

Creating and Managing Tables

EX_NO:1

DATE:

S

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

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle Application Express interface. In the top navigation bar, the 'SQL Commands' tab is selected. The main area contains the following SQL code:

```
create table dept033(
  id number(7),
  name varchar(25)
);
```

Below the code, the output shows:

Table created.
0.03 seconds

The bottom status bar displays system information including the date (20-02-2024), time (08:42), and weather (24°C, Mostly sunny).

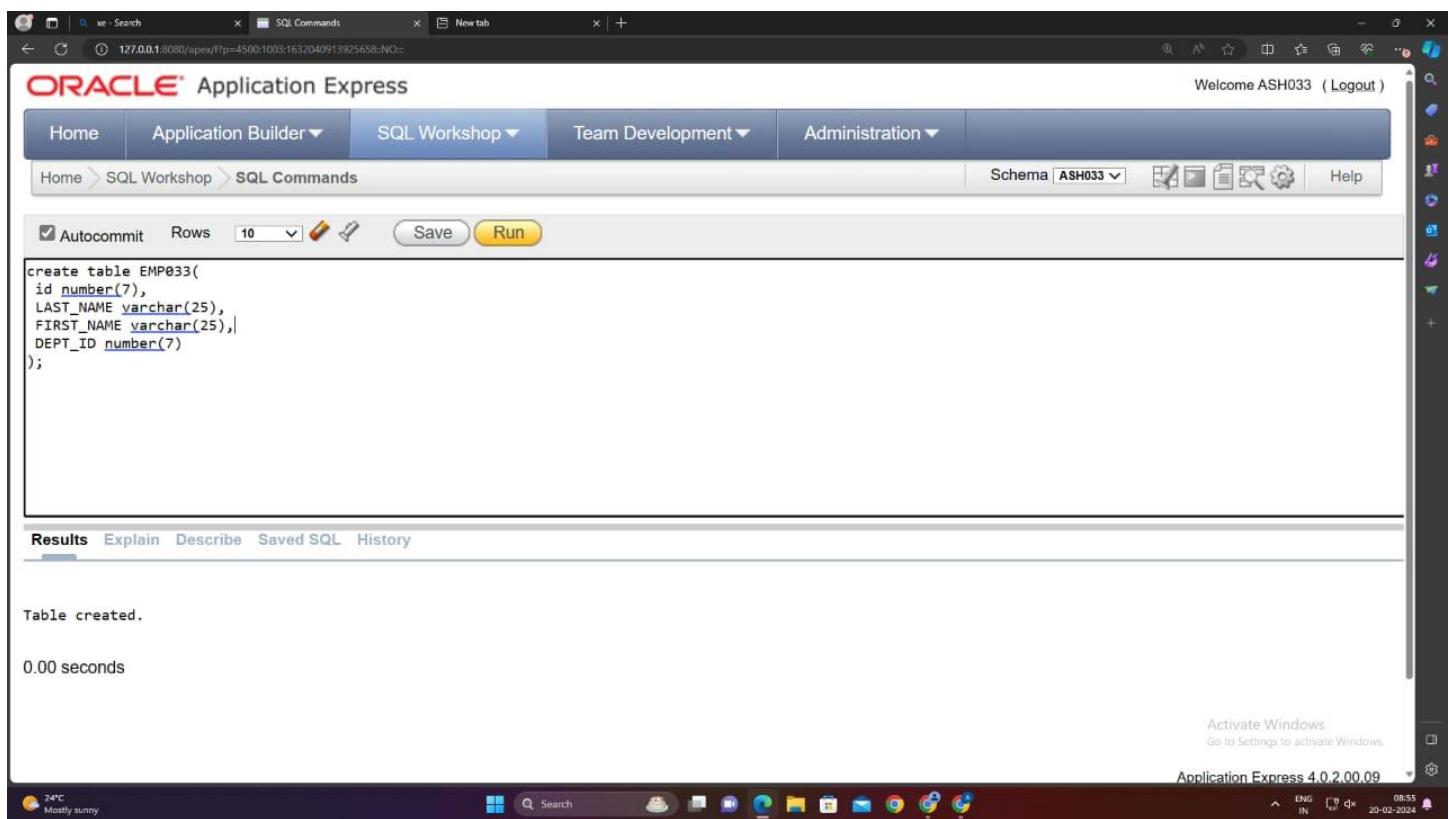
2.Create the EMP table based on the following instance chart. Confirm that the table is created.

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

QUERY:

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

OUTPUT:



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

```
create table EMP033(
  id number(7),
  LAST_NAME varchar(25),
  FIRST_NAME varchar(25),
  DEPT_ID number(7)
);
```

The 'Run' button is highlighted. Below the code, the results show:

Table created.
0.00 seconds

At the bottom right, there is a message: "Activate Windows Go to Settings to activate Windows".

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 Application Express interface in a web browser. The title bar indicates the URL is 127.0.0.1:8080/apex/f?p=4500:1003:1632040913925658:NO=. The main window is titled "ORACLE® Application Express". The navigation bar includes "Home", "Application Builder", "SQL Workshop" (which is selected), "Team Development", and "Administration". The top right shows "Welcome ASH033 (Logout)". The SQL Workshop toolbar includes "Autocommit" checked, "Rows" set to 10, and "Save" and "Run" buttons. The SQL editor contains the command: `ALTER TABLE EMP033 MODIFY(LAST_NAME varchar(50));`. Below the editor, the results pane shows the output: "Table altered." and "0.00 seconds". The bottom status bar displays system information: "24°C Mostly sunny", "Activate Windows Go to Settings to activate Windows.", "Application Express 4.0.2.00.09", and a date/time stamp "20-02-2024 09:02".

```
ALTER TABLE EMP033 MODIFY(LAST_NAME varchar(50));
```

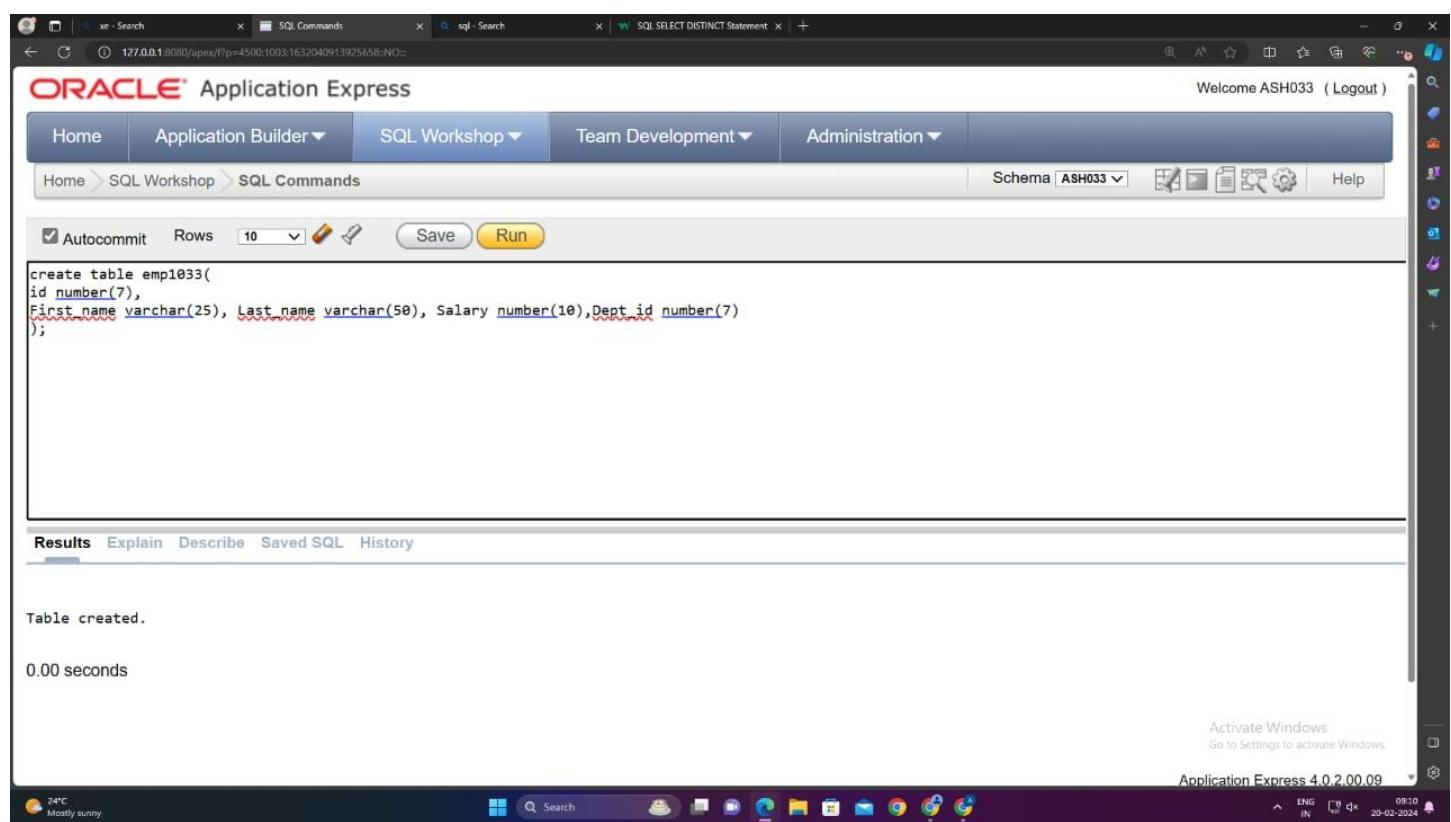
Table altered.
0.00 seconds

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

QUERY:

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

OUTPUT:



The screenshot shows the Oracle Application Express interface. In the SQL Commands tab, a SQL statement is being run to create a new table:

```
create table emp1033(
id number(7),
First_name varchar2(25), Last_name varchar(50), Salary number(10),Dept_id number(7)
);
```

The output window shows the results of the query:

```
Table created.  
0.00 seconds
```

The status bar at the bottom right indicates "Activate Windows Go to Settings to activate Windows".

5.Drop the EMP table.

QUERY:

```
Drop table emp;
```

OUTPUT:

The screenshot shows the Oracle Application Express interface. In the top navigation bar, the 'SQL Commands' tab is selected. The main workspace contains the following SQL command:

```
Drop table EMP1033;
```

Below the command, the output window displays the results:

```
Table dropped.  
0.06 seconds
```

The status bar at the bottom right indicates the application version is 4.0.2.00.09 and the date is 20-02-2024.

6.Rename the EMPLOYEES2 table as EMP.

QUERY:

Rename employees2 to emp;

OUTPUT:

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

```
alter table employees2
Drop column first_name;
```

The code is highlighted with red underlines under 'employees2' and 'first_name'. Below the code, the results show:

Table altered.
0.06 seconds

The status bar at the bottom right indicates the application version is 4.0.2.0.0.09.

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. In the top navigation bar, 'APEX' is selected. The main area is titled 'SQL Commands'. The schema dropdown shows 'WKSP_INDEX033'. The code editor contains the following SQL command:

```
1 comment on table DEPT is 'department info';
```

The results tab shows the output of the command:

```
Statement processed.
```

Below the results, it says '0.04 seconds'.

At the bottom, the browser address bar shows 'index_033.en'. The status bar indicates 'Oracle APEX 23.2.4'.

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

QUERY:

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

OUTPUT:

xe : Search SQL Commands sql : Search SQL SELECT DISTINCT Statement

Welcome ASH033 (Logout)

ORACLE Application Express

Home Application Builder ▾ SQL Workshop ▾ Team Development ▾ Administration ▾

Home > SQL Workshop > SQL Commands

Schema: ASH033

Autocommit Rows 10 Save Run

```
alter table empnew
Drop column First_name;
```

Results Explain Describe Saved SQL History

Table altered.

0.06 seconds

Activate Windows
Go to Settings to activate Windows.

Application Express 4.0.2.00.09

24°C Mostly sunny

Search Home Help

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

Viva(5)	
Total (15)	
Faculty Signature	

RESULT:

MANIPULATING DATA

EX_NO:2

DATE:

1.Create MY_EMPLOYEE table with the following structure

NAME	NULL?	TYPE
ID	Not null	Number(4)
Last_name		Varchar(25)
First_name		Varchar(25)
Userid		Varchar(25)
Salary		Number(9,2)

QUERY:

Create table MY_EMP(ID number(4) not null,last_name varchar(25),first_name varchar(25),user_id varchar(25),salary number(9,2));

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. In the top navigation bar, there are tabs for 'App Builder', 'SQL Workshop' (which is selected), 'Team Development', and 'Gallery'. On the right side, there is a search bar, a user icon, and session information ('ashwin_055 a'). Below the navigation, the 'Schema' dropdown is set to 'WKSP_INDEX055'. The main area is titled 'SQL Commands' and contains a code editor with the following SQL script:

```

1 create table MY_EMP33(
2   id number(4),
3   Last_name varchar(25),
4   First_name varchar(25),
5   userid varchar(25),
6   salary number(9,2)
7 );

```

Below the code editor, there are buttons for 'Save' and 'Run'. The results tab is selected, showing the output of the command:

Table created.
0.03 seconds

The status bar at the bottom indicates 'Activate Windows Go to Settings to activate Windows.' and shows system information like '28°C Partly sunny'.

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

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

QUERY:

```
Insert into MY_EMP values(1,'Patel','Ralph','rpatel',895);
Insert into MY_EMP values(2,'Dancs','Betty','bdancs',860);
```

OUTPUT:

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

```
1 create table MY_EMP33(
2   id number(4),
3   Last_name varchar(25),
4   First_name varchar(25),
5   userid varchar(25),
6   salary number(9,2)
7 );
8
9
10 INSERT INTO MY_EMP33 VALUES(1,'patel','ralph','rpatel',895);
11 INSERT INTO MY_EMP33 VALUES(2,'dancs','betty','bdancs',860);
12 INSERT INTO MY_EMP33 VALUES(3,'biril','ben','bbiril',1100);
13 INSERT INTO MY_EMP33 VALUES(1,'patel','ralph','rpatel',895);
14 INSERT INTO MY_EMP33 VALUES(1,'patel','ralph','rpatel',895);
15 select * from MY_EMP33
```

Below the code, the 'Results' tab is active, displaying the data from the table:

ID	LAST_NAME	FIRST_NAME	USERID	SALARY
3	birl	ben	bbiril	1100
4	newman	chad	Cnewman	750
1	singh	patel	patel	200000
1	patel	ralph	rpatel	895
1	patel	ralph	rpatel	895
2	dancs	betty	bdancs	860

The status bar at the bottom indicates the user's email (220701035@rebelkshvms.edu.in), the session ID (index_055), and the system date and time (23-02-2024 11:17).

3.Display the table with values.

QUERY:

```
Select * from MY_EMP;
```

OUTPUT:

```

1 create table MY_EMP33(
2   id number(4),
3   last_name varchar(25),
4   first_name varchar(25),
5   user_id varchar(25),
6   salary number(9,2)
7 );
8
9
10 INSERT INTO MY_EMP33 VALUES(1,'patel','ralph','rpatel',895);
11 INSERT INTO MY_EMP33 VALUES(2,'dancs','betty','bdancs',860);
12 INSERT INTO MY_EMP33 VALUES(3,'birI','ben','bbiri',1100);
13 INSERT INTO MY_EMP33 VALUES(1,'patel','ralph','rpatel',895);
14 INSERT INTO MY_EMP33 VALUES(1,'patel','ralph','rpatel',895);
15 select * from MY_EMP33

```

Results

ID	LAST_NAME	FIRST_NAME	USERID	SALARY
3	birI	ben	bbiri	1100
4	newman	chad	Cnewman	750
1	singh	patel	patel	200000
1	patel	ralph	rpatel	895
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_EMP values(3,'Biri','Ben','bbiri',1100);

Insert into MY_EMP values(4,'Newman','chad','Cnewman',750);

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. In the top navigation bar, there are tabs for App Builder, SQL Workshop (selected), Team Development, and Gallery. The main area is titled "SQL Commands". The "Language" dropdown is set to "SQL". The "Rows" dropdown is set to "10". There are buttons for "Clear Command" and "Find Tables". On the right side, there are buttons for "Save" and "Run". The schema is set to "WKSP_INDEX055".

```

1 create table MY_EMP33(
2   id number(4),
3   Last_name varchar(25),
4   First_name varchar(25),
5   userid varchar(25),
6   salary number(9,2)
7 );
8 |
9
10 INSERT INTO MY_EMP33 VALUES(1,'patel','ralph','rpatel',895);
11 INSERT INTO MY_EMP33 VALUES(2,'dancs','betty','bdancs',860);
12 INSERT INTO MY_EMP33 VALUES(3,'bir','ben','bbir',1100);
13 INSERT INTO MY_EMP33 VALUES(1,'patel','ralph','rpatel',895);
14 INSERT INTO MY_EMP33 VALUES(1,'patel','ralph','rpatel',895);
15 select * from MY_EMP33

```

The results section shows the data inserted into the MY_EMP33 table:

ID	LAST_NAME	FIRST_NAME	USERID	SALARY
3	bir	ben	bbir	1100
4	newman	chad	Cnewman	750
1	singh	patel	patel	200000
1	patel	ralph	rpatel	895
1	patel	ralph	rpatel	895
2	dancs	betty	bdancs	860

At the bottom of the results table, there are two messages: "Activate Windows" and "Go to Settings to activate Windows".

5. Make the data additions permanent.

QUERY:

Select * from MY_EMP;

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. In the top-left corner, there are tabs for 'SQL Commands' and 'SQL Commands'. The URL in the browser bar is apex.oracle.com/pls/apex//sql-workshop/sqlcommandprocessor?session=843821445241. The main area displays the following SQL code:

```

1 create table MY_EMP33(
2   id number(4),
3   Last_name varchar(25),
4   First_name varchar(25),
5   user_id varchar(25),
6   salary number(9,2)
7 );
8
9
10 INSERT INTO MY_EMP33 VALUES(1,'patel','ralph','rpatel',895);
11 INSERT INTO MY_EMP33 VALUES(2,'dancs','betty','bdancs',860);
12 INSERT INTO MY_EMP33 VALUES(3,'birli','ben','bbirli',1100);
13 INSERT INTO MY_EMP33 VALUES(1,'patel','ralph','rpatel',895);
14 INSERT INTO MY_EMP33 VALUES(1,'patel','ralph','rpatel',895);
15 select * from MY_EMP33

```

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

ID	LAST_NAME	FIRST_NAME	USERID	SALARY
3	birli	ben	bbirli	1100
4	newman	chad	Cnewman	750
1	singh	patel	patel	200000
1	patel	ralph	rpatel	895
1	patel	ralph	rpatel	895
2	dancs	betty	bdancs	860

The bottom status bar shows the session ID 220703055@irepolokahri.edu.in, the schema WKSP_INDEX055, and the date 23-02-2024.

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

QUERY:

Update MY_EMP set last_name = 'Drexler' where ID = '3'

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. In the top navigation bar, there are tabs for Oracle APEX Request, SQL Commands, and Sent Mail. The main workspace is titled "SQL Commands" and contains the following SQL code:

```
1 user define cursor(c1) as
2 salary number(9,2)
3 );
4
5 INSERT INTO MY_EMP33 VALUES(1,'patel','ralph','rpatel',895);
6 INSERT INTO MY_EMP33 VALUES(2,'dances','betty','bdances',860);
7 INSERT INTO MY_EMP33 VALUES(3,'biri','ben','bbiri',1100);
8 INSERT INTO MY_EMP33 VALUES(1,'patel','ralph','rpatel',895);
9 select * from MY_EMP33
10
11 update MY_EMP33 /*6th*/
12 set last_name='Drexer'
13 where id=3;
```

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

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

The status bar at the bottom of the browser window shows the URL https://220701033@repolokahri.edu.in/index_055, the operating system as Windows 10, and the date/time as 23-02-2024.

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

QUERY:

Update MY_EMP set SALARY = '1000' where SALARY < 900

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. In the top navigation bar, the 'APEX' logo is highlighted. Below it, the 'SQL Workshop' tab is selected. The main area is titled 'SQL Commands'. The code editor contains the following SQL command:

```
1 | update MY_EMP33
2 | set salary=1000
3 | where salary<000;
```

Below the code editor, there are tabs for 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History'. The 'Results' tab is active. The output section displays the message: '0 row(s) updated.' and '0.02 seconds'. At the bottom of the page, there is footer information including the URL 'apex.oracle.com/pls/apex// apex/sql-workshop/sqlcommandprocessor?session=124612383118386', the copyright notice 'Copyright © 1999, 2023, Oracle and/or its affiliates.', and the version 'Oracle APEX 23.2.4'.

8.Delete Betty dancs from MY_EMPLOYEE table.

QUERY:

Delete from MY_EMP where LAST_NAME = 'Dancs'

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. In the top navigation bar, the 'APEX' logo is visible along with 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery'. On the right side, there's a search bar, user profile information for 'ashwin_033', and a schema dropdown set to 'WKSP_INDEX033'. The main area is titled 'SQL Commands' and contains a code editor with the following SQL statement:

```
1 delete from MY_EMP33 where LAST_NAME='Dancs'
```

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

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

Execution time is listed as '0.01 seconds'. At the bottom of the page, there are browser status icons and a footer bar with the text 'Copyright © 1999, 2023, Oracle and/or its affiliates. All rights reserved. Oracle APEX 23.2.4'.

9.Empty the fourth row of the emp table.

QUERY:

Delete from MY_EMP where ID = '4'

OUTPUT:

SQL Commands

apex.oracle.com/pls/apex// apex/sql-workshop/sqlcommandprocessor?session=124612383118386

APEX App Builder SQL Workshop Team Development Gallery

Search Schema WKSP_INDEX033

Language SQL Rows 10 Clear Command Find Tables Save Run

SQL Commands

1 delete from MY_EMP33 where ID='4'

Results Explain Describe Saved SQL History

1 row(s) deleted.

0.01 seconds

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:s

INCLUDING CONSTRAINTS

EX_NO:3

DATE:

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

QUERY:

Alter table emp2 add Constraint my_emp_id_pk Primary key(id);

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. In the SQL Commands tab, the command `alter table emp2 add constraint my_emp_id_pk primary key(id);` is entered. After running the command, the Results tab displays the output: "Table altered." and "0.06 seconds". The bottom status bar shows the user's email (220701033@rajalakshmi.edu.in), session ID (index_035), and language (en). The system status bar indicates Oracle APEX 23.2.4, ENG IN, and the date 05-03-2024.

```
1 alter table emp2 add constraint my_emp_id_pk primary key(id);
```

Table altered.
0.06 seconds

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

Oracle APEX 23.2.4

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

QUERY:

Alter table DEPT1 add Constraint my_dept_id_pk Primary key(id);

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. In the top navigation bar, the 'SQL Workshop' tab is selected. The main area displays an SQL command: 'alter table DEPT add constraint my_dept_id_pk primary key(id);'. Below the command, the 'Results' tab is active, showing the output: 'Table altered.' and '0.07 seconds'. At the bottom of the page, the URL is '220701033@rajalakshmi.edu.in index_033 en', the copyright notice 'Copyright © 1999, 2023, Oracle and/or its affiliates.', and the version 'Oracle APEX 23.2.4'.

3.Add a column DEPT_ID to the EMP table. Add a foreign key reference on the EMP table that ensures that the employee is not assigned to nonexistent department. Name the constraint my_emp_dept_id_fk.

QUERY:

Alter table emp add Constraint my_emp_dept_id_fk Foreign key(ID) references emp(ID);

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. In the top navigation bar, there are tabs for 'SQL Commands', 'CSS Syntax', and 'W3Schools Tryt Editor'. Below the navigation bar, the main area has tabs for 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery'. The 'SQL Workshop' tab is selected. The SQL editor window contains the following command:

```
1 alter table emp add constraint my_emp_dept_fk foreign key(dno) references emp(dno);
```

The results pane below the editor shows the output of the command:

```
Table altered.
```

Execution time: 0.06 seconds.

The status bar at the bottom of the browser window displays the URL `https://220701035@sqlapex.mrt.edu.in/index_055/en`, the Oracle APEX version `Oracle APEX 23.2.4`, and the system date and time `01-03-2024 11:49`.

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

QUERY:

```
Alter table emp2 add (Commission number(2,2),Constraint Cn Check(Commission>0));
```

OUTPUT:

SQL Commands

apex.oracle.com/pls/apex// apex/sql-workshop/sql-commandprocessor?session=124612383118386

APEX App Builder SQL Workshop Team Development Gallery

Search Schema WKSP_INDEX033

Language SQL Rows 10 Clear Command Find Tables Save Run

SQL Commands

1 alter table emp2 add (commission number(2,2), constraint cn check(commission>0));

Results Explain Describe Saved SQL History

Table altered.

0.07 seconds

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:

Writing Basic SQL SELECT Statements

EX_NO:4

DATE:

1.The following statement executes successfully.

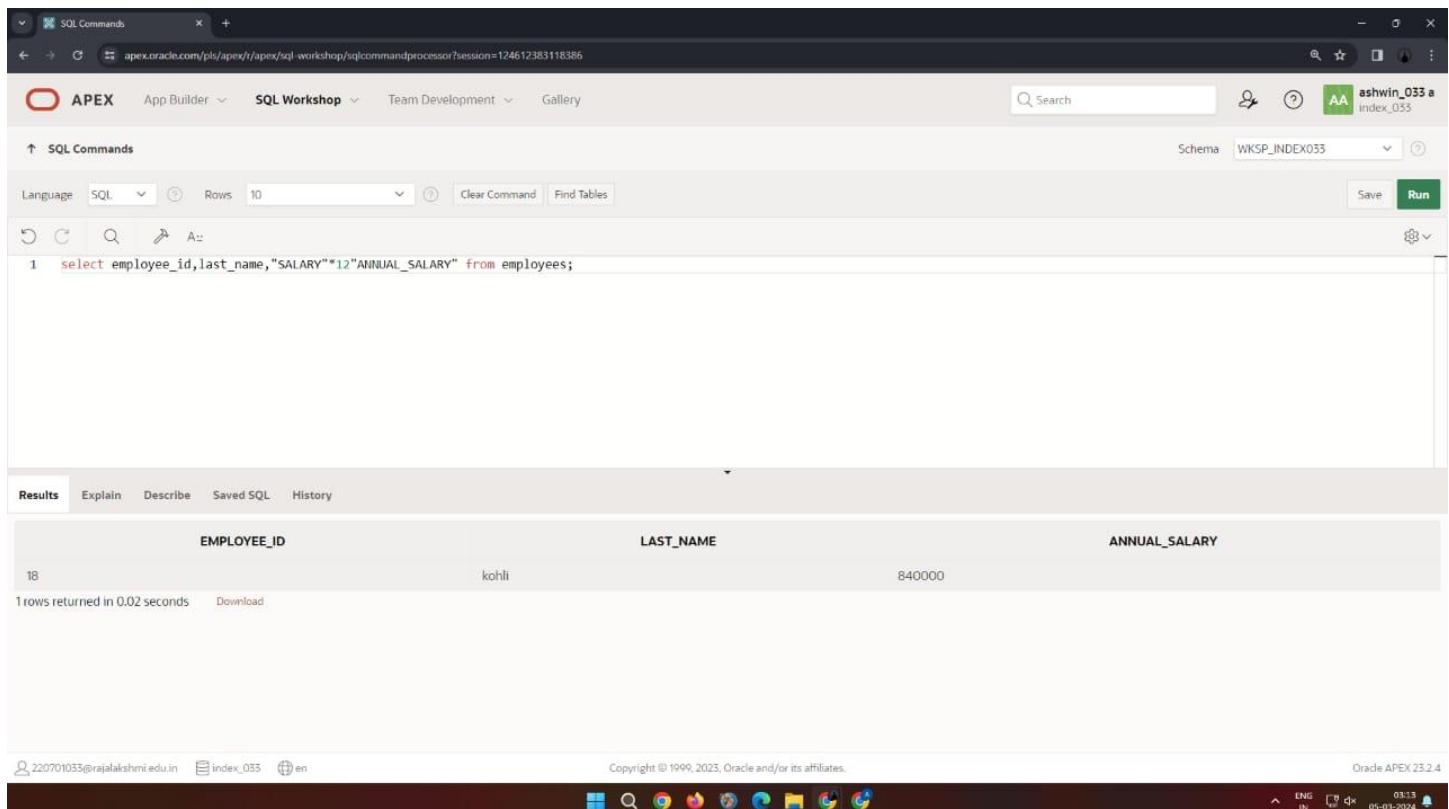
Identify the Errors

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

QUERY:

```
Select employee_id, last_name, "salary"*12 "ANNUAL_SALARY" from employees;
```

OUTPUT:



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

```
1 select employee_id, last_name, "SALARY"*12 "ANNUAL_SALARY" from employees;
```

The results section displays the following data:

EMPLOYEE_ID	LAST_NAME	ANNUAL_SALARY
18	kohli	840000

1 rows returned in 0.02 seconds

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

QUERY:

```
select * from employees;
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. In the SQL Commands tab, a single row of data is selected from the employees table. The results are displayed in a table format below.

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	MANAGER_ID	DEPARTMENT_ID
18	virat	kohli	virat@gmail.com	7418206754	01/08/2002	33	70000	9	3

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

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 as employee_number, last_name, job_id, hire_date from employees;
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. In the top navigation bar, the 'APEX' logo is visible along with 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. On the right, there's a search bar, user profile information for 'ashwin_033', and a schema dropdown set to 'WKSP_INDEX033'. The main area is titled 'SQL Commands' with a sub-section 'Language: SQL'. Below this, there are buttons for 'Rows' (set to 10), 'Clear Command', and 'Find Tables'. The SQL command entered is:

```
1 Select employee_id as employee_number, last_name, job_id, hire_date from employees;
```

Below the command, the results tab is selected. The output shows a single row of data:

EMPLOYEE_NUMBER	LAST_NAME	JOB_ID	HIRE_DATE
18	kohli	33	01/08/2002

Text below the table indicates '1 rows returned in 0.01 seconds' and a 'Download' link.

At the bottom of the page, there are footer links for '220701033@rajalakshmi.edu.in', 'index_033', and 'en'. The copyright notice reads 'Copyright © 1999, 2023, Oracle and/or its affiliates.' and the version 'Oracle APEX 23.2.4' is mentioned. The system status bar at the bottom right shows 'ENGLISH IN 03:16 05-03-2024'.

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. In the top navigation bar, the 'APEX' logo is visible along with 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery'. The 'SQL Workshop' tab is selected. On the right side, there is a search bar, a user profile for 'ashwin_033', and a schema dropdown set to 'WKSP_INDEX033'. The main workspace is titled 'SQL Commands' and contains a SQL editor with the following code:

```
1 Select hire_date as startdate from employees;
```

Below the editor, there are buttons for 'Save' and 'Run'. The results tab is selected, showing the output of the query:

STARTDATE
01/08/2002

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

At the bottom of the page, there is footer information including the URL '220701033@rajalakshmi.edu.in', the schema 'index_033', and the language 'en'. It also includes copyright information 'Copyright © 1999, 2023, Oracle and/or its affiliates.' and the text 'Oracle APEX 23.2.4'. The bottom right corner shows system status icons for 'ENG IN', 'd', '03:16', and the date '05-03-2024'.

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 SQL Workshop interface. In the top navigation bar, the user is in the 'SQL Workshop' section. The main area is titled 'SQL Commands'. A single line of SQL code is entered: 'Select distinct job_id from employees;'. Below the code, the results tab is selected, showing a single row with the value '33' under the 'JOB_ID' column. The status bar at the bottom indicates '1 rows returned in 0.00 seconds'.

```
1 Select distinct job_id from employees;
```

JOB_ID
33

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

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. In the top navigation bar, there are tabs for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there's a search bar, user information for 'ashwin_033', and a schema dropdown set to 'WKSP_INDEX033'. The main area is titled 'SQL Commands' with a sub-section 'Language: SQL'. Below this, there are buttons for Refresh, Undo, Redo, Find, and Paste. The SQL command entered is:

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

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

employee_and_title
kohli,33

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

At the bottom, there are footer links for '220701033@rajalakshmi.edu.in', 'index_033', and 'en'. The copyright notice reads 'Copyright © 1999, 2023, Oracle and/or its affiliates.' The status bar at the bottom right shows 'Oracle APEX 23.2.4', 'EN IN', '03:17', and the date '05-03-2024'.

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

QUERY:

```
select employee_id||','||first_name||','||last_name||','||email||','||phone_number||','||hire_date||',
'||job_id||',
'||salary||','||manager_id||','||','||department_id as "the_output" from employees;
```

OUTPUT:

SQL Commands - x

apex.oracle.com/pls/apex/r/apex/sql-workshop/sqlcommandprocessor?session=124612383118386

APEx App Builder SQL Workshop Team Development Gallery Search ashwin_053 a index_053

↑ SQL Commands Schema WKSP_INDEX053

Language SQL Rows 10 Clear Command Find Tables Save Run

1 select employee_id||','||first_name||','||last_name||','||email||','||phone_number||','||hire_date||','||job_id||',
 2 ||salary||','||manager_id||','||department_id as "the_output" from employees;
 3
 4

Results Explain Describe Saved SQL History

the_output

18,virat,kohli,virat@gmail.com,7418206754,01/08/2002,33,70000,9,3

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

Copyright © 1999, 2023, Oracle and/or its affiliates. Oracle APEX 25.2.4

220701033@rajalakshmi.edu.in index_053 en

ENG IN 03:16 05-03-2024

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

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

RESULT:

RESTRICTING AND STORING DATA

EX.NO:5

DATE:

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

QUERY:

```
SELECT Last_name, Salary FROM EMPLOYEE WHERE Salary > 12000;
```

OUTPUT:

The screenshot shows a browser window for Oracle APEX SQL Workshop. The URL is `apex.oracle.com/pls/apex/r/apex/sql-workshop/sqlcommandprocessor?session=14653851659321`. The page title is "SQL Commands". The top navigation bar includes "APEX", "App Builder", "SQL Workshop", "Team Development", and "Gallery". On the right, there are buttons for "Search", "Help", and user information "ashwin_033 a Index_033". The main area has tabs for "Language", "SQL", "Rows", and "Save/Run". Below these are icons for Undo, Redo, Find, Replace, and Sort. The SQL command entered is:

```
1 select last_name,job_id,hire_date
2   from employees
3  where(hire_date between '02/20/2004' AND '05/01/2015');
4
```

The results section shows a table with three columns: LAST_NAME, JOB_ID, and HIRE_DATE. The data returned is:

LAST_NAME	JOB_ID	HIRE_DATE
sharma	54	03/04/2016
dhorri	7	07/07/2007

Below the table, it says "2 rows returned in 0.01 seconds". The status bar at the bottom shows "Activate Windows Go to Settings to activate Windows.", the user "220701033@mjrejalakshmi.edu.in", the session "Index_033", and the date "08-03-2024". The system also displays the weather as "29°C Mostly sunny".

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

QUERY:

```
SELECT Last_name, Department_id FROM EMPLOY2 WHERE Employee_id = 176;
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. In the top-left corner, there's a browser window titled "SQL Commands" with the URL "apex.oracle.com/pls/apex/r/apex/sql-workshop/sqlcommandprocessor?session=14653851659321". The main menu bar includes "APEX", "App Builder", "SQL Workshop" (which is selected), "Team Development", and "Gallery". On the right side, there are tabs for "Search", "ashwin_033", and "Index_033". Below the menu, the "Schema" dropdown is set to "WKSP_INDEX033". The toolbar has buttons for "Save" and "Run". The SQL editor contains the following code:

```
1 select last_name,job_id,hire_date
2   from employees
3  where(hire_date between '82/20/2004' AND '85/01/2015');
```

Below the editor, the "Results" tab is selected. The results table has three columns: LAST_NAME, JOB_ID, and HIRE_DATE. The data is as follows:

LAST_NAME	JOB_ID	HIRE_DATE
sharma	54	03/04/2016
dholni	7	07/07/2007

At the bottom of the results table, it says "2 rows returned in 0.01 seconds". The status bar at the bottom of the screen shows "Activate Windows Go to Settings to activate Windows.", the user's email "220701033@rmrjalaknandi.edu.in", the session ID "Index_033", and the system date "08-03-2024". It also displays the weather as "29°C Mostly sunny" and the system time as "11:47".

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 EMPLOY2 WHERE Salary NOT BETWEEN 5000 AND 12000;
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. In the top navigation bar, there are tabs for 'APEX', 'App Builder', 'SQL Workshop' (which is selected), 'Team Development', and 'Gallery'. The 'SQL Commands' tab is active. The schema dropdown shows 'WKSP_INDEX033'. The main area contains the following SQL code:

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

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

LAST_NAME	SALARY	COMISSION
gill	14000	.2
dhoni	19000	.2

At the bottom of the results panel, it says '2 rows returned in 0.00 seconds' and has a 'Download' link.

The bottom status bar shows the user's email (220701033@rajalakshmi.edu.in), the schema (index_033), and the system status (Copyright © 1999, 2023, Oracle and/or its affiliates. Oracle APEX 23.2.4). It also displays system icons for battery, signal, and date/time (11-03-2024 22:10).

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,start_date from employe where start_date between '2-10-1998' and '5-1-1998' order by start_date asc;
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. In the SQL Commands tab, a query is entered:

```
1 select last_name,job_id,hire_date
2   from employees
3  where(hire_date between '02/20/2004' AND '05/01/2015');
4
```

The Results tab displays the output:

LAST_NAME	JOB_ID	HIRE_DATE
sharma	54	03/04/2015
dholi	7	07/07/2007

Below the results, it says "2 rows returned in 0.01 seconds". The bottom of the screen shows the Windows taskbar with various icons and system information.

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 EMPLOY2 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 links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows a user profile for 'ashwin_033 index_033'. The main area has tabs for SQL Commands, SQL (selected), Rows (10), Clear Command, Find Tables, Save, and Run. The SQL editor contains the following code:

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

The Results tab is selected, displaying the output:

LAST_NAME	DEPARTMENT_ID
kohli	20

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

At the bottom, the footer includes the URL 220701035@rajalakshmi.edu.in, session index 033, and language en. It also mentions 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, Job_id, Hire_date FROM EMPLOY2 WHERE Salary BETWEEN 5000 AND 12000  
AND Department_id IN (20, 50) ORDER BY Last_name;
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. In the top navigation bar, there are tabs for 'APEX', 'App Builder', 'SQL Workshop' (which is selected), 'Team Development', and 'Gallery'. The URL in the browser is <https://apex.oracle.com/pls/apex/r/apex/sql-workshop/sqlcommandprocessor?session=17077461383931>. The schema dropdown shows 'WKSP_INDEX033'. The main area contains a SQL command editor with the following code:

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

The 'Run' button is highlighted in green. Below the editor, the 'Results' tab is selected. The output table has three columns: LAST_NAME, SALARY, and COMISSION. The data is as follows:

LAST_NAME	SALARY	COMISSION
gill	14000	.2
dhoni	19000	.2

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

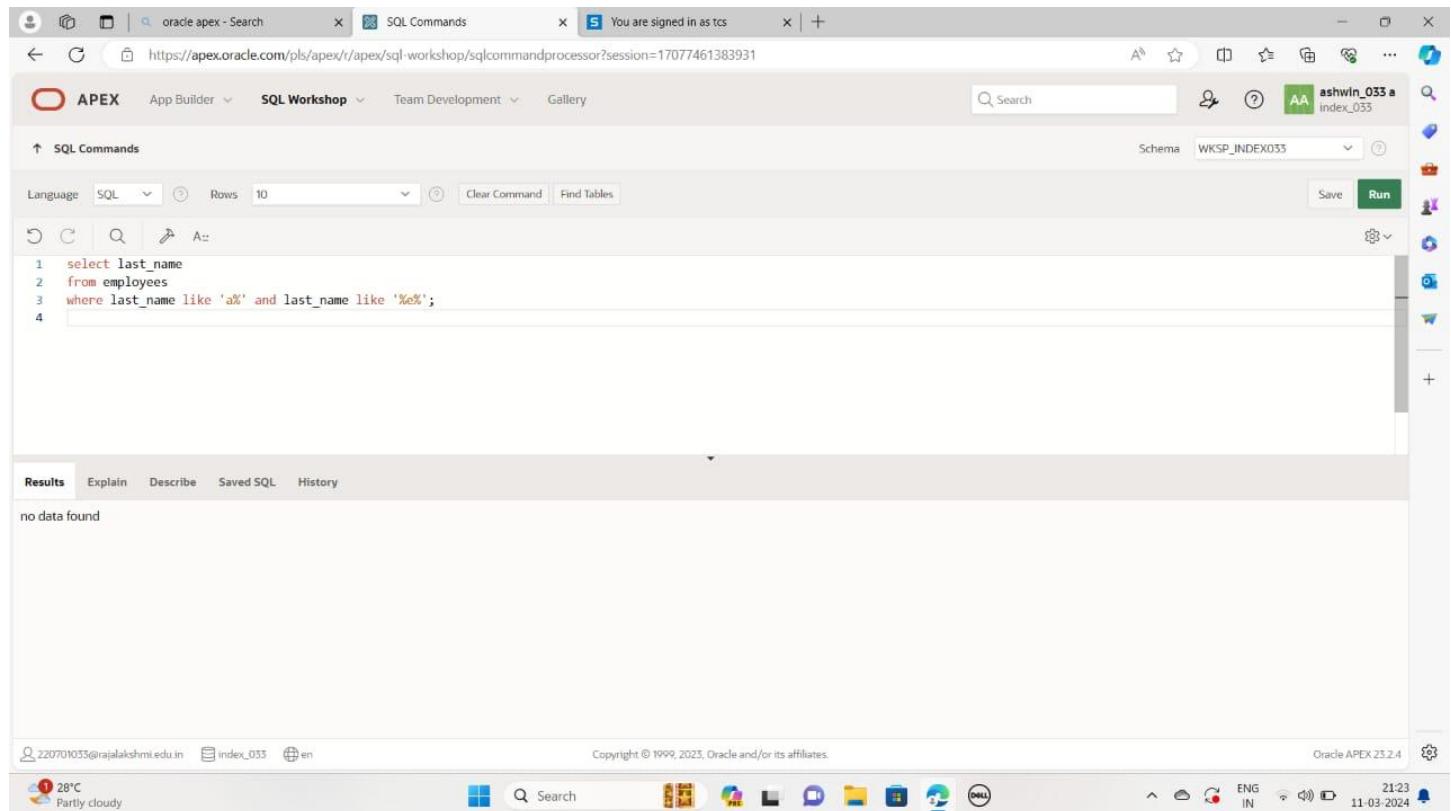
At the bottom of the page, there are status icons for 'Breaking news Unfolding now', a search bar, and system status indicators like 'ENG IN', '22:10', and '11-03-2024'.

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 EMPLOY2 WHERE Hire_date LIKE '1994-%-%';
```

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 Commands tab is selected. The schema dropdown is set to 'WKSP_INDEX033'. The SQL editor contains the following code:

```
1 select last_name
2 from employees
3 where last_name like 'a%' and last_name like '%a';
4
```

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

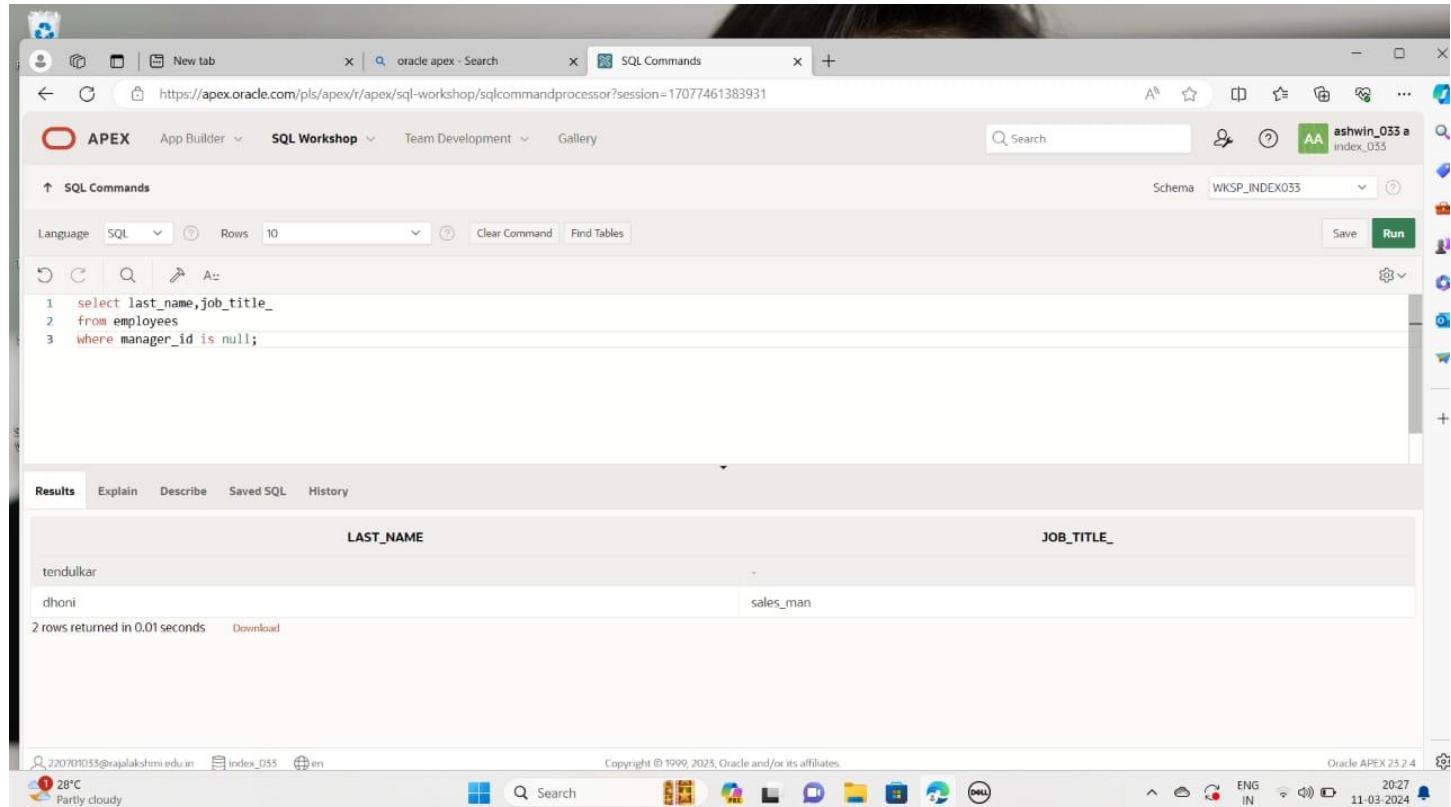
At the bottom of the screen, there is a taskbar with various icons, including a weather widget showing '28°C Partly cloudy', system status icons, and a system clock indicating '11-03-2024 21:23'.

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 EMPLOY2 WHERE Manager_id IS NULL;
```

OUTPUT:



The screenshot shows the Oracle APEX SQL Workshop interface. The URL in the browser is <https://apex.oracle.com/pls/apex/r/apex/sql-workshop/sqlcommandprocessor?session=17077461383931>. The SQL Commands tab is selected. The schema is set to WKSP_INDEX053. The query entered is:

```
1 select last_name,job_title_
2 from employees
3 where manager_id is null;
```

The results section displays the output:

LAST_NAME	JOB_TITLE_
tendulkar	
dhoni	sales_man

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

The system status bar at the bottom shows: 220701053@rajalakshmi.edu.in index_055 en 28°C Partly cloudy Copyright © 1999, 2025, Oracle and/or its affiliates. Oracle APEX 25.2.4 2027 ENG IN 11-03-2024

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

QUERY:

```
SELECT Last_name, Salary, Commission_pct FROM EMPLOY2 WHERE Commission_pct IS NOT NULL  
ORDER BY Salary DESC, Commission_pct DESC;
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The SQL Commands tab is active, displaying the following SQL code:

```
1 select last_name,salary,comission  
2 from employees  
3 where commission is not null  
4 order by salary desc;
```

The Results tab shows the output of the query:

LAST_NAME	JOB_TITLE_
tendulkar	
dhoni	sales_man

2 rows returned in 0.01 seconds

At the bottom of the screen, the Windows taskbar is visible, showing the date (11-03-2024), time (20:34), and weather (28°C, Partly cloudy).

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

QUERY:

```
SELECT Last_name from EMPLOYEE WHERE Last_name LIKE'__a%'
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. In the SQL Commands tab, a query is entered:

```
1 select last_name
2 from employees
3 where last_name like '__a%';
4
```

The results are displayed in a table:

LAST_NAME	JOB_TITLE_
tendulkar	
dhoni	sales_man

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

The system status bar at the bottom shows: 220701055@rajalakshmi.edu.in index_055 Open, 28°C Partly cloudy, Copyright © 1999, 2025, Oracle and/or its affiliates, Oracle APEX 25.2.4, ENG IN, 11-03-2024.

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 EMPLOYEE WHERE Last_name LIKE '%a%' AND Last_name LIKE '%e%';
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The SQL Workshop tab is active, showing the schema as 'WKSP_INDEX033'. The main area contains the SQL command:

```
1 select last_name
2   from employees
3  where last_name like 'a%' and last_name like '%e%';
4
```

The 'Results' tab is selected, showing the message 'no data found'. The bottom status bar displays the user '220701035@rajalakshmi.edu.in', the session ID 'index_033', and the language 'en'. It also includes copyright information for Oracle and the text 'Oracle APEX 25.2.4'.

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

QUERY:

```
select last_name, Job_title,salary from employe where job_title IN('sales_representative','stock_clerk') AND SALARY not IN(2500,3500,7000)
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there's a search bar, user profile, and schema information (ashwin_033 a index_033). The main area has tabs for SQL Commands and Results. Under SQL Commands, the query is displayed:

```
1 select last_name,job_title_,salary
2 from employees
3 where job_title_ in('sales_representative','stock_clerk') and salary not in (2500,3500,7000);
```

Under Results, the output is shown in a table:

LAST_NAME	JOB_TITLE_	SALARY
gill	stock_clerk	14000

Below the table, it says "1 rows returned in 0.01 seconds" and there's a "Download" link. At the bottom of the page, there are footer links for user info, copyright (Copyright © 1999, 2023, Oracle and/or its affiliates.), and the Oracle APEX version (Oracle APEX 23.2.4).

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 employe where commission_pct = 0.2
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. In the SQL Commands tab, a query is entered:

```
1 select last_name,salary,commission
2 from employees
3 where comission=0.2;
```

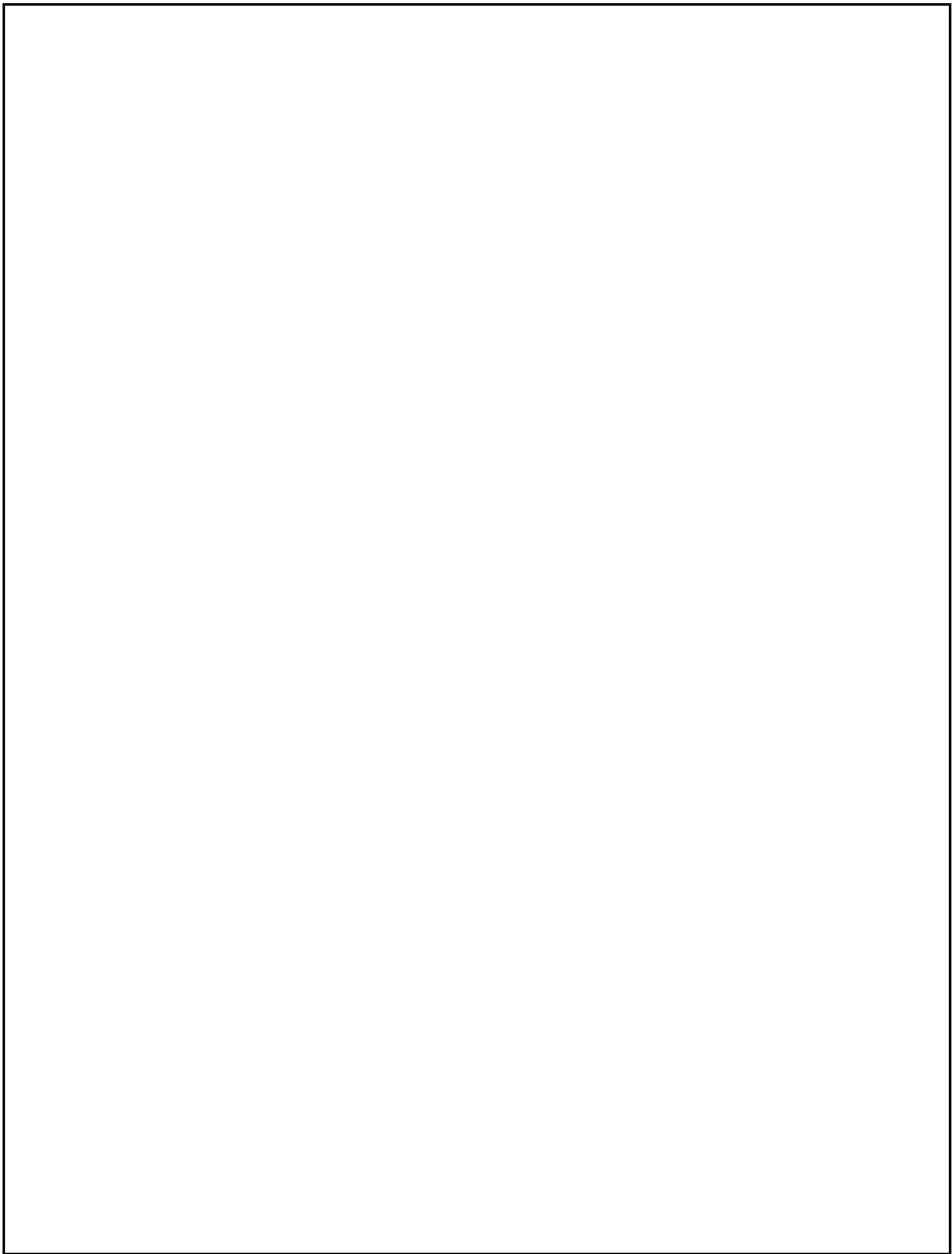
In the Results tab, the output is displayed in a grid:

LAST_NAME	SALARY	COMISSION
gill	14000	.2
dhoni	19000	.2

Below the grid, it says "2 rows returned in 0.00 seconds".

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

RESULT:



SINGLE ROW FUNCTIONS

EX_NO:6

DATE: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'. On the right, there are icons for 'Search', 'User', 'Help', and a session named 'ashwin_033 index_033'. The main workspace has tabs for 'Language' (set to 'SQL'), 'Rows' (set to 10), and 'Clear Command / Find Tables'. Below this is a toolbar with icons for refresh, search, and run. The SQL editor contains the query: '1 select sysdate from dual;'. The results tab is selected, showing the output: 'SYSDATE' with the value '03/10/2024'. Below the results, it says '1 rows returned in 0.01 seconds' and provides a 'Download' link. At the bottom, the footer includes user information ('220701043@rajalakshmi.edu.in', 'bharath', 'en'), copyright ('Copyright © 1999, 2023, Oracle and/or its affiliates.'), and version ('Oracle APEX 23.2.4').

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

QUERY:

```
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, identical to the previous one but with a different query. The top navigation bar and workspace are the same. The SQL editor contains the query: '1 select employee_id, last_name, salary, salary+(15.5/100*salary) "new_salary" from employees;'. The results tab is selected, displaying a table with columns 'EMPLOYEE_ID', 'LAST_NAME', 'SALARY', and 'new_salary'. The data rows are: (113, popp, 6900, 7969.5), (114, rapheal, 11000, 12705), and (2, Mohan, 4000, 4620). Below the results, it says '3 rows returned in 0.01 seconds' and provides a 'Download' link. The footer information is the same as the first screenshot.

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 top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. On the right, there are user profile icons for 'ashwin_033' and 'index_035'. 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 statement:

```
1 select employee_id, last_name, salary, salary+(15.5/100*salary) "new_salary", (salary+(15.5/100*salary))-salary as "Increase" from employees;
2
```

Below the code editor is a results grid with the following columns: EMPLOYEE_ID, LAST_NAME, SALARY, new_salary, and Increase. The data rows are:

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

At the bottom left, it says '3 rows returned in 0.01 seconds'. At the bottom right, it says 'Download' and 'Copyright © 1999, 2023, Oracle and/or its affiliates. Oracle APEX 23.2.4'.

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 top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. On the right, there are user profile icons for 'ashwin_033' and 'index_035'. 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 statement:

```
1 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;
```

Below the code editor is a results grid with the following columns: INITCAP(LAST_NAME) and Length_of_last_name. The data row is:

INITCAP(LAST_NAME)	Length_of_last_name
Mohan	5

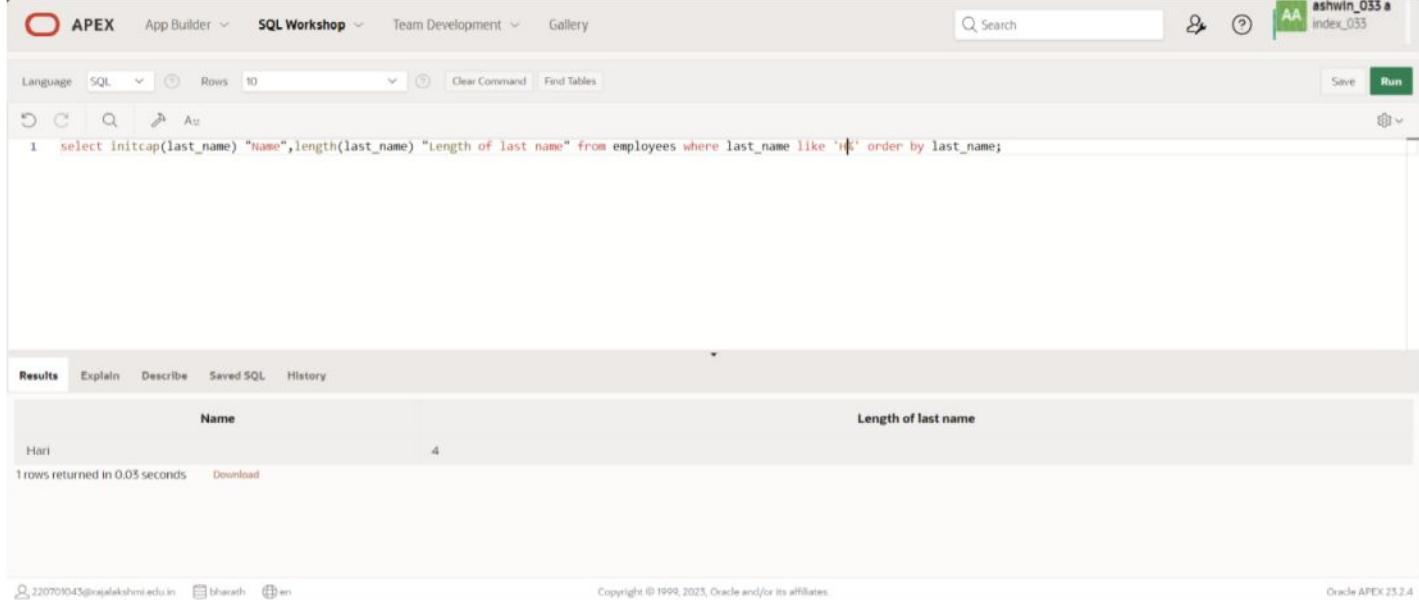
At the bottom left, it says '1 rows returned in 0.01 seconds'. At the bottom right, it says 'Download' and 'Copyright © 1999, 2023, Oracle and/or its affiliates. Oracle APEX 23.2.4'.

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 are icons for search, help, and a session named 'ashwin_OSS a index_033'. The main area has tabs for 'Language', 'SQL', 'Rows' (set to 10), 'Clear Command', and 'Find Tables'. Below this is a toolbar with icons for refresh, search, and run. The SQL command entered is:

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

The results section shows a single row:

Name	Length of last name
Hari	4

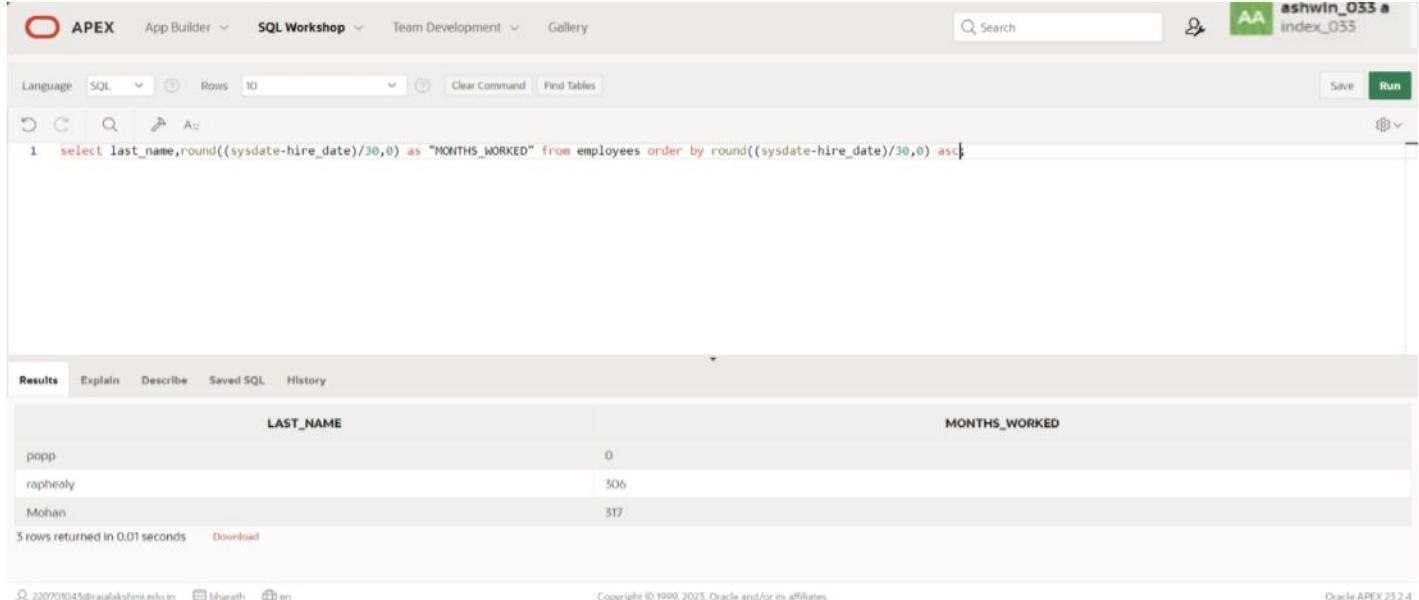
Below the results, it says '1 rows returned in 0.03 seconds' and has a 'Download' link. The bottom footer includes the URL '220701045@rajalakshmi.edu.in', the name 'bharath', and the Oracle APEX version 'Oracle APEX 23.2.4'.

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. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. On the right, there are icons for search, help, and a session named 'ashwin_OSS a index_033'. The main area has tabs for 'Language', 'SQL', 'Rows' (set to 10), 'Clear Command', and 'Find Tables'. Below this is a toolbar with icons for refresh, search, and run. The SQL command entered is:

```
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 section shows three rows:

LAST_NAME	MONTHS_WORKED
popp	0
raphealy	306
Mohan	317

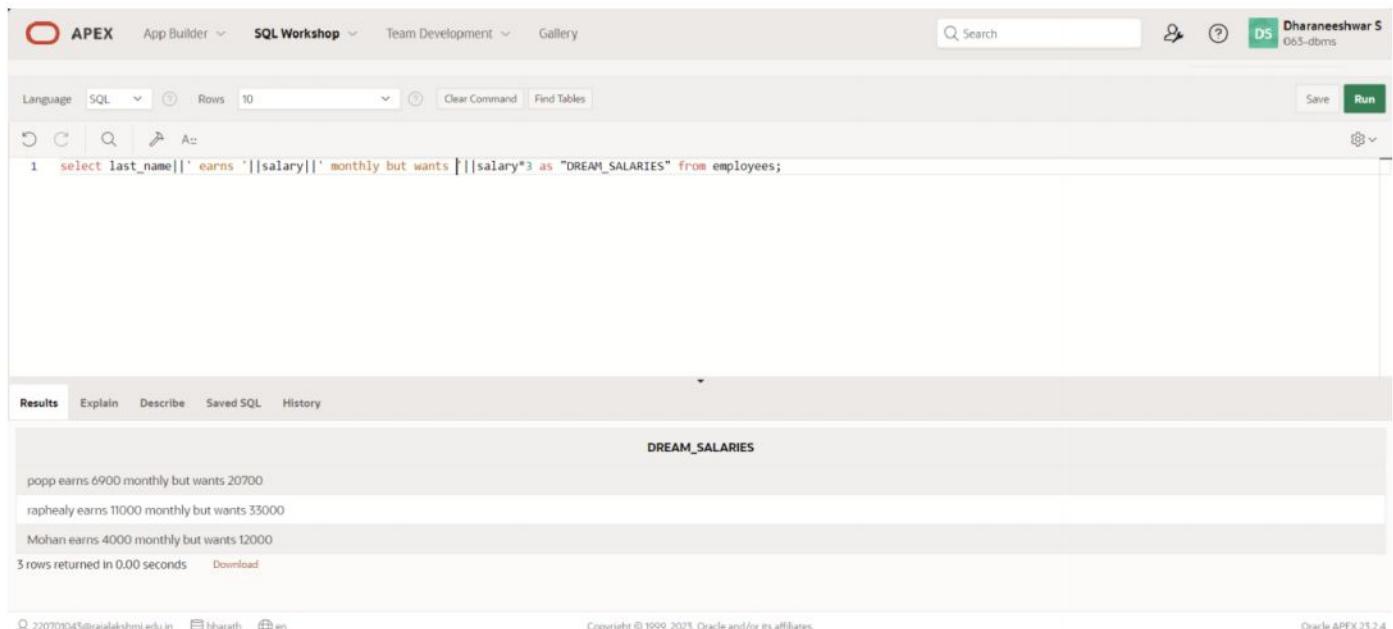
Below the results, it says '3 rows returned in 0.01 seconds' and has a 'Download' link. The bottom footer includes the URL '220701045@rajalakshmi.edu.in', the name 'bharath', and the Oracle APEX version 'Oracle APEX 23.2.4'.

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 search bar, user profile, and a green button for 'Dharaneeshwar S 065-dbms'. The main workspace has tabs for 'Language', 'SQL', 'Rows' (set to 10), 'Clear Command', and 'Find Tables'. Below this is a toolbar with 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

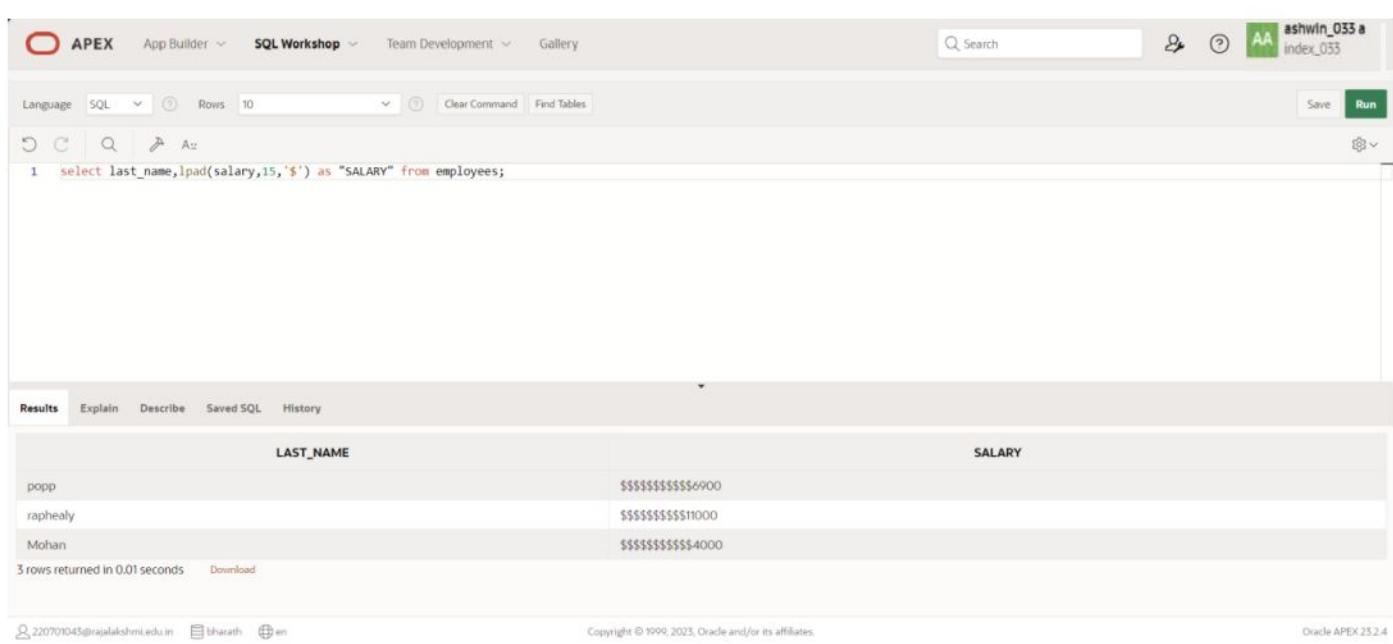
Below the results, it says '3 rows returned In 0.00 seconds' and has a 'Download' link. The bottom footer includes user information (220701045@rajalakshmi.edu.in, bharath, en), copyright notice (Copyright © 1999, 2023, Oracle and/or its affiliates), and version (Oracle APEX 23.2.4).

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 search bar, user profile, and a green button for 'ashwin_033 a index_033'. The main workspace has tabs for 'Language', 'SQL', 'Rows' (set to 10), 'Clear Command', and 'Find Tables'. Below this is a toolbar with 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

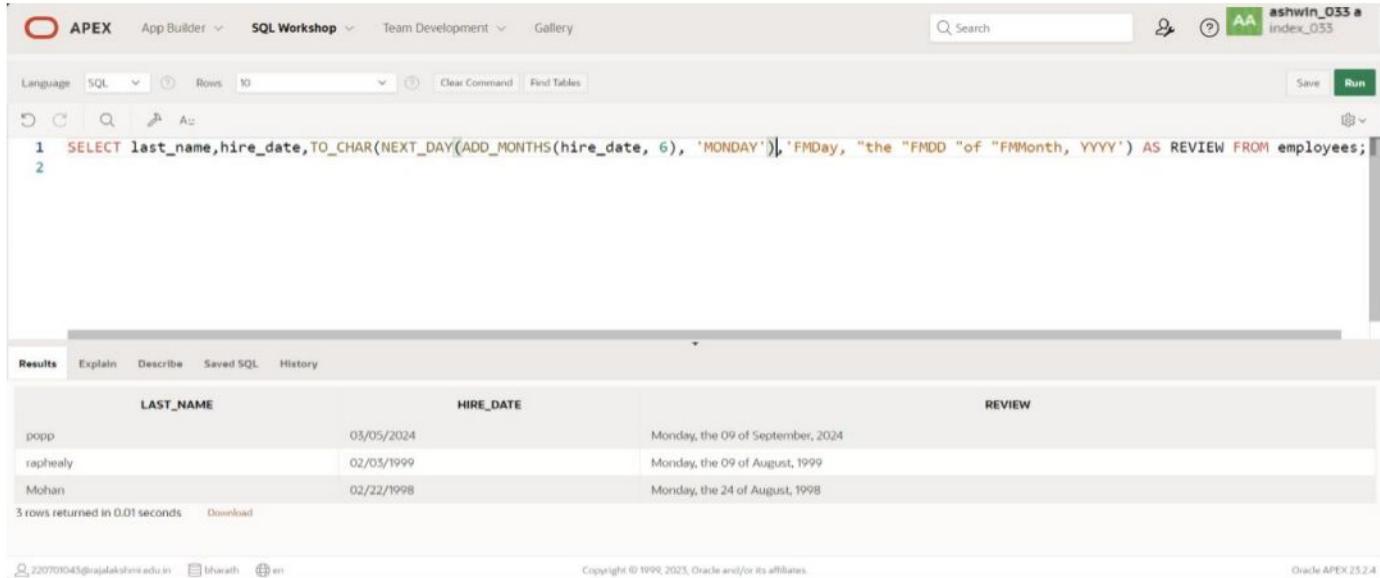
Below the results, it says '3 rows returned In 0.01 seconds' and has a 'Download' link. The bottom footer includes user information (220701045@rajalakshmi.edu.in, bharath, en), copyright notice (Copyright © 1999, 2023, Oracle and/or its affiliates), and version (Oracle APEX 23.2.4).

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

QUERY:

```
SELECT last_name,hire_date,TO_CHAR(NEXT_DAY(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 'ashwin_033' and a search bar. 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 two lines of SQL:

```
1 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;
2
```

Below the code editor is a results grid with three columns: 'LAST_NAME', 'HIRE_DATE', and 'REVIEW'. The data rows are:

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

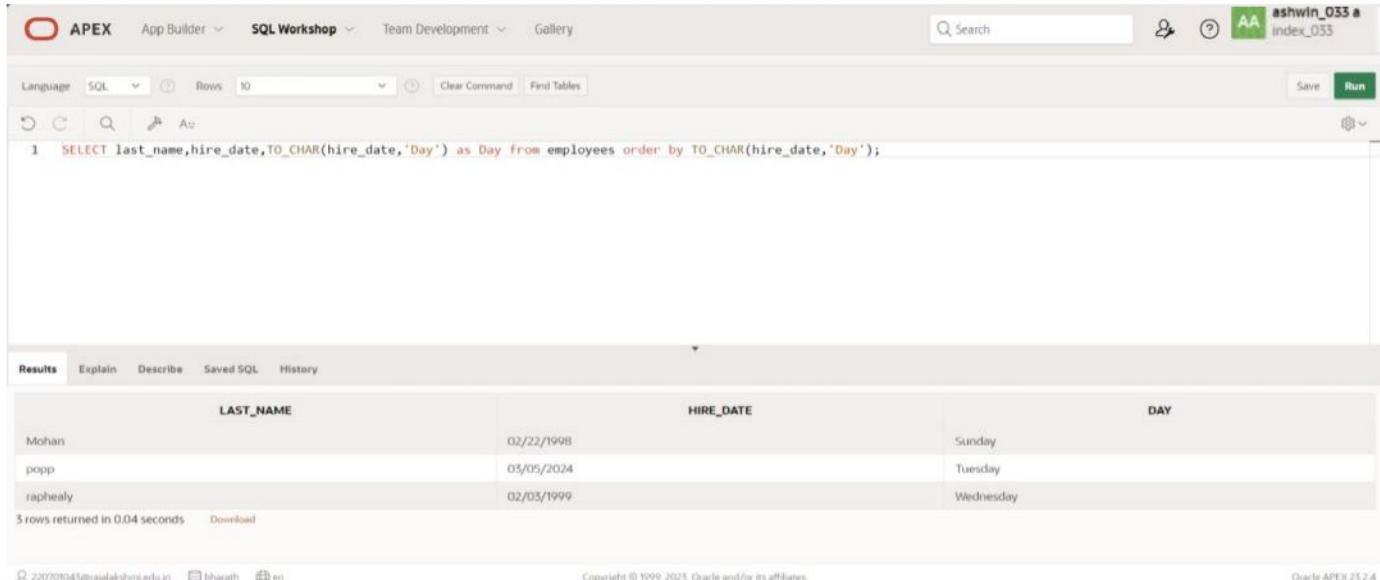
At the bottom left, it says '3 rows returned in 0.01 seconds'. At the bottom right, it says 'Download' and 'Oracle APEX 23.2.4'.

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, identical to the previous one but with a different query. The top navigation bar and user profile are the same. The code editor contains:

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

Below the code editor is a results grid with three columns: 'LAST_NAME', 'HIRE_DATE', and 'DAY'. The data rows are:

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

At the bottom left, it says '3 rows returned in 0.04 seconds'. At the bottom right, it says 'Download' and 'Oracle APEX 23.2.4'.

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

RESULT:

DISPLAYING DATA FROM MULTIPLE TABLES

EX_NO:7

DATE:

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

QUERY:

```
Select e.last_name,e.department_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 user profile for 'ashwin_033' and a search bar. The main area has tabs for 'Language', 'SQL', 'Rows', and 'Clear Command'. Below is a command line with the query: 'select e.last_name,e.department_id,d.department_id from employees e,departments d;'. The results tab is selected, showing a table with three columns: LAST_NAME, DEPARTMENT_ID, and DEPT_NAME. The data is as follows:

LAST_NAME	DEPARTMENT_ID	DEPT_NAME
Sara	100	Public Relations
Marti	20	Public Relations
Michale	150	Public Relations
Sara	100	Finance
Marti	20	Finance
Michale	150	Finance

Below the table, it says '6 rows returned in 0.01 seconds'. The bottom of the page includes copyright information and a link to Oracle APEX 23.2.4.

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

QUERY:

```
select distinct job_id,loc_id from 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 user profile for 'Dharaneeshwar S 063-dms' and a search bar. The main area has tabs for 'Language', 'SQL', 'Rows', and 'Clear Command'. Below is a command line with the query: 'select distinct job_id,LOCATION_ID FROM EMPLOYEES,DEPARTMENTS where employees.department_id=departments.dept_id and employees.department_id=80;'. The results tab is selected, showing a table with two columns: JOB_ID and LOCATION_ID. The data is as follows:

JOB_ID	LOCATION_ID
ac_account	4598

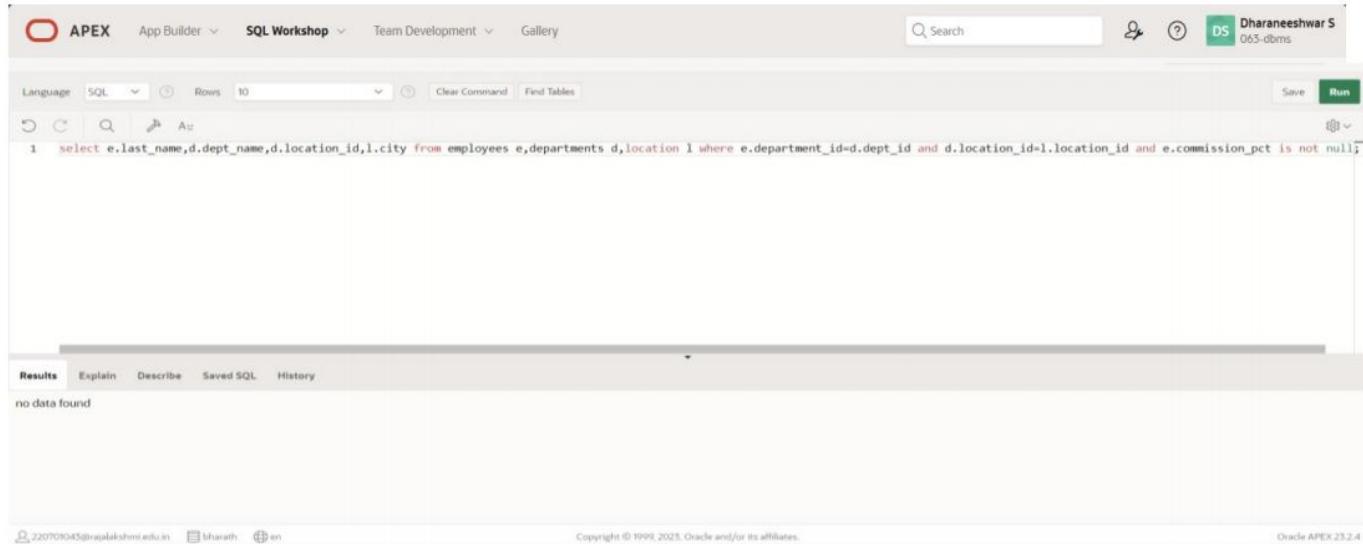
Below the table, it says '1 rows returned in 0.02 seconds'. The bottom of the page includes copyright information and a link to Oracle APEX 23.2.4.

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

QUERY:

```
Select e.last_name,e.department_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'. On the right, there's a user icon, a search bar, and a green button labeled 'DS Dharaneshwar S 065-dbms'. The main workspace has tabs for 'Language' (set to 'SQL'), '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 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;
```

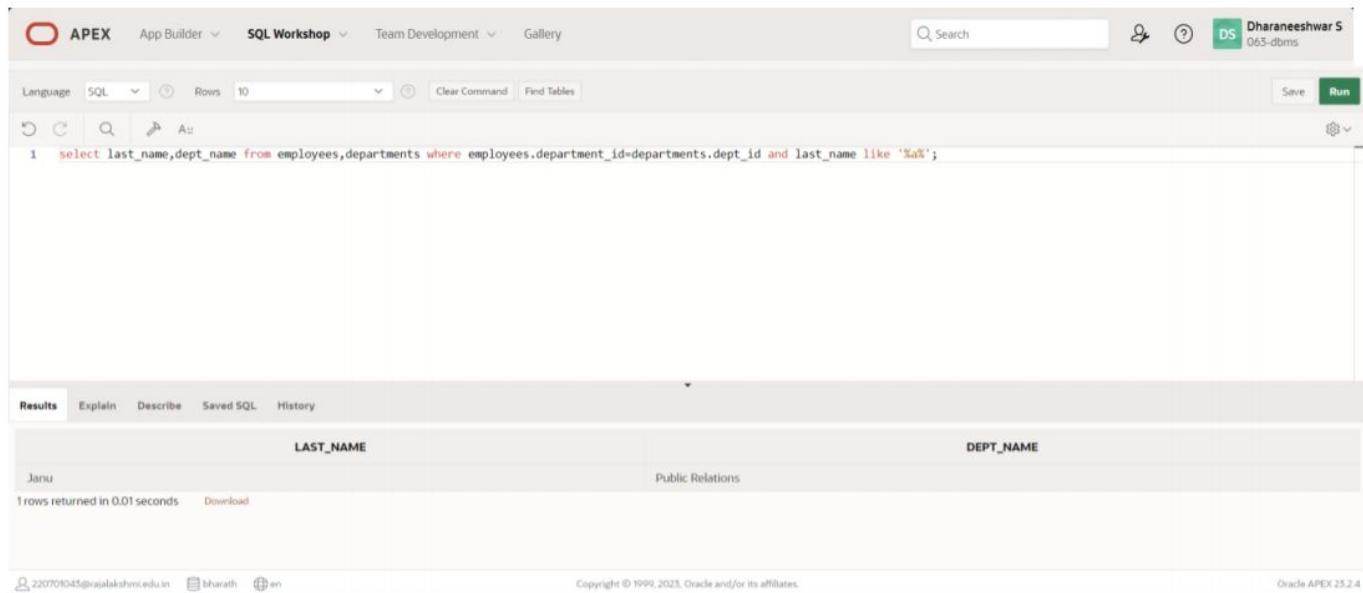
The results section shows the message 'no data found'.

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'. On the right, there's a user icon, a search bar, and a green button labeled 'DS Dharaneshwar S 065-dbms'. The main workspace has tabs for 'Language' (set to 'SQL'), '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,dept_name from employees,departments where employees.department_id=departments.department_id and last_name like '%a%';
```

The results section displays the output:

LAST_NAME	DEPT_NAME
Janu	Public Relations

1 rows returned in 0.01 seconds

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:

```
1 select last_name,job_id,department_id,dept_name from employees join departments d on (department_id=dept_id) join location l on(d.location_id=l.location_id) where lower(l.city)='toronto';
```

Results Explain Describe Saved SQL History

no data found

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:

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

Results Explain Describe Saved SQL History

no data found

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 query executed 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 table has four columns: Employee, Emp#, Manager, and Mgr#. The data is as follows:

Employee	Emp#	Manager	Mgr#
Janu	115	-	-
Mohan	2	-	-
HARI	114	-	-

3 rows returned in 0.01 seconds

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 query executed 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 table has three columns: departments, employees, and colleague. The output message at the bottom says "no data found".

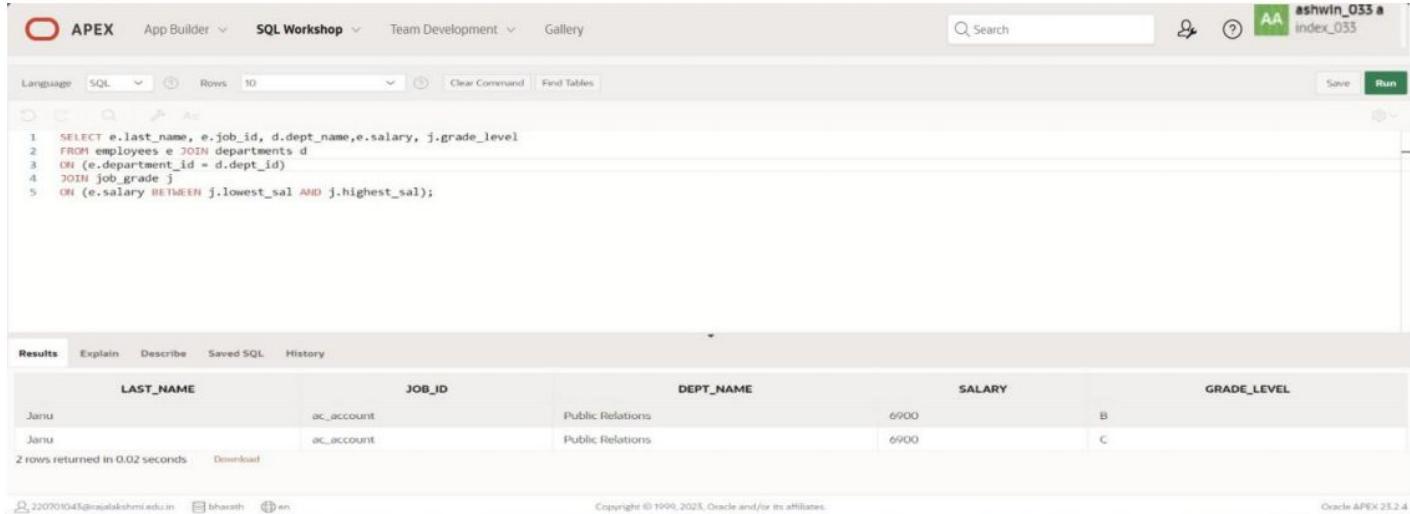
9. Show the structure of the JOB_GRADES table. Create a query that displays the name, job,

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 are user profile icons and a search bar. 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 window containing the query from the previous step. The results tab is selected, displaying a table with columns: LAST_NAME, JOB_ID, DEPT_NAME, SALARY, and GRADE_LEVEL. Two rows are returned, both for 'Janu' with 'AC_ACCOUNT' job ID, 'Public Relations' department, \$6900 salary, and grade 'C'. The status bar at the bottom indicates '2 rows returned in 0.02 seconds'.

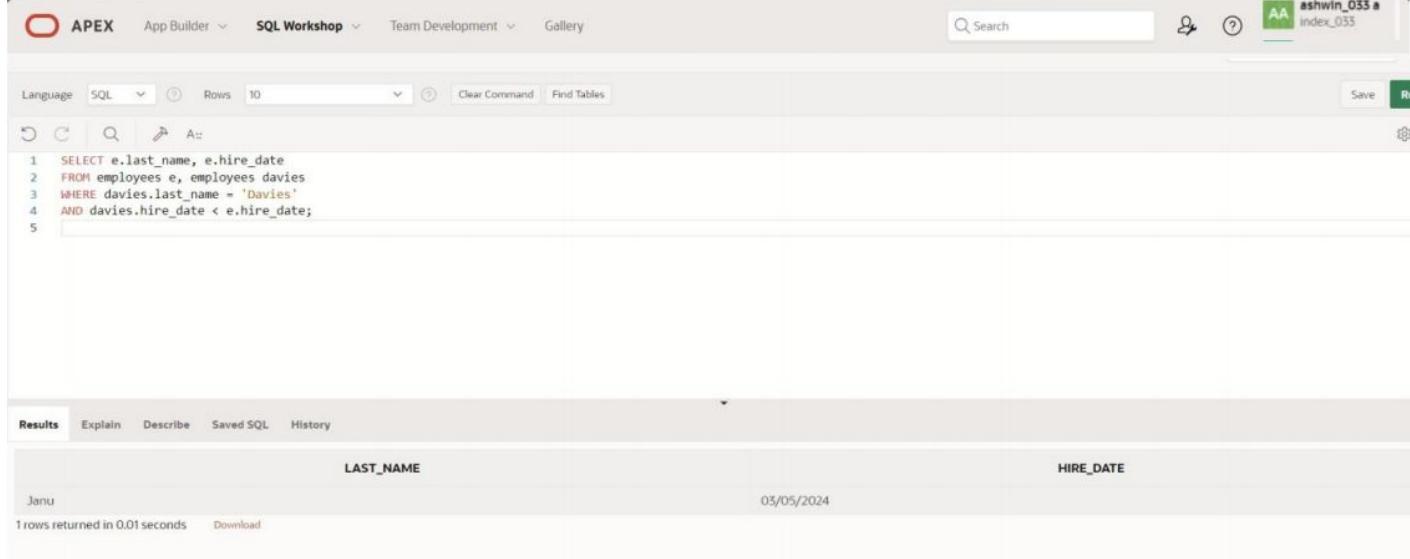
LAST_NAME	JOB_ID	DEPT_NAME	SALARY	GRADE_LEVEL
Janu	AC_ACCOUNT	Public Relations	6900	B
Janu	AC_ACCOUNT	Public Relations	6900	C

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 are user profile icons and a search bar. 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 window containing the query from the previous step. The results tab is selected, displaying a table with columns: LAST_NAME and HIRE_DATE. One row is returned for 'Janu' with a hire date of '03/05/2024'. The status bar at the bottom indicates '1 rows returned in 0.01 seconds'.

LAST_NAME	HIRE_DATE
Janu	03/05/2024

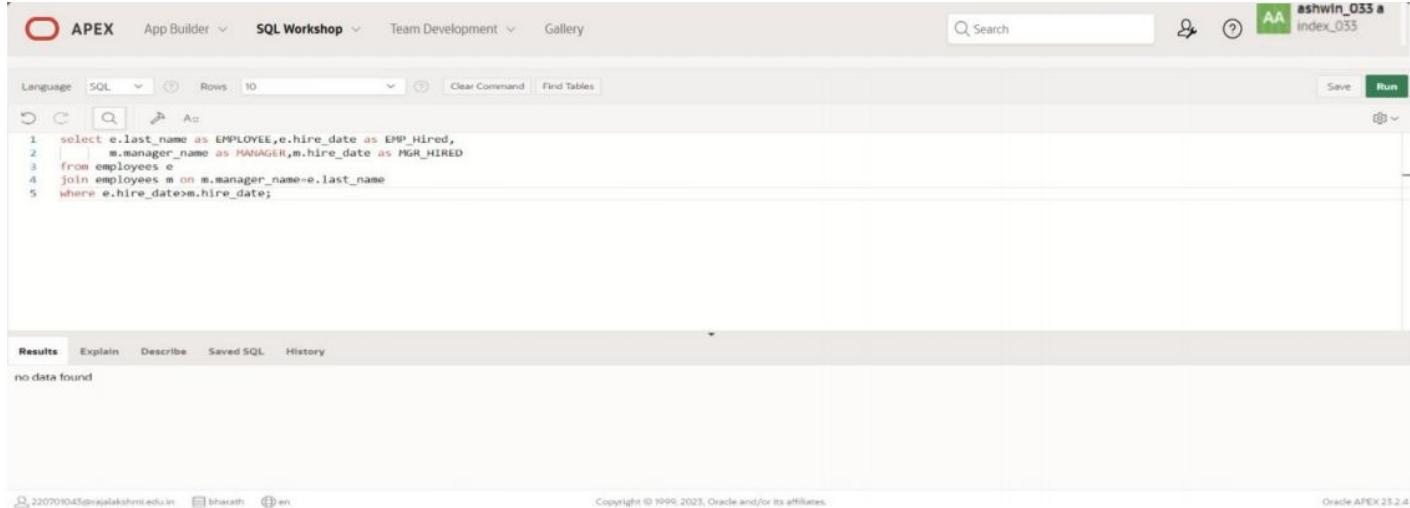
11. Display the names and hire dates for all employees who were hired before their managers,

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 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. The right side shows a user profile 'ashwin_033 a index_033'. Below the navigation is a toolbar with 'Language' (SQL), 'Rows' (10), 'Clear Command', 'Find Tables', 'Save', and 'Run'. The main area contains the SQL query:

```
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 below shows 'no data found'. At the bottom, it displays the URL '220701043@dalakshmi.edu.in', the session 'biharath', and the environment 'en'. Copyright information '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:

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

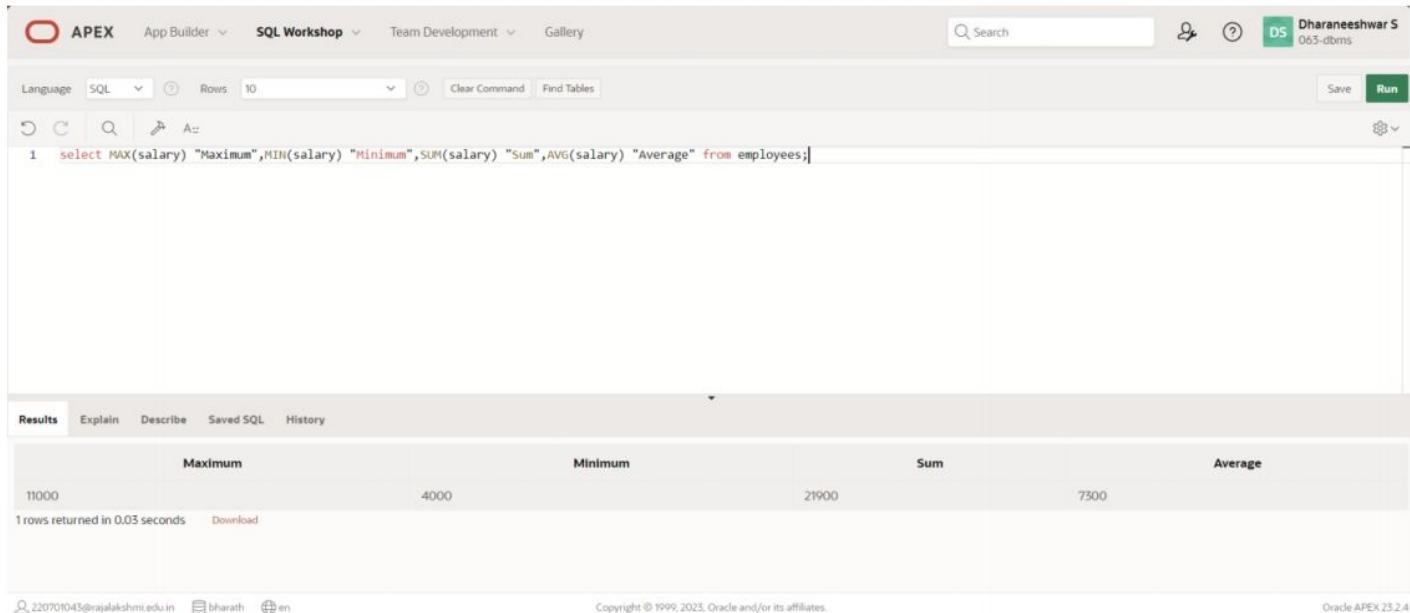
FALSE

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. The right side of the header shows a user profile for 'Dharaneeshwar S' and a database connection for '063-dbms'. The main workspace has a toolbar with various icons. Below the toolbar, there are dropdown menus for Language (SQL selected), Rows (10), and a search bar. To the right are 'Save' and 'Run' buttons. The SQL command input area contains the following code:

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

The results section shows the output of the query:

	Maximum	Minimum	Sum	Average
	11000	4000	21900	7300

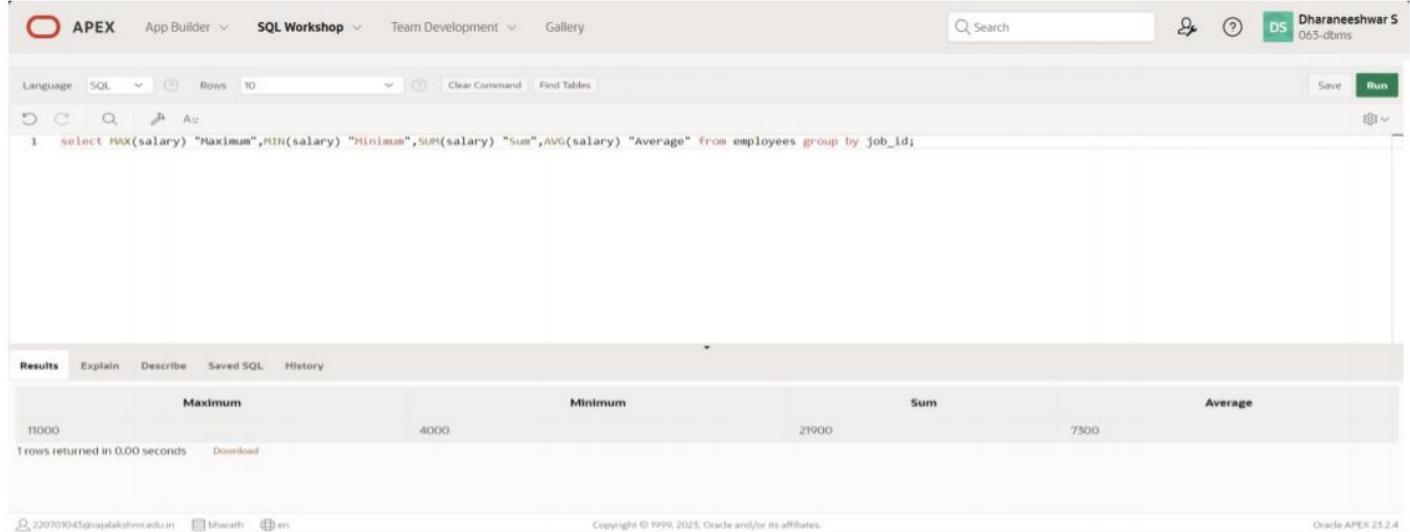
Below the table, it says '1 rows returned in 0.03 seconds' and provides a 'Download' link. At the bottom of the page, there are footer links for 220701043@rajalakshmi.edu.in, bharath, en, Copyright © 1999, 2023, Oracle and/or its affiliates, and Oracle APEX 23.2.4.

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

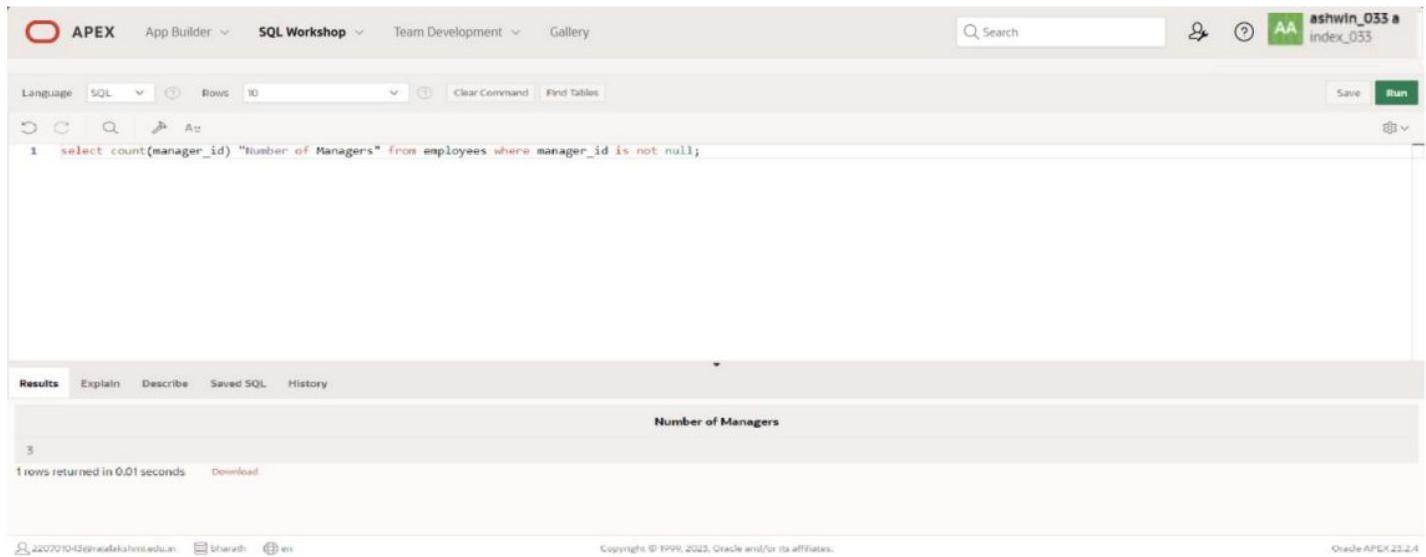
1 rows returned in 0.01 seconds [Download](#)

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:



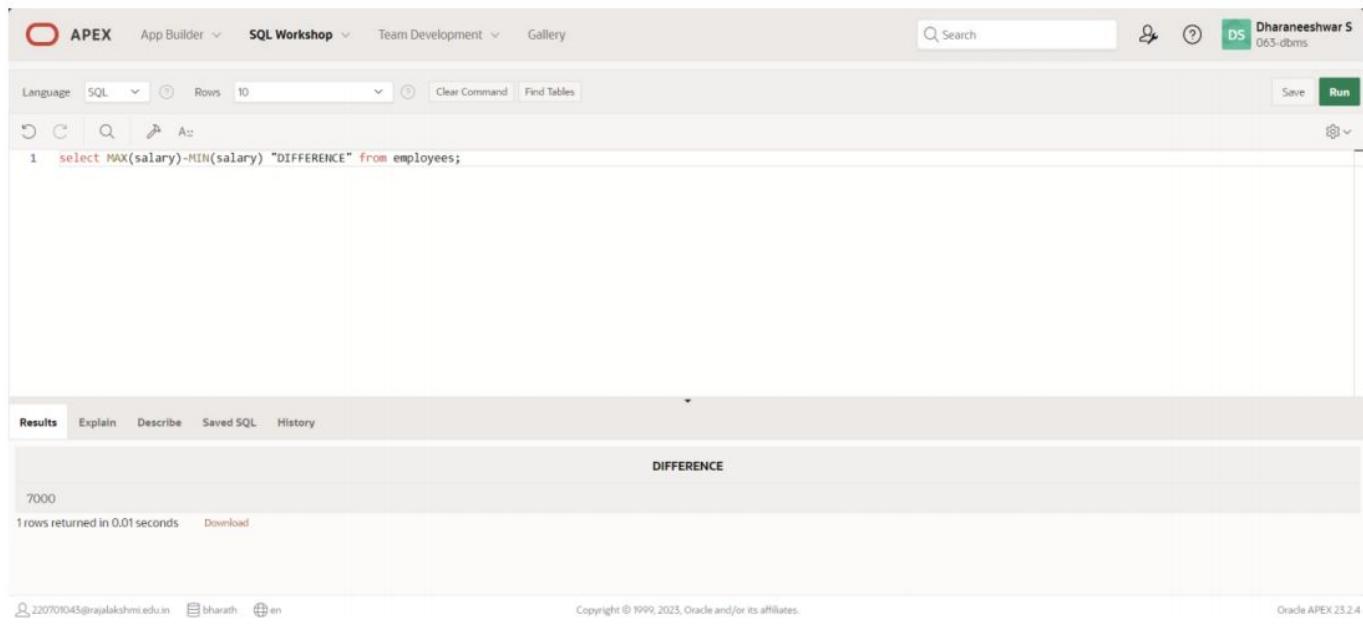
The screenshot shows the Oracle APEX interface with the SQL Workshop tab selected. The query entered is: `select count(manager_id) "Number of Managers" from employees where manager_id is not null;`. The results section displays a single row with the value 3, labeled 'Number of Managers'. The status bar at the bottom indicates '1 rows returned in 0.01 seconds'.

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

QUERY:

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

OUTPUT:



The screenshot shows the Oracle APEX interface with the SQL Workshop tab selected. The query entered is: `select MAX(salary)-MIN(salary) "DIFFERENCE" from employees;`. The results section displays a single row with the value 7000, labeled 'DIFFERENCE'. The status bar at the bottom indicates '1 rows returned in 0.01 seconds'.

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

QUERY:

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

OUTPUT:

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

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

The results table has two columns: MANAGER_ID and LOWEST_SALARY. The data is:

MANAGER_ID	LOWEST_SALARY
100	11000
205	6900

2 rows returned in 0.01 seconds

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

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

The results table has five columns: TOTAL, 1995, 1996, 1997, and 1998. The data is:

TOTAL	1995	1996	1997	1998
3	0	0	0	1

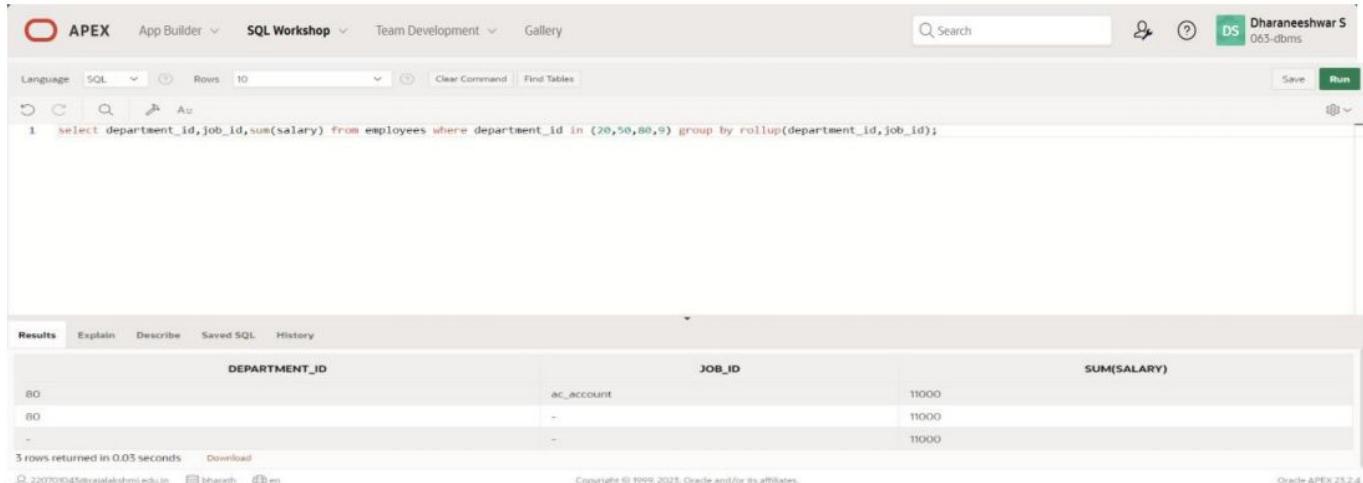
1 rows returned in 0.04 seconds

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

QUERY:

```
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 search bar, a help icon, and a connection status 'DS Dharaneshwar S 065-dbms'. Below the toolbar, the SQL command is displayed:

```
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 pane shows the output of the query:

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

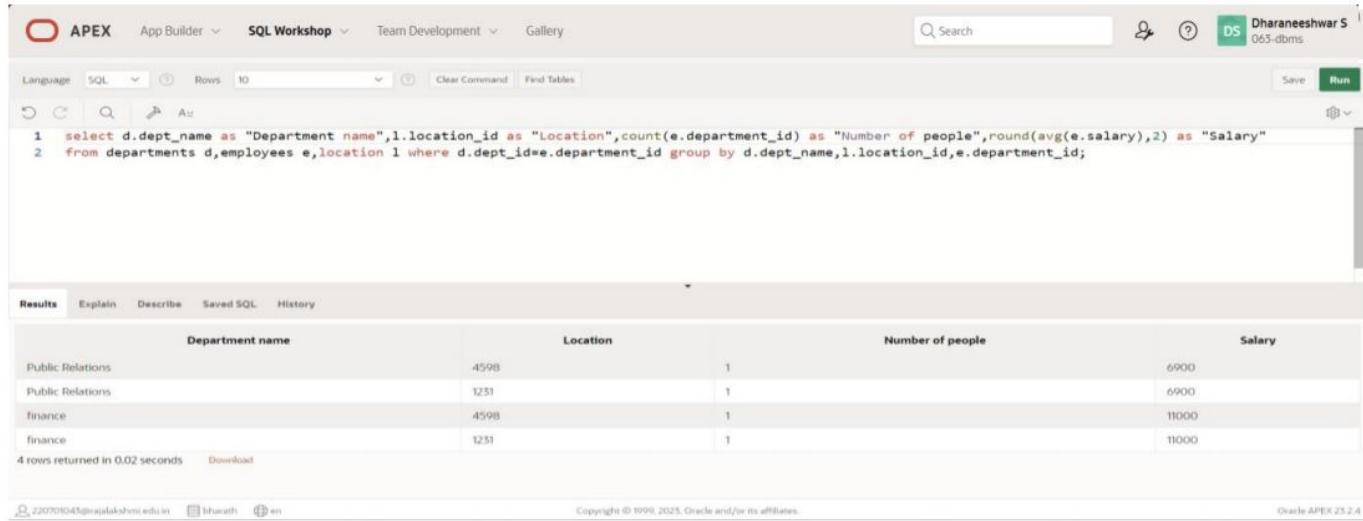
Below the table, it says '3 rows returned in 0.03 seconds'. The bottom of the screen displays copyright information and the Oracle APEX version.

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 search bar, a help icon, and a connection status 'DS Dharaneshwar S 065-dbms'. Below the toolbar, the SQL command is displayed:

```
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 pane shows the output of the query:

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

Below the table, it says '4 rows returned in 0.02 seconds'. The bottom of the screen displays copyright information and the Oracle APEX version.

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

RESULT:

SUB QUERIES

EX_NO:9

DATE:

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

QUERY:

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

OUTPUT:

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

```
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');
```

The results table has columns LAST_NAME and HIRE_DATE. It shows one row for 'Doe' hired on 03/05/1997.

LAST_NAME	HIRE_DATE
Doe	03/05/1997

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

QUERY:

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

OUTPUT:

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

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

The results table has columns EMPLOYEE_ID, LAST_NAME, and SALARY. It shows two rows: employee 1001 (last name Smith, salary 70000) and employee 142 (last name Doe, salary 50000).

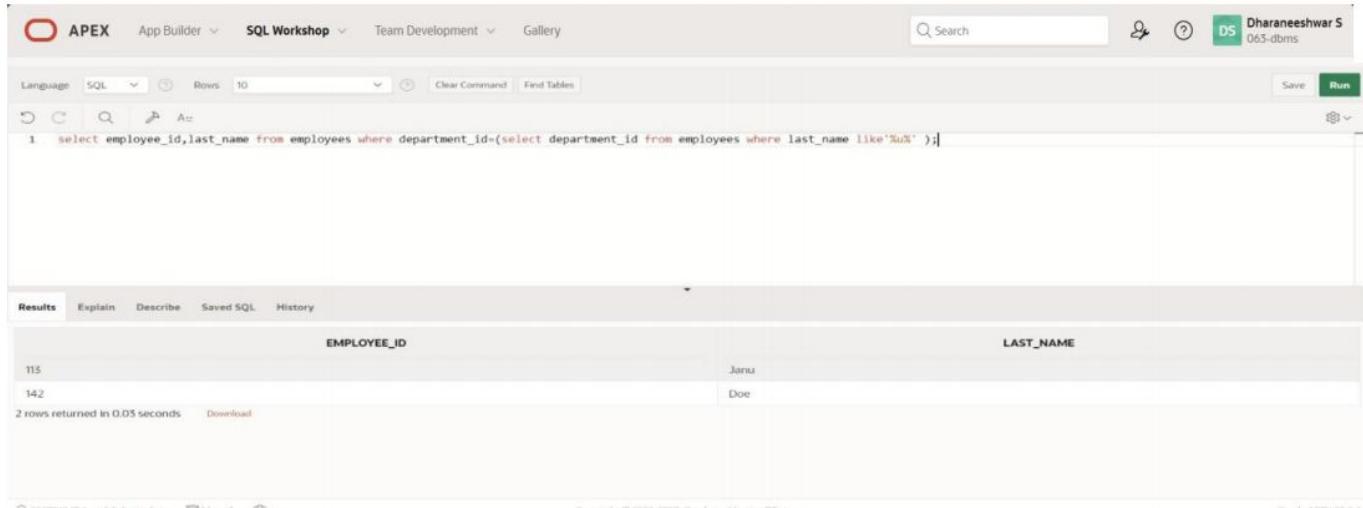
EMPLOYEE_ID	LAST_NAME	SALARY
1001	Smith	70000
142	Doe	50000

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, a user icon, and a connection status 'DS Dharaneshwar S 065-dbms'. 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 employee_id,last_name from employees where department_id=(select department_id from employees where last_name like'%u%');
```

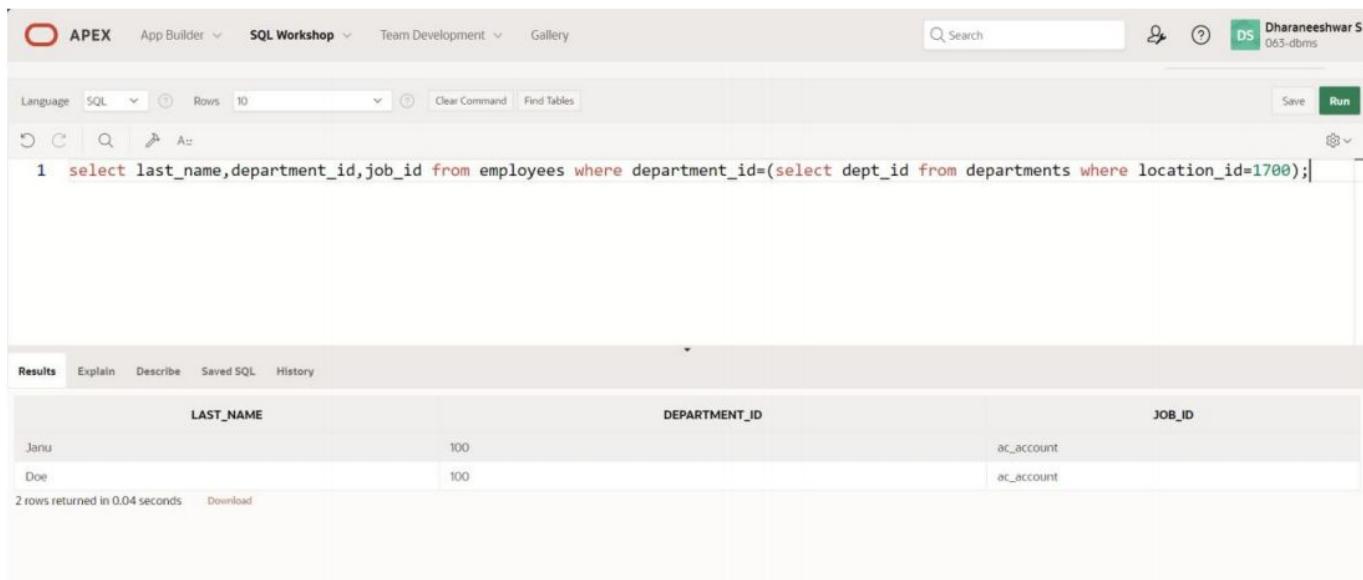
Below the code editor is a results grid with two rows. The columns are labeled 'EMPLOYEE_ID' and 'LAST_NAME'. The first row contains '115' and 'Janu'. The second row contains '142' and 'Doe'. At the bottom left, it says '2 rows returned in 0.03 seconds'. At the bottom right, there are 'Save' and 'Run' buttons.

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, a user icon, and a connection status 'DS Dharaneshwar S 065-dbms'. 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,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 two rows. The columns are labeled 'LAST_NAME', 'DEPARTMENT_ID', and 'JOB_ID'. The first row contains 'Janu' and '100' under 'DEPARTMENT_ID', with 'ac_account' under 'JOB_ID'. The second row contains 'Doe' and '100' under 'DEPARTMENT_ID', with 'ac_account' under 'JOB_ID'. At the bottom left, it says '2 rows returned in 0.04 seconds'. At the bottom right, there are 'Save' and 'Run' buttons.

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 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. On the right, there's a search bar, a user icon, and a session identifier 'ashwin_033 a index_033'. The main workspace has tabs for 'Language' (set to 'SQL'), 'Rows' (set to 10), 'Clear Command', and 'Find Tables'. Below the tabs is a toolbar with icons for search, refresh, and other functions. The SQL command entered is:

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

The results section shows a table with two rows:

LAST_NAME	SALARY
Davies	11000
Mohan	4000

Below the table, it says '2 rows returned in 0.01 seconds' and has 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 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. On the right, there's a search bar, a user icon, and a session identifier 'Dharaneeshwar S 063-dbms'. The main workspace has tabs for 'Language' (set to 'SQL'), 'Rows' (set to 10), 'Clear Command', and 'Find Tables'. Below the tabs is a toolbar with icons for search, refresh, and other functions. The SQL command entered is:

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

The results section shows a table with three rows:

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 has 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 module 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 are displayed in a grid:

EMPLOYEE_ID	LAST_NAME	SALARY
1001	Smith	70000
142	Doeu	30000

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

At the bottom of the page, there is a table with the following data:

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:

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

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:

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

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:

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. On the right, there are user profile icons for 'ashwin_033' and 'index_033'. The main workspace shows a SQL command window with the following code:

```

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

```

The results pane displays the output of the query:

JOB_ID	DEPARTMENT_ID
ac_account	20
hr_rep	20

Below the results, it says "2 rows returned in 0.01 seconds". The bottom of the screen shows the URL "220701045@rajalakshmi.edu.in", the user "bhavath", and the session ID "101". Copyright information "Copyright © 1999-2025, Oracle and/or its affiliates." and the version "Oracle APEX 23.2.4" are also visible.

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:

A screenshot of the Oracle APEX SQL Workshop interface, similar to the previous one but with a different query. The top navigation bar and user profile are identical. The SQL command window 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 pane displays the output of the query:

JOB_ID	EMPLOYEE_ID
ac_account	115
ac_account	142
sales_rep	1001

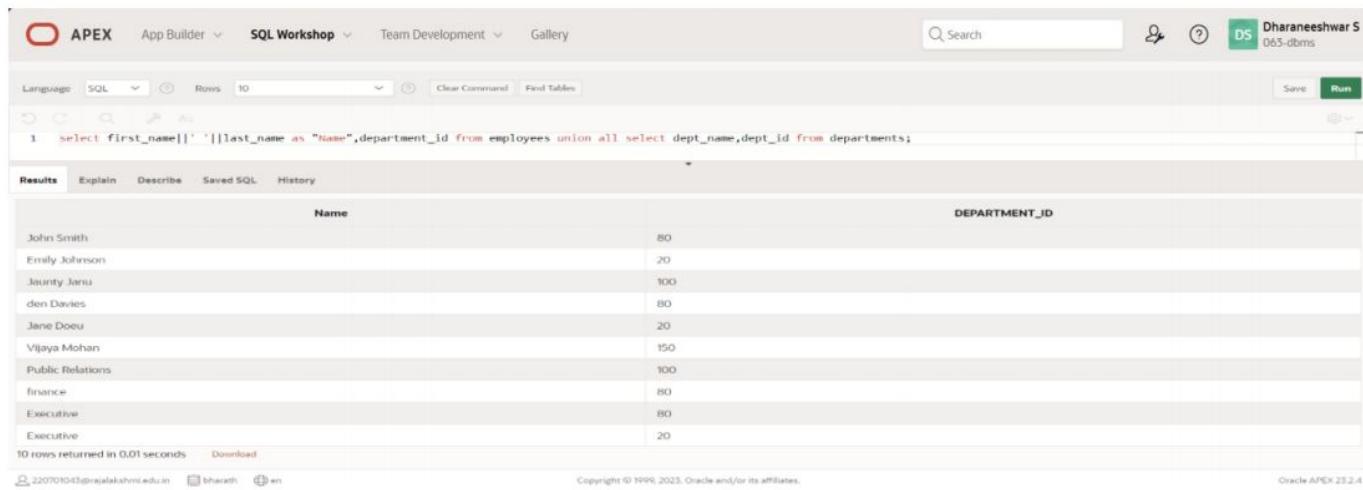
Below the results, it says "3 rows returned in 0.03 seconds". The bottom of the screen shows the URL "220701045@rajalakshmi.edu.in", the user "bhavath", and the session ID "101". Copyright information "Copyright © 1999-2025, Oracle and/or its affiliates." and the version "Oracle APEX 23.2.4" are also visible.

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. The query entered is:

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

The results table has 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

10 rows returned in 0.01 seconds

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

RESULT:

CREATING VIEWS

EX_NO:11

DATE:

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

QUERY:

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

OUTPUT:

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

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

Below the command, the results section shows the message: "View created." and "0.07 seconds".

2.) Display the contents of the EMPLOYEES_VU view.**QUERY:**

```
select * from employees_vu;
```

OUTPUT:

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

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

The results section displays the data from the view:

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

At the bottom, it says "6 rows returned in 0.03 seconds".

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

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

OUTPUT:

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

```

1 SELECT view_name, text
2 FROM user_views;
3

```

Under the 'Results' tab, the output is displayed in two columns: 'VIEW_NAME' and 'TEXT'. The result for 'EMPLOYEES_VU' is:

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

Below the results, it says '1 rows returned in 0.06 seconds' and has a 'Download' link.

At the bottom of the page, there are user details: '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.

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

QUERY:

`SELECT employee, department_id FROM employees_vu;`

OUTPUT:

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

```

1 SELECT employee, department_id FROM employees_vu;

```

Under the 'Results' tab, the output is displayed in two columns: 'EMPLOYEE' and 'DEPARTMENT_ID'. The results are:

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

Below the results, it says '6 rows returned in 0.01 seconds' and has a 'Download' link.

At the bottom of the page, there are user details: '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.

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'. On the right, there's a search bar, user profile, and a green status badge for 'Dharaneeshwar S 065-dbms'. The main workspace has tabs for 'Language: SQL', 'Rows: 10', 'Clear Command', and 'Find Tables'. Below these are icons for 'New', 'Edit', 'Run', and 'Save'. The SQL command entered is:

```
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;
```

The results tab shows the message 'View created.' and a execution time of '0.06 seconds'. At the bottom, it displays the session information '220701043@rajalakshmi.edu.in', 'Bharath', and 'en', along with copyright notice 'Copyright © 1999-2023, Oracle and/or its affiliates.' and 'Oracle APEX 23.2.4'.

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

QUERY:

```
Describe dept50;
```

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, and a green status badge for 'ashwin_033 index_033'. The main workspace has tabs for 'Language: SQL', 'Rows: 10', 'Clear Command', and 'Find Tables'. Below these are icons for 'New', 'Edit', 'Run', and 'Save'. The SQL command entered is:

```
1 DESCRIBE dept50 ;
```

The results tab shows the structure of the DEPT50 view. It lists three columns: EMPNO (NUMBER), EMPLOYEE (VARCHAR2(20)), and DEPTNO (NUMBER). The 'Primary Key' column indicates that DEPTNO is the primary key. The 'Comment' column is empty. At the bottom, it displays the session information '220701043@rajalakshmi.edu.in', 'Bharath', and 'en', along with copyright notice 'Copyright © 1999-2023, Oracle and/or its affiliates.' and 'Oracle APEX 23.2.4'.

7.) Attempt to reassign Matos to department 80

QUERY:

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

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. On the right, there are search, help, and user profile icons. The main area shows a SQL command window with the following code:

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

The results tab shows the output of the query:

0 row(s) updated.
0.05 seconds

At the bottom, the footer displays the user's email (220701043@rajalakshmi.edu.in), session information (bhavath), and the Oracle APEX version (23.2.4).

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

QUERY:

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

OUTPUT:

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

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

The results tab shows the output of the query:

View created.
0.06 seconds

At the bottom, the footer displays the user's email (220701043@rajalakshmi.edu.in), session information (bhavath), and the Oracle APEX version (23.2.4).

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	NULLABLE
id	pk	NUMBER	6	0		No
name		VARCHAR2	50			Yes
date_opened		DATE				No
address		VARCHAR2	50			No
city		VARCHAR2	30			No
zip_postal_code		VARCHAR2	12			Yes
phone		VARCHAR2	20			Yes
email	uk	VARCHAR2	75			Yes
manager_id		NUMBER	6	0		Yes
emergency_contact		VARCHAR2	20			Yes

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

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

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

Table Created.

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

```
DESCRIBE f_global_locations;
```

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

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

```

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

```

PRIMARY KEY, FOREIGN KEY, and CHECK Constraints

1. What is the purpose of a
 - PRIMARY KEY
 - FOREIGN KEY
 - CHECK CONSTRAINT

a. PRIMARY KEY

Uniquely identify each row in table.

b. FOREIGN KEY

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

c. CHECK CONSTRAINT

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

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

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

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

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

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

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

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

```
SELECT * FROM animals;
```

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

COLUMN LEVEL STATEMENT:

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

TABLE LEVEL STATEMENT:

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

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

- a. ON DELETE CASCADE

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

- b. ON DELETE SET NULL

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

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

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

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

PRACTICE PROBLEM

Managing Constraints

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

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 tablenames 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 tablenames 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?

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

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

4. REPLACE view_d_songs. Add type_code to the column list. Use aliases for all columns. Or use alias after the CREATE statement as shown.

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

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

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

6. It is company policy that only upper-level management be allowed access to individual employee salaries. The department managers, however, need to know the minimum, maximum, and averagesalaries, 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

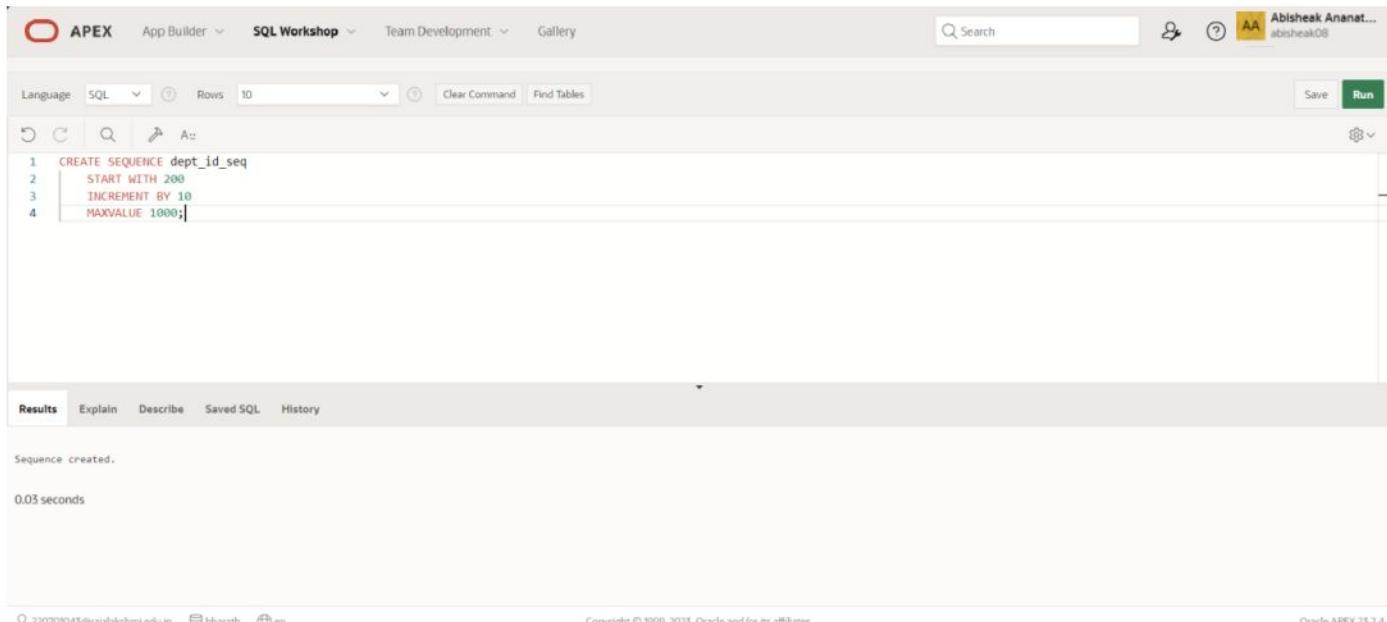
DATE:

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. Below it, 'SQL Workshop' is chosen from a dropdown menu. 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;|
```

At the bottom of the interface, the 'Results' tab is active. The output shows:

Sequence created.
0.05 seconds

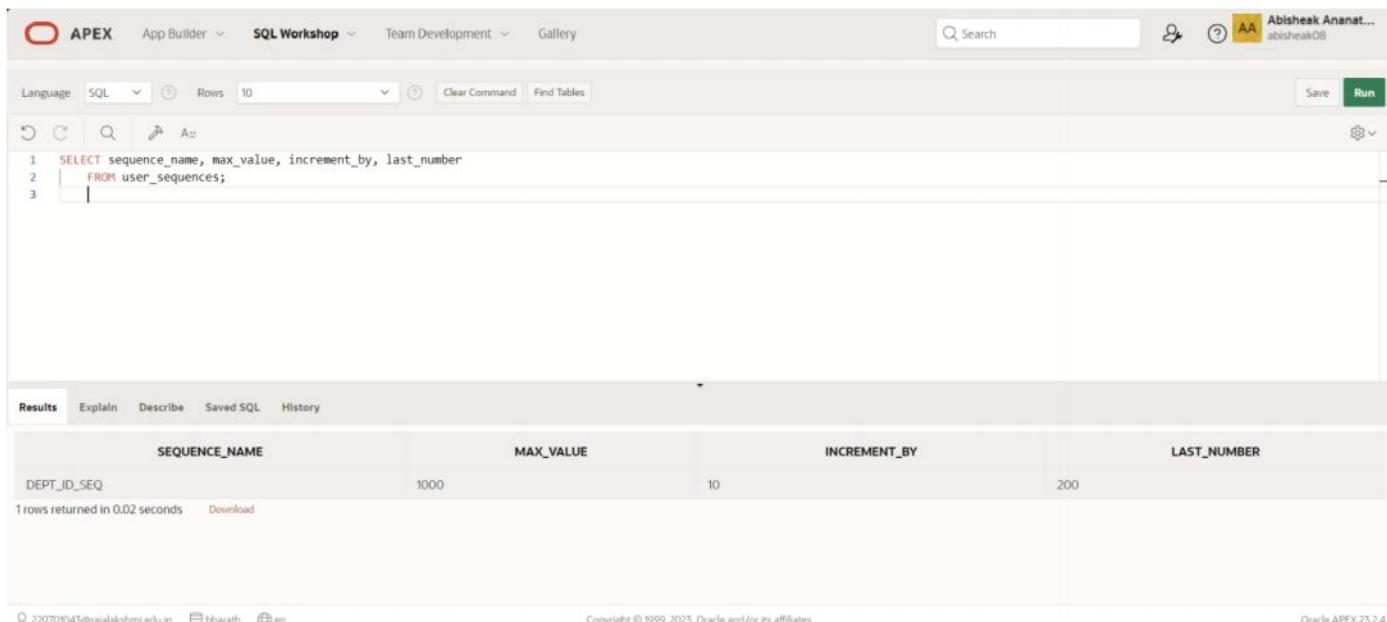
At the very bottom, the footer includes the user information '220701045@rajalakshmi.edu.in bharath en' and the copyright notice 'Copyright © 1999, 2023, Oracle and/or its affiliates.' followed by 'Oracle APEX 25.2.4'.

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

QUERY:

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

OUTPUT:



The screenshot shows the Oracle APEX SQL Workshop interface. In the top navigation bar, 'APEX' is selected. Below it, 'SQL Workshop' is chosen from a dropdown menu. The main area contains the following SQL code:

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

At the bottom of the interface, the 'Results' tab is active. The output shows a table with the following data:

SEQUENCE_NAME	MAX_VALUE	INCREMENT_BY	LAST_NUMBER
DEPT_ID_SEQ	1000	10	200

Below the table, it says '1 rows returned in 0.02 seconds'. At the bottom, there is a 'Download' link. The footer includes the user information '220701045@rajalakshmi.edu.in bharath en' and the copyright notice 'Copyright © 1999, 2023, Oracle and/or its affiliates.' followed by 'Oracle APEX 25.2.4'.

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. On the right, there's a search bar, a user icon for 'Abishek Anant...', and a status message 'Abishek Anant... abishek08'. Below the navigation is a summary table with columns: Script, View, Number, Elapsed, Statement, Feedback, and Rows. The 'Summary' view is selected. The table shows one row: Number 1, Elapsed 0.05, Statement 'INSERT INTO dept VALUES (dept_id_seq.nextval, 'Education')', Feedback '1 row(s) inserted.', and Rows 1. Below the table, it says 'Download' and 'row(s) 1 - 1 of 1'. At the bottom, there are three status indicators: 'Statements Processed' (1), 'Successful' (1), and 'With Errors' (0). The footer includes session information (220701045@rajalakshmi.edu.in, bharath, en), copyright notice (Copyright © 1999, 2023, Oracle and/or its affiliates.), and version (Oracle APEX 25.2.4).

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 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. On the right, there's a search bar, a user icon for 'Abishek Anant...', and a status message 'Abishek Anant... abishek08'. Below the navigation is a SQL editor with tabs: Language (SQL), Rows (10), Clear Command, Find Tables, Save, and Run. The SQL code entered is: 'CREATE INDEX emp_dept_id_idx ON EMPLOYEES (department_id);'. The results tab at the bottom shows the output: 'Index created.' and '0.03 seconds'. The footer includes session information (220701045@rajalakshmi.edu.in, bharath, en), copyright notice (Copyright © 1999, 2023, Oracle and/or its affiliates.), and version (Oracle APEX 25.2.4).

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 are user profile icons for 'Abishek Ananat...' and a search bar. The main area has a toolbar with various icons like Refresh, Search, and Run. Below the toolbar, the SQL editor shows the query:

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

The results section shows the output of the query:

INDEX_NAME	TABLE_NAME	UNIQUENESS
EMP_DEPT_ID_IDX	EMPLOYEES	NONUNIQUE

Below the table, it says "1 rows returned in 0.03 seconds" and there is a "Download" link. At the bottom of the page, there are footer links for Oracle ID, Help, and Log Out, along with copyright information: "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:

CONTROLLING USER ACCESS

EX_NO:15

DATE:

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

The CREATE SESSION system privilege

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

The CREATE TABLE privilege

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

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

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

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

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

The ALTER USER statement

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

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

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

7. Query all the rows in your DEPARTMENTS table.

SELECT * FROM departments;

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

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

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

9. Query the USER_TABLES data dictionary to see information about the tables that you own.

SELECT table_name FROM user_tables;

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

Team 1 revokes the privilege.

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

Team 2 revokes the privilege.

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

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

Team 1 executes this INSERT statement.

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

Team 2 executes this INSERT statement.

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

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

RESULT:

PL/SQL

CONTROL STRUCTURES

EX NO:

DATE:

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

QUERY:

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

OUTPUT:

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.00V_DBMS_SQL_APEX_230200", line 801 ORA-06550: line 4, column 3: PL/SQL: Statement ignored". The code in the editor is:

```
1  DECLARE
2    "WELCOME" varchar2(10) := 'welcome';
3  BEGIN
4    | DBMS_Output.Put_Line("Welcome");
5  END;
6 /
7
```

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.00V_DBMS_SQL_APEX_230200", line 801 ORA-06550: line 4, column 3: PL/SQL: Statement ignored". The code in the editor is identical to the first screenshot.

```
1  DECLARE
2    | WELCOME varchar2(10) := 'welcome';
3  BEGIN
4    | DBMS_Output.Put_Line("Welcome");
5  END;
6 /
7
```

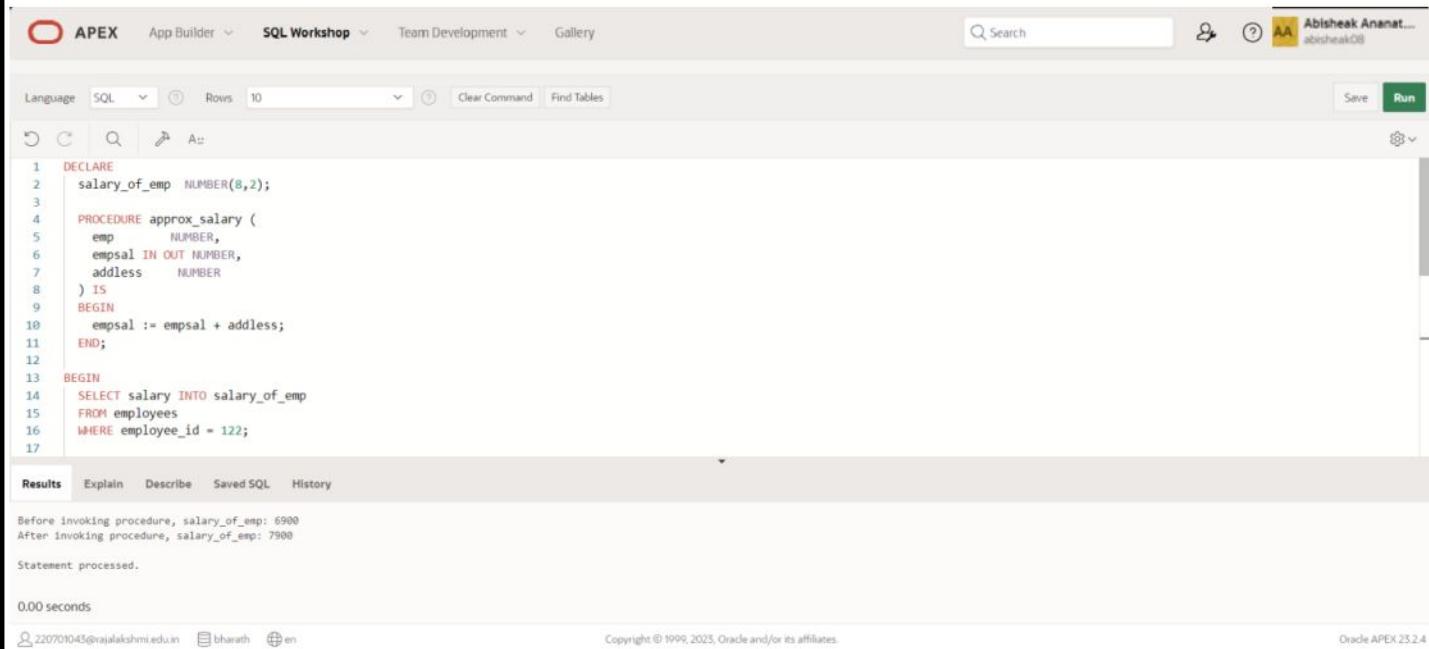
3.) Write a PL/SQL block to adjust the salary of the employee whose ID 122.

QUERY:

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

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

OUTPUT:



The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. The right side shows a user profile for 'Abishek Anand' (abishek08). The main workspace displays the PL/SQL code from the previous section. The code includes a declaration of a variable 'salary_of_emp', a procedure 'approx_salary' with parameters 'emp', 'empsal' (IN OUT), and 'addless', and logic to update 'empsal'. It also includes a 'BEGIN' block with a 'SELECT' statement to fetch the salary for employee_id 122, followed by two 'DBMS_OUTPUT.PUT_LINE' statements to print the current value of 'salary_of_emp' before and after calling the procedure. The code is numbered from 1 to 17. Below the code, the 'Results' tab is selected, showing the output of the executed statements. The output shows the initial value of 'salary_of_emp' as 6900, the output of the 'approx_salary' procedure as 7900, and a final message 'Statement processed.' at the bottom.

```
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
220701043@najalakshmi.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. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. On the right, there's a user profile for 'Abishek Ananat' (abisheak08). The main workspace has a toolbar with various icons. Below the toolbar is a code editor window containing the PL/SQL procedure code. The code is numbered from 1 to 15. The procedure creates a new procedure named 'pri_bool' with parameters 'boo_name' (VARCHAR2) and 'boo_val' (BOOLEAN). It uses an IF-ELSIF-ELSE-END IF construct to output the name and its value ('NULL', 'TRUE', or 'FALSE') using DBMS_OUTPUT.PUT_LINE. The code ends with a slash (/). At the bottom of the code editor, there are tabs for Results, Explain, Describe, Saved SQL, and History. The results tab is currently selected. The output pane below displays the message 'Procedure created.' and '0.03 seconds'. The footer of the page includes copyright information for Oracle and the APEX version (23.2.4).

```
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 |
```

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. On the right, there is a search bar, a user profile for 'Abishek Ananat...', and buttons for Save and Run.

The main workspace displays the PL/SQL code for the 'pat_match' procedure. The code uses the 'DBMS_OUTPUT.PUT_LINE' function to print 'TRUE' or 'FALSE' based on whether the 'test_string' matches the 'pattern'. It also demonstrates case sensitivity by comparing 'B%a_e' and 'B%A_E'.

Below the code, the 'Results' tab is selected, showing the output of the executed procedure. The results are:

```
TRUE
FALSE
```

At the bottom, it says 'Statement processed.' and '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

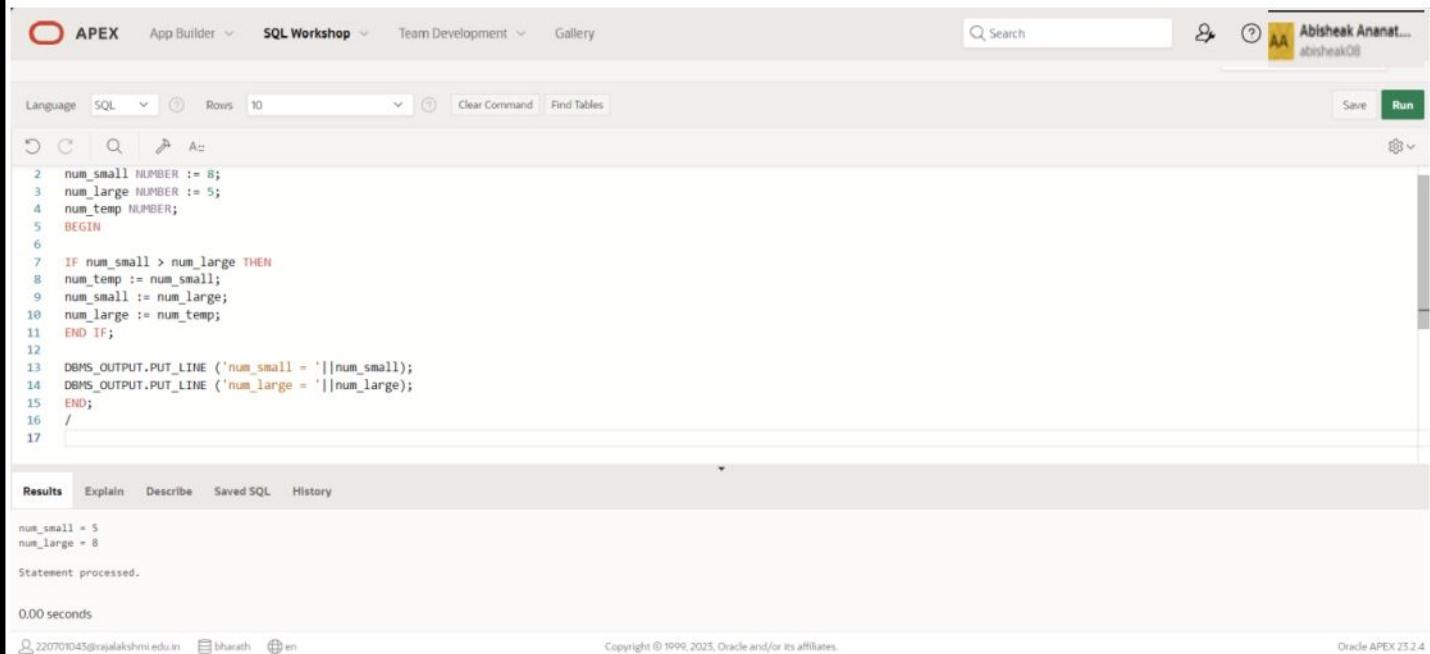
QUERY:

```
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:



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

Results Explain Describe Saved SQL History

```
num_small = 5
num_large = 8

Statement processed.

0.00 seconds
```

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

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. On the right, there are user profile icons and a search bar. The main workspace has a toolbar with various icons. The code area contains the PL/SQL procedure definition and execution details. The results pane at the bottom shows the output of the executed code.

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

Results Explain Describe Saved SQL History

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

1 row(s) updated.

0.02 seconds
```

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

QUERY:

```

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

```

The screenshot shows the Oracle APEX interface with the SQL Workshop tab selected. The code area contains the PL/SQL block provided above. The results tab shows the output of the procedure calls:

```

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

```

At the bottom, it shows the user information (220701045@rajalakshmi.edu.in, bharath, en) and the copyright notice (Copyright © 1999, 2023, Oracle and/or its affiliates).

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. On the right, there's a search bar, user profile for Abishek Ananat, and a 'Save' button. The main workspace has tabs for Language (set to SQL), Rows (set to 10), Clear Command, and Find Tables. Below these are standard toolbar icons for Undo, Redo, Search, and Run. The code editor contains a PL/SQL block:

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

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

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

At the bottom, it shows "0.00 seconds" for execution time. The footer includes copyright information for Oracle and links for user support and documentation.

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 Save Run  
A:=  
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));
```

Results Explain Describe Saved SQL History

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

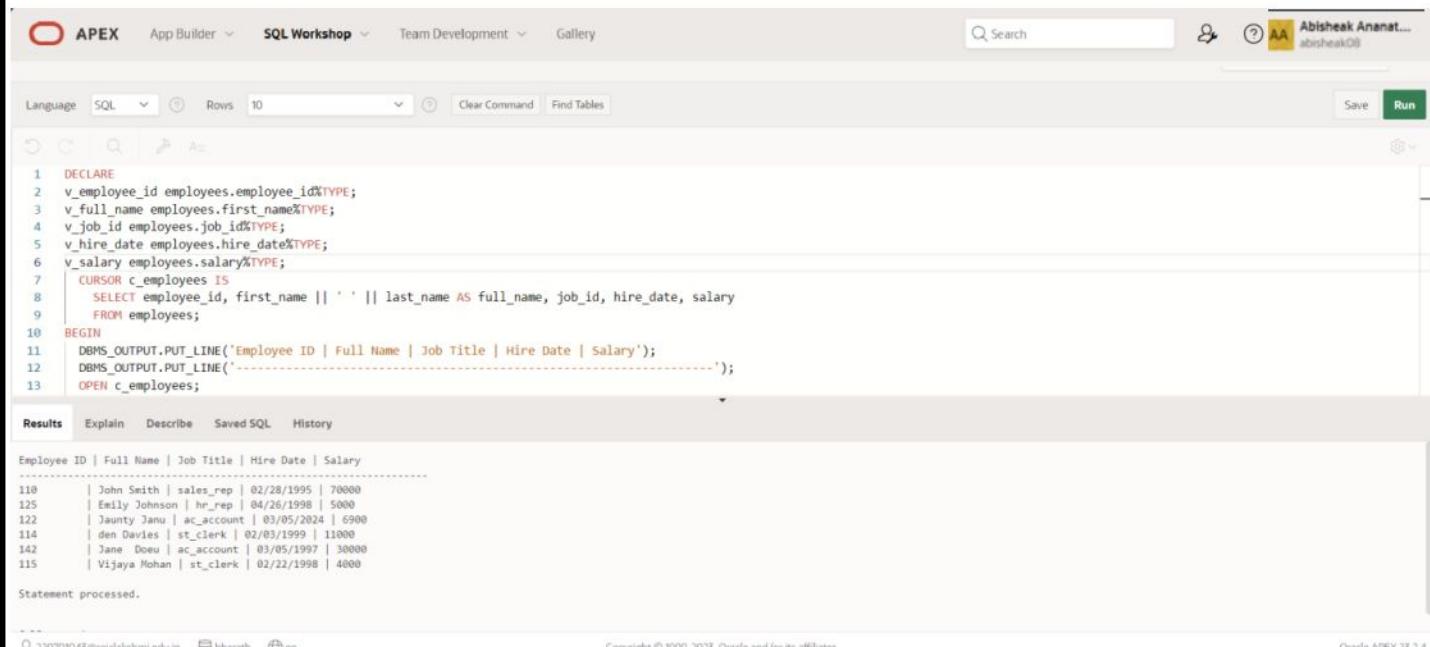
220701045@rajalakshmi.edu.in bharath en Copyright © 1999, 2025, Oracle and/or its affiliates. Oracle APEX 23.2

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

QUERY:

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

OUTPUT:



```
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

118	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

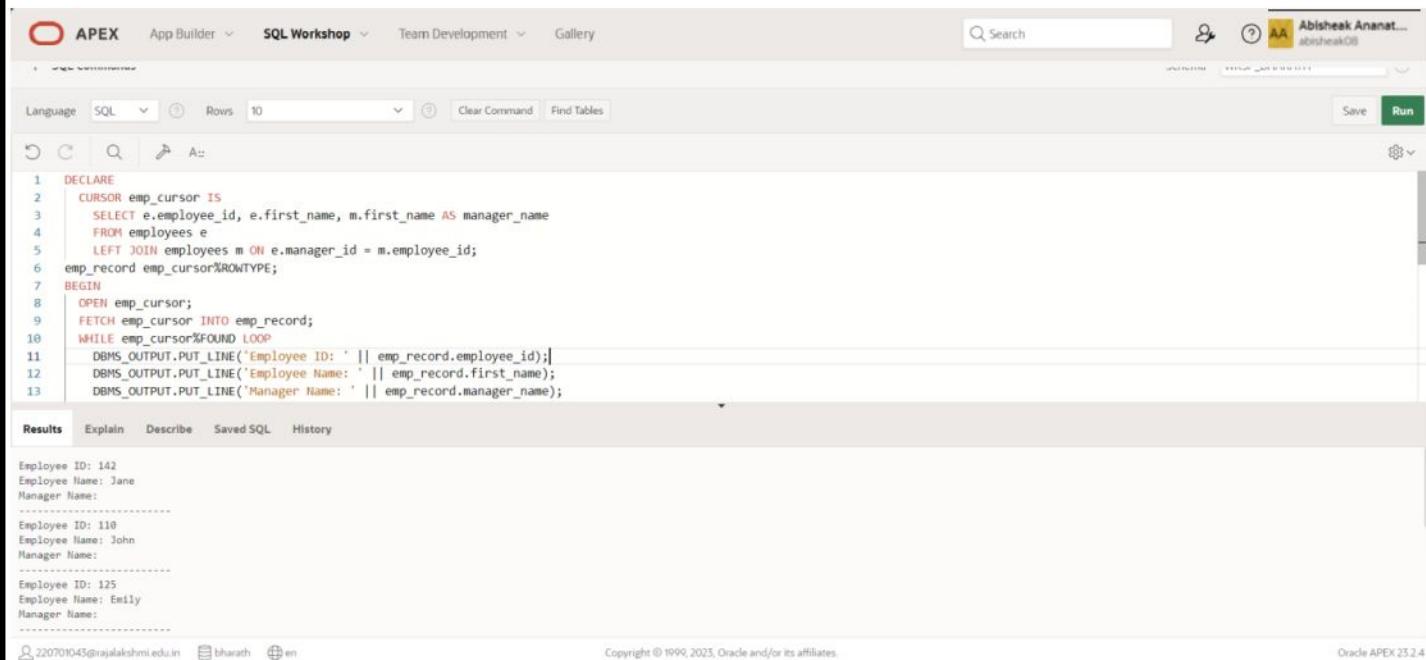
Statement processed.

- 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     END LOOP;
15     CLOSE emp_cursor;
16  END;
17 /
```

Employee ID	Employee Name	Manager Name
142	Jane	
110	John	
125	Emily	

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. The results pane displays the output of the DBMS_OUTPUT.PUT_LINE statements.

```

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;

```

Results

```

Job ID: sales_rep
Minimum Salary: 2500
-----
Job ID: hr_rep
Minimum Salary: 2500
-----
Job ID: ac_account
Minimum Salary: 2500
-----
Job ID: st_clerk
Minimum Salary: 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 code in the editor is identical to the one above. In the results pane, there is a table with four columns: Employee ID, Last Name, Job Id, and Start Date. The data is as follows:

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

Statement processed.

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

QUERY:

```
DECLARE
```

```

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

```

OUTPUT:

The screenshot shows the Oracle APEX interface with the SQL Workshop tab selected. The code area contains a PL/SQL block. The results pane displays the output of the DBMS_OUTPUT.PUT_LINE statements.

```

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

Statement processed.

```

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

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: 2.) Write a query in a script to display the following information about your sequences: sequence

The screenshot shows the Oracle APEX SQL Workshop interface. In the top navigation bar, 'APEX' is selected. Below it, 'SQL Workshop' is chosen from a dropdown menu. 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." and "0.05 seconds". The bottom of the screen shows the Oracle APEX footer with copyright information.

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. Below it, 'SQL Workshop' is chosen from a dropdown menu. 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

The results pane also shows "1 rows returned in 0.02 seconds". The bottom of the screen shows the Oracle APEX footer with copyright information.

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. The search bar contains 'Search'. On the right, there are user icons and the session identifier 'ashwin_033 a index_033'. The main area displays the results of a script execution. The 'Script' dropdown shows 'exercise14' with a status of 'Complete'. The 'Summary' view is selected, showing 15 rows processed. The results table has columns: 'Number', 'Elapsed', 'Statement', 'Feedback', and 'Rows'. One row is listed: '1' with an elapsed time of '0.05' and the statement 'INSERT INTO dept VALUES (dept_id_seq.nextval, 'Education')'. The feedback indicates '1 row(s) inserted.' and the row count is '1'. Below the table, it says 'Download' and 'row(s) 1 - 1 of 1'. At the bottom, it shows 'Statements Processed: 1', 'Successful: 1', and 'With Errors: 0'.

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. The search bar contains 'Search'. On the right, there are user icons and the session identifier 'ashwin_033 a index_033'. The main area shows the SQL command being run. The 'Language' dropdown is set to 'SQL'. The code entered is: 'CREATE INDEX emp_dept_id_idx ON EMPLOYEES (department_id);'. The results tab at the bottom shows the message 'Index created.' and a time of '0.03 seconds'. The bottom footer includes the user information '220701045@rajalakshmi.edu.in bharath en', copyright notice 'Copyright © 1999, 2023, Oracle and/or its affiliates.', and the version '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 (selected), Team Development, and Gallery. On the right, there are search, refresh, help, and user profile icons. The main area has a toolbar with Language (SQL), Rows (10), Clear Command, Find Tables, Save, and Run buttons. Below the toolbar is a code editor window containing the following SQL query:

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

Below the code editor, the results tab is selected. The output table has three columns: INDEX_NAME, TABLE_NAME, and UNIQUENESS. One row is returned, showing:

INDEX_NAME	TABLE_NAME	UNIQUENESS
EMP_DEPT_ID_IDX	EMPLOYEES	NONUNIQUE

At the bottom left, it says "1 rows returned in 0.03 seconds". At the bottom right, it says "Oracle APEX 23.2.4".

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

RESULT:

CONTROLLING USER ACCESS

EX_NO:15

DATE:

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

The CREATE SESSION system privilege

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

The CREATE TABLE privilege

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

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

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

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

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

The ALTER USER statement

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

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

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

7. Query all the rows in your DEPARTMENTS table.

SELECT * FROM departments;

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

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

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

9. Query the USER_TABLES data dictionary to see information about the tables that you own.

SELECT table_name FROM user_tables;

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

Team 1 revokes the privilege.

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

Team 2 revokes the privilege.

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

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

Team 1 executes this INSERT statement.

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

Team 2 executes this INSERT statement.

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

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

RESULT:

PL/SQL

CONTROL STRUCTURES

EX_NO:

DATE:

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. The main workspace displays the PL/SQL code. The code block is as follows:

```
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;
```

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

```
Incentive = 8400
Statement processed.

0.00 seconds
```

At the bottom of the page, there are footer links for user information (220701043@rajalakshmi.edu.in, bharath, en) and copyright information (Copyright © 1999, 2023, Oracle and/or its affiliates). The date '19 May 2024' and version 'Oracle APEX 23.2.4' are also visible.

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.00V_DBMS_SQL_APEX_230200", line 801 ORA-06550: line 4, column 3: PL/SQL: Statement ignored". The code in the editor is:

```
1 DECLARE
2   "WELCOME" varchar2(10) := 'welcome';
3 BEGIN
4   DBMS_Output.Put_Line("Welcome");
5 END;
6 /
7
```

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.00V_DBMS_SQL_APEX_230200", line 801 ORA-06550: line 4, column 3: PL/SQL: Statement ignored". The code in the editor is:

```
1 DECLARE
2   | WELCOME varchar2(10) := 'welcome';
3 BEGIN
4   | DBMS_Output.Put_Line("Welcome");
5 END;
6 /
7
```

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

QUERY:

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

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

OUTPUT:

```
1  DECLARE
2      salary_of_emp  NUMBER(8,2);
3
4  PROCEDURE approx_salary (
5      emp      NUMBER,
6      empsal IN OUT NUMBER,
7      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

220701043@najalakshmi.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. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. On the right, there's a user icon for 'ashwin_033' and a page number 'index_033'. The main area is a code editor with syntax highlighting for PL/SQL. The code is identical to the one provided in the query section. Below the code editor is a toolbar with icons for refresh, search, and run. The bottom of the screen shows the results tab is selected, displaying the message 'Procedure created.' and '0.03 seconds' execution time. The footer includes copyright information for Oracle and the APEX version '23.2.4'.

```
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 |
```

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. On the right, there are user profile icons for 'ashwin_033' and 'index_033'. The main workspace displays the PL/SQL code from the previous section. Below the code, the 'Results' tab is selected, showing the output of the procedure execution. The output consists of two rows of data: 'TRUE' and 'FALSE'. At the bottom of the results pane, it says 'Statement processed.' and '0.00 seconds'. The footer of the page includes copyright information for Oracle and the APEX version.

```
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 Explain Describe Saved SQL History

TRUE
FALSE

Statement processed.

0.00 seconds

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

Oracle APEX 23.2.4

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

QUERY:

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

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

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. The right side of the header shows a user profile for 'ashwin_033' and the session identifier 'index_033'. The main workspace is a code editor with the following content:

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

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

```
num_small = 5
num_large = 8

Statement processed.
```

At the bottom of the interface, it shows the user information '220701043@rajalakshmi.edu.in', the developer name 'bhurath', 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.

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

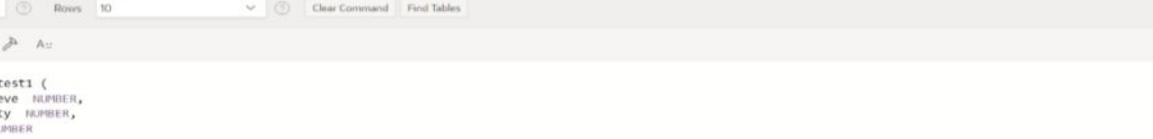
QUERY:

```

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.

QUIERY

```

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

```

The screenshot shows the Oracle APEX interface with the SQL Workshop tab selected. The code area contains the PL/SQL block provided above. The results tab shows the output of the executed code:

```

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

```

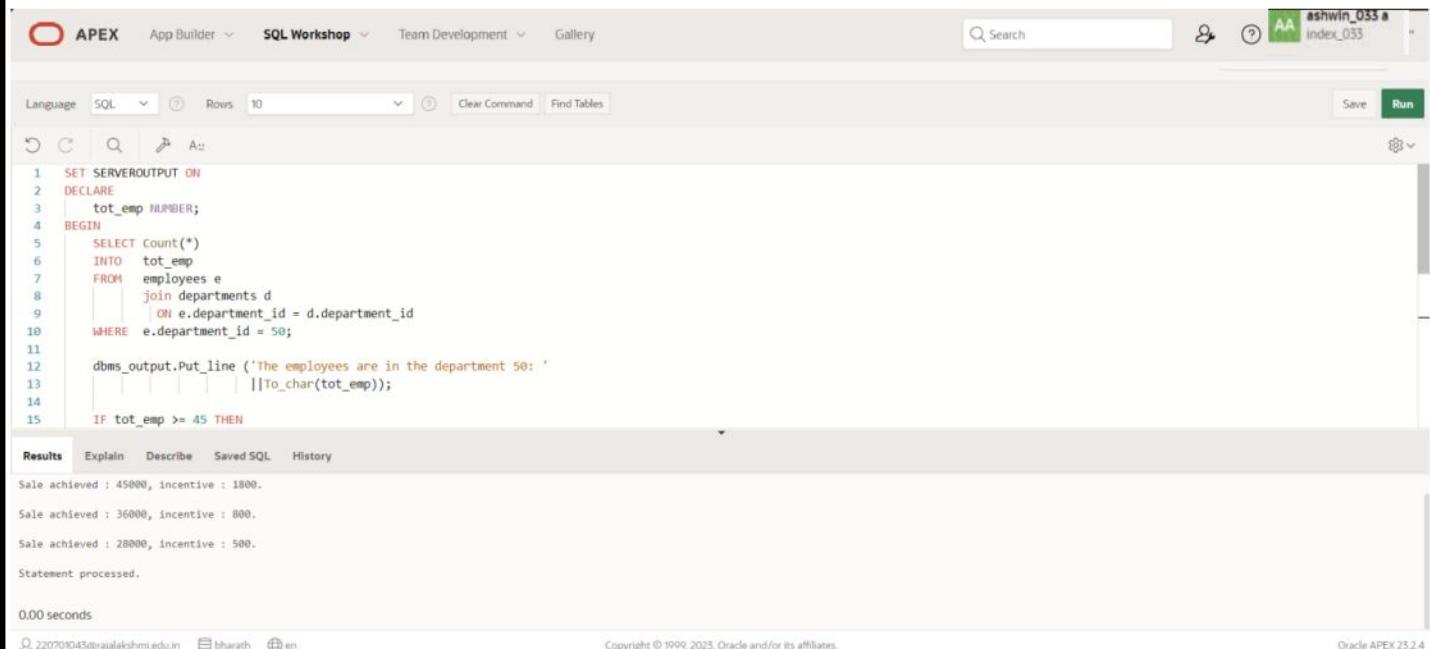
At the bottom, it shows the user information (bhara...@rajalakshmi.edu.in) and the copyright notice (Copyright © 1999, 2023, Oracle and/or its affiliates).

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 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery'. The right side shows a user profile for 'ashwin_053'.

The SQL editor window displays the following PL/SQL code:

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

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

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

0.00 seconds
```

At the bottom, it shows the user information '220701045@rajalakshmi.edu.in' and 'bhарат', the language 'en', and the copyright notice 'Copyright © 1999, 2023, Oracle and/or its affiliates.' and 'Oracle APEX 23.2.4'.

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

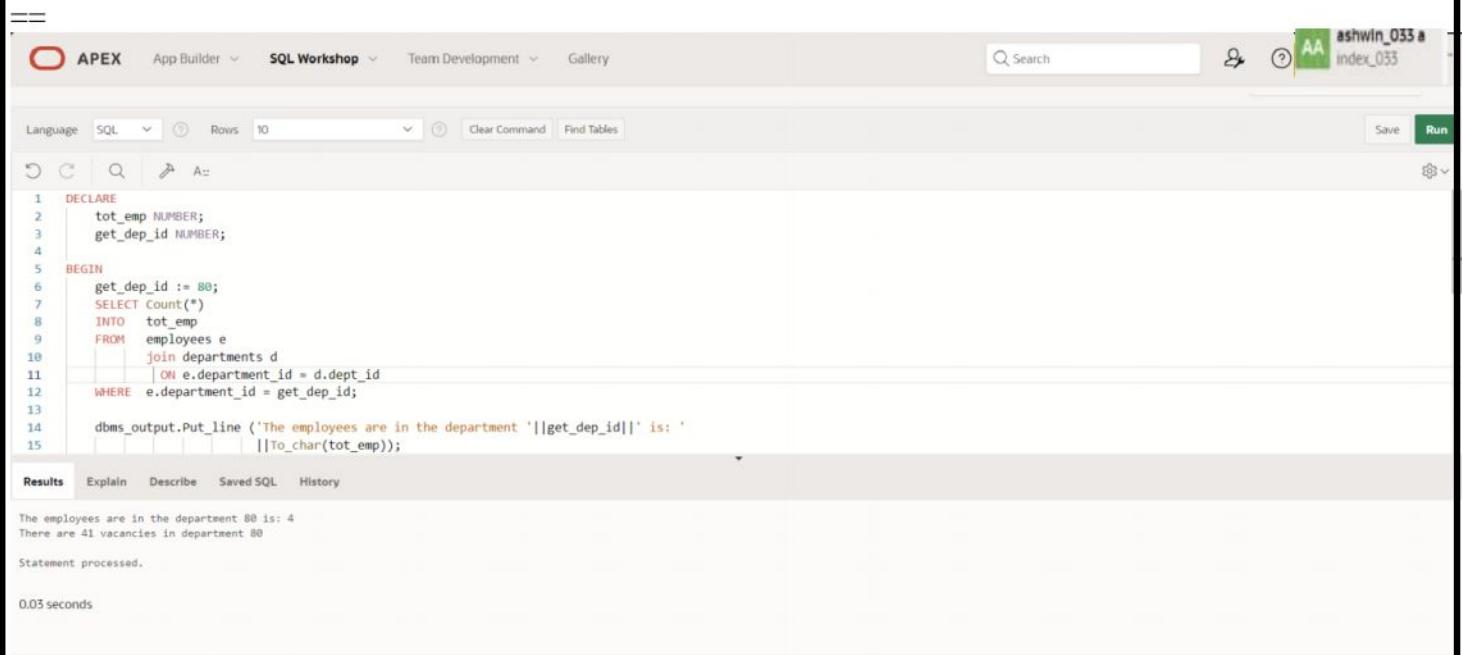
QUERY:

```
DECLARE
    tot_emp NUMBER;
    get_dep_id NUMBER;

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

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

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

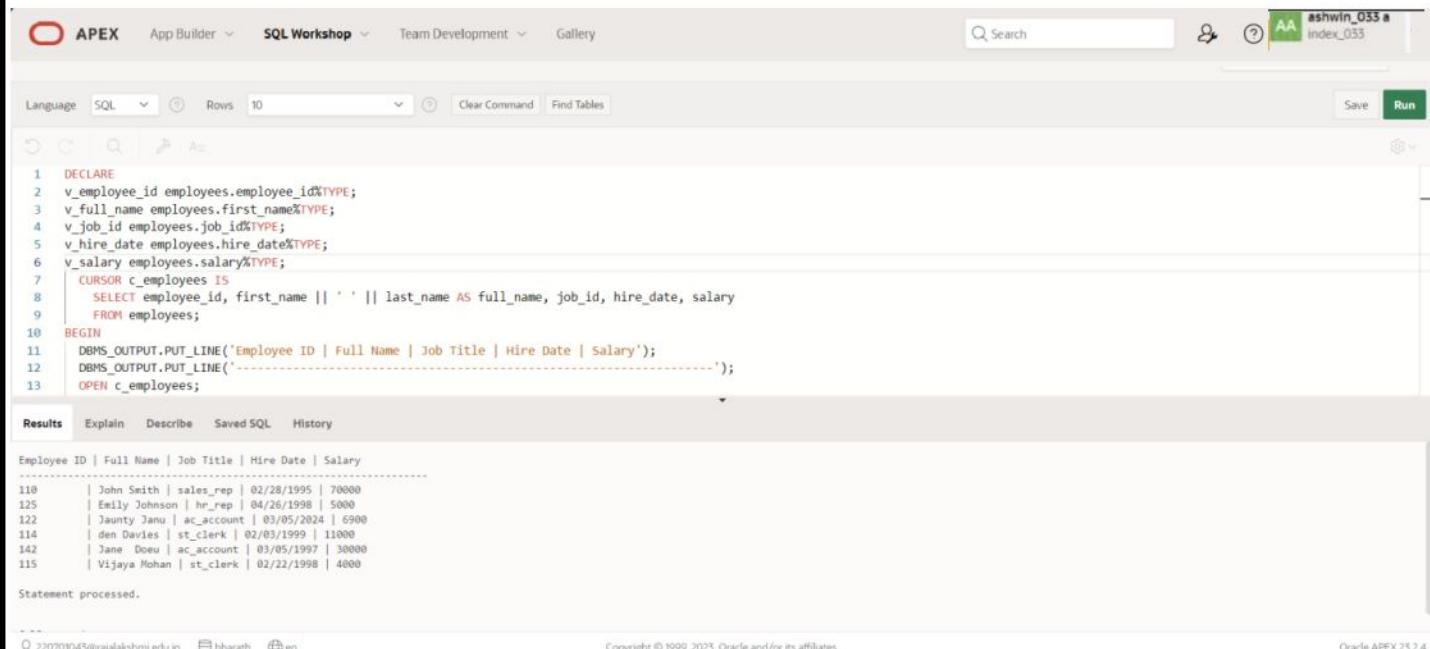
OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there's a search bar, a refresh icon, and a user profile for 'ashwin_033 index_033'. The main workspace displays the PL/SQL code. Below the code, the 'Results' tab is selected, showing the output of the executed query. The output text is:
The employees are in the department 80 is: 4
There are 41 vacancies in department 80
Statement processed.
0.03 seconds

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

QUERY:

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

OUTPUT:

```
Employee ID | Full Name | Job Title | Hire Date | Salary
-----|-----|-----|-----|-----
118 | 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
```

- 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);
-----
```

Employee ID	Employee Name	Manager Name
142	Jane	
110	John	
125	Emily	

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 entered is a PL/SQL block that retrieves job IDs and their minimum salaries from the job_grade and employees tables. The output is displayed in the Results tab, showing four records: sales_rep, hr_rep, ac_account, and st_clerk, all with a minimum salary of 2500.

```

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

```

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

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

QUERY:

```

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

```

OUTPUT:

The screenshot shows the Oracle APEX interface with the SQL Workshop tab selected. The code area contains the PL/SQL block provided above. The results tab displays the output of the program, which includes the employee ID, last name, job ID, and start date for two employees.

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

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 'ashwin_033 a index_033'. The main area has tabs for 'Language' (set to 'SQL'), 'Rows' (set to 10), 'Clear Command', 'Find Tables', 'Save', and 'Run'. Below this is a code editor with the following PL/SQL block:

```

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);

```

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

```

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

Statement processed.

```

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

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

RESULT:

PROCEDURES AND FUNCTIONS

EX_NO: 17

DATE:

1.) Factorial of a number using function.

QUERY:

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

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. The right side shows a user profile for 'ashwin_035 a index_035'. The main area is titled 'SQL Commands' with a 'Language' dropdown set to 'SQL'. Below the code editor, there are buttons for 'Save' and 'Run'. The code itself is a PL/SQL block to calculate the factorial of a number. The output panel at the bottom shows the result '479001600' and a message 'Statement processed.' The execution time was 0.00 seconds.

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

Results Explain Describe Saved SQL History

479001600
Statement processed.
0.00 seconds

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

QUERY:

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

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

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

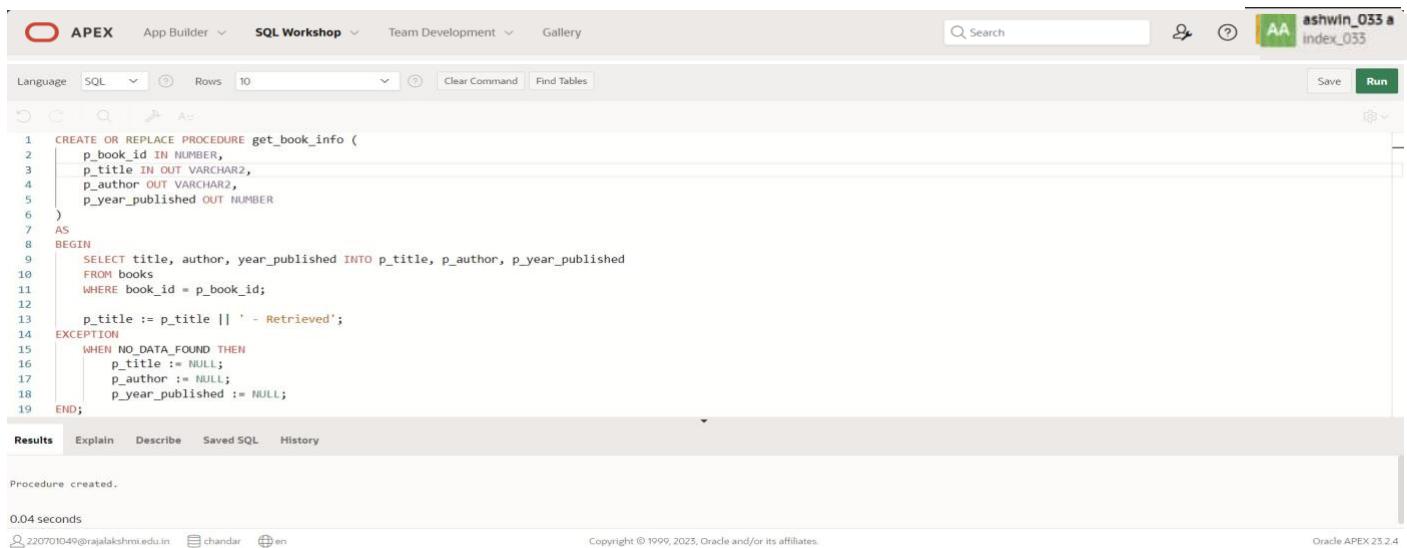
```

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

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

```

OUTPUT:



The screenshot shows the Oracle APEX 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 'ashwin_053 a index_053'. The main area displays the PL/SQL code for the 'get_book_info' procedure. The code uses DBMS_OUTPUT to print the title, author, and year published. It also handles the case where no data is found by setting all parameters to NULL. The code is as follows:

```

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' tab is selected at the bottom, showing the message 'Procedure created.' and a execution time of '0.04 seconds'. The footer includes copyright information for Oracle and the APEX version 'Oracle APEX 23.2.4'.

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

RESULT:

TRIGGER

EX_NO: 18

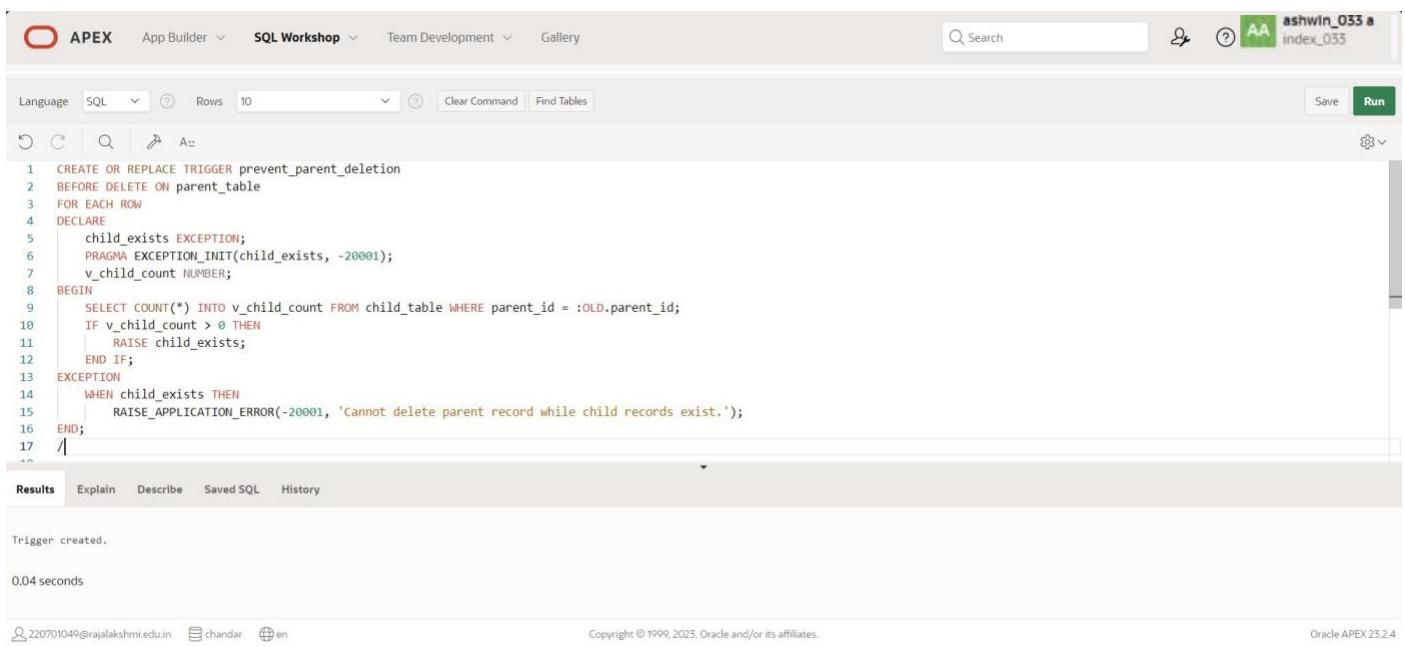
DATE:

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

QUERY:

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

OUTPUT:



The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. The main workspace shows the SQL command for creating the trigger:

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

The 'Results' tab at the bottom displays 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, 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery' are listed. On the right side, there is a search bar, a user icon, a help icon, and a session ID 'ashwin_033 a index_033'. The main area is titled 'SQL Workshop' with a dropdown menu set to 'Rows: 10'. Below this are buttons for 'Clear Command' and 'Find Tables'. The code editor contains the following PL/SQL code:

```

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

```

Below the code editor, tabs for 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History' are visible. The results section shows the message 'Trigger created.' and a execution time of '0.04 seconds'. At the bottom, the user information '220701049@rajalakshmi.edu.in' and 'chandar' is shown, along with copyright information 'Copyright © 1999, 2023, Oracle and/or its affiliates.' 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'. On the right, there's a search bar, a user icon, and a session identifier 'ashwin_033 a index_033'. The main workspace is titled 'Language: SQL' with 'Rows: 10' and a 'Run' button. Below the title, there are icons for refresh, search, and other operations. The code area contains 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;
```

Below the code, the results tab is selected, showing the message 'Trigger created.' and a execution time of '0.05 seconds'. The bottom status bar shows the user '220701049@rajalakshmi.edu.in', the session 'chandar', and the language 'en'. It also includes copyright information 'Copyright © 1999, 2025, Oracle and/or its affiliates.' and the version 'Oracle APEX 25.2.4'.

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 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. On the right, there's a user icon for 'ashwin_033 a' and a workspace name 'index_033'. The main area has tabs for 'Language' (set to 'SQL'), 'Rows' (set to 10), 'Clear Command', and 'Find Tables'. Below these are standard database navigation icons (refresh, search, etc.). The SQL editor contains the trigger creation script:

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

The 'Results' tab is selected, showing the output: "Trigger created." and "0.04 seconds". At the bottom, the footer includes user information (220701049@rajalakshmi.edu.in, chandar, en), copyright notice (Copyright © 1999, 2023, Oracle and/or its affiliates.), and the version (Oracle APEX 25.2.4).

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

```

1 CREATE OR REPLACE TRIGGER 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 output:

Trigger created.

0.05 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'.

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:

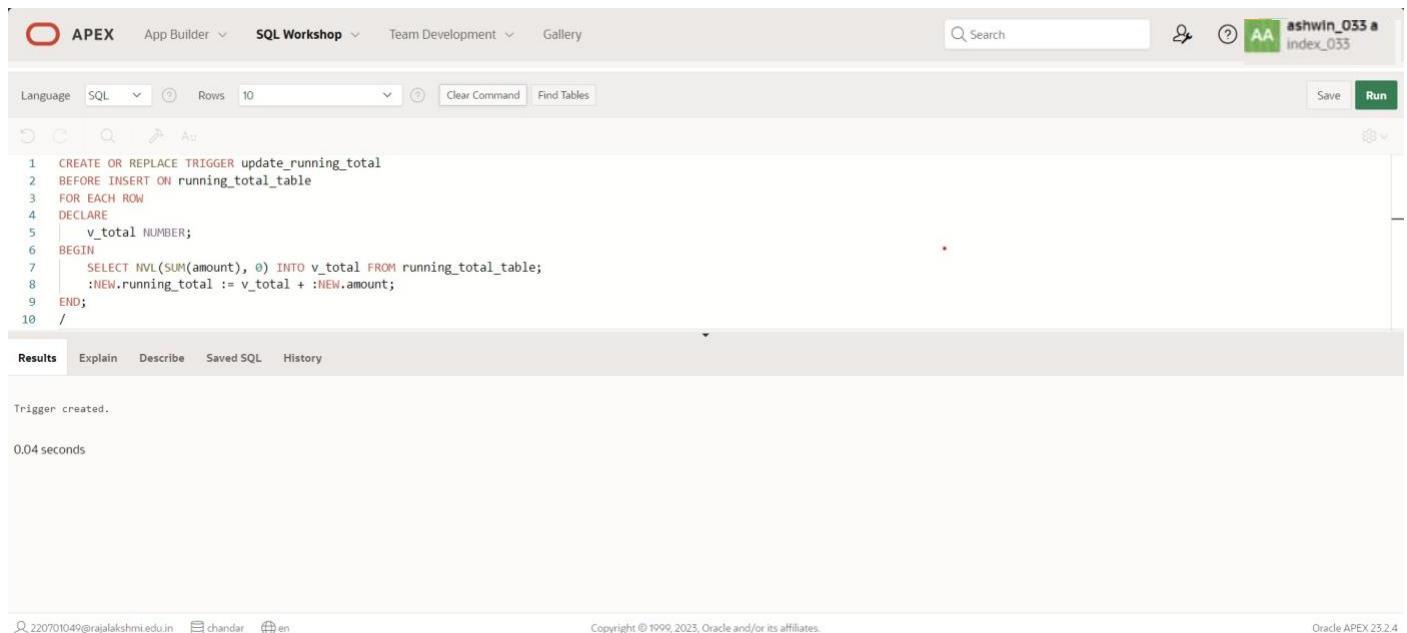
```

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 SQL editor contains the following PL/SQL code:

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

The code creates a trigger named "update_running_total" that runs before an insert operation on the "running_total_table". It declares a variable "v_total" of type NUMBER. Inside the trigger body, it selects the sum of "amount" from the table and stores it in "v_total". Then, it updates the "running_total" column of the new row by adding the value of "amount" to "v_total". Finally, it ends the trigger definition with a slash (/).

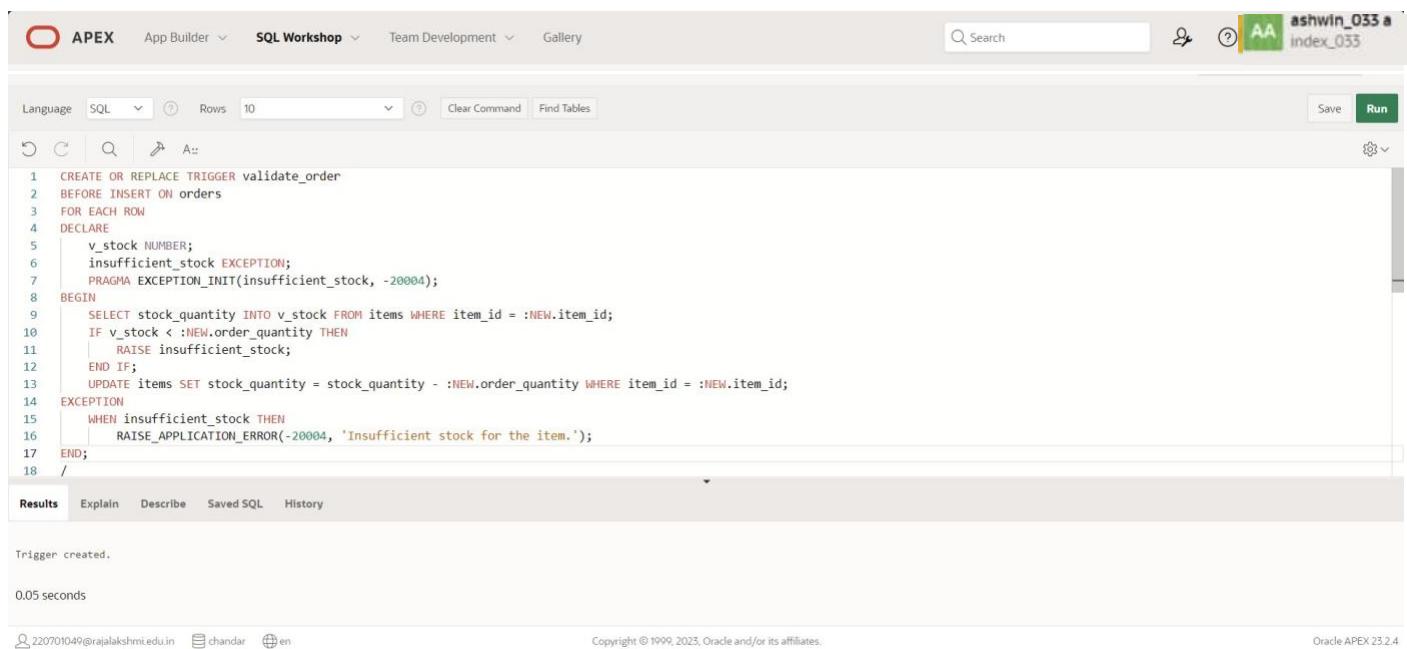
Below the code, the "Results" tab is active, showing the output: "Trigger created." and "0.04 seconds". The bottom of the screen displays user information and copyright notices.

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 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. The search bar contains 'Search'. On the right, there's a user icon for 'ashwin_033' and a file named 'index_033'. The main workspace is titled 'SQL' and shows the following PL/SQL code:

```
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 /
```

Below the code, the 'Results' tab is selected, showing the output: 'Trigger created.' and '0.05 seconds'. The bottom footer includes user information (220701049@rajalakshmi.edu.in, chandar, en), copyright notice (Copyright © 1999, 2023, Oracle and/or its affiliates.), and version information (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. In the top bar, there are buttons for English, Recent, Login, and Sign up. Below the bar, there is a search input field labeled "Enter a title...". Underneath it, there are two tabs: "MongoDB" (selected) and "Info". On the right side, there are "Run" and "Save" buttons. The main area contains a code editor with the following MongoDB query:

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

To the right of the code editor is an "Output" window. It shows the command "mycompiler_mongodb> ...". Below it, the response from the database is displayed:

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

At the bottom of the output window, there is a decorative footer with icons and the text "Redesign the way you jam with 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 survey dates.

QUERY:

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

OUTPUT:

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

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

To the right of the code editor is an "Output" window. It shows the command "mycompiler_mongodb> ...". Below it, the response from the database is displayed:

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

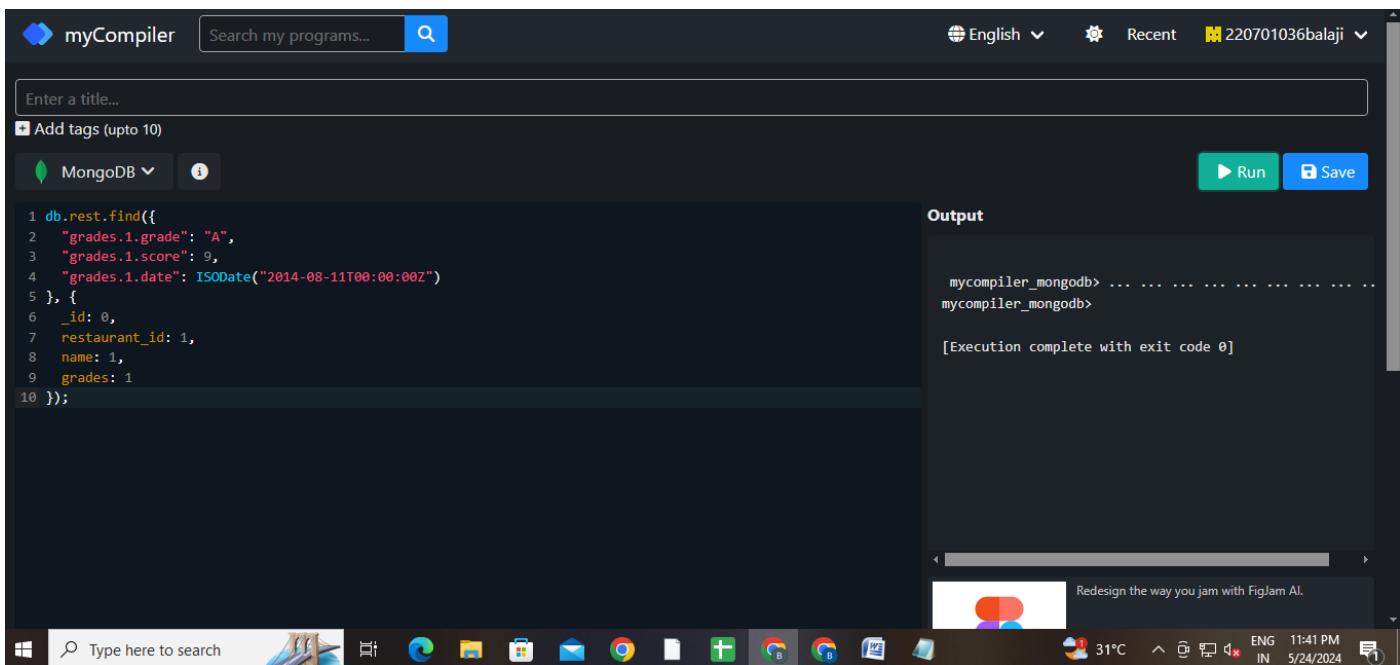
At the bottom of the output window, there is a decorative footer with icons and the text "Redesign the way you jam with FigJam AI."

3.) Write a MongoDB query to find the restaurant Id, name and grades for those restaurants where the 2nd element of grades array contains a grade of "A" and score 9 on an ISODate "2014-08-11T00:00:00Z".

QUERY:

```
db.restaurants.find( {"grades.1.grade": "A", "grades.1.score": 9, "grades.1.date": ISODate("2014-08-1T00:00:00Z") }, { restaurant_id: 1, name: 1, grades: 1 } );
```

OUTPUT:



The screenshot shows the myCompiler application interface. In the top bar, there's a search bar for programs and a language selection dropdown set to English. On the right, there are buttons for Recent projects and a user ID (220701036balaji). Below the header, there's a text input field for titles and a button to add tags (up to 10). A dropdown menu shows 'MongoDB' is selected. On the left, the code editor contains a MongoDB query:

```
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});
```

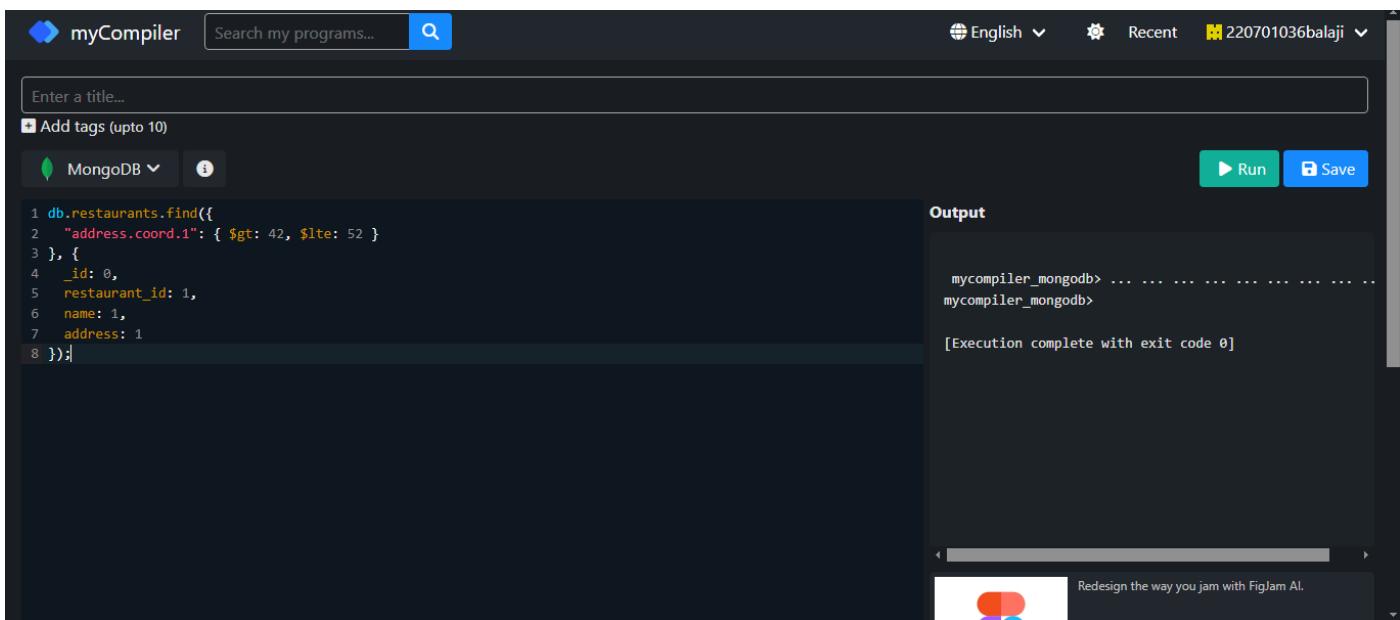
On the right, the 'Output' panel shows the command being run and the completion message: [Execution complete with exit code 0]. At the bottom of the window, there's a toolbar with various icons and a status bar showing the date and time (5/24/2024, 11:41 PM).

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 application interface. In the top bar, there's a search bar for programs and a language selection dropdown set to English. On the right, there are buttons for Recent projects and a user ID (220701036balaji). Below the header, there's a text input field for titles and a button to add tags (up to 10). A dropdown menu shows 'MongoDB' is selected. On the left, the code editor contains a MongoDB query:

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

On the right, the 'Output' panel shows the command being run and the completion message: [Execution complete with exit code 0]. At the bottom of the window, there's a toolbar with various icons and a status bar showing the date and time (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 application interface. In the top bar, there is a search bar for programs and a language selection dropdown set to English. On the right, there are buttons for Recent projects and a user profile. The main area has tabs for MongoDB and MySQL, with MongoDB selected. A code editor window contains the following MongoDB query:

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

To the right of the code editor is an "Output" panel. It displays the command-line interface of a MongoDB shell session:

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

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

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

QUERY:

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

OUTPUT:

The screenshot shows the myCompiler application interface. In the top bar, there is a search bar for programs and a language selection dropdown set to English. On the right, there are buttons for Recent projects and a user profile. The main area has tabs for MongoDB and MySQL, with MongoDB selected. A code editor window contains the following MongoDB query:

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

To the right of the code editor is an "Output" panel. It displays the command-line interface of a MongoDB shell session:

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

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

7.) Write a MongoDB query to arranged the name of the cuisine in ascending order and for that same cuisine borough should be in descending order.

QUERY:

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

OUTPUT:

The screenshot shows the myCompiler interface. In the top bar, there are icons for English, Recent, and a user profile (220701036balaji). Below the bar, there's a search bar with placeholder text "Enter a title..." and a magnifying glass icon. A "Add tags (upto 10)" button is also present. The main area has tabs for "MongoDB" and "Run". A code editor window contains the following MongoDB query:

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

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

8.) Write a MongoDB query to know whether all the addresses contains the street or not.

QUERY:

```
db.restaurants.find({ "address.street": { $exists: true, $ne: "" } })
```

OUTPUT:

The screenshot shows the myCompiler interface. In the top bar, there are icons for English, Recent, and a user profile (220701036balaji). Below the bar, there's a search bar with placeholder text "Enter a title..." and a magnifying glass icon. A "Add tags (upto 10)" button is also present. The main area has tabs for "MongoDB" and "Run". A code editor window contains the following MongoDB query:

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

To the right of the code editor is an "Output" panel. It displays the command "mycompiler_mongodb>" followed by "[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 interface. In the code editor, a MongoDB query is written:

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

The output window shows the results of the query execution:

```
mycompiler_mongodb> ... ... ... ...
[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 interface. In the code editor, a MongoDB query is written:

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

The output window shows the results of the query execution:

```
mycompiler_mongodb> ... ...
[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 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 selection ("English"), recent projects ("Recent"), and user information ("220701036balaji"). Below the search bar is a text input field with placeholder "Enter a title...". Underneath it is a "Add tags (upto 10)" button. The main area has tabs for "MongoDB" (selected) and "i". A code editor window contains the following MongoDB query:

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

To the right of the code editor is an "Output" panel. It shows the command "mycompiler_mongodb> ..." followed by "[Execution complete with exit code 0]". At the bottom right of the interface, there's a small advertisement for FigJam AI.

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

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

The "Output" panel shows the command "mycompiler_mongodb> ..." followed by "[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 web application 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 for entering a title, followed by a checkbox for adding tags. A MongoDB connection dropdown is set to "MongoDB". On the right side of the interface are two buttons: "Run" and "Save". The main area contains 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. It displays the command "mycompiler_mongodb> ...", the response "mycompiler_mongodb> ...", and the message "[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 web application interface. The setup is identical to the previous screenshot, with the MongoDB connection set to "MongoDB". The code editor contains the same MongoDB query as question 14. The output panel shows the command "mycompiler_mongodb> ...", the response "mycompiler_mongodb> ...", and the message "[Execution complete with exit code 0]".

15. Write a MongoDB query to find the restaurants that have at least one grade with a score of less than 5 and that are located in the borough of Manhattan or Brooklyn.

QUERY:

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

OUTPUT:

The screenshot shows the myCompiler interface. At the top, there's a search bar with "Search my programs..." and a magnifying glass icon. To the right are language settings (English), recent projects (Recent), and user information (220701036balaji). Below the search bar is a text input field with placeholder "Enter a title...". Underneath it is a "Add tags (upto 10)" button. On the left, there's a "MongoDB" dropdown and an "Info" button. 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 output of the query:

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

16. Write a MongoDB query to find the restaurants that have at least one grade with a score of less than 5 and that are located in the borough of Manhattan or Brooklyn, and their cuisine is not American.

QUERY:

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

OUTPUT:

The screenshot shows the myCompiler interface. At the top, there's a search bar with "Search my programs..." and a magnifying glass icon. To the right are language settings (English), recent projects (Recent), and user information (220701036balaji). Below the search bar is a text input field with placeholder "Enter a title...". Underneath it is a "Add tags (upto 10)" button. On the left, there's a "MongoDB" dropdown and an "Info" button. On the right, there are "Run" and "Save" buttons. The main area contains a code editor with the following MongoDB query:

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

Below the code editor is a terminal window showing the output of the query:

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

17. Write a MongoDB query to find the restaurants that have at least one grade with a score of less than 5 and that are located in the borough of Manhattan or Brooklyn, and their cuisine is not American or Chinese.

QUERY:

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

OUTPUT:

The screenshot shows the myCompiler MongoDB interface. In the top left, there's a logo and the text "myCompiler". A search bar contains "Search my programs..." with a magnifying glass icon. On the right, there are language ("English"), recent projects ("Recent"), and user information ("220701036balaji"). Below the search bar is a text input field with placeholder "Enter a title...". Underneath it is a button "Add tags (upto 10)". The main area has tabs for "MongoDB" (selected) and "i". On the far right are "Run" and "Save" buttons. The code input field contains the following query:

```
1$lt: 5 }, cuisine: { $nin: ["American", "Chinese"] } }, { _id: 0, name: 1, borough: 1, address: 1, cuisine: 1 })
```

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

```
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 MongoDB interface. The setup is identical to the previous one, with the "MongoDB" tab selected. The code input field contains the query from question 18:

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

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

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

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

QUERY:

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

OUTPUT:

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

```
mycompiler_mongodb> ...
mycompiler_mongodb>

[Execution complete with exit code 0]
```



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

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

QUERY:

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

OUTPUT:

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

```
mycompiler_mongodb> ...
mycompiler_mongodb>

[Execution complete with exit code 0]
```



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

21. Write a MongoDB query to find the restaurants that have a grade with a score of 2 and a grade with a score of 6 and are located in the borough of Manhattan or Brooklyn, and their cuisine is not American.

QUERY:

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

OUTPUT:

22. Write a MongoDB query to find the restaurants that have a grade with a score of 2 and a grade with a score of 6 and are located in the borough of Manhattan or Brooklyn, and their cuisine is not American or Chinese.

QUERY:

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

OUTPUT:

Add tags (upto 10)
 MongoDB   Run  Save

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

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

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

QUERY:

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

OUTPUT:

Enter a title...

Add tags (upto 10)

MongoDB  

```
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]
```

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

RESULT:

MONGO DB

EX_NO: 20

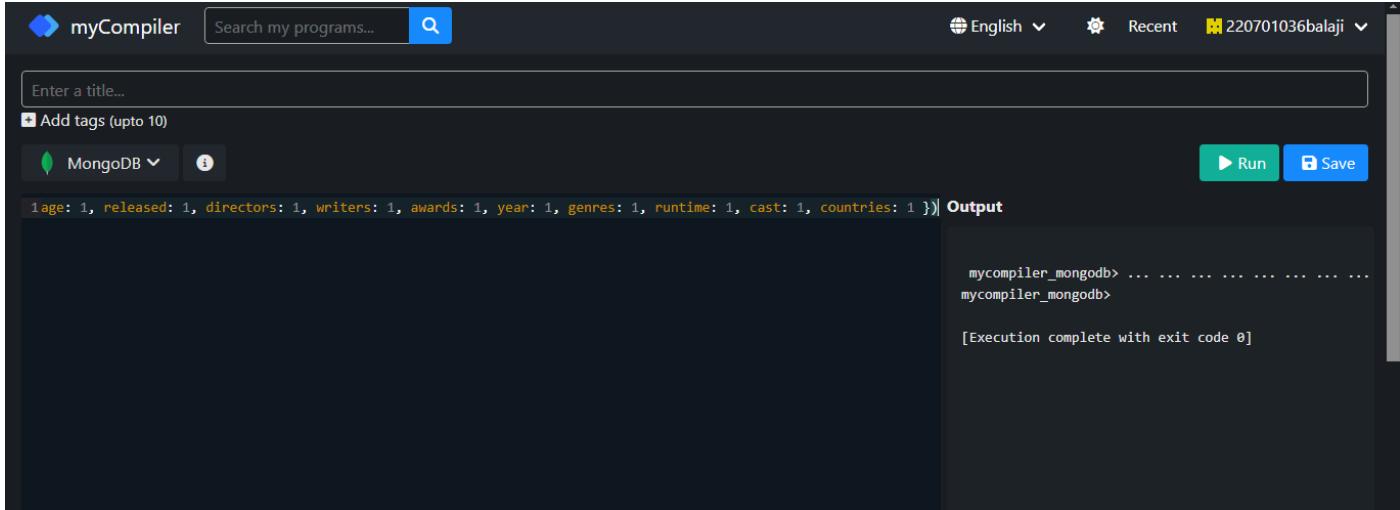
DATE:

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

QUERY:

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

OUTPUT:



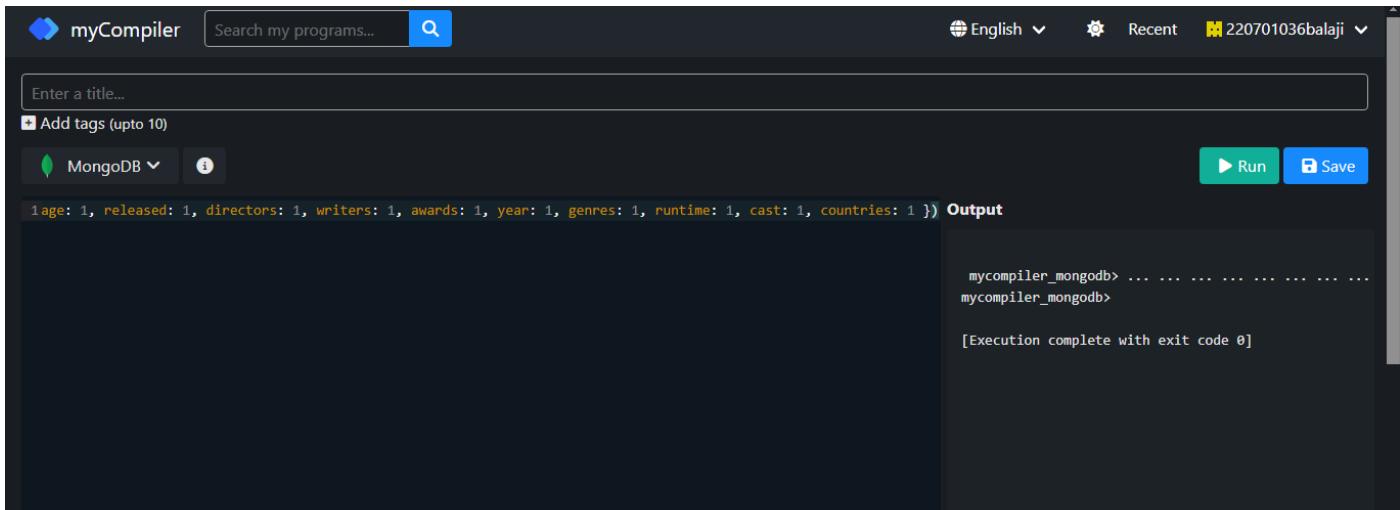
The screenshot shows the myCompiler interface with a dark theme. At the top, there's a search bar labeled "Search my programs..." and a "Run" button. Below the search bar, there's a text input field with placeholder text "Enter a title..." and a "Add tags (upto 10)" checkbox. On the left, there's a dropdown menu set to "MongoDB" and a help icon. On the right, there 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:



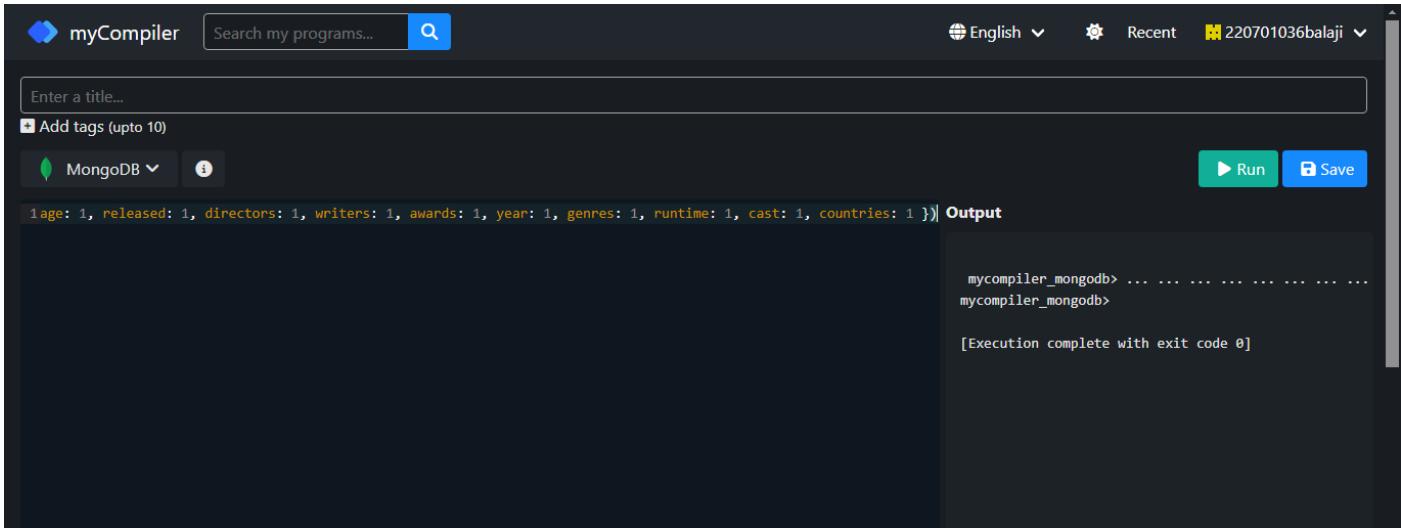
The screenshot shows the myCompiler interface with a dark theme. At the top, there's a search bar labeled "Search my programs..." and a "Run" button. Below the search bar, there's a text input field with placeholder text "Enter a title..." and a "Add tags (upto 10)" checkbox. On the left, there's a dropdown menu set to "MongoDB" and a help icon. On the right, there 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]

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

QUERY:

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

OUTPUT:



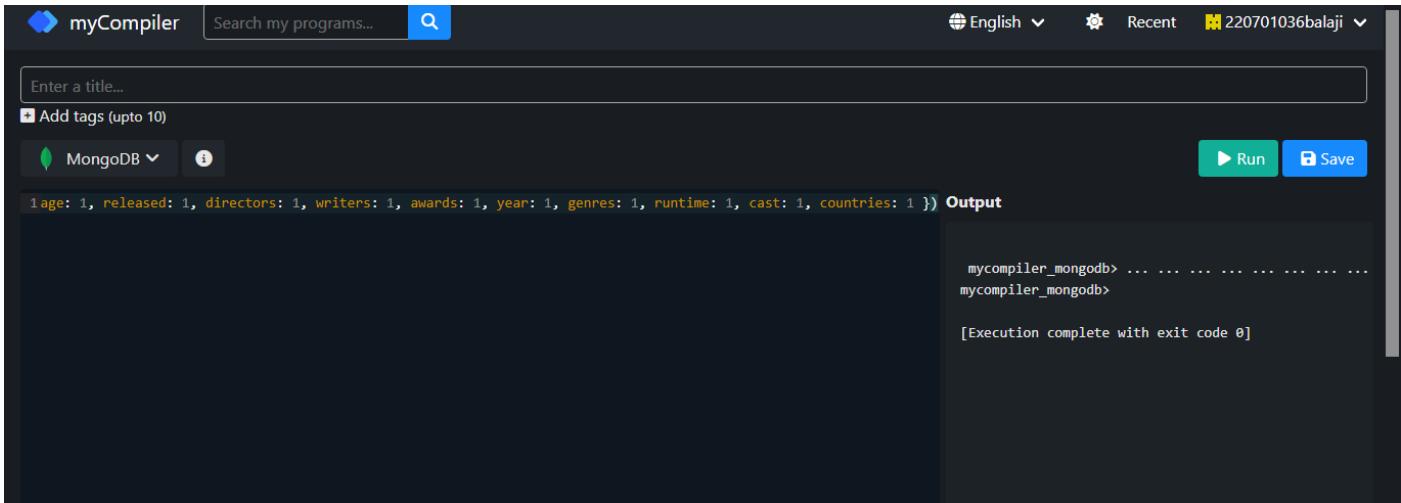
The screenshot shows the myCompiler interface. At the top, there's a search bar labeled "Search my programs..." and a "Run" button. Below the search bar, there's a text input field for "Enter a title..." and a "Add tags (upto 10)" button. On the left, there's a "MongoDB" dropdown and a help icon. On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

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 labeled "Search my programs..." and a "Run" button. Below the search bar, there's a text input field for "Enter a title..." and a "Add tags (upto 10)" button. On the left, there's a "MongoDB" dropdown and a help icon. On the right, there's a "Run" button and a "Save" button. The main area is titled "Output" and contains the following text:
1 age: 1, released: 1, directors: 1, writers: 1, awards: 1, year: 1, genres: 1, runtime: 1, cast: 1, countries: 1 }
mycompiler_mongodb> ...
mycompiler_mongodb>
[Execution complete with exit code 0]

5.) Retrieve all movies from the 'movies' collection that were released in the USA and include complete information for each movie.

QUERY:

```
db.movies.find({ countries: 'USA' })
```

OUTPUT:

The screenshot shows the myCompiler interface. At the top, there's a search bar with placeholder text "Search my programs..." and a magnifying glass icon. To the right are language selection ("English"), recent projects, and user information ("220701036balaji"). Below the header is a title input field ("Enter a title...") and a "Add tags (upto 10)" button. On the left, there's a "MongoDB" dropdown with a green leaf icon and a "Run" button with a play icon. The main area contains a code editor with the following MongoDB query:

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

Below the code editor is an "Output" section showing the command prompt:

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

At the bottom right of the output section 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:

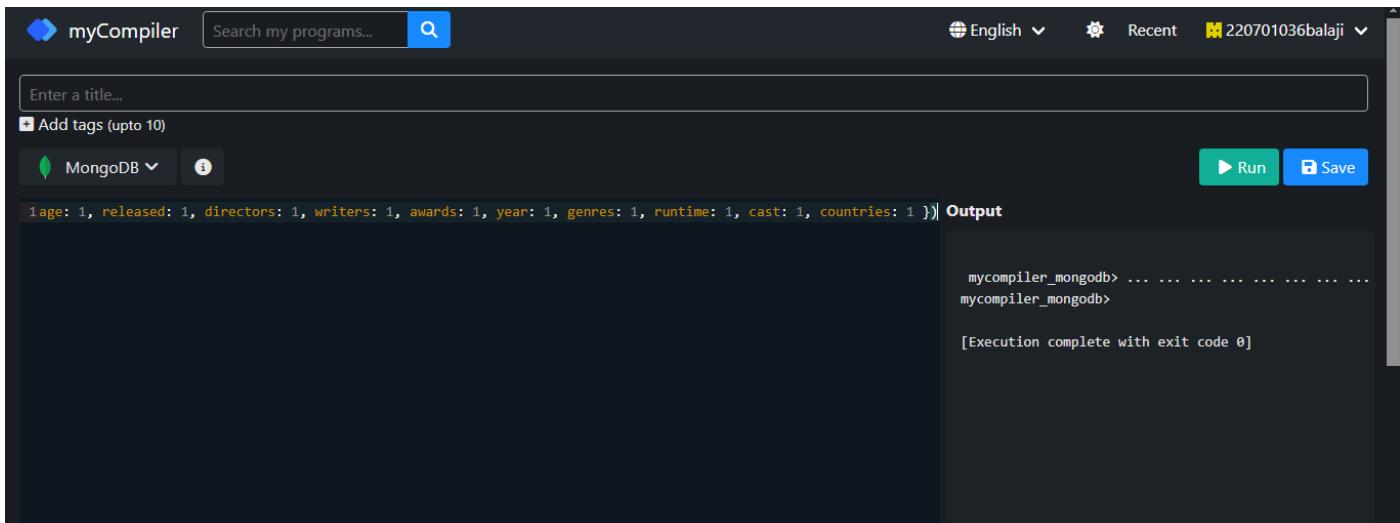
The screenshot shows the myCompiler interface. At the top, there's a search bar with the placeholder "Search my programs..." and a magnifying glass icon. To the right are language selection ("English"), recent projects, and user information ("220701036balaji"). Below the header is a text input field with placeholder "Enter a title...". Underneath it is a "Add tags (upto 10)" button. The main workspace has a "MongoDB" dropdown and a "Run" button with a play icon. A sidebar on the left contains a green circular icon and a "Save" button with a disk icon. The central area displays a MongoDB query: `1 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 query is a "Output" section showing the command prompt "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:



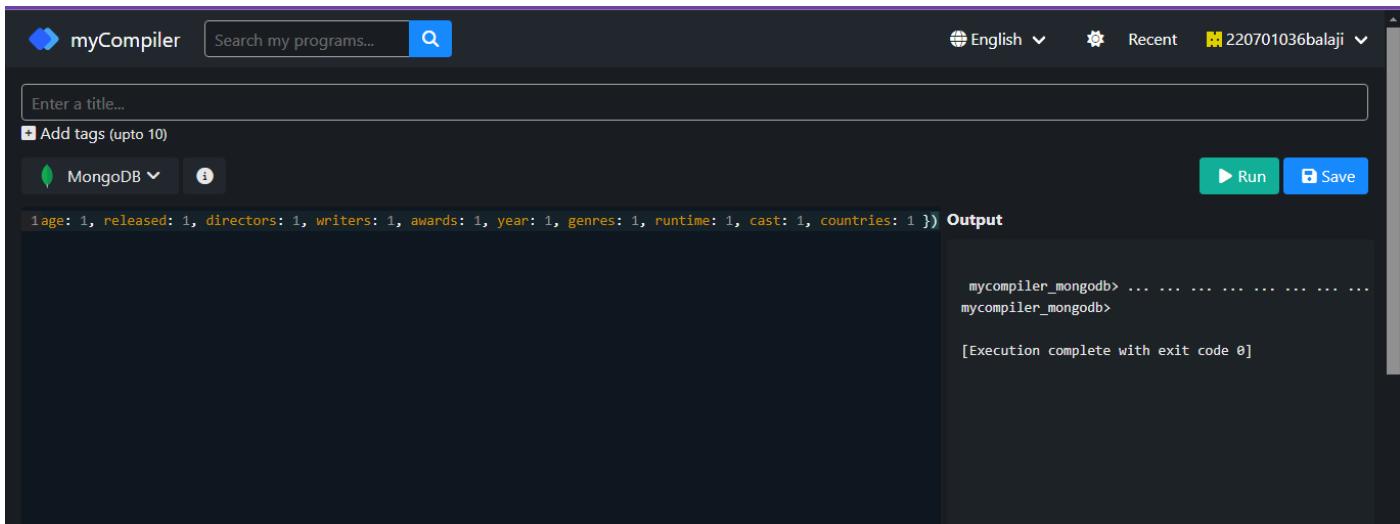
The screenshot shows the myCompiler interface. In the top bar, there is a search bar labeled "Search my programs..." with a magnifying glass icon, and a dropdown menu set to "English". On the right, it shows the user "220701036balaji". Below the search bar, there is a text input field with placeholder text "Enter a title...". Underneath it, there is a button labeled "+ Add tags (upto 10)". To the left of the main area, there is a "MongoDB" dropdown menu and a small info icon. On the right, there are two buttons: "Run" and "Save". The main area is titled "Output" and contains the following text:
`1age: 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:



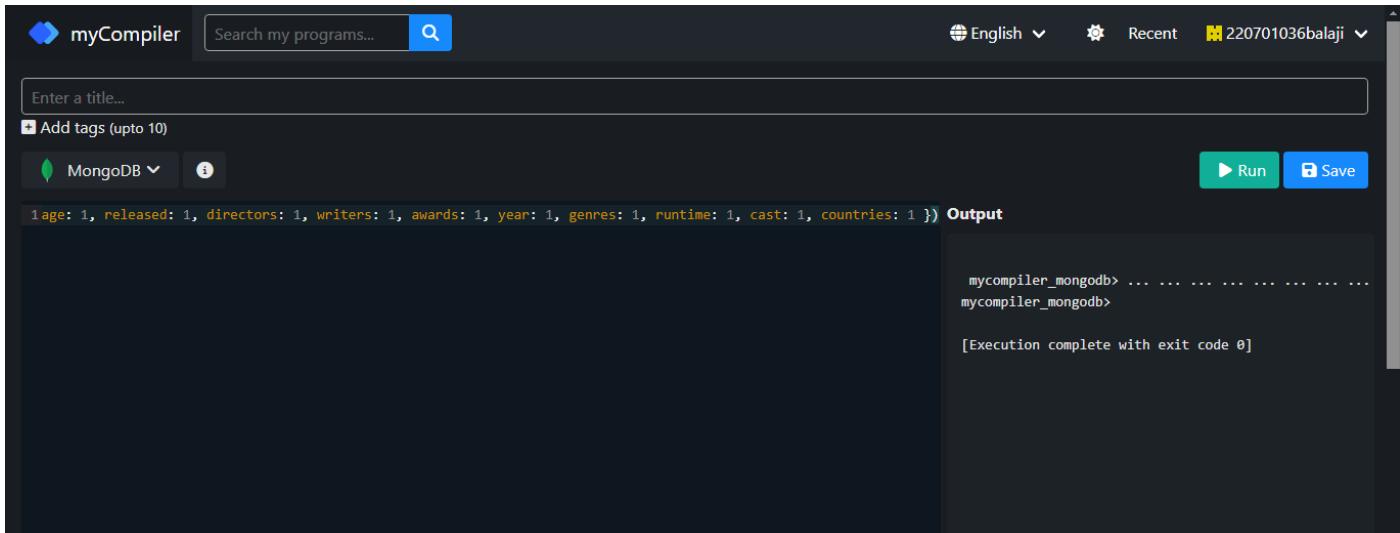
The screenshot shows the myCompiler interface. It has the same layout as the previous one, with a search bar, language dropdown, and user info. The "MongoDB" dropdown is selected. The main output area shows the following text:
`1age: 1, released: 1, directors: 1, writers: 1, awards: 1, year: 1, genres: 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 selector set to English. Below the search bar, there's a text input field labeled "Enter a title..." and a checkbox for "Add tags (upto 10)". On the left, there's a MongoDB connection dropdown and a help icon. On the right, there are "Run" and "Save" buttons. The main area contains a command line interface with the following text:

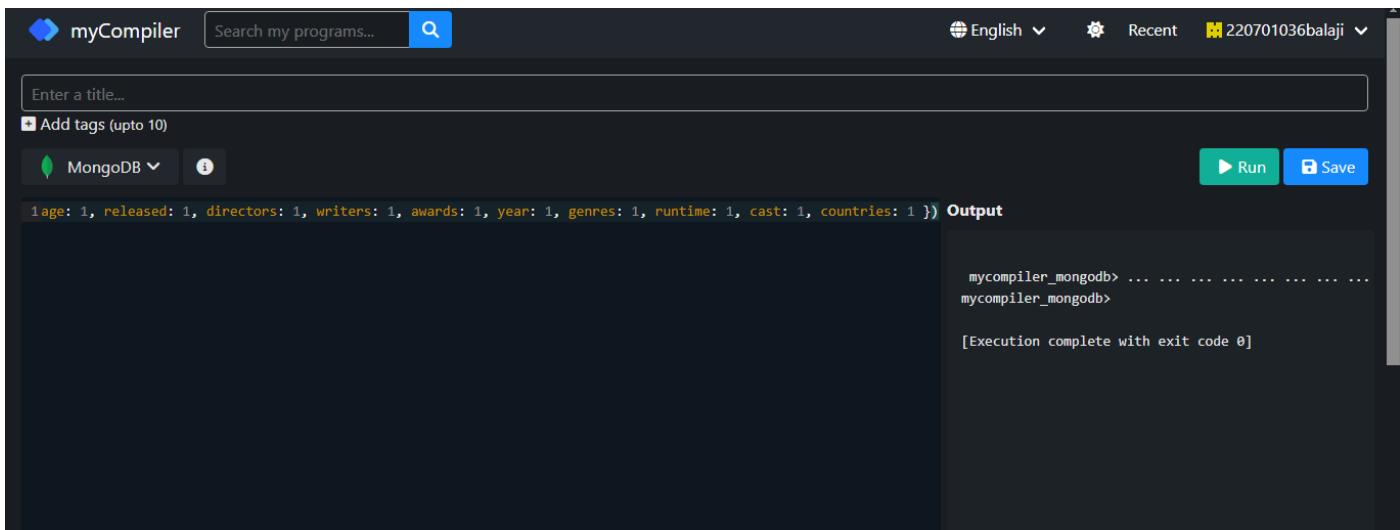
```
1 age: 1, released: 1, directors: 1, writers: 1, awards: 1, year: 1, genres: 1, runtime: 1, cast: 1, countries: 1 }} Output  
mycompiler_mongodb> ... . . . . .  
mycompiler_mongodb>  
[Execution complete with exit code 0]
```

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

QUERY:

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

OUTPUT:



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

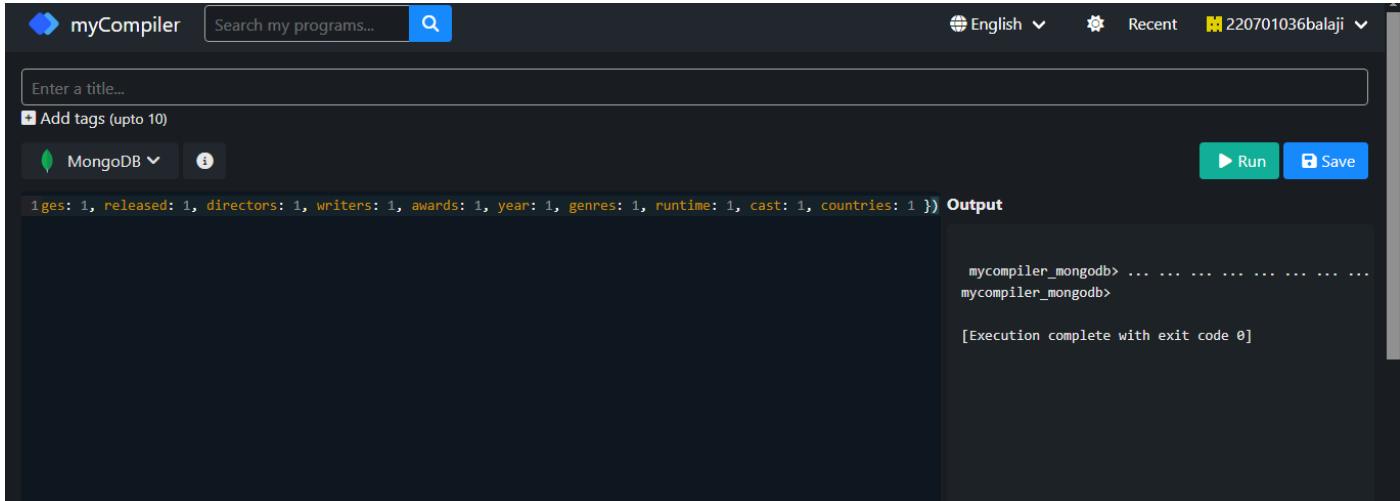
```
1 age: 1, released: 1, directors: 1, writers: 1, awards: 1, year: 1, genres: 1, runtime: 1, cast: 1, countries: 1 }} Output  
mycompiler_mongodb> ... . . . . .  
mycompiler_mongodb>  
[Execution complete with exit code 0]
```

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

QUERY:

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

OUTPUT:



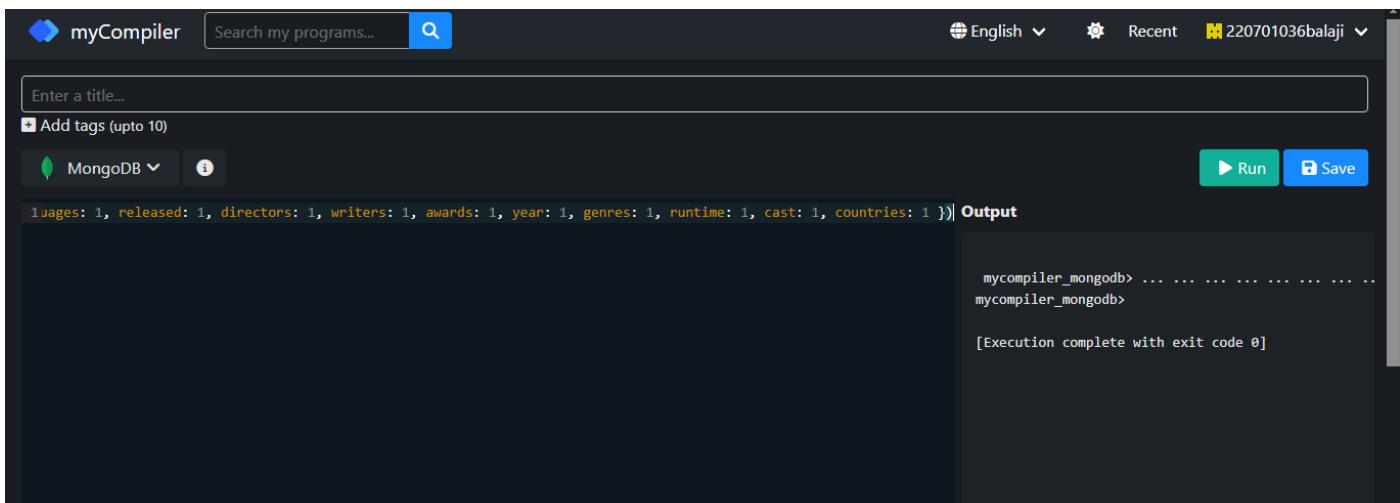
The screenshot shows the myCompiler interface. In the top bar, there are tabs for 'myCompiler', 'Search my programs...', and a magnifying glass icon. On the right, there are language selection dropdowns ('English'), a recent projects dropdown, and a user ID ('220701036balaji'). Below the search bar is a text input field labeled 'Enter a title...' and a checkbox for 'Add tags (upto 10)'. Underneath these are two buttons: 'MongoDB' with a dropdown arrow and an info icon, and 'Run' with a save icon. The main area is titled 'Output' and contains the MongoDB 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 })'. To the right of the output area, the terminal window shows the command 'mycompiler_mongodb> ...' followed by '[Execution complete with exit code 0]'. A vertical scroll bar is visible on the right side of the output panel.

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:



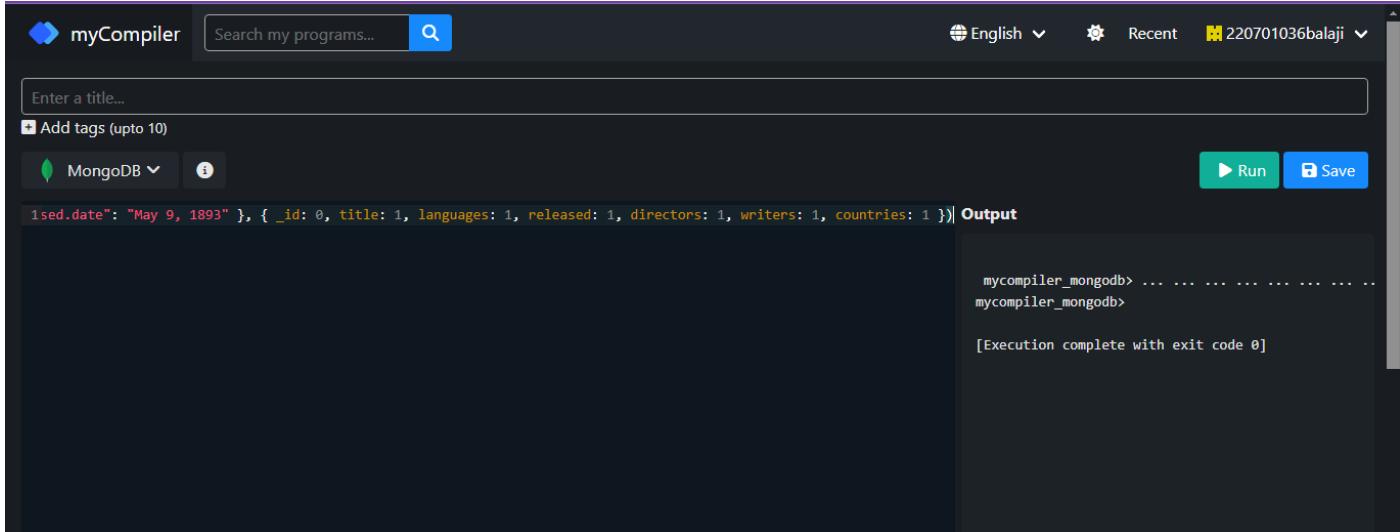
The screenshot shows the myCompiler interface. The layout is identical to the previous one, with tabs for 'myCompiler', 'Search my programs...', and a magnifying glass icon. The top bar also includes language selection ('English') and recent projects ('Recent'). The user ID '220701036balaji' is visible. The 'MongoDB' button has a dropdown arrow and an info icon. The 'Run' and 'Save' buttons are present. The 'Output' panel contains the MongoDB 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 })'. The terminal window on the right shows 'mycompiler_mongodb> ...' and '[Execution complete with exit code 0]'. A vertical scroll bar is visible on the right side of the output panel.

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 is a search bar labeled "Search my programs..." and a magnifying glass icon. On the right, there are language selection ("English"), recent projects ("Recent"), and user information ("220701036balaji"). Below the search bar, there is a text input field with placeholder "Enter a title..." and a "Add tags (upto 10)" button. A dropdown menu shows "MongoDB" selected. 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:

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

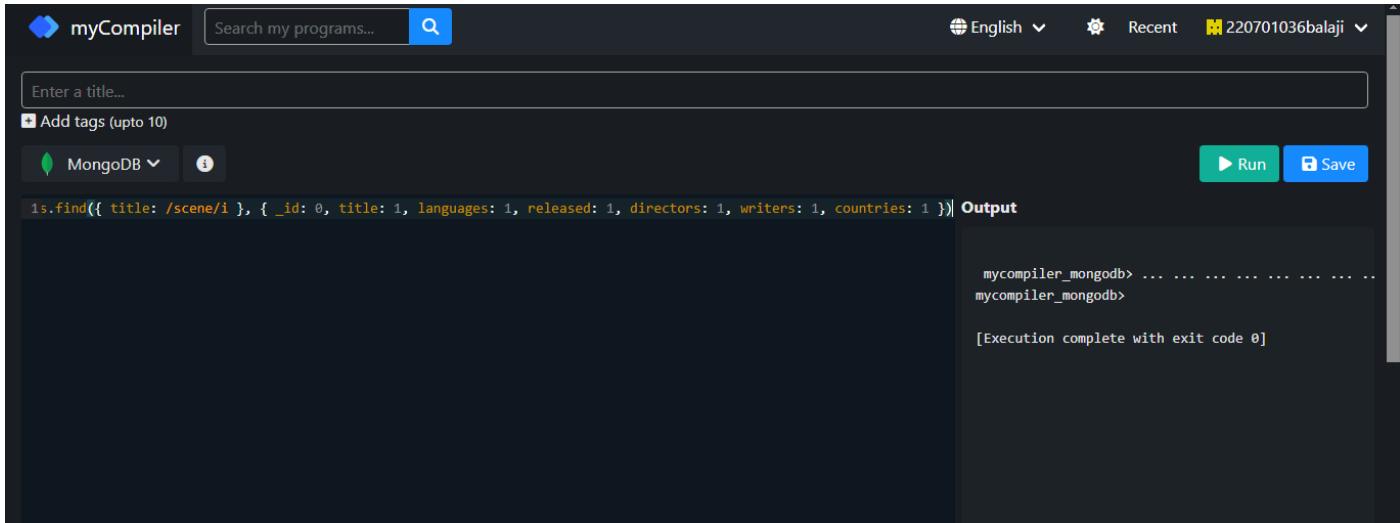
Below the code editor, the output window shows the command prompt "mycompiler_mongodb> ...". It then displays the results of the query, which are empty. At the bottom of the output window, it says "[Execution complete with exit code 0]".

14.) Retrieve all movies with title, languages, released, directors, writers, countries from the 'movies' collection in MongoDB that have a word "scene" in the title.

QUERY:

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

OUTPUT:



The screenshot shows the myCompiler interface. The layout is identical to the previous screenshot, with a search bar, language selection, and user information at the top. The code editor contains the following MongoDB query:

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

The output window shows the command prompt "mycompiler_mongodb> ...". It then displays the results of the query, which are empty. At the bottom of the output window, it says "[Execution complete with exit code 0]".

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

RESULT: