

Creating and Managing Tables

EX_NO:1

DATE:

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

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. In the SQL Commands section, the command `create table dept(id number(7),name varchar2(25));` is entered. The Results tab is selected, displaying the output "Table created." and a execution time of "0.03 seconds". The bottom of the screen shows standard Oracle APEX footer information.

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 APEX SQL Workshop interface. In the top navigation bar, 'APEX' is selected, followed by 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery'. A search bar is present, along with user profile icons for 'abilash m' and 'abilash06'. The main area is titled 'SQL Commands' with a 'Schema' dropdown set to 'WKSP_ABILASH06'. Below the title, there are buttons for 'Language' (set to 'SQL'), 'Rows' (set to 10), 'Clear Command', 'Find Tables', 'Save', and 'Run'. The command input field contains the SQL code: 'create table emp(id number(7),Last_name varchar2(25),first_name varchar2(25),Dept_id number(7));'. The results tab is active, showing the output: 'Table created.' and a execution time of '0.03 seconds'. The bottom footer includes user information ('220701006@rajalakshmi.edu.in', 'abilash06', 'en') and copyright information ('Copyright © 1999, 2023, Oracle and/or its affiliates.', 'Oracle APEX 23.2.4').

220701006@rajalakshmi.edu.in

abilash06

en

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

Oracle APEX 23.2.4

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. A search bar and user profile are also present. The main area is titled "SQL Commands" and contains a text input field with the query: "1 alter table emp modify(Last_name varchar2(25));". Below the input field are buttons for Language (SQL), Rows (set to 10), Clear Command, Find Tables, Save, and Run. The "Run" button is highlighted in green. The results section at the bottom shows the output: "Table altered." and "0.06 seconds". The bottom footer includes user information (220701006@rajalakshmi.edu.in, abilash06, en), copyright notice (Copyright © 1999, 2023, Oracle and/or its affiliates.), and the version (Oracle APEX 23.2.4).

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

Table altered.
0.06 seconds

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. In the top navigation bar, 'APEX' is selected, along with 'SQL Workshop'. The schema is set to 'WKSP_ABILASH06'. The main area is titled 'SQL Commands' and contains the following SQL code:

```
1 create table employees2(id number(7),first_Name varchar2(25),Last_name varchar2(25),salary int,Dept_id number(7));
```

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

Table created.
0.04 seconds

At the bottom of the page, there are footer links: 220701006@rajalakshmi.edu.in, abilash06, en, Copyright © 1999, 2023, Oracle and/or its affiliates., and Oracle APEX 23.2.

5. Drop the EMP table.

QUERY:

Drop table emp;

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for App Builder, SQL Workshop (selected), Team Development, and Gallery. A search bar and user profile are also present. The main area is titled "SQL Commands". The language is set to SQL, and the schema is WKSP_ABILASH06. A command is being entered in the text area: "drop table emp;". Below the text area, the results tab is selected, showing the output: "Table dropped." and a execution time of "0.07 seconds". The bottom footer displays copyright information for Oracle and the APEX version.

```
drop table emp;
```

Results Explain Describe Saved SQL History

Table dropped.
0.07 seconds

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

6. Rename the EMPLOYEES2 table as EMP.

QUERY:

Rename employees2 to emp;

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. A search bar and user profile are also present. The main workspace is titled "SQL Commands" and contains the following content:

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

Schema: WKSP_ABILASH06

```
1  rename employees2 to emp;
```

The results section shows the output of the executed command:

Statement processed.
0.05 seconds

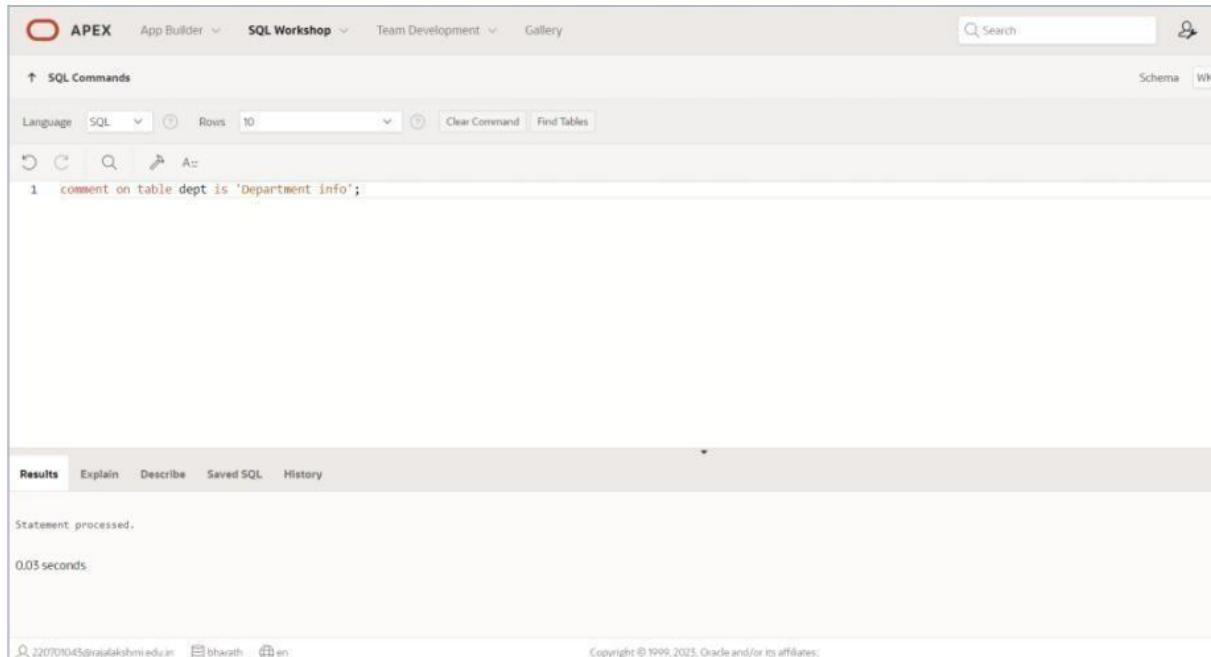
At the bottom, there are footer links for 220701006@rajalakshmi.edu.in, abilash06, and en, along with copyright information: Copyright © 1999, 2023, Oracle and/or its affiliates. and Oracle APEX 23.2.4.

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 SQL Workshop interface. In the top navigation bar, 'APEX' is selected. Below it, the 'SQL Workshop' tab is active. The main area is titled 'SQL Commands'. A command is entered in the text input field: 'comment on table dept is 'Department info'';'. The 'Results' tab is selected in the bottom navigation bar. The results pane displays the message 'Statement processed.' and '0.05 seconds'.

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

Statement processed.

0.05 seconds
```

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle 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 the user's profile (abilash m, abilash06). The main area is titled "SQL Commands" with a "Schema" dropdown set to "WKSP_ABILASH06". The query editor contains the command: "1 alter table emp drop column first_name;". Below the editor, the results tab is selected, showing the output: "Table altered." and "0.09 seconds". The bottom footer includes copyright information: "Copyright © 1999, 2023, Oracle and/or its affiliates." and "Oracle APEX 23.2.4".

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

Table altered.
0.09 seconds

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

RESULT:

MANIPULATING DATA

EX_NO:2

DATE:

1. Create MY_EMPLOYEE table with the following structure

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. The user is logged in as 'abilash m' with session ID 'abilash06'. The SQL Commands tab is selected, showing the schema 'WKSP_ABILASH06'. The SQL editor contains the following command:

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

The results tab shows the output of the query:

```
Table created.
```

Execution time: 0.04 seconds.

Page footer information includes:

- User: 220701006@rajalakshmi.edu.in
- Session: abilash06
- Language: en
- Copyright: Copyright © 1999, 2023, Oracle and/or its affiliates.
- Version: Oracle APEX 23.2.4

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

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

QUERY:

```
insert into myemployees(1,'patel','ralph','rpatel',895);
insert into myemployees(2,'dancs','betty','bdancs',60);
```

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 'abilash m' (abilash06). The main workspace is titled 'SQL Commands' and shows the schema 'WKSP_ABILASH06'. The SQL editor contains the following command:

```
1 select * from myemployees;
```

The results section displays the data from the 'myemployees' table:

ID	LAST_NAME	FIRST_NAME	USERID	SALARY
2	dancs	betty	bdancs	60
1	patel	ralph	rpatel	95

Below the results, it says '2 rows returned in 0.04 seconds' and provides a 'Download' link. The bottom of the page includes footer information: '220701006@rajalakshmi.edu.in', 'abilash06', 'en', 'Copyright © 1999, 2023, Oracle and/or its affiliates.', and 'Oracle APEX 23.2.4'.

3. Display the table with values.

QUERY:

Select * from myemployees;

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 the user's profile: abilash m (username abilash06). The main area is titled "SQL Commands" with a "Schema" dropdown set to WKSP_ABILASH06. The SQL editor contains the command: "select * from myemployees;". Below the editor, the "Results" tab is selected, displaying a table with two rows of data. The columns are ID, LAST_NAME, FIRST_NAME, USERID, and SALARY. The data is as follows:

ID	LAST_NAME	FIRST_NAME	USERID	SALARY
2	dancs	betty	bdancs	60
1	patel	ralph	rpatel	95

Below the table, it says "2 rows returned in 0.04 seconds" and there is a "Download" link. At the bottom, it shows the user's email (220701006@rajalakshmi.edu.in), the schema (abilash06), and the session language (en). Copyright information for Oracle is at the bottom center, and the version "Oracle APEX 23.2.4" is on the right.

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 myemployees(id,Last_name,first_name,Userid,Salary)
values
(4,'newman','chad','cnewman',50);
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. A search bar and user profile 'abilash m abilash06' are also present. The main workspace is titled 'SQL Commands' and contains the following SQL code:

```
1 insert into myemployees(id,Last_name,first_name,Userid,Salary)
2 values
3 (4,'newman','chad','cnewman',50);
```

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

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

Execution time is listed as 0.00 seconds. The bottom of the page displays copyright information and the version Oracle APEX 23.2.4.

5. Make the data additions permanent.

QUERY:

Select * from myemployees;

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 user is logged in as 'abilash m' (abilash06). The schema selected is 'WKSP_ABILASH06'. The SQL Commands section contains the query: 'select * from myemployees;'. The Results tab is selected, displaying the following data:

ID	LAST_NAME	FIRST_NAME	USERID	SALARY
2	dancs	betty	bdancs	60
1	patel	ralph	rpatel	95
3	biri	ben	bbiri	100
4	newman	chad	cnewman	50

At the bottom, the footer includes the URL '220701006@rajalakshmi.edu.in', the schema 'abilash06', and the session ID 'en'. It also states 'Copyright © 1999, 2023, Oracle and/or its affiliates.' and 'Oracle APEX 23.2.4'.

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

QUERY:

```
update myemployees set Last_name='drexler' where id=1;
```

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 the user's profile: abilash m (username) and abilash06 (schema). The main area is titled "SQL Commands" and contains a command input field with the following content:

```
1 update myemployees set Last_name='drexler' where id=1;
```

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

1 row(s) updated.
0.02 seconds

At the bottom of the page, there are footer links for 220701006@rajalakshmi.edu.in, abilash06, en, Copyright © 1999, 2023, Oracle and/or its affiliates, and Oracle APEX 23.2.4.

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

QUERY:

update myemployees set salary=1000 where salary<900;

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. A search bar and user profile are also present. The main area is titled "SQL Commands". The language is set to SQL, rows to 10, and the schema is WKSP_ABILASH06. The command entered is "update myemployees set salary=1000 where salary<900;". Below the command, the results tab is selected, showing the output: "1 row(s) updated." and "0.01 seconds". The bottom footer displays copyright information and the version Oracle APEX 23.2.4.

```
1 update myemployees set salary=1000 where salary<900;
```

Results Explain Describe Saved SQL History

1 row(s) updated.
0.01 seconds

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

8. Delete Betty dancs from MY_EMPLOYEE table.

QUERY:

```
delete from myemployees where Last_name='dancs';
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. A search bar and user profile are also present. The main area is titled "SQL Commands" and contains a code editor with the following content:

```
1 delete from myemployees where Last_name='dancs';
```

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 query:

1 row(s) deleted.
0.00 seconds

At the bottom of the page, footer information includes the user's email (220701006@rajalakshmi.edu.in), session ID (abilash06), and language (en). Copyright information for Oracle is also shown, along with the APEX version (23.2.4).

9. Empty the fourth row of the emp table.

QUERY:

Delete from emp where id=4;

OUTPUT:

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

```
1 delete from emp where id=4;
```

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

0 row(s) deleted.
0.03 seconds

At the bottom of the page, there are footer links for user information (220701006@rajalakshmi.edu.in, abilash06, 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

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 MY_EMPLOYEE add constraint my_emp_id_pk primary key(id);
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. In the top navigation bar, 'APEX' is selected. The main area is titled 'SQL Commands'. A single line of SQL code is entered in the command input field:

```
1 alter table MY_EMPLOYEE add constraint my_emp_id_pk primary key(id);
2
```

The 'Run' button is visible in the top right corner of the command input area. Below the input area, there is a results pane. The 'Results' tab is selected. The output shows:

```
Table altered.
0.07 seconds
```

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

QUERY:

```
Create table dept02(my_dept_id_pk int primary key);
```

OUTPUT:

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

```
1 create table dept(dept_id number(6),dept_name varchar2(16),
2 constraints my_dept_id_pk primary key(dept_id));
3
```

Below the code editor, the 'Results' tab is active. The output pane displays the results of the command execution:

```
Table created.

0.05 seconds
```

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

QUERY:

```
Alter table employee add constraint my_emp_dept_id_fk foreign key(dept_id)references employee(id);
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. In the top navigation bar, 'APEX' is selected. Below it, tabs for 'App Builder', 'SQL Workshop' (which is active), 'Team Development', and 'Gallery' are visible. On the right side of the header, the user 'aadhiytha rk' and session ID 'aadhiytha02' are displayed. The main workspace is titled 'SQL Commands'. It contains a toolbar with icons for Undo, Redo, Search, Find Tables, Clear Command, and Run. Below the toolbar, a dropdown menu shows 'Language: SQL' and 'Rows: 10'. The SQL editor area contains the following command:

```
1 alter table employee add constraint my_emp_dept1_id_fk foreign key(dept1_id)references employee(id);
```

After running the command, the results pane shows the output:

```
Table altered.  
0.05 seconds
```

The bottom status bar indicates the session details: '220701002@rajalakshmi.edu.in', 'aadhiytha02', 'en', and the system information: 'Copyright © 1999, 2023, Oracle and/or its affiliates.', 'Oracle APEX 25.2.0', '11:33 AM', '01-03-2024'.

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 MY_EMPLOYEE add(commission number(2,2) constraint ck_1 check(commission>0);

OUTPUT:

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

```
1 alter table MY_EMPLOYEE add(commission number(2,2) constraint ck_1 check(commission>0));
2
```

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

```
Table altered.

0.05 seconds
```

At the bottom of the results pane, it says 'Copyright © 1999, 2023, Oracle and/or its affiliates.' and 'Oracle APEX 25.2.4'.

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

RESULT:

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 employee;
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (which is selected), Team Development, and Gallery. The right side of the header shows the user's profile (abilash m) and workspace (abilash06). The main workspace is titled 'SQL Commands'. It features a search bar, schema dropdown (WKSP_ABILASH06), and buttons for Save and Run. Below the title, there are filters for Language (SQL), Rows (10), and buttons for Clear Command and Find Tables. The SQL command entered is: `1 Select EMPLOYEE_ID, LAST_NAME, "SALARY"*12"ANNUAL_SALARY" from employee;`. The results section shows a single row of data:

EMPLOYEE_ID	LAST_NAME	ANNUAL_SALARY
1	prasad	168000

Below the results, it says '1 rows returned in 0.01 seconds' and has a 'Download' link. At the bottom, it shows the user's email (220701006@rajalakshmi.edu.in), workspace (abilash06), and language (en). Copyright information for Oracle is at the bottom center, and the version 'Oracle APEX 23.2.4' is at the bottom right.

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

QUERY:

Select * from employee;

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. A search bar and user profile 'abilash m abilash06' are also present. The main area is titled 'SQL Commands' with a schema dropdown set to 'WKSP_ABILASH06'. The SQL editor contains the command 'select * from employee;'. Below the editor, the results tab is selected, displaying a table with columns: EMPLOYEE_ID, FIRST_NAME, LAST_NAME, EMAIL, PHONE_NUMBER, HIRE_DATE, JOB_ID, SALARY, MANAGER_ID, DEPARTMENT_ID, and ANNUAL_SALARY. One row is shown: 1, rajesh, prasad, raj@gmail.com, 989321875, 0, 35, 14000, 13, 1, 300000. A message at the bottom indicates '1 rows returned in 0.02 seconds' and provides a download link. The footer includes copyright information for Oracle and the APEX version 'Oracle APEX 23.2.4'.

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	MANAGER_ID	DEPARTMENT_ID	ANNUAL_SALARY
1	rajesh	prasad	raj@gmail.com	989321875	0	35	14000	13	1	300000

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 employee;

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. A search bar and user profile are also present. The main area is titled "SQL Commands" with a "Schema" dropdown set to "WKSP_ABILASH06". The SQL editor contains the following query:

```
1 Select EMPLOYEE_ID as EMPLOYEE_NUMBER, LAST_NAME, JOB_ID, HIRE_DATE from employee;
```

The results section displays the output of the query:

EMPLOYEE_NUMBER	LAST_NAME	JOB_ID	HIRE_DATE
1	prasad	35	0

Below the results, it says "1 rows returned in 0.02 seconds" and provides a "Download" link. The bottom of the page includes standard footer links for Oracle APEX and copyright information.

4. Provide an alias STARTDATE for the hire date.

QUERY:

Select hire_date as startdate from employee;

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. A search bar and user profile are also present. The main workspace is titled "SQL Commands" and contains a single line of SQL code: "Select HIRE_DATE as START_DATE from employee;". The "Run" button is highlighted in green. Below the workspace, the results tab is selected, showing a single row with the column header "START_DATE" and the value "0". The status bar at the bottom indicates "1 rows returned in 0.00 seconds".

START_DATE
0

Results Explain Describe Saved SQL History

0
1 rows returned in 0.00 seconds Download

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

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

QUERY:

Select distinct job_id from empployee;

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. A search bar and user profile are also present. The main area is titled "SQL Commands" and contains a language selector (SQL), row limit (10), and buttons for Clear Command and Find Tables. The query "Select distinct JOB_ID from empployee;" is entered in the command field. Below the command is a results grid with one row containing the value "35". The bottom of the screen displays copyright information and the version "Oracle APEX 23.2.4".

JOB_ID
35

1 rows returned in 0.01 seconds Download

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. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. A search bar and user profile are also present. The main area is titled 'SQL Commands' and shows the schema 'WKSP_ABILASH06'. The SQL editor contains the query:

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

The results tab is selected, displaying the output:

EMPLOYEE_AND_TITLE
prasad,35

Below the results, it says '1 rows returned in 0.01 seconds' and provides a download link. The bottom footer includes user information (220701006@rajalakshmi.edu.in, abilash06, en), copyright notice (Copyright © 1999, 2023, Oracle and/or its affiliates.), and version information (Oracle APEX 23.2.4).

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

QUERY:

```
select employee_id||','||first_name||','||last_name||','||email||','||phone_number||','||hire_date||',
'||job_id||'||salary||','||manager_id||','||department_id as "the_output" 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. A search bar and user profile are also present. The main workspace is titled 'SQL Commands' and contains the following SQL code:

```
1  ||EMAIL||','||PHONE_NUMBER||','||HIRE_DATE||','||JOB_ID||','||SALARY||','||MANAGER_ID||','||DEPARTMENT_ID as "the_output" FROM employees;
```

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

the_output
1,rajesh,prasad,raj@gmail.com,989321875,0,35,14000,13,1

Below the results, it says '1 rows returned in 0.01 seconds' and provides a 'Download' link. The bottom of the page shows user information (email, session ID) and copyright details.

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

RESULT:

RESTRICTING AND SORTING DATA

EX_NO:5

DATE:

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

QUERY:

```
select LAST_NAME,SALARY from MY_EMPLOYEES where SALARY>12000;
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The query `select LAST_NAME, SALARY from MY_EMPLOYEES where SALARY > 12000;` has been run, and the results are displayed in a table. The results show two rows: one for 'kumar' with a salary of 15000 and another for 'raj' with a salary of 20000. The table has columns 'LAST_NAME' and 'SALARY'.

LAST_NAME	SALARY
kumar	15000
raj	20000

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

QUERY:

```
select LAST_NAME,DEPARTMENT_ID from MY_EMPLOYEES where EMPLOYEE_ID=176
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The query `select LAST_NAME, DEPARTMENT_ID from MY_EMPLOYEES where EMPLOYEE_ID = 176;` has been run, and the results are displayed in a table. The results show one row for 'kumar' with a department ID of 54. The table has columns 'LAST_NAME' and 'DEPARTMENT_ID'.

LAST_NAME	DEPARTMENT_ID
kumar	54

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 MY_EMPLOYEEESS where SALARY not between 5000 and 12000;

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. The SQL Workshop tab is selected. The schema dropdown is set to WKSP_ABILASH06. The main area contains the SQL command:

```
1 select LAST_NAME, SALARY from MY_EMPLOYEEESS where SALARY not between 5000 and 12000;
```

The results section displays the output:

LAST_NAME	SALARY
kumar	15000
raj	20000

At the bottom, there are footer links for 220701006@rajalakshmi.edu.in, abilash06, en, Copyright © 1999, 2023, Oracle and/or its affiliates., and Oracle APEX 23.2.4.

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

QUERY:

select LAST_NAME,JOB_ID,HIRE_DATE from MY_EMPLOYEEESS where HIRE_DATE between '02/20/ 1998' and '05/01/1998' order by HIRE_DATE;

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. The SQL Workshop tab is selected. The schema dropdown is set to WKSP_ABILASH06. The main area contains the SQL command:

```
1 select LAST_NAME, JOB_ID, HIRE_DATE from MY_EMPLOYEEESS where HIRE_DATE between '02/20/1998' and '05/01/1998' order by HIRE_DATE;
```

The results section displays the output:

LAST_NAME	JOB_ID	HIRE_DATE
kumar	21	02/20/1998
modi	56	03/21/1998

At the bottom, there are footer links for 220701006@rajalakshmi.edu.in, abilash06, en, Copyright © 1999, 2023, Oracle and/or its affiliates., and Oracle APEX 23.2.4.

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

QUERY:

```
select LAST_NAME,DEPARTMENT_ID from MY_EMPLOYEESS where DEPARTMENT_ID in ('20','50') order by LAST_NAME;
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', 'Gallery', a search bar, and a user profile 'abilash m abilash06'. The main toolbar has buttons for 'SQL Commands', 'Language (SQL)', 'Rows (10)', 'Clear Command', 'Find Tables', 'Save', and 'Run'. The SQL command entered is: `select LAST_NAME,DEPARTMENT_ID from MY_EMPLOYEESS where DEPARTMENT_ID in ('20','50') order by LAST_NAME;`. The results tab is selected, displaying the output:

LAST_NAME	DEPARTMENT_ID
raj	50

1 rows returned in 0.00 seconds

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

QUERY:

```
select LAST_NAME as "MY_EMPLOYEESS",SALARY as "MONTHLY SALARY" from MY_EMPLOYEESS where (SALARY between 5000 and 12000) and (DEPARTMENT_ID in(20,50)) order by LAST_NAME asc;
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', 'Gallery', a search bar, and a user profile 'abilash m abilash06'. The main toolbar has buttons for 'SQL Commands', 'Language (SQL)', 'Rows (10)', 'Clear Command', 'Find Tables', 'Save', and 'Run'. The SQL command entered is: `select EMPLOYEE,SALARY as MONTHLYSALARY from MY_EMPLOYEESS where SALARY between 5000 and 12000 and DEPARTMENT_ID between 20 and 50 order by LAST_NAME;`. The results tab is selected, displaying the output:

EMPLOYEE	MONTHLYSALARY
mani	12000
modi	10000

2 rows returned in 0.01 seconds

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 MY_EMPLOYEES where HIRE_DATE like '%94';
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', 'Gallery', and a search bar. The right side shows a user profile for 'abilash m' (abilash06). The main area is titled 'SQL Commands' with tabs for 'Language' (set to 'SQL'), 'Rows' (set to 10), 'Clear Command', and 'Find Tables'. A toolbar below has icons for undo, redo, search, and run. The command input field contains: 1 select LAST_NAME,HIRE_DATE from MY_EMPLOYEES where HIRE_DATE like '%94';. The results section is titled 'Results' and displays a single row: mani | 02/06/1994. Below the results, it says '1 rows returned in 0.01 seconds'. The bottom footer includes copyright information for Oracle and the APEX version.

LAST_NAME	HIRE_DATE
mani	02/06/1994

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 MY_EMPLOYEES WHERE MANAGER_ID IS NULL;
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', 'Gallery', and a search bar. The right side shows a user profile for 'abilash m' (abilash06). The main area is titled 'SQL Commands' with tabs for 'Language' (set to 'SQL'), 'Rows' (set to 10), 'Clear Command', and 'Find Tables'. A toolbar below has icons for undo, redo, search, and run. The command input field contains: 1 select LAST_NAME,JOB_NAME from MY_EMPLOYEES where MANAGER_ID is null;. The results section is titled 'Results' and displays a single row: kumar | salesrep. Below the results, it says '1 rows returned in 0.01 seconds'. The bottom footer includes copyright information for Oracle and the APEX version.

LAST_NAME	JOB_NAME
kumar	salesrep

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

QUERY:

```
SELECT last_name, salary, commission_pct FROM EMP WHERE commission_pct IS NOT NULL ORDER BY salary DESC, commission_pct DESC;
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. A search bar and user profile are also present. The main area is titled "SQL Commands" with a "Schema" dropdown set to "WKSP_ABILASH06". The SQL editor contains the following query:

```
1 select LAST_NAME, SALARY, COMMISSION_AMT from MY_EMPLOYEESS where COMMISSION_AMT is not null order by SALARY, COMMISSION_AMT desc;
```

The results section displays the output of the query:

LAST_NAME	SALARY	COMMISSION_AMT
modi	10000	1200
mani	12000	3500
lunam	15000	1000

At the bottom of the interface, there are copyright notices for Oracle and APEX, along with the version information "Oracle APEX 23.2.4".

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

QUERY:

```
SELECT last_name FROM EMP WHERE last_name LIKE '_a%';
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. A search bar and user profile are also present. The main area is titled "SQL Commands" with a "Schema" dropdown set to "WKSP_ABILASH06". The SQL editor contains the following query:

```
1 select LAST_NAME from MY_EMPLOYEESS where LAST_NAME like '_a%';
```

The results section displays the output of the query:

LAST_NAME
mani
raj

At the bottom of the interface, there are copyright notices for Oracle and APEX, along with the version information "Oracle APEX 23.2.4".

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

QUERY:

```
SELECT last_name FROM EMP 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 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', 'Gallery', a search bar, and a user profile for 'abilash m abilash06'. The 'SQL Commands' tab is selected. The SQL editor contains the following query:

```
1 select LAST_NAME from MY_EMPLOYEESS where LAST_NAME like '%a%' and LAST_NAME like '%e%';
```

The results section shows a single row returned in 0.01 seconds:

LAST_NAME
abidhea

Below the results, it says '1 rows returned in 0.01 seconds' and provides download options. The bottom status bar shows the URL '220701006@rajalakshmi.edu.in', session 'abilash06', and locale 'en'. Copyright information and 'Oracle APEX 23.2.4' are also present.

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_name,salary from EMP where job_name in ('hr', 'manager') and salary not in (2500,3500,7000);
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', 'Gallery', a search bar, and a user profile for 'abilash m abilash06'. The 'SQL Commands' tab is selected. The SQL editor contains the following query:

```
1 employee",SALARY as "monthly salary" from MY_EMPLOYEESS where (SALARY between 5000 and 12000) and (DEPARTMENT_ID in(20,50)) order by LAST_NAME;
```

The results section shows a single row returned in 0.00 seconds:

employee	monthly salary
mani	6000

Below the results, it says '1 rows returned in 0.00 seconds' and provides download options. The bottom status bar shows the URL '220701006@rajalakshmi.edu.in', session 'abilash06', and locale 'en'. Copyright information and 'Oracle APEX 23.2.4' are also present.

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_AMT from MY_EMPLOYEEESS where  
COMMISSION_AMT=SALARY*0.20;
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. A search bar and user profile 'abilash m abilash06' are also present. The main workspace is titled 'SQL Commands' and contains the following SQL query:

```
1 select LAST_NAME,SALARY,COMMISSION_AMT from MY_EMPLOYEEESS where COMMISSION_AMT=SALARY*0.20;
```

Below the query, the results tab is selected, showing the message "no data found". The bottom footer displays copyright information for Oracle and the APEX version.

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

RESULT:

SINGLE ROW FUNCTIONS

EX_NO:6

DATE:

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

QUERY:

```
select sysdate from dual;
```

OUTPUT:

The screenshot shows the Oracle APEX interface with the SQL Workshop tab selected. The query `select sysdate from dual` is entered in the command editor. The results show a single row with the value `04/15/2024` under the column `SYSDATE`.

EMPLOYEE_ID	LAST_NAME	SALARY	NEW SALARY
176	kumar	15000	17325
35	jeyaraj	6000	6930
32	abidhea	10000	11550
33	yohan	20000	23100

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 employee;`

OUTPUT:

The screenshot shows the Oracle APEX interface with the SQL Workshop tab selected. The query `select employee_id, last_name, salary, salary+(15.5/100*salary) as "NEW SALARY" from employee;` is entered in the command editor. The results show the employee ID, last name, salary, and the calculated new salary for four employees.

EMPLOYEE_ID	LAST_NAME	SALARY	NEW SALARY
176	kumar	15000	17325
35	jeyaraj	6000	6930
32	abidhea	10000	11550
33	yohan	20000	23100

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 employee;
```

OUTPUT:

The screenshot shows the Oracle SQL Developer interface. The SQL tab contains the query: `select (New_Salary - salary)"Increase" from employee`. The Results tab displays the output:

Increase
930
2325
3410
1705
3255

Below the table, it says "5 rows returned in 0.02 seconds". The bottom right corner of the interface shows "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 employee 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 SQL Developer interface. The SQL tab contains the query: `select initcap(LAST_NAME)as "Name", length(LAST_NAME)as "Length of Name" from MY_EMPLOYEESS where LAST_NAME like 'J%' or LAST_NAME like 'a%' or LAST_NAME like 'M%' order by LAST_NAME asc;`. The Results tab displays the output:

Name	Length of Name
Abidhea	7

Below the table, it says "1 rows returned in 0.01 seconds". The bottom right corner of the interface shows "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 last_name from employee where last_name like 'H%' OUTPUT:

The screenshot shows the Oracle APEX SQL developer interface. The top bar includes 'Language: SQL', 'Rows: 10', 'Clear Command', 'Find Tables', 'Save', and 'Run' buttons. The SQL editor contains the command: '1 select last_name from employee where last_name like 'H%''. The results section shows two rows: 'Harsha' and 'Harris'. Below the results, it says '2 rows returned in 0.00 seconds' and has a 'Download' link. The bottom footer includes user information (22070103@rajalakshmi.edu.in, aaditya03, en), copyright notice (Copyright © 1999, 2023, Oracle and/or its affiliates.), and version information (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 employee order by round((sysdate-hire_date)/30,0) asc;

OUTPUT:

The screenshot shows the Oracle APEX SQL developer interface. The top bar includes 'Language: SQL', 'Rows: 10', 'Clear Command', 'Find Tables', 'Save', and 'Run' buttons. The SQL editor contains the command: '1 select last_name,round((sysdate-hire_date)/30,0) as "MONTHS_WORKED" from employee order by round((sysdate-hire_date)/30,0) asc'. The results section displays a table with columns 'LAST_NAME' and 'MONTHS_WORKED'. The data includes: Kohli (291), Mohan (316), Harris (316), Abhijeeth (318), and Harsha (364). Below the results, it says '5 rows returned in 0.01 seconds' and has a 'Download' link. The bottom footer includes user information (22070103@rajalakshmi.edu.in, aaditya03, en), copyright notice (Copyright © 1999, 2023, Oracle and/or its affiliates.), and version information (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 employee;
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. A search bar and user profile 'abilash m abilash06' are also present. The main area has tabs for 'SQL Commands' and 'Results'. The 'Results' tab is selected, showing the output of the query:

```
1 select LAST_NAME||' earns '$||SALARY||' monthly but wants '$||SALARY*3 as "Dream salary" from MY_EMPLOYEES;
```

The output table has a single column labeled 'Dream salary' containing the results:

Dream salary
kumar earns \$15000 monthly but wants \$45000
jeyaraj earns \$6000 monthly but wants \$18000
abidhea earns \$10000 monthly but wants \$30000
yohan earns \$20000 monthly but wants \$60000

Below the table, it says '4 rows returned in 0.03 seconds' and has a 'Download' link. The bottom of the page shows copyright information and the 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 employee;
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface, similar to the previous one but with a different query. The top navigation bar and user profile are the same. The main area has tabs for 'SQL Commands' and 'Results'. The 'Results' tab is selected, showing the output of the query:

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

The output table has two columns: 'LAST_NAME' and 'SALARY'. The 'SALARY' column uses the \$ format specified in the query:

LAST_NAME	SALARY
Abhijeet	\$\$\$\$\$\$\$\$\$\$6000
Mohan	\$\$\$\$\$\$\$\$\$\$15000
Harsha	\$\$\$\$\$\$\$\$\$\$22000
Harris	\$\$\$\$\$\$\$\$\$\$11000
Kohli	\$\$\$\$\$\$\$\$\$\$21000

Below the table, it says '5 rows returned in 0.01 seconds' and has a 'Download' link. The bottom of the page shows copyright information and the 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 employee;

OUTPUT:

The screenshot shows the Oracle APEX SQL developer interface. The top section contains the SQL command:

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

The bottom section shows the results table:

LAST_NAME	HIRE_DATE	REVIEW
Abhijeeth	02/02/1998	Monday, the 03 of August, 1998
Mohan	03/24/1998	Monday, the 28 of September, 1998
Harsha	05/03/1994	Monday, the 07 of November, 1994
Harris	04/05/1998	Monday, the 12 of October, 1998
Kohli	04/03/2000	Monday, the 09 of October, 2000

Details at the bottom: 5 rows returned in 0.01 seconds, Copyright © 1999, 2023, Oracle and/or its affiliates, 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 employee order by TO_CHAR(hire_date,'Day');

OUTPUT:

The screenshot shows the Oracle APEX SQL developer interface. The top section contains the SQL command:

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

The bottom section shows the results table:

LAST_NAME	HIRE_DATE	DAY
Abhijeeth	02/02/1998	Monday
Kohli	04/03/2000	Monday
Harris	04/05/1998	Sunday
Mohan	03/24/1998	Tuesday
Harsha	05/03/1994	Tuesday

Details at the bottom: 5 rows returned in 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:

DISPLAYING DATA FROM MULTIPLE TABLES

EX_NO:7

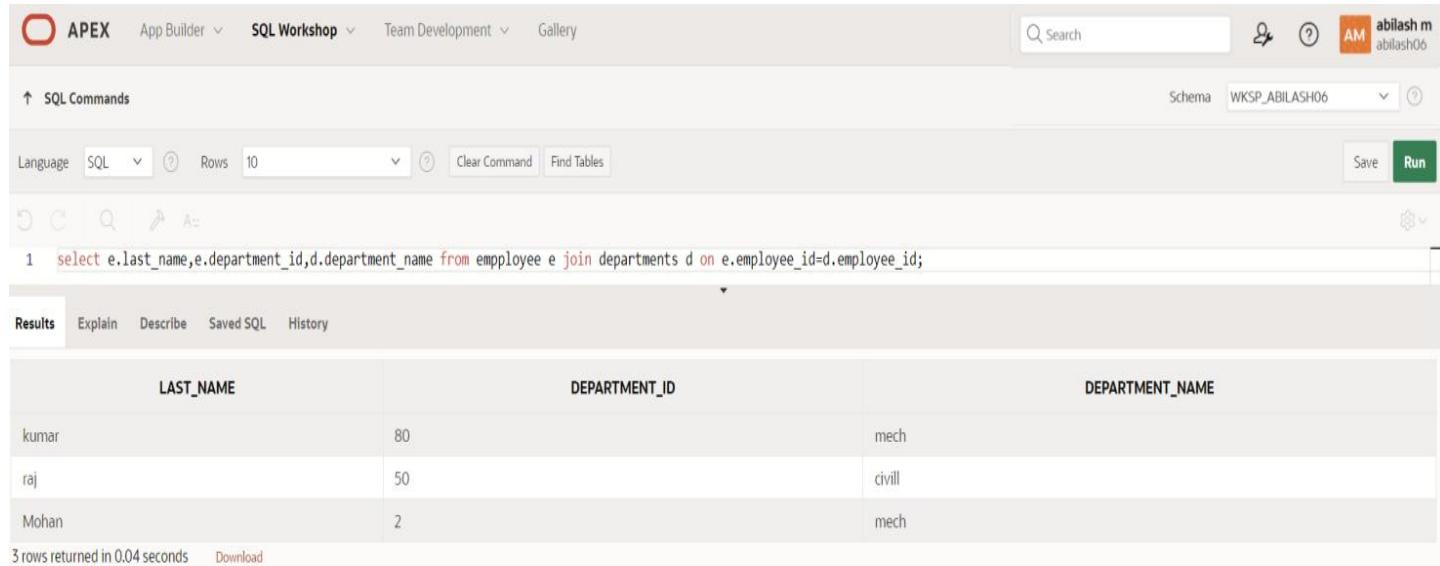
DATE:

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

QUERY:

```
Select e.last_name,e.department_number,d.dept_id from employee e,departments d where e.department_number=d.dept_id;
```

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows a user profile for 'abilash m' (WKSP_ABILASH06). The main area has tabs for SQL Commands, Results (selected), Explain, Describe, Saved SQL, and History. The SQL tab contains the query: 'select e.last_name,e.department_number,d.department_name from employee e join departments d on e.employee_id=d.employee_id;'. The Results tab displays a table with three rows:

LAST_NAME	DEPARTMENT_ID	DEPARTMENT_NAME
kumar	80	mech
raj	50	civil
Mohan	2	mech

Below the table, it says '3 rows returned in 0.04 seconds' and there is a 'Download' link.

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 employee e,departments d where e.department_number=d.dept_id and e.department_number=80;
```

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface, identical to the previous one but with a different query. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows a user profile for 'abilash m' (WKSP_ABILASH06). The main area has tabs for SQL Commands, Results (selected), Explain, Describe, Saved SQL, and History. The SQL tab contains the query: 'select e.job_name,d.location_id from employee e join departments d on e.employee_id=d.employee_id where e.department_id=80'. The Results tab displays a table with one row:

JOB_NAME	LOCATION_ID
manager	67

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

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

QUERY:

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

OUTPUT:

LAST_NAME	DEPARTMENT_NAME	LOCATION_ID	LOCATION_CITY
kumar	mech	67	nagpur
raj	civill	33	chennai
Mohan	mech	67	nagpur

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

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

QUERY:

```
Select e.last_name,d.dept_name from empployee,departments where e.department_number=d.dept_id and last_name like '%a%';
```

OUTPUT:

LAST_NAME	DEPARTMENT_NAME
arun	civill

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

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

QUERY:

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

OUTPUT:

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

```
select last_name,job_name,department_id,department_name from employee where location_city='toronto'
```

The results table displays one row:

LAST_NAME	JOB_NAME	DEPARTMENT_ID	DEPARTMENT_NAME
raj	sales executive	34	cse

1 rows returned in 0.02 seconds

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

QUERY:

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

OUTPUT:

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

```
select last_name as "employee",emp_id as "emp#",manager_name as "manager",manager_id as "mgr" from employee;
```

The results table displays five rows:

employee	emp#	manager	mgr
raj	12	kumar	-
eshawar	4	vinoth	13
kumar	1	thosh	12
arun	2	kamaraj	9
Mohan	1	akilan	19

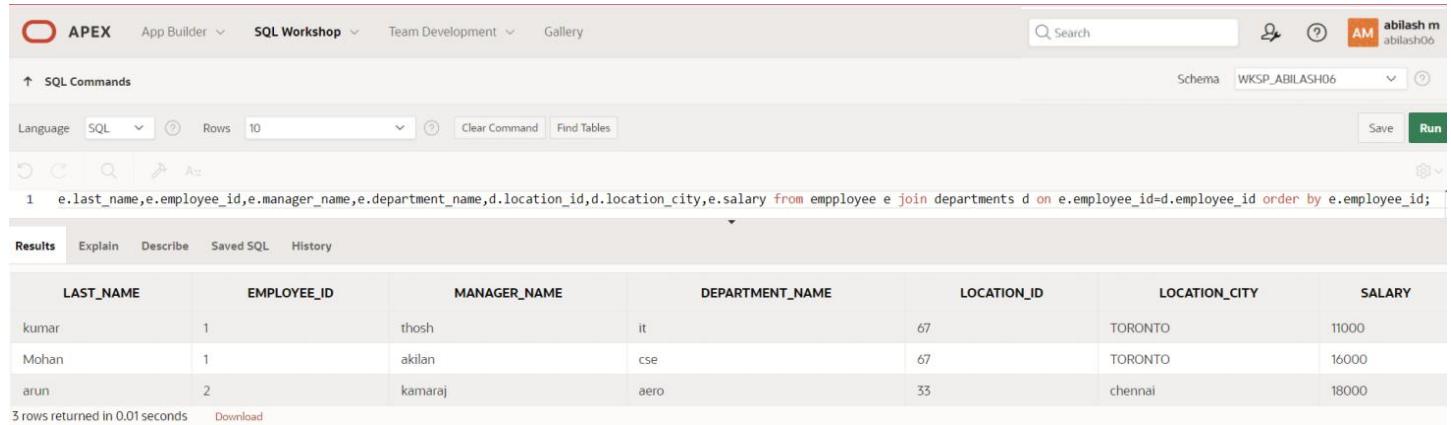
5 rows returned in 0.01 seconds

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

QUERY:

Select w.last_name “Employee”,w.emp_id “emp#”,m.last_name ‘manager’,m.emp_id “Mgr#” from empemployee w left outer join empemployee m on (w.manager_id=m.emp_id);

OUTPUT:



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

```
1 e.last_name,e.employee_id,e.manager_name,e.department_name,d.location_id,d.location_city,e.salary from empemployee e join departments d on e.employee_id=d.employee_id order by e.employee_id;
```

The results table displays the following data:

LAST_NAME	EMPLOYEE_ID	MANAGER_NAME	DEPARTMENT_NAME	LOCATION_ID	LOCATION_CITY	SALARY
kumar	1	thosh	it	67	TORONTO	11000
Mohan	1	akilan	cse	67	TORONTO	16000
arun	2	kamaraj	aero	33	chennai	18000

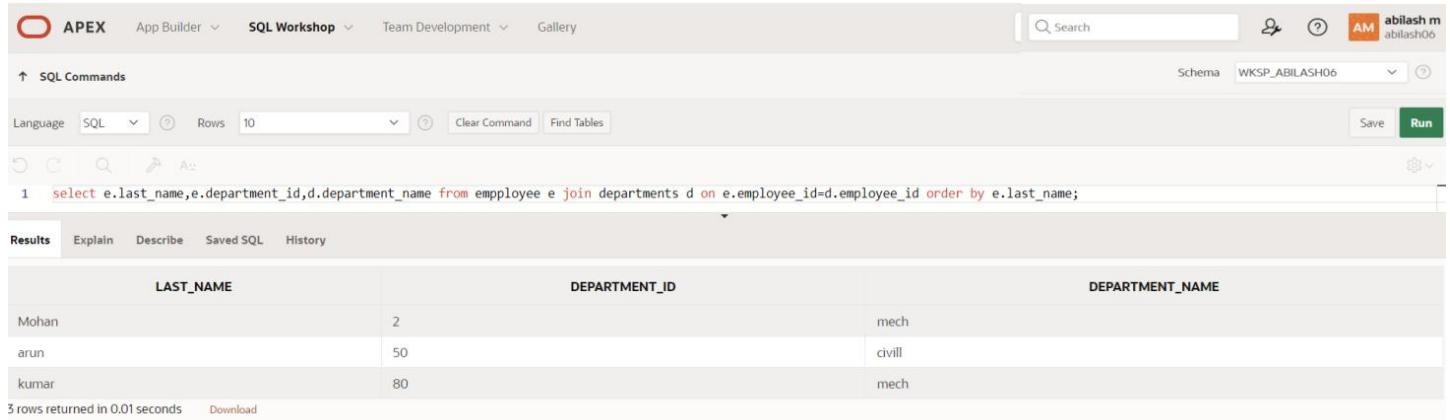
3 rows returned in 0.01 seconds [Download](#)

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

QUERY:

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

OUTPUT:



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

```
1 select e.last_name,e.department_id,d.department_name from empemployee e join departments d on e.employee_id=d.employee_id order by e.last_name;
```

The results table displays the following data:

LAST_NAME	DEPARTMENT_ID	DEPARTMENT_NAME
Mohan	2	mech
arun	50	civil
kumar	80	mech

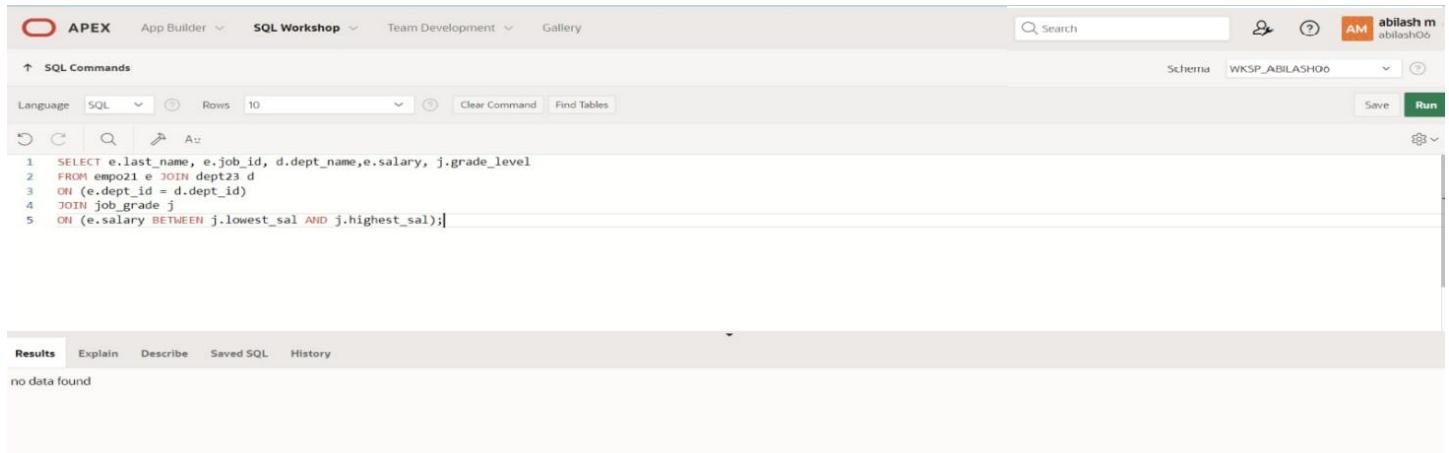
3 rows returned in 0.01 seconds [Download](#)

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 employee e JOIN departments d ON (e.dept_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 SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, and Gallery. The SQL Workshop tab is selected. The main area shows the following SQL command:

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

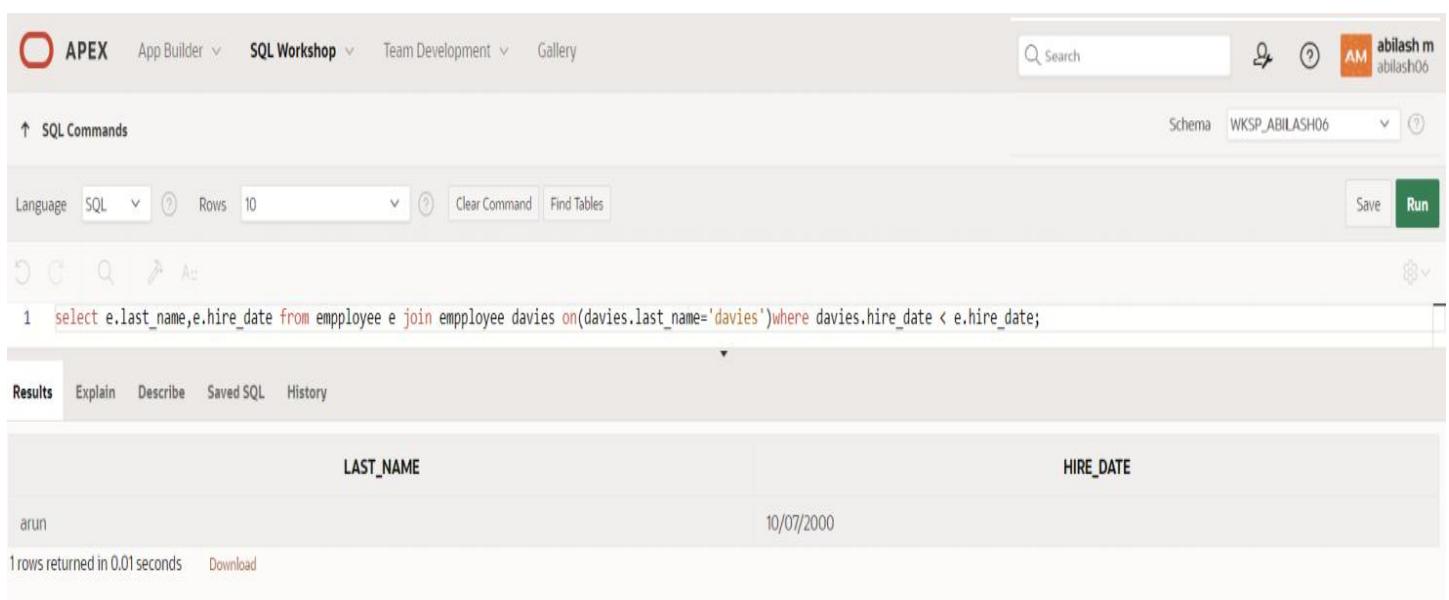
The results section below shows the message "no data found".

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 employee e, employee davies
WHERE davies.last_name = 'Davies'
AND davies.hire_date < e.hire_date;
```

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, and Gallery. The SQL Workshop tab is selected. The main area shows the following SQL command:

```
1 select e.last_name, e.hire_date from employee e join employee davies on(davies.last_name='davies')where davies.hire_date < e.hire_date;
```

The results section below shows the output:

LAST_NAME	HIRE_DATE
arun	10/07/2000

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

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

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows a user profile for 'abilash m' and the workspace 'WKSP_ABILASH06'. The main area has tabs for SQL Commands, Results (selected), Explain, Describe, Saved SQL, and History. The SQL Commands tab contains the following code:

```
1 select e.last_name as "employee",e.hire_date as "emp_hire",e.manager_name as "manager",d.manager_hiredate as "mgr_hire"from employee e join departments d on (e.employee_id=d.employee_id)  
2 where e.hire_date < d.manager_hiredate;
```

The Results tab displays the query results in a grid format:

employee	emp_hire	manager	mgr_hire
kumar	04/05/1998	thosh	08/07/2000
arun	10/07/2000	kamaraj	04/24/2001
Mohan	07/09/1994	akilan	08/07/2000

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

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

RESULT:

AGGREGATING DATA USING GROUP FUNCTIONS

EX_NO:8

DATE:

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

TRUE

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

FALSE

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

FALSE

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. A search bar and user profile 'abilash m abilash06' are also present. The main workspace displays the SQL command entered:

```
1 select Round(Max (salary),0)"Maximum", Round (Min (salary),0) "Minimum",
2 round(sum(salary),0)"sum", round (avg(salary),0) "Average" from MY_EMPLOYEESS;
3
```

The results section shows the output of the query:

	Maximum	Minimum	sum	Average
20000	6000	51000	12750	

Below the results, it says '1 rows returned in 0.05 seconds'. The bottom of the page includes footer links for Explain, Describe, Saved SQL, History, and language selection (en).

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 EMPLOYEE group by job_id;
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (which is selected), Team Development, and Gallery. A search bar and user information for 'abilash m abilash06' are also present. The main area is titled 'SQL Commands' with a 'Run' button. The code entered is:

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

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

JOB_ID	MAXIMUM	Minimum	sum	average
21	15000	15000	15000	15000
56	10000	10000	10000	10000
53	6000	6000	6000	6000
24	20000	20000	20000	20000

4 rows returned in 0.00 seconds

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (which is selected), Team Development, and Gallery. A search bar and user information for 'abilash m abilash06' are also present. The main area is titled 'SQL Commands' with a 'Run' button. The code entered is:

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

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

JOB_ID	COUNT(*)
47	1

1 rows returned in 0.01 seconds

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 empLOYEE;
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. A search bar and user profile are also present. The main workspace shows the following SQL command:

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

The results section displays the output:

Number of managers
3

Below the table, it says "1 rows returned in 0.00 seconds". The bottom of the page includes copyright information and a link to the Oracle APEX homepage.

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. A search bar and user profile are also present. The main workspace shows the following SQL command:

```
1 select max(salary)-min(salary) difference from MY_EMPLOYEESS;
```

The results section displays the output:

DIFFERENCE
14000

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

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 empemployee where manager_id is not null group by manager_id having min(salary) >6000 order by min(salary) desc;
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. The right side shows a user profile for 'abilash m abilash06'. The main area has tabs for 'SQL Commands' and 'Results'. Under 'SQL Commands', the query is pasted:

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

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

MANAGER_ID	MIN(SALARY)
25	20000
71	10000

Below the table, it says '2 rows returned in 0.01 seconds' and has a 'Download' link. At the bottom, it shows the URL '220701006@rajalakshmi.edu.in', the schema 'abilash06', the language 'en', copyright information 'Copyright © 1999, 2023, Oracle and/or its affiliates.', and the version 'Oracle APEX 23.2.4'.

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

QUERY:

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

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 'abilash m abilash06'. The main area has tabs for 'SQL Commands' and 'Results'. Under 'SQL Commands', the query is pasted:

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

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

MANAGER_ID	MIN(SALARY)
25	20000
71	10000

Below the table, it says '2 rows returned in 0.01 seconds' and has a 'Download' link. At the bottom, it shows the URL '220701006@rajalakshmi.edu.in', the schema 'abilash06', the language 'en', copyright information 'Copyright © 1999, 2023, Oracle and/or its affiliates.', and the version 'Oracle APEX 23.2.4'.

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. A search bar and user information (AM abilash m abilash06) are also present. The SQL Commands tab is selected, showing the schema as WKSP_ABILASH06. The SQL editor contains the query:

```
1 select job_id "job", sum(decode(department_id,20,salary))"Dept20",sum (decode(department_id ,50, salary))  
2 "dept50",sum (decode(department_id ,80, salary)) "dept80",sum (decode(department_id ,90, salary)) "dept90",sum(salary)  
3 "TOTAL" from MY_EMPPLOYEES group by job_id
```

The results section displays the query output as a matrix:

job	Dept20	dept50	dept80	dept90	TOTAL
56	-	-	-	-	10000
53	6000	-	-	-	6000
47	-	-	-	-	15000
24	-	20000	-	-	20000

Below the table, it says "4 rows returned in 0.01 seconds". The bottom of the page includes footer information: 220701006@rajalakshmi.edu.in, abilash06, en, Copyright © 1999, 2023, Oracle and/or its affiliates, and Oracle APEX 23.2.4.

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

QUERY:

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

OUTPUT:

Language SQL Rows 10 Clear Command Find Tables Save Run

```
1 t(*) "Number of people",round(avg(salary),2) "salary" from mydept d inner join employee e on(d.deptid =e.department_id ) group by d.dept_name ,d.loc;
```

Results Explain Describe Saved SQL History

dept_name	department location	Number of people	salary
CSBS	Bangaluru	1	15000
CSE	Chennai	3	16333.33

2 rows returned in 0.05 seconds Download 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:

SUB QUERIES

EX_NO:9

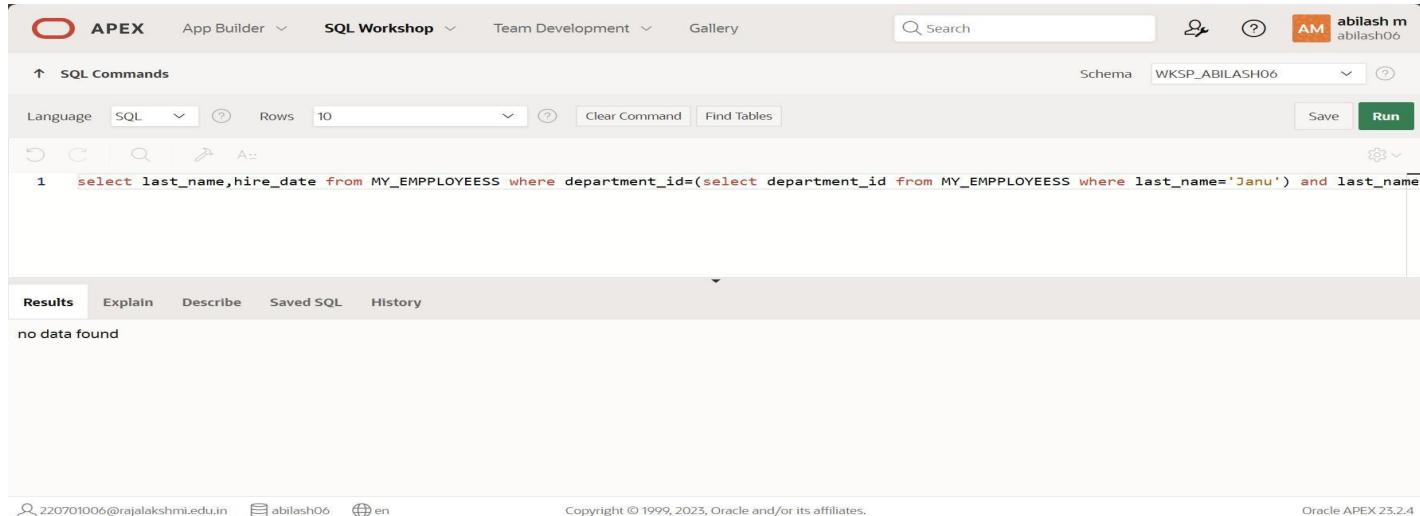
DATE:

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

QUERY:

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

OUTPUT:



The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. The right side shows a user profile for 'abilash m' (abilash06). The main area has tabs for 'SQL Commands' and 'Results'. The SQL command entered is:

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

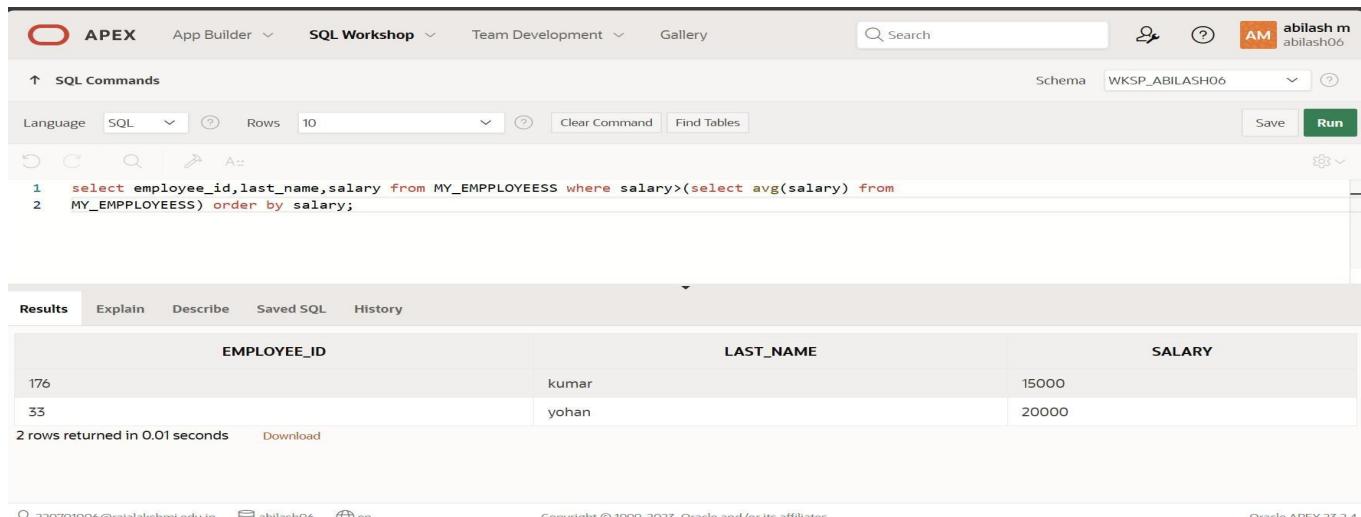
The results tab shows the message 'no data found'.

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

QUERY:

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

OUTPUT:



The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. The right side shows a user profile for 'abilash m' (abilash06). The main area has tabs for 'SQL Commands' and 'Results'. The SQL command entered is:

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

The results tab displays a table with columns 'EMPLOYEE_ID', 'LAST_NAME', and 'SALARY'. The data is:

EMPLOYEE_ID	LAST_NAME	SALARY
176	kumar	15000
33	yohan	20000

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

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. A search bar and user profile 'abilash m abilash06' are also present. The main area is titled 'SQL Commands' with tabs for 'Language' (set to 'SQL'), 'Rows' (set to 10), and buttons for 'Clear Command', 'Find Tables', 'Save', and 'Run'. The code entered is:

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

The results section shows a single row returned:

EMPLOYEE_ID	LAST_NAME
176	kumar

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

At the bottom, the footer includes user information (220701006@rajalakshmi.edu.in, abilash06, en), copyright (Copyright © 1999, 2023, Oracle and/or its affiliates.), and version (Oracle APEX 23.2.4).

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface, identical to the previous one but with a different query. The results section indicates 'no data found'.

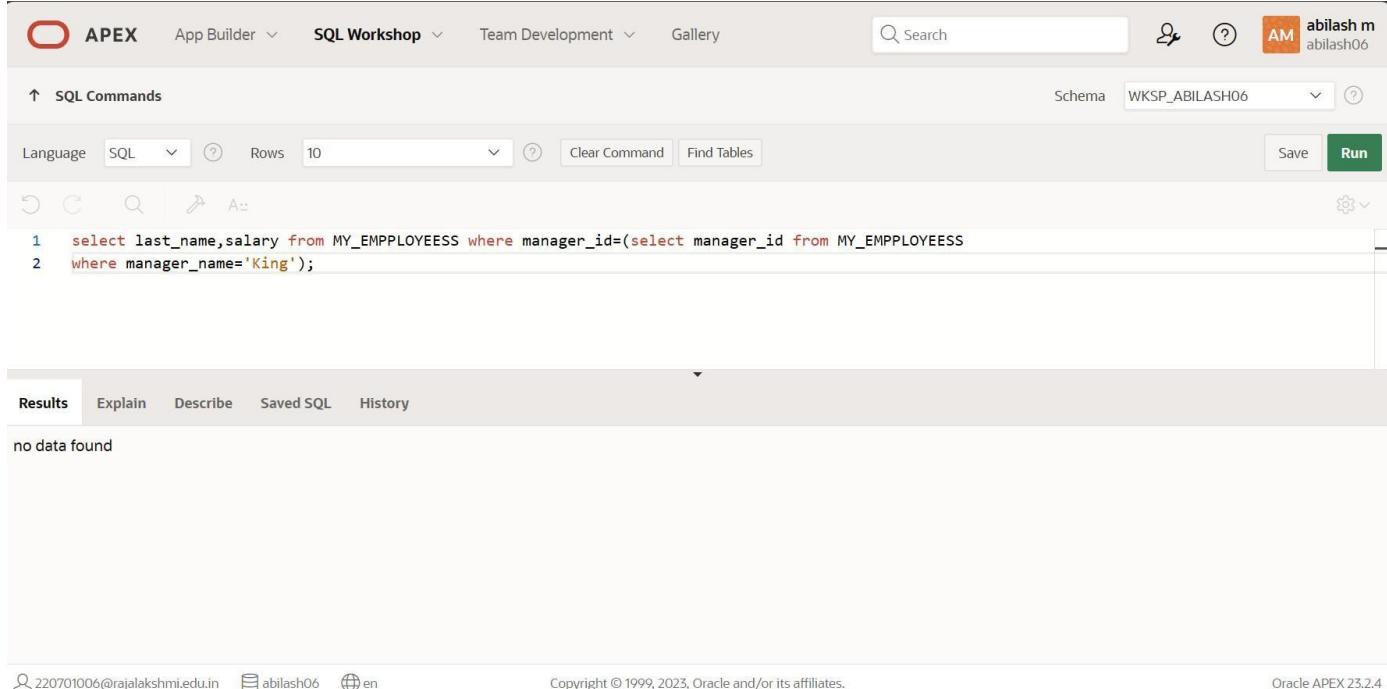
At the bottom, the footer includes user information (220701006@rajalakshmi.edu.in, abilash06, en), copyright (Copyright © 1999, 2023, Oracle and/or its affiliates.), and version (Oracle APEX 23.2.4).

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

QUERY:

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

OUTPUT:



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 side, there's a search bar, user information ('abilash m abilash06'), and a 'Run' button. Below the tabs, there are buttons for 'Language' (set to 'SQL'), 'Rows' (set to 10), 'Clear Command', and 'Find Tables'. The main area contains the SQL code:

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

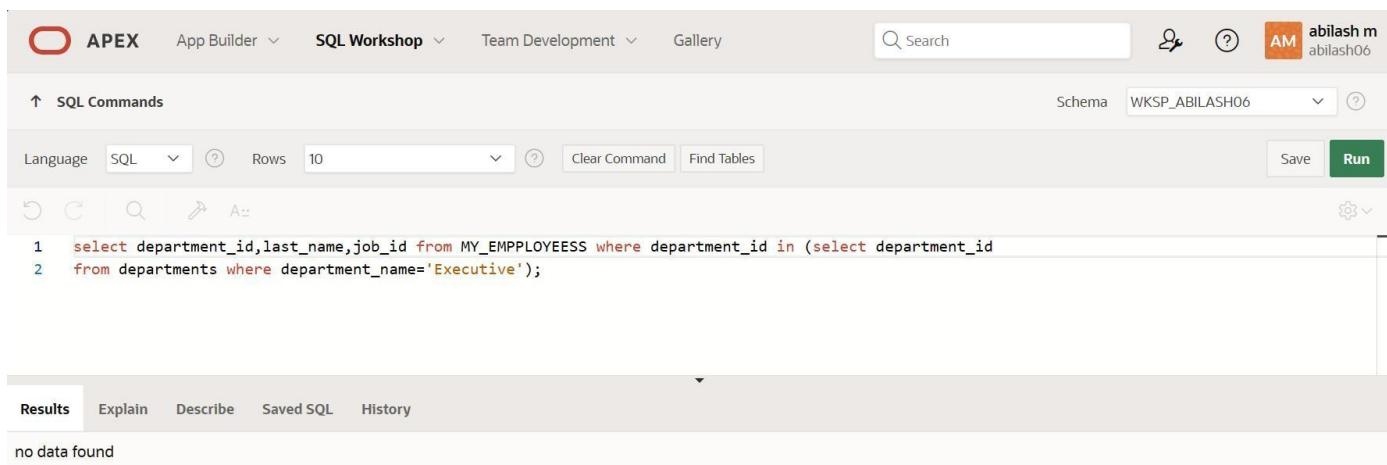
Below the code, there are tabs for 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History'. The 'Results' tab is selected, showing the message 'no data found'. At the bottom of the page, there are footer links for user info, language, and copyright information.

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

QUERY:

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

OUTPUT:



The screenshot shows the Oracle APEX SQL Workshop interface, identical to the previous one but with different SQL code. The 'SQL Workshop' tab is selected at the top. The main area contains the SQL code:

```
1 select department_id,last_name,job_id from MY_EMPLOYEES where department_id in (select department_id  
2 from departments where department_name='Executive');
```

Below the code, there are tabs for 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History'. The 'Results' tab is selected, showing the message 'no data found'. At the bottom of the page, there are footer links for user info, language, and copyright information.

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. A search bar and user information (AM abilash06) are also present. The main area is titled "SQL Commands" and shows the following SQL code:

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

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

EMPLOYEE_ID	LAST_NAME	SALARY
33	yohan	20000

Below the grid, it says "1 rows returned in 0.00 seconds". At the bottom of the page, there are footer links for user info (220701006@rajalakshmi.edu.in), schema (abilash06), and language (en). The copyright notice reads "Copyright © 1999, 2023, Oracle and/or its affiliates." and the version is "Oracle APEX 23.2.4".

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

RESULT:

USING THE SET OPERATORS

EX_NO:10

DATE:

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', 'Gallery', a search bar, and a user profile 'abilash m abilash06'. The main area is titled 'SQL Commands' with tabs for 'Language' (set to 'SQL'), 'Rows' (set to 10), 'Clear Command', 'Find Tables', 'Save', and 'Run'. The SQL editor contains the following code:

```
1 select department_id from departments minus select department_id from MY_EMPLOYEES where  
2 job_name='st_clerk';
```

The results section is titled 'Results' and displays the output:

DEPARTMENT_ID
12
45
-

Below the results, it says '3 rows returned in 0.01 seconds' and has a 'Download' link. The bottom footer includes user information (220701006@rajalakshmi.edu.in, abilash06, en), copyright notice (Copyright © 1999, 2023, Oracle and/or its affiliates.), and version information (Oracle APEX 23.2.4).

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', 'Gallery', a search bar, and a user profile 'abilash m abilash06'. The main area is titled 'SQL Commands' with tabs for 'Language' (set to 'SQL'), 'Rows' (set to 10), 'Clear Command', 'Find Tables', 'Save', and 'Run'. The SQL editor contains the following code:

```
1 SELECT country_id,country_name FROM country MINUS SELECT 1.country_id,c.country_name FROM location 1, country c WHERE 1.country_id = c.country_id
```

The results section is titled 'Results' and displays the output:

COUNTRY_ID	COUNTRY_NAME
19	New Zealand

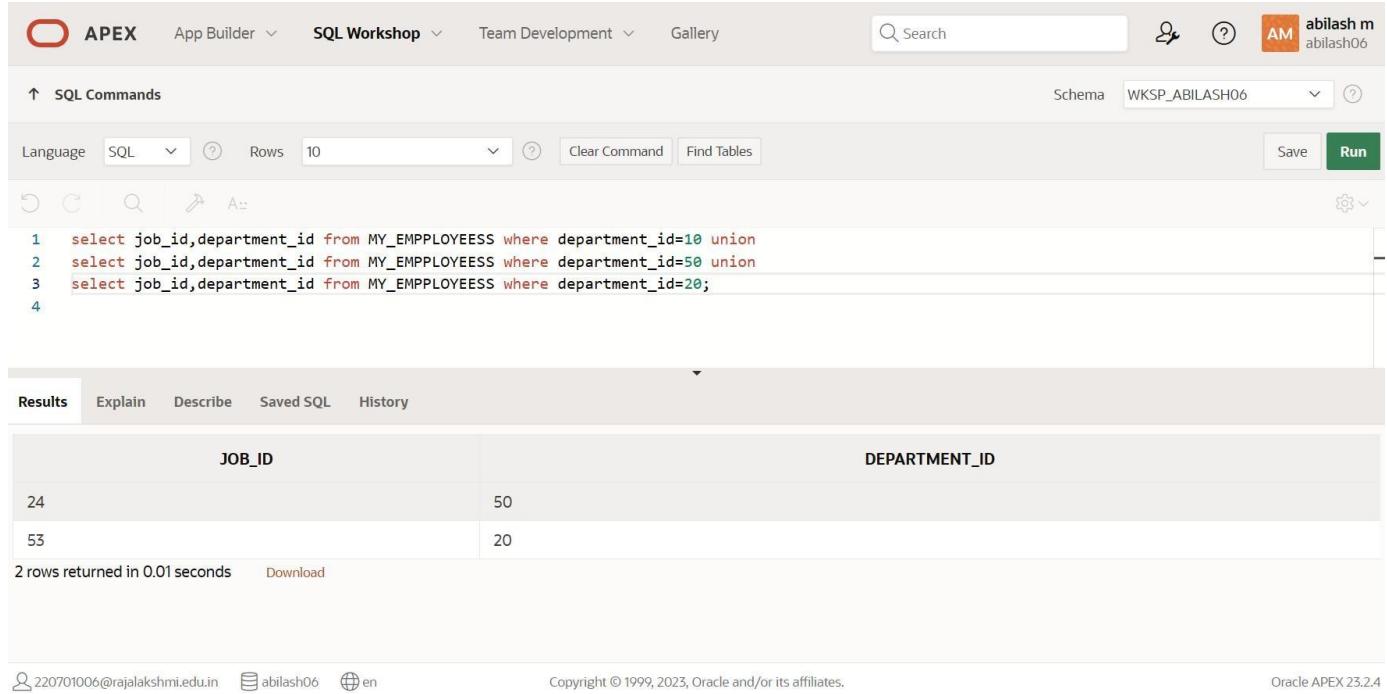
Below the results, it says '1 rows returned in 0.02 seconds' and has a 'Download' link. The bottom footer includes user information (220701006@rajalakshmi.edu.in, abilash06, en), copyright notice (Copyright © 1999, 2023, Oracle and/or its affiliates.), and version information (Oracle APEX 23.2.4).

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

QUERY:

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

OUTPUT:



The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. A search bar and user profile are also present. The main area is titled "SQL Commands" and contains the following SQL code:

```
1 select job_id,department_id from MY_EMPLOYEESS where department_id=10 union
2 select job_id,department_id from MY_EMPLOYEESS where department_id=50 union
3 select job_id,department_id from MY_EMPLOYEESS where department_id=20;
4
```

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

JOB_ID	DEPARTMENT_ID
24	50
53	20

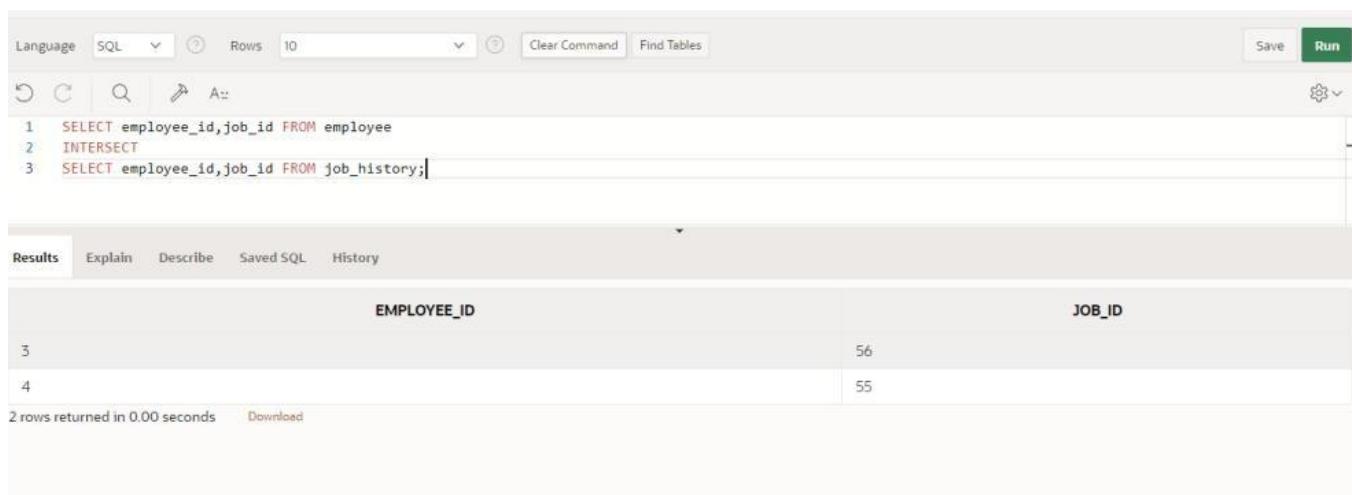
Below the table, it says "2 rows returned in 0.01 seconds" and provides a "Download" link. The bottom of the page shows user information (220701006@rajalakshmi.edu.in, abilash06, en), copyright notice (Copyright © 1999, 2023, Oracle and/or its affiliates.), and the version (Oracle APEX 25.2.4).

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

QUERY:

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

OUTPUT:



The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. A search bar and user profile are also present. The main area is titled "SQL Commands" and contains the following SQL code:

```
1 SELECT employee_id,job_id FROM employee
2 INTERSECT
3 SELECT employee_id,job_id FROM job_history;
```

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

EMPLOYEE_ID	JOB_ID
3	56
4	55

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

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. A search bar and user profile are also present. The main area is titled 'SQL Commands' and shows the following SQL code:

```
1 select first_name||' '||last_name as "Name",department_id from MY_EMPLOYEES union all select department_name,department_id from departments;
```

The results tab is selected, displaying a table with two columns: 'Name' and 'DEPARTMENT_ID'. The data is as follows:

Name	DEPARTMENT_ID
ganesh kumar	36
mani jeyaraj	20
ravi abidhea	43
mohan yohan	50

At the bottom, there are footer links for support, copyright information (Copyright © 1999, 2023, Oracle and/or its affiliates.), and the Oracle APEX version (Oracle APEX 23.2.4).

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

RESULT:

CREATING VIEWS

EXP NO: 11

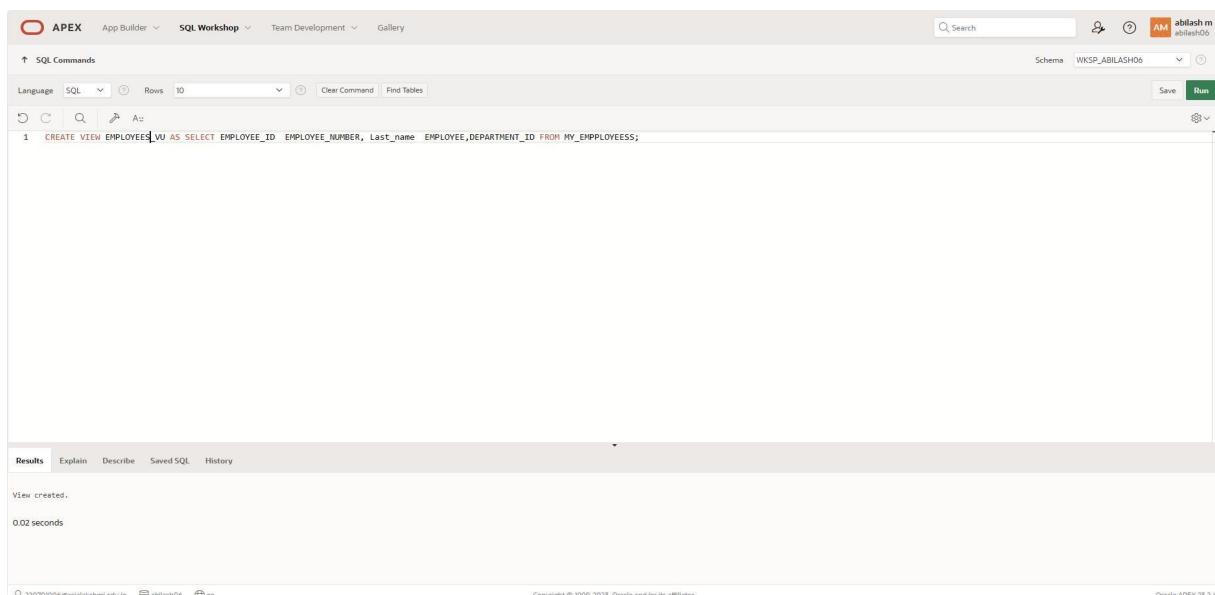
DATE:

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 VIEW EMPLOYEE_VU AS SELECT EMP_ID AS EMPLOYEE_NUMBER, Last_name AS EMPLOYEE,  
DEPT_ID FROM EMPLOY;
```

OUTPUT:



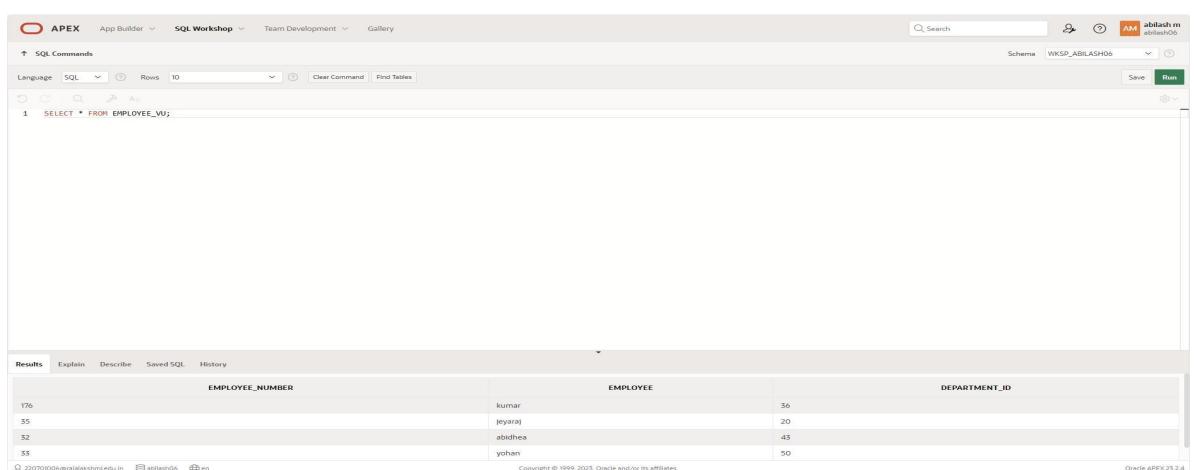
The screenshot shows the Oracle APEX SQL Workshop interface. In the top navigation bar, 'APEX' is selected. Below it, 'SQL Workshop' is active. The main area is titled 'SQL Commands'. A single line of SQL code is entered in the command editor:
`1 CREATE VIEW EMPLOYEE_VU AS SELECT EMPLOYEE_ID AS EMPLOYEE_NUMBER, last_name AS EMPLOYEE, DEPARTMENT_ID FROM MY_EMPLOYEES;`

2. Display the contents of the EMPLOYEE_VU view.

QUERY:

```
SELECT * FROM EMPLOYEE_VU;
```

OUTPUT:



The screenshot shows the Oracle APEX SQL Workshop interface again. The 'Results' tab is selected. The output of the previous query is displayed in a table:

EMPLOYEE_NUMBER	EMPLOYEE	DEPARTMENT_ID
176	kumar	36
35	jeeyraj	20
32	abisheha	45
35	yohan	50

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 highlighted. The main area contains the SQL command:

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

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

VIEW_NAME	TEXT
EMPLOYEES_VU	SELECT EMPLOYEE_ID, EMPLOYEE_NUMBER, Last_name, DEPARTMENT_ID FROM MY_EMPLOYEES
EMPLOYEE_VU	SELECT EMPLOYEE_ID, EMPLOYEE_NUMBER, Last_name, DEPARTMENT_ID FROM MY_EMPLOYEES

Text at the bottom of the results pane: "2 rows returned in 0.04 seconds" and "Download".

At the bottom right of the interface: "Copyright © 1999, 2023, Oracle and/or its affiliates." and "Oracle APEX 23.2.4".

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

QUERY:

```
SELECT 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 highlighted. The main area contains the SQL command:

```
1 SELECT EMPLOYEE, DEPT_ID
2 FROM EMPLOYEE_VU;
3
4
```

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

EMPLOYEE	DEPT_ID
larry	20
DUVIES	20
harry	20
manny	90
HAZEL	5

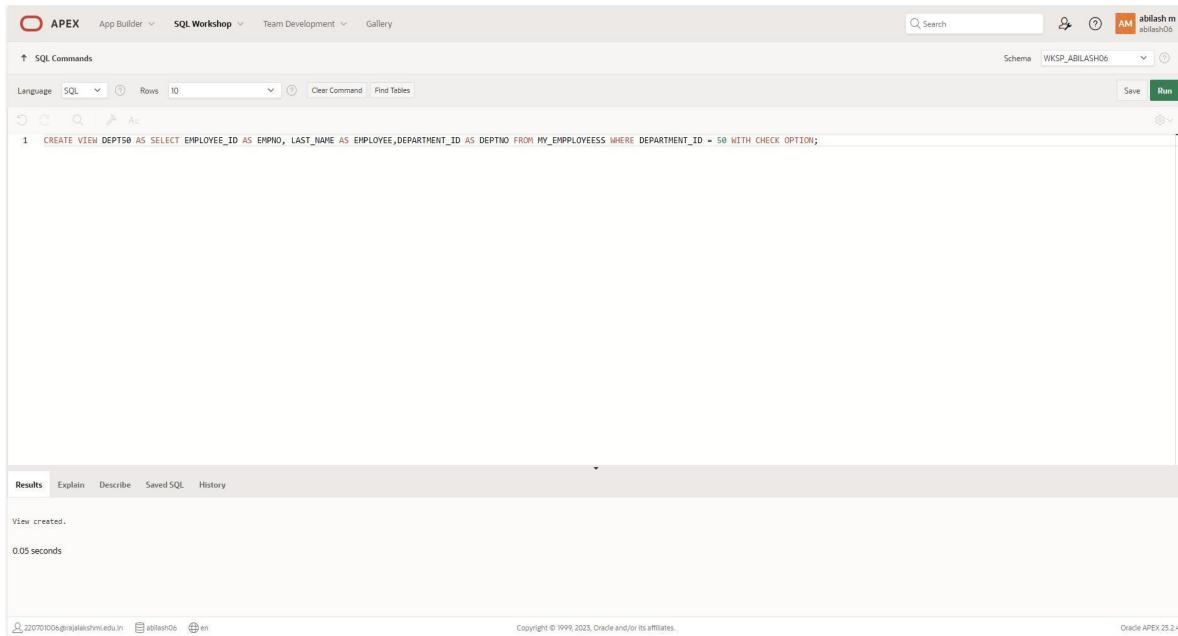
Text at the bottom of the results pane: "5 rows returned in 0.01 seconds" and "Download".

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

QUERY:

```
CREATE VIEW DEPT50 AS SELECT EMP_ID AS EMPNO, LAST_NAME AS EMPLOYEE,DEPT_ID AS DEPTNO FROM EMPLOY WHERE DEPT_ID = 50 WITH CHECK OPTION;
```

OUTPUT:



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

```
1 CREATE VIEW DEPT50 AS SELECT EMPLOYEE_ID AS EMPNO, LAST_NAME AS EMPLOYEE,DEPARTMENT_ID AS DEPTNO FROM MY_EMPLOYEES WHERE DEPARTMENT_ID = 50 WITH CHECK OPTION;
```

Below the command, the results of the execution are shown:

View created.
0.05 seconds

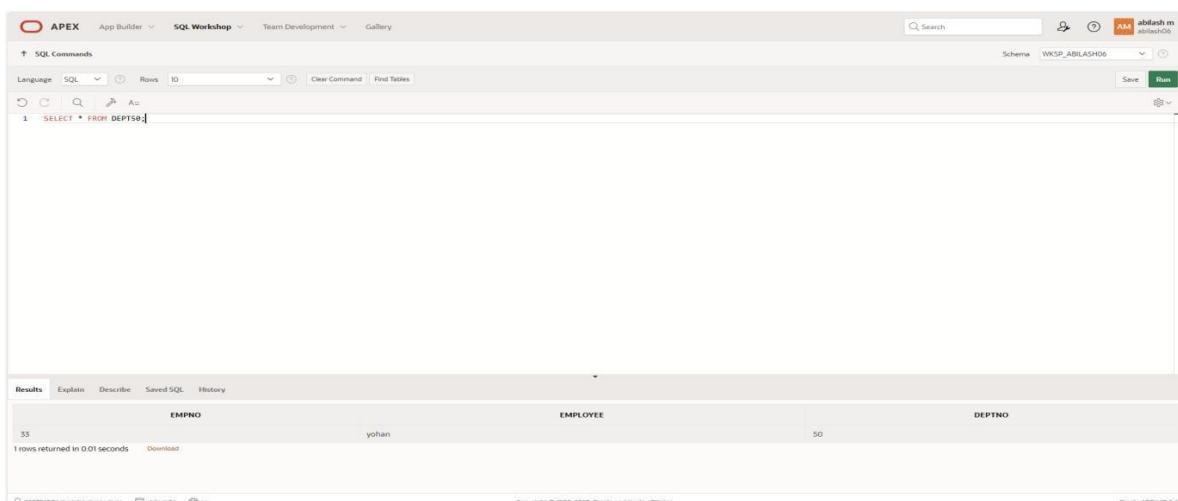
At the bottom, the URL is 202701004@rajalakshmi.edu.in, the session is abilash06, and the page is 1 of 1. The footer indicates Copyright © 1999-2023, Oracle and/or its affiliates, and Oracle APEX 25.2.4.

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

QUERY:

```
SELECT * FROM DEPT50;
```

OUTPUT:



The screenshot shows the Oracle APEX SQL Workshop interface. In the top navigation bar, 'APEX' is selected. Below it, 'SQL Commands' is highlighted. The main area contains the SQL command for selecting from the DEPT50 view:

```
1 SELECT * FROM DEPT50;
```

Below the command, the results of the execution are shown:

EMPNO	EMPLOYEE	DEPTNO
33	yohan	50

1 rows returned in 0.01 seconds

At the bottom, the URL is 202701004@rajalakshmi.edu.in, the session is abilash06, and the page is 1 of 1. The footer indicates Copyright © 1999-2023, Oracle and/or its affiliates, and Oracle APEX 25.2.4.

7. Attempt to reassign Matos to department 80.

QUERY:

```
UPDATE DEPARTMENT50 SET DEPTNO = 80 WHERE EMPLOYEE = 'Matos';
```

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 'abilash m' and the schema 'WKSP_ABILASH06'. The main area is titled 'SQL Commands' and contains a command input field with the following SQL:

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

Below the command, the results section shows the output:

```
0 row(s) updated.  
0.06 seconds
```

At the bottom, the URL is https://apex.oracle.com/pls/apex/r/apex/team-development/teamdev..., the copyright notice is Copyright © 1999, 2023, Oracle and/or its affiliates, and the version is Oracle APEX 23.2.4.

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

QUERY:

```
CREATE VIEW SALARY_VU AS SELECT E.LAST_NAME AS Employee, D.DEPARTMENT_NAME AS Department, E.SALARY AS Salary, JG.GRADE_LEVEL AS Grade FROM EMPLOY E JOIN DEPARTMENT D ON E.DEPT_ID = D.DEPARTMENT_ID JOIN JOB_GRADE JG ON E.SALARY BETWEEN JG.LOWEST_SAL AND JG.HIGHEST_SAL;
```

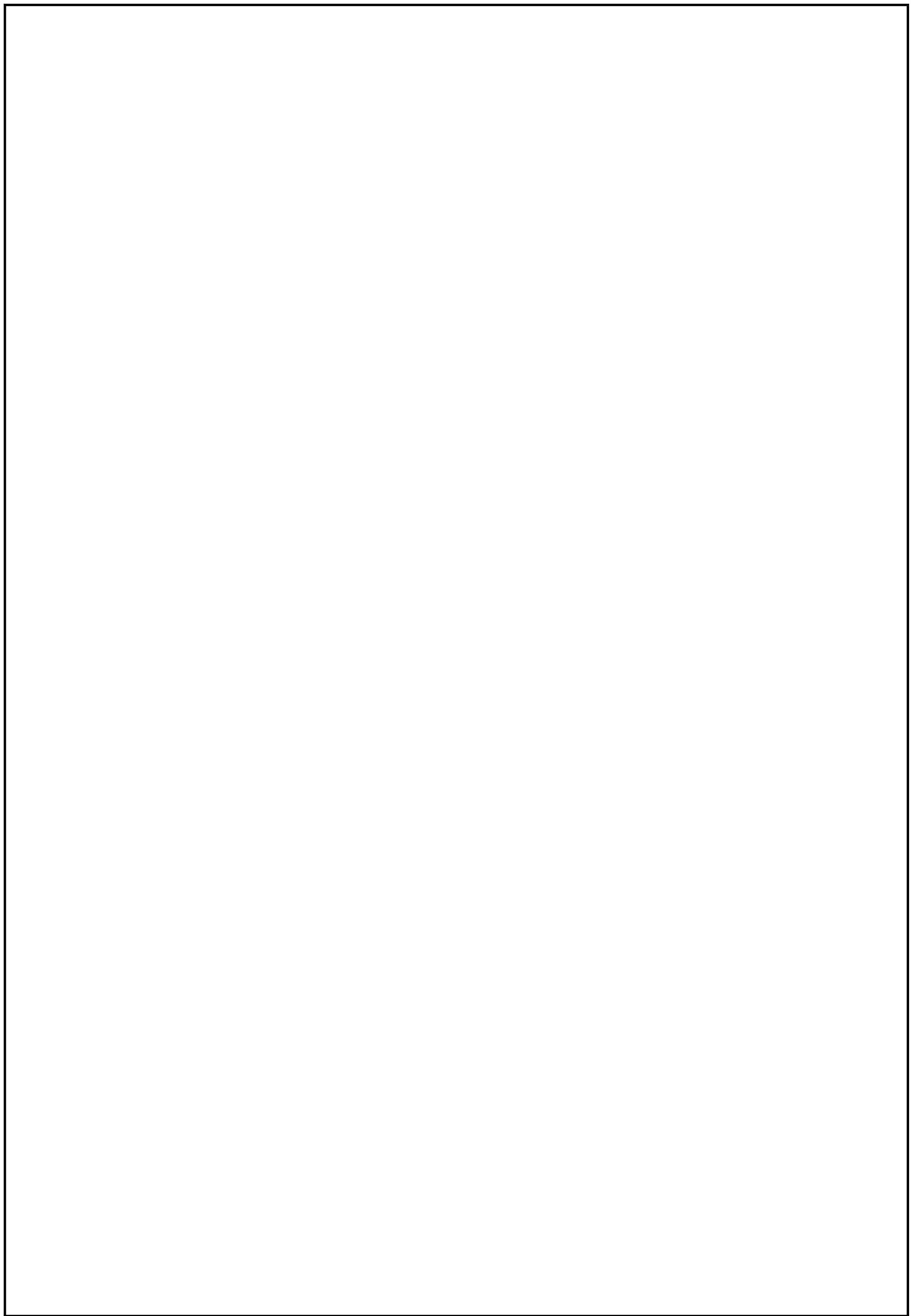
OUTPUT:

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 'abilash m' and the schema 'WKSP_ABILASH06'. The main area is titled 'SQL Commands' and contains the SQL code for creating a view:

```
1 CREATE VIEW SALARY_VU AS  
2 SELECT E.LAST_NAME AS Employee,  
3 D.DEPARTMENT_NAME AS Department,  
4 E.SALARY AS Salary,  
5 JG.GRADE_LEVEL AS Grade  
6 FROM EMPLOY E  
7 JOIN DEPARTMENT D ON E.DEPT_ID = D.DEPARTMENT_ID  
8 JOIN JOB_GRADE JG ON E.SALARY BETWEEN JG.LOWEST_SAL AND JG.HIGHEST_SAL;
```

Below the command, the results section shows the output:

```
View created.  
0.05 seconds
```



RESULT:

EXERCISE 12

PRACTICE QUESTIONS

Intro to Constraints; NOT NULL and UNIQUE Constraints

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

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

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

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

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

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

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

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

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

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

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

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

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

```

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

```

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

Table Created.

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

DESCRIBE f_global_locations;

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

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

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

PRIMARY KEY, FOREIGN KEY, and CHECK Constraints

1. What is the purpose of a

- PRIMARY KEY
- FOREIGN KEY
- CHECK CONSTRAINT

a. PRIMARY KEY

Uniquely identify each row in table.

b. FOREIGN KEY

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

c. CHECK CONSTRAINT

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

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

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

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

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

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

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

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

```
SELECT * FROM animals;
```

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

COLUMN LEVEL STATEMENT:

```
ALTER TABLE animals
```

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

TABLE LEVEL STATEMENT:

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

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

- a. ON DELETE CASCADE

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

- b. ON DELETE SET NULL

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

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

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

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

PRACTICE PROBLEM

Managing Constraints

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

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

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

- ADD
- DROP
- ENABLE
- DISABLE

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

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

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

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

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

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

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

COPY_D_CLT_CLIENT_NUMBER_PK

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

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

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

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

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

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

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

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

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

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

1 row(s) inserted.

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

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

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

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

1 row(s) deleted.

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

Table altered.

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

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

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

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

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

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

EXERCISE 13

Creating Views

1. What are three uses for a view from a DBA's perspective?

- **Restrict access and display selective columns**
- **Reduce complexity of queries from other internal systems. So, providing a way to view same data in a different manner.**
- **Let the app code rely on views and allow the internal implementation of tables to be modified later.**

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

CREATE VIEW view_d_songs AS

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

3. SELECT * FROM view_d_songs. What was returned?

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

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

4. REPLACE view_d_songs. Add type_code to the column list. Use aliases for all columns.

Or use alias after the CREATE statement as shown.

CREATE OR REPLACE VIEW view_d_songs AS

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

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

CREATE OR REPLACE VIEW view_d_events_pkgs AS

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

6. It is company policy that only upper-level management be allowed access to individual employee salaries. The department managers, however, need to know the minimum, maximum, and average salaries, grouped by department. Use the Oracle database to prepare a view that displays the needed information for department

managers.

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

DML Operations and Views

Use the DESCRIBE statement to verify that you have tables named copy_d_songs, copy_d_events, copy_d_cds, and copy_d_clients in your schema. If you don't, write a query to create a copy of each.

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

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

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

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

2. Use the CREATE or REPLACE option to create a view of *all* the columns in the copy_d_songs table called view_copy_d_songs.

```
CREATE OR REPLACE VIEW view_copy_d_songs AS
```

```
SELECT *
```

```
FROM copy_d_songs;
```

```
SELECT * FROM view_copy_d_songs;
```

3. Use view_copy_d_songs to INSERT the following data into the underlying copy_d_songs table. Execute a SELECT * from copy_d_songs to verify your DML command. See the graphic.

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

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

4. Create a view based on the DJs on Demand COPY_D_CDS table. Name the view read_copy_d_cds. Select all columns to be included in the view. Add a WHERE clause to restrict the year to 2000. Add the WITH READ ONLY option.

```
CREATE OR REPLACE VIEW read_copy_d_cds AS
```

```
SELECT *
```

```
FROM copy_d_cds
```

```
WHERE year = '2000'
```

```
WITH READ ONLY ;
```

```
SELECT * FROM read_copy_d_cds;
```

5. Using the read_copy_d_cds view, execute a DELETE FROM read_copy_d_cds WHERE cd_number = 90;

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

6. Use REPLACE to modify read_copy_d_cds. Replace the READ ONLY option with WITH CHECK OPTION CONSTRAINT ck_read_copy_d_cds. Execute a SELECT * statement to verify that the view exists.

```
CREATE OR REPLACE VIEW read_copy_d_cds AS
```

```
SELECT *
FROM copy_d_cds
WHERE year = '2000'
WITH CHECK OPTION CONSTRAINT ck_read_copy_d_cds;
```

7. Use the read_copy_d_cds view to delete any CD of year 2000 from the underlying copy_d_cds.

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

8. Use the read_copy_d_cds view to delete cd_number 90 from the underlying copy_d_cds table.

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

9. Use the read_copy_d_cds view to delete year 2001 records.

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

10. Execute a SELECT * statement for the base table copy_d_cds. What rows were deleted?

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

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

DELETE,INSERT,MODIFY restricted if it contains:

Group functions
GROUP BY CLAUSE
DISTINCT
pseudocolumn ROWNUM Keyword

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

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

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

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

Managing Views

1. Create a view from the copy_d_songs table called view_copy_d_songs that includes only the title and artist.

Execute a SELECT * statement to verify that the view exists.

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

```
SELECT * FROM view_copy_d_songs;
```

- 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

- 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;
```

- 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;
```

- 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

- 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. The top navigation bar includes links for APEX, App Builder, SQL Workshop (which is selected), Team Development, and Gallery. A search bar and user profile 'abilash m abilash06' are also present. The main workspace is titled 'SQL Commands' and shows the following command being run:

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

Below the command, the results tab is active, displaying the output:

```
Sequence created.
```

Execution time: 0.01 seconds.

At the bottom, the footer includes the URL '220701006@rajalakshmi.edu.in', the schema 'abilash06', the language 'en', copyright information 'Copyright © 1999, 2023, Oracle and/or its affiliates.', and the version 'Oracle APEX 23.2.4'.

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, similar to the previous one but with a different query. The top navigation bar and workspace are identical. The results tab is active, showing the output of the following query:

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

The output table displays the following data:

SEQUENCE_NAME	MAX_VALUE	INCREMENT_BY	LAST_NUMBER
DEPT_ID_SEQ	1000	10	200

Execution time: 0.02 seconds.

At the bottom, the footer includes the URL '220701006@rajalakshmi.edu.in', the schema 'abilash06', the language 'en', copyright information 'Copyright © 1999, 2023, Oracle and/or its affiliates.', and the version 'Oracle APEX 23.2.4'.

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 tabs for APEX, App Builder, SQL Workshop (which is selected), Team Development, and Gallery. On the right, there's a search bar, user information (abilash m abilash06), and a schema dropdown set to WKSP_ABILASH06. Below the toolbar, there are buttons for Language (SQL), Rows (10), Clear Command, Find Tables, Save, and Run. The main workspace contains the SQL command: `1 INSERT INTO dept VALUES (dept_id_seq.nextval, 'Education');`. The results section at the bottom shows the output: "1 row(s) inserted." and "0.02 seconds". The footer displays copyright information and the version Oracle APEX 23.2.4.

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes tabs for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there's a search bar, user information (abilash m abilash06), and a schema dropdown set to WKSP_ABILASH06. Below the toolbar, there are buttons for Language (SQL), Rows (10), Clear Command, Find Tables, Save, and Run. The main workspace contains the SQL command: `1 CREATE INDEX emp_dept_id_idx ON MY_EMPLOYEES (department_id);`. The results section at the bottom shows the output: "Index created." and "0.04 seconds". The footer displays copyright information 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 (which is selected), Team Development, and Gallery. A search bar and a user profile for 'abilash06' are also at the top. Below the tabs, there's a toolbar with icons for SQL, Clear Command, Find Tables, Save, and Run. The main area contains the SQL command: `SELECT index_name,table_name,uniqueness FROM user_indexes WHERE table_name='MY_EMPLOYEES';`. The results tab is selected, showing the output: 'Index created.' and '0.04 seconds'. The bottom right corner of the interface displays 'Oracle APEX 23.2.4'.

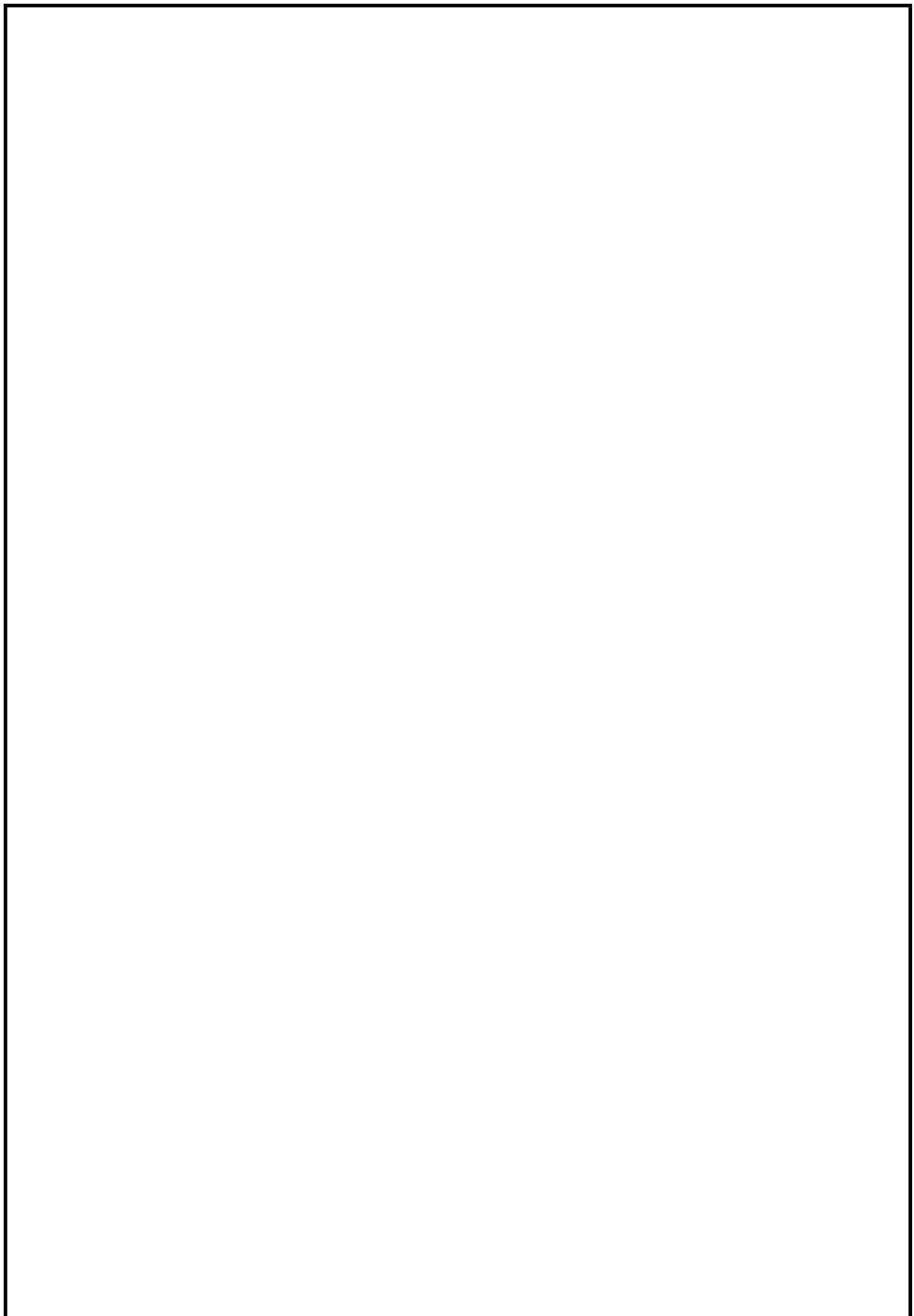
<https://apex.oracle.com/pls/apex/r/apex/sql-workshop/sql-workshop?...>

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:



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:

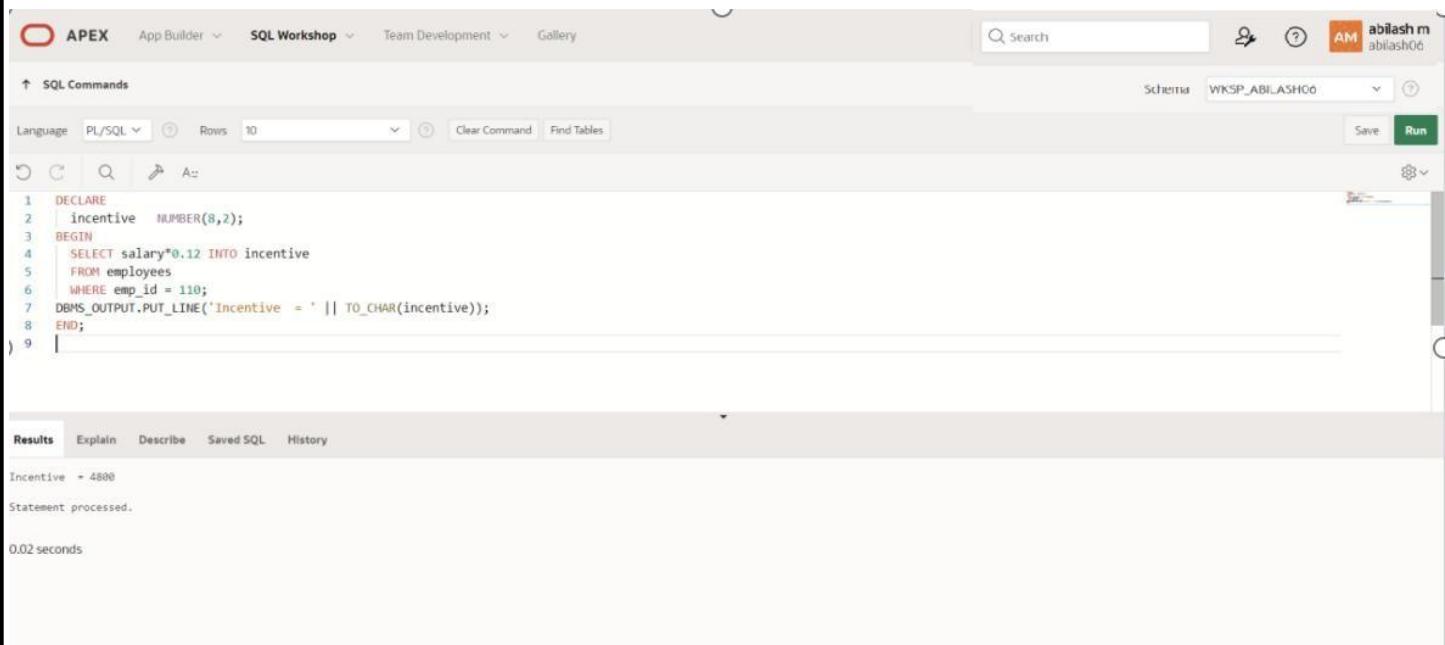
DATE:

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 SQL Workshop interface. In the top navigation bar, 'APEX' is selected. Below it, the tabs are 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery'. On the right side, there's a search bar, a user icon for 'abilash m abilash06', and buttons for 'Save' and 'Run'. The main area is titled 'SQL Commands'. It contains a code editor with the following PL/SQL block:

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

Below the code editor, there's a results panel. The 'Results' tab is active. The output shows:

```
Incentive = 4800
Statement processed.

0.02 seconds
```

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

QUERY:

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

OUTPUT:

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

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

In the results pane, there is an error message:

```
Error at line 4/23: ORA-06550: line 4, column 23:
PLS-00201: identifier 'Welcome' must be declared
ORA-06512: at "SYS.0MV_DBMS_SQL_APEX_230200", line 881
ORA-06550: line 4, column 1:
PL/SQL: Statement ignored
```

Below the error message, the rest of the code is shown:

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

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

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

In the results pane, there is an error message:

```
Error at line 4/23: ORA-06550: line 4, column 23:
PLS-00201: identifier 'Welcome' must be declared
ORA-06512: at "SYS.0MV_DBMS_SQL_APEX_230200", line 881
ORA-06550: line 4, column 1:
PL/SQL: Statement ignored
```

Below the error message, the rest of the code is shown:

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

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

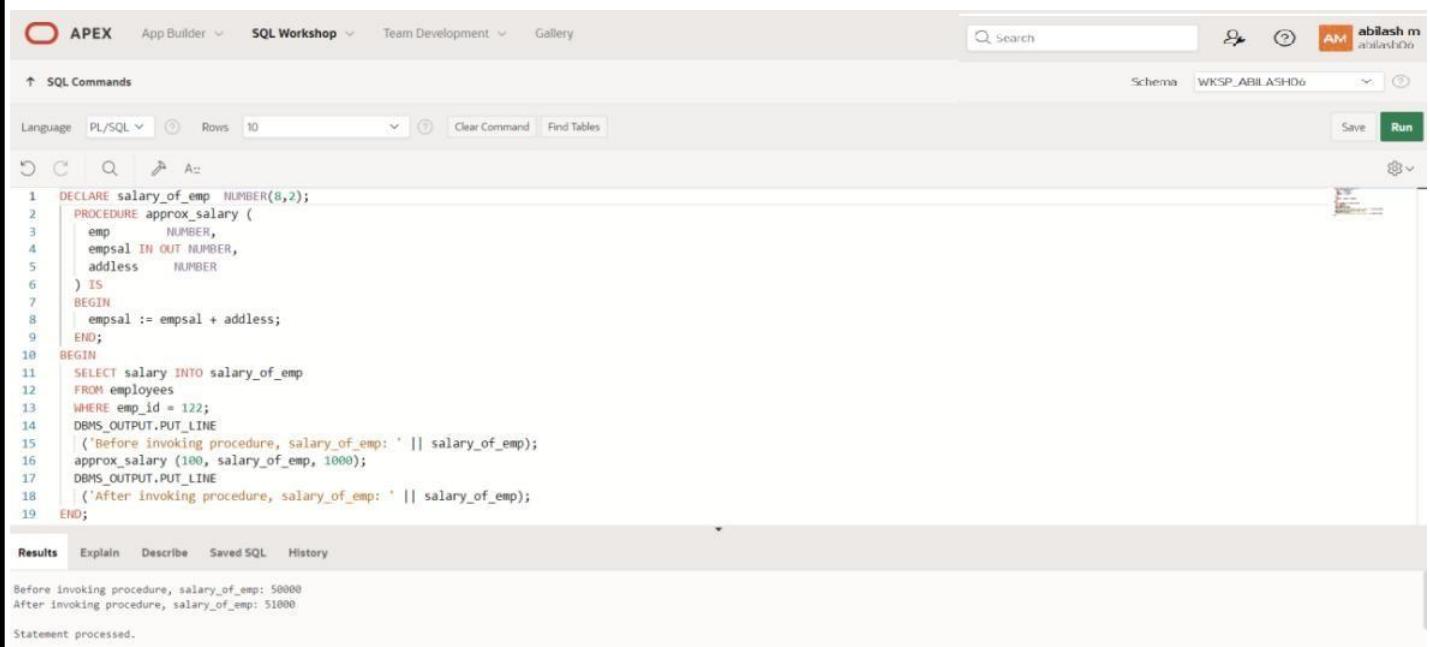
QUERY:

```
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 SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, and Gallery. The SQL Workshop tab is selected. The schema is set to WKSP_ABILASHD06. The main area displays the PL/SQL code with line numbers. The code declares a variable salary_of_emp, defines a procedure approx_salary with parameters emp, empsal (IN OUT), and addless, and then performs a SELECT into salary_of_emp from employees where employee_id = 122, followed by DBMS_OUTPUT.PUT_LINE statements for before and after invoking the procedure. The code ends with an END; statement and a slash (/). Below the code, the Results tab is active, showing the output of the executed code. The output consists of two lines: 'Before invoking procedure, salary_of_emp: 50000' and 'After invoking procedure, salary_of_emp: 51000'. A note at the bottom states 'Statement processed.'

```
1  DECLARE salary_of_emp NUMBER(8,2);
2  PROCEDURE approx_salary (
3    emp    NUMBER,
4    empsal IN OUT NUMBER,
5    addless  NUMBER
6  ) IS
7  BEGIN
8    empsal := empsal + addless;
9  END;
10 END;
11 /
12
13
14
15
16
17
18
19
```

Results Explain Describe Saved SQL History

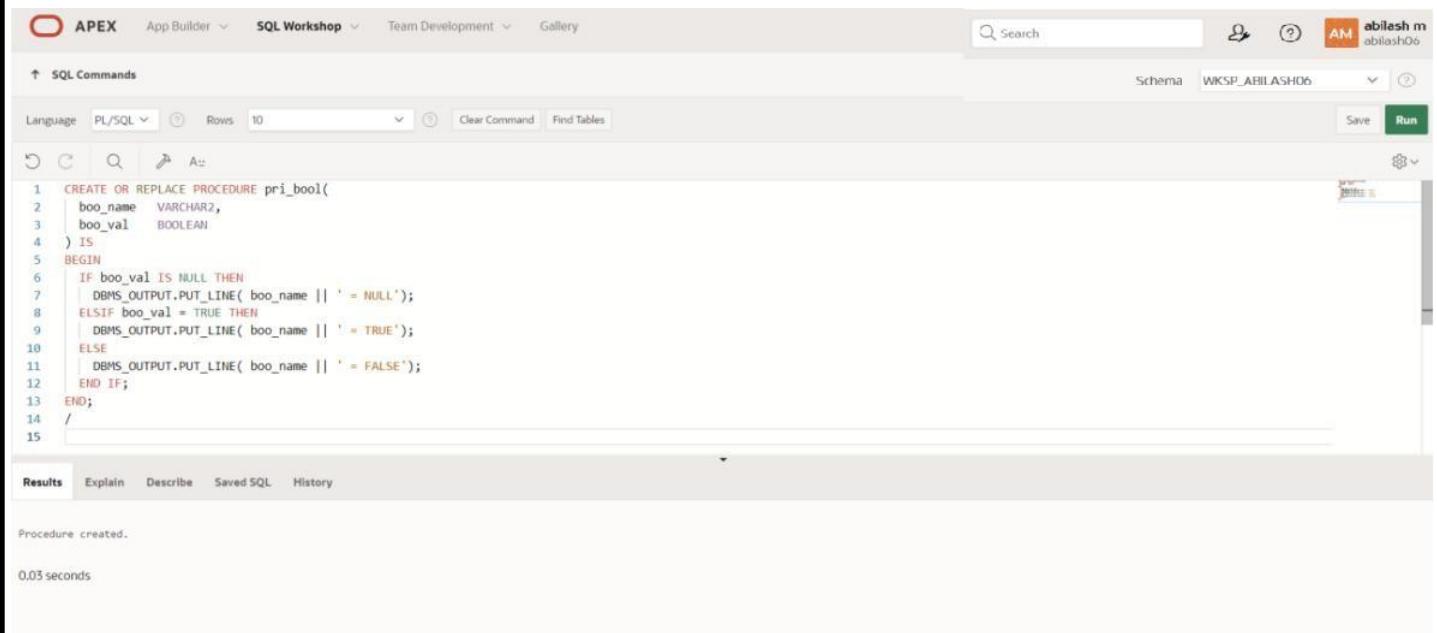
Before invoking procedure, salary_of_emp: 50000
After invoking procedure, salary_of_emp: 51000
Statement processed.

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 SQL Workshop interface. In the top navigation bar, 'APEX' is selected. The main area displays the PL/SQL code for the 'pri_bool' procedure. The code is identical to the one provided in the query section. Below the code, the 'Results' tab is active, showing the message 'Procedure created.' and a execution time of '0.03 seconds'. The schema is listed as 'WKSP_ABILASH06'.

```
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 /
```

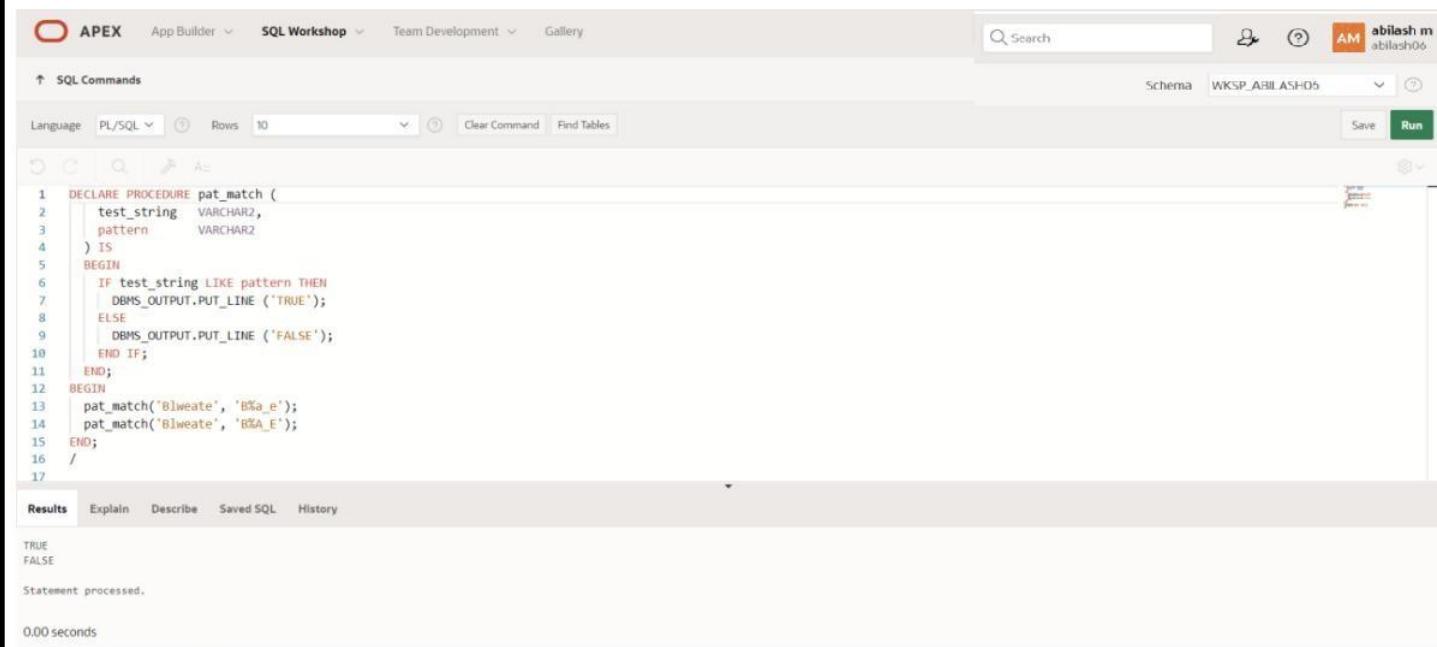
Procedure created.
0.03 seconds

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

QUERY:

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

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there's a search bar, a user icon for 'abilash m', and session information 'WKSP_ABILASH05'. The main workspace displays the PL/SQL code from the previous block. Below the code, the 'Results' tab is selected, showing the output: 'TRUE' and 'FALSE'. The status bar at the bottom indicates 'Statement processed.' and '0.00 seconds'.

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

Results Explain Describe Saved SQL History

TRUE
FALSE

Statement processed.
0.00 seconds

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

QUERY:

DECLARE

num_small NUMBER := 8;

num_large NUMBER := 5;

num_temp NUMBER;

BEGIN

IF num_small > num_large THEN

num_temp := num_small;

num_small := num_large;

num_large := num_temp;

END IF;

DBMS_OUTPUT.PUT_LINE ('num_small = '||num_small);

DBMS_OUTPUT.PUT_LINE ('num_large = '||num_large);

END;

/

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery'. The 'SQL Workshop' tab is active. The main area is titled 'SQL Commands' with a 'Save' and 'Run' button. The code editor contains the provided PL/SQL block. The results tab shows the output: 'num_small = 5' and 'num_large = 8'. The status bar at the bottom indicates 'Statement processed.' and '0.01 seconds'.

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

Results Explain Describe Saved SQL History

num_small = 5
num_large = 8

Statement processed.

0.01 seconds

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

QUERY:

DECLARE

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

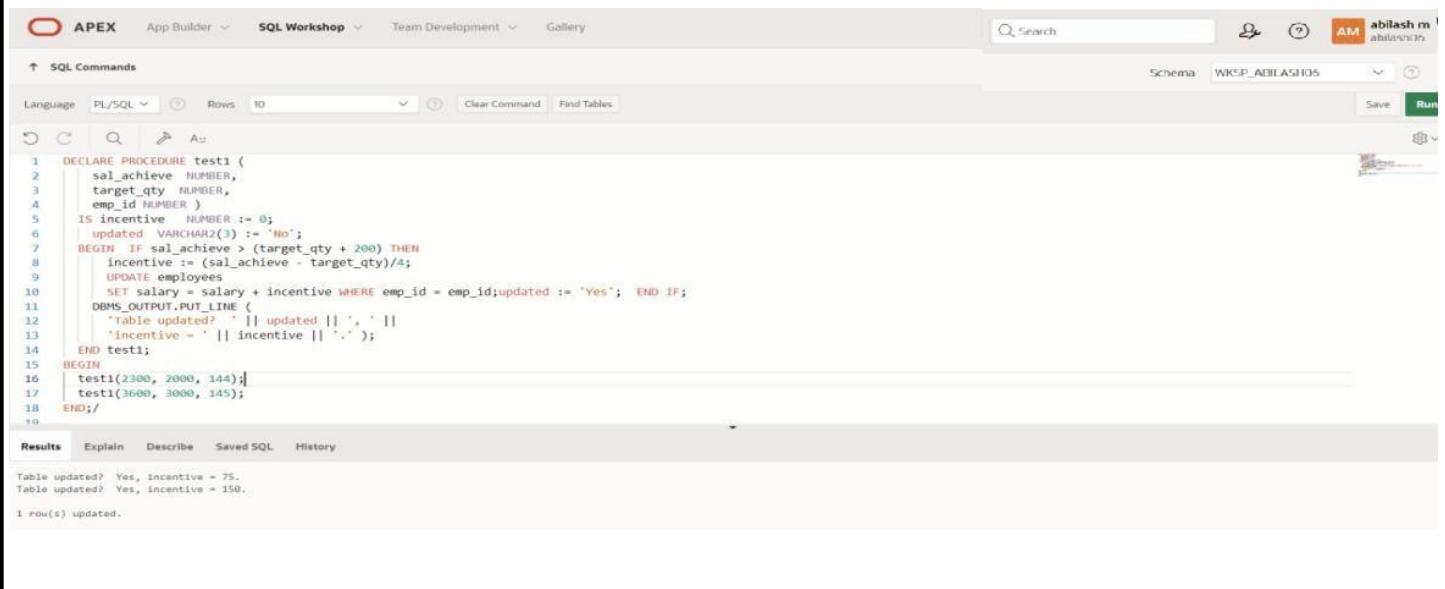
BEGIN

```
test1(2300, 2000, 144);
test1(3600, 3000, 145);
```

END;

/

OUTPUT:



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

- Toolbar:** Includes icons for APEX, App Builder, SQL Workshop, Team Development, and Gallery.
- Header:** Shows the schema as WKSP_AELIASHOS and the user as AM abilashm@192.168.1.103.
- Search Bar:** A search input field with placeholder text "Search".
- Language Selector:** Set to PL/SQL.
- Run Button:** A green "Run" button.
- Code Area:** Displays the PL/SQL code for the `test1` procedure and its execution block.
- Results Tab:** Active tab, showing the output of the procedure execution.
- Output Content:**

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

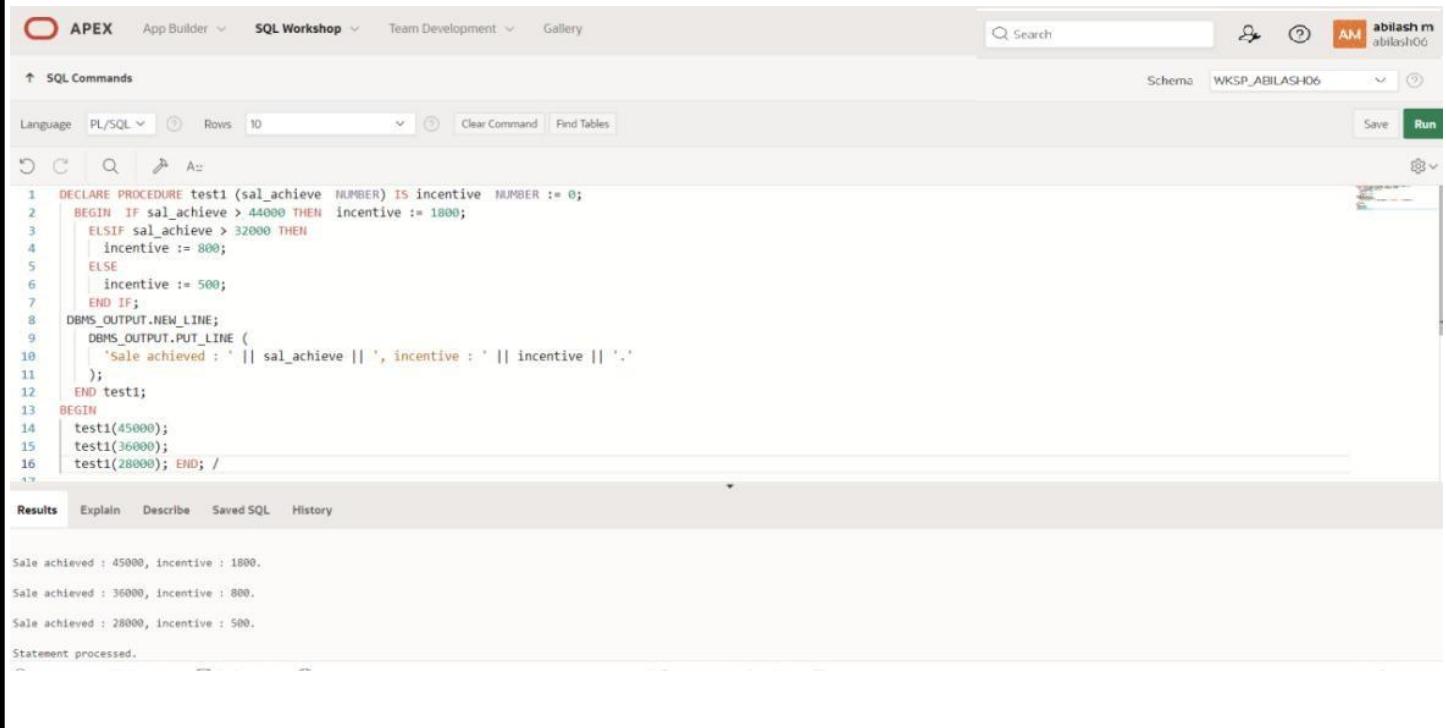
1 row(s) updated.
```

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

QUERY:

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

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, and Gallery. The right side shows a user profile for 'abilash m' and the schema 'WKSP_ABILASH06'. The main area is titled 'SQL Commands' and contains the PL/SQL code from the previous section. The code is numbered 1 to 16. Below the code, there are tabs for Results, Explain, Describe, Saved SQL, and History. The Results tab is selected, displaying the output of the executed code: 'Sale achieved : 45000, incentive : 1800.', 'Sale achieved : 36000, incentive : 800.', 'Sale achieved : 28000, incentive : 500.', and 'Statement processed.'

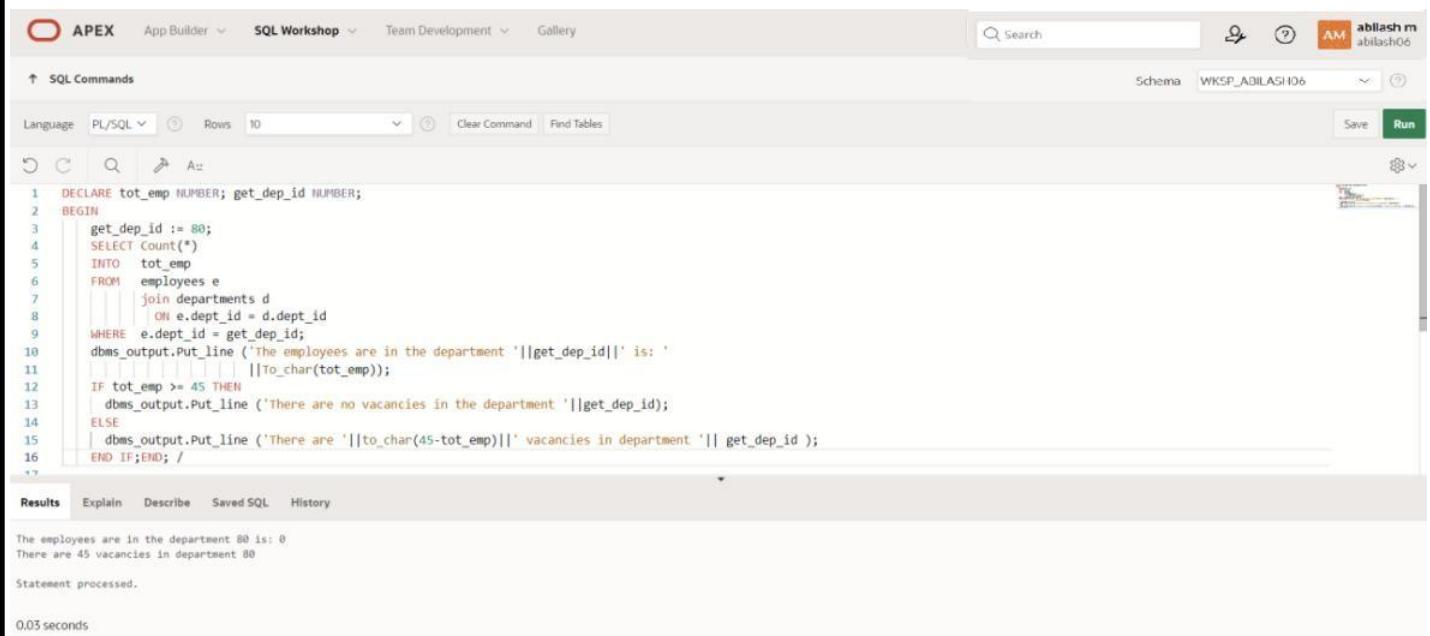
9.) Write a PL/SQL program to count number of employees in department 50 and check whether this department have any vacancies or not. There are 45 vacancies in this department.

QUERY:

```
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 SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, and Gallery. The SQL Workshop tab is selected. The schema is set to WKSP_ABILASH106. The code area contains the PL/SQL block provided above. The results pane at the bottom displays the output of the program execution.

```
1  DECLARE tot_emp NUMBER; get_dep_id NUMBER;
2  BEGIN
3      get_dep_id := 80;
4      SELECT Count(*)
5          INTO tot_emp
6          FROM employees e
7              join departments d
8                  ON e.department_id = d.department_id
9      WHERE e.department_id = get_dep_id;
10     dbms_output.Put_line ('The employees are in the department'||get_dep_id||' is: '
11         ||To_char(tot_emp));
12     IF tot_emp >= 45 THEN
13         dbms_output.Put_line ('There are no vacancies in the department'||get_dep_id);
14     ELSE
15         dbms_output.Put_line ('There are'||to_char(45-tot_emp)||' vacancies in department'||get_dep_id );
16     END IF;END; /
```

The results pane shows the following output:

```
The employees are in the department 80 is: 0
There are 45 vacancies in department 80

Statement processed.

0.03 seconds
```

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

QUERY:

DECLARE

```
tot_emp NUMBER;  
get_dep_id NUMBER;
```

BEGIN

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

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

```
IF tot_emp >= 45 THEN
```

```
dbms_output.Put_line ('There are no vacancies in the department'||get_dep_id);
```

```
ELSE
```

```
dbms_output.Put_line ('There are'||to_char(45-tot_emp)||' vacancies in department'||  
get_dep_id );
```

```
END IF;
```

```
END;
```

```
/
```

OUTPUT:

The screenshot shows the Oracle 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 'abilash m' and the schema 'WKSP_ABILASH06'. The main area is titled 'SQL Commands' and contains the following PL/SQL code:

```
1 DECLARE tot_emp NUMBER; get_dep_id NUMBER;  
2 BEGIN get_dep_id := 80;  
3 SELECT Count(*) INTO tot_emp  
4 FROM employees e join departments d ON e.dept_id = d.dept_id  
5 WHERE e.dept_id = get_dep_id;  
6 dbms_output.Put_line ('The employees are in the department'||get_dep_id||' is: '  
||To_char(tot_emp));  
7  
8 IF tot_emp >= 45 THEN  
9 dbms_output.Put_line ('There are no vacancies in the department'||get_dep_id);  
10 ELSE  
11 dbms_output.Put_line ('There are'||to_char(45-tot_emp)||' vacancies in department'|| get_dep_id );  
12 END IF;  
13  
14 END;  
15 /  
16
```

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

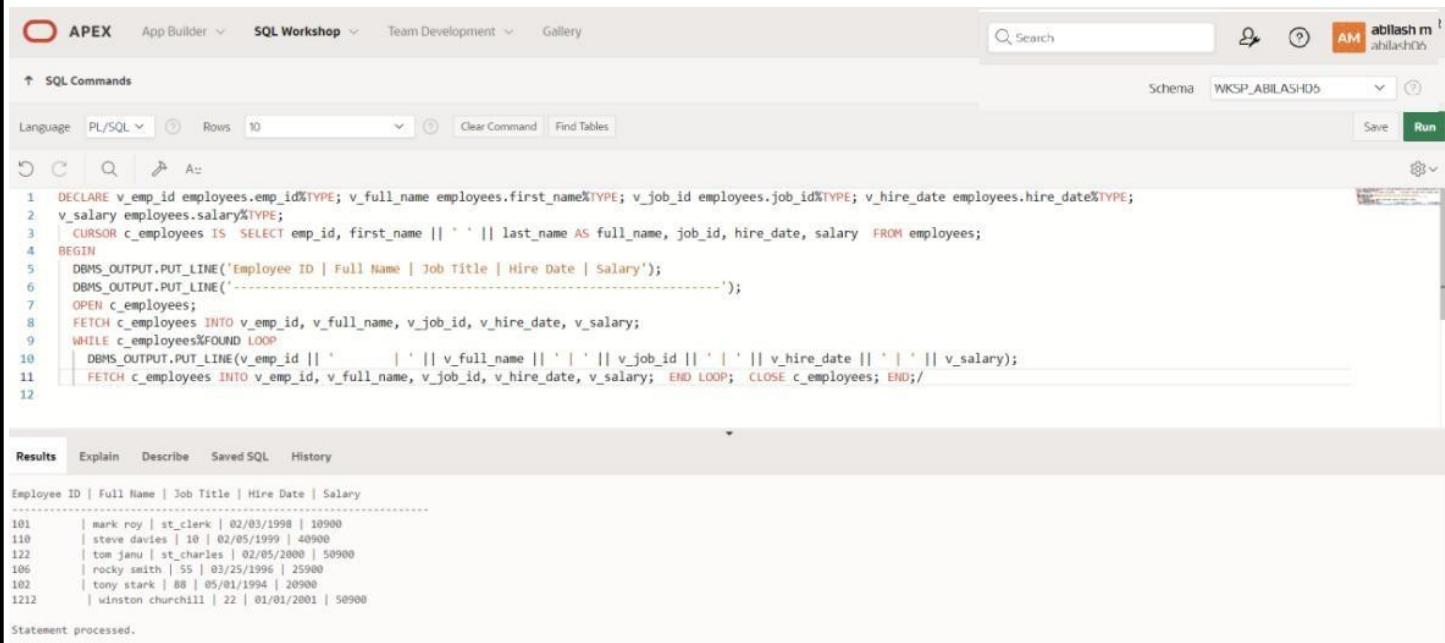
```
The employees are in the department 80 is: 80  
There are 45 vacancies in department 80  
Statement processed.  
0.00 seconds
```

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:



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 'abilash m' and the schema 'WKSP_ABILASH05'. The main area has tabs for 'SQL Commands' (selected) and 'Results'. In the 'SQL Commands' tab, the PL/SQL code is pasted. In the 'Results' tab, the output is displayed as a table:

Employee ID	Full Name	Job Title	Hire Date	Salary
101	mark roy	st_clerk	02/03/1998	10900
110	steve davies	10	02/05/1999	40900
122	tom janu	st_charles	02/05/2000	50900
106	rocky smith	55	03/25/1996	25900
102	tony stark	88	05/01/1994	20900
1212	winston churchill	22	01/01/2001	50900

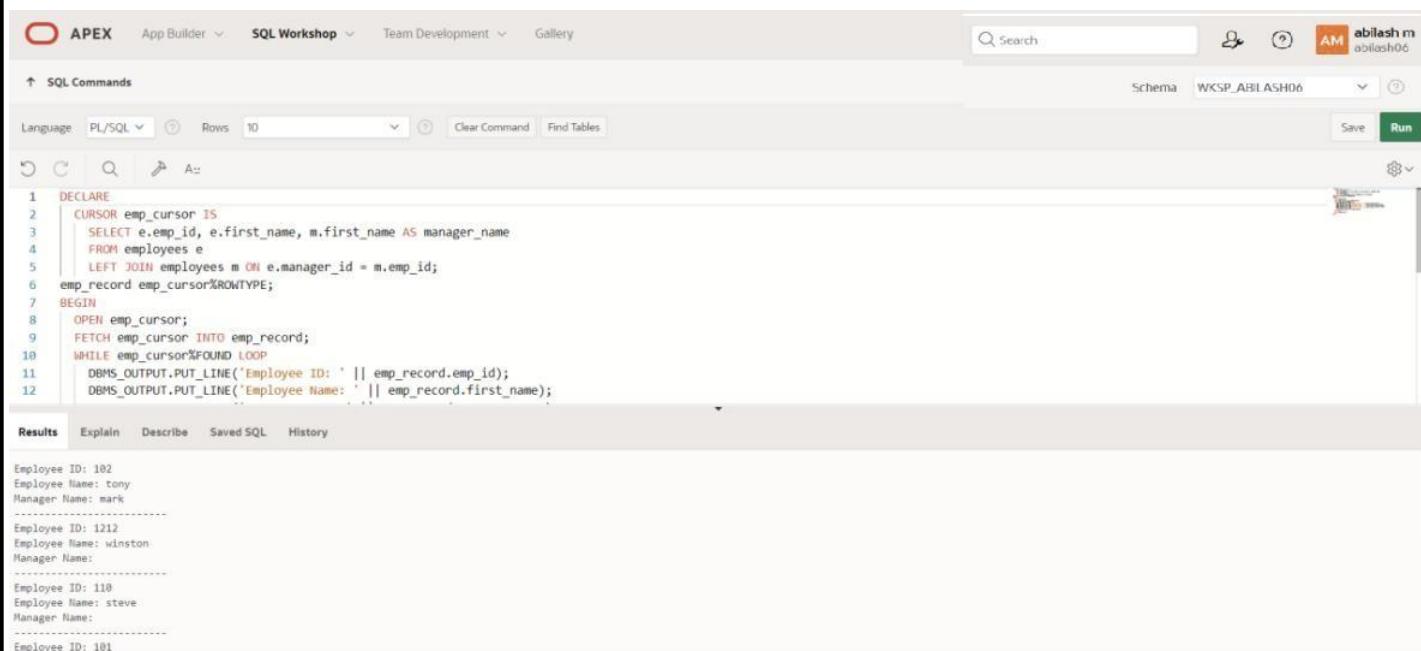
Below the table, a message says 'Statement processed.'

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:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. The top right shows a user icon 'abilash m' and the schema 'WKSP_ABILASH06'. The main area has tabs for 'SQL Commands' (selected), 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History'. The 'Results' tab displays the output of the executed PL/SQL block. The output consists of several lines of text, each starting with 'Employee ID:' followed by an employee ID, 'Employee Name:' followed by the employee's first name, and 'Manager Name:' followed by the manager's name. There are three distinct sections separated by dashed lines, each representing a different employee record fetched from the cursor.

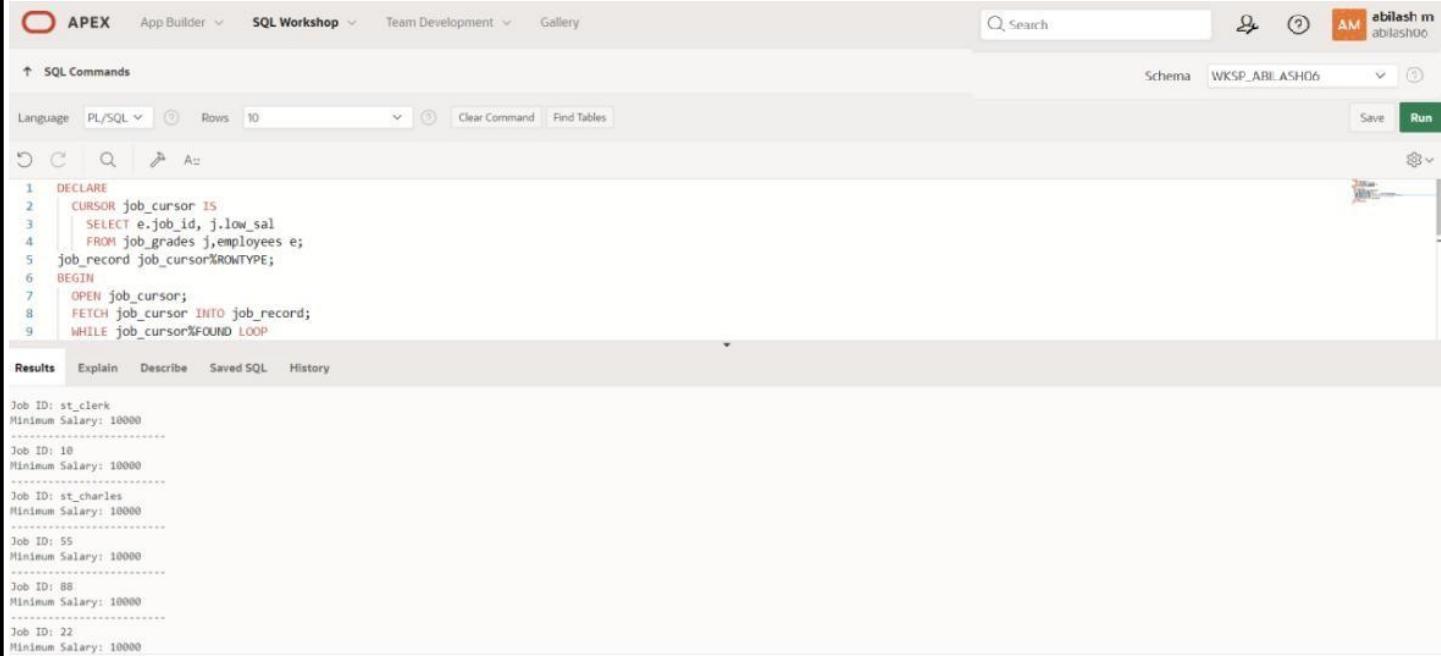
```
Employee ID: 102
Employee Name: tony
Manager Name: mark
-----
Employee ID: 1211
Employee Name: winston
Manager Name:
-----
Employee ID: 118
Employee Name: steve
Manager Name:
-----
Employee ID: 101
```

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 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 'abilash m abilash06'. The main area is titled 'SQL Commands' with tabs for Language (PL/SQL selected), Rows (10), Clear Command, Find Tables, Schema (WKSP_ABILASH06), Save, and Run.

The code area contains the PL/SQL block from the previous section. The results tab shows the output of the program, which lists job IDs and their minimum salaries:

```
Job ID: st_clerk
Minimum Salary: 10000
-----
Job ID: 10
Minimum Salary: 10000
-----
Job ID: st_charles
Minimum Salary: 10000
-----
Job ID: 55
Minimum Salary: 10000
-----
Job ID: 88
Minimum Salary: 10000
-----
Job ID: 22
Minimum Salary: 10000
```

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:



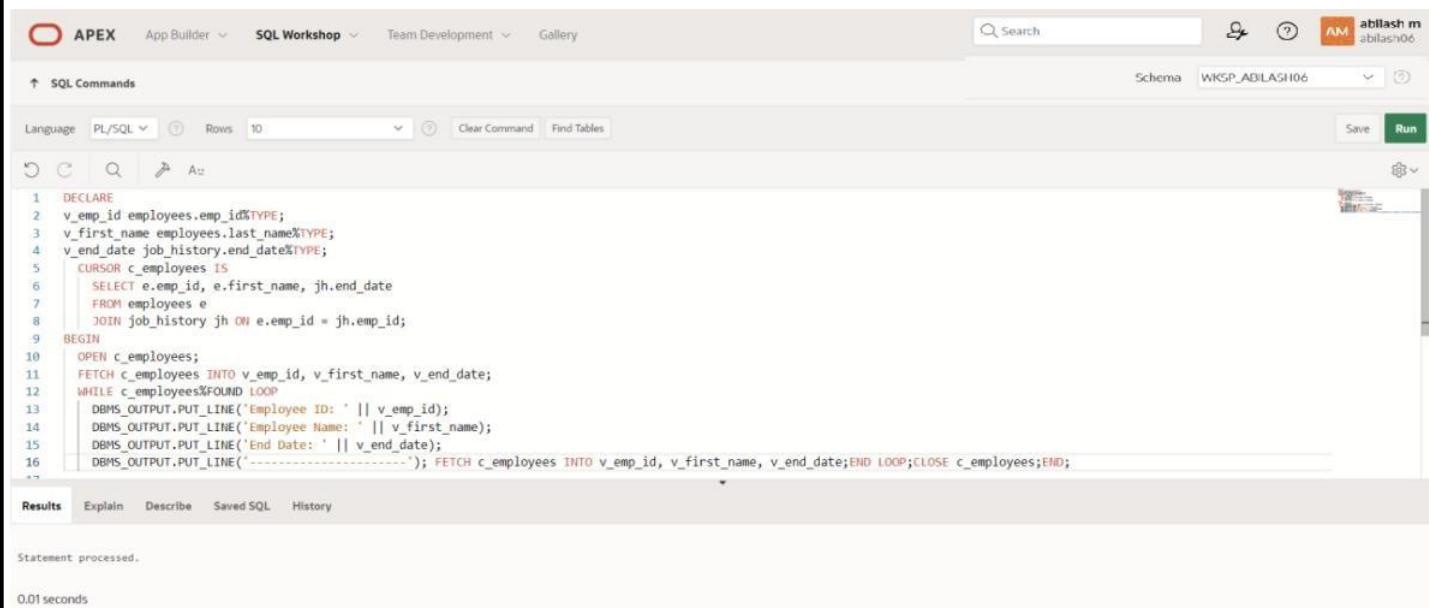
The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery'. The right side shows a user profile 'abilash06' and a workspace 'WKSP_ABILASH06'. The main area has tabs for 'SQL Commands' and 'Results'. The SQL Commands tab contains the PL/SQL code provided above. The Results tab displays the output of the code execution, which is empty in this view.

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 SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery'. The right side shows a user profile 'abilash m' and the schema 'WKSP_ABILASH06'. The main area is titled 'SQL Commands' with a 'Language' dropdown set to 'PL/SQL'. Below the title are buttons for 'Rows' (set to 10), 'Clear Command', and 'Find Tables'. The code area contains the PL/SQL block from above. The bottom section is labeled 'Results' and displays the output: 'Statement processed.' and '0.01 seconds'.

```
1  DECLARE
2    v_emp_id employees.emp_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.emp_id, e.first_name, jh.end_date
7        FROM employees e
8       JOIN job_history jh ON e.emp_id = jh.emp_id;
9 BEGIN
10  OPEN c_employees;
11  FETCH c_employees INTO v_emp_id, v_first_name, v_end_date;
12  WHILE c_employees%FOUND LOOP
13    DBMS_OUTPUT.PUT_LINE('Employee ID: ' || v_emp_id);
14    DBMS_OUTPUT.PUT_LINE('Employee Name: ' || v_first_name);
15    DBMS_OUTPUT.PUT_LINE('End Date: ' || v_end_date);
16    DBMS_OUTPUT.PUT_LINE('-----'); FETCH c_employees INTO v_emp_id, v_first_name, v_end_date;END LOOP;CLOSE c_employees;END;
```

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. In the top navigation bar, 'APEX' is selected. The main area displays a PL/SQL block:

```
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;
```

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

```
120
Statement processed.

0.01 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 SQL Workshop interface. In the top navigation bar, 'APEX' is selected, followed by 'App Builder', 'SQL Workshop' (which is highlighted in blue), 'Team Development', and 'Gallery'. On the right side, there's a search bar, user profile, and a schema dropdown set to 'WKSP_ABILASH06'. Below the navigation, the 'SQL Commands' tab is active, showing a code editor with PL/SQL code. The code creates a procedure named 'get_book_info' that takes a book ID and returns the title, author, and year published. It includes a cursor loop and an exception handling block for 'NO_DATA_FOUND'. The results tab shows the message 'Procedure created.' and a execution time of '0.04 seconds'.

```
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;
```

Procedure created.
0.04 seconds

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

RESULT:

TRIGGER

EX_NO: 18

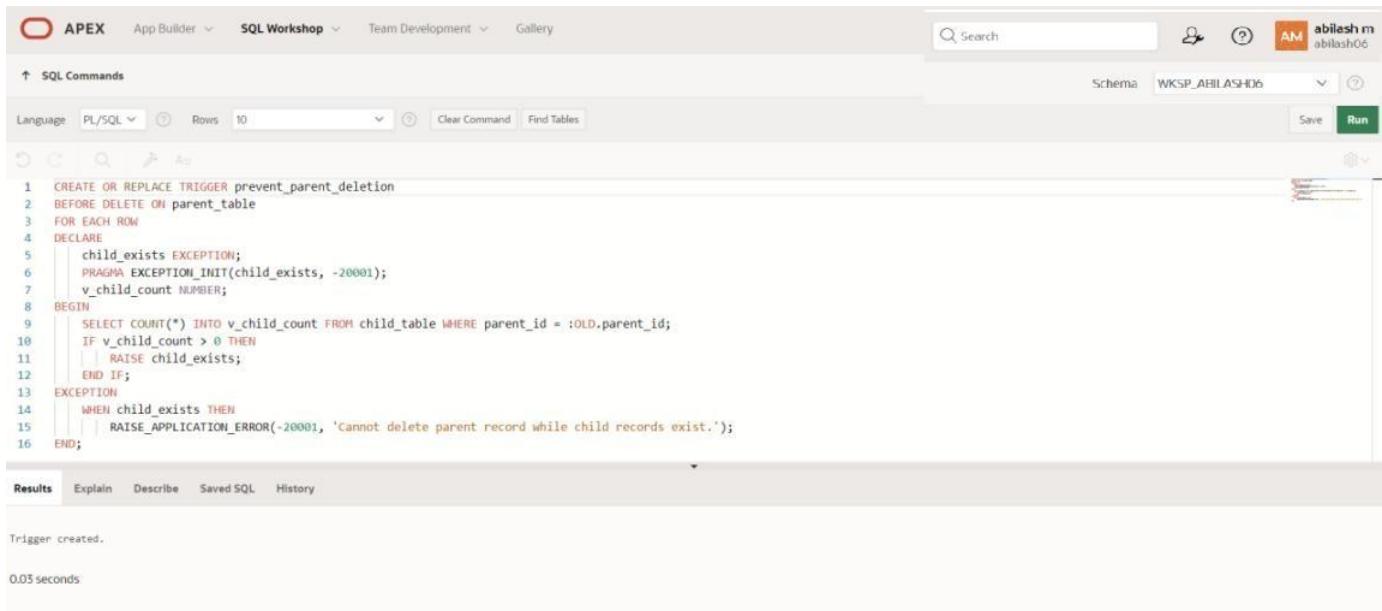
DATE:

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

QUERY:

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

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. In the top navigation bar, 'APEX' is selected. Below it, the tabs include 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery'. On the right side, there's a user icon and the text 'abilash m abilash06'. The main workspace is titled 'SQL Commands'. It shows the PL/SQL code for the trigger. At the bottom, the 'Results' tab is active, displaying the message 'Trigger created.' and '0.03 seconds'.

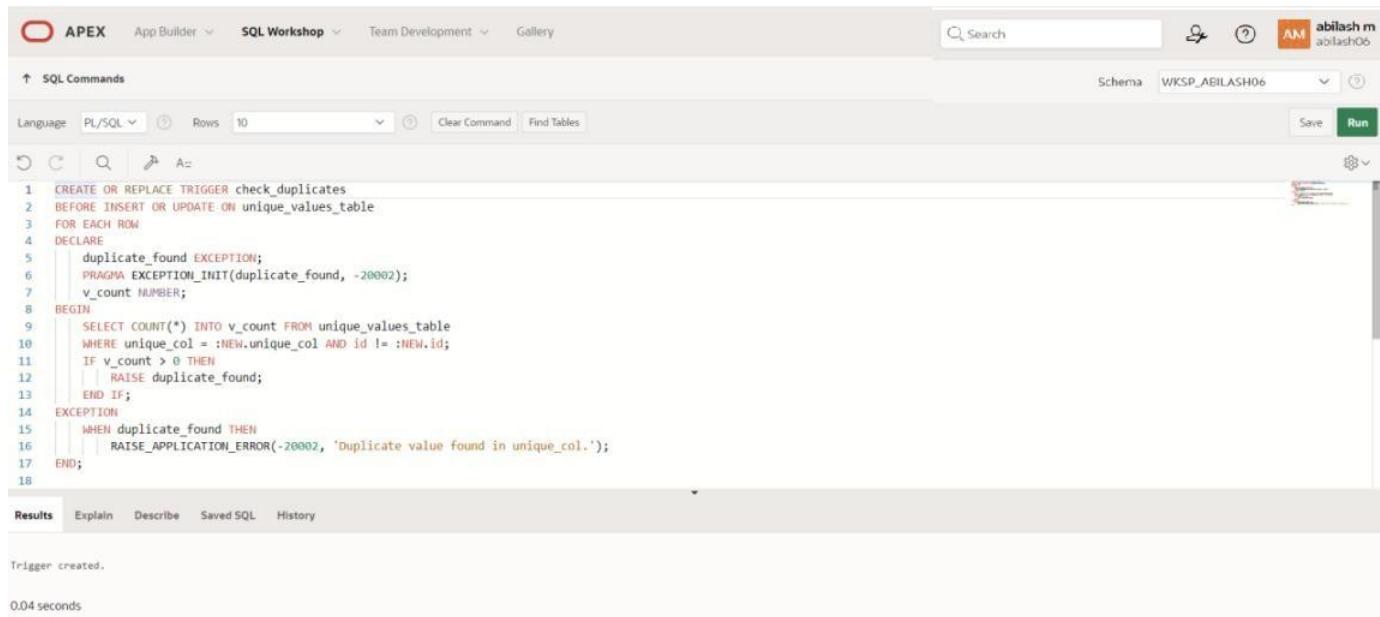
```
1 CREATE OR REPLACE TRIGGER prevent_parent_deletion
2 BEFORE DELETE ON parent_table
3 FOR EACH ROW
4 DECLARE
5     child_exists EXCEPTION;
6     PRAGMA EXCEPTION_INIT(child_exists, -20001);
7     v_child_count NUMBER;
8 BEGIN
9     SELECT COUNT(*) INTO v_child_count FROM child_table WHERE parent_id = :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;
```

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

QUERY:

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

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. In the top navigation bar, 'APEX' is selected. Below it, the tabs 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery' are visible. On the right side, there's a user profile icon for 'abilash m abilash06'. The main area is titled 'SQL Commands'. A toolbar below the title includes icons for Undo, Redo, Search, and Run. The command window contains the PL/SQL code for creating the 'check_duplicates' trigger. The code is highlighted in green. The status bar at the bottom indicates 'Trigger created.' and '0.04 seconds'.

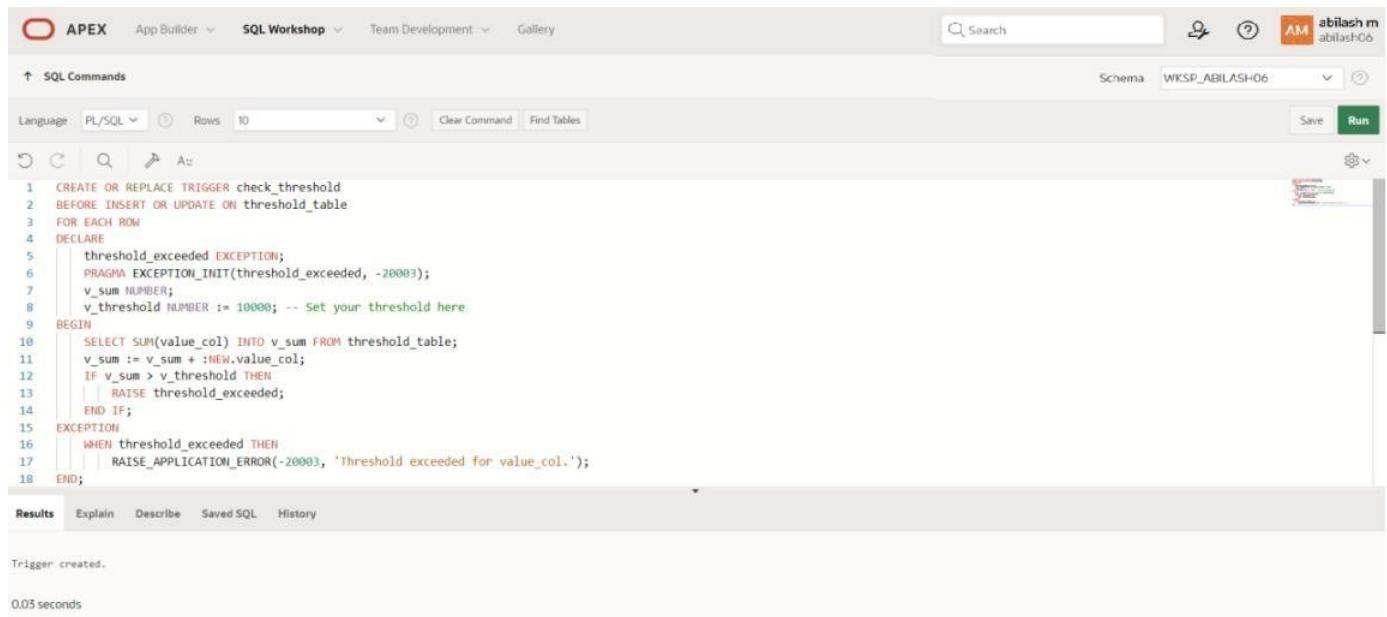
```
1 CREATE OR REPLACE TRIGGER check_duplicates
2 BEFORE INSERT OR UPDATE ON unique_values_table
3 FOR EACH ROW
4 DECLARE
5     duplicate_found EXCEPTION;
6     PRAGMA EXCEPTION_INIT(duplicate_found, -20002);
7     v_count NUMBER;
8 BEGIN
9     SELECT COUNT(*) INTO v_count FROM unique_values_table
10    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;
18
```

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

QUERY:

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

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there are tabs for Schema (WKSP_ABILASH06), Save, and Run. The main area is titled 'SQL Commands' and contains the PL/SQL code for the trigger. The code is numbered from 1 to 18. The 'Results' tab at the bottom shows the output: 'Trigger created.' and '0.03 seconds'.

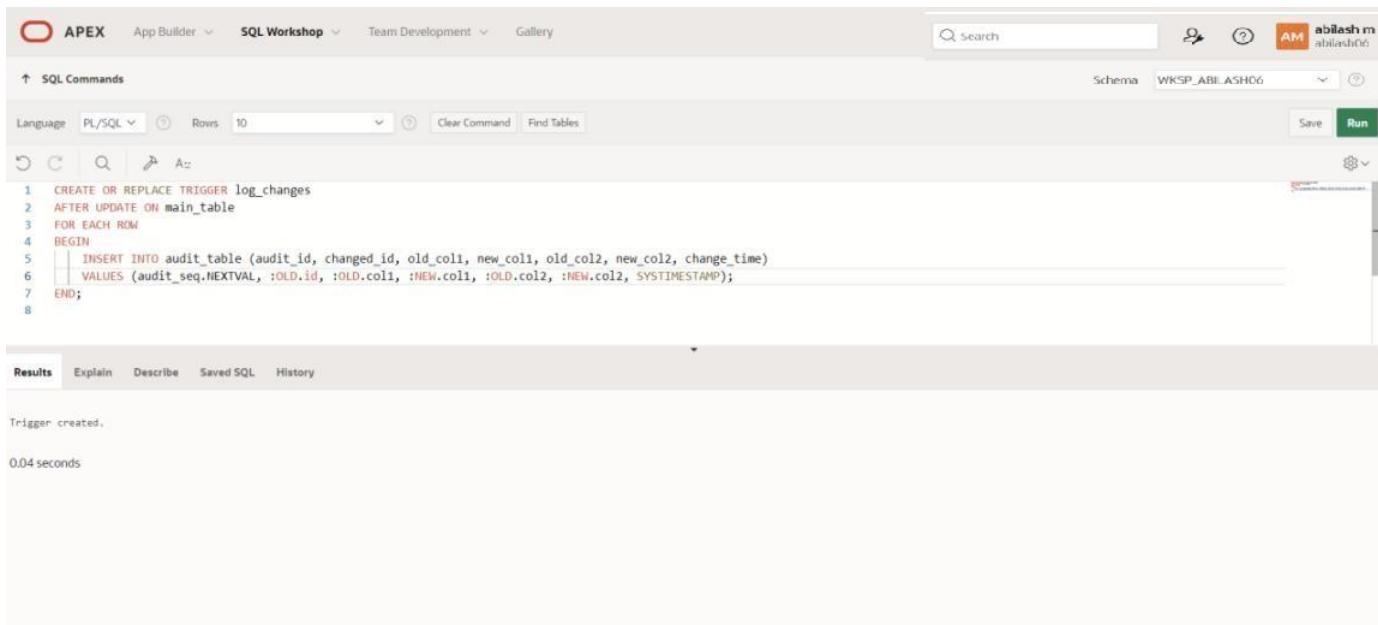
```
1 CREATE OR REPLACE TRIGGER check_threshold
2 BEFORE INSERT OR UPDATE ON threshold_table
3 FOR EACH ROW
4 DECLARE
5     threshold_exceeded EXCEPTION;
6     PRAGMA EXCEPTION_INIT(threshold_exceeded, -20003);
7     v_sum NUMBER;
8     v_threshold NUMBER := 10000; -- Set your threshold here
9 BEGIN
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;
```

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

QUERY:

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

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. In the top navigation bar, 'APEX' is selected. Below it, the tabs 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery' are visible. On the right side, there's a user profile icon for 'abilash m' and a schema dropdown set to 'WKSP_ABILASH06'. The main area is titled 'SQL Commands' and contains a code editor. The code editor has a toolbar with 'Language' (set to 'PL/SQL'), 'Rows' (set to 10), 'Clear Command', 'Find Tables', 'Save', and 'Run' buttons. The code itself is the trigger definition provided above. Below the code editor is a results panel with tabs for 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History'. The 'Results' tab is active, displaying the message 'Trigger created.' and a timestamp '0.04 seconds'.

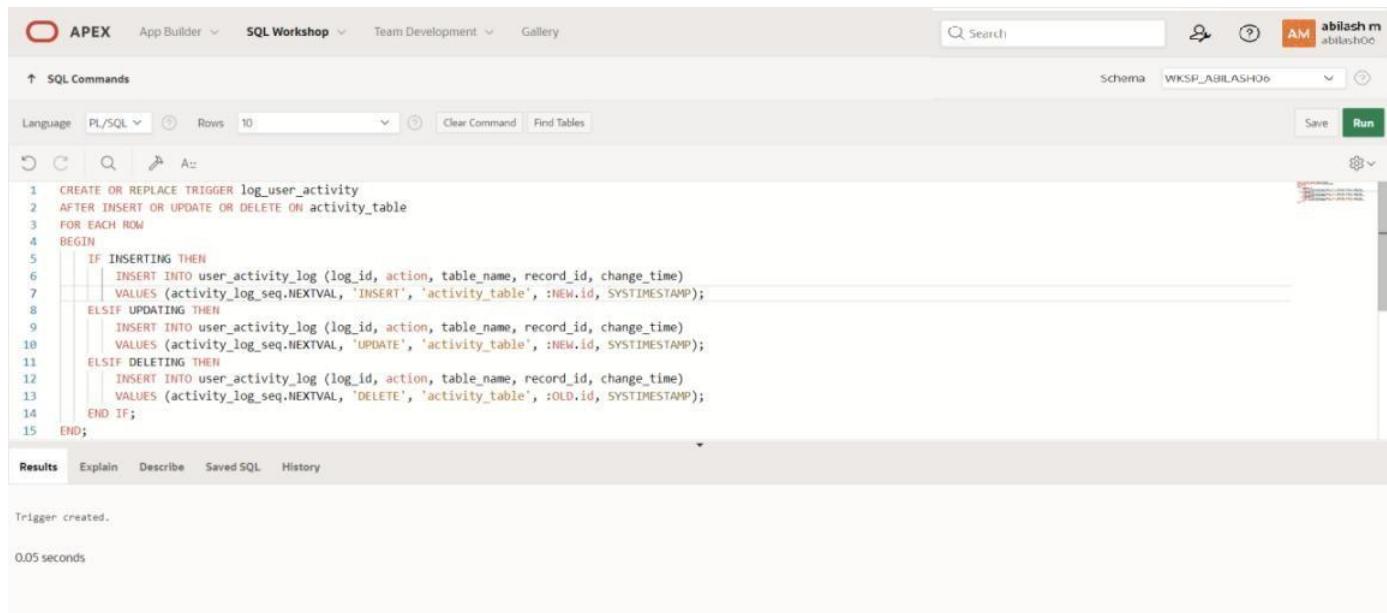
```
1 CREATE OR REPLACE TRIGGER log_changes
2 AFTER UPDATE ON main_table
3 FOR EACH ROW
4 BEGIN
5     INSERT INTO audit_table (audit_id, changed_id, old_col1, new_col1, old_col2, new_col2,
6 change_time)
7     VALUES (audit_seq.NEXTVAL, :OLD.id, :OLD.col1, :NEW.col1, :OLD.col2, :NEW.col2,
8 SYSTIMESTAMP);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

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

QUERY:

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

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. In the top navigation bar, 'APEX' is selected. The main area displays the PL/SQL code for the trigger. The code is highlighted in red and blue, indicating syntax. The code itself is identical to the one provided in the question. Below the code, the 'Results' tab is active, showing the message 'Trigger created.' and a execution time of '0.05 seconds'. The schema is set to 'WKSP_ABILASH06'.

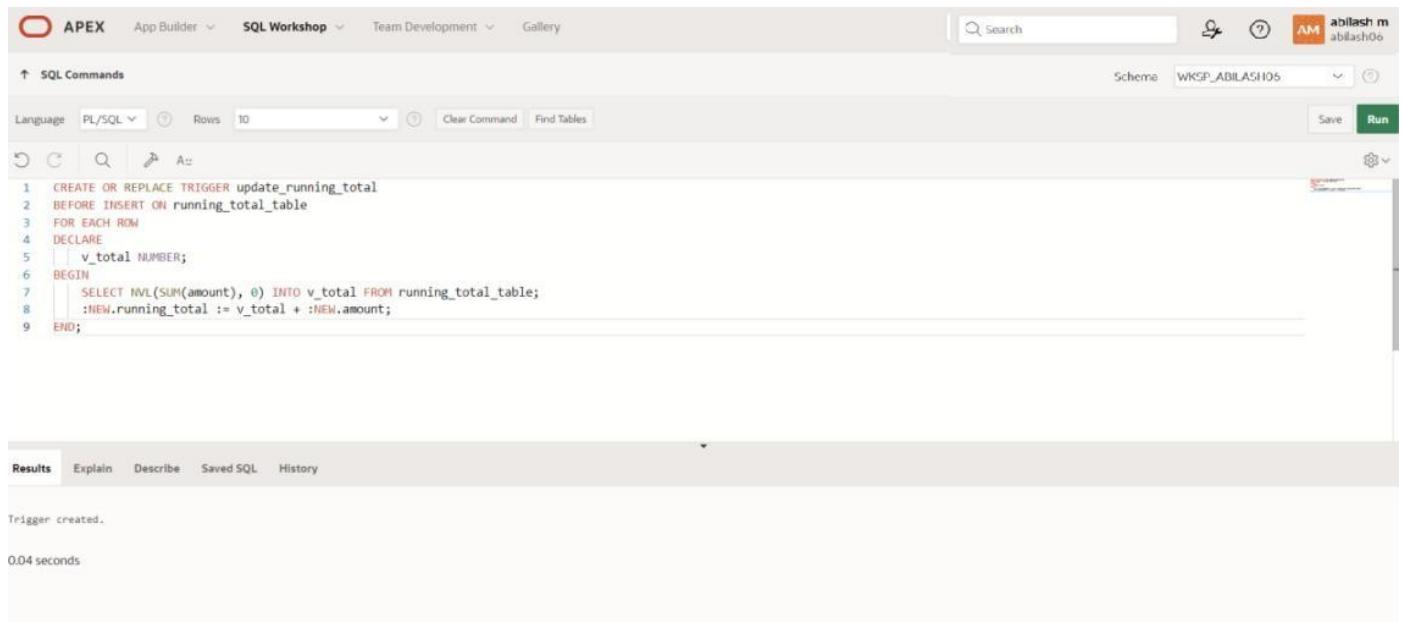
```
1 CREATE OR REPLACE TRIGGER log_user_activity
2 AFTER INSERT OR UPDATE OR DELETE ON activity_table
3 FOR EACH ROW
4 BEGIN
5   IF INSERTING THEN
6     INSERT INTO user_activity_log (log_id, action, table_name, record_id, change_time)
7     VALUES (activity_log_seq.NEXTVAL, 'INSERT', 'activity_table', :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;
```

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

QUERY:

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

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. In the top navigation bar, 'APEX' is selected. The main area displays the PL/SQL code for the trigger. The code is as follows:

```
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;
```

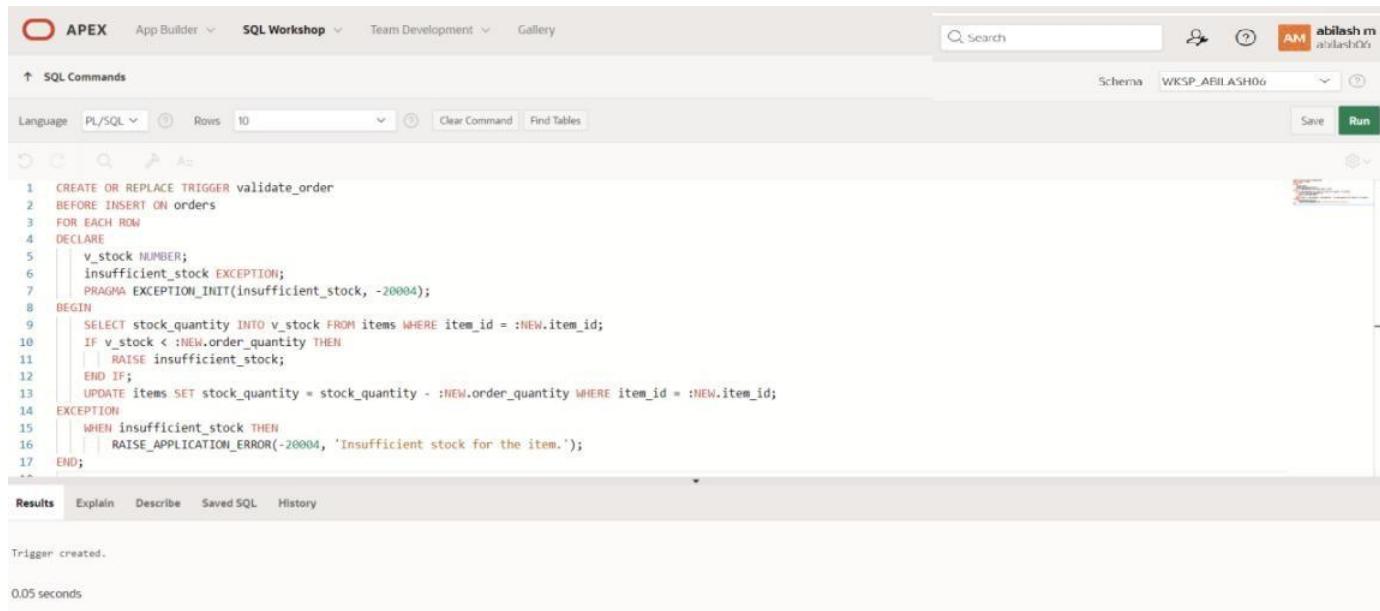
Below the code, the 'Results' tab is active, showing the message "Trigger created." and a execution time of "0.04 seconds".

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

QUERY:

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

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. In the top navigation bar, 'APEX' is selected. Below it, the tabs are 'App Builder', 'SQL Workshop' (which is active), 'Team Development', and 'Gallery'. On the right side, there's a user profile for 'abilash m' and a workspace named 'WKSP_ABILASH06'. The main area is titled 'SQL Commands'. At the top left of this area, there are buttons for 'Language' (set to 'PL/SQL'), 'Rows' (set to 10), and 'Clear Command'. To the right are 'Save' and 'Run' buttons. Below these buttons, there are links for 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History'. The SQL command itself is displayed in the main text area, starting with 'CREATE OR REPLACE TRIGGER validate_order'. The command is fully visible and includes the logic for checking stock and raising an exception if不足.

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 a MongoDB shell interface. At the top, there is a search bar labeled "Enter a title...". Below it, a dropdown menu is set to "MongoDB" and a "Run" button is visible. The main area contains a command line with the following text:
1: {
 \$or: [{ name: /^Wil/ }, { cuisine: { \$nin: ['American', 'Chinese'] } }] ,
 { restaurant_id: 1, name: 1, borough: 1, cuisine: 1 } };
Output
mycompiler_mongodb>
mycompiler_mongodb>
[Execution complete with exit code 0]

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

QUERY:

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

OUTPUT:

myCompiler

English Recent

Enter a title...

MongoDB

Output

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

```
mycompiler_mongodb>
mycompiler_mongodb>

[Execution complete with exit code 0]
```

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

QUERY:

```
db.restaurants.find( {"grades.1.grade": "A", "grades.1.score": 9, "grades.1.date": ISODate("2014-08-11T00:00:00Z") }, { restaurant_id: 1, name: 1, grades: 1 } );
```

OUTPUT:

Enter a title...

MongoDB

Output

```
1: "A", "grades.1.score": 9, "grades.1.date": ISODate("2014-08-11T00:00:00Z") }, { restaurant_id: 1, name: 1, grades: 1 }; Output
```

```
mycompiler_mongodb>
mycompiler_mongodb>

[Execution complete with exit code 0]
```

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 a MongoDB query editor interface. At the top, there is a search bar labeled "Enter a title..." and a dropdown menu set to "MongoDB". Below the search bar are two buttons: a blue one with a gear icon and a red one with a trash bin icon. To the right is a green "Run" button with a play icon. The main area has a light blue header bar with the text "1{\$and : [{"address.coord.1": {\$gt : 42}}, {"address.coord.1": {\$lte : 52}}]}, {_id:0, restaurant_id:1, name:1, address:1} | Output". The output panel below shows the command and its execution results. The command is identical to the one above. The results show the prompt "mycompiler_mongodb>" followed by "mycompiler_mongodb>". Below that, a message indicates "[Execution complete with exit code 0]".

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 a MongoDB shell interface. In the top left, there is a search bar labeled "Enter a title...". Below it, a toolbar includes a MongoDB icon, a dropdown menu, and a help icon. On the far right of the toolbar are "Ctrl+" and a green "Run" button with a play icon. The main area contains a code input field with the following command:

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

To the right of the code input is a vertical "Output" panel. It displays the command prompt "mycompiler_mongodb>" followed by the output of the command:

```
mycompiler_mongodb>
mycompiler_mongodb>
```

At the bottom of the output panel, the message "[Execution complete with exit code 0]" is visible.

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 a MongoDB shell interface. In the top left, there is a search bar labeled "Enter a title...". Below it, a toolbar includes a MongoDB icon, a dropdown menu, and a help icon. On the far right of the toolbar are "Ctrl+" and a green "Run" button with a play icon. The main area contains a code input field with the following command:

```
1 ${$and : [{"address.coord.1": {$gt : 42}}, {"address.coord.1": {$lte : 52}}]}, { _id:0, restaurant_id:1, name:1, address:1}]]| Output
```

To the right of the code input is a vertical "Output" panel. It displays the command prompt "mycompiler_mongodb>" followed by the output of the command:

```
mycompiler_mongodb>
mycompiler_mongodb>
```

At the bottom of the output panel, the message "[Execution complete with exit code 0]" is visible.

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 MongoDB shell interface. On the left, there is an input field labeled "Enter a title..." and a toolbar with a MongoDB icon, a dropdown menu, and a help icon. Below the toolbar is a code editor containing the MongoDB query: `1 db.restaurants.find({}, { _id: 0 }).sort({ cuisine: 1, borough: -1 })`. On the right, there is an "Output" panel with the heading "mycompiler_mongodb>". The output shows the command being run and the response: `mycompiler_mongodb>`, followed by "[Execution complete with exit code 0]". A green "Run" button is located at the top right of the output panel.

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 MongoDB shell interface. On the left, there is an input field labeled "Enter a title..." and a toolbar with a MongoDB icon, a dropdown menu, and a help icon. Below the toolbar is a code editor containing the MongoDB query: `1 db.restaurants.find({ "address.street": { $exists: true, $ne: "" } })`. On the right, there is an "Output" panel with the heading "mycompiler_mongodb>". The output shows the command being run and the response: `mycompiler_mongodb>`, followed by "[Execution complete with exit code 0]". A green "Run" button is located at the top right of the output panel.

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 a MongoDB shell interface. On the left, there is a text input field labeled 'Enter a title...'. Below it are two buttons: 'MongoDB' with a dropdown arrow and a small info icon. To the right is a green 'Run' button with a play icon. In the main area, a command is typed into the console:

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

On the right, there is a vertical panel titled 'Output' containing the results of the query:

```
mycompiler_mongodb>
mycompiler_mongodb>

[Execution complete with exit code 0]
```

10. Write a MongoDB query which will select the restaurant Id, name and grades for those restaurants which returns 0 as a remainder after dividing the score by 7.

QUERY:

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

OUTPUT:

Enter a title...

MongoDB ▾ ⓘ Run

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

Output

```
mycompiler_mongodb>
mycompiler_mongodb>

[Execution complete with exit code 0]
```

11. Write a MongoDB query to find the restaurant name, borough, longitude and attitude and cuisine for those restaurants which contains 'mon' as three letters somewhere in its name.

QUERY:

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

OUTPUT:

Enter a title...

MongoDB ▾ ⓘ Run

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

Output

```
mycompiler_mongodb>
mycompiler_mongodb>

[Execution complete with exit code 0]
```

12. Write a MongoDB query to find the restaurant name, borough, longitude and latitude and cuisine for those restaurants which contain 'Mad' as first three letters of its name.

QUERY:

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

OUTPUT:

The screenshot shows a MongoDB shell interface. On the left, there is a text input field with placeholder text 'Enter a title...'. Below it are two buttons: 'MongoDB' with a dropdown arrow and a small icon, and a 'Run' button with a play icon. The main area contains a code editor with the following content:

```
1 db.restaurants.find({ name: /^Mad/i }, { name: 1, borough: 1, "address.coord": 1, cuisine: 1 })
2
3
```

To the right of the code editor is a 'Run' button. Further to the right is a 'Output' panel with the following text:

```
mycompiler_mongodb>
mycompiler_mongodb>

[Execution complete with exit code 0]
```

13. Write a MongoDB query to find the restaurants that have at least one grade with a score of less than 5.

QUERY:

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

OUTPUT:

Enter a title...

MongoDB ▾



```
1 db.restaurants.find({ "grades": { $elemMatch: { "score": { $lt: 5 } } } })  
2  
3
```

Output

```
mycompiler_mongodb>  
mycompiler_mongodb>
```

```
[Execution complete with exit code 0]
```

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

QUERY:

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

OUTPUT:

Enter a title...

MongoDB ▾



```
1 db.restaurants.find({ "grades": { $elemMatch: { "score": { $lt: 5 } } }, "borough": "Manhattan" })  
2  
3
```

Output

```
mycompiler_mongodb>  
mycompiler_mongodb>
```

```
[Execution complete with exit code 0]
```

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

QUERY:

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

OUTPUT:

The screenshot shows a MongoDB shell interface. At the top, there is a search bar labeled "Enter a title...". Below it, a dropdown menu says "MongoDB" with a "▼" icon, and a small info icon. On the right, there is a green "Run" button with a play icon. The main area contains the following text:

```
1.find({ "grades": { $elemMatch: { "score": { $lt: 5 } } }, $or: [{ "borough": "Manhattan" }, { "borough": "Brooklyn" }] }) Output
2
3
```

On the right side of the interface, there is a terminal window with the following content:

```
mycompiler_mongodb>
mycompiler_mongodb>
```

At the bottom right of the terminal window, the text "[Execution complete with exit code 0]" is visible.

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:

Enter a title...

MongoDB ▾



▶ Run

```
1{ "score": { $lt: 5 } }, $or: [{ "borough": "Manhattan" }, { "borough": "Brooklyn" }], "cuisine": { $ne: "American" }}] Output
```

```
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:

Enter a title...

MongoDB ▾



▶ Run

```
1{ $lt: 5 } }, $or: [{ "borough": "Manhattan" }, { "borough": "Brooklyn" }], "cuisine": { $nin: ["American", "Chinese"] } }] Output
```

```
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 image shows a screenshot of a MongoDB shell interface. At the top, there is a search bar labeled "Enter a title...". Below it, a dropdown menu says "MongoDB" with a "Run" button next to it. The main area contains a code block with the following content:

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

To the right of the code, there is a "Output" section. It shows the command prompt "mycompiler_mongodb>" followed by "[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...

MongoDB Run

```
1 $and: [{ "grades.grade": "A", "grades.score": 2 }, { "grades.grade": "A", "grades.score": 6 }], "borough": "Manhattan" })} Output
```

```
mycompiler_mongodb>
mycompiler_mongodb>

[Execution complete with exit code 0]
```

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...

MongoDB Run

```
1, "score": 2 }, { "grades.grade": "A", "grades.score": 6 }], $or: [{ "borough": "Manhattan" }, { "borough": "Brooklyn" }] })} Output
```

```
mycompiler_mongodb>
mycompiler_mongodb>

[Execution complete with exit code 0]
```

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

grade with a score of 6 and are located in the borough of Manhattan or Brooklyn, and their cuisine is not American.

QUERY:

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

OUTPUT:

The screenshot shows a MongoDB query editor interface. At the top, there is a search bar labeled "Enter a title...". Below it, there are two dropdown menus: one set to "MongoDB" and another with a question mark icon. To the right are "Run" and "Stop" buttons. The main area contains a code block with the query and its output. The output shows the command run and the resulting database session:

```
1 'A", "grades.score": 6 }], $or: [{ "borough": "Manhattan" }, { "borough": "Brooklyn" }], "cuisine": { $ne: "American" } }) Output
```

```
mycompiler_mongodb>
mycompiler_mongodb>
```

[Execution complete with exit code 0]

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

QUERY:

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

OUTPUT:

The screenshot shows a MongoDB query editor interface. At the top, there is a search bar labeled "Enter a title...". Below it, there are two dropdown menus: one set to "MongoDB" and another with a question mark icon. To the right are "Run" and "Stop" buttons. The main area contains a code block with the query and its output. The output shows the command run and the resulting database session:

```
1 "score": 6 }], $or: [{ "borough": "Manhattan" }, { "borough": "Brooklyn" }], "cuisine": { $nin: ["American", "Chinese"] } }) 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:

The screenshot shows a MongoDB shell interface. On the left, there is a text input field with placeholder text 'Enter a title...'. Below it are two dropdown menus: one set to 'MongoDB' and another with a question mark icon. To the right are two buttons: a green 'Run' button and a blue 'Save' button. The main area contains a code snippet: '1 db.restaurants.find({ \$or: [{ "grades.score": 2 }, { "grades.score": 6 }] })'. To the right, under the heading 'Output', the response is shown: 'mycompiler_mongodb> mycompiler_mongodb> [Execution complete with exit code 0]'. The entire interface is contained within a dark-themed window.

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 a MongoDB shell interface. On the left, there is a text input field with placeholder text "Enter a title...". Below it are two buttons: "MongoDB" with a dropdown arrow and a help icon. On the right, there is a "Run" button with a play icon and a copy icon. The main area contains a command line with the text "1 db.movies.find({ year: 1893 })". To the right, under the heading "Output", the response is shown: "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 a MongoDB shell interface. On the left, there is a text input field with placeholder text "Enter a title...". Below it are two buttons: "MongoDB" with a dropdown arrow and a help icon. On the right, there is a "Run" button with a play icon and a copy icon. The main area contains a command line with the text "1 db.movies.find({ runtime: { \$gt: 120 } })". To the right, under the heading "Output", the response is shown: "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 MongoDB Compass interface. On the left, there is a search bar labeled "Enter a title...". Below it are two buttons: "MongoDB" with a dropdown arrow and a small info icon. To the right is a green "Run" button with a play icon. In the center-left panel, the query `db.movies.find({ genres: 'Short' })` is typed into the command line. In the center-right panel, the output shows the results of the query: `mycompiler_mongodb>`, `mycompiler_mongodb>`, and the message `[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 MongoDB Compass interface. On the left, there is a search bar labeled "Enter a title...". Below it are two buttons: "MongoDB" with a dropdown arrow and a small info icon. To the right is a green "Run" button with a play icon. In the center-left panel, the query `db.movies.find({ directors: 'William K.L. Dickson' })` is typed into the command line. In the center-right panel, the output shows the results of the query: `mycompiler_mongodb>`, `mycompiler_mongodb>`, and the message `[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 MongoDB shell interface. On the left, there is a text input field with placeholder text "Enter a title...". Below it are two buttons: one with a green leaf icon labeled "MongoDB" and another with an info icon. To the right is a "Run" button with a play icon. The main area contains the command "1 db.movies.find({ countries: 'USA' })". To the right, under the heading "Output", the results are displayed: "mycompiler_mongodb>" followed by "mycompiler_mongodb>". Below that 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 MongoDB shell interface. On the left, there is a text input field with placeholder text "Enter a title...". Below it are two buttons: one with a green leaf icon labeled "MongoDB" and another with an info icon. To the right is a "Run" button with a play icon. The main area contains the command "1 db.movies.find({ rated: 'UNRATED' })". To the right, under the heading "Output", the results are displayed: "mycompiler_mongodb>" followed by "mycompiler_mongodb>". Below that is the message "[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:

Enter a title...

MongoDB Run

```
1 db.movies.find({ 'imdb.votes': { $gt: 1000 } })
```

Output

```
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:

Enter a title...

MongoDB Run

```
1 db.movies.find({ 'imdb.rating': { $gt: 7 } })
```

Output

```
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 a MongoDB shell interface. On the left, there is a search bar labeled "Enter a title...". Below it, a dropdown menu says "MongoDB" with a dropdown arrow, and a small info icon. On the right, there is a green "Run" button with a play icon. In the center, a code editor window displays the following command:

```
1 db.movies.find({ 'tomatoes.viewer.rating': { $gt: 4 } })
```

To the right of the code editor is a "Output" panel. It contains the following text:

```
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 a MongoDB shell interface. On the left, there is a search bar labeled "Enter a title...". Below it, a dropdown menu says "MongoDB" with a dropdown arrow, and a small info icon. On the right, there is a green "Run" button with a play icon. In the center, a code editor window displays the following command:

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

To the right of the code editor is a "Output" panel. It contains the following text:

```
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 MongoDB Compass interface. At the top, there is a search bar labeled "Enter a title...". Below it, a dropdown menu says "MongoDB" with a "▼" icon, and a small info icon. On the right side of the interface, there is a green "Run" button. The main area displays the query results:

```
1, languages: 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]

At the bottom right of the interface, there is a "Feedback" button and a note: "Explore design possibilities with Fi easily translate your work into coc".

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:

Enter a title...

MongoDB ▾ ⓘ

Run

```
1, languages: 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]
```

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:

Enter a title...

Ctrl+Enter

MongoDB ▾ ⓘ

Run

```
1: ISODate("1893-05-09T00:00:00.000Z" ), { title: 1, languages: 1, released: 1, directors: 1, writers: 1, countries: 1 }] Output
```

```
mycompiler_mongodb>
mycompiler_mongodb>
```

```
[Execution complete with exit code 0]
```

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

QUERY:

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

OUTPUT:



The screenshot shows a MongoDB shell interface. On the left, there is a search bar with placeholder text "Enter a title...". Below it are two buttons: "MongoDB" with a dropdown arrow and a small info icon. To the right is a "Run" button with a play icon. The main area contains a code editor with the following text:

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

To the right of the code editor is a "Output" panel. It displays the command entered and the resulting output from the MongoDB database:

```
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: