code editor with the query from the previous step. The 'Results' tab is selected, displaying a table with two rows of data. The columns are 'LAST_NAME', 'JOB_ID', 'DEPT_NAME', 'SALARY', and 'GRADE_LEVEL'. The data shows two entries for 'Janu' with 'AC_ACCOUNT' as the job and 'Public Relations' as the department, both having a salary of 6900 and a grade level of 'C'. At the bottom, it says '2 rows returned in 0.02 seconds' and has a 'Download' link.

LAST_NAME	JOB_ID	DEPT_NAME	SALARY	GRADE_LEVEL
Janu	AC_ACCOUNT	Public Relations	6900	B
Janu	AC_ACCOUNT	Public Relations	6900	C

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

QUERY:

```
SELECT e.last_name, e.hire_date
FROM employees e, employees 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 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. On the right, there's a search bar, a user icon for 'Bharath B bharath_b', and 'Save' and 'Run' buttons. The main area has tabs for 'Language' (set to 'SQL'), 'Rows' (set to 10), 'Clear Command', and 'Find Tables'. Below this is a code editor with the query from the previous step. The 'Results' tab is selected, displaying a table with one row of data. The columns are 'LAST_NAME' and 'HIRE_DATE'. The data shows 'Janu' with a hire date of '03/05/2024'. At the bottom, it says '1 rows returned in 0.01 seconds' and has a 'Download' link.

LAST_NAME	HIRE_DATE
Janu	03/05/2024

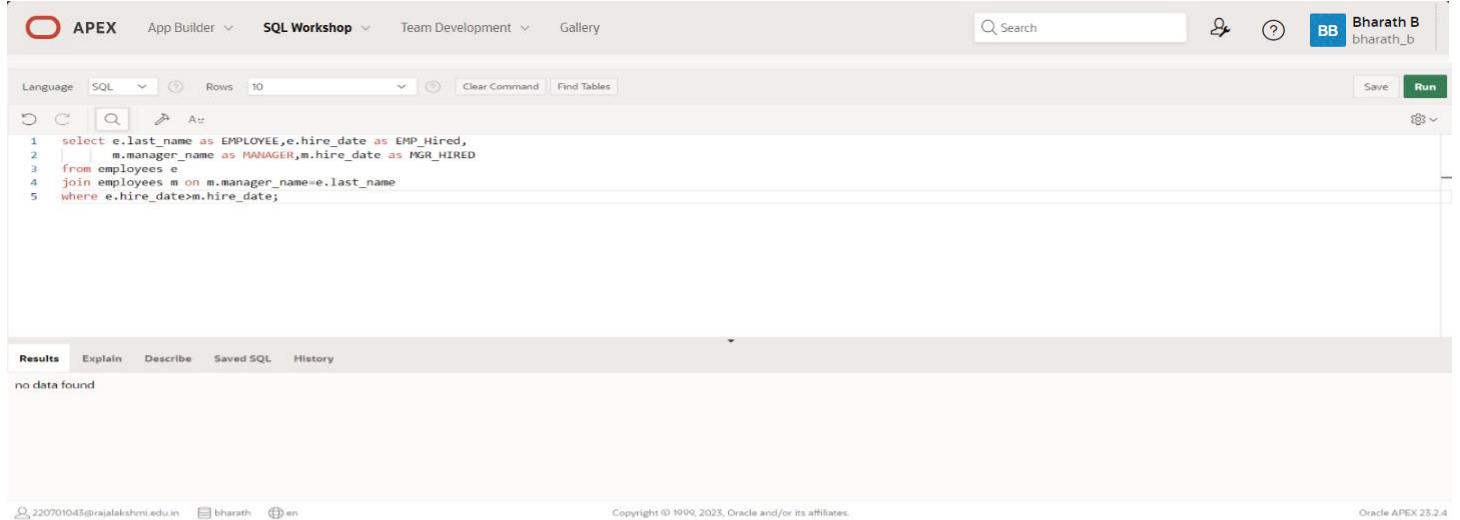
11.Display the names and hire dates for all employees who were hired before their managers,

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 employees e
JOIN employees m ON e.manager_name = m.
last_name WHERE e.hire_date < m.hire_date;
```

OUTPUT:



The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. On the right, there is a search bar, a help icon, and a user profile for 'Bharath B bharath_b'. Below the navigation is a toolbar with language selection (SQL), row counts (10), and command-related buttons like Clear Command and Find Tables. The main workspace displays the SQL query and its execution results. The query itself is:

```
1 select e.last_name as EMPLOYEE,e.hire_date as EMP_HIRED,
2       m.manager_name as MANAGER,m.hire_date as MGR_HIRED
3  from employees e
4 join employees m on m.manager_name=e.last_name
5 where e.hire_date>m.hire_date;
```

The results section shows the message "no data found". At the bottom of the page, there are footer links for user information (220701043@rajalakshmi.edu.in, bharath, en), copyright notice (Copyright © 1999, 2023, Oracle and/or its affiliates.), and system information (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

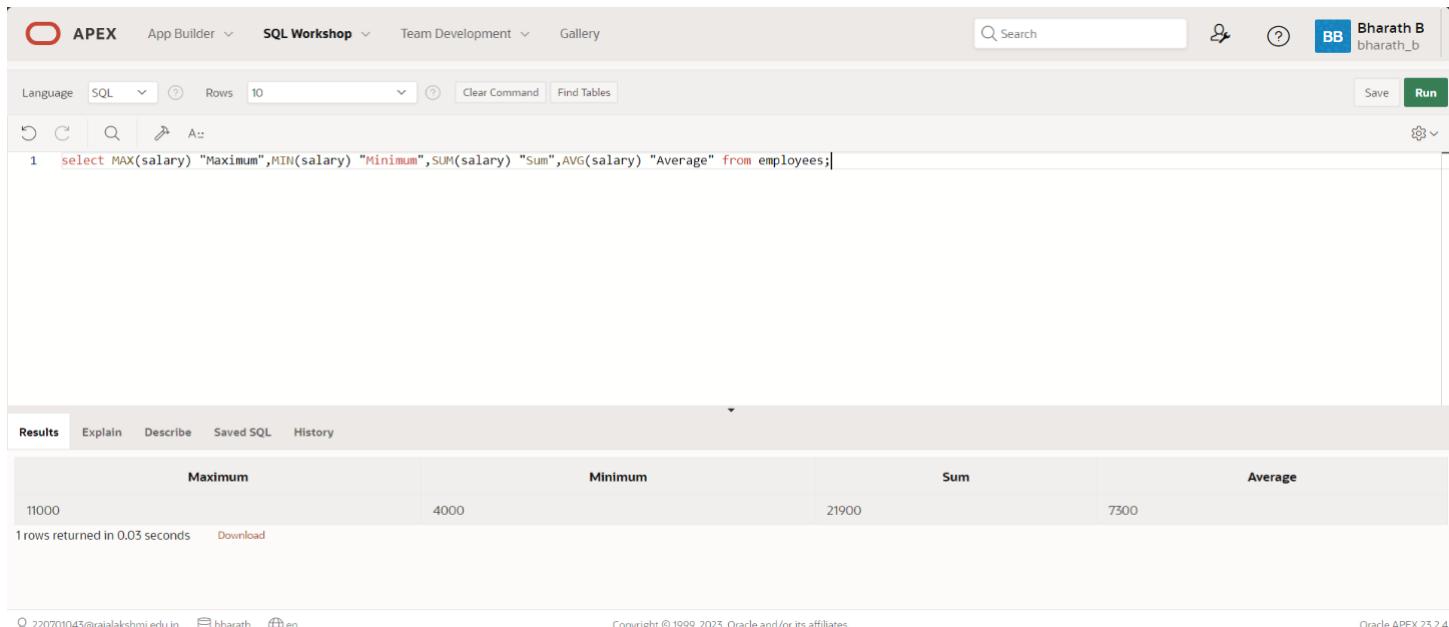
FALSE

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 EMPLOYEES;
```

OUTPUT:



The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there's a search bar, a user icon for 'Bharath B bharath_b', and buttons for Save and Run.

The SQL editor window contains the following query:

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

The Results tab displays the output of the query:

Maximum	Minimum	Sum	Average
11000	4000	21900	7300

Below the table, it says "1 rows returned in 0.03 seconds" and provides a "Download" link.

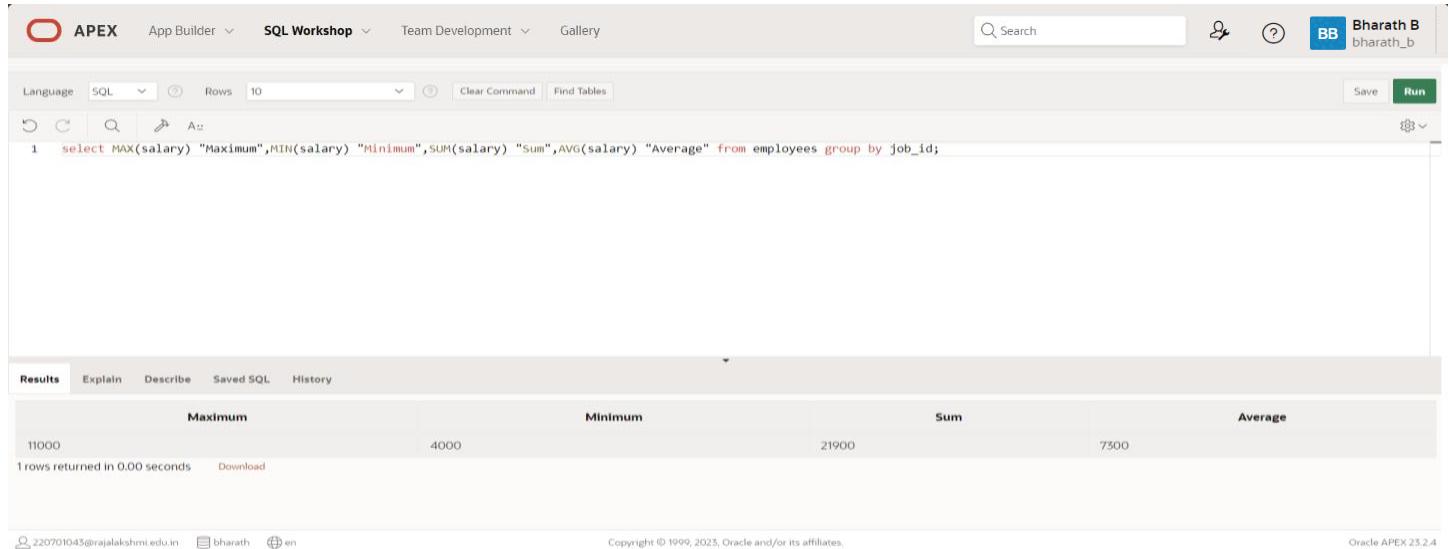
At the bottom of the page, footer information includes the URL "220701043@rajalakshmi.edu.in", the user name "bharath", the language "en", copyright notice "Copyright © 1999, 2023, Oracle and/or its affiliates.", and the version "Oracle APEX 23.2.4".

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

QUERY:

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

OUTPUT:



Maximum	Minimum	Sum	Average
11000	4000	21900	7500

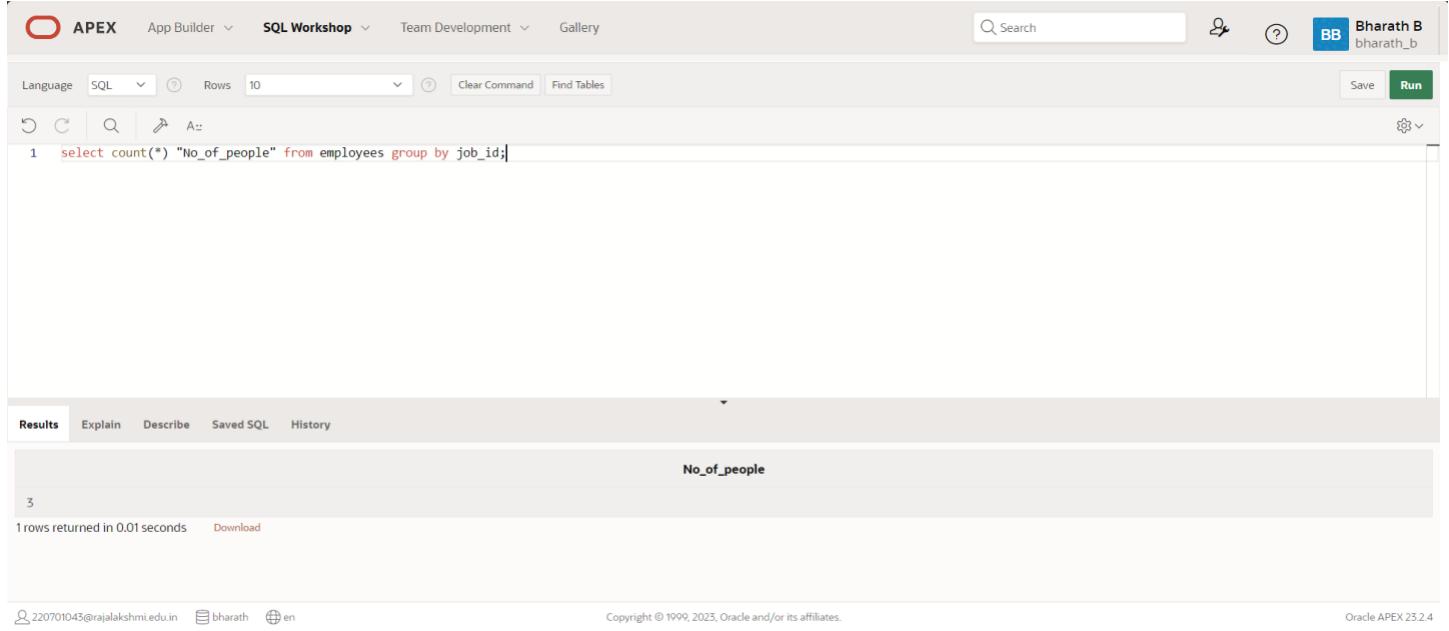
1 rows returned in 0.00 seconds [Download](#)

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

QUERY:

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

OUTPUT:



No_of_people
3

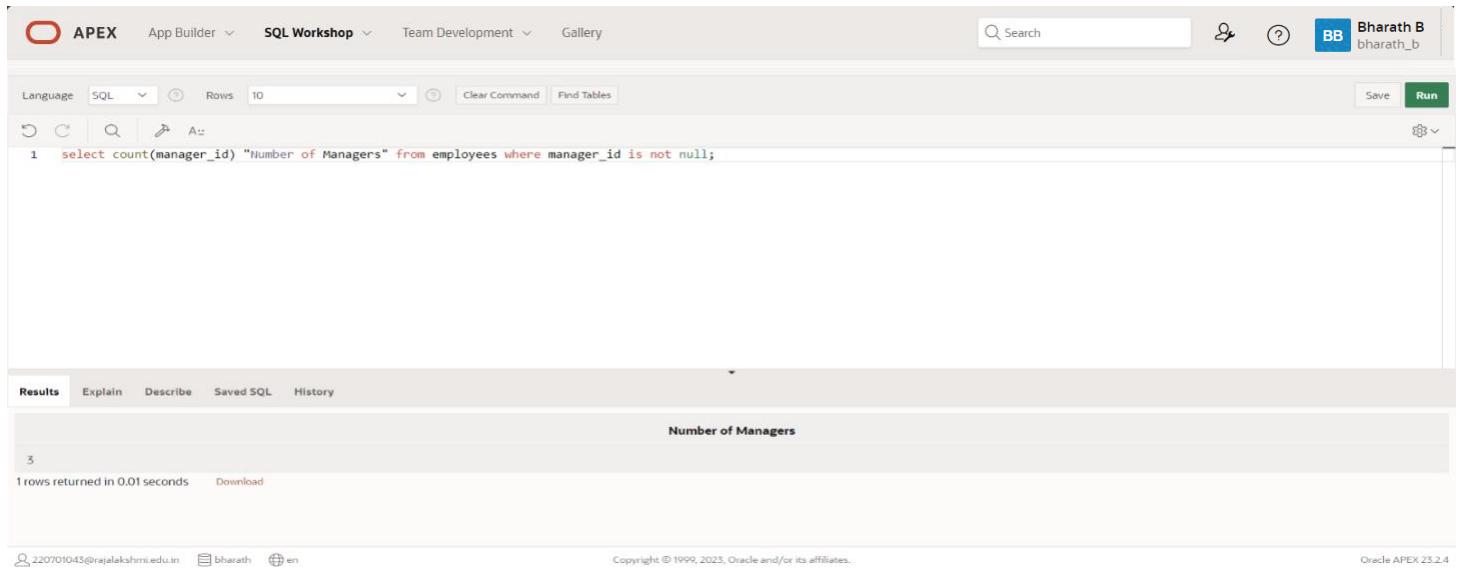
1 rows returned in 0.01 seconds [Download](#)

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

QUERY:

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

OUTPUT:



The screenshot shows the Oracle 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, and a connection named "Bharath B bharath_b". The main workspace has tabs for Language (SQL selected), Rows (10), Clear Command, and Find Tables. Below is a code editor with the following SQL query:

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

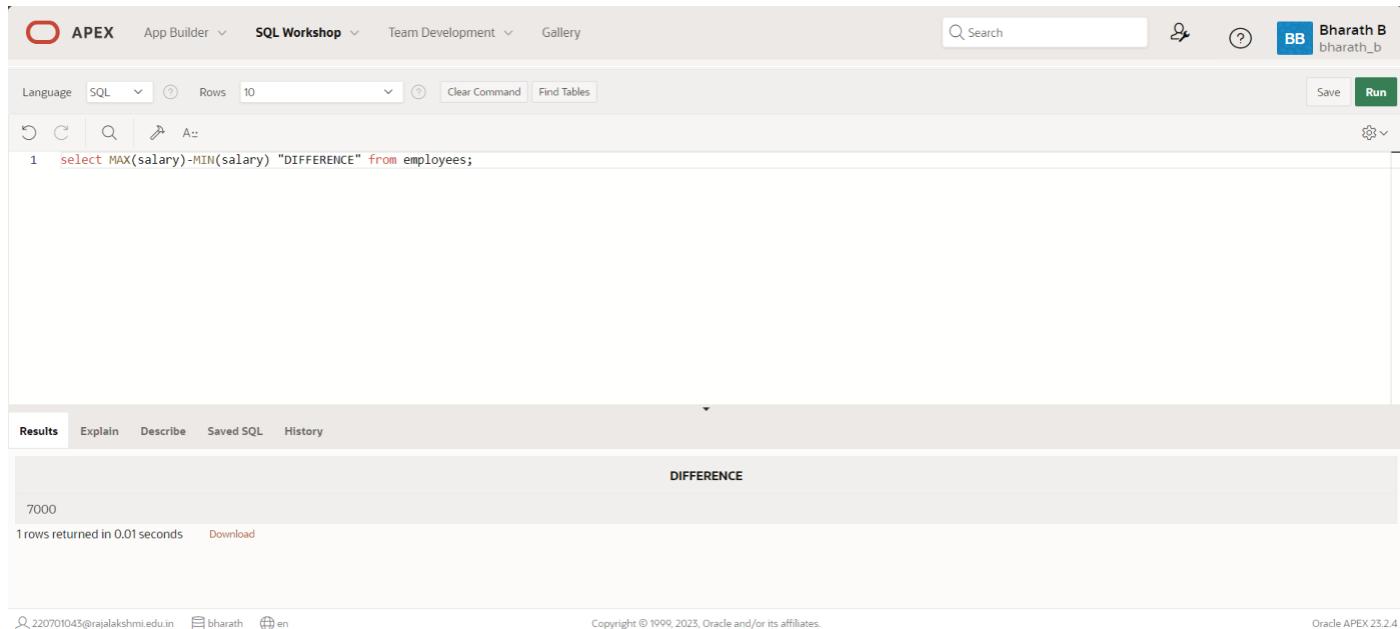
The results section is titled "Number of Managers" and displays a single row with the value 3. It also indicates 1 row was returned in 0.01 seconds and provides a download link.

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

QUERY:

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

OUTPUT:



The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. On the right, there's a search bar, a user icon, and a connection named "Bharath B bharath_b". The main workspace has tabs for Language (SQL selected), Rows (10), Clear Command, and Find Tables. Below is a code editor with the following SQL query:

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

The results section is titled "DIFFERENCE" and displays a single row with the value 7000. It also indicates 1 row was returned in 0.01 seconds and provides a download link.

9.Create a report to display the manager number and the salary of the lowest-paid employee for that manager.

Exclude anyone whose manager is not known. Exclude any groups where the minimum salary is \$6,000 or less.

Sort the output in descending order of salary.

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', 'Gallery', a search bar, and a user profile for 'Bharath B'. The SQL command entered is:

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

The results section displays the output:

MANAGER_ID	LOWEST_SALARY
100	11000
205	6900

2 rows returned in 0.01 seconds

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

Oracle APEX 23.2.4 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 employees;

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', 'Gallery', a search bar, and a user profile for 'Bharath B'. The SQL command entered is:

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

The results section displays the output:

TOTAL	1995	1996	1997	1998
3	0	0	0	1

1 rows returned in 0.04 seconds

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

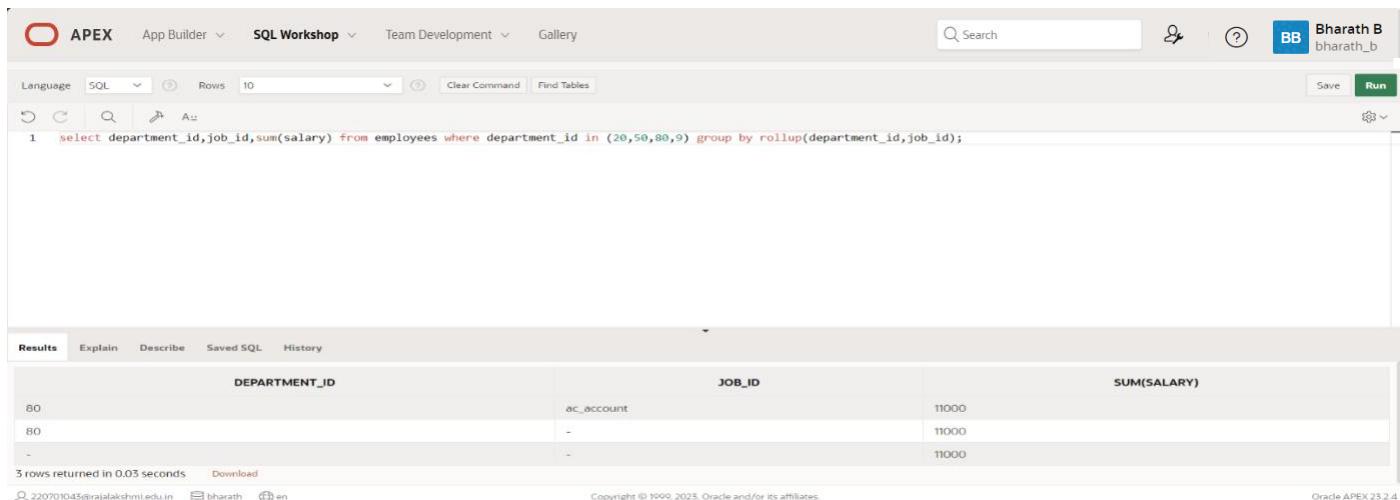
Oracle APEX 25.2.4

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 department_id,job_id,sum(salary) from employees where department_id in (20,50,80,90) group by rollup(department_id,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 'Bharath B' (bharath_b). The main area has tabs for 'Language' (set to 'SQL'), 'Rows' (set to '10'), 'Clear Command', and 'Find Tables'. Below these are icons for search, refresh, and run. The SQL command entered is:

```
1 select department_id,job_id,sum(salary) from employees where department_id in (20,50,80,90) group by rollup(department_id,job_id);
```

The results section shows a table with three rows:

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

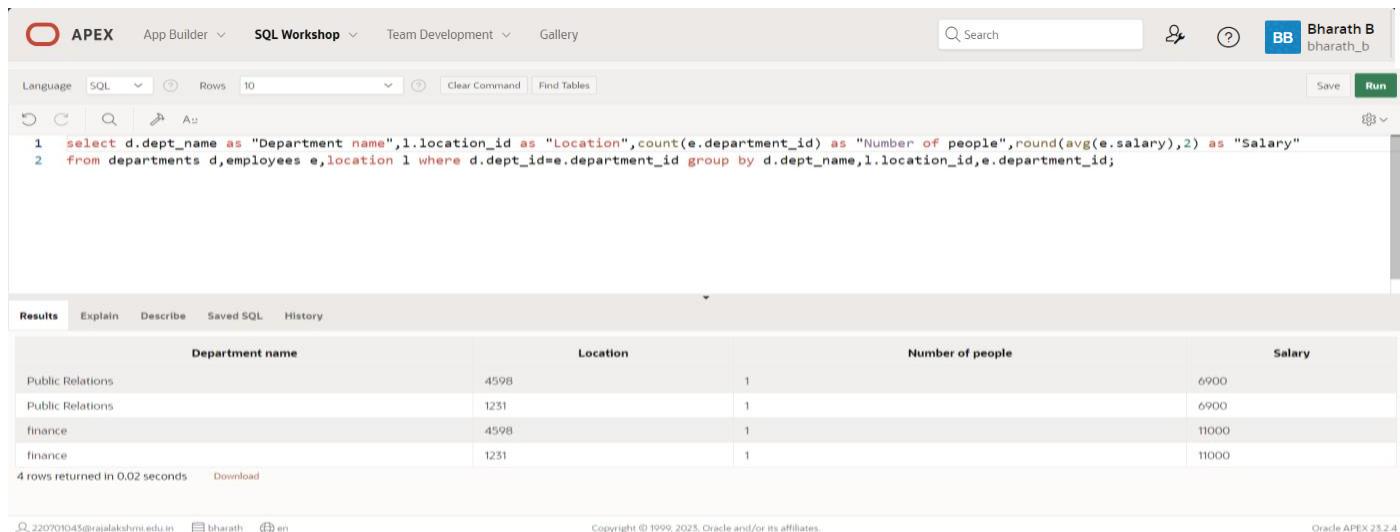
Below the table, it says '3 rows returned in 0.03 seconds' and has 'Download' and 'History' buttons.

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

QUERY:

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

OUTPUT:



The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. On the right, there's a user profile for 'Bharath B' (bharath_b). The main area has tabs for 'Language' (set to 'SQL'), 'Rows' (set to '10'), 'Clear Command', and 'Find Tables'. Below these are icons for search, refresh, and run. The SQL command entered is:

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

The results section shows a table with four rows:

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

Below the table, it says '4 rows returned in 0.02 seconds' and has 'Download' and 'History' buttons.

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

RESULT:

SUB QUERIES

EX_NO:9

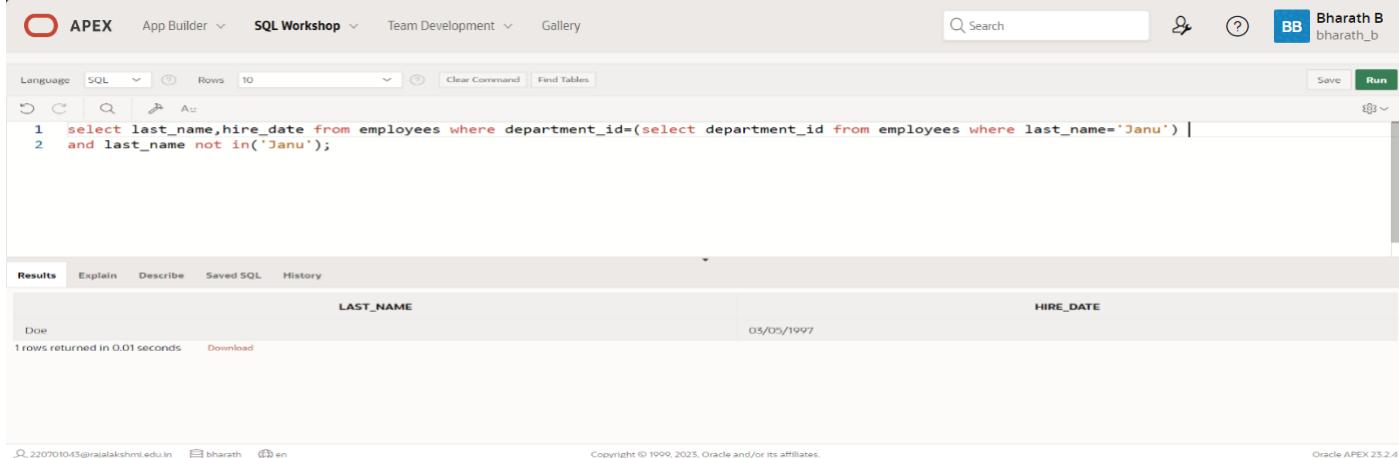
DATE:

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 search bar, a help icon, and a user profile 'Bharath B bharath_b'. The main workspace shows the following SQL code:

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

Below the code, the results section displays a single row:

LAST_NAME	HIRE_DATE
Doe	05/05/1997

Details at the bottom indicate '1 rows returned in 0.01 seconds'.

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

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

OUTPUT:



The screenshot shows the Oracle 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 help icon, and a user profile 'Bharath B bharath_b'. The main workspace shows the following SQL code:

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

Below the code, the results section displays two rows:

EMPLOYEE_ID	LAST_NAME	SALARY
142	Doe	30000
1001	Smith	70000

Details at the bottom indicate '2 rows returned in 0.01 seconds'.

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. On the right, there's a search bar, user profile 'Bharath B bharath_b', and a 'Run' button. The main area has tabs for 'Language' (set to 'SQL'), 'Rows' (set to 10), 'Clear Command', and 'Find Tables'. Below this is a code editor with the following SQL command:

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

Below the code editor is a results grid with two columns: 'EMPLOYEE_ID' and 'LAST_NAME'. The data shows two rows:

EMPLOYEE_ID	LAST_NAME
113	Janu
142	Doe

At the bottom of the results grid, it says '2 rows returned in 0.03 seconds' and has a 'Download' link. The footer of the page includes copyright information and the text 'Oracle APEX 25.2.4'.

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. On the right, there's a search bar, user profile 'Bharath B bharath_b', and a 'Run' button. The main area has tabs for 'Language' (set to 'SQL'), 'Rows' (set to 10), 'Clear Command', and 'Find Tables'. Below this is a code editor with the following SQL command:

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

Below the code editor is a results grid with three columns: 'LAST_NAME', 'DEPARTMENT_ID', and 'JOB_ID'. The data shows two rows:

LAST_NAME	DEPARTMENT_ID	JOB_ID
Janu	100	ac_account
Doe	100	ac_account

At the bottom of the results grid, it says '2 rows returned in 0.04 seconds' and has a 'Download' link. The footer of the page includes copyright information and the text 'Oracle APEX 25.2.4'.

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. The search bar contains the placeholder "Search". The user profile "Bharath B bharath_b" is visible on the right. The main workspace has a toolbar with icons for Undo, Redo, Search, Find Tables, Clear Command, and Run. The SQL editor contains the following query:

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

The results section displays the output of the query:

LAST_NAME	SALARY
Davies	11000
Mohan	4000

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

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

QUERY:

```
select department_id,last_name,job_id from employees where department_id in  
(select department_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 search bar contains the placeholder "Search". The user profile "Bharath B bharath_b" is visible on the right. The main workspace has a toolbar with icons for Undo, Redo, Search, Find Tables, Clear Command, and Run. 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  
2 where dept_name='Executive');
```

The results section displays the output of the query:

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

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

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 interface with the 'SQL Workshop' tab selected. A SQL statement is entered in the query editor:

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

The results page displays the output of the query:

EMPLOYEE_ID	LAST_NAME	SALARY
1001	Smith	70000
142	Doeu	30000

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

At the bottom of the page, there is a footer with user information and copyright details:

220701043@rajalakshmi.edu.in bharath en Copyright © 1999, 2023, Oracle and/or its affiliates. Oracle APEX 25.2.4

A large table is overlaid on the bottom right of the page:

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 query entered is:

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

The results show two rows of data:

DEPARTMENT_ID
20
100

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

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 query 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;
```

The results show one row of data:

COUNTRY_ID	STATE_PROVINCE
58	toronto

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

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 links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there's a search bar, user profile for Bharath B (bharath_b), and a 'Run' button.

The SQL editor contains the following query:

```

1 select job_id,department_id from employees where department_id=10 union
2 select job_id,department_id from employees where department_id=5|union
3 select job_id,department_id from employees where department_id=20;

```

The results section shows a table with two columns: JOB_ID and DEPARTMENT_ID. The data is:

JOB_ID	DEPARTMENT_ID
ac_account	20
hr_rep	20

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

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

QUERY:

```
select 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, identical to the previous one but with a different query.

The SQL editor contains the following query:

```

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

```

The results section shows a table with two columns: JOB_ID and EMPLOYEE_ID. The data is:

JOB_ID	EMPLOYEE_ID
ac_account	115
ac_account	142
sales_rep	1001

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

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

QUERY:

```
select first_name||" "||last_name as "Name",department_id from employees union all
select dept_name,department_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 (which is selected), Team Development, and Gallery. On the right, there's a search bar, user profile, and a blue button for 'Bharath B' with the ID 'bharath_b'. Below the tabs, there are buttons for Language (SQL), Rows (10), Clear Command, and Find Tables. The main area contains the SQL command:

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

Below the command is a results grid with two columns: 'Name' and 'DEPARTMENT_ID'. The data is as follows:

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

At the bottom left, it says '10 rows returned in 0.01 seconds' and has a 'Download' link. At the bottom right, it says 'Copyright © 1999, 2023, Oracle and/or its affiliates.' and 'Oracle APEX 23.2.4'.

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

RESULT:

CREATING VIEWS

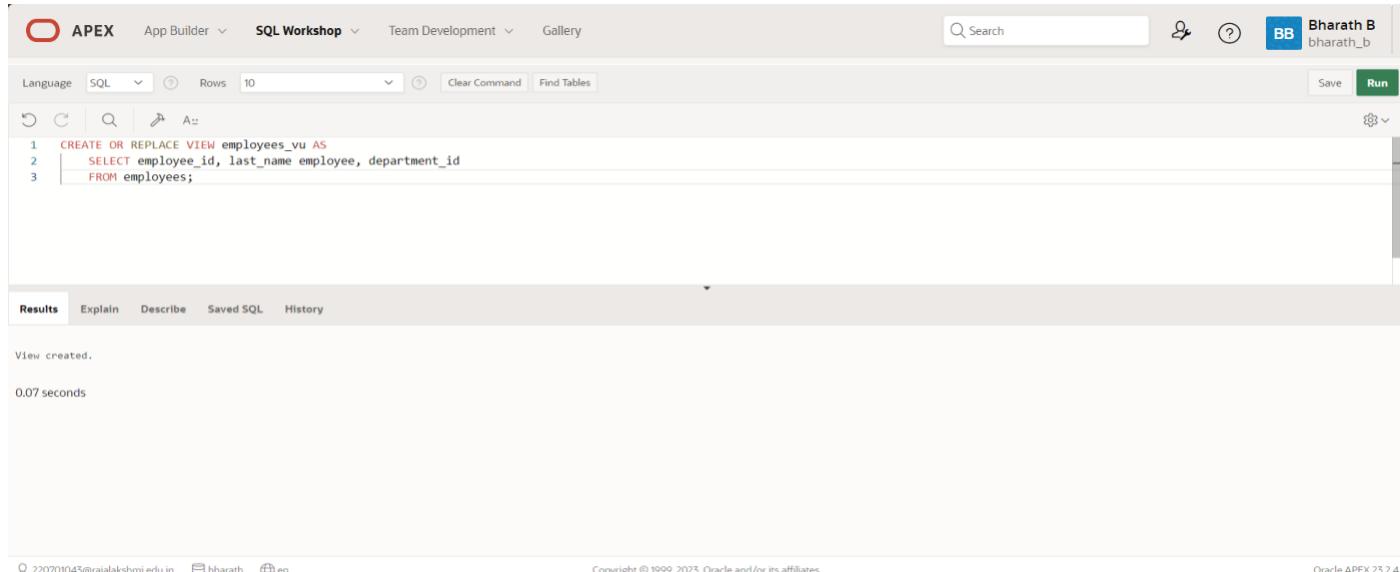
EX_NO:11

DATE:

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. On the right, there's a user icon for 'Bharath B' (bharath_b). The main workspace has a search bar and a toolbar with icons for refresh, search, and run. Below the toolbar, the SQL command is displayed:

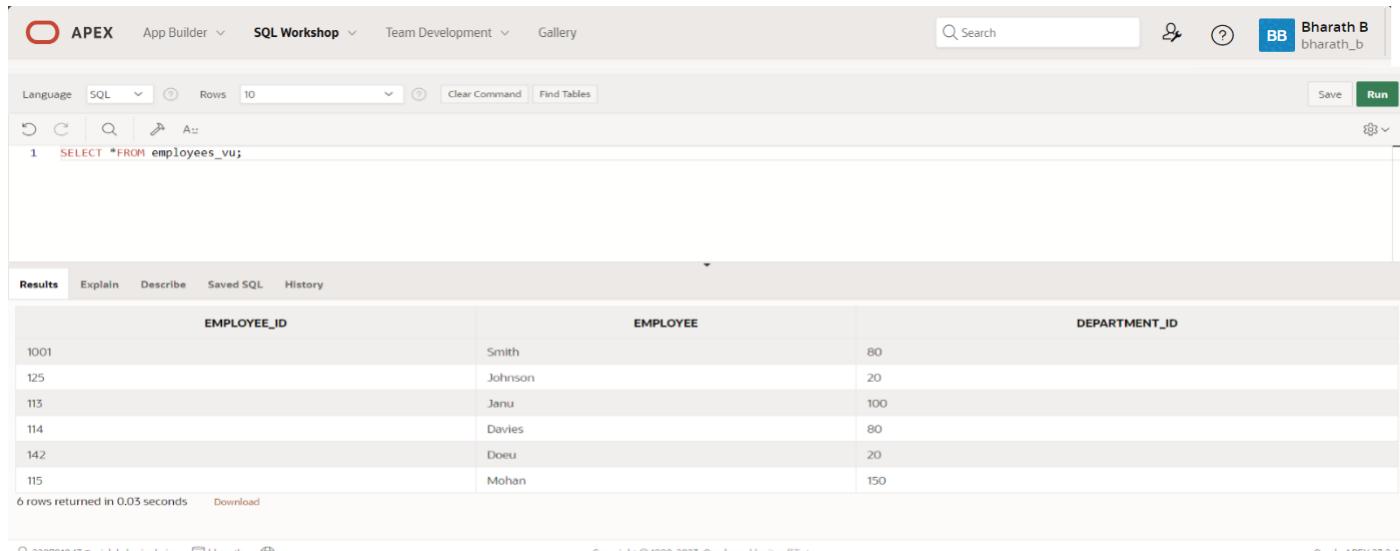
```
1 CREATE OR REPLACE VIEW employees_vu AS
2   SELECT employee_id, last_name employee, department_id
3   FROM employees;
```

Below the SQL area, the results tab is selected, showing the message 'View created.' and a execution time of '0.07 seconds'. At the bottom, it shows the session information '220701043@rajalakshmi.edu.in bharath en' and the copyright notice 'Copyright © 1999, 2023, Oracle and/or its affiliates.' followed by 'Oracle APEX 23.2.4'.

2.) Display the contents of the EMPLOYEES_VU view.

QUERY:

```
select * from employees_vu;
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface with the same layout as the previous screenshot. The SQL command is:

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

The results tab displays a table with three columns: 'EMPLOYEE_ID', 'EMPLOYEE', and 'DEPARTMENT_ID'. The data is as follows:

EMPLOYEE_ID	EMPLOYEE	DEPARTMENT_ID
1001	Smith	80
125	Johnson	20
113	Janu	100
114	Davies	80
142	Doeu	20
115	Mohan	150

At the bottom, it says '6 rows returned in 0.05 seconds' and provides a 'Download' link. The session and copyright information are at the bottom.

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 interface with the SQL Workshop tab selected. In the command editor, the following SQL code is entered:

```

1 SELECT view_name, text
2 FROM user_views;
3

```

The results page displays the output of the query:

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

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

At the bottom, the footer includes the URL 220701043@rajalakshmi.edu.in, the user bharath, and the session ID en. It also states Copyright © 1999, 2023, Oracle and/or its affiliates, and 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 interface with the SQL Workshop tab selected. In the command editor, the following SQL code is entered:

```

1 SELECT employee, department_id FROM employees_vu;
2

```

The results page displays the output of the query:

EMPLOYEE	DEPARTMENT_ID
Smith	80
Johnson	20
Janu	100
Davies	80
Doeu	20
Mohan	150

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

At the bottom, the footer includes the URL 220701043@rajalakshmi.edu.in, the user bharath, and the session ID en. It also states Copyright © 1999, 2023, Oracle and/or its affiliates, and 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 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. The search bar contains 'Search'. On the right, there's a user icon for 'Bharath B bharath_b'. The main workspace displays the SQL command for creating the 'dept50' view:

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

Below the code, the results pane shows the message 'View created.' and a duration of '0.06 seconds'. At the bottom, the URL is '220701043@rajalakshmi.edu.in', the session ID is 'bharath', and the page is 'en'. The copyright notice 'Copyright © 1999, 2023, Oracle and/or its affiliates.' and the version 'Oracle APEX 23.2.4' are also visible.

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

QUERY:

```
Describe dept50;
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface with the 'Describe' tab selected. The workspace contains the SQL command 'DESCRIBE dept50 ;'. The results pane displays the structure of the 'DEPT50' view:

Object Type	VIEW	Object	DEPT50						
Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
DEPT50	EMPNO	NUMBER	-	6	0	-	-	-	-
	EMPLOYEE	VARCHAR2	20	-	-	-	-	-	-
	DEPTNO	NUMBER	-	4	0	-	✓	-	-

At the bottom, the URL is '220701043@rajalakshmi.edu.in', the session ID is 'bharath', and the page is 'en'. The copyright notice 'Copyright © 1999, 2023, Oracle and/or its affiliates.' and the version 'Oracle APEX 23.2.4' are also visible.

7.) Attempt to reassign Matos to department 80

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. At the top, there are tabs for APEX, App Builder, SQL Workshop (which is selected), Team Development, and Gallery. On the right, there's a search bar, a user icon, and a session identifier "Bharath B bharath_b". The main area has a toolbar with icons for Undo, Redo, Search, and Find Tables. Below the toolbar, the SQL editor contains the following code:

```

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

```

Below the code, the results tab is selected, showing the output: "0 row(s) updated." and "0.05 seconds".

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

QUERY:

```
create or replace view salary_vu as select e.last_name "Employee",d.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. The setup is identical to the previous one, with the SQL Workshop tab selected. The SQL editor contains the same CREATE VIEW statement as above. The results tab shows the output: "View created." and "0.06 seconds".

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						

- 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.

- 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.

- 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..

- 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	NULABLE
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_NUMBER	ADMIT_DATE	ADOPTION_ID	VACCINATION_DATE
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	

RESULT:

EXERCISE 13

Creating Views

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

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

3. SELECT * FROM view_d_songs. What was returned?

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' is 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 displays the output:

```
Sequence created.
```

Execution time: 0.03 seconds.

At the bottom, the URL is 220701043@rajalakshmi.edu.in, the session ID is bharath, and the page is en. Copyright information and Oracle APEX version 23.2.4 are also visible.

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

QUERY:

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

OUTPUT:

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

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

Below the code, the results pane displays 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 URL is 220701043@rajalakshmi.edu.in, the session ID is bharath, and the page is en. Copyright information and Oracle APEX version 23.2.4 are also visible.

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 links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. On the right, there are user profile icons for 'Bharath B' and 'bharath_b'. The main area displays a summary of a script named 'exercise14' which is 'Complete'. It shows 15 rows processed, with 1 row inserted successfully. Below this, three detailed status bars are shown: 'Statements Processed' (1), 'Successful' (1), and 'With Errors' (0). 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 version information (Oracle APEX 23.2.4).

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. On the right, there are user profile icons for 'Bharath B' and 'bharath_b'. The main area shows the SQL command being run: 'CREATE INDEX emp_dept_id_idx ON EMPLOYEES (department_id);'. The results section at the bottom shows the message 'Index created.' and a runtime of '0.03 seconds'. The footer includes user information (220701043@rajalakshmi.edu.in, bharath, en), copyright notice (Copyright © 1999, 2023, Oracle and/or its affiliates.), and version information (Oracle APEX 23.2.4).

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. At the top, there are tabs for 'APEX', 'App Builder', 'SQL Workshop' (which is selected), 'Team Development', and 'Gallery'. On the right, there's a search bar, a help icon, and a user profile for 'Bharath B bharath_b'. Below the tabs, there are buttons for 'Language', 'SQL', 'Rows' (set to 10), 'Clear Command', and 'Find Tables'. On the far right, there are 'Save' and 'Run' buttons.

The main area contains the SQL query:

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

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

INDEX_NAME	TABLE_NAME	UNIQUENESS
EMP_DEPT_ID_IDX	EMPLOYEES	NONUNIQUE

Below the table, it says '1 rows returned in 0.03 seconds' and has a 'Download' link. At the bottom of the page, there are links for '220701043@rajalakshmi.edu.in', 'bharath', and 'en'. The copyright notice 'Copyright © 1999, 2023, Oracle and/or its affiliates.' and the version 'Oracle APEX 23.2.4' are also present.

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

RESULT:

CONTROLLING USER ACCESS

EX_NO:15

DATE:

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

The CREATE SESSION system privilege

2. What privilege should a user be given to create

tables? The CREATE TABLE privilege

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

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

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

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

5. What command do you use to change your

password? The ALTER USER statement

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

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

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

7. Query all the rows in your DEPARTMENTS table.

SELECT * FROM departments;

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

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

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

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

10. Revoke the SELECT privilege on your table from the other team. Team 1 revokes the privilege.

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

Team 2 revokes the privilege.

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

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

Team 1 executes this INSERT statement.

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

Team 2 executes this INSERT statement.

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

<u>Evaluation Procedure</u>	<u>Marks awarded</u>
Practice Evaluation (5)	
Viva(5)	
Total (10)	
<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. In the top navigation bar, 'APEX' is selected, along with 'SQL Workshop'. The main area contains the PL/SQL code provided above. Below the code, the 'Results' tab is active, showing the output of the executed query. The output displays the calculated incentive value and a message indicating the statement was processed. At the bottom of the page, there are footer links and copyright information.

```
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;
```

Incentive = 8400
Statement processed.
0.00 seconds

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

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. In the code editor, there is a syntax error highlighted in yellow. The error message is: "Error at line 4/25: ORA-06550: line 4, column 25: PLS-00201: identifier 'Welcome' must be declared ORA-06512: at "SYS.IMP_DBMIS_SQL_APEX_230200", line 801 ORA-06550: line 4, column 3: PL/SQL: Statement ignored". The code block is as follows:

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

The results pane shows the error message again.

The screenshot shows the Oracle APEX SQL Workshop interface. Similar to the previous one, it highlights a syntax error in the code editor. The error message is: "Error at line 4/25: ORA-06550: line 4, column 25: PLS-00201: identifier 'Welcome' must be declared ORA-06512: at "SYS.IMP_DBMIS_SQL_APEX_230200", line 801 ORA-06550: line 4, column 3: PL/SQL: Statement ignored". The code block is identical to the first one:

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

The results pane shows the error message again.

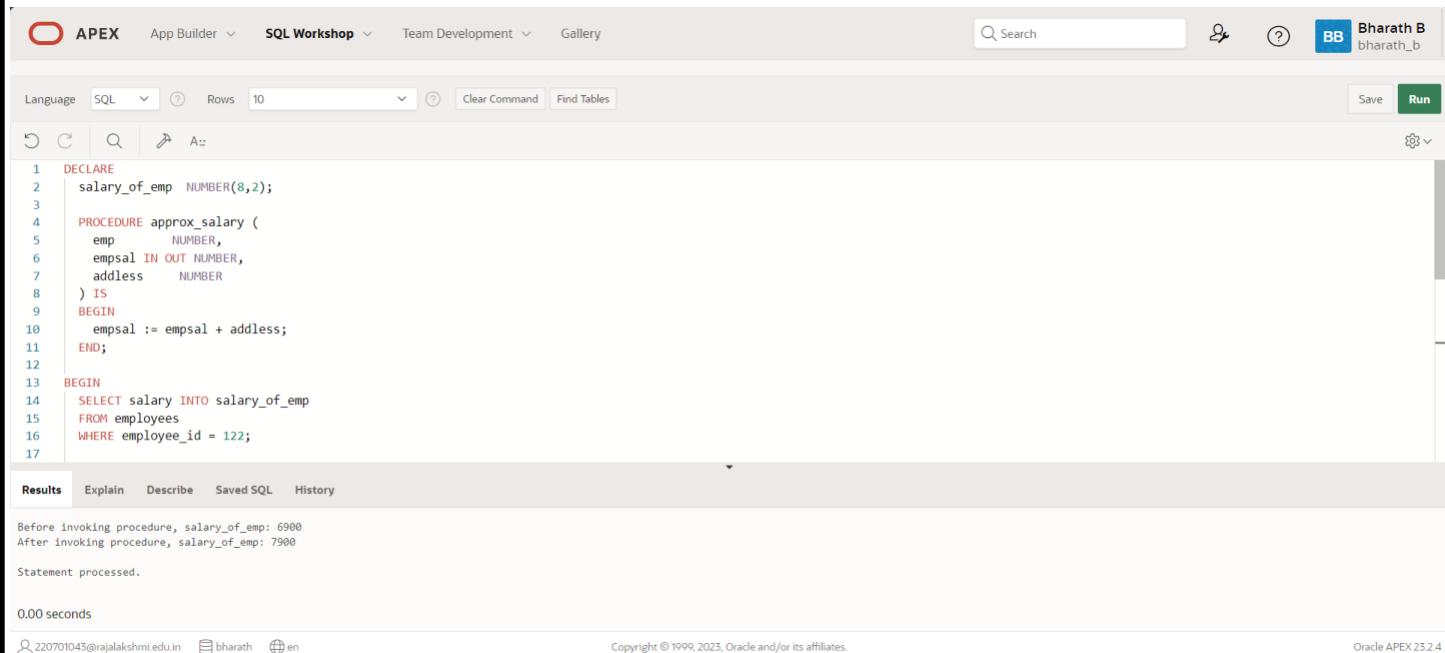
3.) Write a PL/SQL block to adjust the salary of the employee whose ID 122.

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 links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows a user profile for 'Bharath B' (bharath_b). The main workspace displays the PL/SQL code from the previous section. The code is numbered from 1 to 17. Lines 1-12 define a procedure 'approx_salary' that adds two numbers and returns the result. Lines 13-16 show the execution of this procedure within a block that also performs a SELECT statement to find the salary of employee ID 122. The output window at the bottom shows the results of the query and the output of the DBMS_OUTPUT.PUT_LINE statements.

```
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

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

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. In the top navigation bar, 'APEX' is selected. Below the toolbar, the language is set to 'SQL'. The code area contains the PL/SQL procedure definition. The results tab shows the message 'Procedure created.' and a execution time of '0.03 seconds'. The bottom footer includes user information and copyright details.

```
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 /
15 |
```

Results Explain Describe Saved SQL History

Procedure created.
0.03 seconds

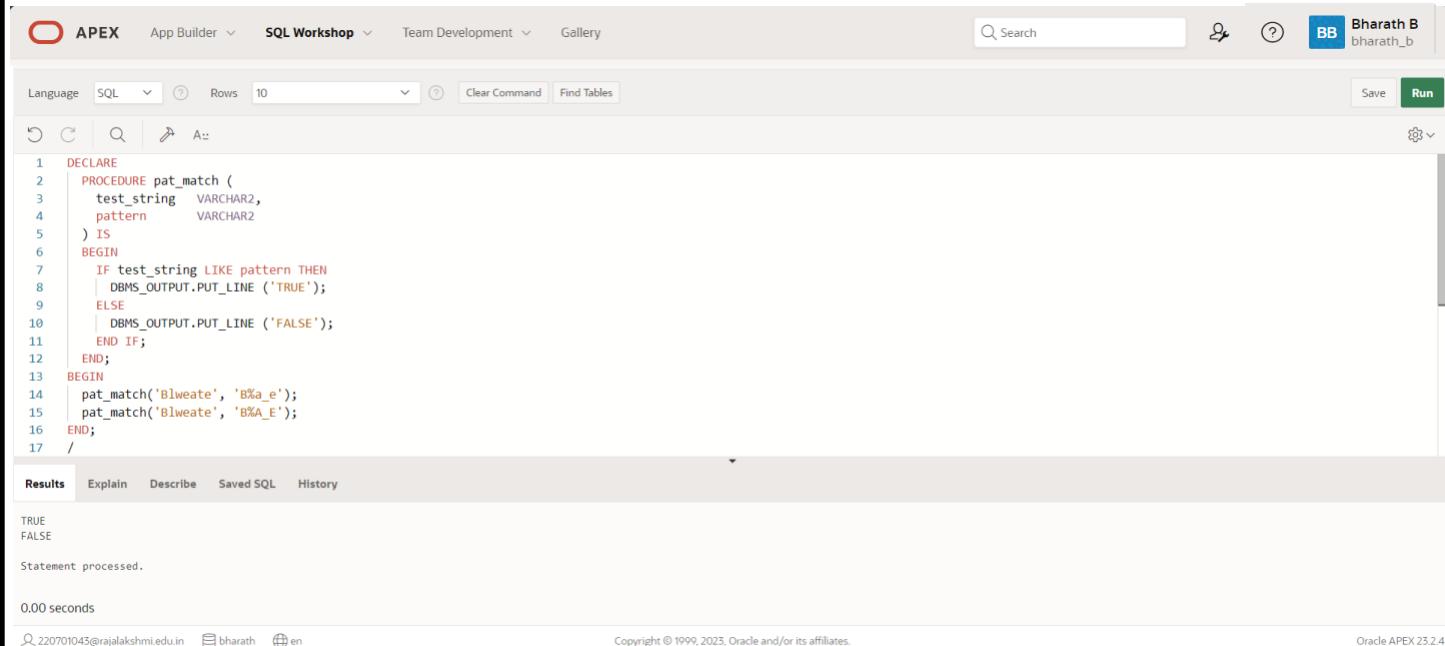
220701045@rajalakshmi.edu.in bharath en Copyright © 1999, 2023, Oracle and/or its affiliates. Oracle APEX 23.2.4

- 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 interface with the SQL Workshop tab selected. The code area contains the provided PL/SQL procedure. The results section shows the output of the execute command.

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

Results

```
TRUE
FALSE

Statement processed.

0.00 seconds
```

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

QUERY:

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

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

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. The search bar contains 'Search'. On the right, there's a user icon for 'Bharath B' and a 'Run' button.

The main workspace displays the PL/SQL code. The code is numbered from 1 to 17. Lines 1 through 12 represent the declaration and assignment of variables. Lines 13 and 14 show the use of DBMS_OUTPUT.PUT_LINE to print the values of num_small and num_large. Lines 15 and 16 are the END block and a slash. Line 17 is a blank line.

Below the code, the 'Results' tab is selected. The output section shows the printed values: 'num_small = 5' and 'num_large = 8'. Below this, it says 'Statement processed.' and '0.00 seconds'.

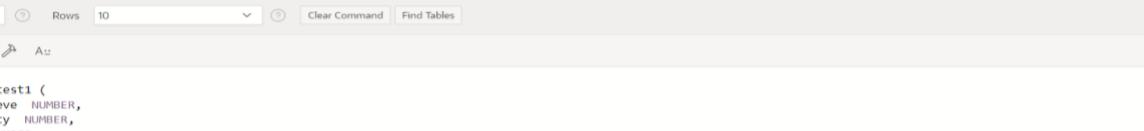
At the bottom, the footer includes the URL '220701045@rajalakshmi.edu.in', the name 'bharath', the language 'en', copyright information 'Copyright © 1999, 2023, Oracle and/or its affiliates.', and the version 'Oracle APEX 23.2.4'.

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

QUERY:

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

OUTPUT:



```
1 DECLARE
2   PROCEDURE test1 (
3     sal_achieve NUMBER,
4     target_qty NUMBER,
5     emp_id NUMBER
6   )
7   IS
8     incentive NUMBER := 0;
9     updated VARCHAR2(3) := 'No';
10    BEGIN
11      IF sal_achieve > (target_qty + 200) THEN
12        incentive := (sal_achieve - target_qty)/4;
13      UPDATE employees
14        SET salary = salary + incentive
15        WHERE employee_id = emp_id;
16    END;
17
```

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

OLIERY

```

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 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 'Bharath B' (bharath_b). The main workspace has tabs for Language (SQL selected), Rows (10), Clear Command, and Find Tables. Below these are standard file operations like Undo, Redo, Save, and Run. The code area contains the PL/SQL procedure shown above. The Results tab is active, displaying the output of three calls to the test1 procedure with inputs 45000, 36000, and 28000, which prints 'Sale achieved : [value], incentive : [value] .' for each. The bottom status bar shows the user email (220701045@rajalakshmi.edu.in), session ID (bhарат), and Oracle APEX version (23.2.4).

```

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 Explain Describe Saved SQL History
Sale achieved : 45000, incentive : 1800.
Sale achieved : 36000, incentive : 800.
Sale achieved : 28000, incentive : 500.
Statement processed.

0.00 seconds
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 SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. The right side shows a user profile for 'Bharath b'.

The SQL Workshop interface has tabs for Language (SQL), Rows (10), Clear Command, Find Tables, Save, and Run. Below these are standard edit tools (Undo, Redo, Search, etc.).

The code area contains the PL/SQL block from above, numbered 1 to 15. Lines 10 and 11 show the department ID being set to 50. Line 12 shows the output message. Line 15 shows the IF condition for vacancies.

The Results tab displays the output of the executed code:

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

At the bottom, footer information includes the URL 220701043@rajalakshmi.edu.in, the user name bharath, and the language en. Copyright information for Oracle APEX 23.2.4 is also present.

- 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:

```
===
APEX App Builder SQL Workshop Team Development Gallery Search Run
Language SQL Rows 10 Clear Command Find Tables Save Run
Results Explain Describe Saved SQL History

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 employees are in the department 80 is: 4
There are 41 vacancies in department 80
Statement processed.
0.03 seconds

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

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

QUERY:

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

OUTPUT:

```
1  DECLARE
2    v_employee_id employees.employee_id%TYPE;
3    v_full_name employees.first_name%TYPE;
4    v_job_id employees.job_id%TYPE;
5    v_hire_date employees.hire_date%TYPE;
6    v_salary employees.salary%TYPE;
7    CURSOR c_employees IS
8      SELECT employee_id, first_name || ' ' || last_name AS full_name, job_id, hire_date, salary
9        FROM employees;
10   BEGIN
11     DBMS_OUTPUT.PUT_LINE('Employee ID | Full Name | Job Title | Hire Date | Salary');
12     DBMS_OUTPUT.PUT_LINE('-----');
13     OPEN c_employees;
```

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 6900
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

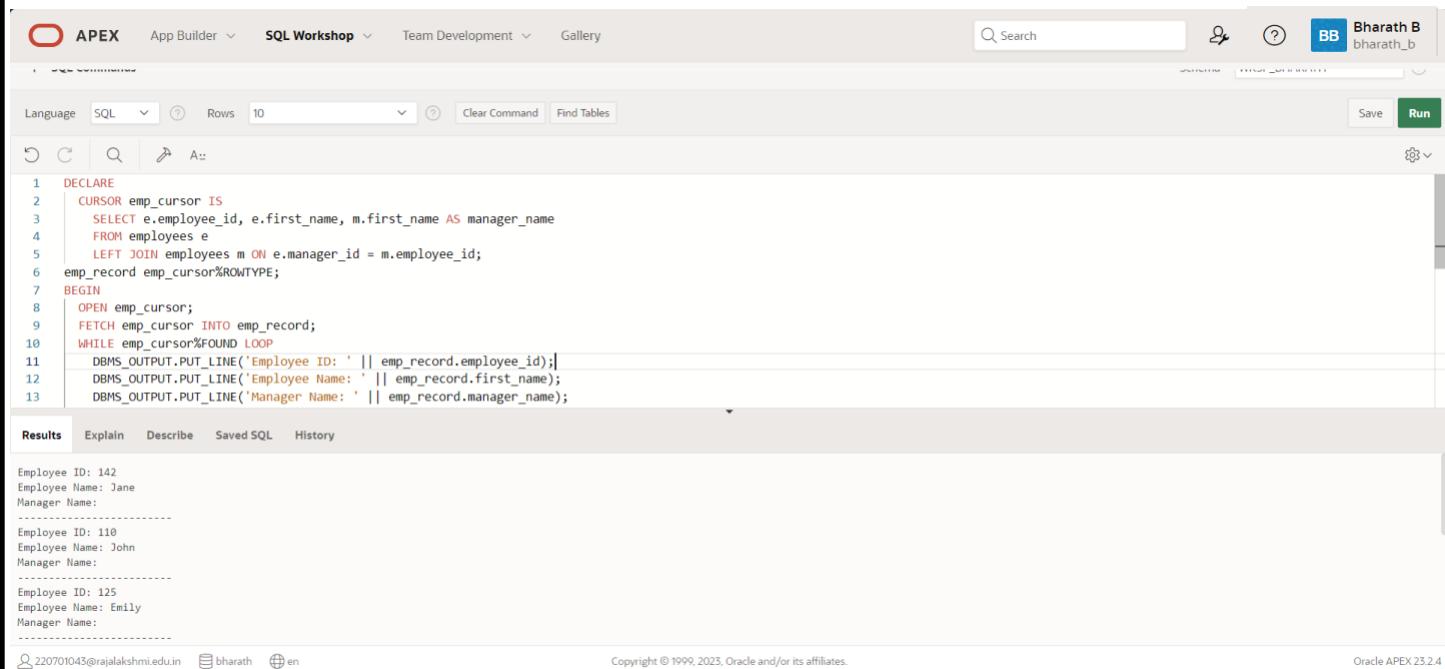
Statement processed.

- 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:



```
1 DECLARE
2 CURSOR emp_cursor IS
3 SELECT e.employee_id, e.first_name, m.first_name AS manager_name
4 FROM employees e
5 LEFT JOIN employees m ON e.manager_id = m.employee_id;
6 emp_record emp_cursor%ROWTYPE;
7 BEGIN
8 OPEN emp_cursor;
9 FETCH emp_cursor INTO emp_record;
10 WHILE emp_cursor%FOUND LOOP
11 DBMS_OUTPUT.PUT_LINE('Employee ID: ' || emp_record.employee_id);
12 DBMS_OUTPUT.PUT_LINE('Employee Name: ' || emp_record.first_name);
13 DBMS_OUTPUT.PUT_LINE('Manager Name: ' || emp_record.manager_name);
14 DBMS_OUTPUT.PUT_LINE('-----');
```

Employee ID: 142
Employee Name: Jane
Manager Name:

Employee ID: 110
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. The code area contains a PL/SQL block that retrieves job IDs and their minimum salaries from the job_grade and employees tables. The output pane 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 SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The search bar contains the placeholder 'Search'. On the right, there is a user profile for 'Bharath B' (bharath_b).

The main workspace displays the PL/SQL code. The code declares a cursor for employees, performs a natural join with job_history, and then loops through the results. Inside the loop, it prints each row's employee ID, last name (right-padded to 25), job ID (right-padded to 35), and start date (in dd-mon-yyyy format). The output section shows the results for employee 125.

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

At the bottom, the status bar indicates the statement was processed, and the footer shows copyright information for Oracle APEX 23.2.4.

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

QUERY:

```
DECLARE
```

```

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

```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows a user profile for 'Bharath B' (bharath_b). The main area has tabs for Language (SQL selected), Rows (10), Clear Command, Find Tables, Save, and Run. The code area contains a PL/SQL block. The results pane displays the output of the code execution.

```

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: 125
Employee Name: Emily
End Date: 03/21/1997
-----
Statement processed.

```

At the bottom, it shows the user email (220701043@rajalakshmi.edu.in), session name (bharath), and language (en). Copyright information (Copyright © 1999, 2023, Oracle and/or its affiliates) and the Oracle APEX version (Oracle APEX 23.2.4) are also present.

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 a SQL developer interface with the following details:

- Toolbar:** Language (SQL), Rows (10), Clear Command, Find Tables, Save, Run.
- Code Area:** Displays the PL/SQL block provided in the question. Lines 1 through 10 are visible, ending with the final closing brace of the block.
- Results Tab:** Shows the output of the executed statement.
- Output Content:**
 - 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';
```

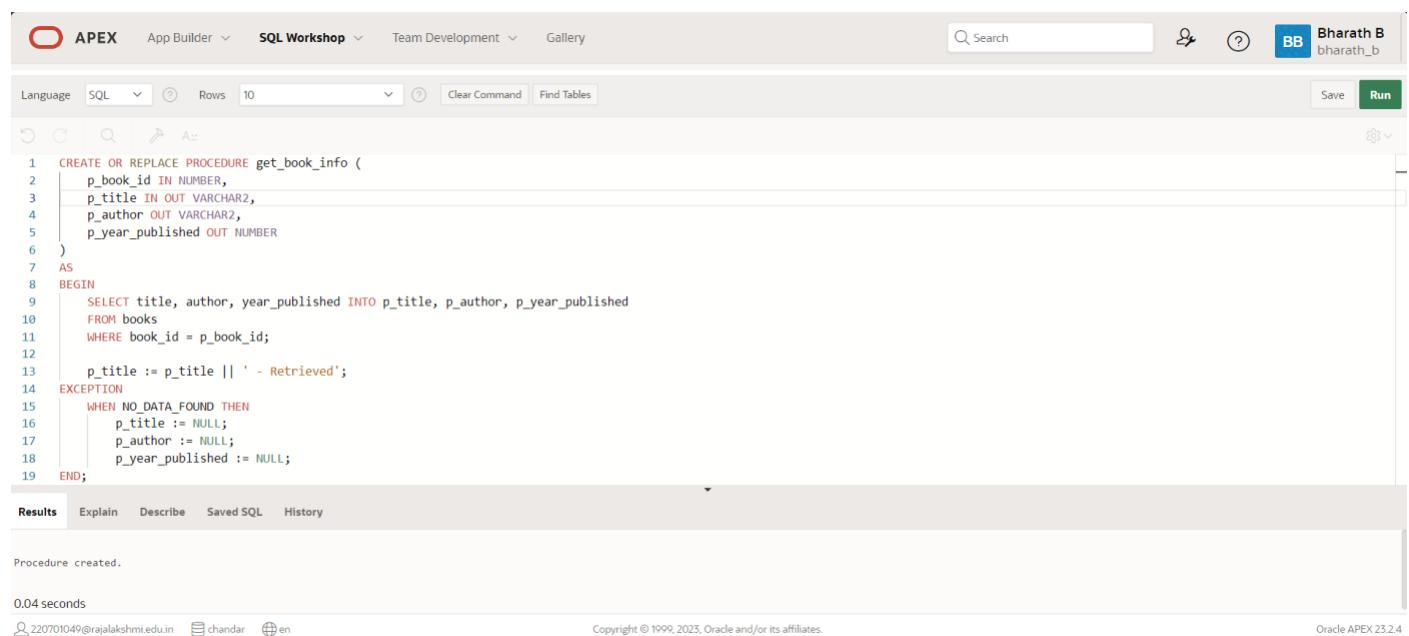
```

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 SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. On the right, there are 'Search', 'Help', and a user icon for 'Bharath B bharath_b'. Below the toolbar, the language is set to 'SQL' and rows are set to 10. The main area contains the PL/SQL code for the 'get_book_info' procedure. The code uses DBMS_OUTPUT to print the title, author, and year published. The 'Run' button is visible at the top right of the code editor.

```

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;

```

The results section shows the message 'Procedure created.' and a execution time of '0.04 seconds'. The bottom footer includes copyright information for Oracle and the APEX version 'Oracle APEX 23.2.0'.

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:

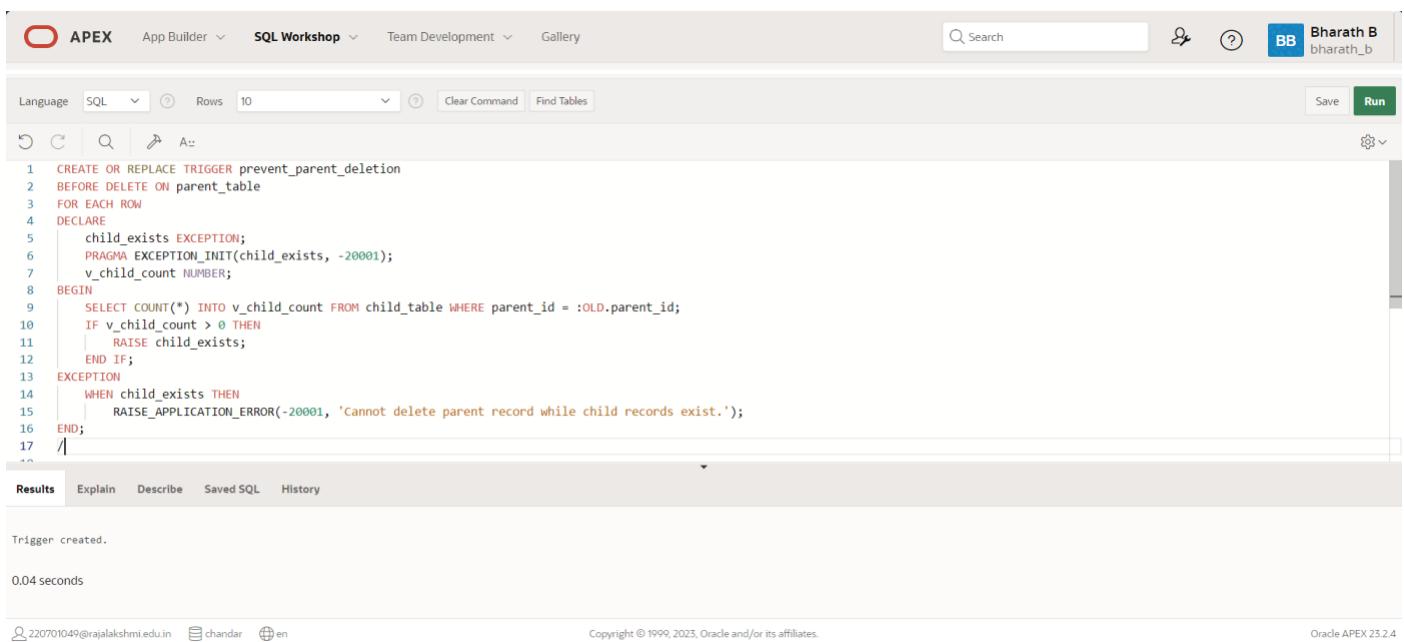
```
CREATE OR REPLACE TRIGGER  
prevent_parent_deletion BEFORE DELETE ON  
parent_table FOR EACH ROW
```

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. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The search bar contains the placeholder 'Search'. On the right, there are icons for a user profile ('Bharath B'), a help icon, and a 'Run' button.

The main workspace displays the PL/SQL code for the trigger. The code is as follows:

```
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.');
```

The 'Results' tab at the bottom shows the output: 'Trigger created.' and '0.04 seconds'. The bottom footer includes copyright information for Oracle and the APEX version.

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, tabs for 'App Builder', 'SQL Workshop' (which is active), 'Team Development', and 'Gallery' are visible. On the right side, a user profile for 'Bharath B' (bharath_b) is shown.

The main workspace displays 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, there are tabs for 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History'. The 'Results' tab is selected, showing the output: 'Trigger created.' and '0.04 seconds'. At the bottom, the URL '220701049@rajalakshmi.edu.in', the session ID 'chandar', and the page number 'en' are displayed, along with copyright information and 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'. The search bar contains 'Search'. On the right, there's a user icon for 'Bharath B' and a session ID 'bharath_b'. The main workspace displays the PL/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;
```

The 'Results' tab is selected, showing the output: 'Trigger created.' Below it, the execution time is listed as '0.05 seconds'. At the bottom, the user information '220701049@rajalakshmi.edu.in chandar en' is shown, along with copyright information 'Copyright © 1999, 2023, Oracle and/or its affiliates.' and the version '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

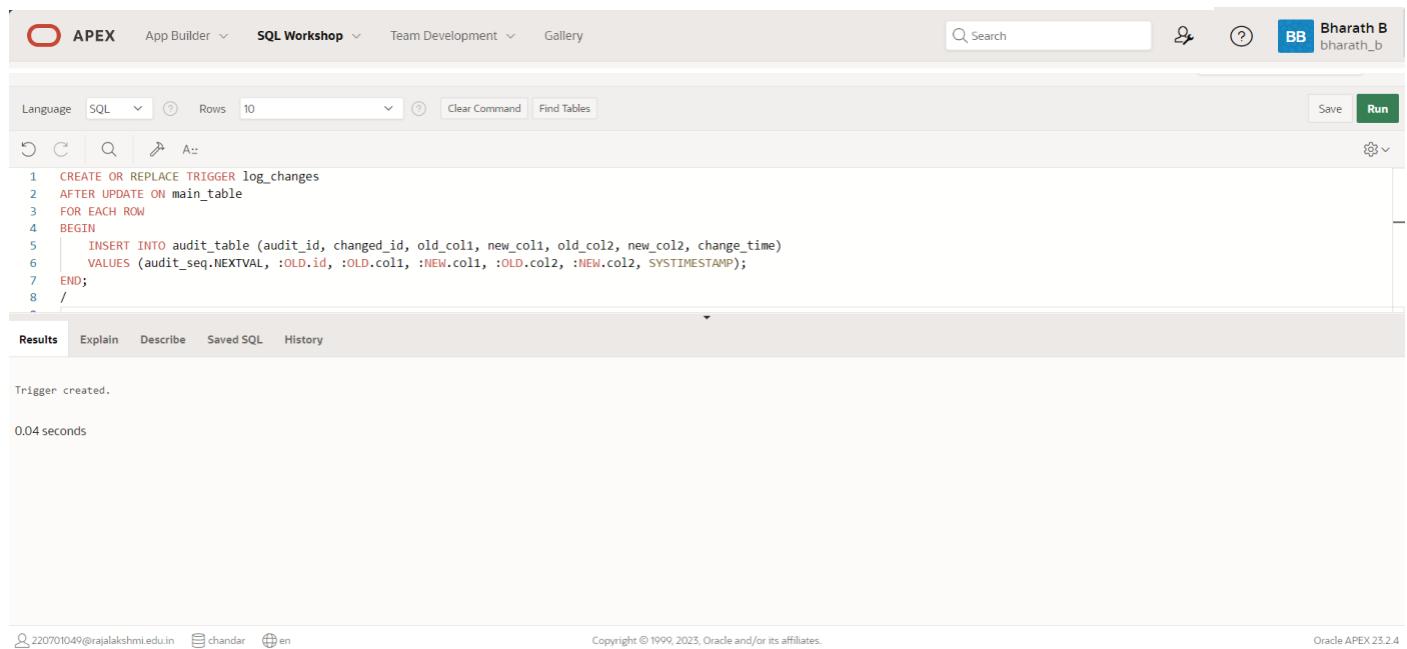
BEGIN

```
  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. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there's a search bar, a user profile for 'Bharath B bharath_b', and standard application icons.

The main workspace is a code editor with 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 /
```

Below the code editor, the results tab is selected. The output shows:

```
Trigger created.  
0.04 seconds
```

The bottom of the page displays user information (220701049@rajalakshmi.edu.in, chandar, en), copyright notice (Copyright © 1999, 2023, Oracle and/or its affiliates.), and the APEX 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. The SQL Workshop tab is selected. The main area displays the following PL/SQL code:

```

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

```

Below the code, the "Results" tab is selected, showing the message "Trigger created." and a execution time of "0.05 seconds". The bottom of the page includes copyright information and a note about Oracle APEX version 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 user has run the following SQL code to create a trigger:

```
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 pane shows the message "Trigger created." and a execution time of "0.04 seconds".

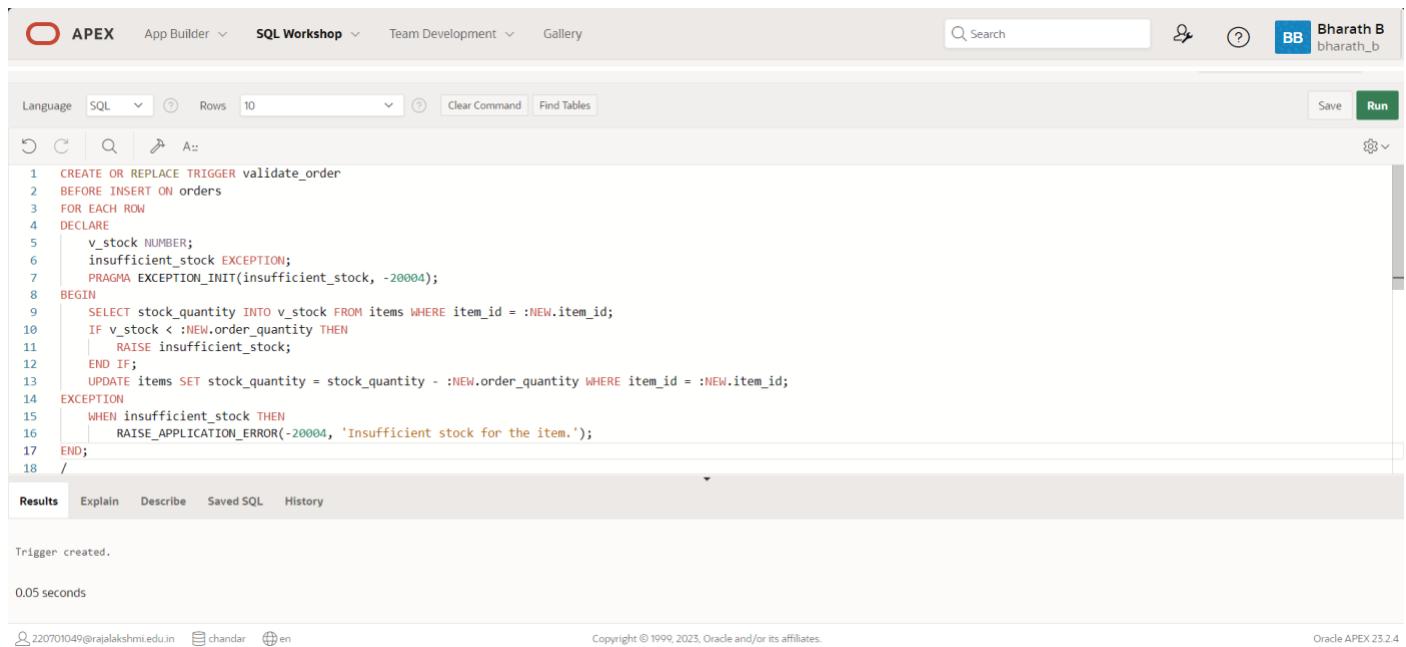
At the bottom, the footer includes the URL "220701049@rajalakshmi.edu.in", the name "chandar", and the language "en". It also states "Copyright © 1999, 2023, Oracle and/or its affiliates." and "Oracle APEX 23.2.4".

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

QUERY:

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

OUTPUT:



The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side of the header has a search bar, a help icon, and a user profile for 'Bharath B bharath_b'. The main workspace is titled 'Language: SQL' with a 'Run' button. Below the title are standard database navigation icons: back, forward, search, and refresh. The SQL command area contains the PL/SQL code for the 'validate_order' trigger. The code is numbered from 1 to 18. Lines 1 through 17 represent the trigger definition and logic, while line 18 ends the trigger with a slash. The bottom of the screen shows the results of the execution: 'Trigger created.' and a execution time of '0.05 seconds'. The 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:

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. On the left, there is a code editor window 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});
```

On the right, there is an "Output" window showing the command-line interface:

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

At the bottom of the interface, there is a decorative banner 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. On the left, there is a code editor window 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});
```

On the right, there is an "Output" window showing the command-line interface:

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

At the bottom of the interface, there is a decorative banner 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("2014-08-11T00:00:00Z") }, { restaurant_id: 1, name: 1, grades: 1 } );
```

OUTPUT:

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

```
1 db.restaurants.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});
```

In the output panel, the command is executed:

```
mycompiler_mongodb> ...  
mycompiler_mongodb>
```

[Execution complete with exit code 0]

The system tray at the bottom right shows the date and time as 5/24/2024, 11:41 PM.

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 interface. In the code editor, the following MongoDB query is written:

```
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 });
```

In the output panel, the command is executed:

```
mycompiler_mongodb> ...  
mycompiler_mongodb>
```

[Execution complete with exit code 0]

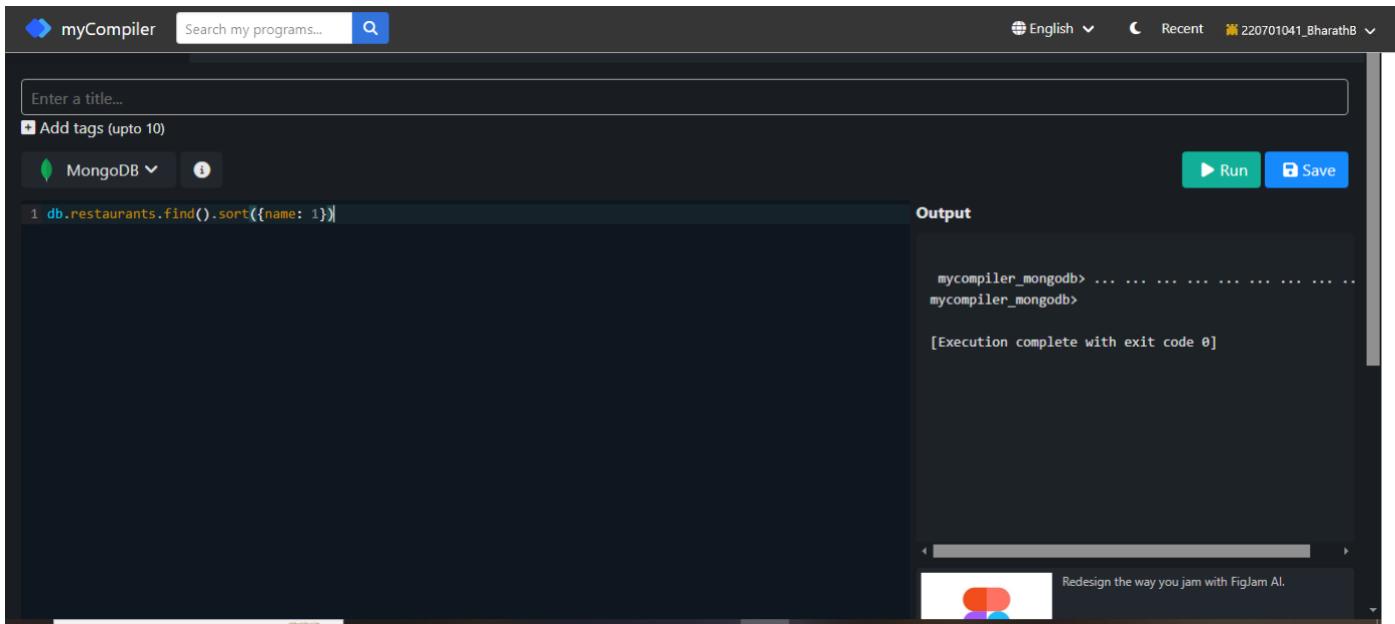
The system tray at the bottom right shows the date and time as 5/24/2024, 11:41 PM.

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 interface. In the top bar, there are icons for English, Recent, and a user profile (220701041_BharathB). Below the bar, there's a search bar labeled "Search my programs..." and a magnifying glass icon. A title input field says "Enter a title...". Below it, a "Add tags (upto 10)" button is visible. On the left, a "MongoDB" dropdown menu is open. The main area contains a code editor with the following query:

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

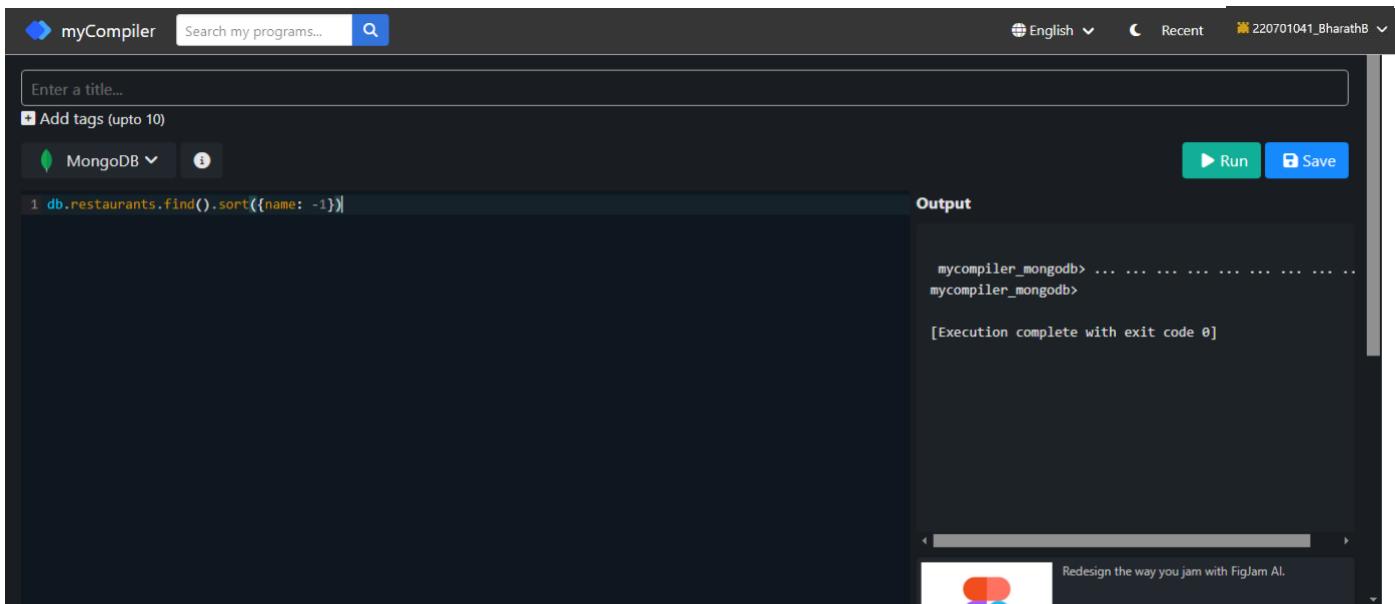
To the right of the code editor is an "Output" panel. It displays the MongoDB shell prompt "mycompiler_mongodb> ...". The output shows the results of the query, which are not explicitly visible in the screenshot. At the bottom right of the interface, there's a watermark for "Redesign the way you jam with FigJam AI." and a timestamp "FRI 11:45 PM".

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 interface. The top bar and search bar are identical to the previous screenshot. The title input field is empty. The "Add tags (upto 10)" button is present. The "MongoDB" dropdown menu is open. The code editor contains the same query as the previous screenshot:

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

The "Output" panel shows the MongoDB shell prompt "mycompiler_mongodb> ...". The output is identical to the previous screenshot, showing the results of the query. The watermark and timestamp are also present at the bottom right.

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 MongoDB interface. In the code editor, a query is written: `1 db.restaurants.find({}, {_id:0, cuisine:1, borough:1}).sort({cuisine: 1, borough: -1})`. The output window 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 MongoDB interface. In the code editor, a query is written: `1 db.restaurants.find({ "address.street": { $exists: true } }).count() == db.restaurants.count()`. The output window 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 top left, there's a logo and the text "myCompiler". A search bar says "Search my programs..." with a magnifying glass icon. On the right, there are language ("English"), recent projects ("Recent"), and user ("220701041_Bharath") dropdowns. Below the search bar is a text input field with placeholder "Enter a title..." and a "Add tags (upto 10)" button. To the right of the input field are "MongoDB" and "Run" buttons. The main area has a code editor with the following query:

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

Below the code editor is an "Output" panel with the following text:

```
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. The layout is identical to the previous one, with the "MongoDB" and "Run" buttons visible. The code editor contains the same query as above:

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

The "Output" panel 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, the following 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 execution:

```
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, the following 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 execution:

```
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 MongoDB interface. In the top bar, there are tabs for English, Recent, and a user profile. Below the bar, there's a search bar labeled "Search my programs..." and a magnifying glass icon. A text input field says "Enter a title..." and a button "+ Add tags (upto 10)" is next to it. On the left, there's a "MongoDB" dropdown and a help icon. On the right, there are "Run" and "Save" buttons. The main area has a code editor with the following MongoDB query:

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

Below the code editor is an "Output" panel showing the results of the query execution:

```
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 MongoDB interface. The setup is identical to the previous one, with tabs for English, Recent, and a user profile, a search bar, and a text input field for titles. On the left, there's a "MongoDB" dropdown and a help icon. On the right, there are "Run" and "Save" buttons. The main area has a code editor with the following MongoDB query:

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

Below the code editor is an "Output" panel showing the results of the query execution:

```
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 MongoDB interface. In the top bar, there are icons for myCompiler, search, language selection (English), recent projects, and a session ID (220701041_BharathB). Below the bar, there's a title input field ("Enter a title...") and a tags input field ("Add tags (upto 10)"). A MongoDB dropdown menu is open, showing "MongoDB". On the right, there 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 a terminal window showing the command "mycompiler_mongodb> ...". The output of the query is displayed as follows:

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

A progress bar at the bottom indicates the process is complete. A watermark for FigJam AI is visible in the bottom right corner.

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 MongoDB interface. The setup is identical to the previous screenshot, with the MongoDB dropdown menu open. The code editor contains the same query as above:

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

The terminal output is identical to the previous screenshot:

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

A progress bar at the bottom indicates the process is complete. A watermark for FigJam AI is visible in the bottom right corner.

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 the placeholder "Search my programs..." and a magnifying glass icon. To the right are language selection ("English"), recent projects ("Recent"), and a session ID ("220701041_BharathB"). Below the search bar is a text input field with the placeholder "Enter a title...". Underneath it is a button labeled "+ Add tags (upto 10)". On the left, there's a dropdown menu for "MongoDB" with a green leaf icon and a help icon. On the right, there are "Run" and "Save" buttons. The main area displays a MongoDB query and its output. The query is:1\$lt: 5 }, cuisine: { \$nin: ["American", "Chinese"] } }, { _id: 0, name: 1, borough: 1, address: 1, cuisine: 1 })

The output shows the results of the query in the MongoDB shell:mycompiler_mongodb> ...
mycompiler_mongodb>
[Execution complete with exit code 0]

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. At the top, there's a search bar with the placeholder "Search my programs..." and a magnifying glass icon. To the right are language selection ("English"), recent projects ("Recent"), and a session ID ("220701041_BharathB"). Below the search bar is a text input field with the placeholder "Enter a title...". Underneath it is a button "+ Add tags (upto 10)". On the left, there's a dropdown menu for "MongoDB" with a dropdown arrow and an information icon. The main area contains a code editor with the following MongoDB query:

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

Below the code editor is a green "Run" button with a play icon and a blue "Save" button with a save icon. To the right of the code editor is a terminal window titled "Output" showing the results of the query:

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

At the bottom right, there's an advertisement for FigJam AI with the text "Redesign the way you jam with FigJam AI." and "ADS VIA CARBON".

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:

The screenshot shows the myCompiler MongoDB interface. At the top, there's a search bar with "Search my programs..." and a magnifying glass icon. On the right, there are language selection ("English"), recent projects ("Recent"), and a user ID ("220701041_BharathB"). Below the search bar is a text input field with placeholder "Enter a title...". Underneath it is a "Add tags (upto 10)" button. A dropdown menu for "MongoDB" is open, showing "MongoDB" and an information icon. To the right are "Run" and "Save" buttons. The main area contains a code snippet and its output. The code is:`1grades.score": { $in: [2, 6] }, borough: "Manhattan" }, { _id: 0, name: 1, borough: 1, address: 1, cuisine: 1 }) Output`

The output shows the command being run: "mycompiler_mongodb> ...". It then displays the results of the query, which are empty. Below the results, a message says "[Execution complete with exit code 0]". At the bottom right, there's an advertisement for FigJam AI.

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:

The screenshot shows the myCompiler MongoDB interface. At the top, there's a search bar with "Search my programs..." and a magnifying glass icon. On the right, there are language selection ("English"), recent projects ("Recent"), and a user ID ("220701041_BharathB"). Below the search bar is a text input field with placeholder "Enter a title...". Underneath it is a "Add tags (upto 10)" button. A dropdown menu for "MongoDB" is open, showing "MongoDB" and an information icon. To the right are "Run" and "Save" buttons. The main area contains a code snippet and its output. The code is:`1rough: "Brooklyn"]}, "grades.score": { $in: [2, 6] } }, { _id: 0, name: 1, borough: 1, address: 1, cuisine: 1 }) Output`

The output shows the command being run: "mycompiler_mongodb> ...". It then displays the results of the query, which are empty. Below the results, a message says "[Execution complete with exit code 0]". At the bottom right, there's an advertisement for FigJam AI.

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:

The screenshot shows the myCompiler interface. At the top, there's a search bar with placeholder text "Search my programs..." and a magnifying glass icon. To the right are language selection ("English"), recent projects ("Recent"), and a session ID ("220701041_BharathB"). Below the search bar is a text input field with placeholder "Enter a title...". Underneath it is a button labeled "Add tags (upto 10)" with a checkmark icon. On the left, there's a dropdown menu for "MongoDB" with a green gear icon and a help icon. On the right, there are two buttons: "Run" (green with white play icon) and "Save" (blue with white save icon). The main area contains a code editor with the following MongoDB query:

```
1{ $in: [2, 6] } }, { cuisine: { $ne: "American" } }] }, { _id: 0, name: 1, borough: 1, address: 1, cuisine: 1 })
```

Next to the code is a bold "Output" label. Below the code editor is a terminal window showing the command "mycompiler_mongodb> ...". The terminal output is as follows:

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

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:

The screenshot shows the myCompiler interface. At the top, there's a search bar with placeholder text "Search my programs..." and a magnifying glass icon. To the right are links for "English", "Recent", and a timestamp "220701041_BharathB". Below the search bar is a text input field with placeholder "Enter a title...". Underneath it is a checkbox labeled "Add tags (upto 10)" with an info icon. On the left, there's a MongoDB dropdown with an info icon and a "Run" button. The main area contains a code editor with the following query:

```
1] } }, { cuisine: { $nin: ["American", "Chinese"] } } }, { _id: 0, name: 1, borough: 1, address: 1, cuisine: 1 })| Output
```

To the right of the code editor is a terminal window showing the output of the command:

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

At the bottom right, there's a small advertisement for FigJam AI.

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:

myCompiler Search my programs... 

Enter a title...

Add tags (upto 10) 

MongoDB  

```
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

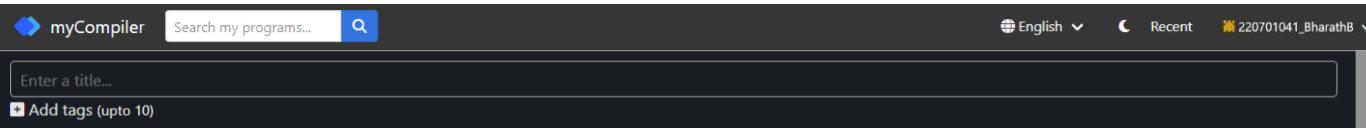
DATE:

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. At the top, there's a search bar with the placeholder "Search my programs..." and a magnifying glass icon. To the right are language selection ("English"), recent projects ("Recent"), and a timestamp ("220701041_BharathB"). Below the search bar is a title input field with the placeholder "Enter a title...". Underneath it is a checkbox labeled "Add tags (upto 10)" with a dropdown menu showing "MongoDB" and an information icon. On the far right are "Run" and "Save" buttons. The main area contains a command-line interface window with the following text:

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

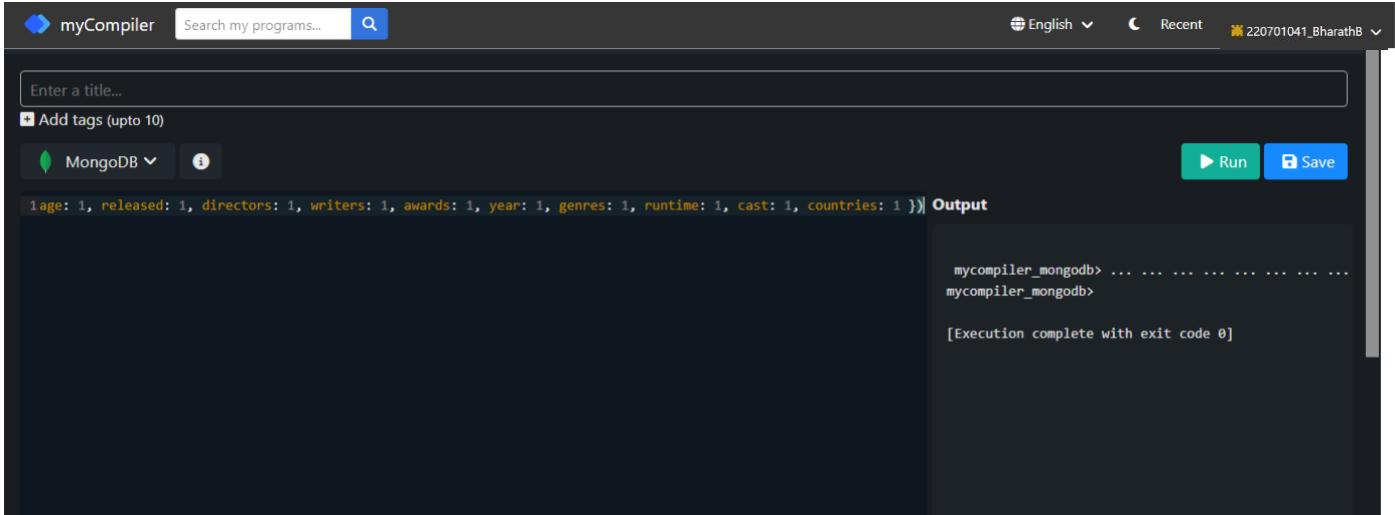
The screenshot shows the myCompiler interface. At the top, there's a search bar with "Search my programs..." and a magnifying glass icon. On the right, there are language selection ("English"), recent projects ("Recent"), and a session ID ("220701041_BharathB"). Below the search bar is a text input field with placeholder "Enter a title...". Underneath it is a checkbox labeled "Add tags (upto 10)". To the left, there's a dropdown menu set to "MongoDB" with an info icon next to it. On the right, there are "Run" and "Save" buttons. The main area contains a code snippet: `l:age: 1, released: 1, directors: 1, writers: 1, awards: 1, year: 1, genres: 1, runtime: 1, cast: 1, countries: 1 })`. To its right is a "Output" button. The output panel shows the command "mycompiler_mongodb> ...". Below the command is the message "[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:



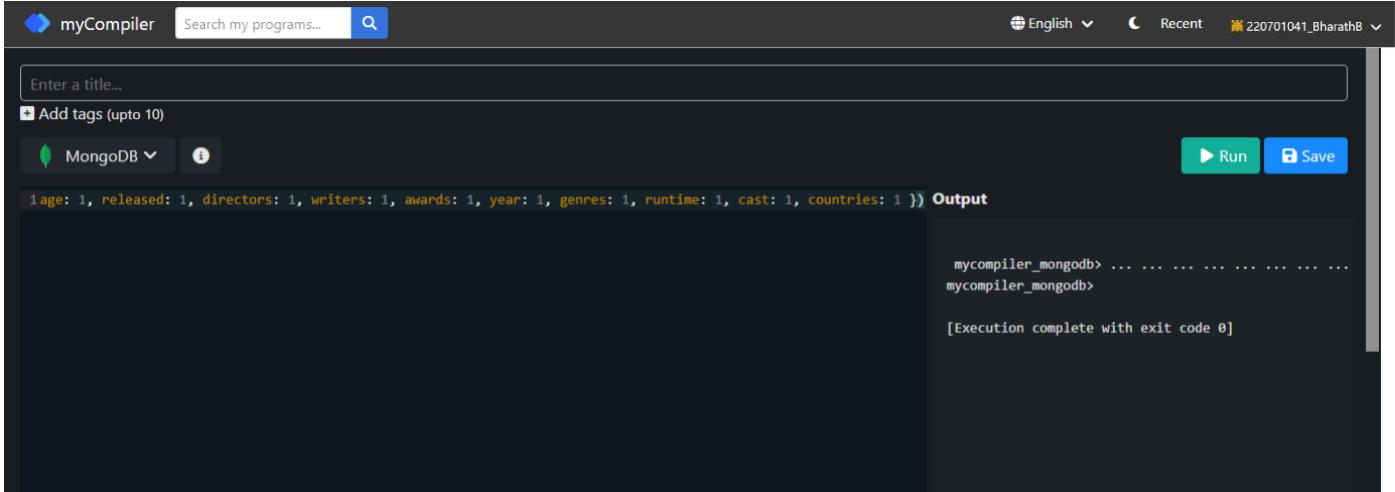
The screenshot shows the myCompiler interface. At the top, there's a search bar for programs and a language selector set to English. Below the search bar is a text input field labeled 'Enter a title...' and a checkbox for 'Add tags (upto 10)'. A MongoDB dropdown menu is open, showing 'MongoDB'. On the right side of the interface, there are 'Run' and 'Save' buttons. The main area is titled 'Output' and contains the MongoDB command: 'db.movies.find({ genres: "Short" })'. Below the command, the terminal window shows the response: 'mycompiler_mongodb> ...'. The terminal also displays the message '[Execution complete with exit code 0]'. The background of the interface is dark.

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. 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)'. A MongoDB dropdown menu is open, showing 'MongoDB'. On the right side of the interface, there are 'Run' and 'Save' buttons. The main area is titled 'Output' and contains the MongoDB command: 'db.movies.find({ directors: "William K.L. Dickson" })'. Below the command, the terminal window shows the response: 'mycompiler_mongodb> ...'. The terminal also displays the message '[Execution complete with exit code 0]'. The background of the interface is dark.

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 with a dark theme. At the top, there's a search bar with placeholder text "Search my programs..." and a magnifying glass icon. To the right are language selection ("English"), recent projects ("Recent"), and a session ID ("220701041_BharathB"). Below the search bar is a text input field with placeholder "Enter a title...". Underneath it is a checkbox labeled "Add tags (upto 10)". On the left, there's a MongoDB connection dropdown set to "MongoDB" with a dropdown arrow, and a help icon (info symbol). On the right, there are two buttons: "Run" (green background with white play icon) and "Save" (blue background with white save icon). The main area displays a command-line interface output. The command entered was "age: 1, released: 1, directors: 1, writers: 1, awards: 1, year: 1, genres: 1, runtime: 1, cast: 1, countries: 1 } | Output". The output shows the command prompt "mycompiler_mongodb>" followed by several dots indicating continuation. Below the prompt is the message "[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:

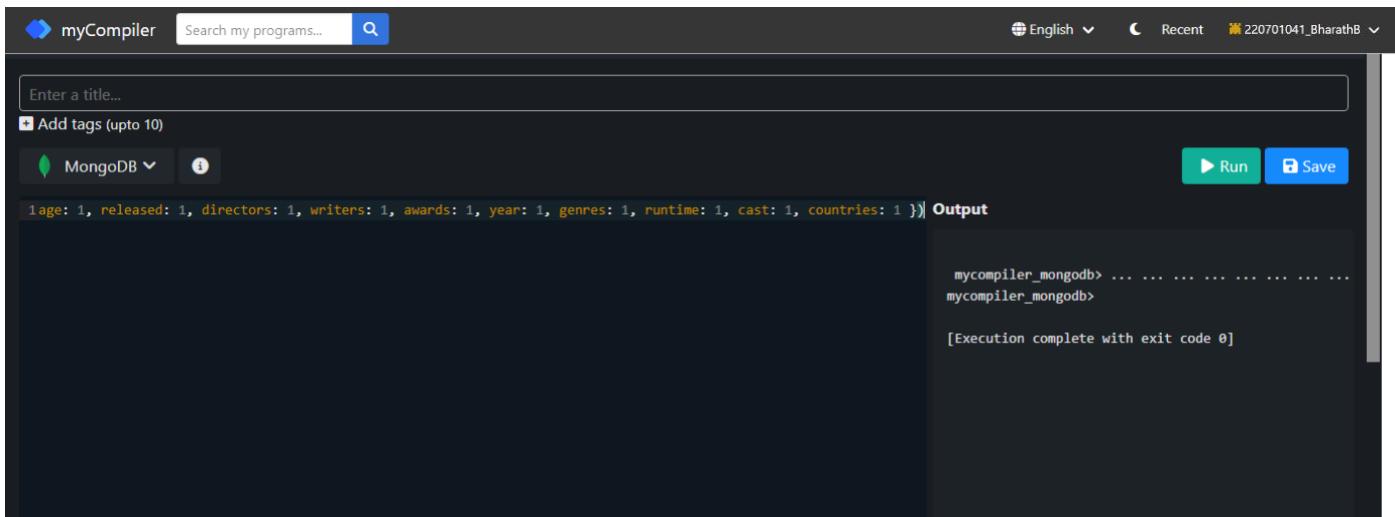
The screenshot shows the myCompiler interface. At the top, there's a search bar with the placeholder "Search my programs..." and a magnifying glass icon. To the right are language selection ("English"), recent projects ("Recent"), and a session ID ("220701041_BharathB"). Below the search bar is a text input field with the placeholder "Enter a title...". Underneath it is a checkbox labeled "Add tags (upto 10)". On the left, there's a "MongoDB" dropdown menu with a green leaf icon and a "Save" button. The main area contains a code snippet: `age: 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 is a "Run" button with a play icon and a "Save" button with a disk icon. Below the code is a section titled "Output" containing the command `mycompiler_mongodb> ...` followed by "[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:



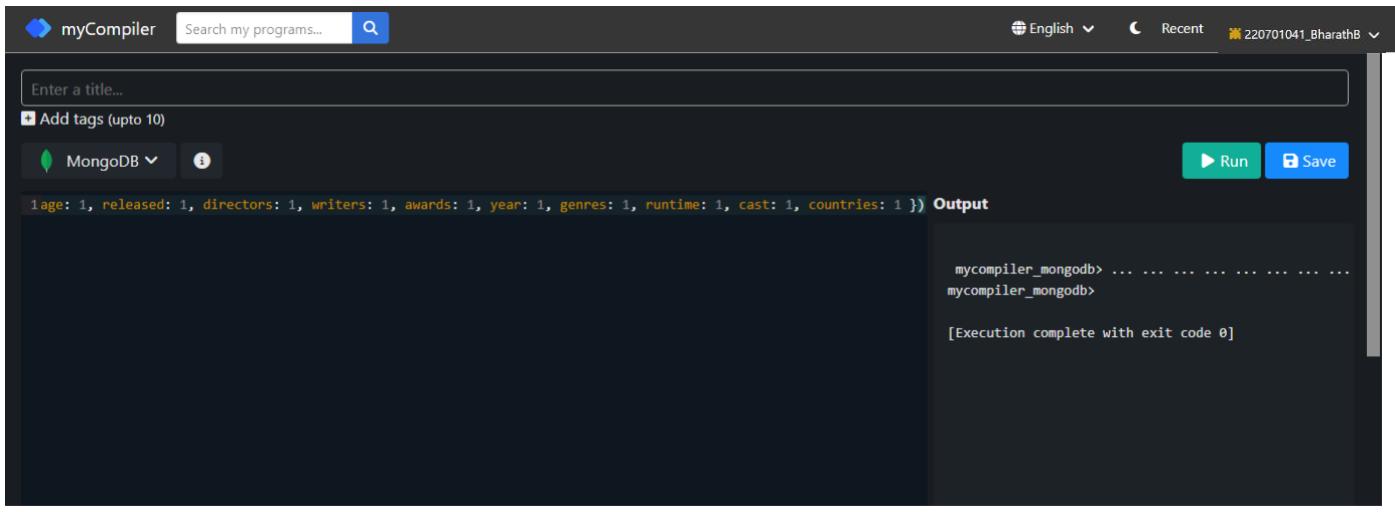
The screenshot shows the myCompiler interface. At the top, there's a search bar for programs and a language dropdown set to English. Below the search bar is a text input field labeled "Enter a title...". Underneath it, there's a "Add tags (upto 10)" button. On the left, there's a "MongoDB" dropdown and a help icon. On the right, there are "Run" and "Save" buttons. The main area is titled "Output" and contains the following text:
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]

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:



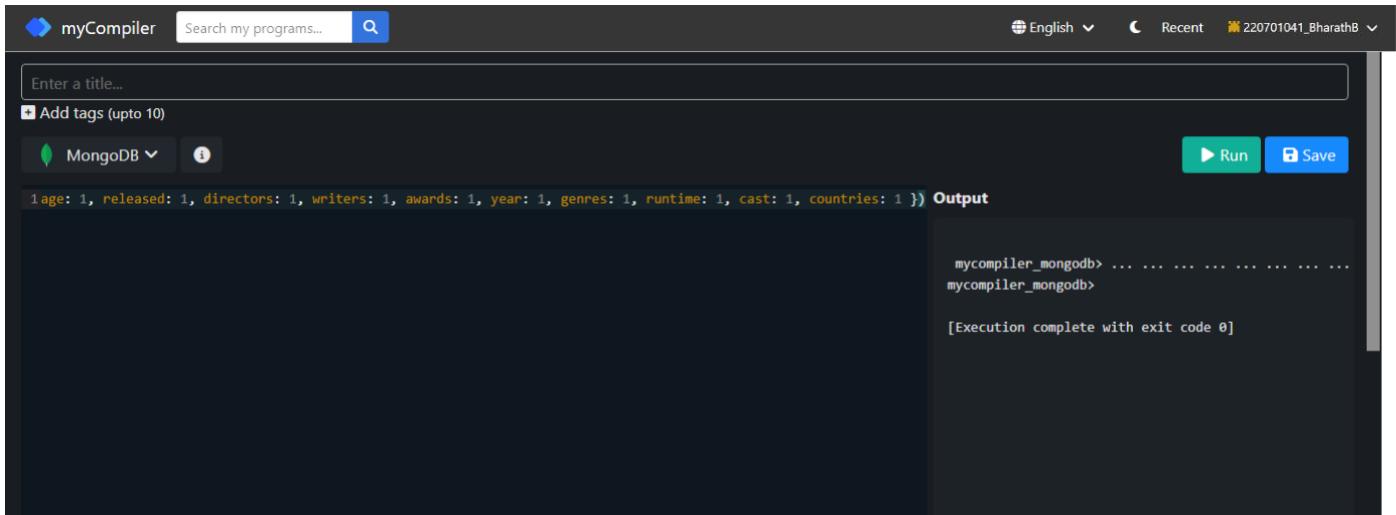
The screenshot shows the myCompiler interface. At the top, there's a search bar for programs and a language dropdown set to English. Below the search bar is a text input field labeled "Enter a title...". Underneath it, there's a "Add tags (upto 10)" button. On the left, there's a "MongoDB" dropdown and a help icon. On the right, there are "Run" and "Save" buttons. The main area is titled "Output" and contains the following text:
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]

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:



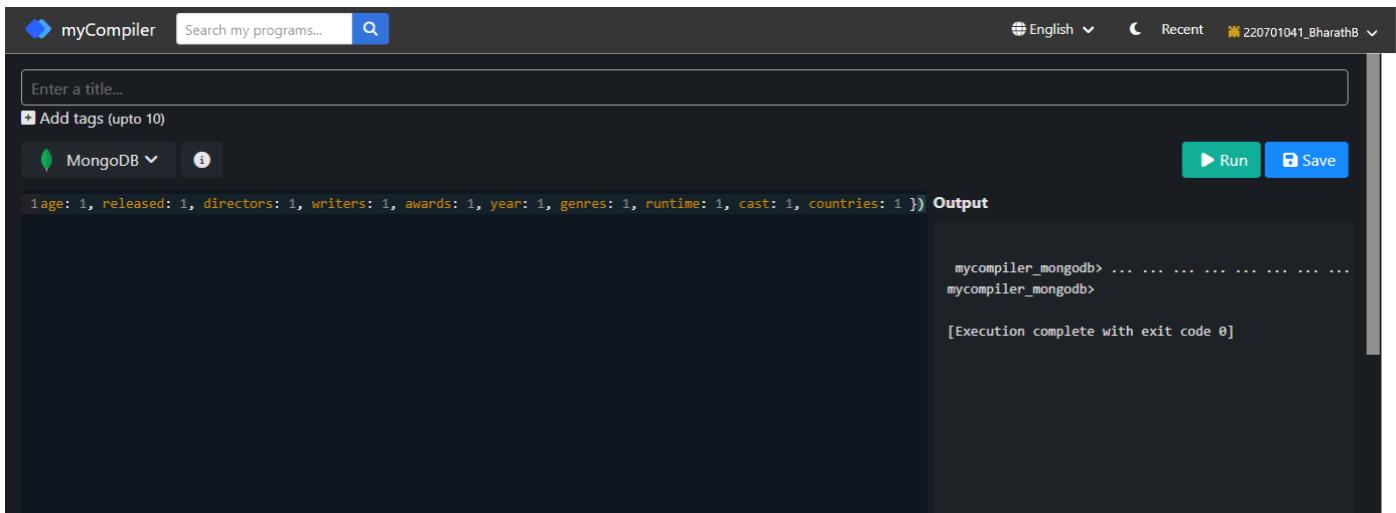
The screenshot shows the myCompiler interface. At the top, there's a search bar for programs and a language selection dropdown set to English. Below the search bar is a text input field labeled "Enter a title..." and a button to "Add tags (upto 10)". On the left, there's a MongoDB connection status icon and a help button. On the right, there are "Run" and "Save" buttons. The main area is titled "Output" and contains the MongoDB command: "db.movies.find({ 'tomatoes.viewer.rating': { \$gt: 4 } })". The output window shows the command being run and the response: "mycompiler_mongodb> ...". It also indicates "[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. The setup is identical to the previous one: English selected, "Enter a title..." input field, "Add tags (upto 10)" button, MongoDB connection status, and "Run" and "Save" buttons. The "Output" window displays the MongoDB command: "db.movies.find({ 'awards.wins': { \$gt: 0 } })". The output shows the command being run and the response: "mycompiler_mongodb> ...". It also indicates "[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:

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:

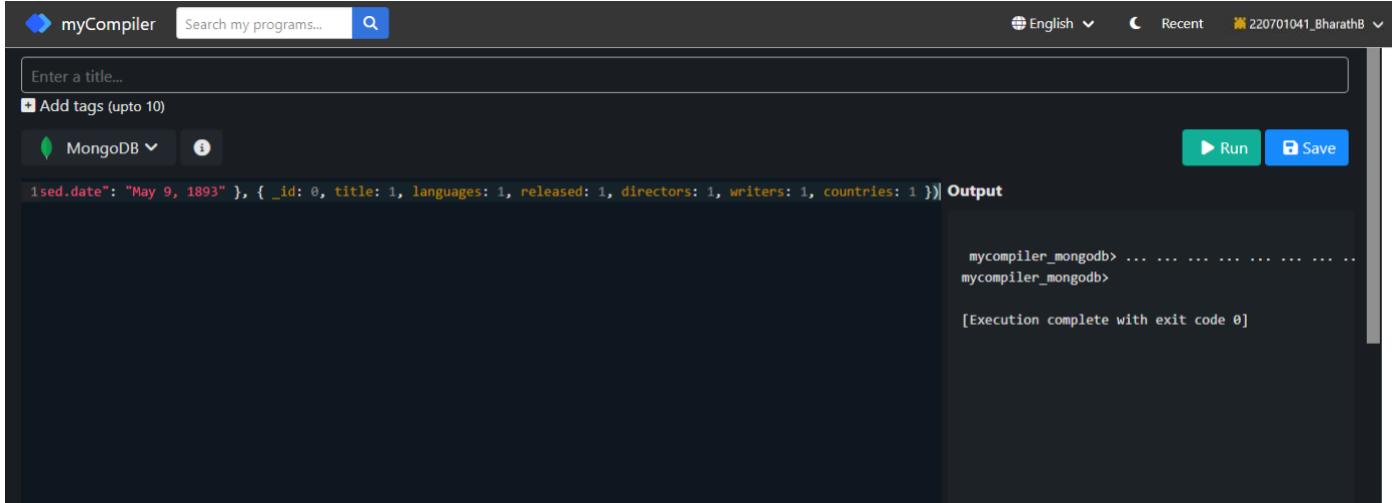
The screenshot shows the myCompiler interface. At the top, there's a search bar with 'Search my programs...' and a magnifying glass icon. On the right, there are language selection ('English'), recent projects ('Recent'), and a user ID ('220701041_BharathB'). Below the header, a text input field says 'Enter a title...'. Underneath it, a checkbox labeled 'Add tags (upto 10)' is checked. To the left, there's a MongoDB dropdown and an info icon. On the right, there are 'Run' and 'Save' buttons. The main area contains a code snippet: `languages: 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 is a 'Output' button. The output panel shows the command `mycompiler_mongodb> ...` followed by '[Execution complete with exit code 0]'. A status bar at the bottom indicates the file is saved.

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 icons for English, Recent, and a user profile (220701041_BharathB). Below the bar, there's a search bar labeled "Search my programs..." and a magnifying glass icon. A text input field says "Enter a title...". Underneath it, there's a "Add tags (upto 10)" button and a dropdown menu set to "MongoDB". On the right side of the interface, there are "Run" and "Save" buttons. The main area contains a code editor with the following MongoDB query:

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

Below the code editor, the terminal window shows the command "mycompiler_mongodb> ...". The output of the command is:

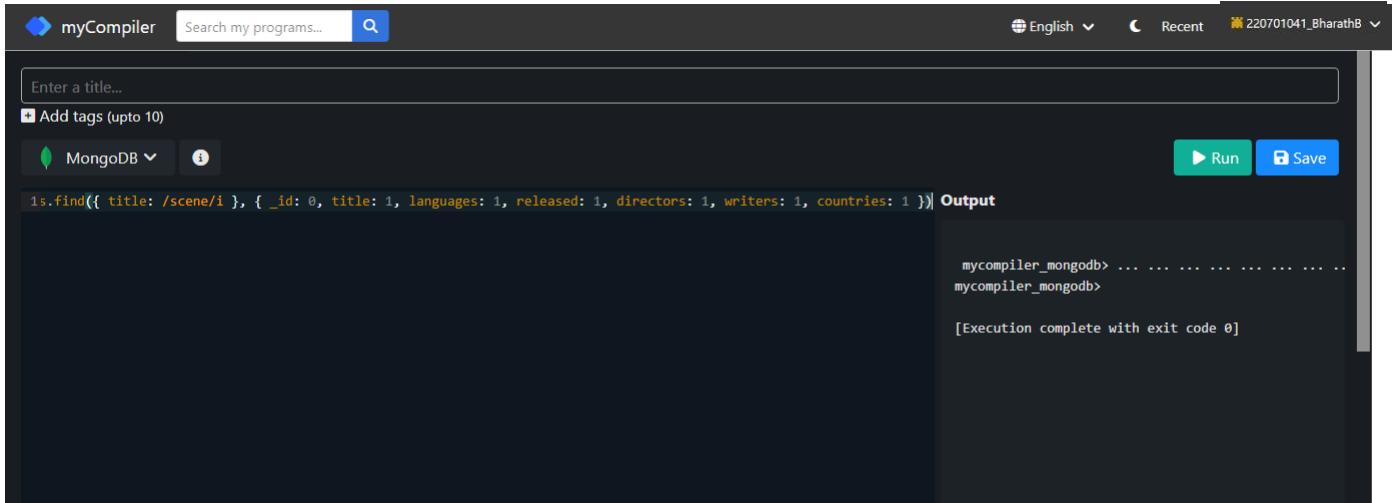
```
mycompiler_mongodb> ...  
mycompiler_mongodb>  
[Execution complete with exit code 0]
```

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 top bar and search bar are identical to the previous screenshot. The text input field says "Enter a title...". Below it, there's a "Add tags (upto 10)" button and a dropdown menu set to "MongoDB". On the right side, there are "Run" and "Save" buttons. The main area contains a code editor with the following MongoDB query:

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

Below the code editor, the terminal window shows the command "mycompiler_mongodb> ...". The output of the command is:

```
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: