

Creating and Managing Tables

EX_NO:1

DATE:

- 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 deptt(dept_id Number(6),dept_name varchar(20),manager_id number(6),location_id number(4));

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, Gallery, a search bar, and a user profile for 'Blessy Abidha'. The main area is titled 'SQL Commands' with a schema dropdown set to 'WKSP_BLESSYWORKSPACE'. The SQL editor contains the command: 'create table deptt(dept_id Number(6),dept_name varchar(20),Manager_id number(6),location_id number(4))'. Below the editor, the 'Results' tab is selected, showing the output: 'Table created.' and '0.04 seconds'. Other tabs available are Explain, Describe, Saved SQL, and History.

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 employee(id number(6),last_Name varchar2(25),First_Name varchar2(25),salary  
number(8,2),Dept_id number(8,2));
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes links for App Builder, SQL Workshop, Team Development, Gallery, and a user profile for 'Blessy Abidha'. The main workspace is titled 'SQL Commands' and shows the schema 'WKSP_BLESSYWORKSPACE'. The SQL editor contains the following code:

```
1 create table employee(id number(6)not null,first_name varchar(25) not null,last_name varchar(25) not null,  
2 salary number(8,2),dept_id Number(8,2) not null)
```

Below the editor, there are tabs for Results, Explain, Describe, Saved SQL, and History. The results section displays the message "Table created." and a execution time of "0.04 seconds".

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

QUERY:

```
Alter table employee modify>Last_Name varchar(50);
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', 'Gallery', a search bar, and a user profile for 'Blessy Abidha'. Below the navigation is a toolbar with icons for undo, redo, search, and refresh, followed by 'Run' and 'Save' buttons. The main area is titled 'SQL Commands' with a schema dropdown set to 'WKSP_BLESSYWORKSPACE'. The SQL editor contains the command: 'alter table employeee modify(last_name varchar(50))'. The results tab is selected, showing the output: 'Table altered.' and a execution time of '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(6),first_name varchar2(25),Last_name varchar2(25),Salary Number,Dept_id number(6));
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', 'Gallery', a search bar, and a user profile for 'Blessy Abidha'. Below the navigation is a toolbar with icons for undo, redo, search, and refresh, followed by 'Run' and 'Save' buttons. The main area is titled 'SQL Commands' with a schema dropdown set to 'WKSP_BLESSYWORKSPACE'. The SQL editor contains the command: 'create table employee2 (id number(6) not null,first_name varchar(20),last_name varchar(25) not null, salary number(8,2),dept_id number(6) not null)'. The results tab is selected, showing the output: 'Table created.' and a execution time of '3.65 seconds'.

5. Drop the EMP table.

QUERY:

```
Drop table emp;
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', 'Gallery', a search bar, and a user profile for 'Blessy Abidha'. The 'SQL Commands' tab is selected. The SQL editor contains the command: '1 drop table employee'. The results section shows the output: 'Table dropped.' and a execution time of '2.52 seconds'.

```
1 drop table employee
```

Table dropped.
2.52 seconds

6. Rename the EMPLOYEES2 table as EMP.

QUERY:

```
Rename employees2 to employee;
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', 'Gallery', a search bar, and a user profile for 'Blessy Abidha'. The 'SQL Commands' tab is selected. The SQL editor contains the command: '1 rename employee2 to employee'. The results section shows the output: 'Statement processed.' and a execution time of '0.06 seconds'.

```
1 rename employee2 to employee
```

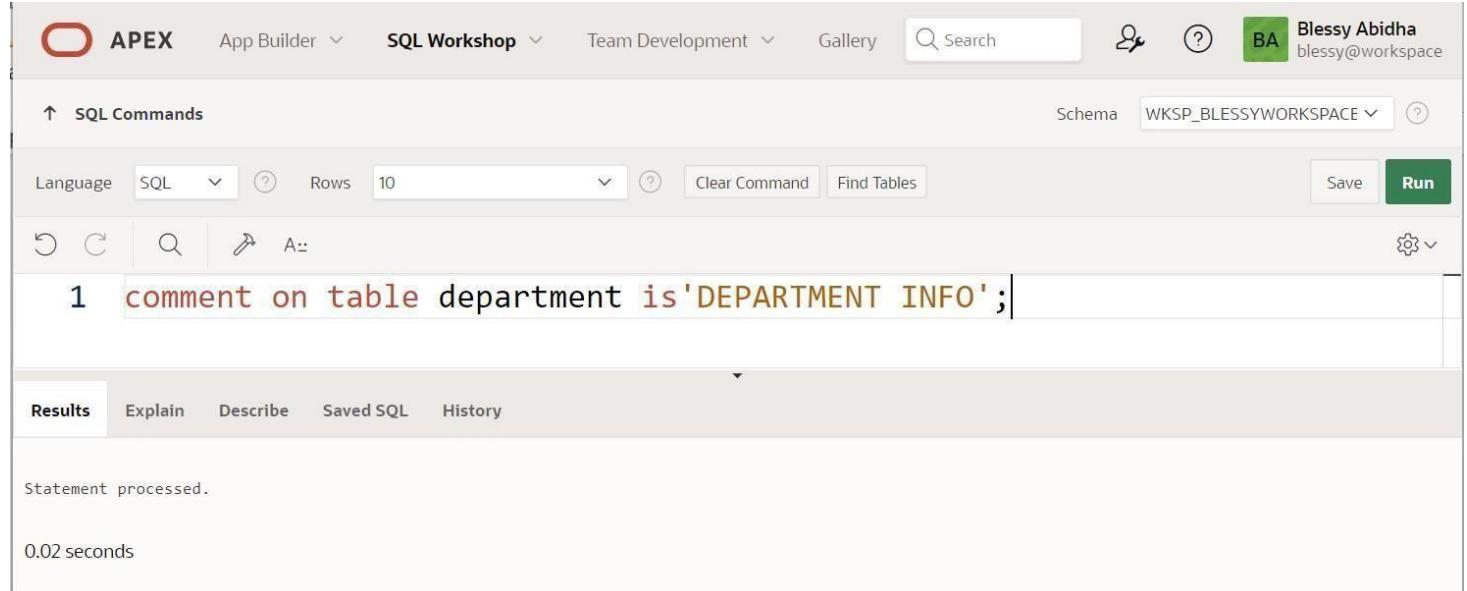
Statement processed.
0.06 seconds

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

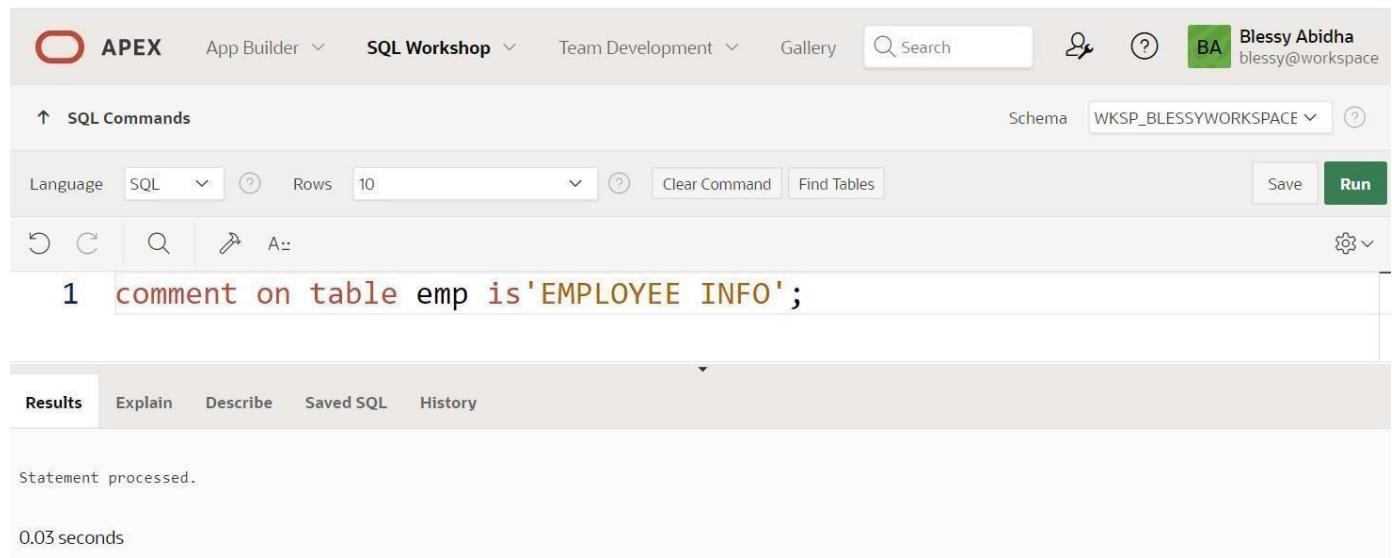
QUERY:

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

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', 'Gallery', a search bar, and a user profile for 'Blessy Abidha'. The main area is titled 'SQL Commands' with a schema dropdown set to 'WKSP_BLESSYWORKSPACE'. The command entered is: 'comment on table department is 'DEPARTMENT INFO'';. Below the command, the 'Results' tab is selected, showing the output: 'Statement processed.' and '0.02 seconds' execution time.



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', 'Gallery', a search bar, and a user profile for 'Blessy Abidha'. The main area is titled 'SQL Commands' with a schema dropdown set to 'WKSP_BLESSYWORKSPACE'. The command entered is: 'comment on table emp is 'EMPLOYEE INFO'';. Below the command, the 'Results' tab is selected, showing the output: 'Statement processed.' and '0.03 seconds' execution time.

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

QUERY:

Alter table employee drop column first_name;

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, and Gallery. A search bar and user information for 'Blessy Abidha' are also present. The main area is titled 'SQL Commands' with a schema dropdown set to 'WKSP_BLESSYWORKSPACE'. The SQL editor contains the command: '1 alter table employeee drop column first_name'. Below the editor, the 'Results' tab is selected, showing the output: 'Table altered.' and '0.06 seconds'. Other tabs include Explain, Describe, Saved SQL, and History.

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

RESULT:

MANIPULATING DATA

EX_NO:2

DATE:

1. Create MY_EMPLOYEE table with the following structure

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

QUERY:

```
Create table my_employee(id number(4)not null,last_name varchar(25),userid varchar(25),salary number(9,2))
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', 'Gallery', a search bar, and user information 'BA Blessy Abidha blessy@workspace'. The main area is titled 'SQL Commands' with a schema dropdown set to 'WKSP_BLESSYWORKSPACE'. Below the title are buttons for 'Language' (set to 'SQL'), 'Rows' (set to 10), 'Clear Command', 'Find Tables', 'Save', and 'Run'. The SQL command entered is:

```
1 create table My_Employee(id number(4)not null,last_name varchar(25),first_name varchar(25),userid varchar(25),salary number(9,2))
```

The results section shows the output of the command:

```
Table created.
```

Execution time is listed as 0.04 seconds.

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

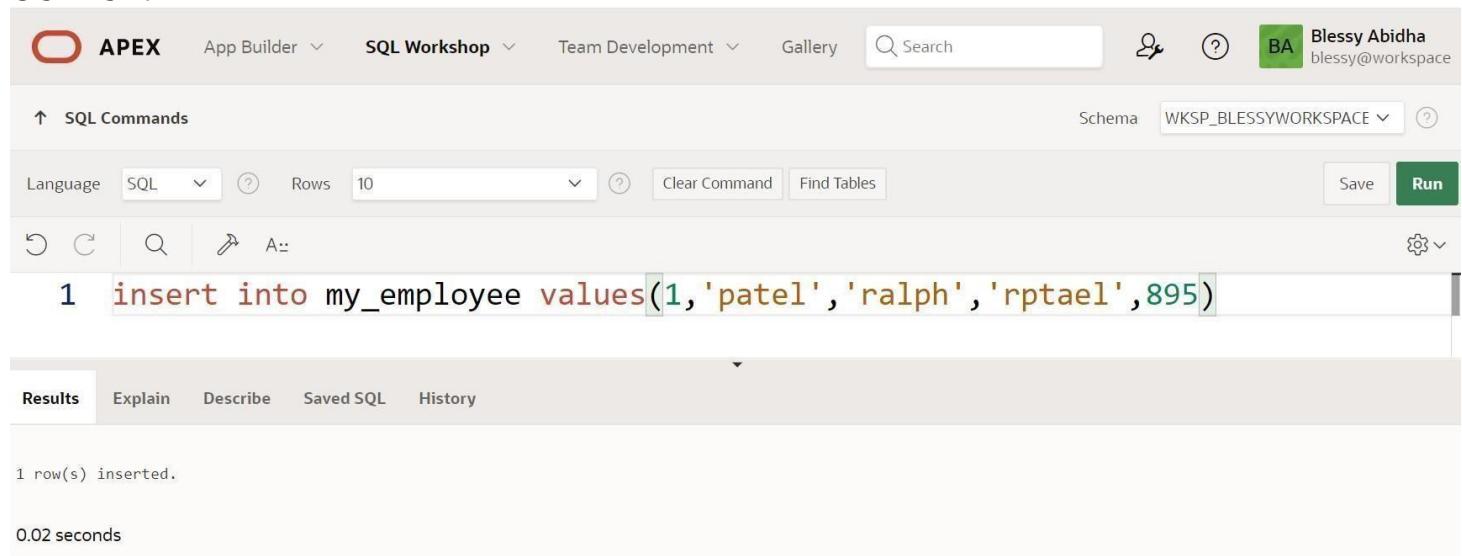
ID	Last_name	First_name	Userid	salary
1	Patel	Ralph	rpatel	895
2	Dancs	Betty	bdancs	860
3	Biri	Ben	bbiri	1100
4	Newman	Chad	Cnewman	750
5	Ropebur	Audrey	aropebur	1550

QUERY:

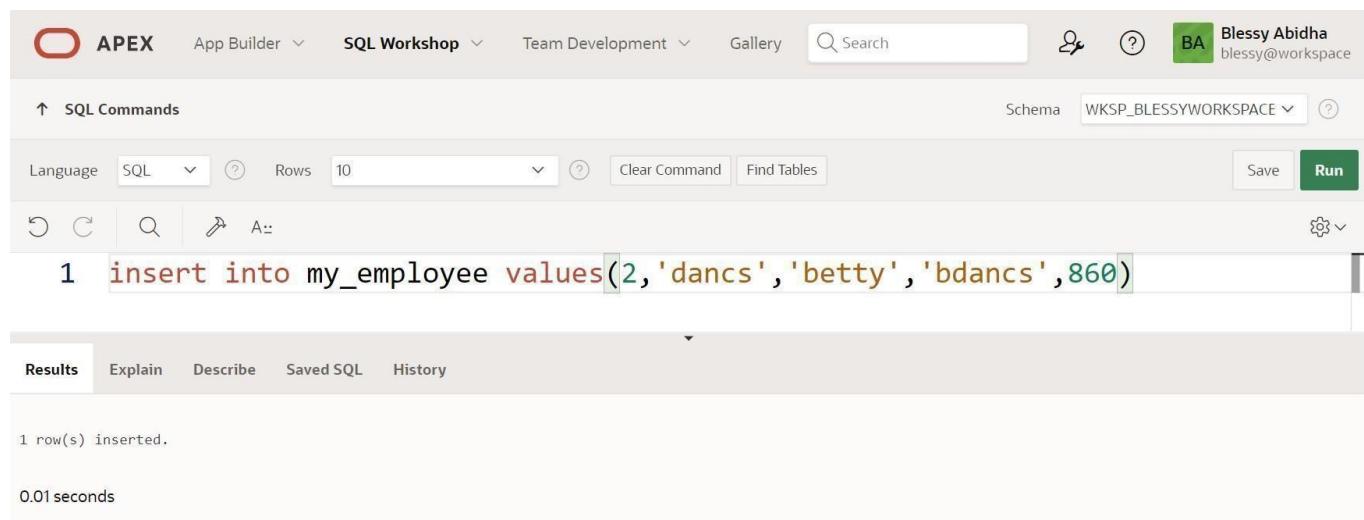
Insert into my_employee values(1,'patel','ralph','rpatel',895)

Insert into my_employee values(2,'dancs','betty','bdancs',860)

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, Gallery, a search bar, and a user profile for 'Blessy Abidha'. The main area is titled 'SQL Commands' with a schema dropdown set to 'WKSP_BLESSYWORKSPACE'. The SQL editor contains the command: '1 insert into my_employee values(1, 'patel', 'ralph', 'rptael', 895)'. Below the editor, the 'Results' tab is selected, showing the output: '1 row(s) inserted.' and '0.02 seconds'. Other tabs like Explain, Describe, Saved SQL, and History are also visible.



The screenshot shows the Oracle SQL Workshop interface again. The top navigation bar is identical. The main area is titled 'SQL Commands' with a schema dropdown set to 'WKSP_BLESSYWORKSPACE'. The SQL editor contains the command: '1 insert into my_employee values(2, 'dancs', 'betty', 'bdancs', 860)'. Below the editor, the 'Results' tab is selected, showing the output: '1 row(s) inserted.' and '0.01 seconds'. Other tabs like Explain, Describe, Saved SQL, and History are also visible.

3. Display the table with values.

QUERY:

Select * from my_employee

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. At the top, there are tabs for 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. A search bar and user information ('Blessy Abidha, blessy@workspace') are also at the top. Below the tabs, there's a toolbar with icons for 'SQL Commands', 'Schema' (set to 'WKSP_BLESSYWORKSPACE'), and 'Run' (green button). The main area shows a SQL command line with the query 'select * from MY_employee'. The results tab is selected, displaying a table with two rows of data. The columns are labeled 'ID', 'LAST_NAME', 'FIRST_NAME', 'USERID', and 'SALARY'. The data is as follows:

ID	LAST_NAME	FIRST_NAME	USERID	SALARY
1	patel	ralph	rptael	895
2	dancs	betty	bdancs	860

At the bottom left, it says '2 rows returned in 0.01 seconds'. There are also 'Download' and 'Copy' buttons.

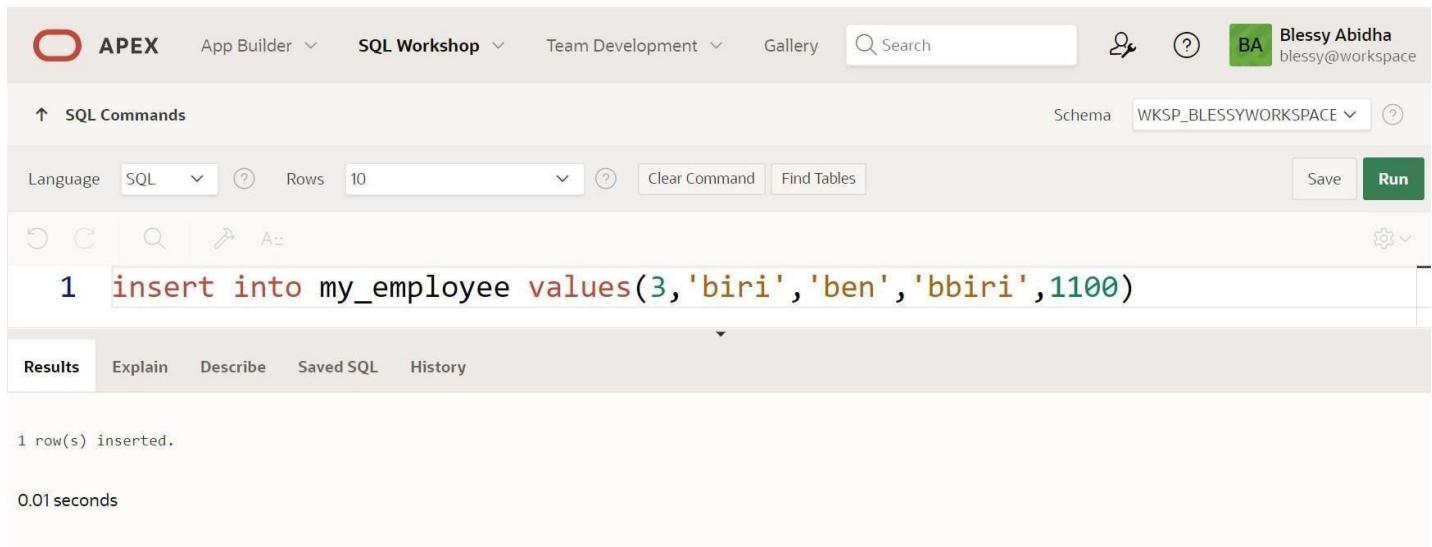
4. Populate the next two rows of data from the sample data. Concatenate the first letter of the first_name with the first seven characters of the last_name to produce Userid.

QUERY:

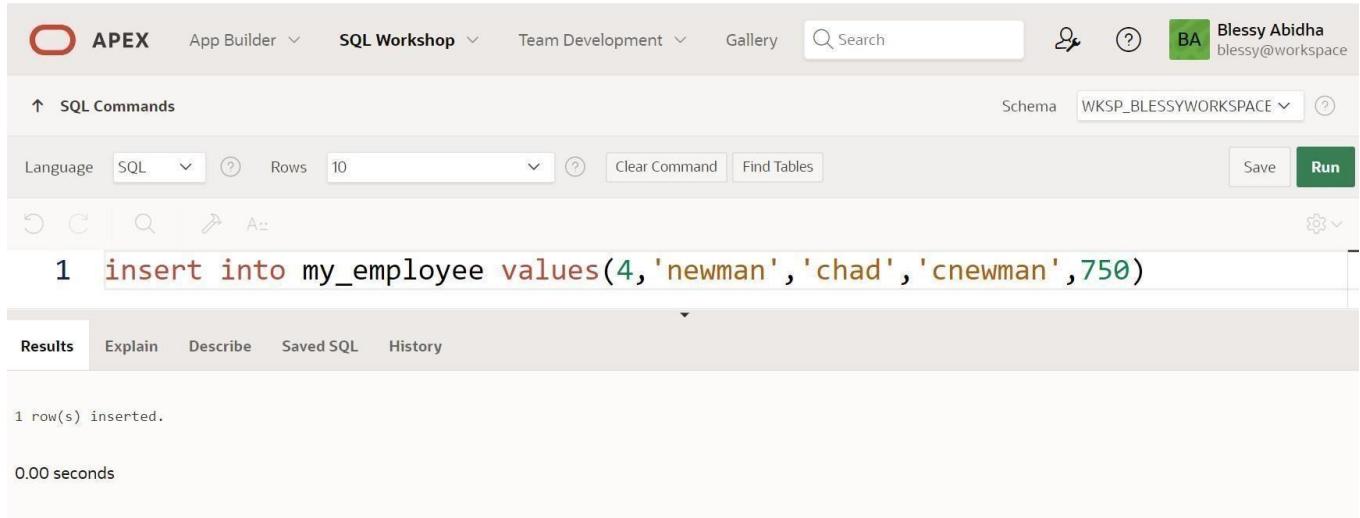
```
Insert into my_employee values(3,'biri','ben','bbiri',1100)
```

```
Insert into my_employee values(4,'newman','chad','cnewman',750)
```

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', 'Gallery', a search bar, and user information ('Blessy Abidha' and 'blessy@workspace'). Below the navigation is a toolbar with icons for 'SQL Commands', 'Schema' (set to 'WKSP_BLESSYWORKSPACE'), and various filters ('Language: SQL', 'Rows: 10'). The main area displays the SQL command: '1 insert into my_employee values(3, 'biri', 'ben', 'bbiri', 1100)'. The results section shows the output: '1 row(s) inserted.' and '0.01 seconds' execution time.



The screenshot shows the Oracle SQL Workshop interface, identical to the first one but with a different SQL command. The top navigation bar and schema selection are the same. The main area displays the SQL command: '1 insert into my_employee values(4, 'newman', 'chad', 'cnewman', 750)'. The results section shows the output: '1 row(s) inserted.' and '0.00 seconds' execution time.

5. Make the data additions permanent.

QUERY:

Select * from my_employee

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. At the top, there are tabs for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there's a user profile for 'Blessy Abidha' (blessy@workspace). Below the tabs, there's a search bar and some icons. The main area has a toolbar with Language (SQL), Rows (10), Clear Command, Find Tables, Save, and Run buttons. The SQL command entered is 'select * from my_employee'. The results tab is selected, showing a table with four 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
1	patel	ralph	rptael	895
2	dancs	betty	bdancs	860
3	biri	ben	bbiri	1100
4	newman	chad	cnewman	750

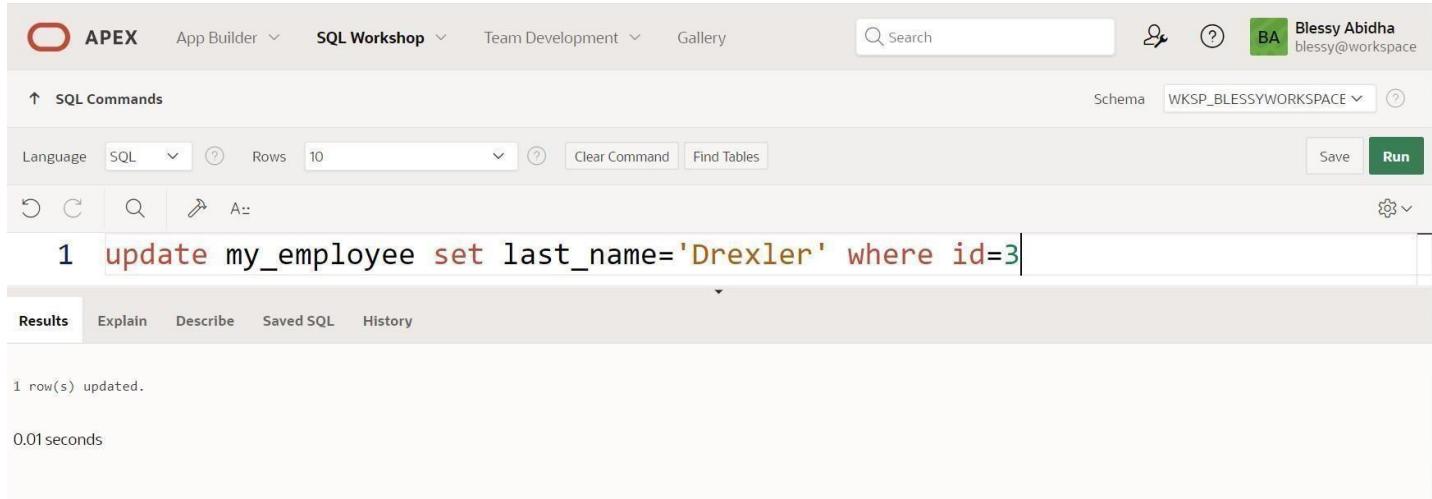
Below the table, it says '4 rows returned in 0.00 seconds' and there's a 'Download' link.

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

QUERY:

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

OUTPUT:



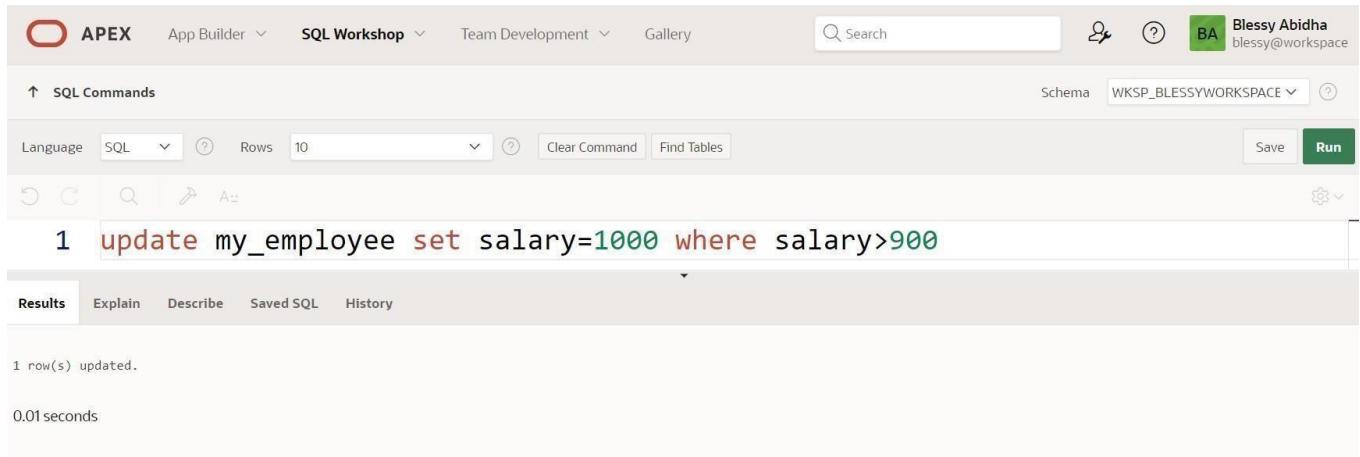
The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there's a user profile for 'Blessy Abidha' (blessy@workspace). The main area is titled 'SQL Commands'. The command entered is: `update my_employee set last_name='Drexler' where id=3`. Below the command, the results show: '1 row(s) updated.' and '0.01 seconds'. There are tabs for Results, Explain, Describe, Saved SQL, and History.

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

QUERY:

Update my_employee set salary=1000 where salary<900

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there's a user profile for 'Blessy Abidha' (blessy@workspace). The main area is titled 'SQL Commands'. The command entered is: `update my_employee set salary=1000 where salary<900`. Below the command, the results show: '1 row(s) updated.' and '0.01 seconds'. There are tabs for Results, Explain, Describe, Saved SQL, and History.

8. Delete Betty dancs from MY_EMPLOYEE table.

QUERY:

Delete from my_employee where first_name ='betty' and last_name =Dancs;

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. A search bar and user information (BA Blessy Abidha) are also present. The main area is titled "SQL Commands" with a "Results" tab selected. The query entered is: `1 delete from my_employee where first_name='betty' and last_name='Dancs'`. The results section shows "0 row(s) deleted." and a execution time of "0.02 seconds".

9. Empty the fourth row of the emp table.

QUERY:

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. A search bar and user information (BA Blessy Abidha) are also present. The main area is titled "SQL Commands" with a "Results" tab selected. The query entered is: `1 delete from my_employee where id=4`. The results section shows "1 row(s) deleted." and a execution time of "0.00 seconds".

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

RESULT:

INCLUDING CONSTRAINTS

EX_NO:3

DATE:

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

QUERY:

```
Create table emp(employee_id number(6),lastname varchar2(25),salary number(8,2),constraint emp_id primary key (employee_id))
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', 'Gallery', a search bar, and a user profile for 'Blessy Abidha'. The main workspace is titled 'SQL Commands' under 'Schema WKSP_BLESSYWORKSPACE'. The SQL editor contains the following command:

```
1 create table emp(employee_id number(6),lastname varchar2(25),salary number(8,2),constraint emp_id primary key(employee_id))
```

The results tab shows the output:

```
Table created.
```

Execution time: 0.06 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 depart(dept_id number(6),dept_name varchar(20),manager_id number(6),location_id number(4),constraint my_dept_id primary key(dept_id))
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', 'Gallery', a search bar, and a user profile for 'Blessy Abidha'. The main workspace is titled 'SQL Commands' under 'Schema WKSP_BLESSYWORKSPACE'. The SQL editor contains the following command:

```
1 create table depart(dept_id number(6),dept_name varchar(20),manager_id number(6),location_id number(4),constraint my_dept_id primary key(dept_id))
```

The results tab shows the output:

```
Table created.
```

Execution time: 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 emp add dept_id number(6);
```

```
Alter table emp add constraint my_emp_dept_id_fk foreign key(dept_id) References emp(employee_id)
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. A search bar and user information for 'Blessy Abidha' are also present. The main area is titled 'SQL Commands'. The language is set to SQL, and the results show 10 rows. The command entered is 'alter table emp add dept_id number(6)'. The results tab shows the output: 'Table altered.' and a execution time of '0.06 seconds'.

```
1 alter table emp add dept_id number(6)
```

Results Explain Describe Saved SQL History

Table altered.
0.06 seconds

The screenshot shows the Oracle 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 for 'Blessy Abidha' are also present. The main area is titled 'SQL Commands'. The language is set to SQL, and the results show 10 rows. The command entered is 'alter table emp add constraint my_emp_dept_id_fk foreign key(dept_id) References emp(employee_id)'. The results tab shows the output: 'Table altered.' and a execution time of '0.06 seconds'.

```
1 alter table emp add constraint my_emp_dept_id_fk foreign key(dept_id) References emp(employee_id)
```

Results Explain Describe Saved SQL History

Table altered.
0.06 seconds

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

QUERY:

Alter table depart add commission number(2,2)

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. At the top, there's a navigation bar with 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', 'Gallery', a search bar, and a user profile for 'Blessy Abidha'. Below the navigation bar is a toolbar with various icons for SQL operations like 'Run', 'Save', and 'Find Tables'. The main area is titled 'SQL Commands' and contains a text input field with the SQL command: '1 alter table depart add commission number(2,2)'. Below the input field, there's a results panel with tabs for 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History'. The 'Results' tab is selected, displaying the output: 'Table altered.' and '0.06 seconds'.

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

RESULT:

Writing Basic SQL SELECT Statements

EX_NO:4

DATE:

1. The following statement executes successfully.

Identify the Errors

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

QUERY:

Select employee_id, last_name, salary*12 annual_salary from employees

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. At the top, there's a navigation bar with tabs for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. A search bar and user information for 'Blessy Abidha' are also present. Below the toolbar, the main workspace has a title 'SQL Commands'. It includes buttons for Language (set to SQL), Rows (set to 10), Clear Command, Find Tables, Save, and Run. The SQL command entered is:

```
1 select emp_id, last_name, salary*12 annual_salary from emp1
```

Under the 'Results' tab, it displays the message 'no data found'.

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

QUERY:

Desc department;

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', 'Gallery', a search bar, and a user profile for 'Blessy Abidha'. The main area is titled 'SQL Commands' with a schema dropdown set to 'WKSP_BLESSYWORKSPACE'. Below the title are buttons for 'Language' (set to 'SQL'), 'Rows' (set to 10), 'Clear Command', 'Find Tables', 'Save', and 'Run'. The command 'desc department' is entered in the text area. The results tab is selected, showing the table structure for 'DEPARTMENT' with four columns: DEPT_ID, DEPT_NAME, MANAGER_ID, and LOCATION_ID.

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
DEPARTMENT	DEPT_ID	NUMBER	-	6	0	-	✓	-	-
	DEPT_NAME	VARCHAR2	20	-	-	-	✓	-	-
	MANAGER_ID	NUMBER	-	6	0	-	✓	-	-
	LOCATION_ID	NUMBER	-	6	0	-	✓	-	-

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

QUERY:

Select emp_number,last_name,job_code,hire_date from emp1

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', 'Gallery', a search bar, and a user profile for 'Blessy Abidha'. The main area is titled 'SQL Commands' with a schema dropdown set to 'WKSP_BLESSYWORKSPACE'. Below the title are buttons for 'Language' (set to 'SQL'), 'Rows' (set to 10), 'Clear Command', 'Find Tables', 'Save', and 'Run'. The command 'select emp_number, last_name, job_code, hire_date from emp1' is entered in the text area. The results tab is selected, displaying the output of the query with two rows. The columns are labeled 'EMP_NUMBER', 'LAST_NAME', 'JOB_CODE', and 'HIRE_DATE'.

EMP_NUMBER	LAST_NAME	JOB_CODE	HIRE_DATE
2	Anisha	33	06/14/2000
1	Arul	32	06/02/2000

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

4. Provide an alias STARTDATE for the hire date.

QUERY:

Select hire_date as start_date from emp1

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, Gallery, a search bar, and a user profile for 'Blessy Abidha'. The SQL Workshop tab is selected. Below the toolbar, there are buttons for Language (SQL), Rows (10), Clear Command, Find Tables, Save, and Run. The main area contains the SQL command: '1 select hire_date as start_date from emp1'. The results section shows the output: 'START_DATE' followed by two rows: '06/14/2000' and '06/02/2000'. A note at the bottom says '2 rows returned in 0.01 seconds'.

START_DATE
06/14/2000
06/02/2000

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

QUERY:

Select unique job_code from emp1

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, Gallery, a search bar, and a user profile for 'Blessy Abidha'. The SQL Workshop tab is selected. Below the toolbar, there are buttons for Language (SQL), Rows (10), Clear Command, Find Tables, Save, and Run. The main area contains the SQL command: '1 select unique job_code from emp1'. The results section shows the output: 'JOB_CODE' followed by two rows: '33' and '32'. A note at the bottom says '2 rows returned in 0.01 seconds'.

JOB_CODE
33
32

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_code as employee_and_title from emp1

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, Gallery, a search bar, and a user profile for 'Blessy Abidha'. The SQL Workshop tab is selected. Below the toolbar, there are buttons for Language (SQL), Rows (10), Clear Command, Find Tables, Save, and Run. The main area contains the following SQL command:

```
1 select last_name||', '||job_code as employee_and_title from emp1
```

The results section shows the output:

EMPLOYEE_AND_TITLE
Anisha,33
Arul,32

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

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

QUERY:

Select emp_id||','||emp_number||','||last_name||','||salary||','||job_code||','||hire_date||,' as the_output from emp2

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, Gallery, a search bar, and a user profile for 'Blessy Abidha'. The SQL Workshop tab is selected. Below the toolbar, there are buttons for Language (SQL), Rows (10), Clear Command, Find Tables, Save, and Run. The main area contains the following SQL command:

```
1 select emp_id||','||emp_number||','||last_name||','||salary||','||  
2 job_code||','||hire_date ||',' as the_output from emp1
```

The results section shows the output:

THE_OUTPUT
2,2,Anisha,4000,33,06/14/2000,
1,1,Arul,1000,32,06/02/2000,

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

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

RESULT:

RESTRICTINGANDSORTINGDATA

EX.NO:5

DATE:

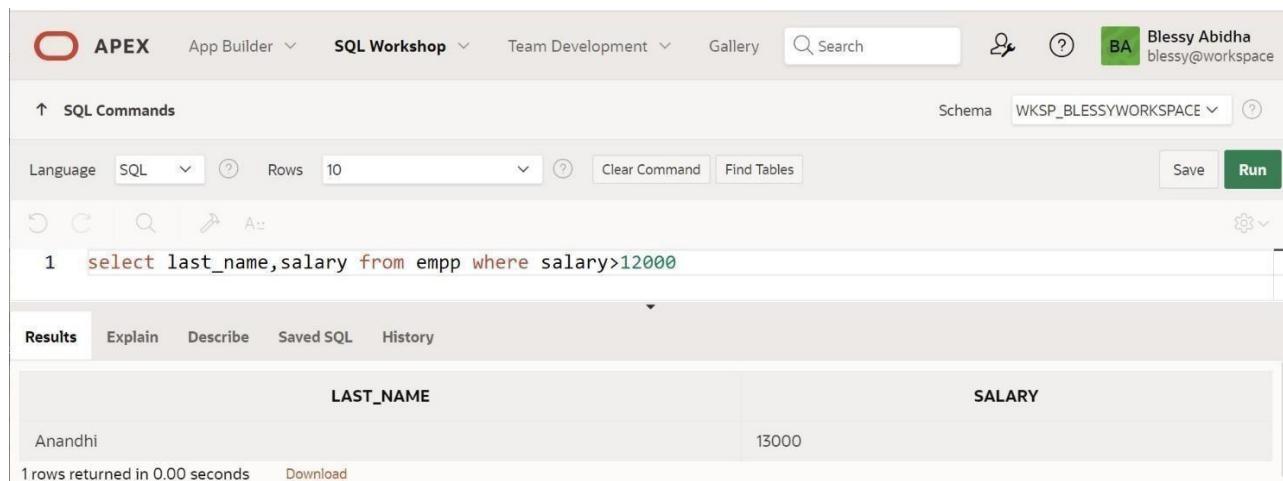
Find the Solution for the following:

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

QUERY:

```
select last_name,salary from empp where salary>12000;
```

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', 'Gallery', a search bar, and a user profile for 'Blessy Abidha'. The main area is titled 'SQL Commands' with a schema dropdown set to 'WKSP_BLESSYWORKSPACE'. The SQL editor contains the query: 'select last_name,salary from empp where salary>12000'. The results tab is selected, displaying a single row: Anandhi with a salary of 13000. A note at the bottom says '1 rows returned in 0.00 seconds'.

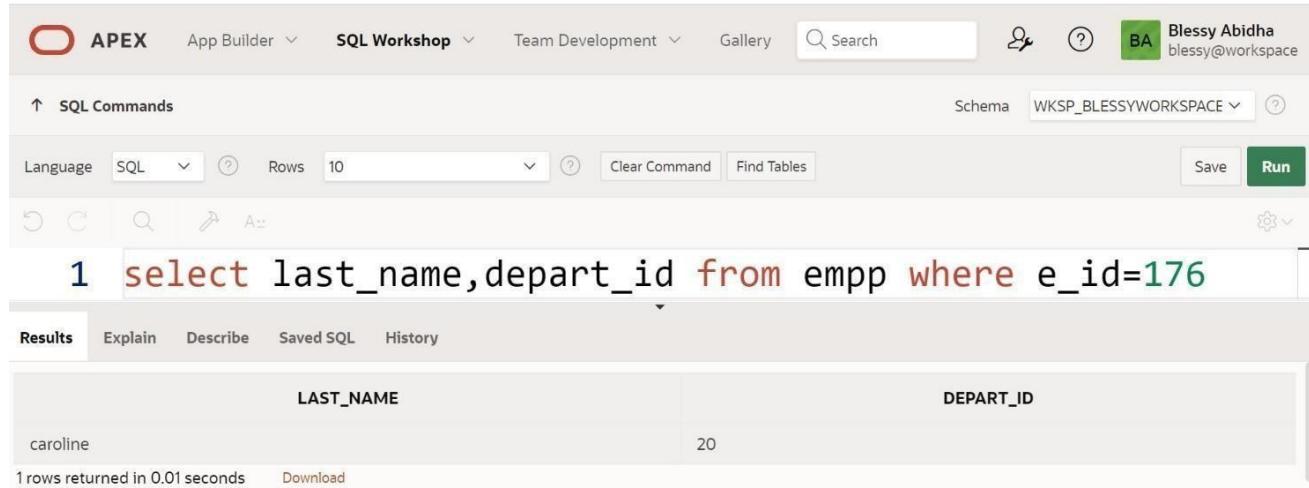
LAST_NAME	SALARY
Anandhi	13000

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

QUERY:

Select last_name, depart_id from empp where e_id=176;

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', 'Gallery', a search bar, and a user profile for 'Blessy Abidha'. The main area is titled 'SQL Commands' with a schema dropdown set to 'WKSP_BLESSYWORKSPACE'. The SQL editor contains the query: '1 select last_name,depart_id from empp where e_id=176'. The results tab is selected, displaying a single row: 'caroline' in the LAST_NAME column and '20' in the DEPART_ID column.

LAST_NAME	DEPART_ID
caroline	20

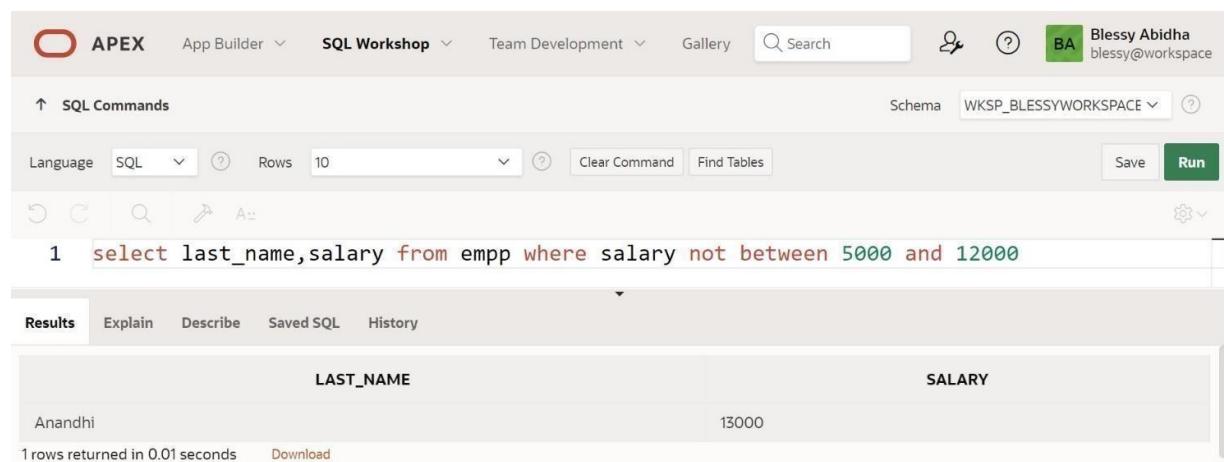
1 rows returned in 0.01 seconds [Download](#)

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 empp where salary not between 5000 and 12000

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', 'Gallery', a search bar, and a user profile for 'Blessy Abidha'. The main area is titled 'SQL Commands' with a schema dropdown set to 'WKSP_BLESSYWORKSPACE'. The SQL editor contains the query: '1 select last_name,salary from empp where salary not between 5000 and 12000'. The results tab is selected, displaying a single row: 'Anandhi' in the LAST_NAME column and '13000' in the SALARY column.

LAST_NAME	SALARY
Anandhi	13000

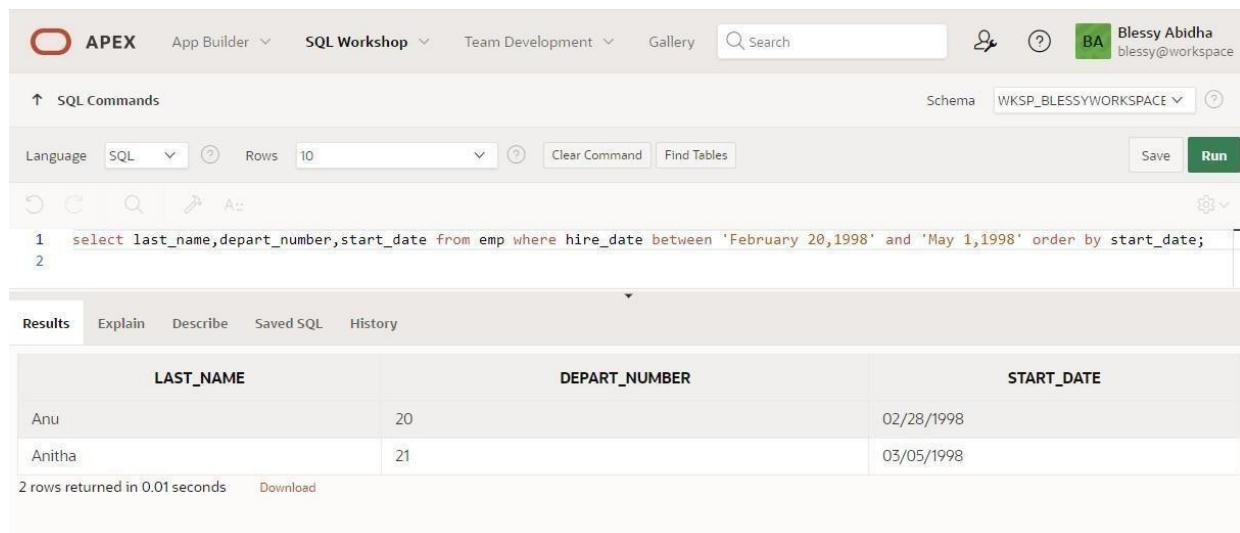
1 rows returned in 0.01 seconds [Download](#)

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

QUERY:

Select last_name, depart_number,start_date from empp where hire_date between 'February 20,1998' and 'May 1,1998' order by start_date;

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, Gallery, a search bar, and user information for 'Blessy Abidha'. Below the toolbar, the schema is set to 'WKSP_BLESSYWORKSPACE'. The main area contains a SQL command window with the following content:

```
1 select last_name,depart_number,start_date from emp where hire_date between 'February 20,1998' and 'May 1,1998' order by start_date;
2
```

Below the command window, there are tabs for Results, Explain, Describe, Saved SQL, and History. The Results tab is selected, displaying the following table:

LAST_NAME	DEPART_NUMBER	START_DATE
Anu	20	02/28/1998
Anitha	21	03/05/1998

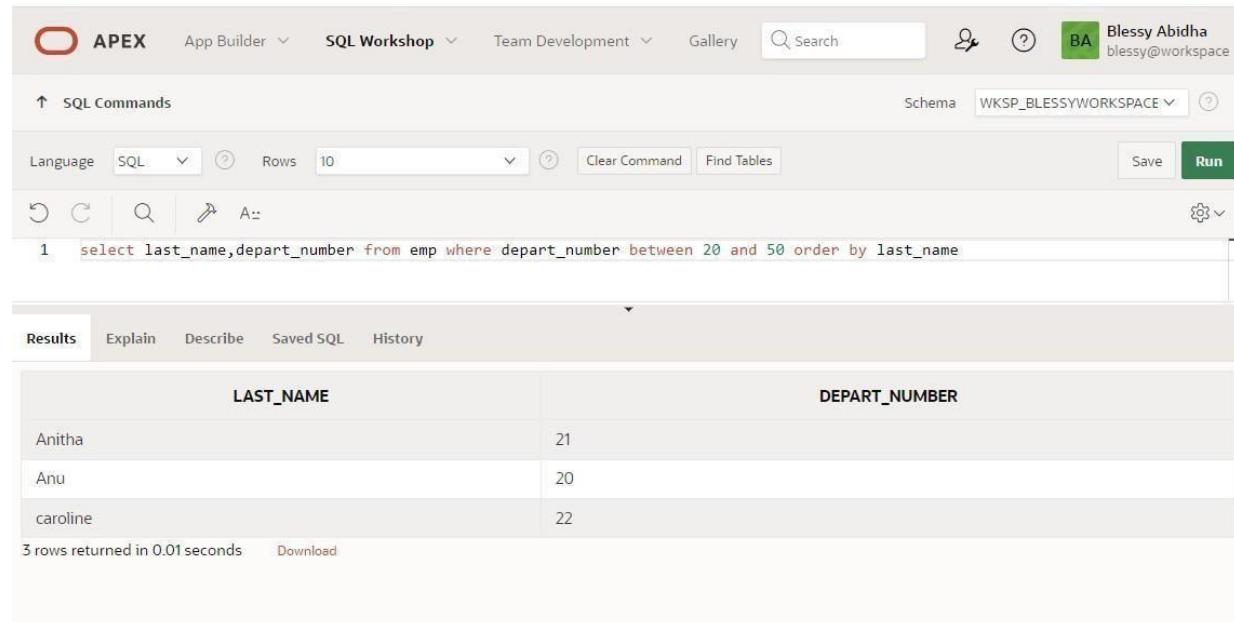
At the bottom left of the results panel, it says '2 rows returned in 0.01 seconds'. There is also a 'Download' link.

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,depart_number from empp where depart_number in('20','50') order by last_name;
```

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, Gallery, a search bar, and user information for 'Blessy Abidha'.

The SQL Commands panel shows the query:

```
1 select last_name,depart_number from emp where depart_number between 20 and 50 order by last_name
```

The Results tab displays the output:

LAST_NAME	DEPART_NUMBER
Anitha	21
Anu	20
caroline	22

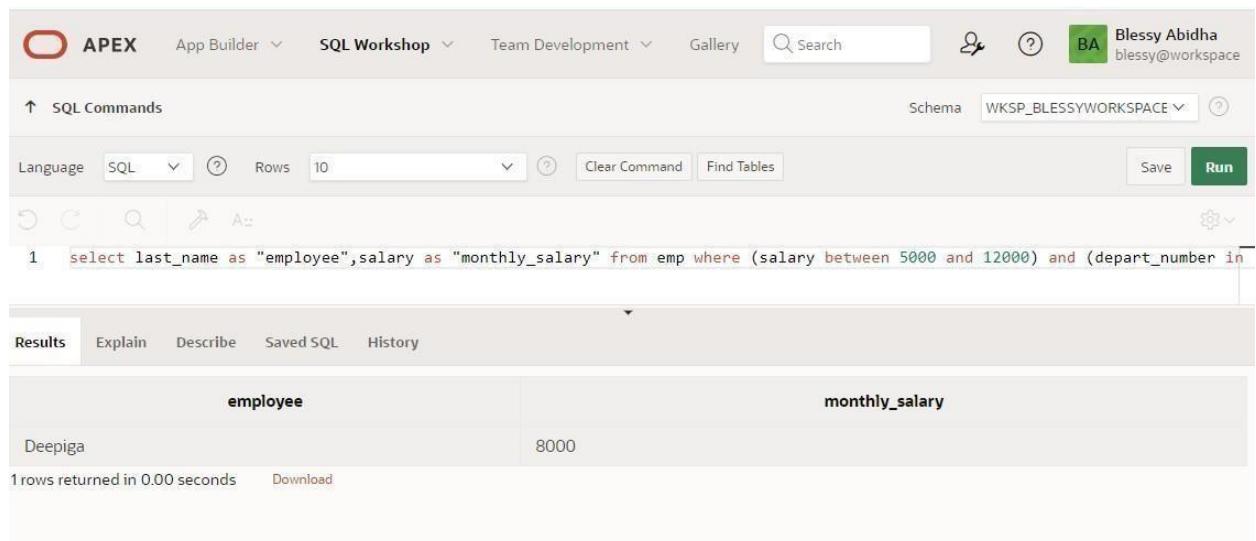
3 rows returned in 0.01 seconds

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 “employee” ,salary as “monthly salary” from empp where salary between 5000 and 12000 and depart_number between 20 and 50 order by last_name

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, Gallery, a search bar, and a user profile for BA Blessy Abidha. The SQL Commands tab is active, showing the following query:

```
1 select last_name as "employee", salary as "monthly_salary" from emp where (salary between 5000 and 12000) and (depart_number in
```

The Results tab is selected, displaying the output:

employee	monthly_salary
Deepika	8000

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

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 empp where hire_date like '%1994';

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. At the top, there's a navigation bar with tabs for APEX, App Builder, SQL Workshop, Team Development, and Gallery. A search bar and user information for 'Blessy Abidha' are also present. Below the navigation bar is a toolbar with various icons for SQL operations like Insert, Update, Delete, and Select.

In the main area, under the 'SQL Commands' tab, a SQL command is entered:

```
1  Select last_name ,hire_date from empp where hire_date like '%1994'
```

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

LAST_NAME	HIRE_DATE
Anisha	02/16/1994
Anisha	02/16/1994

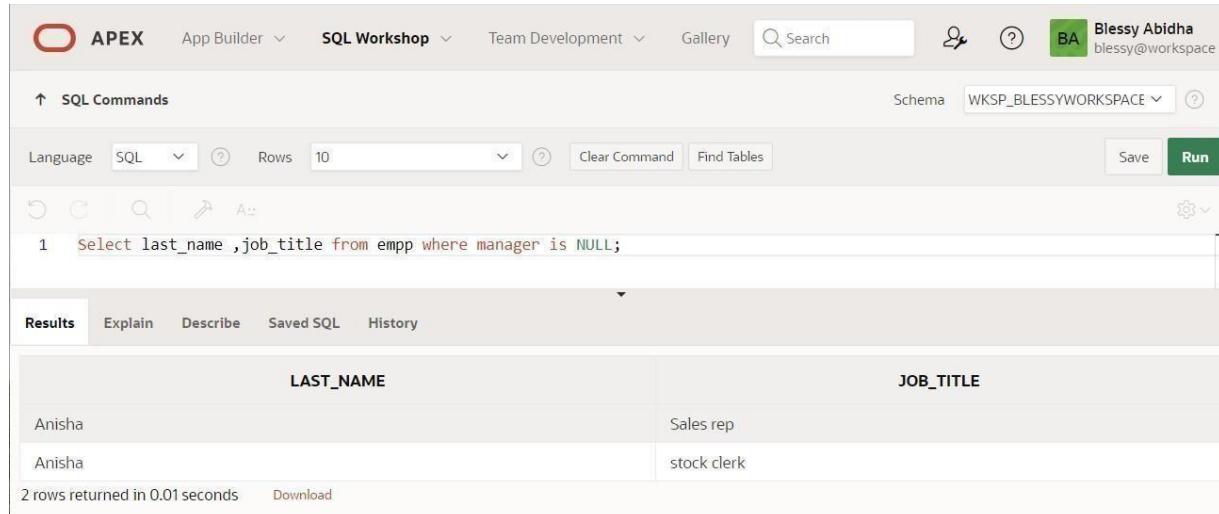
The results table displays two rows, both corresponding to the employee 'Anisha' with a hire date of '02/16/1994'. At the bottom left, it says '2 rows returned in 0.01 seconds'.

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_title from empp where manager is null;

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, Gallery, a search bar, and a user profile for 'BA' (Blessy Abidha). The SQL Workshop tab is selected. The main area shows the SQL command entered: 'Select last_name ,job_title from empp where manager is null;'. Below the command, the results are displayed in a table:

LAST_NAME	JOB_TITLE
Anisha	Sales rep
Anisha	stock clerk

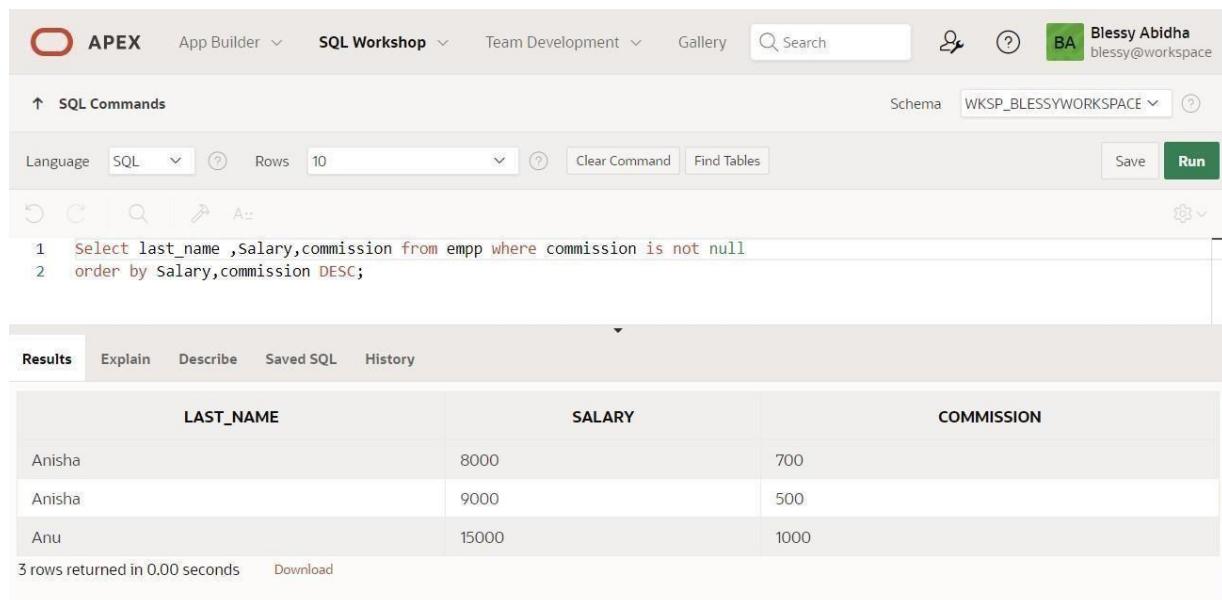
Below the table, it says '2 rows returned in 0.01 seconds' and there is a 'Download' link.

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

QUERY:

Select last_name ,salary,commission from empp where commission is not null order by salary,commission desc;

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, Gallery, a search bar, and a user profile for 'BA Blessy Abidha'. The SQL Workshop tab is selected. The schema dropdown shows 'WKSP_BLESSYWORKSPACE'. The main area displays the SQL command:

```
1 Select last_name ,Salary,commission from empp where commission is not null
2 order by Salary,commission DESC;
```

The results section shows a table with three rows:

LAST_NAME	SALARY	COMMISSION
Anisha	8000	700
Anisha	9000	500
Anu	15000	1000

Below the table, it says '3 rows returned in 0.00 seconds' and has a 'Download' link.

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

QUERY:

Select last_name from empp where last_name like '_a%';

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, Gallery, a search bar, and a user profile for 'BA Blessy Abidha'. The SQL Workshop tab is selected. In the SQL Commands section, the schema is set to 'WKSP_BLESSYWORKSPACE'. The query 'Select last_name from empp where last_name LIKE '_a%';' is entered in the command line. The results section shows a single row with the value 'Anandhi' under the column 'LAST_NAME'. Below the results, it says '1 rows returned in 0.01 seconds'.

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

QUERY:

Select last_name from empp where last_name LIKE '%a%' and last_name like '%e%';

OUTPUT:

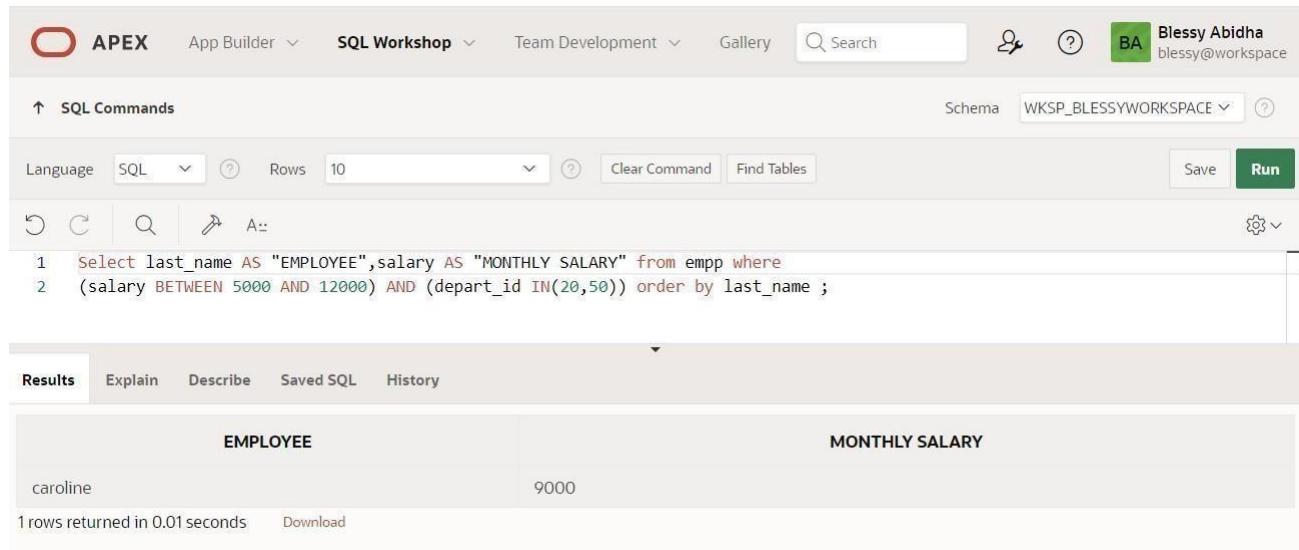
The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, Gallery, a search bar, and a user profile for 'BA Blessy Abidha'. The SQL Workshop tab is selected. In the SQL Commands section, the schema is set to 'WKSP_BLESSYWORKSPACE'. The query 'Select last_name from empp where last_name LIKE '%a%' AND last_name LIKE '%e%';' is entered in the command line. The results section shows a single row with the value 'caroline' under the column 'LAST_NAME'. Below the results, it says '1 rows returned in 0.01 seconds'.

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

QUERY:

Select last_name as "employee",Salary as "monthly salary" from empp where (Salary between 5000 and 12000) and (depart_id in(20,50)) order by last_name ;

OUTPUT:



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

```
1 Select last_name AS "EMPLOYEE",salary AS "MONTHLY SALARY" from empp where
2 (salary BETWEEN 5000 AND 12000) AND (depart_id IN(20,50)) order by last_name ;
```

The Results tab is selected, displaying the output:

EMPLOYEE	MONTHLY SALARY
caroline	9000

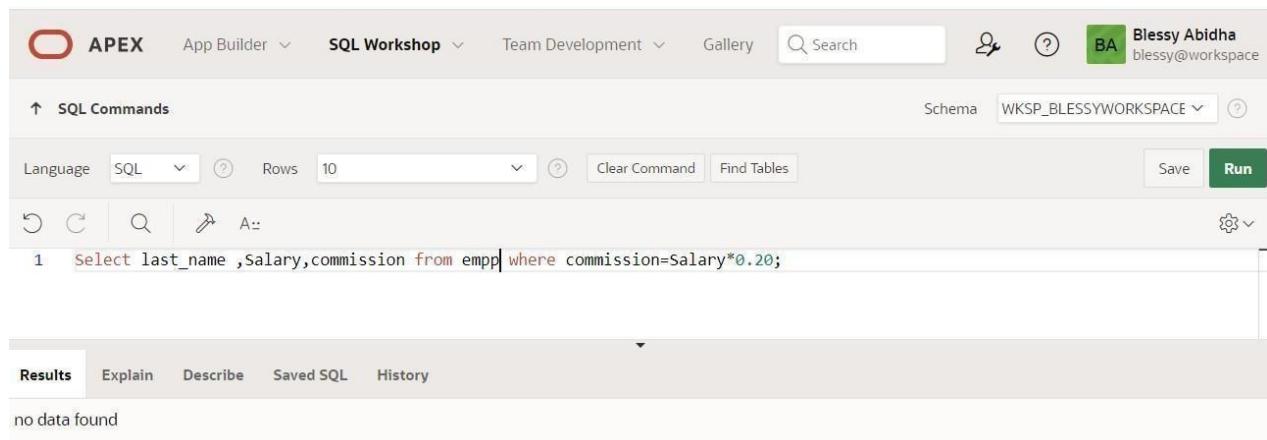
1 rows returned in 0.01 seconds [Download](#)

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 from empp where commission=Salary*0.20;

OUTPUT:



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

```
1 Select last_name ,Salary,commission from empp where commission=Salary*0.20;
```

The Results tab is selected, displaying the output:

EMPLOYEE	MONTHLY SALARY	COMMISSION
caroline	9000	1800

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

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

RESULT:

SINGLE ROW FUNCTIONS

EX.NO.6

DATE:

Find the Solution for the following:

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

QUERY:

SELECT SYSDATE AS "DATE" FROM DUAL;

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The query `SELECT SYSDATE AS "DATE" FROM DUAL;` is entered in the command line. The results pane displays a single row with the column `DATE` containing the value `03/13/2024`.

DATE
03/13/2024

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 e_id, last_name, Salary, Salary+(15.5/100*Salary) "NEW_SALARY" From Empp;

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The query `SELECT e_id, last_name, Salary, Salary+(15.5/100*Salary) "NEW_SALARY" From Empp;` is entered in the command line. The results pane displays a table with columns `E_ID`, `LAST_NAME`, `SALARY`, and `NEW_SALARY`. The data is as follows:

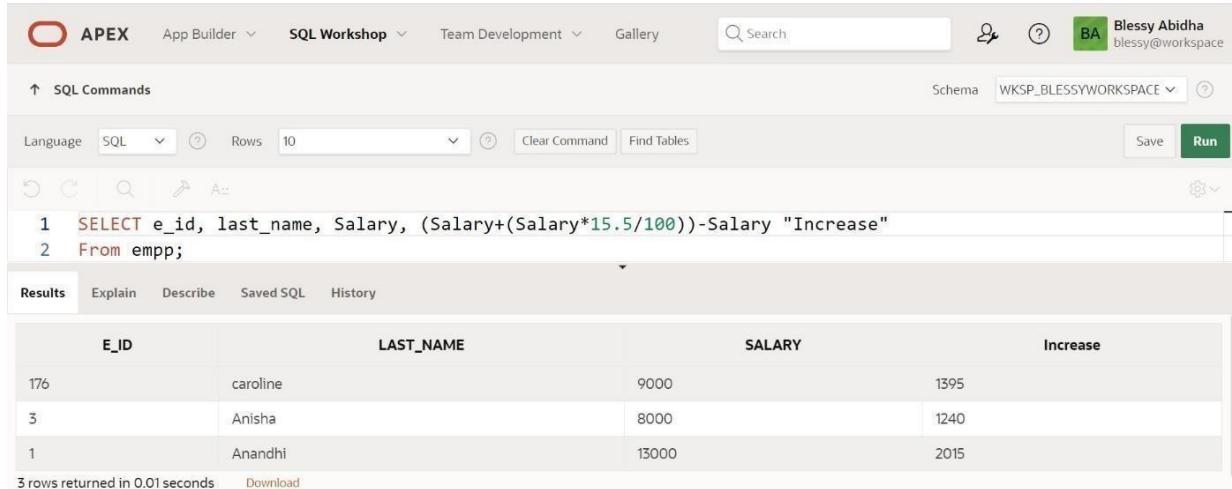
E_ID	LAST_NAME	SALARY	NEW_SALARY
176	caroline	9000	10395
3	Anisha	8000	9240
1	Anandhi	13000	15015

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 e_id, last_name, Salary, (Salary+(Salary*15.5/100))-Salary "Increase"  
From Empp;
```

OUTPUT:



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

```
1 SELECT e_id, last_name, Salary, (Salary+(Salary*15.5/100))-Salary "Increase"  
2 From empp;
```

The results table displays three rows of data:

E_ID	LAST_NAME	SALARY	Increase
176	caroline	9000	1395
3	Anisha	8000	1240
1	Anandhi	13000	2015

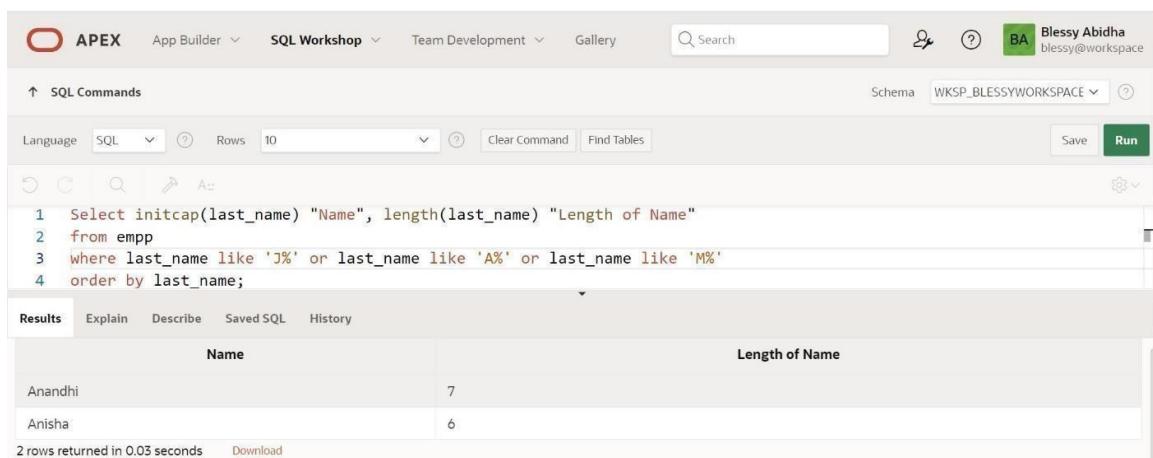
3 rows returned in 0.01 seconds

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) "Name", length(last_name) "Length of Name" from  
Empp  
where last_name like 'J%' or last_name like 'A%' or last_name like 'M%' order  
by last_name;
```

OUTPUT:



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

```
1 Select initcap(last_name) "Name", length(last_name) "Length of Name"  
2 from emp  
3 where last_name like 'J%' or last_name like 'A%' or last_name like 'M%'  
4 order by last_name;
```

The results table displays two rows of data:

Name	Length of Name
Anandhi	7
Anisha	6

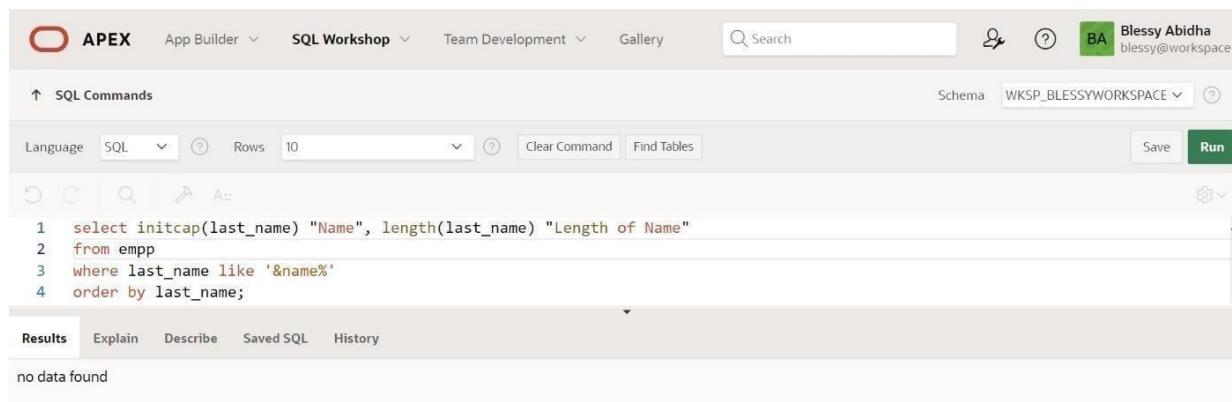
2 rows returned in 0.03 seconds

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

QUERY:

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

OUTPUT:



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

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

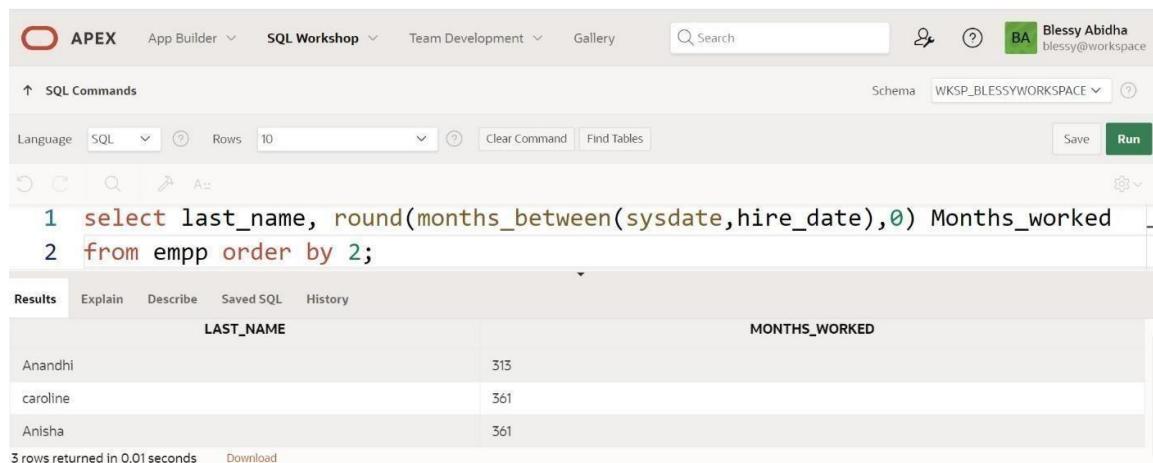
The results tab is selected, showing the message 'no data found'.

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

QUERY:

```
select last_name, round(months_between(sysdate,hire_date),0) Months_worked  
from empp order by 2;
```

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface with the same top navigation and user profile as the previous screenshot. The code editor contains the same SQL query as above. The results tab is selected, displaying the following table:

LAST_NAME	MONTHS_WORKED
Anandhi	315
caroline	361
Anisha	361

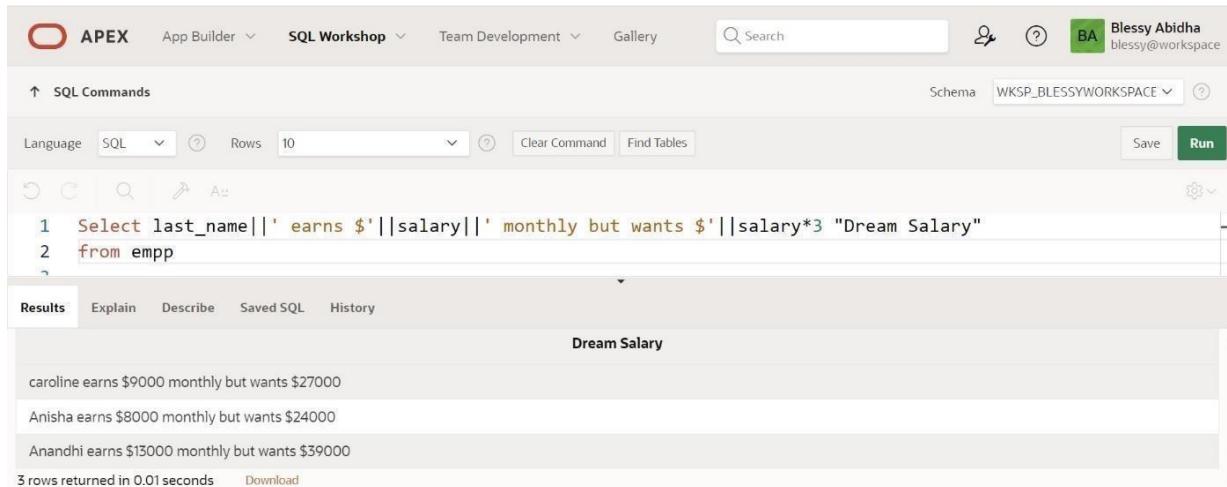
At the bottom, it says '3 rows returned in 0.01 seconds' and has a 'Download' link.

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 "Dream Salary"
from empp;

OUTPUT:



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

```
1 Select last_name||' earns $'||salary||' monthly but wants $'||salary*3 "Dream Salary"
2 from empp
```

The results are displayed in a table with one column labeled "Dream Salary". The output is:

Dream Salary
caroline earns \$9000 monthly but wants \$27000
Anisha earns \$8000 monthly but wants \$24000
Anandhi earns \$13000 monthly but wants \$39000

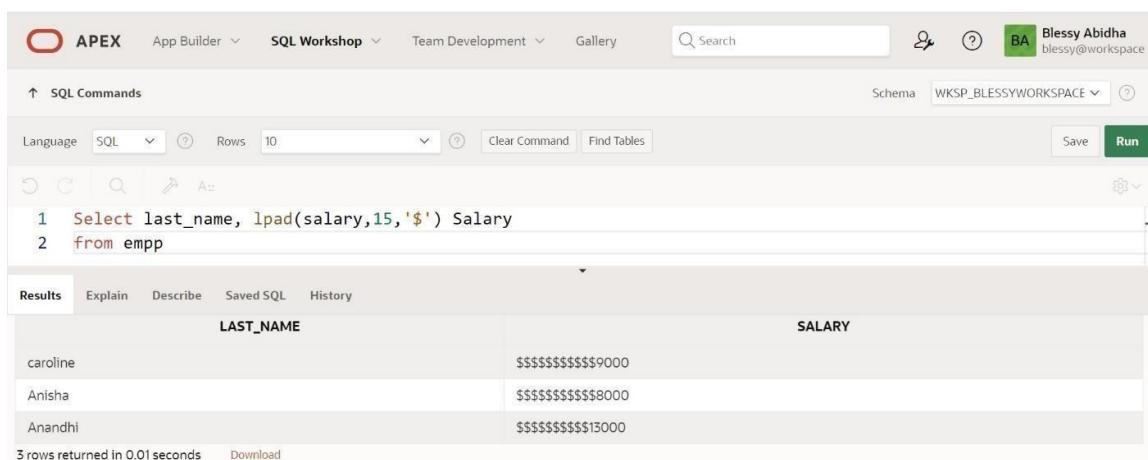
3 rows returned in 0.01 seconds

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

QUERY:

Select last_name, lpad(salary,15,'\$') Salary
from empp;

OUTPUT:



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

```
1 Select last_name, lpad(salary,15,'$') Salary
2 from empp
```

The results are displayed in a table with two columns: "LAST_NAME" and "SALARY". The output is:

LAST_NAME	SALARY
caroline	\$\$\$\$\$\$\$\$\$\$9000
Anisha	\$\$\$\$\$\$\$\$\$\$8000
Anandhi	\$\$\$\$\$\$\$\$\$\$13000

3 rows returned in 0.01 seconds

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

QUERY:

```
select last_name, hire_date, to_char((next_day(hire_date,'Monday')),'fmday," the  
"ddspth "of" month,yyyy') "REVIEW" from empp
```

OUTPUT:

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

```
1 select last_name, hire_date, to_char((next_day(hire_date,'Monday')),'fmday," the  
2 "ddspth "of" month,yyyy') "REVIEW" from empp
```

The results table displays three rows of data:

LAST_NAME	HIRE_DATE	REVIEW
caroline	02/16/1994	monday, the twenty-first of february,1994
Anisha	02/16/1994	monday, the twenty-first of february,1994
Anandhi	02/21/1998	monday, the twenty-third of february,1998

3 rows returned in 0.01 seconds

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

QUERY:

```
Select Last_name, hire_date, to_char(hire_date,'Day') "Day" from  
empp
```

```
order by to_char(hire_date-1,'d')
```

OUTPUT:

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

```
1 Select Last_name, hire_date, to_char(hire_date,'Day') "Day"  
2 from empp  
3 order by to_char(hire_date-1,'d')
```

The results table displays three rows of data:

LAST_NAME	HIRE_DATE	Day
caroline	02/16/1994	Wednesday
Anisha	02/16/1994	Wednesday
Anandhi	02/21/1998	Saturday

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

RESULT:

DISPLAYING DATA FROM MULTIPLE TABLES

EX_NO:7

DATE:

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

QUERY:

```
SELECT E.LAST_NAME, E.DEPARTMENT_ID, D.DEPT_NAME  
FROM EMPLOYEES E, DEPARTMENT D  
WHERE E.DEPARTMENT_ID = D.DEPT_ID;
```

OUTPUT:

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

```
47  SELECT E.LAST_NAME, E.DEPARTMENT_ID, D.DEPT_NAME  
48  FROM EMPLOYEES E, DEPARTMENT D  
49  WHERE E.DEPARTMENT_ID = D.DEPT_ID;
```

The Results tab displays the query output as a table:

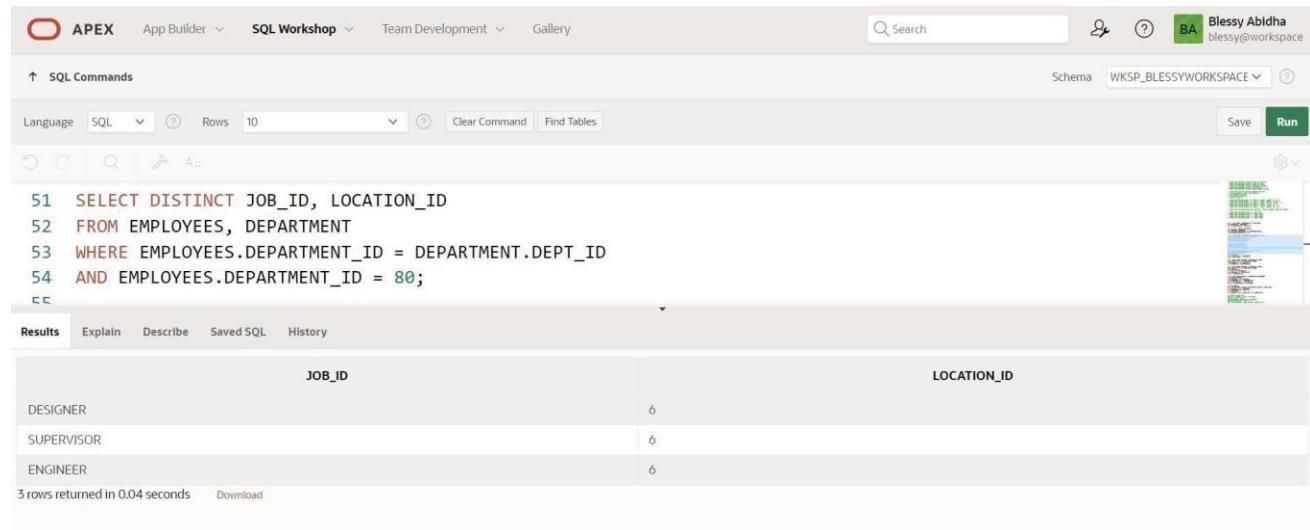
LAST_NAME	DEPARTMENT_ID	DEPT_NAME
PARTHI	20	STOCK
JANE	50	HR
KOHLI	80	MANUFACTURING
DAVIES	30	FINANCE
JAY	10	MARKETING
EMANUEL	30	FINANCE
JAM	80	MANUFACTURING

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, LOCATION_ID  
FROM EMPLOYEES, DEPARTMENT  
WHERE EMPLOYEES.DEPARTMENT_ID = DEPARTMENT.DEPT_ID  
AND EMPLOYEES.DEPARTMENT_ID = 80;
```

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there's a user profile for 'Blessy Abidha' (blessy@workspace). The main area displays the SQL command and its execution results.

SQL Commands:

```
51  SELECT DISTINCT JOB_ID, LOCATION_ID  
52  FROM EMPLOYEES, DEPARTMENT  
53  WHERE EMPLOYEES.DEPARTMENT_ID = DEPARTMENT.DEPT_ID  
54  AND EMPLOYEES.DEPARTMENT_ID = 80;
```

Results:

JOB_ID	LOCATION_ID
DESIGNER	6
SUPERVISOR	6
ENGINEER	6

3 rows returned in 0.04 seconds Download

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

QUERY:

```
SELECT E.LAST_NAME, D.DEPARTMENT_NAME, D.LOCATION_ID, L.CITY  
FROM EMPLOYEES E, DEPARTMENT D, LOCATION L  
WHERE DEPARTMENT_ID = DEPT_ID  
AND D.LOCATION_ID = L.LOCATION_ID  
AND COMMISSION_PCT IS NOT NULL;
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. At the top, there are tabs for 'APEX', 'App Builder', 'SQL Workshop' (which is selected), 'Team Development', and 'Gallery'. On the right, there's a user profile for 'Blessy Abidha' with the email 'blessy@workspace'. Below the tabs, there's a search bar and a schema dropdown set to 'WKSP_BLESSYWORKSPACE'. The main area has tabs for 'SQL Commands' (selected), 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History'. The 'SQL Commands' tab contains the following SQL code:

```
>>  
56  SELECT E.LAST_NAME, D.DEPARTMENT_NAME, D.LOCATION_ID, L.CITY  
57  FROM EMPLOYEES E, DEPARTMENT D, LOCATION L  
58  WHERE DEPARTMENT_ID = DEPT_ID  
59  AND D.LOCATION_ID = L.LOCATION_ID  
60  AND COMMISSION_PCT IS NOT NULL;
```

The 'Results' tab displays the output of the query as a table:

LAST_NAME	DEPT_NAME	LOCATION_ID	CITY
JANE	HR	5	LONDON
KOHLI	MANUFACTURING	6	TORONTO
DAVIES	FINANCE	3	VALHALLA
JAY	MARKETING	1	CHENNAI
EMANUEL	FINANCE	3	VALHALLA
JAM	MANUFACTURING	6	TORONTO
DEV	MANUFACTURING	6	TORONTO
RAVI	MARKETING	1	CHENNAI
VIJAY	MANAGEMENT	4	DC

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

QUERY:

```
SELECT LAST_NAME, DEPT_NAME  
FROM EMPLOYEES, DEPARTMENT  
WHERE DEPARTMENT_ID = DEPT_ID  
AND LAST_NAME LIKE '%a%';
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. On the right, there's a user profile for 'Blessy Abidha' and a schema dropdown for 'WKSP_BLESSYWORKSPACE'. The main area has tabs for 'SQL Commands' (selected), 'Explain', 'Describe', 'Saved SQL', and 'History'. The SQL command entered is:

```
61  
62 SELECT LAST_NAME, DEPT_NAME  
63 FROM EMPLOYEES, DEPARTMENT  
64 WHERE DEPARTMENT_ID = DEPT_ID  
65 AND LAST_NAME LIKE '%a%'; -- SHOULD BE '%a' BUT SINCE I ENTERED THE DATAS IN CAPS, O/P COMES TO BE NO DATA FOUND  
66
```

The results table has two columns: 'LAST_NAME' and 'DEPT_NAME'. The data is as follows:

LAST_NAME	DEPT_NAME
PARTHI	STOCK
JANE	HR
DAVIES	FINANCE
JAY	MARKETING
EMANUEL	FINANCE
JAM	MANUFACTURING
RAVI	MARKETING
UMA	STOCK
VIJAY	MANAGEMENT

At the bottom, there are footer links for '220701047@rajalakshmi.edu.in', 'blessy@workspace', and 'en'. The copyright notice reads 'Copyright © 1999-2023, Oracle and/or its affiliates.' and the version 'Oracle APEX 23.2.4'.

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', 'Gallery', a search bar, and a user profile for 'Blessy Abidha'. Below the toolbar, the schema is set to 'WKSP_BLESSYWORKSPACE'. The main area displays the SQL command entered by the user, which selects data from the EMPLOYEES, DEPARTMENT, and LOCATION tables where the city is 'toronto'. The results section shows three rows of data:

LAST_NAME	JOB_ID	DEPARTMENT_ID	DEPT_NAME
KOHLI	SUPERVISOR	80	MANUFACTURING
JAM	DESIGNER	80	MANUFACTURING
DEV	ENGINEER	80	MANUFACTURING

At the bottom, it indicates '3 rows returned in 0.02 seconds' and provides a 'Download' link.

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. The right side shows a user profile for 'Blessy Abidha' with the email 'blessy@workspace'. The main area has tabs for 'SQL Commands' and 'Results'. In the 'SQL Commands' tab, the query is displayed with line numbers 73 through 77. The 'Results' tab displays the output in a grid format.

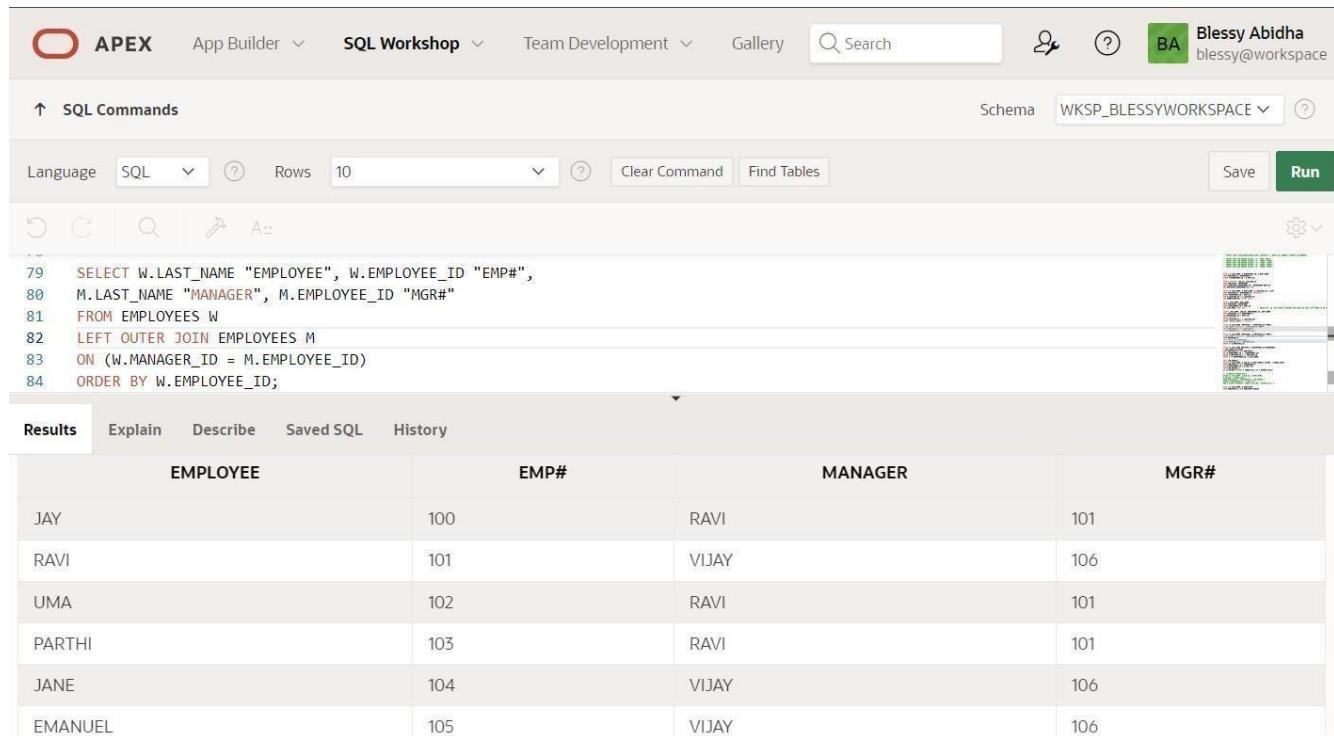
EMPLOYEE	EMP#	MANAGER	MGR#
DAVIES	110	EMANUEL	105
DEV	108	JAM	107
PARTHI	103	RAVI	101
JAY	100	RAVI	101
UMA	102	RAVI	101
JANE	104	VIJAY	106
KOHLI	109	VIJAY	106

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

QUERY:

```
SELECT W.LAST_NAME "EMPLOYEE", W.EMPLOYEE_ID "EMP#",  
M.LAST_NAME "MANAGER", M.EMPLOYEE_ID "MGR#"  
FROM EMPLOYEES W  
LEFT OUTER JOIN EMPLOYEES M  
ON (W.MANAGER_ID = M.EMPLOYEE_ID)  
ORDER BY W.EMPLOYEE_ID;
```

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. At the top, there are tabs for APEX, App Builder, SQL Workshop (which is selected), Team Development, and Gallery. A search bar and user information for 'Blessy Abidha' are also at the top. Below the tabs, there's a toolbar with icons for Undo, Redo, Find, and others. The main area contains the SQL code and its execution results.

SQL Commands

Language: SQL Rows: 10 Schema: WKSP_BLESSYWORKSPACE

79: SELECT W.LAST_NAME "EMPLOYEE", W.EMPLOYEE_ID "EMP#",
80: M.LAST_NAME "MANAGER", M.EMPLOYEE_ID "MGR#"
81: FROM EMPLOYEES W
82: LEFT OUTER JOIN EMPLOYEES M
83: ON (W.MANAGER_ID = M.EMPLOYEE_ID)
84: ORDER BY W.EMPLOYEE_ID;

Results

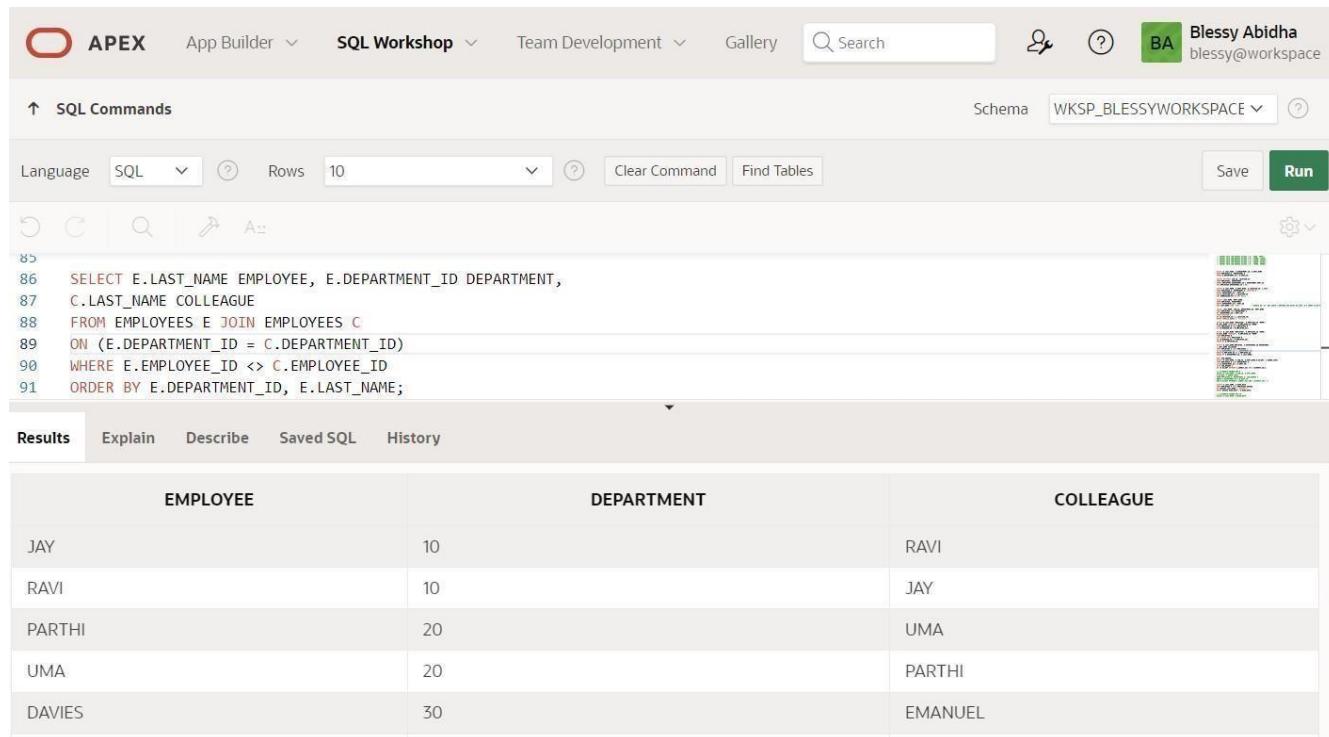
EMPLOYEE	EMP#	MANAGER	MGR#
JAY	100	RAVI	101
RAVI	101	VIJAY	106
UMA	102	RAVI	101
PARTHI	103	RAVI	101
JANE	104	VIJAY	106
EMANUEL	105	VIJAY	106

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

QUERY:

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

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, Gallery, a search bar, and a user profile for 'Blessy Abidha'.

The SQL Commands tab is active, displaying the following query:

```
85
86  SELECT E.LAST_NAME EMPLOYEE, E.DEPARTMENT_ID DEPARTMENT,
87  C.LAST_NAME COLLEAGUE
88  FROM EMPLOYEES E JOIN EMPLOYEES C
89  ON (E.DEPARTMENT_ID = C.DEPARTMENT_ID)
90  WHERE E.EMPLOYEE_ID <> C.EMPLOYEE_ID
91  ORDER BY E.DEPARTMENT_ID, E.LAST_NAME;
```

The Results tab is selected, showing the output of the query:

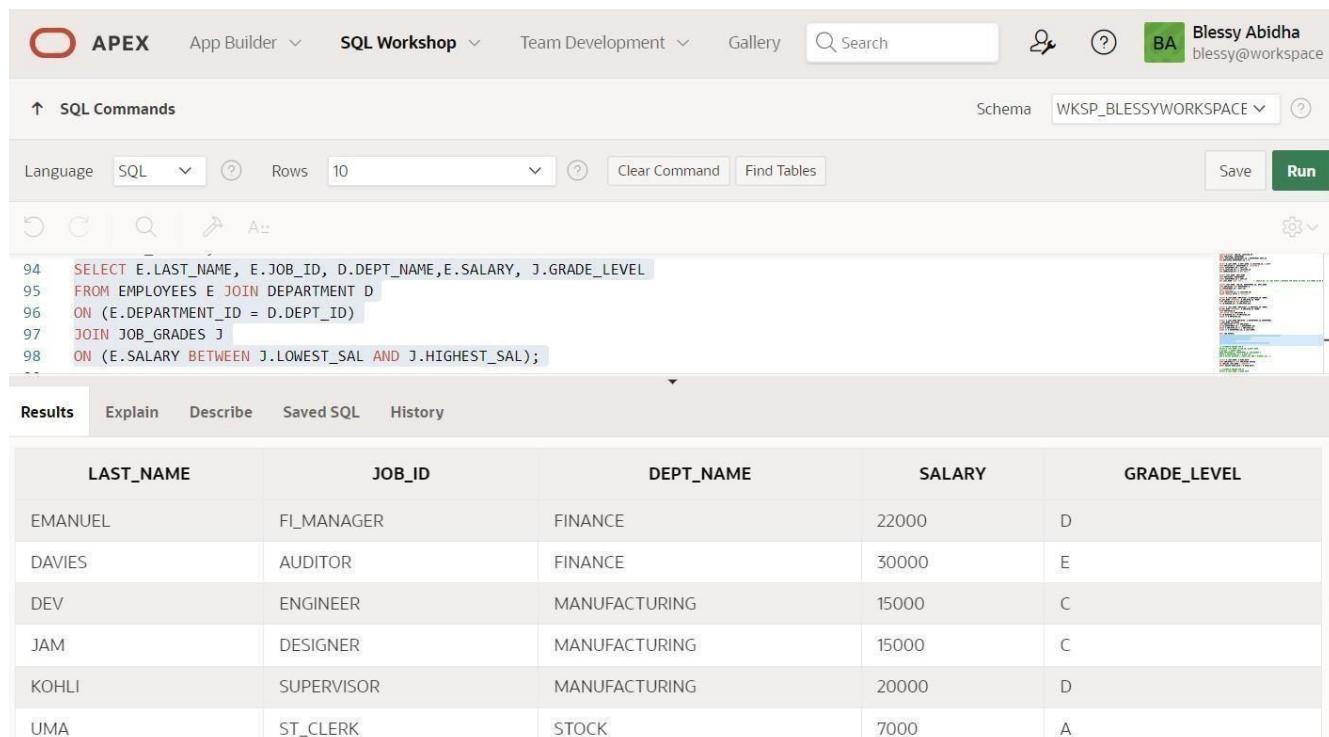
EMPLOYEE	DEPARTMENT	COLLEAGUE
JAY	10	RAVI
RAVI	10	JAY
PARTHI	20	UMA
UMA	20	PARTHI
DAVIES	30	EMANUEL

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

QUERY:

```
SELECT E.LAST_NAME, E.JOB_ID, D.DEPT_NAME, E.SALARY, J.GRADE_LEVEL  
  
FROM EMPLOYEES E JOIN DEPARTMENT D  
  
ON (E.DEPARTMENT_ID = D.DEPT_ID)  
  
JOIN JOB_GRADES J  
  
ON (E.SALARY BETWEEN J.LOWEST_SAL AND J.HIGHEST_SAL);
```

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', 'Gallery', a search bar, and a user profile for 'Blessy Abidha'.

The main area is titled 'SQL Commands'. It shows the following SQL code:

```
94 SELECT E.LAST_NAME, E.JOB_ID, D.DEPT_NAME, E.SALARY, J.GRADE_LEVEL  
95 FROM EMPLOYEES E JOIN DEPARTMENT D  
96 ON (E.DEPARTMENT_ID = D.DEPT_ID)  
97 JOIN JOB_GRADES J  
98 ON (E.SALARY BETWEEN J.LOWEST_SAL AND J.HIGHEST_SAL);
```

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

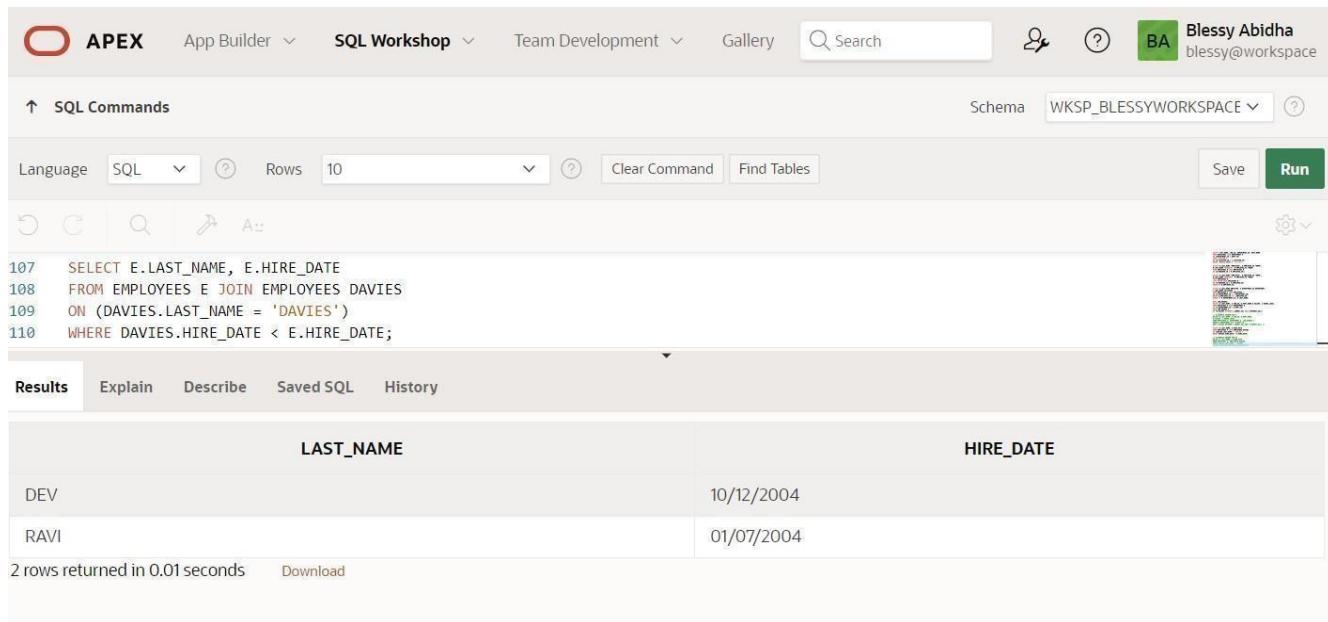
LAST_NAME	JOB_ID	DEPT_NAME	SALARY	GRADE_LEVEL
EMANUEL	FI_MANAGER	FINANCE	22000	D
DAVIES	AUDITOR	FINANCE	30000	E
DEV	ENGINEER	MANUFACTURING	15000	C
JAM	DESIGNER	MANUFACTURING	15000	C
KOHLI	SUPERVISOR	MANUFACTURING	20000	D
UMA	ST_CLERK	STOCK	7000	A

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

QUERY:

```
SELECT E.LAST_NAME, E.HIRE_DATE  
  
FROM EMPLOYEES E JOIN EMPLOYEES DAVIES  
  
ON (DAVIES.LAST_NAME = 'DAVIES')  
  
WHERE DAVIES.HIRE_DATE < E.HIRE_DATE;
```

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', 'Gallery', a search bar, and a user profile for 'Blessy Abidha'. The main area has tabs for 'SQL Commands' and 'Results'. In the 'SQL Commands' tab, the query is pasted. In the 'Results' tab, the output is displayed in a table:

LAST_NAME	HIRE_DATE
DEV	10/12/2004
RAVI	01/07/2004

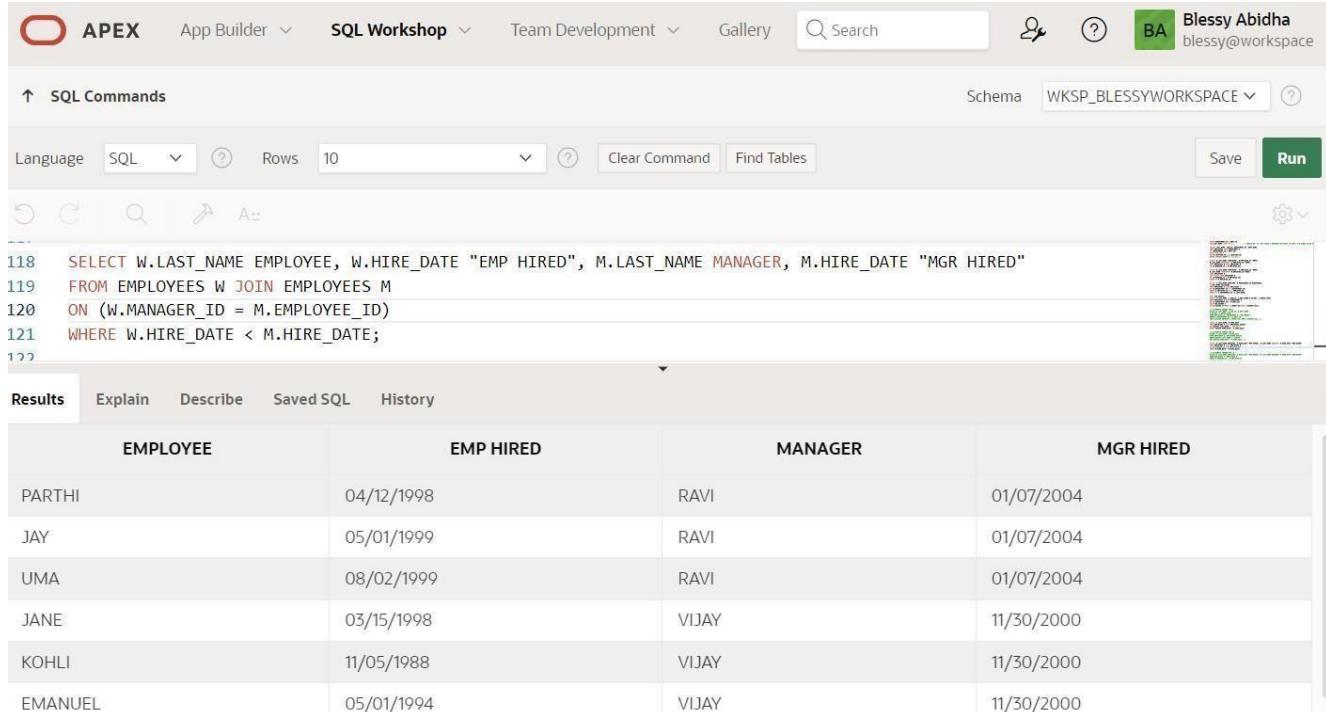
Below the table, it says '2 rows returned in 0.01 seconds' and there is a 'Download' link.

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

QUERY:

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

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, Gallery, a search bar, and user information for 'Blessy Abidha' (blessy@workspace). Below the navigation is a toolbar with icons for Undo, Redo, Find, and Refresh, followed by a dropdown for rows (set to 10), Clear Command, Find Tables, Save, and Run buttons. The main area displays the SQL code and its execution results.

```
--  
118 SELECT W.LAST_NAME EMPLOYEE, W.HIRE_DATE "EMP HIRED", M.LAST_NAME MANAGER, M.HIRE_DATE "MGR HIRED"  
119 FROM EMPLOYEES W JOIN EMPLOYEES M  
120 ON (W.MANAGER_ID = M.EMPLOYEE_ID)  
121 WHERE W.HIRE_DATE < M.HIRE_DATE;  
122
```

The Results tab is selected, showing the following data:

EMPLOYEE	EMP HIRED	MANAGER	MGR HIRED
PARTHI	04/12/1998	RAVI	01/07/2004
JAY	05/01/1999	RAVI	01/07/2004
UMA	08/02/1999	RAVI	01/07/2004
JANE	03/15/1998	VIJAY	11/30/2000
KOHLI	11/05/1988	VIJAY	11/30/2000
EMANUEL	05/01/1994	VIJAY	11/30/2000

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) AS "MAXIMUM", ROUND(MIN(SALARY),0) AS "MINIMUM", ROUND(SUM(SALARY),0) AS "SUM",  
ROUND(AVG(SALARY),0) AS "AVERAGE" FROM EMPLOYEES;
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, Gallery, and a search bar. On the right, there's a user profile for 'Blessy Abidha' with the email 'blessy@workspace'. Below the navigation is a toolbar with various icons for SQL operations like SELECT, INSERT, UPDATE, DELETE, and others. The main area is titled 'SQL Commands' and contains the following SQL code:

```
6  SELECT ROUND(MAX(SALARY),0) AS "MAXIMUM", ROUND(MIN(SALARY),0) AS "MINIMUM", ROUND(SUM(SALARY),0) AS "SUM",  
7  ROUND(AVG(SALARY),0) AS "AVERAGE" FROM EMPLOYEES;
```

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

MAXIMUM	MINIMUM	SUM	AVERAGE
33400	5500	233351	17950

At the bottom left, it says '1 rows returned in 0.00 seconds' and there's a 'Download' link.

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. 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 'Blessy Abidha' are also at the top. Below the tabs, there's a toolbar with icons for SQL Commands, Language (set to SQL), Rows (set to 10), Clear Command, Find Tables, Save, and Run. The main area contains the SQL code and its results. The SQL code is:

```
9  SELECT JOB_ID,ROUND(MAX(SALARY),0) AS "MAXIMUM", ROUND(MIN(SALARY),0) AS "MINIMUM", ROUND(SUM(SALARY),0) AS "SUM",  
10 ROUND(AVG(SALARY),0) AS "AVERAGE" FROM EMPLOYEES GROUP BY JOB_ID;  
11
```

The results section displays a table with the following data:

JOB_ID	MAXIMUM	MINIMUM	SUM	AVERAGE
FI_MANAGER	22000	22000	22000	22000
DEVELOPER	20000	20000	20000	20000
ST_CLERK	8651	7000	15651	7825
SUPERVISOR	20000	20000	20000	20000
MANAGER	33400	33400	33400	33400
ENGINEER	15000	15000	15000	15000
MK_MANAGER	20000	20000	20000	20000

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, Gallery, a search bar, and a user profile for 'Blessy Abidha'.

The SQL Commands tab is active, displaying the following SQL code:

```
13  SELECT JOB_ID,COUNT(JOB_ID) FROM EMPLOYEES  
14  GROUP BY JOB_ID;  
15
```

The Results tab is selected, showing the output of the query:

JOB_ID	COUNT(JOB_ID)
FI_MANAGER	1
DEVELOPER	1
ST_CLERK	2
SUPERVISOR	1
MANAGER	1
ENGINEER	1

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

Use the MANAGER_ID column to determine the number of managers.

QUERY:

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

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, Gallery, a search bar, and user information for 'Blessy Abidha' (blessy@workspace). The main area is titled 'SQL Commands'. The query entered is:

```
16  SELECT COUNT(DISTINCT MANAGER_ID) "NUMBER OF MANAGERS"  
17  FROM EMPLOYEES;
```

The results tab is selected, displaying the output:

NUMBER OF MANAGERS
5

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

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, Gallery, a search bar, and user information for 'Blessy Abidha' (blessy@workspace). The main area is titled 'SQL Commands'. The query entered is:

```
18  
19  SELECT MAX(SALARY)-MIN(SALARY) AS "DIFFERENCE" FROM EMPLOYEES;  
20
```

The results tab is selected, displaying the output:

DIFFERENCE
27900.12

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

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

QUERY:

```
SELECT MANAGER_ID, MIN(SALARY) FROM EMPLOYEES WHERE MANAGER_ID IS NOT NULL GROUP BY MANAGER_ID
```

```
HAVING MIN(SALARY)> 6000 ORDER BY MIN(SALARY) DESC;
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, Gallery, a search bar, and a user profile for 'Blessy Abidha'. The SQL Commands tab is active, showing the following SQL code:

```
21  SELECT MANAGER_ID,MIN(SALARY)
22  FROM EMPLOYEES
23  WHERE MANAGER_ID IS NOT NULL
24  GROUP BY MANAGER_ID
25  HAVING MIN(SALARY) > 6000
26  ORDER BY MIN(SALARY) DESC;
```

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

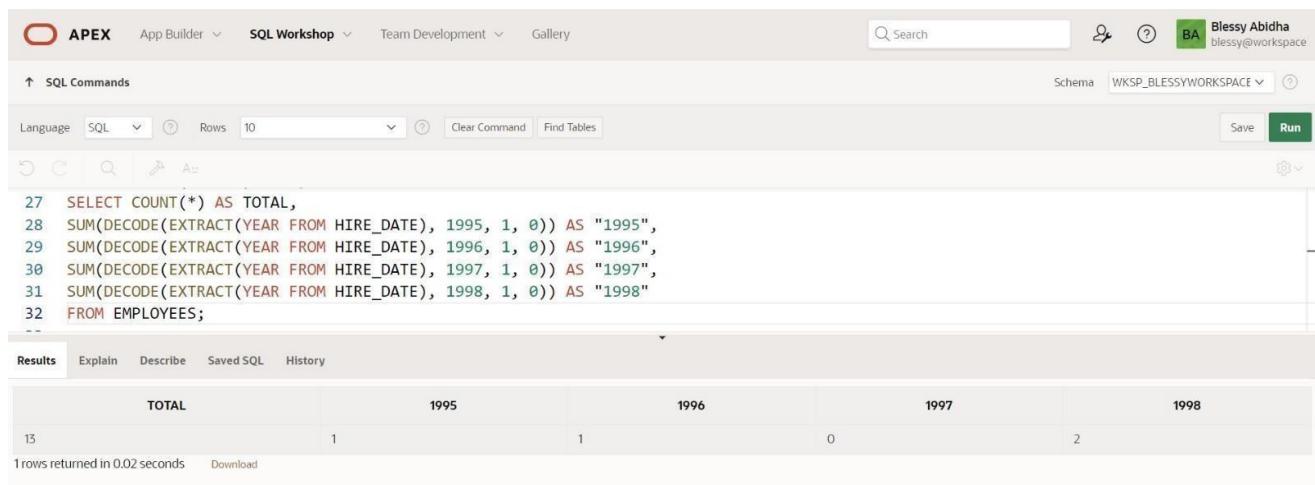
MANAGER_ID	MIN(SALARY)
105	30000
106	15000
107	15000
101	7000

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

QUERY:

```
SELECT COUNT(*) AS TOTAL, SUM (DECODE(EXTRACT(YEAR FROM HIRE_DATE), 1995, 1, 0)) AS "1995",
       SUM (DECODE(EXTRACT(YEAR FROM HIRE_DATE), 1996, 1, 0)) AS "1996",
       SUM (DECODE(EXTRACT(YEAR FROM HIRE_DATE), 1997, 1, 0)) AS "1997",
       SUM (DECODE(EXTRACT(YEAR FROM HIRE_DATE), 1998, 1, 0)) AS "1998"
FROM EMPLOYEES.
```

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there's a user profile for 'Blessy Abidha' (blessy@workspace). The main area is titled 'SQL Commands'. The SQL editor contains the query from the previous step. The results tab is selected, displaying the following output:

	TOTAL	1995	1996	1997	1998
13	1	1	0	2	

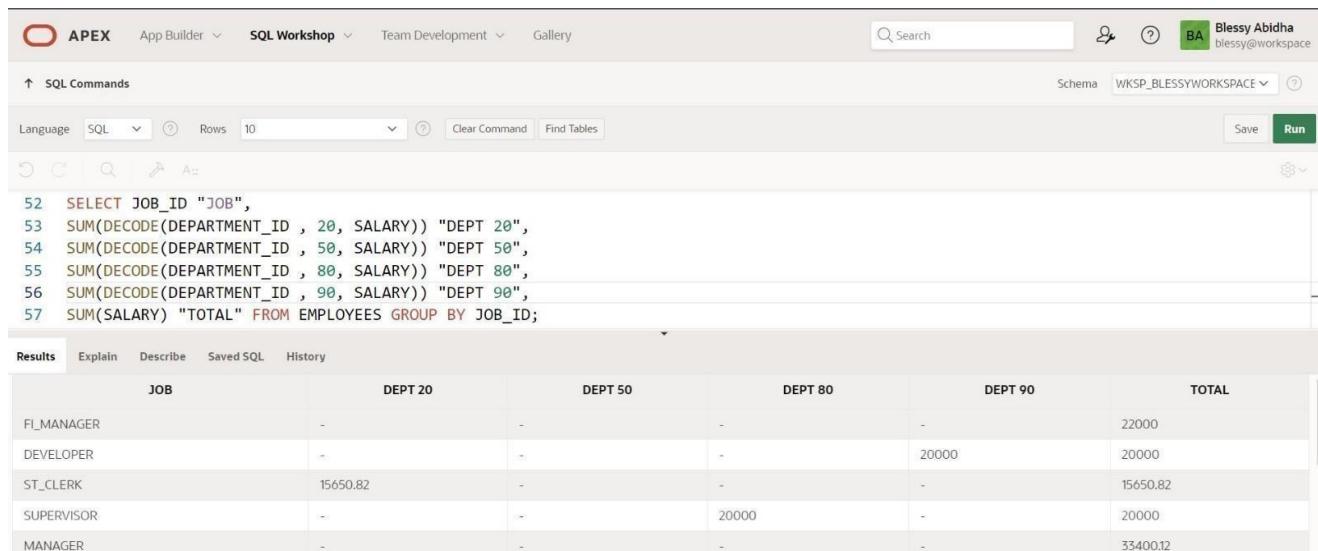
1 rows returned in 0.02 seconds [Download](#)

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

QUERY:

```
SELECT JOB_ID "JOB",
       SUM(DECODE(DEPARTMENT_ID , 20, SALARY)) "DEPT 20",
       SUM(DECODE(DEPARTMENT_ID , 50, SALARY)) "DEPT 50",
       SUM(DECODE(DEPARTMENT_ID , 80, SALARY)) "DEPT 80",
       SUM(DECODE(DEPARTMENT_ID , 90, SALARY)) "DEPT 90",
       SUM(SALARY) "TOTAL" FROM EMPLOYEES GROUP BY JOB_ID;
```

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there's a user profile for 'Blessy Abidha' (blessy@workspace). The main area has tabs for SQL Commands, Results (selected), Explain, Describe, Saved SQL, and History. The SQL Commands tab shows the executed query. The Results tab displays the output in a grid format.

JOB	DEPT 20	DEPT 50	DEPT 80	DEPT 90	TOTAL
FL_MANAGER	-	-	-	-	22000
DEVELOPER	-	-	-	20000	20000
ST_CLERK	15650.82	-	-	-	15650.82
SUPERVISOR	-	-	20000	-	20000
MANAGER	-	-	-	-	33400.12

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there is a user profile for 'Blessy Abidha' (blessy@iworkspace). The main workspace has a search bar and various toolbar icons. The SQL command area contains the following code:

```
60  SELECT D.DEPT_NAME "NAME",
61  D.LOCATION_ID "LOCATION ", COUNT(*) "NUMBER OF PEOPLE",
62  ROUND(AVG(SALARY),2) "SALARY" FROM EMPLOYEES E,
63  DEPARTMENT D WHERE E.DEPARTMENT_ID = D.DEPARTMENT_ID
64  GROUP BY D.DEPT_NAME, D.LOCATION_ID;
65
```

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

NAME	LOCATION	NUMBER OF PEOPLE	SALARY
STOCK	2	2	7825.41
FINANCE	3	2	26000
MARKETING	1	2	16700.06
IT	5	2	12750
HR	5	1	23400.12

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

RESULT:

SUB-QUERIES

EX.NO:9

DATE:

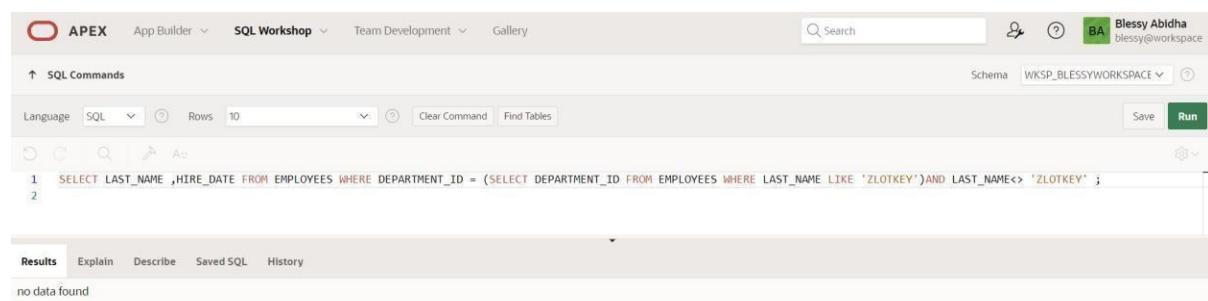
Find the Solution for the following:

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 like 'Zlotkey')and  
last_name <> 'Zlotkey';
```

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there's a user profile for 'Blessy Abidha' and a workspace dropdown for 'WKSP_BLESSYWORKSPACE'. The main area has tabs for SQL Commands, Results (selected), Explain, Describe, Saved SQL, and History. The SQL editor contains the following code:

```
1  SELECT LAST_NAME ,HIRE_DATE FROM EMPLOYEES WHERE DEPARTMENT_ID = (SELECT DEPARTMENT_ID FROM EMPLOYEES WHERE LAST_NAME LIKE 'ZLOTKEY')AND LAST_NAME<> 'ZLOTKEY' ;  
2
```

The results tab shows the output: "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 SQL Workshop interface. At the top, there's a navigation bar with 'APEX', 'App Builder', 'SQL Workshop' (which is selected), 'Team Development', and 'Gallery'. On the right side of the header, there's a user profile for 'Blessy Abidha' with the email 'blessy@workspace'. Below the header, the main area has tabs for 'SQL Commands' and 'Results'. Under 'SQL Commands', the following SQL code is visible:

```
14 SELECT EMPLOYEE_ID, LAST_NAME, SALARY FROM EMPLOYEES  
15 WHERE SALARY > (SELECT AVG(SALARY) FROM EMPLOYEES)  
16 ORDER BY SALARY;
```

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

EMPLOYEE_ID	LAST_NAME	SALARY
111	JOBSS	20000
109	KÖHLI	20000
101	RAVI	20000
105	EMANUEL	22000
104	JANE	23400.12
110	DAVIES	30000
106	VIJAY	33400.12

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 dept_id in (select dept_id from employees where last_name like  
'%u%');
```

OUTPUT:

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

```
18  SELECT EMPLOYEE_ID, LAST_NAME  
19  FROM   EMPLOYEES A  
20  JOIN (SELECT DEPARTMENT_ID FROM EMPLOYEES WHERE LAST_NAME LIKE '%u%') B  
21  ON A.DEPARTMENT_ID = B.DEPARTMENT_ID;
```

The results table displays the following data:

EMPLOYEE_ID	LAST_NAME
105	PARTHI
110	DAVIES
105	EMANUEL
102	UMA

4 rows returned in 0.02 seconds

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

QUERY:

```
select last_name, department_id, job_id  
from employees  
where dept_id in (select dept_id from department where loc_id =1700);
```

OUTPUT:

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

```
28  
29  SELECT LAST_NAME, DEPARTMENT_ID, JOB_ID  
30  FROM EMPLOYEES A  
31  JOIN (SELECT DEPT_ID FROM DEPARTMENT WHERE LOCATION_ID=1700) B  
32  ON A.DEPARTMENT_ID=B.DEPARTMENT_ID;  
33
```

The results table displays the following data:

LAST_NAME	DEPARTMENT_ID	JOB_ID
no data found		

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_no in (select employee_id from employees where  
last_name='King');
```

OUTPUT:



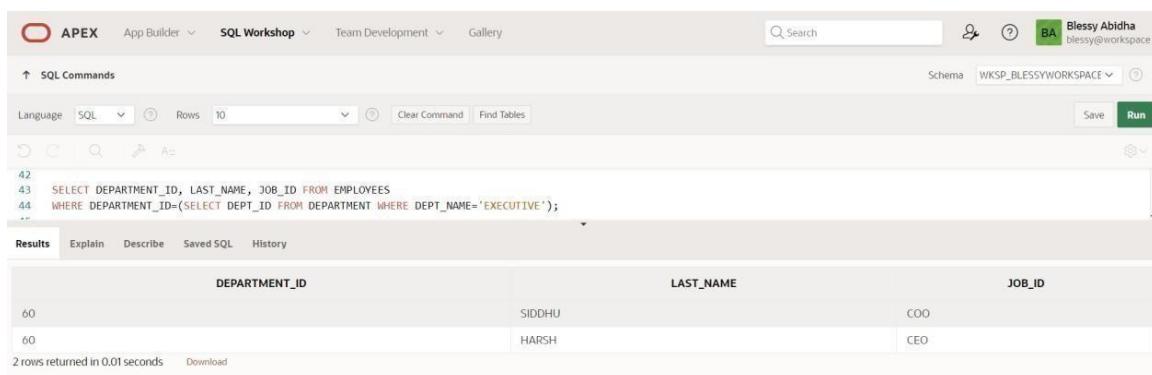
```
43  SELECT LAST_NAME,SALARY FROM EMPLOYEES  
44  WHERE MANAGER_ID=(SELECT EMPLOYEE_ID FROM EMPLOYEES WHERE LAST_NAME='KING');  
45
```

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

OUTPUT:



DEPARTMENT_ID	LAST_NAME	JOB_ID
60	SIDDHU	COO
60	HARSH	CEO

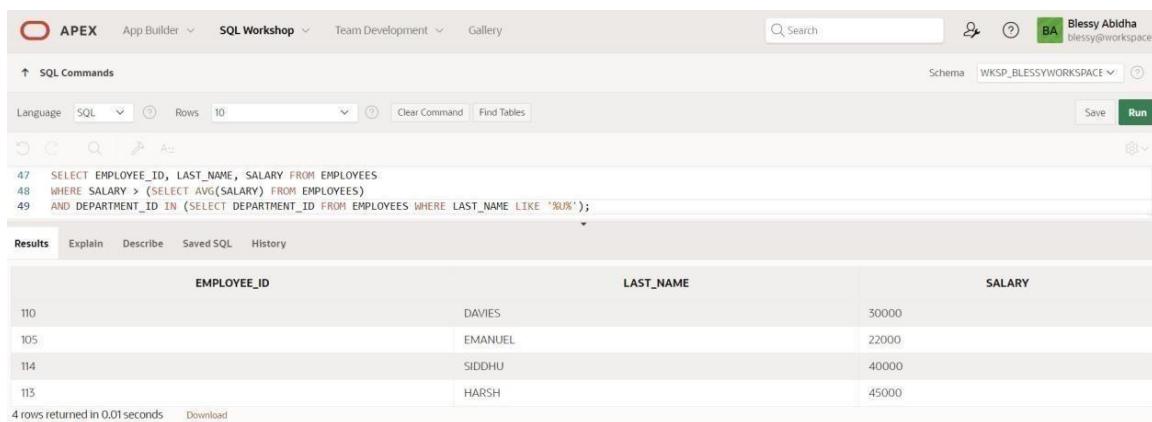
2 rows returned in 0.01 seconds Download

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

QUERY:

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

OUTPUT:



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

```
47  SELECT EMPLOYEE_ID, LAST_NAME, SALARY FROM EMPLOYEES  
48  WHERE SALARY > (SELECT AVG(SALARY) FROM EMPLOYEES)  
49  AND DEPARTMENT_ID IN (SELECT DEPARTMENT_ID FROM EMPLOYEES WHERE LAST_NAME LIKE '%u%');
```

The Results tab displays the output of the query:

EMPLOYEE_ID	LAST_NAME	SALARY
110	DAVIES	30000
105	EMANUEL	22000
114	SIDDHU	40000
113	HARSH	45000

Below the table, it says "4 rows returned in 0.01 seconds".

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

RESULT:

USING THE SET OPERATORS

EX.NO:10

DATE:

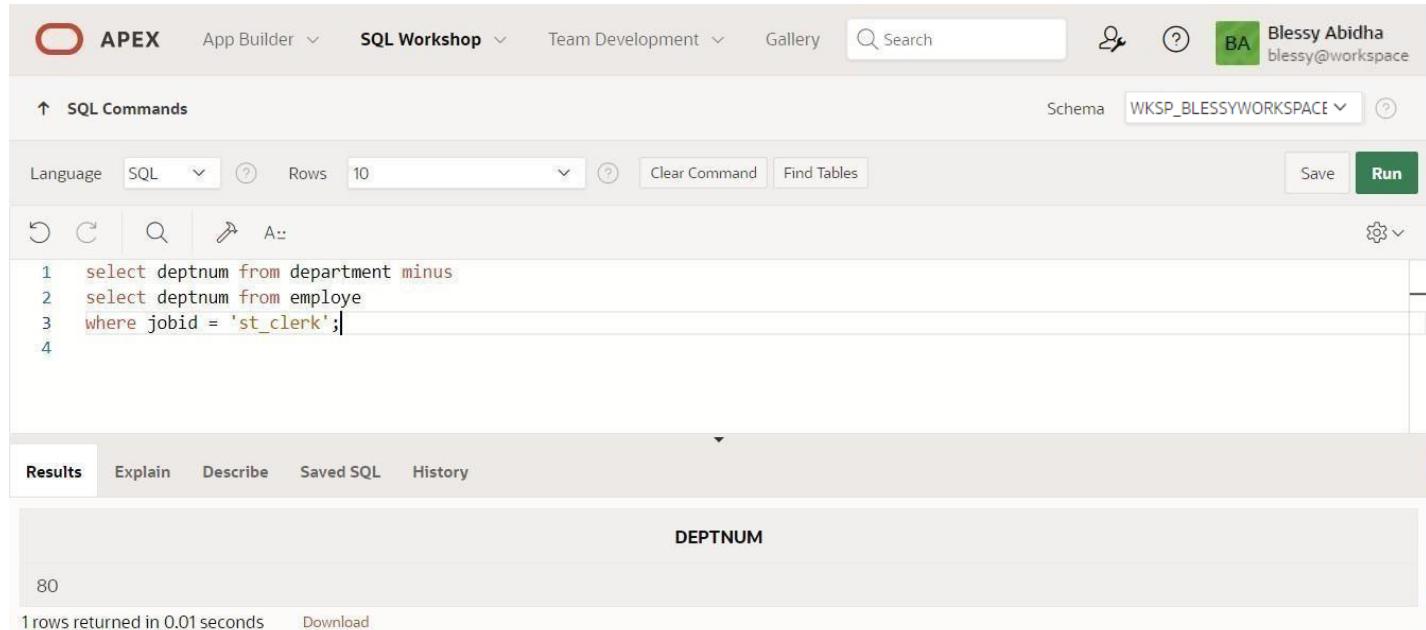
Find the Solution for the following:

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 deptnum from department minus  
select deptnum from employe  
where jobid = 'st_clerk';
```

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, Gallery, a search bar, and a user profile for 'Blessy Abidha'.

The SQL Commands tab is active, showing the following SQL code:

```
1 select deptnum from department minus  
2 select deptnum from employe  
3 where jobid = 'st_clerk';  
4
```

The Results tab is selected, displaying the output:

DEPTNUM
80

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

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 cid,cname from
country
minus
select l.cid,c cname from
location l, country c where
l.cid = c.cid;
```

OUTPUT:

The screenshot shows the Oracle 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 profile for 'BA Blessy Abidha' with the email 'blessy@workspace'. Below the toolbar, the schema is set to 'WKSP_BLESSYWORKSPACE'. The main area is titled 'SQL Commands' with a 'Language' dropdown set to 'SQL'. The query entered is:

```
1 select cid,cname from country
2 minus
3 select l.cid,c cname from location l, country c where l.cid = c.cid;
4
```

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

CID	CNAME
15	africa

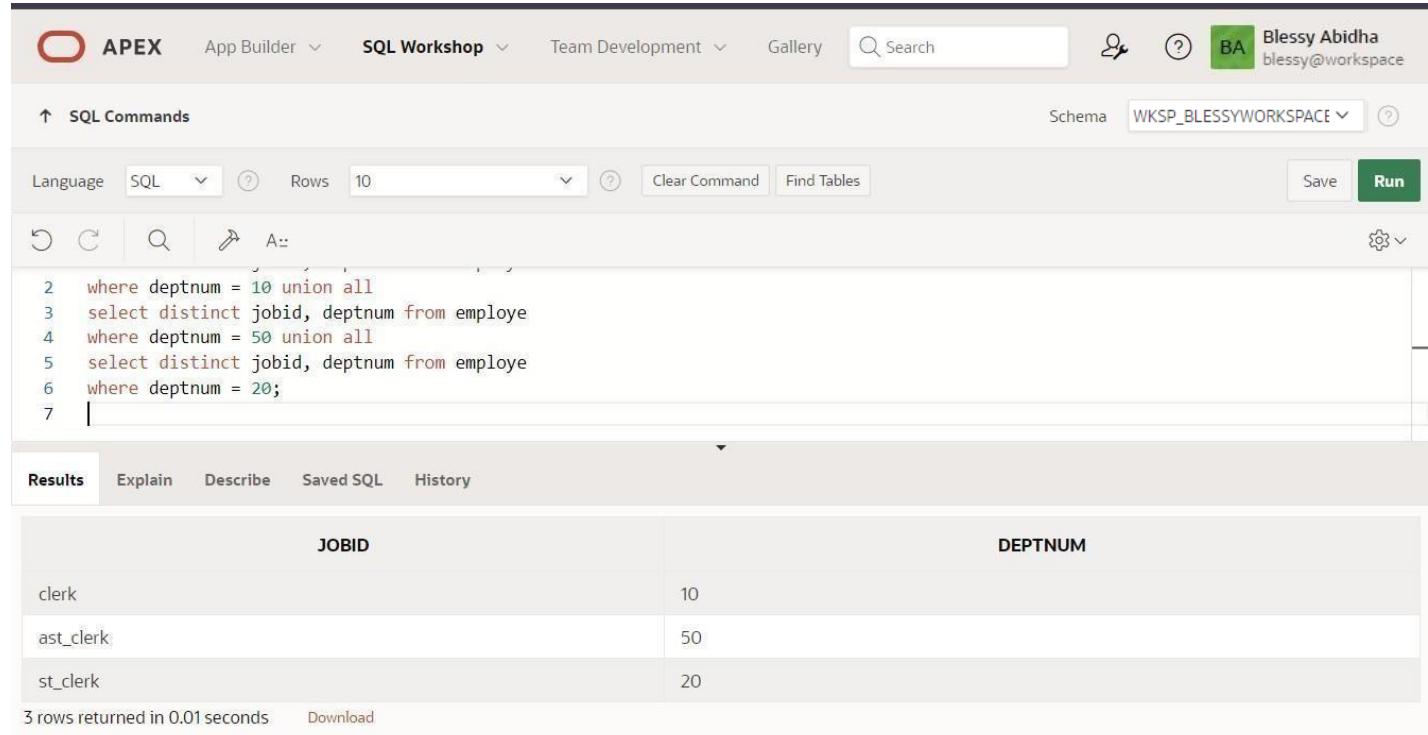
Below the table, it says '1 rows returned in 0.01 seconds' and has a 'Download' link.

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

QUERY:

```
select distinct jobid, deptnum from employe
where deptnum = 10 union all
select distinct jobid, deptnum from employe
where deptnum = 50 union all
select distinct jobid, deptnum from employe
where deptnum = 20;
```

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, Gallery, a search bar, and user information for 'Blessy Abidha' (blessy@workspace). Below the navigation is a toolbar with icons for Undo, Redo, Search, and Run. The main area displays the SQL command entered by the user:

```
2 where deptnum = 10 union all
3 select distinct jobid, deptnum from employe
4 where deptnum = 50 union all
5 select distinct jobid, deptnum from employe
6 where deptnum = 20;
7
```

Below the command is a results grid with two columns: JOBID and DEPTNUM. The data returned is:

JOBID	DEPTNUM
clerk	10
ast_clerk	50
st_clerk	20

At the bottom left, it says '3 rows returned in 0.01 seconds'. There is also a 'Download' link.

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

QUERY:

```
select empid,jobno from
employe intersect select
empid,jobno from
jobhistory;
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, Gallery, a search bar, and a user profile for 'Blessy Abidha'.

In the SQL Commands section, the schema is set to 'WKSP_BLESSYWORKSPACE'. The code entered is:

```
1 select empid,jobno from employe intersect
2 select empID,jobno from jobhistory;
3 |
```

The Results tab is selected, showing the output:

EMPID	JOBNO
6	90

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

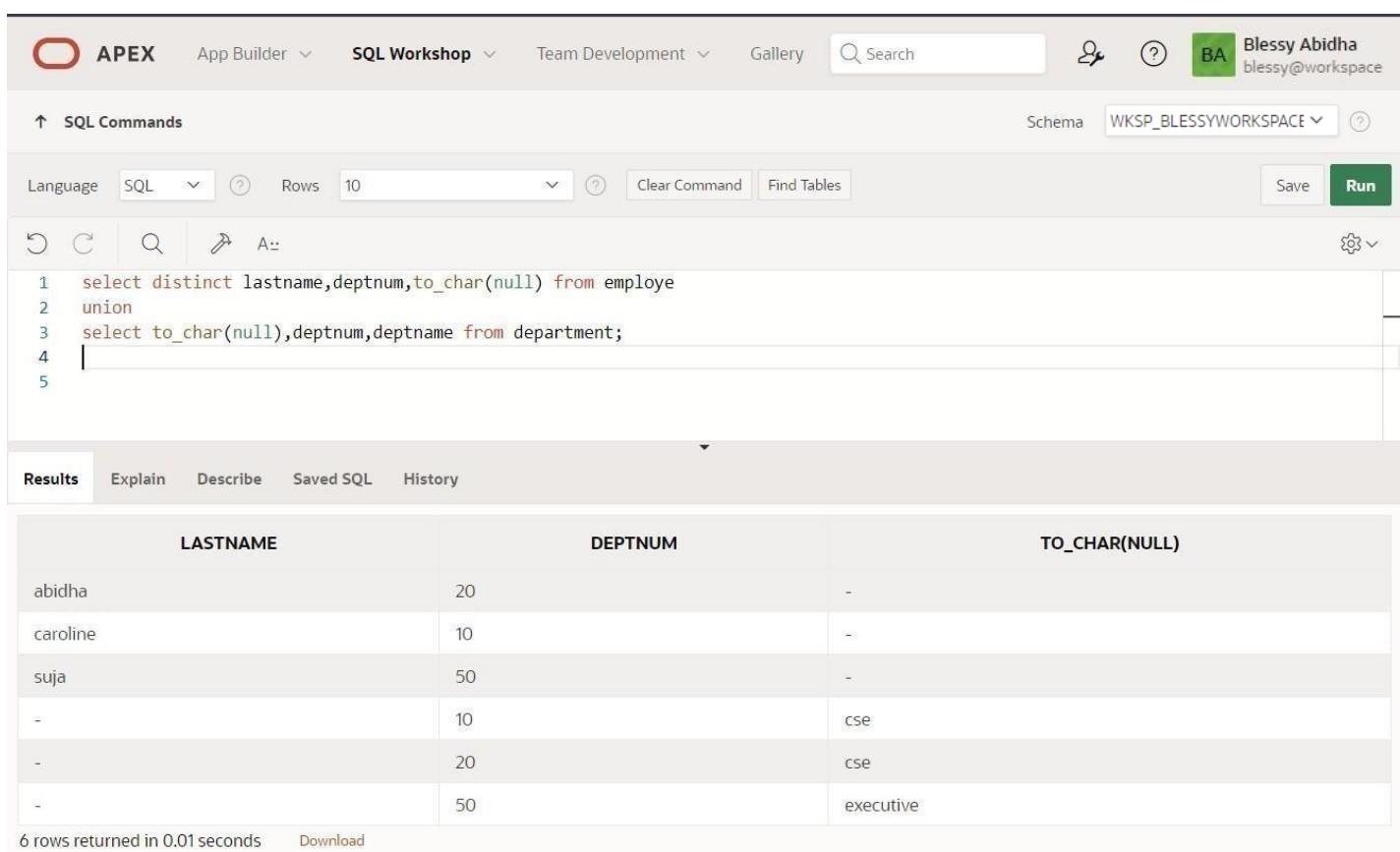
5. The HR department needs a report with the following specifications:

- Last name and department ID of all the employees from the EMPLOYEES table, regardless of whether or not they belong to a department.
- Department ID and department name of all the departments from the DEPARTMENTS table, regardless of whether or not they have employees working in them Write a compound query to accomplish this.

QUERY:

```
select distinct
lastname,deptnum,to_char(null) from
employe
union
select to_char(null),deptnum,deptname
from department;
```

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, Gallery, a search bar, and a user profile for 'Blessy Abidha'.

In the SQL Workshop tab, the schema is set to 'WKSP_BLESSYWORKSPACE'. The SQL command window contains the following code:

```
1 select distinct lastname,deptnum,to_char(null) from employe
2 union
3 select to_char(null),deptnum,deptname from department;
4
5
```

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

The results section displays the output of the query:

LASTNAME	DEPTNUM	TO_CHAR(NULL)
abidha	20	-
caroline	10	-
suja	50	-
-	10	cse
-	20	cse
-	50	executive

Below the table, it says '6 rows returned in 0.01 seconds' and there is a 'Download' link.

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

RESULT:

CREATING VIEWS

EXP NO: 11

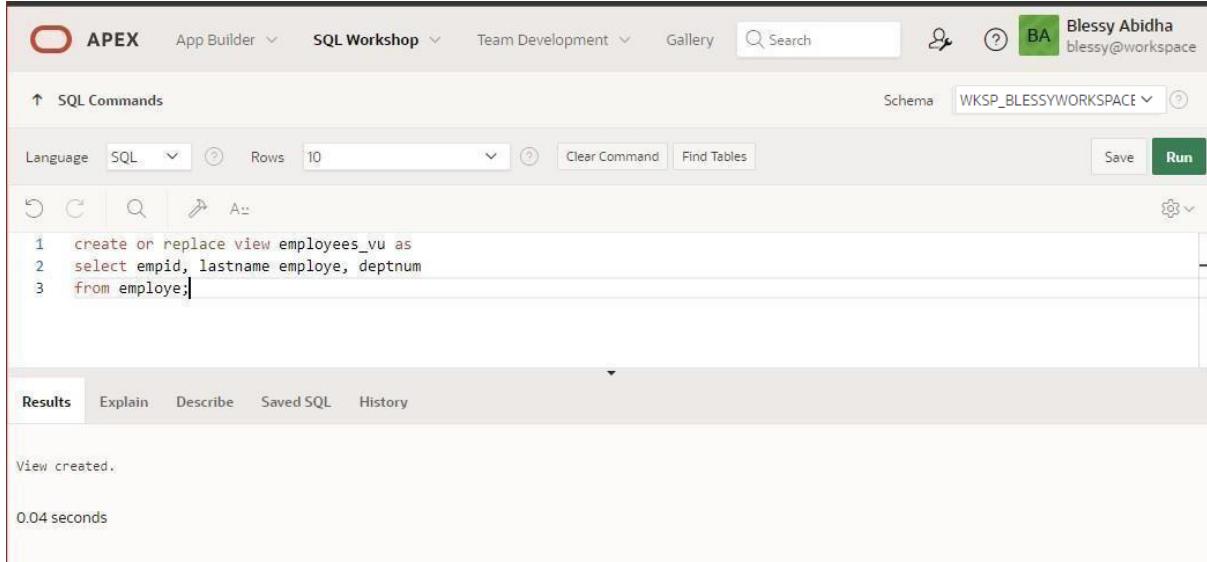
DATE:

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 employees_vu as select empid, lastname employee, deptnum from employee;
```

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', 'Gallery', a search bar, and user information 'Blessy Abidha' and 'WKSP_BLESSYWORKSPACE'. The main area is titled 'SQL Commands' and contains a code editor with the following SQL code:

```
1 create or replace view employees_vu as
2 select empid, lastname employee, deptnum
3 from employee;
```

Below the code editor, there are tabs for 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History'. The 'Results' tab is selected, displaying the message 'View created.' and '0.04 seconds' execution time.

2. Display the contents of the EMPLOYEES_VU view.

QUERY:

```
select * from employees_vu;
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', 'Gallery', a search bar, and a user profile 'BA Blessy Abidha'. The 'Schema' dropdown is set to 'WKSP_BLESSYWORKSPACE'. The main area has a 'SQL Commands' tab selected, showing the query 'select * from employees_vu;'. Below the command is a results grid with three rows. The columns are labeled 'EMPID', 'EMPLOYEE', and 'DEPTNUM'. The data is as follows:

EMPID	EMPLOYEE	DEPTNUM
6	caroline	10
8	abidha	20
7	suja	50

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

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

QUERY:

```
select view_name, text from user_views;
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', 'Gallery', a search bar, and a user profile 'BA Blessy Abidha'. The 'Schema' dropdown is set to 'WKSP_BLESSYWORKSPACE'. The main area has a 'SQL Commands' tab selected, showing the query 'select view_name, text from user_views;'. Below the command is a results grid with three rows. The columns are labeled 'VIEW_NAME' and 'TEXT'. The data is as follows:

VIEW_NAME	TEXT
DEPT50	select empid empno, lastname employee,deptnum deptno from employee where deptnum = 50 with check option
EMPLOYEES_VU	SELECT empid, lastname employee, deptnum FROM employee
SALARY_VU	SELECT e.lastname "Employee", d.deptname "Department", e.salary "Salary", j.grade "Grades" FROM employee e, department d, grade j WHERE e.deptnum = d.deptnum AND e.salary BETWEEN j.lowestsal and j.highestsal

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

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

QUERY:

```
select employe, deptnum from employees_vu;
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. At the top, there's a navigation bar with 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', 'Gallery', a search bar, and a user profile for 'Blessy Abidha'. Below the bar, the 'SQL Commands' section is active, showing the query 'select employe, deptnum from employees_vu;'. The 'Results' tab is selected in the bottom navigation bar, displaying a table with three rows:

EMPLOYEE	DEPTNUM
caroline	10
abidha	20
suja	50

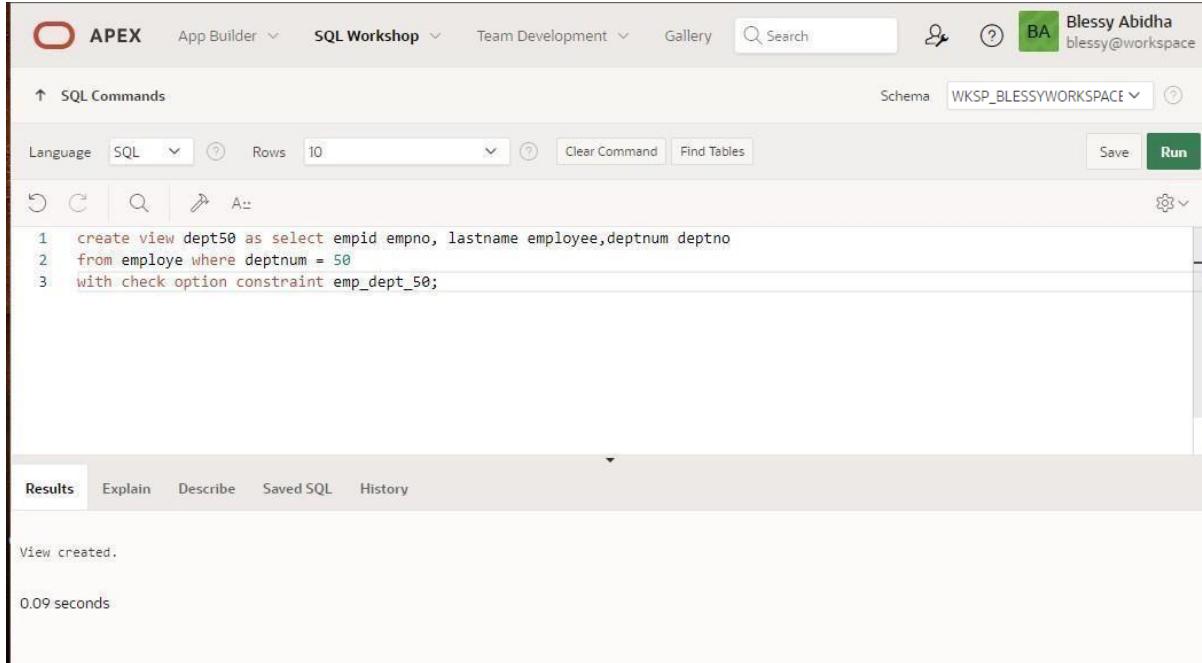
Below the table, it says '3 rows returned in 0.01 seconds' and has a 'Download' link.

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 empid empno, lastname employee,deptnum deptno from employe where deptnum= 50 with check option constraint emp_dept_50;
```

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. In the top navigation bar, 'APEX' is selected. The main area is titled 'SQL Commands'. The schema is set to 'WKSP_BLESSYWORKSPACE'. A SQL command is entered in the editor:

```
1 create view dept50 as select empid empno, lastname employee,deptnum deptno
2 from employe where deptnum = 50
3 with check option constraint emp_dept_50;
```

The 'Run' button is highlighted in green. Below the editor, the results tab is active, showing the message 'View created.' and a execution time of '0.09 seconds'.

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

QUERY:

Describe dept50;

select * from dept50;

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, Gallery, and a search bar. The user is logged in as BA (Blessy Abidha) with the schema WKSP_BLESSYWORKSPACE. The main area is titled 'SQL Commands' with a 'Language' dropdown set to 'SQL'. Below the command input field, there are buttons for 'Save' and 'Run'. The command entered is 'describe dept50;'. The results pane shows the description of the DEPT50 view, which has three columns: EMPNO, EMPLOYEE, and DEPTNO. The 'Describe' tab is selected at the bottom. The results table is as follows:

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

7. Attempt to reassign Matos to department 80.

QUERY:

update dept50 set deptno= 80 where employee= 'Matos';

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, Gallery, and a search bar. The user is logged in as BA (Blessy Abidha) with the schema WKSP_BLESSYWORKSPACE. The main area is titled 'SQL Commands' with a 'Language' dropdown set to 'SQL'. Below the command input field, there are buttons for 'Save' and 'Run'. The commands entered are 'select * from dept50;' followed by a blank line. The results pane shows the output of the query, which returns one row with EMPNO 7, EMPLOYEE suja, and DEPTNO 50. The 'Results' tab is selected at the bottom. The output table is as follows:

EMPNO	EMPLOYEE	DEPTNO
7	suja	50

1 rows returned in 0.02 seconds Download

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. At the top, there's a navigation bar with links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. A search bar is also present. On the right side of the header, there's a user profile for 'Blessy Abidha' with the email 'blessy@workspace'. Below the header, the main area is titled 'SQL Commands'. The schema is set to 'WKSP_BLESSYWORKSPACE'. The code entered is:

```
1 update dept50 set deptno = 80
2 where employee = 'Matos';
3
```

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

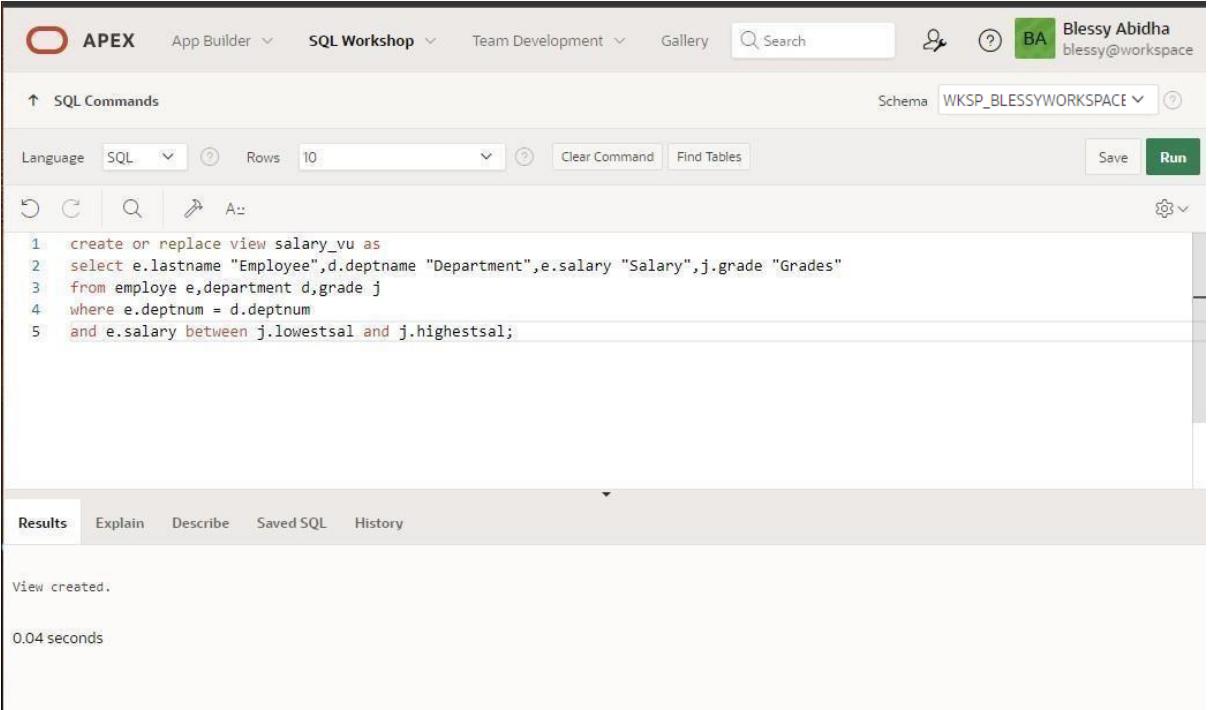
0 row(s) updated.
0.03 seconds

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

QUERY:

```
create view salary_vu as select e.lastname 'Employee', d.deptname 'Department', e.salary ' Salary', j.grade 'Grade' from employe e ,department d ,grade j where e.deptnum=d.deptnum and e.salary between j.lowestsal and j.highestsal;
```

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, Gallery, a search bar, and a user profile for 'Blessy Abidha' (blessy@workspace). The main area is titled 'SQL Commands'. The language is set to SQL, and the number of rows is set to 10. The schema is 'WKSP_BLESSYWORKSPACE'. Below the toolbar, there are icons for Undo, Redo, Find, and Paste. The SQL command entered is:

```
1 create or replace view salary_vu as
2 select e.lastname "Employee",d.deptname "Department",e.salary "Salary",j.grade "Grades"
3 from employe e,department d,grade j
4 where e.deptnum = d.deptnum
5 and e.salary between j.lowestsal and j.highestsal;
```

At the bottom of the SQL pane, there are tabs for Results, Explain, Describe, Saved SQL, and History. The results pane displays the message 'View created.' and '0.04 seconds'.

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

RESULT:

EXERCISE 12

PRACTICE QUESTIONS

Intro to Constraints; NOT NULL and UNIQUE Constraints

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

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

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;
```

2. Issue a DROP view_copy_d_songs. Execute a SELECT * statement to verify that the view has been deleted.

```
DROP VIEW view_copy_d_songs;
SELECT * FROM view_copy_d_songs;
```

ORA-00942: table or view does not exist

3. Create a query that selects the last name and salary from the Oracle database. Rank the salaries from highest to lowest for the top three employees.

```
SELECT * FROM
(SELECT last_name, salary FROM employees ORDER BY salary DESC)
WHERE ROWNUM <= 3;
```

4. Construct an inline view from the Oracle database that lists the last name, salary, department ID, and maximum salary for each department. Hint: One query will need to calculate maximum salary by department ID.

```
SELECT empm.last_name, empm.salary, dptmx.department_id
FROM
(SELECT dpt.department_id, MAX(NVL(emp.salary,0)) max_dpt_sal
FROM departments dpt LEFT OUTER JOIN employees emp ON dpt.department_id =
emp.department_id
GROUP BY dpt.department_id) dptmx LEFT OUTER JOIN employees empm ON
dptmx.department_id = empm.department_id
WHERE NVL(empm.salary,0) = dptmx.max_dpt_sal;
```

5. Create a query that will return the staff members of Global Fast Foods ranked by salary from lowest to highest.

```
SELECT ROWNUM, last_name, salary
FROM
(SELECT * FROM f_staffs ORDER BY SALARY);
```

Indexes and Synonyms

1. What is an index and what is it used for?

Definition: These are schema objects which make retrieval of rows from table faster.

Purpose: An index provides direct and fast access to row in table. They provide indexed path to locate data quickly, so hereby reduce necessity of heavy disk input/output operations.

2. What is a ROWID, and how is it used?

Indexes use ROWID's (base 64 string representation of the row address containing block identifier, row location in the block and the database file identifier) which is the fastest way to access any particular row.

3. When will an index be created automatically?

Primary key/unique key use already existing unique index but if index is not present already, it is created while applying unique/primary key constraint.

4. Create a nonunique index (foreign key) for the DJs on Demand column (cd_number) in the D_TRACK_LISTINGS table. Use the Oracle Application Express SQL Workshop Data Browser to confirm that the index was created.

```
CREATE INDEX d_tlg_cd_number_fk_i  
on d_track_listings (cd_number);
```

5. Use the join statement to display the indexes and uniqueness that exist in the data dictionary for the DJs on Demand D_SONGS table.

```
SELECT ucm.index_name, ucm.column_name, ucm.column_position, uix.uniqueness  
FROM user_indexes uix INNER JOIN user_ind_columns ucm ON uix.index_name = ucm.index_name  
WHERE ucm.table_name = 'D_SONGS';
```

6. Use a SELECT statement to display the index_name, table_name, and uniqueness from the data dictionary USER_INDEXES for the DJs on Demand D_EVENTS table.

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

7. Write a query to create a synonym called dj_tracks for the DJs on Demand d_track_listings table.

```
CREATE SYNONYM dj_tracks FOR d_track_listings;
```

8. Create a function-based index for the last_name column in DJs on Demand D_PARTNERS table that makes it possible not to have to capitalize the table name for searches. Write a SELECT statement that would use this index.

```
CREATE INDEX d_ptr_last_name_idx  
ON d_partners(LOWER(last_name));
```

9. Create a synonym for the D_TRACK_LISTINGS table. Confirm that it has been created by querying the data dictionary.

CREATE SYNONYM dj_tracks2 FOR d_track_listings;

SELECT * FROM user_synonyms WHERE table_NAME = UPPER('d_track_listings');

10. Drop the synonym that you created in question

DROP SYNONYM dj_tracks2;

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

RESULT:

OTHER DATABASE OBJECTS

EX_NO:14

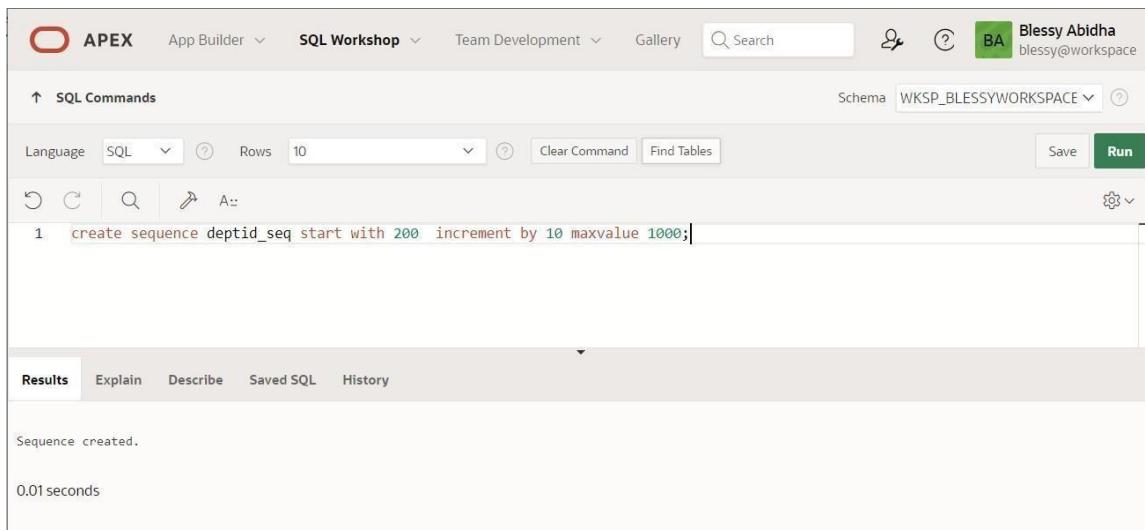
DATE:

1.)Create a sequence to be used with the primary key column of the DEPT table. The sequence should start at 200 and have a maximum value of 1000. Have your sequence increment by ten numbers. Name the sequence DEPT_ID_SEQ

QUERY:

```
create sequence deptid_seq start with 200 increment by 10 maxvalue 1000;
```

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, Gallery, a search bar, and a user profile for 'Blessy Abidha'. The main workspace is titled 'SQL Commands' and shows the schema 'WKSP_BLESSYWORKSPACE'. 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 area contains the SQL code: '1 create sequence deptid_seq start with 200 increment by 10 maxvalue 1000;'. The results pane at the bottom shows the output: 'Sequence created.' and '0.01 seconds'.

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

QUERY:

```
select sequence_name, max_value, increment_by, last_number from user_sequences;
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. At the top, there are tabs for APEX, App Builder, SQL Workshop, Team Development, and Gallery. A search bar and a user profile for 'Blessy Abidha' are also at the top. The main area is titled 'SQL Commands' with a schema dropdown set to 'WKSP_BLESSYWORKSPACE'. The query entered is 'select sequence_name, max_value, increment_by, last_number from user_sequences;'. The results section shows a single row of data:

SEQUENCE_NAME	MAX_VALUE	INCREMENT_BY	LAST_NUMBER
DEPTID_SEQ	1000	10	200

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

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

QUERY:

insert into dept values (deptid_seq.nextval, 'Education');

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface again. The query entered is 'insert into dept values (deptid_seq.nextval, 'Education');'. The results section shows the output: '1 row(s) inserted.' and '0.01 seconds'.

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

QUERY:

```
create index emp_deptid_idx on employe (deptnum);
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, Gallery, a search bar, and a user profile for 'Blessy Abidha'. The SQL Commands tab is active. The schema is set to 'WKSP_BLESSYWORKSPACE'. The query editor contains the command: 'create index emp_deptid_idx on employe (deptnum);'. The results section shows the output: 'Index created.' and a execution time of '0.04 seconds'.

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

QUERY:

```
select index_name,table_name,uniqueness from user_indexes where table_name='EMPLOYEE';
```

OUTPUT:

APEX App Builder SQL Workshop Team Development Gallery Search

SQL Commands Schema WKSP_BLESSYWORKSPACE

Language SQL Rows 10 Clear Command Find Tables Save Run

Results Explain Describe Saved SQL History

```
1 select index_name,table_name,uniqueness from user_indexes where table_name='EMPLOYEE';
```

INDEX_NAME	TABLE_NAME	UNIQUENESS
EMP_DEPTID_IDX	EMPLOYEE	NONUNIQUE

1 rows returned in 0.07 seconds Download

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

RESULT:

CONTROLLING USER ACCESS

EX_NO:15

DATE:

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

The CREATE SESSION system privilege

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

The CREATE TABLE privilege

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

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

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

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

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

The ALTER USER statement

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

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

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

7. Query all the rows in your DEPARTMENTS table.

SELECT * FROM departments;

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

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

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

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

SELECT table_name FROM user_tables;

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

Team 1 revokes the privilege.

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

Team 2 revokes the privilege.

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

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

Team 1 executes this INSERT statement.

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

Team 2 executes this INSERT statement.

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

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

RESULT:

PL/SQL

CONTROL STRUCTURES

EX_NO:

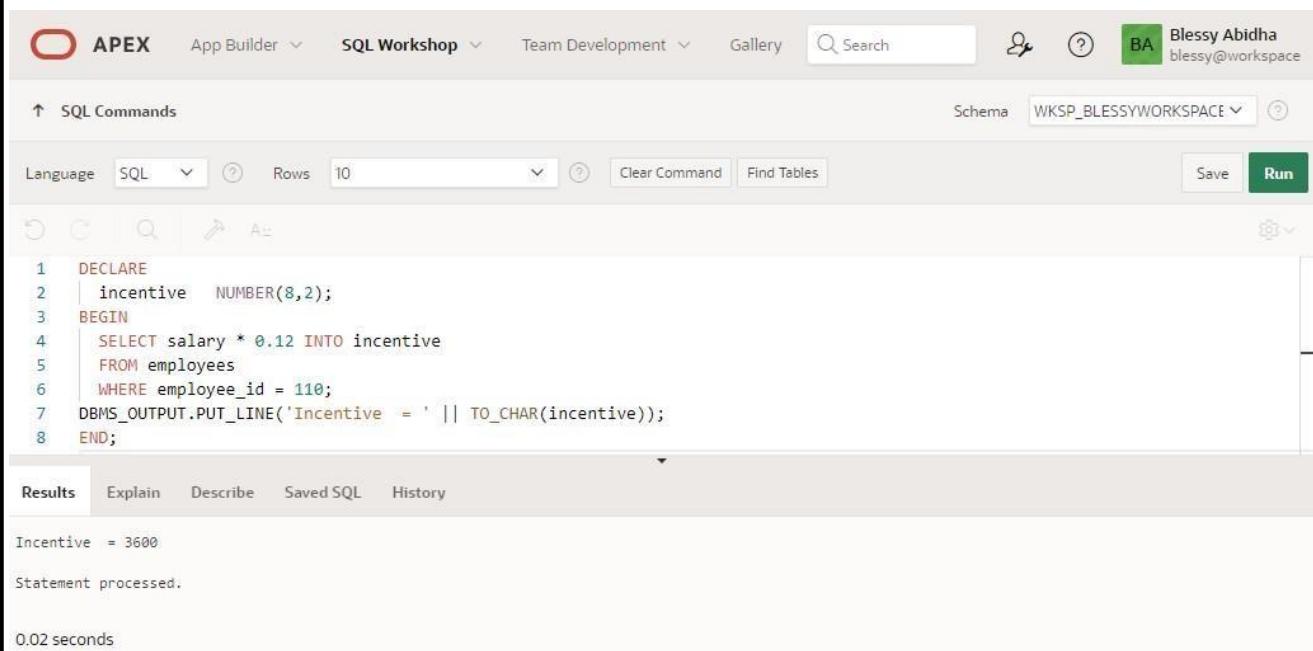
DATE:

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

QUERY:

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

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, Gallery, a search bar, and a user profile for 'Blessy Abidha'. The main workspace is titled 'SQL Commands' and shows the following code in the editor:

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

The code is run, and the results panel displays:

```
Incentive = 3600
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. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. A search bar and user information (Blessy Abidha, blessy@workspace) are also present. The main area is titled "SQL Commands". The code entered is:

```
1  DECLARE
2  |  "WELCOME" varchar2(10) := 'welcome'; -- identifier with quotation
3  BEGIN
4  |  DBMS_Output.Put_Line("Welcome"); --reference to the identifier with quotation and different case
5  END;
6  /
```

The "Results" tab is selected, displaying the error message:

```
Error at line 4/25: ORA-06550: line 4, column 25:
PLS-00201: identifier 'Welcome' must be declared
ORA-06512: at "SYS.WWV_DBMS_SQL_APEX_230200", line 801
ORA-06550: line 4, column 3:
PL/SQL: Statement ignored
```

Below the error message, the original code is repeated:

```
2.  "WELCOME" varchar2(10) := 'welcome'; -- identifier with quotation
3. BEGIN
4. DBMS_Output.Put_Line("Welcome"); --reference to the identifier with quotation
and different case
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,

address 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:' || s)

approx salary (100, salary of emp, 1000);

DBMS_OUTPUT.PUT_LINE

('After invoking procedure, salary of emp:' || salary of emp);

END:

1

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. On the right, there's a user profile for 'BA Blessy Abidha' and a search bar. The main workspace is titled 'SQL Commands'. It features a code editor with syntax highlighting for PL/SQL, showing a procedure definition. Below the code, there are tabs for 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History'. The results pane displays the output of the executed procedure, showing the salary being updated from 5500 to 6500.

```
1  DECLARE
2      salary_of_emp NUMBER(8,2);
3  PROCEDURE approx_salary (
4      emp          NUMBER,
5      empsal IN OUT NUMBER,
6      addless      NUMBER
7  ) IS
8  BEGIN
9      empsal := empsal + addless;
10 END;
11 BEGIN
12     SELECT salary INTO salary_of_emp
13     FROM employees
14     WHERE employee_id = 122;
15     DBMS_OUTPUT.PUT_LINE
16     ('Before invoking procedure, salary_of_emp: ' || salary_of_emp);
17     approx_salary (100, salary_of_emp, 1000);
18     DBMS_OUTPUT.PUT_LINE
19     ('After invoking procedure, salary_of_emp: ' || salary_of_emp);
20 END;
```

Results

```
Before invoking procedure, salary_of_emp: 5500
After invoking procedure, salary_of_emp: 6500

Statement processed.

0.01 seconds
```

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

QUERY:

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

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 'Blessy Abidha' with the email 'blessy@workspace'. The main workspace is titled 'SQL Commands'. The language is set to 'SQL'. The code area contains a PL/SQL procedure named 'pri_bool'. The procedure declares variables 'salary_of_emp' and 'emp', and a parameter 'empsal'. It uses the 'DBMS_OUTPUT.PUT_LINE' function to print the value of 'salary_of_emp' before and after invoking a procedure 'approx_salary'. The code is as follows:

```
1  DECLARE
2      salary_of_emp  NUMBER(8,2);
3      PROCEDURE approx_salary (
4          emp          NUMBER,
5          empsal IN OUT NUMBER,
6          address      NUMBER
7      ) IS
8      BEGIN
9          empsal := empsal + address;
10     END;
11    BEGIN
12        SELECT salary INTO salary_of_emp
13        FROM employees
14        WHERE employee_id = 122;
15        DBMS_OUTPUT.PUT_LINE
16        ('Before invoking procedure, salary_of_emp: ' || salary_of_emp);
17        approx_salary (100, salary_of_emp, 1000);
18        DBMS_OUTPUT.PUT_LINE
19        ('After invoking procedure, salary_of_emp: ' || salary_of_emp);
20    END;
```

The results section shows the output of the procedure execution:

```
Before invoking procedure, salary_of_emp: 5500
After invoking procedure, salary_of_emp: 6500

Statement processed.

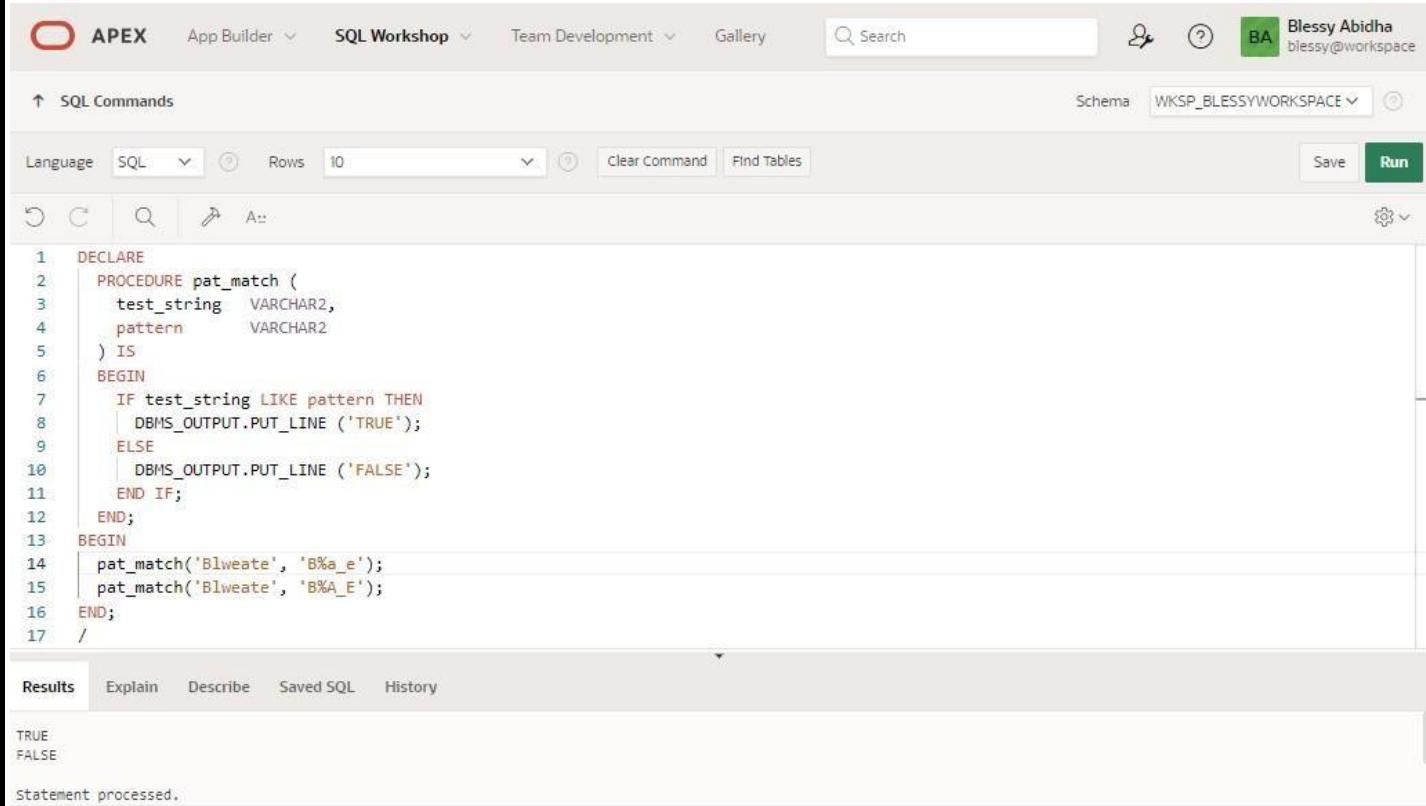
0.01 seconds
```

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, Gallery, and a search bar. The user is logged in as Blessy Abidha (BA) with the email address blessy@workspace. The schema is set to WKSP_BLESSYWORKSPACE. The main workspace displays the PL/SQL code from the previous step. The code defines a procedure pat_match that checks if a test string matches a given pattern using the LIKE operator. It uses DBMS_OUTPUT.PUT_LINE to print 'TRUE' or 'FALSE'. The code is run, and the results are shown in the Results tab: 'TRUE' for the first call and 'FALSE' for the second. The Explain, Describe, Saved SQL, and History tabs are also visible at the bottom.

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

Results	Explain	Describe	Saved SQL	History
TRUE FALSE				
Statement processed.				

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

QUERY:

DECLARE

```
num_small NUMBER := 8;
```

```
num_large NUMBER := 5;
```

```
num_temp NUMBER;
```

```
BEGIN
```

```
IF num_small > num_large THEN
```

```
    num_temp := num_small;
```

```
    num_small := num_large;
```

```
    num_large := num_temp;
```

```
END IF;
```

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

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

```
END;
```

```
/
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, Gallery, a search bar, and a user profile for 'Blessy Abidha'. The main workspace is titled 'SQL Commands' and contains the following PL/SQL code:

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

The 'Results' tab is selected at the bottom, displaying the output of the DBMS_OUTPUT.PUT_LINE statements:

```
num_small = 5
num_large = 8

Statement processed.

0.00 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;
/
```

↑ SQL Commands

Schema

WKSP_BLESSYWORKSPACE

Language SQL Rows 10 Clear Command Find Tables

Save

Run

```
1  DECLARE
2  PROCEDURE test1 (sal_achieve  NUMBER)
3  IS
4      incentive  NUMBER := 0;
5  BEGIN
6      IF sal_achieve > 44000 THEN
7          incentive := 1800;
8      ELSIF sal_achieve > 32000 THEN
9          incentive := 800;
10     ELSE
11         incentive := 500;
12     END IF;
13     DBMS_OUTPUT.NEW_LINE;
14     DBMS_OUTPUT.PUT_LINE (
15         'Sale achieved : ' || sal_achieve || ', incentive : ' || incentive || '.'
16     );
17     END test1;
18 BEGIN
19     test1(45000);
20     test1(36000);
21     test1(28000);
22 END;
```

Results Explain Describe Saved SQL History

Sale achieved : 45000, incentive : 1800.

Sale achieved : 36000, incentive : 800.

Sale achieved : 28000, incentive : 500.

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;
```

```
27  DECLARE
28    v_emp_count NUMBER;
29    v_vacancies NUMBER := 45;
30  BEGIN
31    -- Count the number of employees in department 50
32    SELECT COUNT(*)
33    INTO v_emp_count
34    FROM employees
35    WHERE department_id = 50;
36
37    -- Display the number of employees in department 50
38    DBMS_OUTPUT.PUT_LINE('Number of employees in department 50: ' || v_emp_count);
39
40    -- Check if there are any vacancies
41    IF v_emp_count < v_vacancies THEN
42      DBMS_OUTPUT.PUT_LINE('There are vacancies in department 50.');
43    ELSE
44      DBMS_OUTPUT.PUT_LINE('There are no vacancies in department 50.');
45    END IF;
46  END;
```

Results Explain Describe Saved SQL History

```
Number of employees in department 50: 1
There are no vacancies in department 50.

Statement processed.
```

/ 0.02 seconds

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;
```

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The user is logged in as Blessy Abidha (blessy@workspace). The SQL Commands tab is active, showing the following PL/SQL code:

```
52 DECLARE
53     v_dept_count NUMBER;
54     v_vacancies NUMBER := 45;
55 BEGIN
56     -- Count the number of employees in department 50
57     SELECT COUNT(*)
58         INTO v_dept_count
59         FROM employees
60         WHERE department_id = 50;
61
62     -- Display the count
63     DBMS_OUTPUT.PUT_LINE('Number of employees in department 50: ' || v_dept_count);
64
65     -- Check for vacancies
66     IF v_dept_count < v_vacancies THEN
67         DBMS_OUTPUT.PUT_LINE('There are vacancies in department 50.');
68     ELSE
69         DBMS_OUTPUT.PUT_LINE('There are no vacancies in department 50.');
70     END IF;
71 END;
72 /
```

The Results tab at the bottom displays the output of the executed code:

```
Number of employees in department 50: 1
There are vacancies in department 50.

Statement processed.
```

The bottom status bar indicates the execution took 0.01 seconds.

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

QUERY:

```
DECLARE
    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:

↑ SQL Commands

Schema

WKSP_BLESSYWORKSPACE

Save

Run

Language SQL Rows 10 Clear Command Find Tables



```
76
77 DECLARE
78   v_dept_id employees.department_id%TYPE := 50; -- Change this to the desired department ID
79   v_dept_count NUMBER;
80   v_vacancies NUMBER := 45; -- Change this to the number of vacancies in the department
81 BEGIN
82   -- Count the number of employees in the specified department
83   SELECT COUNT(*)
84     INTO v_dept_count
85     FROM employees
86    WHERE department_id = v_dept_id;
87
88   -- Display the count
89   DBMS_OUTPUT.PUT_LINE('Number of employees in department ' || v_dept_id || ': ' || v_dept_count);
90
91   -- Check for vacancies
92   IF v_dept_count < v_vacancies THEN
93     DBMS_OUTPUT.PUT_LINE('There are vacancies in department ' || v_dept_id || '.');
94     DBMS_OUTPUT.PUT_LINE('Number of vacancies: ' || (v_vacancies - v_dept_count));
95   ELSE
96     DBMS_OUTPUT.PUT_LINE('There are no vacancies in department ' || v_dept_id || '.');
97   END IF;
98 END;
99 /
```

Results Explain Describe Saved SQL History

Number of employees in department 50: 1
There are vacancies in department 50.
Number of vacancies: 44

Statement processed.

220701047@rajalakshmi.edu.in

blessy@workspace

en

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

Oracle APEX 23.2.4

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

QUERY:

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

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 identified as 'Blessy Abidha' with the email 'blessy@workspace'. The schema is set to 'WKSP_BLESSYWORKSPACE'. The main workspace shows the PL/SQL code for displaying employee details. The results pane at the bottom shows three distinct sets of employee data, each starting with 'Employee ID: 103', 'Employee ID: 104', and 'Employee ID: 105' respectively.

```
Employee ID: 103
Employee Name: SANJ PARTHI
Job Title: ST_CLERK
Hire Date: 12-APR-1998
Salary: 8650.82
-----
Employee ID: 104
Employee Name: MARY JANE
Job Title: HR_MANAGER
Hire Date: 15-MAR-1998
Salary: 23400.12
-----
Employee ID: 105
Employee Name: AKAY KOHLI
Job Title: SALESREP
Hire Date: 17-JUN-1997
Salary: 11500.00
```

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 APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. The user is identified as 'Blessy Abidha' with the email 'blessy@workspace'. The schema is set to 'WKSP_BLESSYWORKSPACE'. The main workspace shows the PL/SQL code for displaying employee information. The results pane below shows the output for four employees:

```
Employee ID: 103
Employee Name: SANJ PARTHI
Department Name: STOCK
-----
Employee ID: 104
Employee Name: MARY JANE
Department Name: HR
-----
Employee ID: 109
Employee Name: AKAY KOHLI
Department Name: MANUFACTURING
-----
Employee ID: 110
Employee Name: BEN DAVIES
Department Name: FINANCE
-----
```

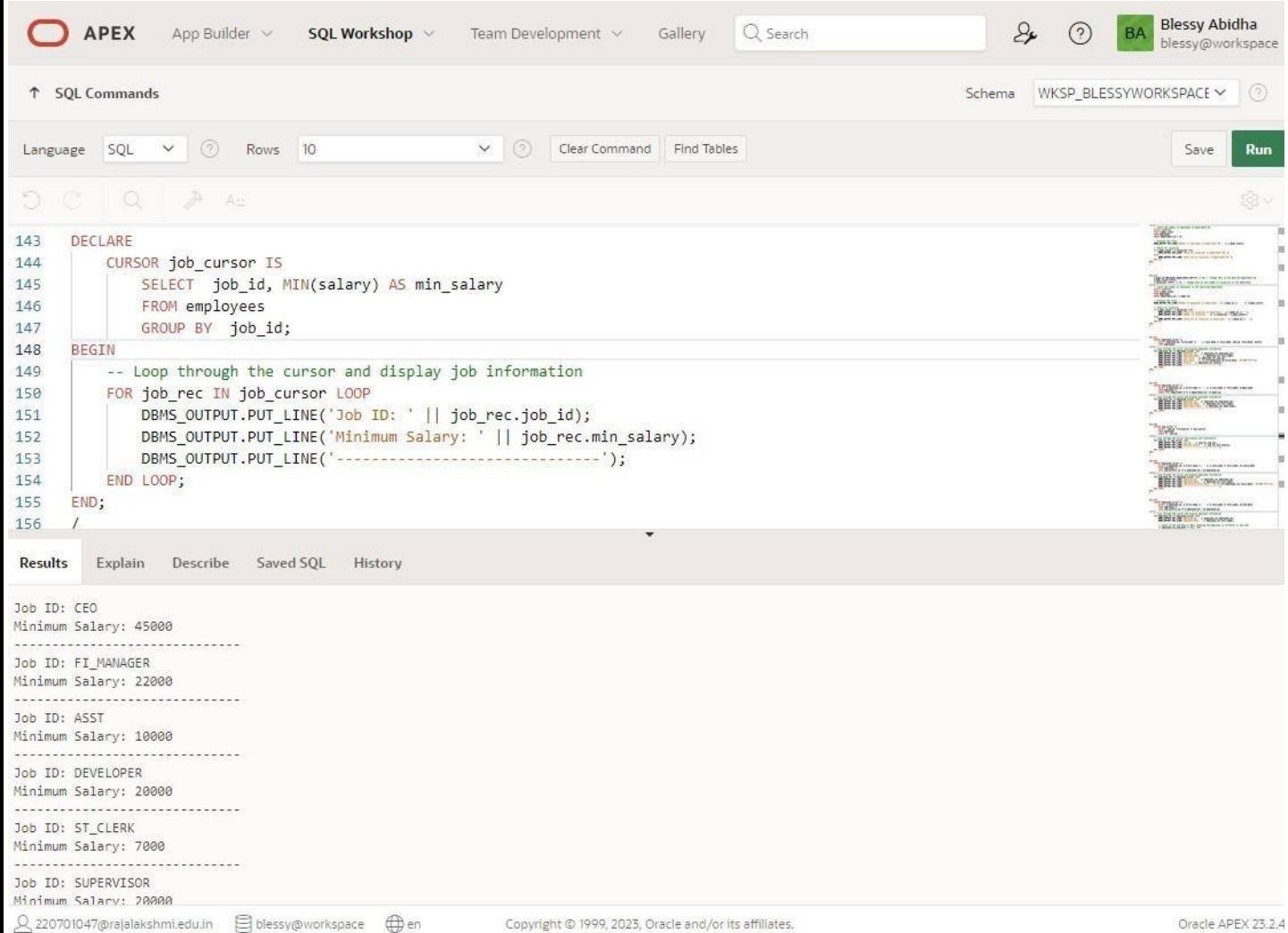
At the bottom of the results pane, there are links for Results, Explain, Describe, Saved SQL, and History. The footer of the page includes copyright information for Oracle and links for user support and documentation.

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

QUERY:

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

OUTPUT:



The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery, along with a search bar and user information for 'Blessy Abidha'.

The main workspace displays the PL/SQL code in the SQL Commands editor. The code declares a cursor to find the minimum salary for each job, then loops through the cursor to print the job ID and its minimum salary. The results are shown in the Results tab at the bottom, displaying the output for each job.

Job ID	Minimum Salary
CEO	45000
FI_MANAGER	22000
ASST	10000
DEVELOPER	20000
ST_CLERK	7000
SUPERVISOR	20000

At the bottom left, there are links for Explain, Describe, Saved SQL, and History. The bottom right corner shows the copyright notice 'Copyright © 1999, 2023, Oracle and/or its affiliates.' and the version 'Oracle APEX 23.2.4'.

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

APEX App Builder SQL Workshop Team Development Gallery Search Schema WKSP_BLESSYWORKSPACE Run

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

158
159 DECLARE
160 CURSOR employee_cursor IS
161 SELECT e.employee_id, e.first_name || ' ' || e.last_name AS full_name, jh.start_date
162 FROM employees e
163 JOIN job_history jh ON e.employee_id = jh.employee_id;
164 BEGIN
165 -- Loop through the cursor and display employee information
166 FOR employee_rec IN employee_cursor LOOP
167 DBMS_OUTPUT.PUT_LINE('Employee ID: ' || employee_rec.employee_id);
168 DBMS_OUTPUT.PUT_LINE('Employee Name: ' || employee_rec.full_name);
169 DBMS_OUTPUT.PUT_LINE('Job History Start Date: ' || TO_CHAR(employee_rec.start_date, 'DD-MON-YYYY'));
170 DBMS_OUTPUT.PUT_LINE('-----');
171 END LOOP;
172 END;

Results Explain Describe Saved SQL History

```
Employee ID: 103
Employee Name: SANJ PARTHI
Job History Start Date: 04-DEC-1998
-----
Employee ID: 100
Employee Name: SAHANA JAY
Job History Start Date: 05-JAN-1999
-----
Statement processed.
```

0.02 seconds

220701047@rajalakshmi.edu.in blessy@workspace en Copyright © 1999, 2023, Oracle and/or its affiliates. Oracle APEX 23.2.4

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

QUERY:

DECLARE

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

OUTPUT:

```
APEX SQL Workshop - Team Development - Gallery - Search - BA - Blessy Abidha - blessy@workspace
```

↑ SQL Commands Schema WKSP_BLESSYWORKSPACE

Language SQL Rows 10 Save Run

```
176 DECLARE
177   CURSOR employee_cursor IS
178     SELECT e.employee_id, e.first_name || ' ' || e.last_name AS full_name, jh.end_date
179     FROM employees e
180     JOIN job_history jh ON e.employee_id = jh.employee_id;
181 BEGIN
182   -- Loop through the cursor and display employee information
183   FOR employee_rec IN employee_cursor LOOP
184     DBMS_OUTPUT.PUT_LINE('Employee ID: ' || employee_rec.employee_id);
185     DBMS_OUTPUT.PUT_LINE('Employee Name: ' || employee_rec.full_name);
186
187     -- Check if the end date is NULL (meaning the employee is currently in the job)
188     IF employee_rec.end_date IS NULL THEN
189       DBMS_OUTPUT.PUT_LINE('Job History End Date: (Still Employed)');
190     ELSE
191       DBMS_OUTPUT.PUT_LINE('Job History End Date: ' || TO_CHAR(employee_rec.end_date, 'DD-MON-YYYY'));
192     END IF;
193
194     DBMS_OUTPUT.PUT_LINE('-----');
195   END LOOP;
196 END;
```

Results Explain Describe Saved SQL History

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

202701047@rajalakshmi.edu.in blessy@workspace 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:

PROCEDURES AND FUNCTIONS

EX. NO: 17

DATE:

1.) Factorial of a number using function.

QUERY:

DECLARE

 fac NUMBER := 1;

 n NUMBER := :1;

BEGIN

 WHILE n > 0 LOOP

 fac := n * fac;

 n := n - 1;

 END LOOP;

 DBMS_OUTPUT.PUT_LINE(fac);

END;

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side of the header has a search bar, user profile, and workspace information (WKSP_BLESSYWORKSPACE). The main area is titled 'SQL Commands' with tabs for Language (SQL selected), Rows (10), Clear Command, Find Tables, Save, and Run. The code area contains a PL/SQL block:

```
1 CREATE OR REPLACE FUNCTION factorial(n IN NUMBER) RETURN NUMBER IS
2     result NUMBER := 1;
3     BEGIN
4         FOR i IN 1..n LOOP
5             result := result * i;
6         END LOOP;
7         RETURN result;
8     END;
9 /
10 DECLARE
11     num NUMBER := 5;
12     fact NUMBER;
13     BEGIN
14     fact := factorial(num);
15     DBMS_OUTPUT.PUT_LINE('Factorial of ' || num || ' is ' || fact);
16 END;
```

The results pane at the bottom shows the output: "Factorial of 5 is 120" and "Statement processed."

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

```
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. The main area displays a PL/SQL script:

```
56 DECLARE
57   v_book_id NUMBER := 1; -- Example book ID
58   v_title VARCHAR2(100);
59   v_author VARCHAR2(100);
60   v_year_published NUMBER;
61 BEGIN
62   -- Call the procedure
63   get_book_info(v_book_id, v_title, v_author, v_year_published);
64
65   -- Display the retrieved information
66   IF v_title IS NOT NULL THEN
67     DBMS_OUTPUT.PUT_LINE('Title: ' || v_title);
68     DBMS_OUTPUT.PUT_LINE('Author: ' || v_author);
69     DBMS_OUTPUT.PUT_LINE('Year Published: ' || v_year_published);
70   ELSE
71     DBMS_OUTPUT.PUT_LINE('Book information could not be retrieved.');
72   END IF;
73 END;
```

Below the code, the results are shown:

```
Title: 1984
Author: George Orwell
Year Published: 1949
```

Statement processed.

0.02 seconds

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

RESULT:

TRIGGER

EX_NO: 18

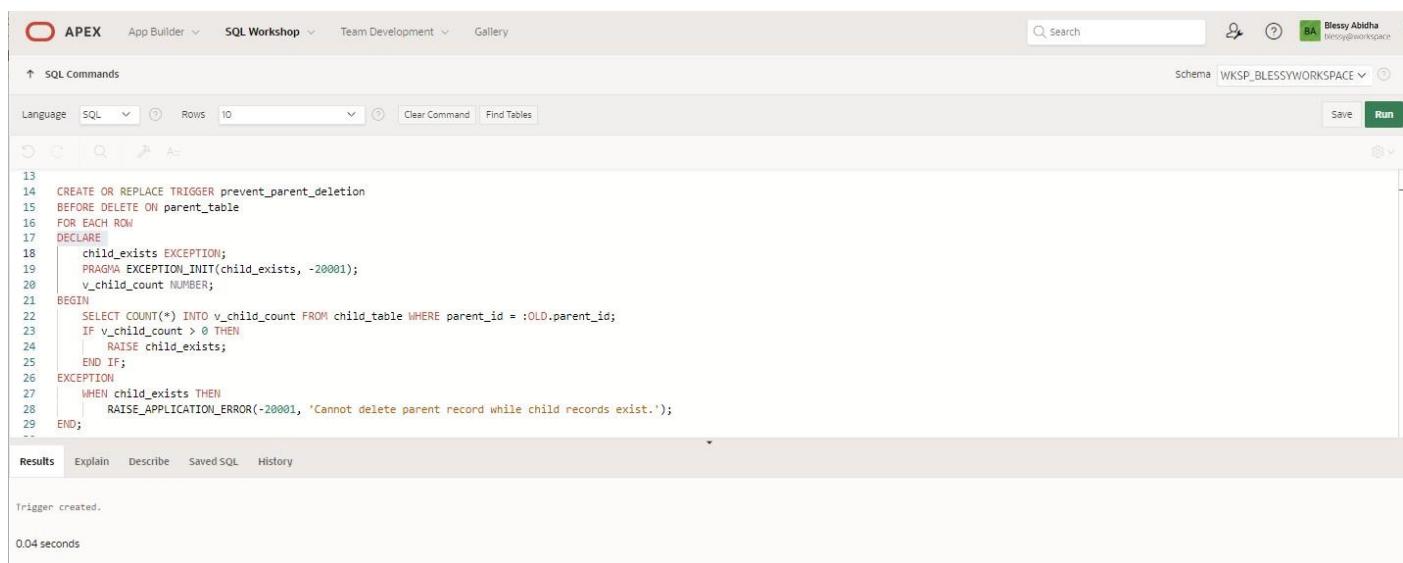
DATE:

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:



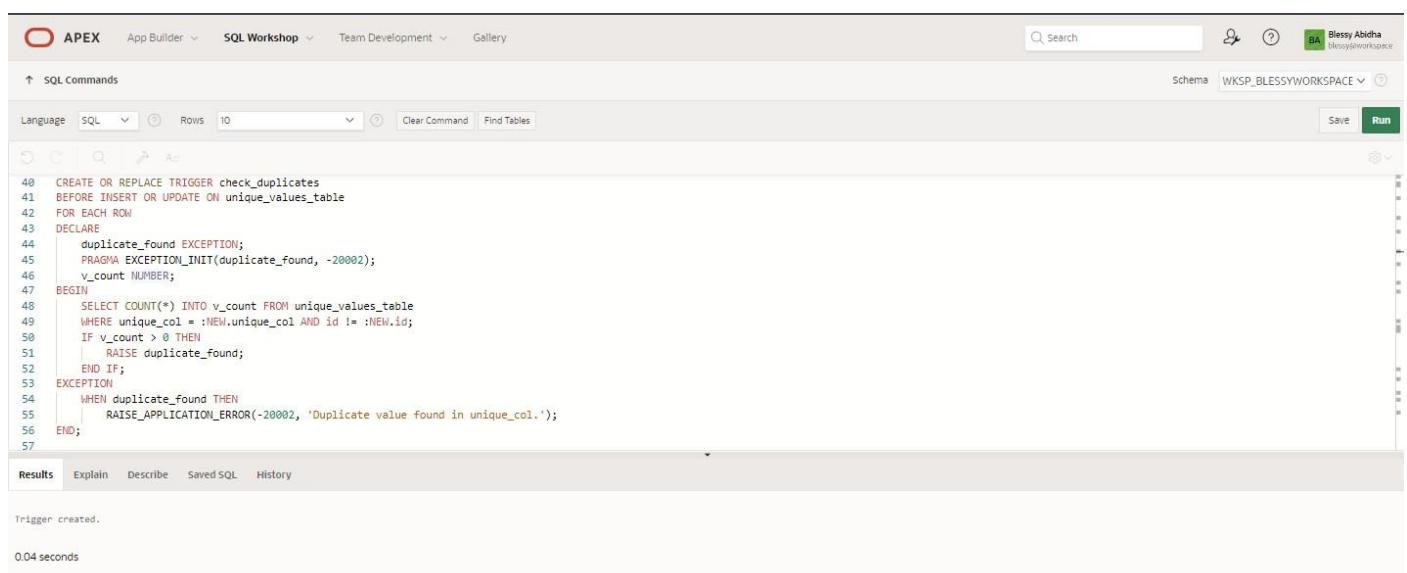
The screenshot shows the Oracle SQL Workshop interface. In the top navigation bar, 'APEX' is selected. Below it, tabs for 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery' are visible. On the right side of the header, there's a search bar, a user icon, and a workspace dropdown set to 'WKSP_BLESSYWORKSPACE'. The main area is titled 'SQL Commands'. A toolbar below the title includes buttons for 'Language' (set to 'SQL'), 'Rows' (set to 10), 'Clear Command', 'Find Tables', 'Save', and 'Run'. The SQL editor contains the PL/SQL code for the trigger. The code is numbered from 13 to 29. Lines 13-29 show the trigger definition, including the declaration of an exception 'child_exists', setting its initialization value to -20001, defining a variable 'v_child_count' as a NUMBER, and the BEGIN block with its logic. Lines 30-32 show the EXCEPTION section with a WHEN clause that raises the 'child_exists' exception with a specific error message. Line 33 shows the END keyword. At the bottom of the SQL editor, tabs for 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History' are present. The status bar at the bottom displays the message 'Trigger created.' and '0.04 seconds'.

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, tabs for 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery' are visible. On the right side of the header, there's a search bar, a user icon, and a workspace dropdown set to 'WKSP_BLESSYWORKSPACE'. The main area is titled 'SQL Commands'. A toolbar below the title includes buttons for 'Language' (set to 'SQL'), 'Rows' (set to 10), 'Clear Command', 'Find Tables', 'Save', and 'Run'. The SQL editor contains the PL/SQL code for the 'check_duplicates' trigger. The code is numbered from 40 to 57. The output pane at the bottom shows the message 'Trigger created.' and '0.04 seconds'.

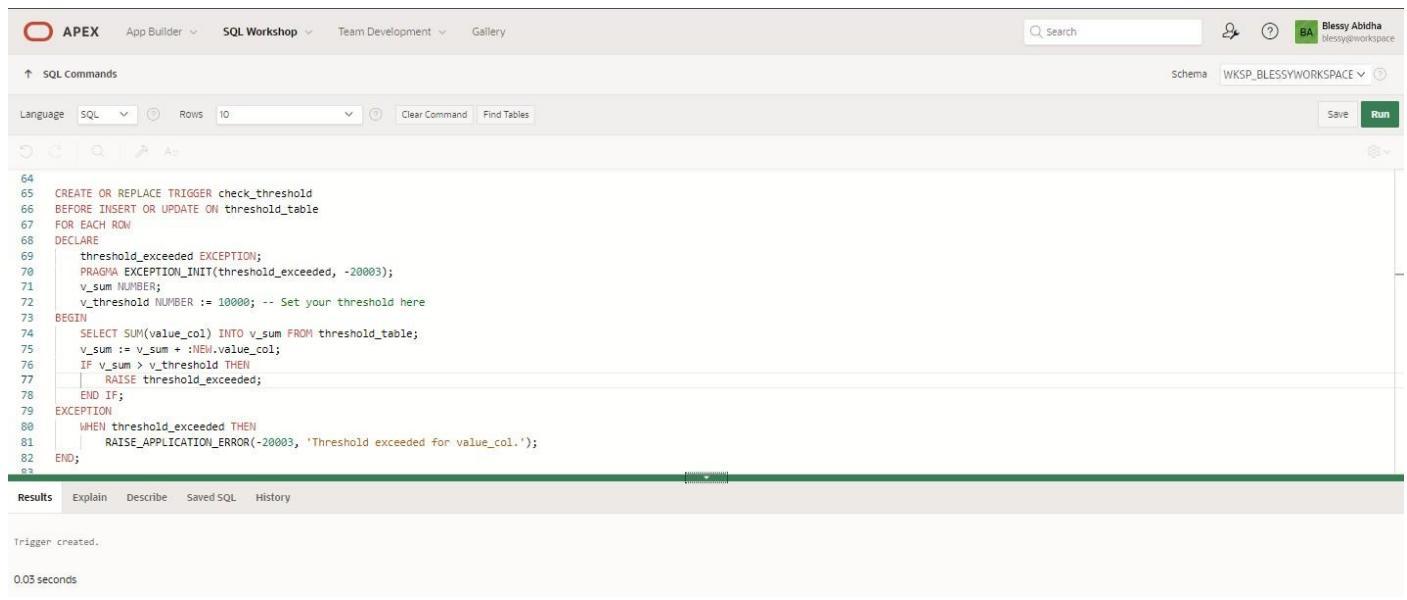
```
40 CREATE OR REPLACE TRIGGER check_duplicates
41 BEFORE INSERT OR UPDATE ON unique_values_table
42 FOR EACH ROW
43 DECLARE
44     duplicate_found EXCEPTION;
45     PRAGMA EXCEPTION_INIT(duplicate_found, -20002);
46     v_count NUMBER;
47 BEGIN
48     SELECT COUNT(*) INTO v_count FROM unique_values_table
49     WHERE unique_col = :NEW.unique_col AND id != :NEW.id;
50     IF v_count > 0 THEN
51         RAISE duplicate_found;
52     END IF;
53 EXCEPTION
54     WHEN duplicate_found THEN
55         RAISE_APPLICATION_ERROR(-20002, 'Duplicate value found in unique_col.');
56 END;
57
```

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

QUERY:

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

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side of the header shows the user 'Blessy Abidha' and the schema 'WKSP_BLESSYWORKSPACE'. The main area is titled 'SQL Commands' with tabs for Language (SQL selected), Rows (10), and Clear Command. Below these are icons for Undo, Redo, Find, Replace, and Save/Run. The SQL editor contains the PL/SQL code provided in the previous section. The code is numbered from 64 to 83. The output pane at the bottom shows the results of the command: 'Trigger created.' and '0.03 seconds'.

```
64 CREATE OR REPLACE TRIGGER check_threshold
65   BEFORE INSERT OR UPDATE ON threshold_table
66   FOR EACH ROW
67   DECLARE
68     threshold_exceeded EXCEPTION;
69     PRAGMA EXCEPTION_INIT(threshold_exceeded, -20003);
70     v_sum NUMBER;
71     v_threshold NUMBER := 10000; -- Set your threshold here
72 BEGIN
73   SELECT SUM(value_col) INTO v_sum FROM threshold_table;
74   v_sum := v_sum + :NEW.value_col;
75   IF v_sum > v_threshold THEN
76     RAISE threshold_exceeded;
77   END IF;
78 EXCEPTION
79   WHEN threshold_exceeded THEN
80     RAISE_APPLICATION_ERROR(-20003, 'Threshold exceeded for value_col.');
81   END;
82 
```

Results Explain Describe Saved SQL History

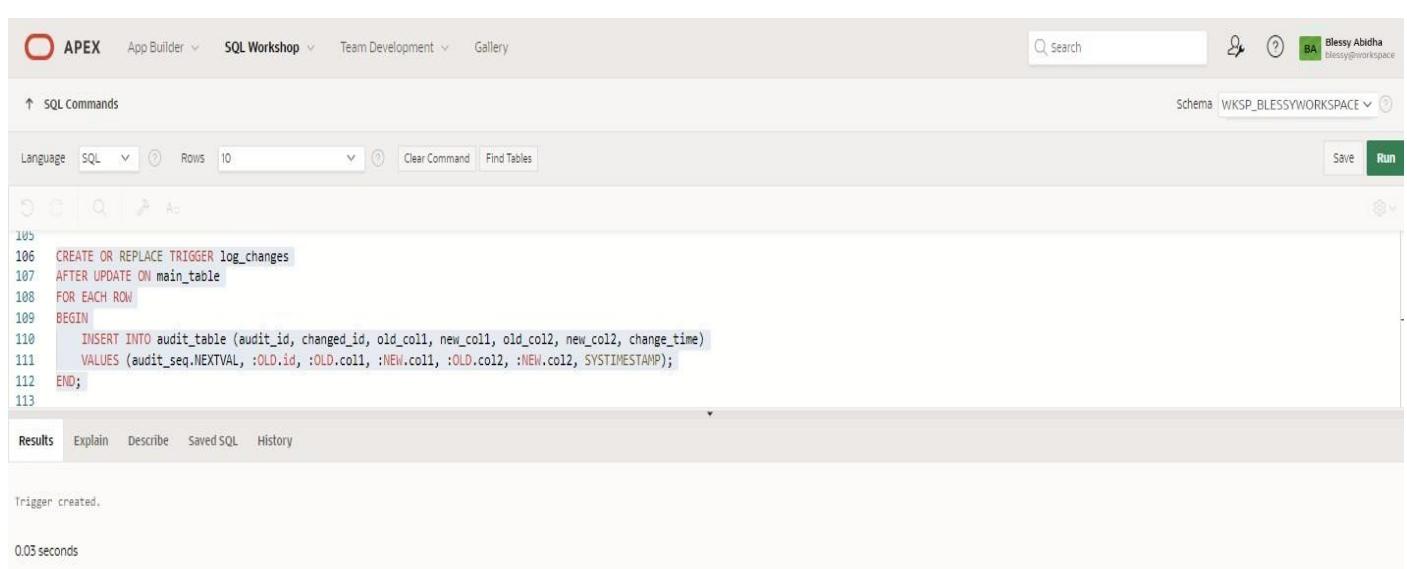
Trigger created.
0.03 seconds

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

QUERY:

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

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there's a search bar, a user icon for 'Blessy Abidha', and a schema dropdown set to 'WKSP_BLESSYWORKSPACE'. The main workspace displays the PL/SQL code for the 'log_changes' trigger. The code is numbered from 105 to 113. Lines 106 through 112 are highlighted in grey, indicating they were run. The output pane at the bottom shows the message 'Trigger created.' and a execution time of '0.03 seconds'.

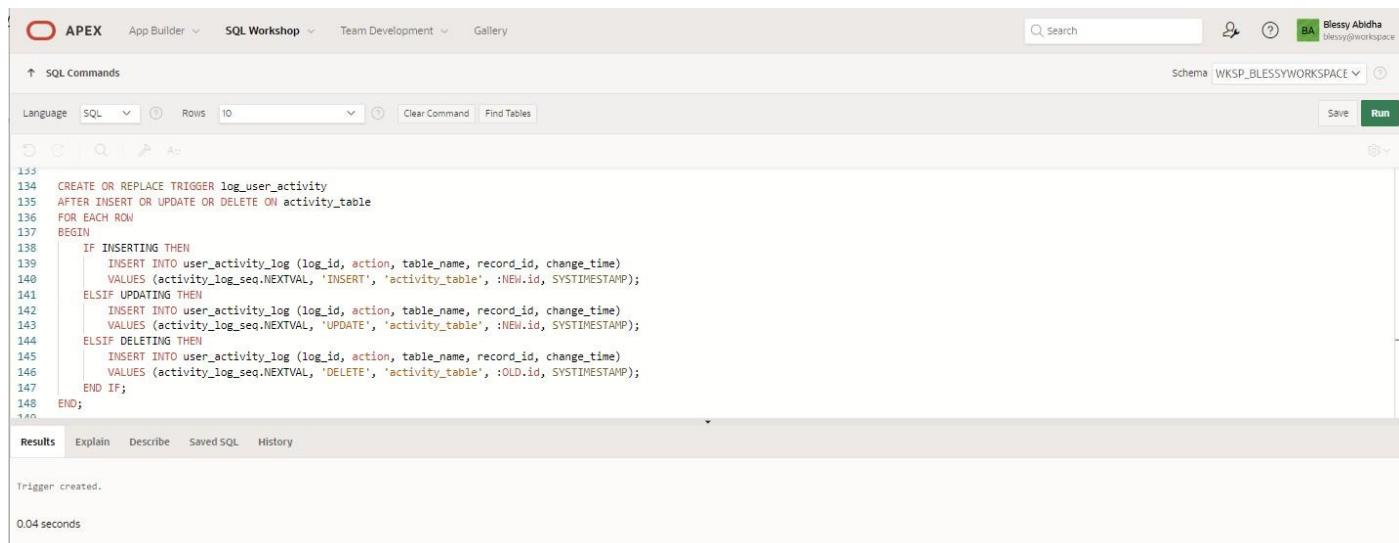
```
105
106 CREATE OR REPLACE TRIGGER log_changes
107 AFTER UPDATE ON main_table
108 FOR EACH ROW
109 BEGIN
110     INSERT INTO audit_table (audit_id, changed_id, old_col1, new_col1, old_col2,
111     new_col2, change_time)
112     VALUES (audit_seq.NEXTVAL, :OLD.id, :OLD.col1, :NEW.col1, :OLD.col2, :NEW.col2, SYSTIMESTAMP);
113 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. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there are tabs for Search, Schema (WKSP_BLESSYWORKSPACE), and a user icon (BA). The main area is titled "SQL Commands". Below it, there are buttons for Language (SQL selected), Rows (10), Clear Command, Find Tables, Save, and Run. The SQL code for the trigger is pasted into the command window. The code is as follows:

```
155
156 CREATE OR REPLACE TRIGGER log_user_activity
157 AFTER INSERT OR UPDATE OR DELETE ON activity_table
158 FOR EACH ROW
159 BEGIN
160     IF INSERTING THEN
161         INSERT INTO user_activity_log (log_id, action, table_name, record_id, change_time)
162         VALUES (activity_log_seq.NEXTVAL, 'INSERT', 'activity_table', :NEW.id, SYSTIMESTAMP);
163     ELSIF UPDATING THEN
164         INSERT INTO user_activity_log (log_id, action, table_name, record_id, change_time)
165         VALUES (activity_log_seq.NEXTVAL, 'UPDATE', 'activity_table', :NEW.id, SYSTIMESTAMP);
166     ELSIF DELETING THEN
167         INSERT INTO user_activity_log (log_id, action, table_name, record_id, change_time)
168         VALUES (activity_log_seq.NEXTVAL, 'DELETE', 'activity_table', :OLD.id, SYSTIMESTAMP);
169     END IF;
170 END;
```

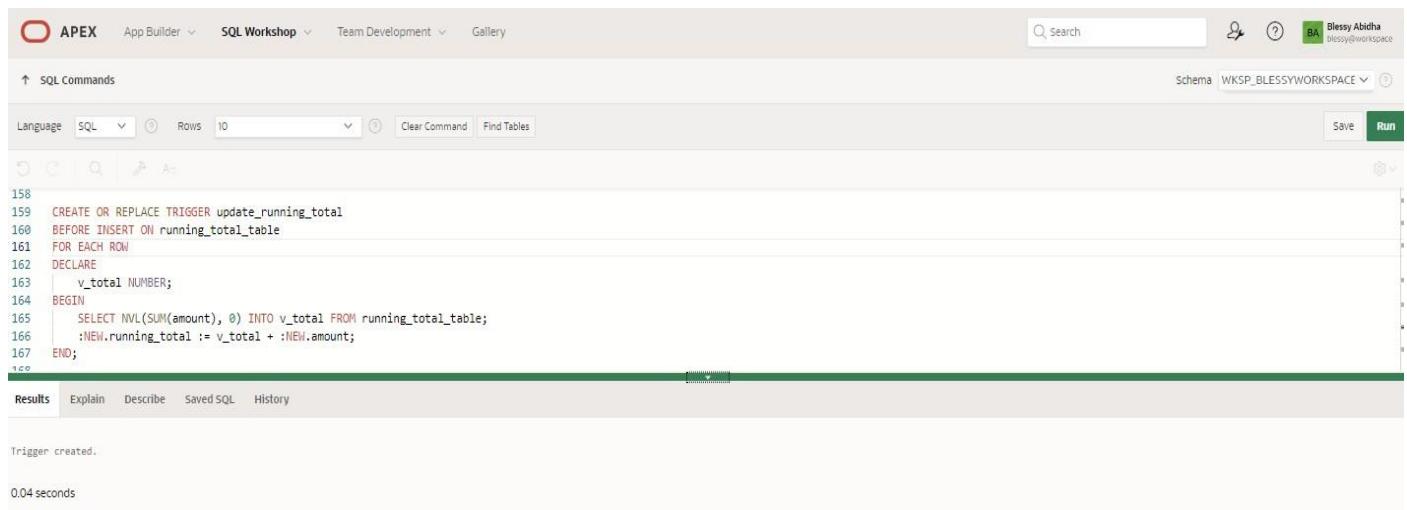
Below the code, the results section shows the message "Trigger created." and a timestamp of "0.04 seconds".

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

QUERY:

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

OUTPUT:



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

The main area is a SQL editor with the following content:

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

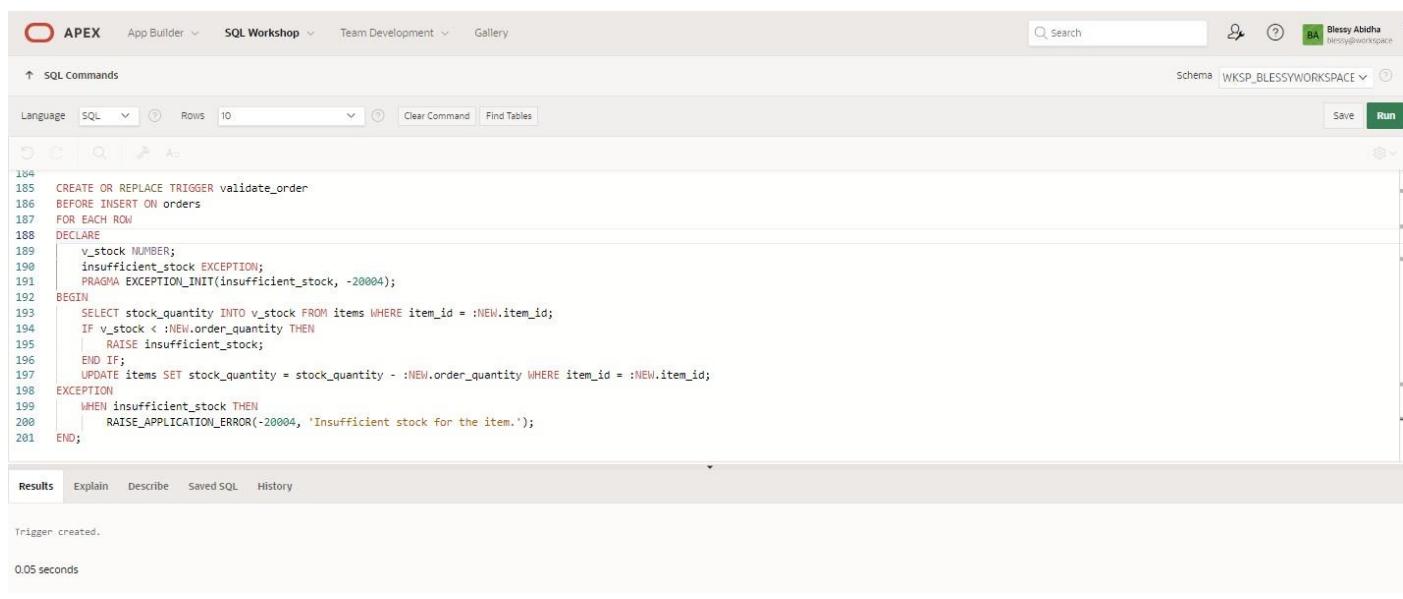
Below the editor, a toolbar has buttons for Results, Explain, Describe, Saved SQL, and History. The 'Results' tab is selected. The output pane shows the message "Trigger created." and "0.04 seconds".

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. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows a user profile for 'Blessy Abidha' with the email 'blessy@workspace'. The main area displays the PL/SQL code for the 'validate_order' trigger. The code is numbered from 184 to 201. The 'Results' tab at the bottom shows the message 'Trigger created.' and a execution time of '0.05 seconds'.

```
184
185 CREATE OR REPLACE TRIGGER validate_order
186 BEFORE INSERT ON orders
187 FOR EACH ROW
188 DECLARE
189     v_stock NUMBER;
190     insufficient_stock EXCEPTION;
191     PRAGMA EXCEPTION_INIT(insufficient_stock, -20004);
192 BEGIN
193     SELECT stock_quantity INTO v_stock FROM items WHERE item_id = :NEW.item_id;
194     IF v_stock < :NEW.order_quantity THEN
195         RAISE insufficient_stock;
196     END IF;
197     UPDATE items SET stock_quantity = stock_quantity - :NEW.order_quantity WHERE item_id = :NEW.item_id;
198 EXCEPTION
199     WHEN insufficient_stock THEN
200         RAISE_APPLICATION_ERROR(-20004, 'Insufficient stock for the item.');
201 END;
```

Trigger created.
0.05 seconds

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:



```
Blessyabidha_47> db.restaurants.find({$and:[{$or:[{cuisine: {$nin: ["American", "Chinese"]}}, {name: /^Wil/}]}]}, {restaurant_id: 1, name: 1, borough: 1, cuisine: 1})
[
  {
    _id: ObjectId('6654a7ffa676986518cdcdf6'),
    borough: 'Bronx',
    cuisine: 'Bakery',
    name: 'Morris Park Bake Shop',
    restaurant_id: '30075445'
  }
]
Blessyabidha_47> |
```

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:

```
mongosh mongodb://127.0.0.1:27017/  
Blessyabidha_47> db.restaurants.find({ "grades": { $elemMatch: { "grade": "A", "score": 11, "date": ISODate("2014-08-11T00:00:00Z") } } }, { restaurant_id: 1, name: 1, grades: 1 })  
Blessyabidha_47> |
```

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

QUERY:

```
db.restaurants.find(  
{  
    "grades.1.grade": "A",  
    "grades.1.score": 9,  
    "grades.1.date": ISODate("2014-08-01T00:00:00Z")  
},  
{  
    restaurant_id: 1,  
    name: 1,  
    grades: 1  

```

OUTPUT:

```
mongosh mongodb://127.0.0.1:27017/  
Blessyabidha_47> 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 })  
Blessyabidha_47> |
```

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:



A screenshot of a terminal window titled "mongosh mongodb://127.0.0.1:27017". The window shows a MongoDB shell command being run:

```
Blessyabidha_47> db.restaurants.find({$and : [{"address.coord.1": {$gt : 42}}, {"address.coord.1": {$lte : 52}}]}, {_id:0, restaurant_id:1, name:1, address:1})
```

The command is partially typed, ending with a pipe character '|'. The terminal interface includes standard window controls (minimize, maximize, close) at the top.

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:

```
mongosh mongodb://127.0.0.1:27017/foodtruck?readPreference=primary&socketTimeoutMS=5000&maxIdleTimeMS=10000&maxPoolSize=10&minPoolSize=0&w=1&wtimeout=10000&useNewUrlParser=true&useUnifiedTopology=true
```

```
Blessyabidha_47> db.restaurants.find({}, { _id: 0 }).sort({ name: 1 })
[{"_id": "5c3f3e05d000000000000001", "name": "Morris Park Bake Shop", "address": {"building": "1007", "coord": [-73.856077, 40.848447], "street": "Morris Park Ave", "zipcode": "10462"}, "borough": "Bronx", "cuisine": "Bakery", "grades": [{"date": ISODate('2014-03-03T00:00:00.000Z'), "grade": "A", "score": 2}, {"date": ISODate('2013-09-11T00:00:00.000Z'), "grade": "A", "score": 6}, {"date": ISODate('2013-01-24T00:00:00.000Z'), "grade": "A", "score": 10}, {"date": ISODate('2011-11-23T00:00:00.000Z'), "grade": "A", "score": 9}, {"date": ISODate('2011-03-10T00:00:00.000Z'), "grade": "B", "score": 14}], "restaurant_id": "30075445"}]
```

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:

```
mongosh mongodb://127.0.0.1:27017/... + -
```

```
Blessyabidha_47> db.restaurants.find({}, { _id: 0 }).sort({ name: 1 })
[ {
  address: {
    building: '1007',
    coord: [ -73.856077, 40.848447 ],
    street: 'Morris Park Ave',
    zipcode: '10462'
  },
  borough: 'Bronx',
  cuisine: 'Bakery',
  grades: [
    {
      date: ISODate('2014-03-03T00:00:00.000Z'),
      grade: 'A',
      score: 10
    }
  ]
}

Blessyabidha_47> db.restaurants.find({}, { _id: 0 }).sort({ name: 1 })
[ {
  address: {
    building: '1007',
    coord: [ -73.856077, 40.848447 ],
    street: 'Morris Park Ave',
    zipcode: '10462'
  },
  borough: 'Bronx',
  cuisine: 'Bakery',
  grades: [
    {
      date: ISODate('2014-03-03T00:00:00.000Z'),
      grade: 'A',
      score: 10
    }
  ]
}

Blessyabidha_47> db.restaurants.find({}, { _id: 0 }).sort({ name: -1 })
[ {
  address: {
    building: '1007',
    coord: [ -73.856077, 40.848447 ],
    street: 'Morris Park Ave',
    zipcode: '10462'
  },
  borough: 'Bronx',
  cuisine: 'Bakery',
  grades: [
    {
      date: ISODate('2014-03-03T00:00:00.000Z'),
      grade: 'A',
      score: 10
    },
    {
      date: ISODate('2013-09-11T00:00:00.000Z'),
      grade: 'A',
      score: 6
    },
    {
      date: ISODate('2013-01-24T00:00:00.000Z'),
      grade: 'A',
      score: 10
    },
    {
      date: ISODate('2011-11-23T00:00:00.000Z'),
      grade: 'A',
      score: 9
    }
  ]
}
```

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:

```
mongosh mongodb://127.0.0.1:27017/... + ▾
```

```
Blessyabidha_47> db.restaurants.find({}, { _id: 0 }).sort({ name: -1 })
[
  {
    address: {
      building: '1007',
      coord: [ -73.856077, 40.848447 ],
      street: 'Morris Park Ave',
      zipcode: '10462'
    },
    borough: 'Bronx',
    cuisine: 'Bakery',
    grades: [
      {
        date: ISODate('2014-03-03T00:00:00.000Z'),
        grade: 'A',
        score: 2
      },
      {
        date: ISODate('2013-09-11T00:00:00.000Z'),
        grade: 'A',
        score: 6
      },
      {
        date: ISODate('2013-01-24T00:00:00.000Z'),
        grade: 'A',
        score: 10
      },
      {
        date: ISODate('2011-11-23T00:00:00.000Z'),
        grade: 'A',
        score: 9
      },
      {
        date: ISODate('2011-03-10T00:00:00.000Z'),
        grade: 'B',
        score: 1
      }
    ]
  }
]
```

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:

```
mongosh mongodb://127.0.0.1:27017/
```

```
Blessyabidha_47> db.restaurants.find({}, { _id: 0 }).sort({ name: 1 })
[ {
  address: {
    building: '1007',
    coord: [ -73.856077, 40.848447 ],
    street: 'Morris Park Ave',
    zipcode: '10462'
  },
  borough: 'Bronx',
  cuisine: 'Bakery',
  grades: [
    {
      date: ISODate('2014-03-03T00:00:00.000Z'),
      grade: 'A',
      score: 100
    }
  ],
  name: 'The Bagelery',
  rating: 4.5
}

Blessyabidha_47> db.restaurants.find({}, { _id: 0 }).sort({ name: 1 })
[ {
  address: {
    building: '1007',
    coord: [ -73.856077, 40.848447 ],
    street: 'Morris Park Ave',
    zipcode: '10462'
  },
  borough: 'Bronx',
  cuisine: 'Bakery',
  grades: [
    {
      date: ISODate('2014-03-03T00:00:00.000Z'),
      grade: 'A',
      score: 100
    }
  ],
  name: 'The Bagelery',
  rating: 4.5
}

Blessyabidha_47> db.restaurants.find({}, { _id: 0 }).sort({ name: -1 })
[ {
  address: {
    building: '1007',
    coord: [ -73.856077, 40.848447 ],
    street: 'Morris Park Ave',
    zipcode: '10462'
  },
  borough: 'Bronx',
  cuisine: 'Bakery',
  grades: [
    {
      date: ISODate('2014-03-03T00:00:00.000Z'),
      grade: 'A',
      score: 100
    }
  ],
  name: 'The Bagelery',
  rating: 4.5
}

Blessyabidha_47> db.restaurants.find({}, { _id: 0 }).sort({ cuisine: 1, borough: -1 })
[ {
  address: {
    building: '1007',
    coord: [ -73.856077, 40.848447 ],
    street: 'Morris Park Ave',
    zipcode: '10462'
  },
  borough: 'Bronx',
  cuisine: 'Bakery',
  grades: [
    {
      date: ISODate('2014-03-03T00:00:00.000Z'),
      grade: 'A',
      score: 100
    }
  ],
  name: 'The Bagelery',
  rating: 4.5
}

Blessyabidha_47> db.restaurants.find({ "address.street": { $exists: true, $ne: "" } })
[ {
  _id: ObjectId('6654a7ffa676986518cdcdf6'),
  address: {
    building: '1007',
    coord: [ -73.856077, 40.848447 ],
    street: 'Morris Park Ave',
    zipcode: '10462'
  },
  borough: 'Bronx',
  cuisine: 'Bakery',
  grades: [
    {
      date: ISODate('2014-03-03T00:00:00.000Z'),
      grade: 'A',
      score: 100
    }
  ],
  name: 'The Bagelery',
  rating: 4.5
}
```

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:

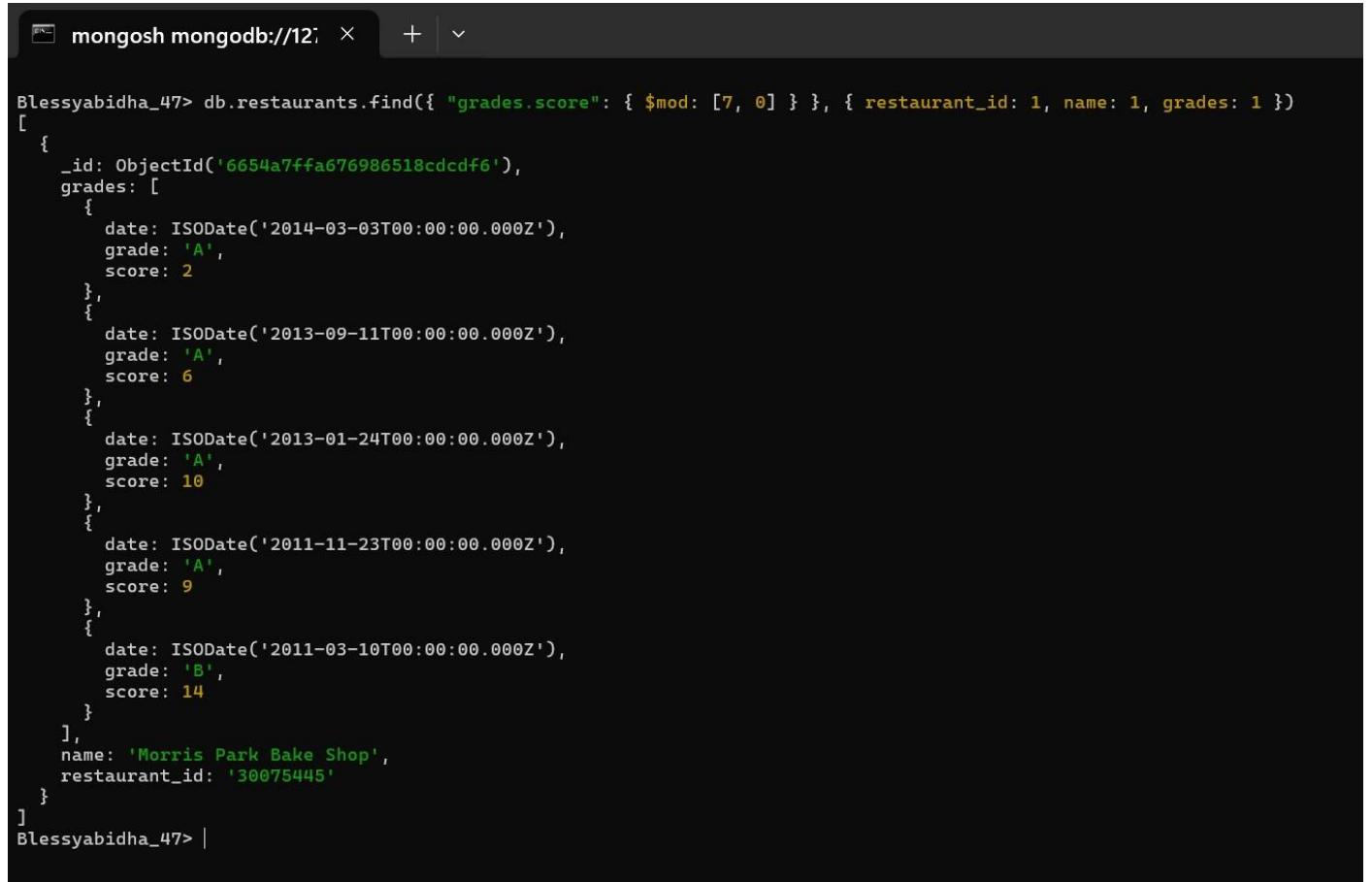
```
mongosh mongodb://127.0.0.1:27017/ | + | -| ↻
Blessyabidha_47> db.restaurants.find({ "address.coord": { $elemMatch: { $type: "double" } } })
[ {
  _id: ObjectId('6654a7ffa676986518cdcdf6'),
  address: {
    building: '1007',
    coord: [ -73.856077, 40.848447 ],
    street: 'Morris Park Ave',
    zipcode: '10462'
  },
  borough: 'Bronx',
  cuisine: 'Bakery',
  grades: [
    {
      date: ISODate('2014-03-03T00:00:00.000Z'),
      grade: 'A',
      score: 2
    },
    {
      date: ISODate('2013-09-11T00:00:00.000Z'),
      grade: 'A',
      score: 6
    },
    {
      date: ISODate('2013-01-24T00:00:00.000Z'),
      grade: 'A',
      score: 10
    },
    {
      date: ISODate('2011-11-23T00:00:00.000Z'),
      grade: 'A',
      score: 9
    },
    {
      date: ISODate('2011-03-10T00:00:00.000Z'),
      grade: 'B',
      score: 14
    }
  ],
  name: 'Morris Park Bake Shop',
  restaurant_id: '30075445'
}
]
Blessyabidha_47> |
```

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:



```
Blessyabidha_47> db.restaurants.find({ "grades.score": { $mod: [7, 0] } }, { restaurant_id: 1, name: 1, grades: 1 })
[
  {
    _id: ObjectId('6654a7ffa676986518cdcdf6'),
    grades: [
      {
        date: ISODate('2014-03-03T00:00:00.000Z'),
        grade: 'A',
        score: 2
      },
      {
        date: ISODate('2013-09-11T00:00:00.000Z'),
        grade: 'A',
        score: 6
      },
      {
        date: ISODate('2013-01-24T00:00:00.000Z'),
        grade: 'A',
        score: 10
      },
      {
        date: ISODate('2011-11-23T00:00:00.000Z'),
        grade: 'A',
        score: 9
      },
      {
        date: ISODate('2011-03-10T00:00:00.000Z'),
        grade: 'B',
        score: 14
      }
    ],
    name: 'Morris Park Bake Shop',
    restaurant_id: '30075445'
  }
]
Blessyabidha_47> |
```

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:



```
Blessyabidha_47> db.restaurants.find({ name: /mon/i }, { name: 1, borough: 1, "address.coord": 1, cuisine: 1 })
Blessyabidha_47> |
```

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:



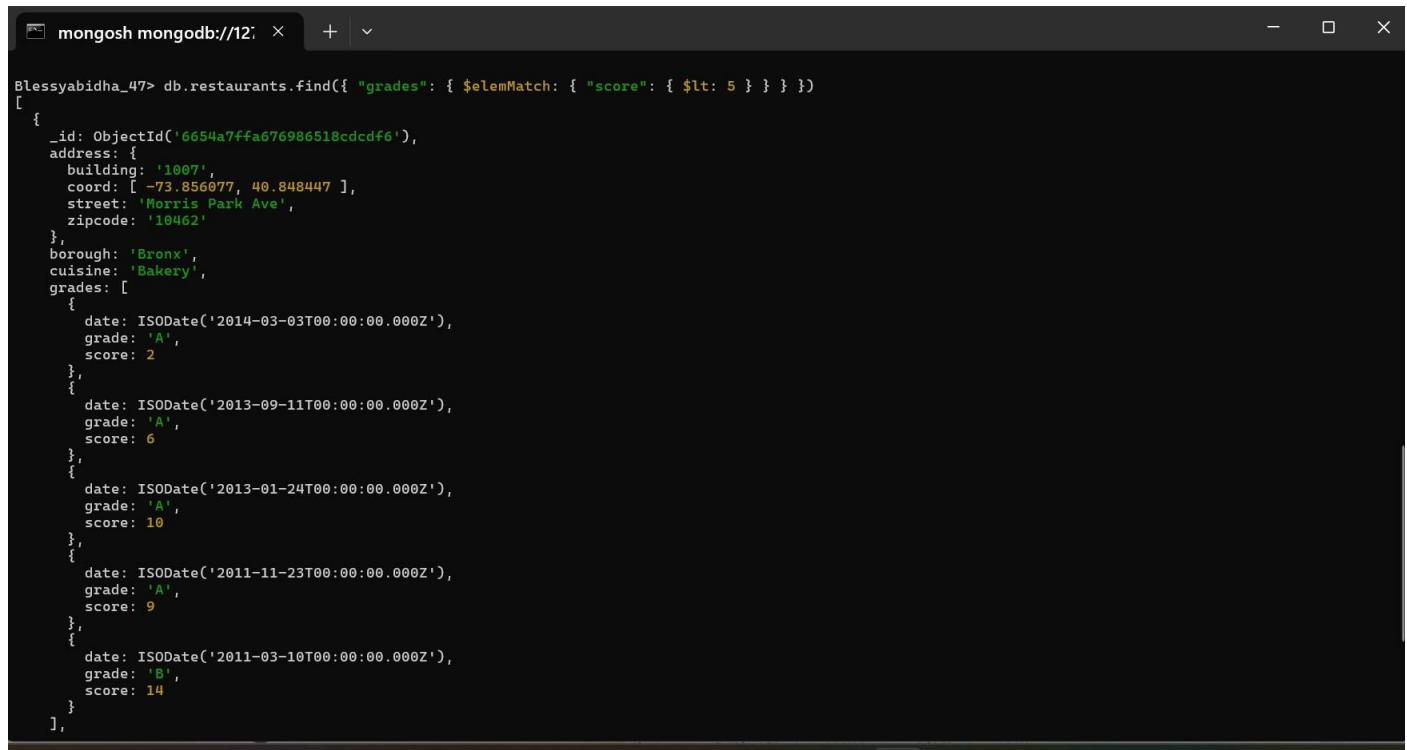
```
mongosh mongodb://127.0.0.1:27017
Blessyabidha_47> db.restaurants.find({ name: /^Mad/i }, { name: 1, borough: 1, "address.coord": 1, cuisine: 1 })
Blessyabidha_47> |
```

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:



```
mongosh mongodb://127.0.0.1:27017
Blessyabidha_47> db.restaurants.find({ "grades": { $elemMatch: { "score": { $lt: 5 } } } })
[
  {
    "_id": ObjectId('6654a7ffa676986518cdcdf6'),
    "address": {
      "building": '1007',
      "coord": [-73.856077, 40.848447],
      "street": 'Morris Park Ave',
      "zipcode": '10462'
    },
    "borough": 'Bronx',
    "cuisine": 'Bakery',
    "grades": [
      {
        "date": ISODate('2014-03-03T00:00:00.000Z'),
        "grade": 'A',
        "score": 2
      },
      {
        "date": ISODate('2013-09-11T00:00:00.000Z'),
        "grade": 'A',
        "score": 6
      },
      {
        "date": ISODate('2013-01-24T00:00:00.000Z'),
        "grade": 'A',
        "score": 10
      },
      {
        "date": ISODate('2011-11-23T00:00:00.000Z'),
        "grade": 'A',
        "score": 9
      },
      {
        "date": ISODate('2011-03-10T00:00:00.000Z'),
        "grade": 'B',
        "score": 14
      }
    ]
  }
]
```

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:



```
mongosh mongodb://127.0.0.1:27017
Blessyabidha_47> db.restaurants.find({ "grades": { $elemMatch: { "score": { $lt: 5 } } }, "borough": "Manhattan" })
Blessyabidha_47> |
```

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:



```
mongosh mongodb://127.0.0.1:27017
Blessyabidha_47> db.restaurants.find({ "grades": { $elemMatch: { "score": { $lt: 5 } } }, $or: [{ "borough": "Manhattan" }, { "borough": "Brooklyn" }] })
Blessyabidha_47> |
```

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:



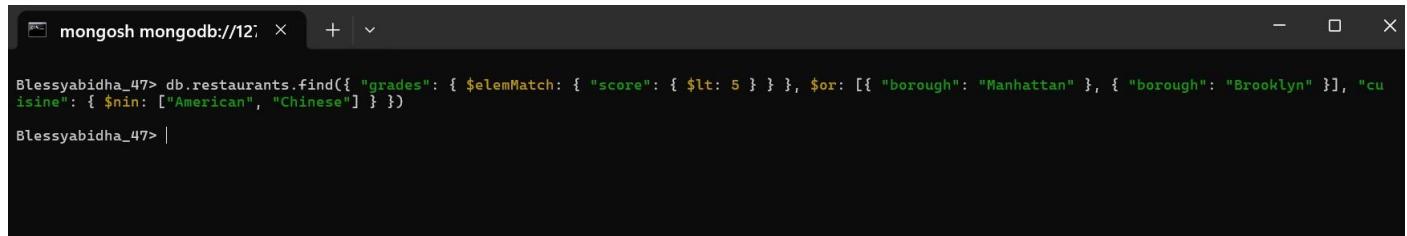
```
mongosh mongodb://127.0.0.1:27017
Blessyabidha_47> db.restaurants.find({ "grades": { $elemMatch: { "score": { $lt: 5 } } }, $or: [{ "borough": "Manhattan" }, { "borough": "Brooklyn" }], "cuisine": { $ne: "American" } })
Blessyabidha_47> |
```

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:



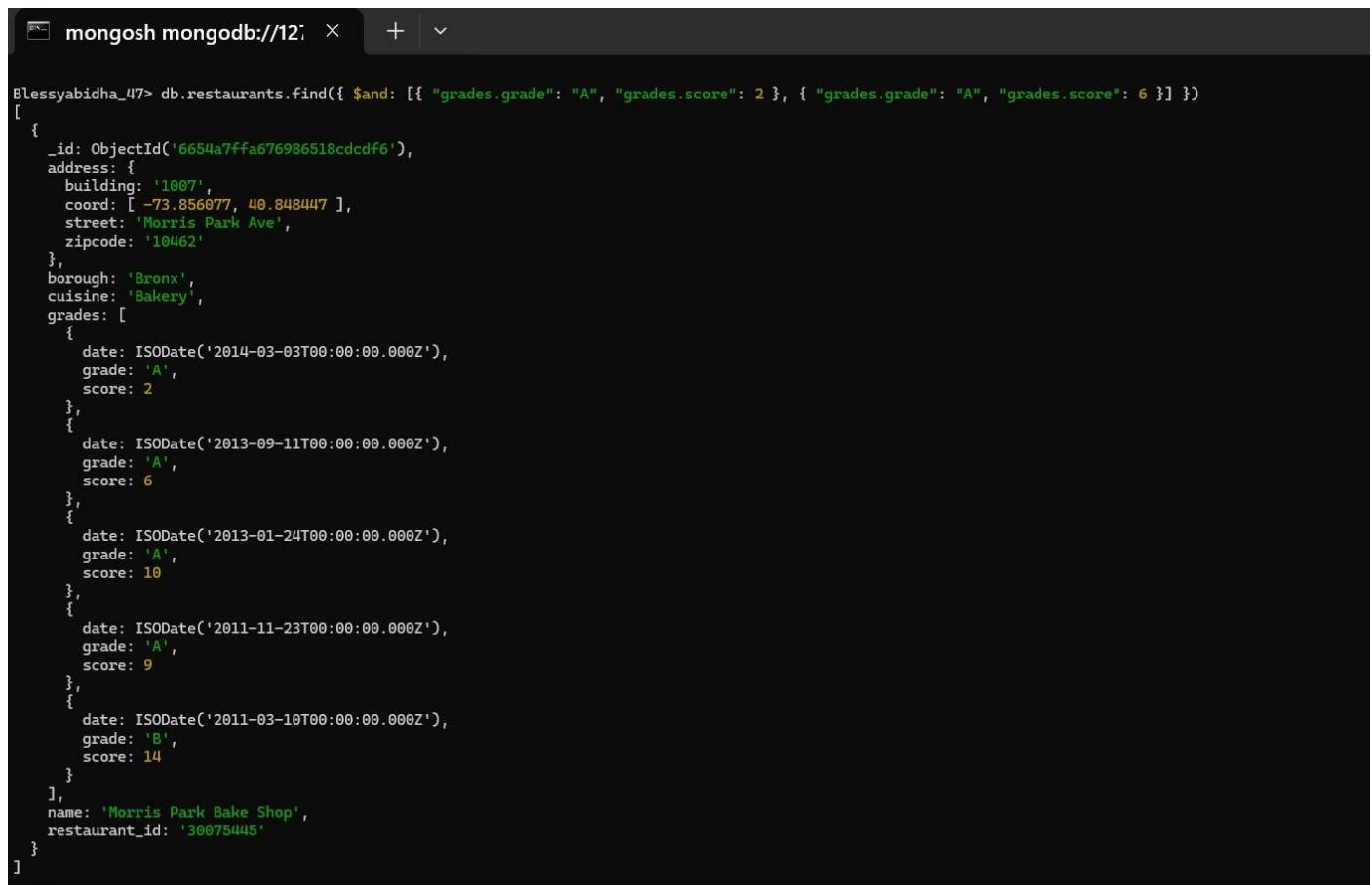
```
mongosh mongodb://127.0.0.1:27017
Blessyabidha_47> db.restaurants.find({ "grades": { $elemMatch: { "score": { $lt: 5 } } }, $or: [{ "borough": "Manhattan" }, { "borough": "Brooklyn" }], "cuisine": { $nin: ["American", "Chinese"] } })
Blessyabidha_47> |
```

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:



```
mongosh mongodb://127.0.0.1:27017
Blessyabidha_47> db.restaurants.find({ $and: [{ "grades.grade": "A", "grades.score": 2 }, { "grades.grade": "A", "grades.score": 6 }] })
[
  {
    _id: ObjectId('6654a7ffa676986518ccdf6'),
    address: {
      building: '1007',
      coord: [-73.856077, 40.848447],
      street: 'Morris Park Ave',
      zipcode: '10462'
    },
    borough: 'Bronx',
    cuisine: 'Bakery',
    grades: [
      {
        date: ISODate('2014-03-03T00:00:00.000Z'),
        grade: 'A',
        score: 2
      },
      {
        date: ISODate('2013-09-11T00:00:00.000Z'),
        grade: 'A',
        score: 6
      },
      {
        date: ISODate('2013-01-24T00:00:00.000Z'),
        grade: 'A',
        score: 10
      },
      {
        date: ISODate('2011-11-23T00:00:00.000Z'),
        grade: 'A',
        score: 9
      },
      {
        date: ISODate('2011-03-10T00:00:00.000Z'),
        grade: 'B',
        score: 14
      }
    ],
    name: 'Morris Park Bake Shop',
    restaurant_id: '30075445'
  }
]
```

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:



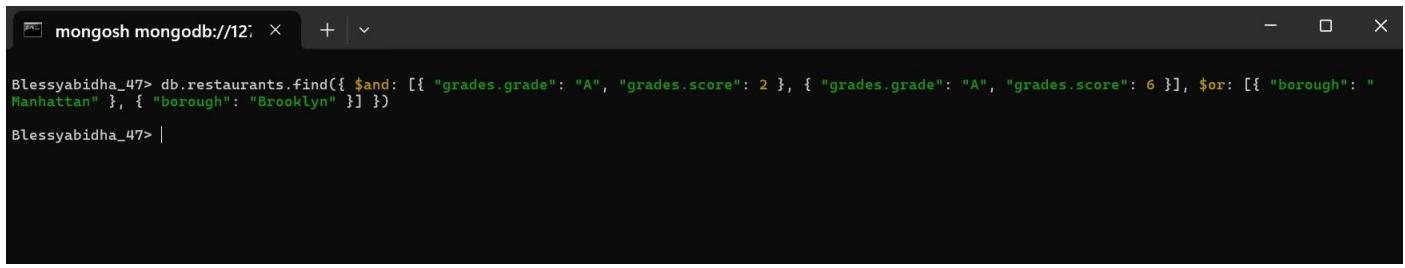
```
mongosh mongodb://127.0.0.1:27017
Blessyabidha_47> db.restaurants.find({ $and: [{ "grades.grade": "A", "grades.score": 2 }, { "grades.grade": "A", "grades.score": 6 }], "borough": "Manhattan" })
Blessyabidha_47>
```

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:



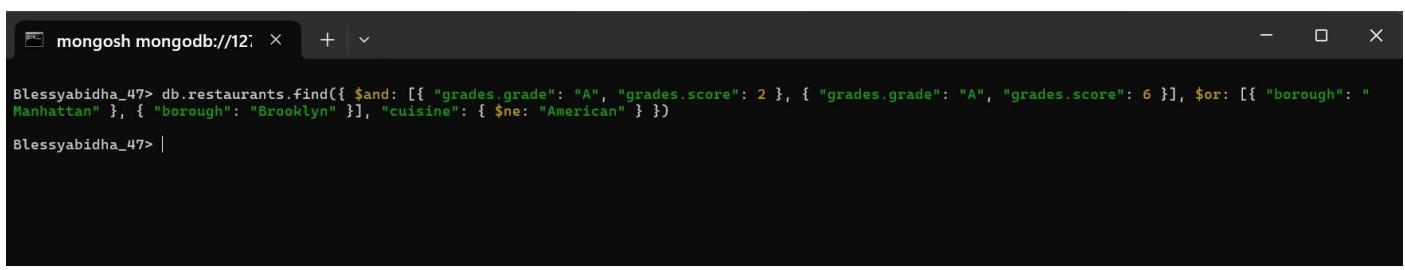
```
mongosh mongodb://127.0.0.1:27017
Blessyabidha_47> db.restaurants.find({ $and: [{ "grades.grade": "A", "grades.score": 2 }, { "grades.grade": "A", "grades.score": 6 }], $or: [{ "borough": "Manhattan" }, { "borough": "Brooklyn" }] })
Blessyabidha_47> |
```

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:



```
mongosh mongodb://127.0.0.1:27017
Blessyabidha_47> db.restaurants.find({ $and: [{ "grades.grade": "A", "grades.score": 2 }, { "grades.grade": "A", "grades.score": 6 }], $or: [{ "borough": "Manhattan" }, { "borough": "Brooklyn" }], "cuisine": { $ne: "American" } })
Blessyabidha_47> |
```

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

QUERY:

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

OUTPUT:



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

QUERY:

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

OUTPUT:

```

mongosh mongodb://127.0.0.1:27017/ -+ | v
Blessyabidha_47> db.restaurants.find({ $or: [{ "grades.score": 2 }, { "grades.score": 6 }] })
[ {
  _id: ObjectId('6654a7ffa676986518cdcdf6'),
  address: {
    building: '1007',
    coord: [ -73.856077, 40.848447 ],
    street: 'Morris Park Ave',
    zipcode: '10462'
  },
  borough: 'Bronx',
  cuisine: 'Bakery',
  grades: [
    {
      date: ISODate('2014-03-03T00:00:00.000Z'),
      grade: 'A',
      score: 2
    },
    {
      date: ISODate('2013-09-11T00:00:00.000Z'),
      grade: 'A',
      score: 6
    },
    {
      date: ISODate('2013-01-24T00:00:00.000Z'),
      grade: 'A',
      score: 10
    },
    {
      date: ISODate('2011-11-23T00:00:00.000Z'),
      grade: 'A',
      score: 9
    },
    {
      date: ISODate('2011-03-10T00:00:00.000Z'),
      grade: 'B',
      score: 14
    }
  ],
  name: 'Morris Park Bake Shop',
  restaurant_id: '30075445'
}
]

```

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:



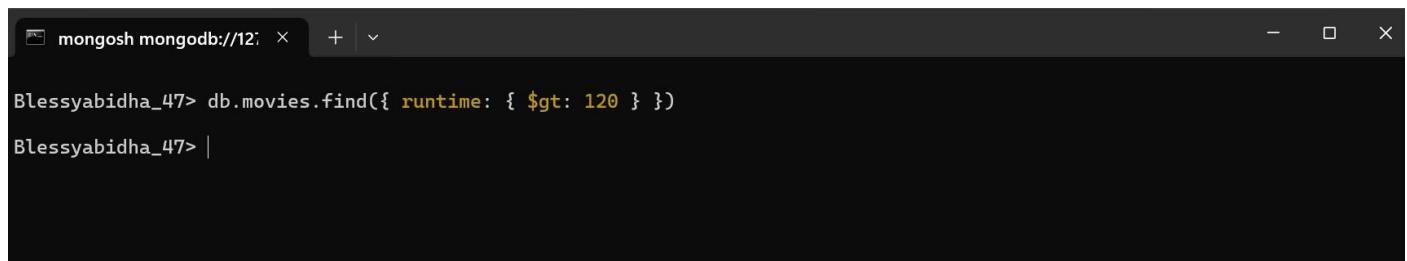
```
mongosh mongodb://127.0.0.1:27017
Blessyabidha_47> db.movies.find({ year: 1893 })
Blessyabidha_47> |
```

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:



```
mongosh mongodb://127.0.0.1:27017
Blessyabidha_47> db.movies.find({ runtime: { $gt: 120 } })
Blessyabidha_47> |
```

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

QUERY:

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

OUTPUT:

```
Blessyabidha_47> db.movies.find({ genres: 'Short' })
[{"_id": ObjectId('573a1390f29313caabcd42e8'),
  plot: "A group of bandits stage a brazen train hold-up, only to find a determined posse hot on their heels.",
  genres: [ 'Short', 'Western' ],
  runtime: 11,
  cast: [
    "A.C. Abadie",
    "Gilbert M. 'Broncho Billy' Anderson",
    "George Barnes",
    "Justus D. Barnes"
  ],
  poster: 'https://m.media-amazon.com/images/M/MV5BMTU3NjE5NzYtYTYYNS00MDVmLWIwYjgtMmYWIXZDYyNzU2XkEyXkFqcGdeQXVyNzQzNzQxNzI@._V1_SY1000_SX677_AL_.jpg',
  title: 'The Great Train Robbery',
  fullplot: "Among the earliest existing films in American cinema – notable as the first film that presented a narrative story to tell – it depicts a group of cowboy outlaws who hold up a train and rob the passengers. They are then pursued by a Sheriff's posse. Several scenes have color included – all hand tinted.",
  languages: [ 'English' ],
  released: ISODate('1903-12-01T00:00:00.000Z'),
  directors: [ 'Edwin S. Porter' ],
  rated: 'TV-G',
  awards: { wins: 1, nominations: 0, text: '1 win.' },
  lastupdated: '2015-08-13 00:27:59.177000000',
  year: 1903,
  imdb: { rating: 7.4, votes: 9847, id: 439 },
  countries: [ 'USA' ],
  type: 'movie',
  tomatoes: {
    viewer: { rating: 3.7, numReviews: 2559, meter: 75 },
    fresh: 6,
    critic: { rating: 7.6, numReviews: 6, meter: 100 },
    rotten: 0,
    lastUpdated: ISODate('2015-08-08T19:16:10.000Z')
  }
}
Blessyabidha_47> |
```

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:

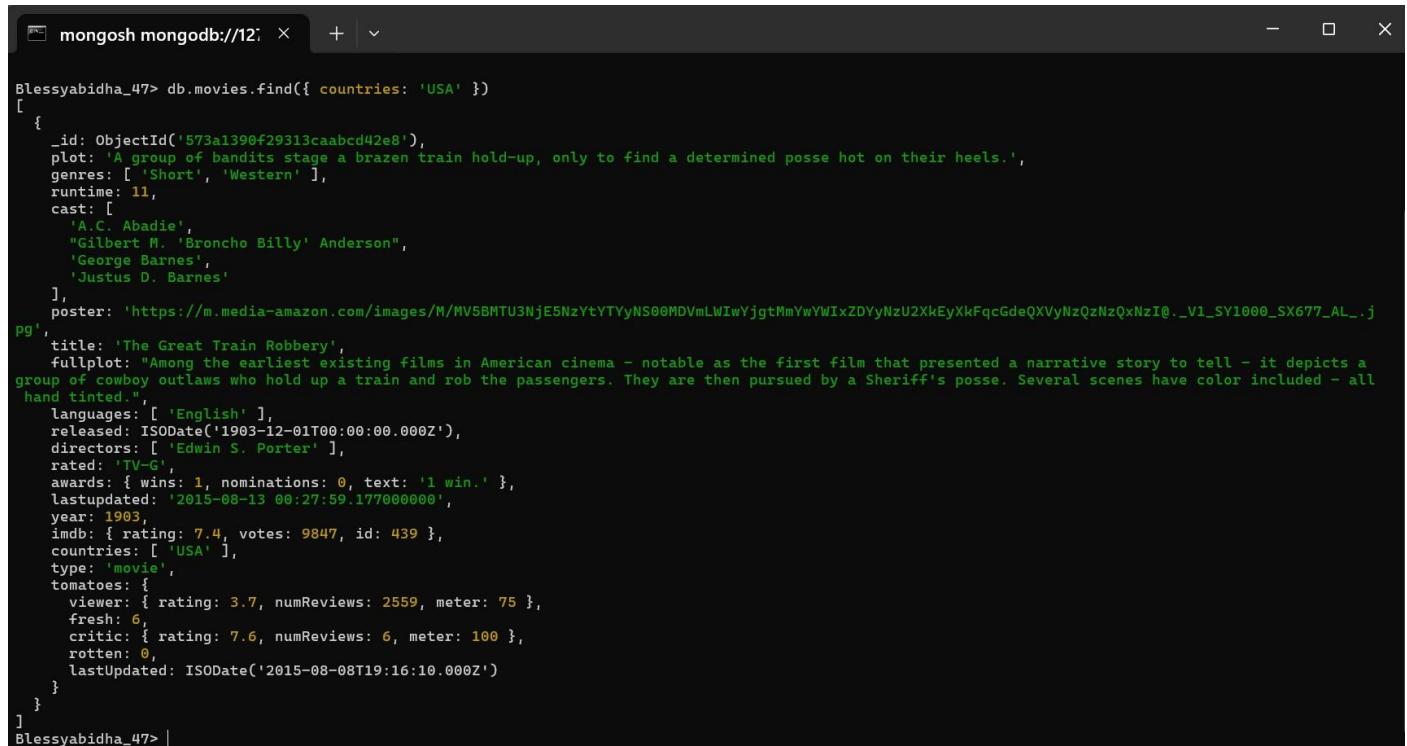
```
Blessyabidha_47> db.movies.find({ directors: 'William K.L. Dickson' })
Blessyabidha_47> |
```

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:



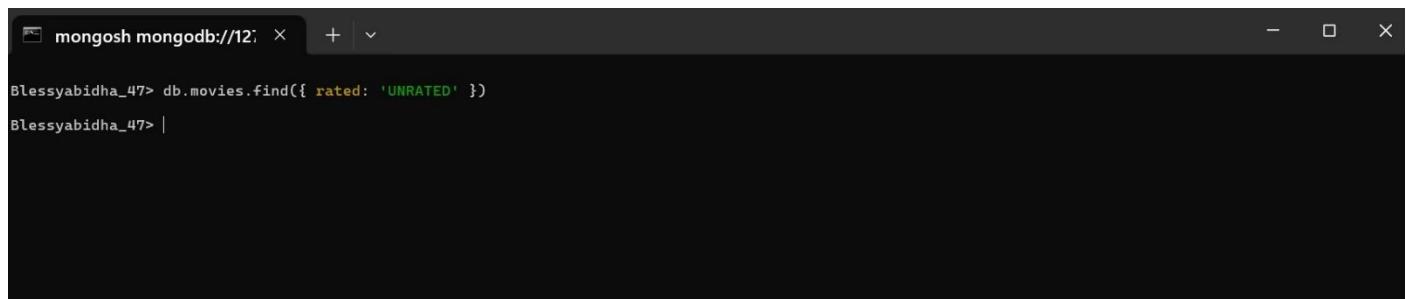
```
Blessyabidha_47> db.movies.find({ countries: 'USA' })
[ {
    _id: ObjectId('573a1390f29313caabcd42e8'),
    plot: 'A group of bandits stage a brazen train hold-up, only to find a determined posse hot on their heels.',
    genres: [ 'Short', 'Western' ],
    runtime: 11,
    cast: [
        'A.C. Abadie',
        "Gilbert M. 'Broncho Billy' Anderson",
        'George Barnes',
        'Justus D. Barnes'
    ],
    poster: 'https://m.media-amazon.com/images/M/MV5BMTU3NjE5NzYtYTYYNS00MDVmLWIwYjgtMmYwYWIxZDYyNzU2XkEyXkFqcGdeQXVyNzQzNzQzI@._V1_SX677_AL_.jpg',
    title: 'The Great Train Robbery',
    fullplot: "Among the earliest existing films in American cinema - notable as the first film that presented a narrative story to tell - it depicts a group of cowboy outlaws who hold up a train and rob the passengers. They are then pursued by a Sheriff's posse. Several scenes have color included - all hand tinted.",
    languages: [ 'English' ],
    released: ISODate('1903-12-01T00:00:00.000Z'),
    directors: [ 'Edwin S. Porter' ],
    rated: 'TV-G',
    awards: { wins: 1, nominations: 0, text: '1 win.' },
    lastupdated: '2015-08-13 00:27:59.177000000',
    year: 1903,
    imdb: { rating: 7.4, votes: 9847, id: 439 },
    countries: [ 'USA' ],
    type: 'movie',
    tomatoes: {
        viewer: { rating: 3.7, numReviews: 2559, meter: 75 },
        fresh: 6,
        critic: { rating: 7.6, numReviews: 6, meter: 100 },
        rotten: 0,
        lastUpdated: ISODate('2015-08-08T19:16:10.000Z')
    }
}
]
Blessyabidha_47> |
```

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

QUERY:

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

OUTPUT:



```
Blessyabidha_47> db.movies.find({ rated: 'UNRATED' })
Blessyabidha_47> |
```

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:

```
mongosh mongodb://127.0.0.1:27017/movies
Blessyabidha_47> db.movies.find({ 'imdb.votes': { $gt: 1000 } })
[ {
  _id: ObjectId('573a1390f29313caabcd42e8'),
  plot: 'A group of bandits stage a brazen train hold-up, only to find a determined posse hot on their heels.',
  genres: [ 'Short', 'Western' ],
  runtime: 11,
  cast: [
    'A.C. Abadie',
    'Gilbert M. "Broncho Billy" Anderson',
    'George Barnes',
    'Justus D. Barnes'
  ],
  poster: 'https://m.media-amazon.com/images/M/MV5BMTU3NjE5NzYtYTYYNS00MDVmLWIwYjgtMmYwYWIxZDYyNzU2XkEyXkFqcGdeQXVyNzQzNzQzNzI@._V1_SY1000_SX677_AL_.jpg',
  title: 'The Great Train Robbery',
  fullplot: "Among the earliest existing films in American cinema - notable as the first film that presented a narrative story to tell - it depicts a group of cowboy outlaws who hold up a train and rob the passengers. They are then pursued by a Sheriff's posse. Several scenes have color included - all hand tinted.",
  languages: [ 'English' ],
  released: ISODate('1903-12-01T00:00:00.000Z'),
  directors: [ 'Edwin S. Porter' ],
  rated: 'TV-G',
  awards: { wins: 1, nominations: 0, text: '1 win.' },
  lastupdated: '2015-08-13 00:27:59.177000000',
  year: 1903,
  imdb: { rating: 7.4, votes: 9847, id: 439 },
  countries: [ 'USA' ],
  type: 'movie',
  tomatoes: {
    viewer: { rating: 3.7, numReviews: 2559, meter: 75 },
    fresh: 6,
    critic: { rating: 7.6, numReviews: 6, meter: 100 },
    rotten: 0,
    lastUpdated: ISODate('2015-08-08T19:16:10.000Z')
  }
}
]
Blessyabidha_47> |
```

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:

```
Blessyabidha_47> db.movies.find({ 'imdb.rating': { $gt: 7 } })
[ {
    _id: ObjectId('573a1390f29313caabcd42e8'),
    plot: 'A group of bandits stage a brazen train hold-up, only to find a determined posse hot on their heels.',
    genres: [ 'Short', 'Western' ],
    runtime: 11,
    cast: [
        'A.C. Abadie',
        "Gilbert M. 'Broncho Billy' Anderson",
        'George Barnes',
        'Justus D. Barnes'
    ],
    poster: 'https://m.media-amazon.com/images/M/MV5BMTU3NjE5NzYtYTYYNS00MDVmLWIwYjgtMmYwYWIxZDYyNzU2XkEyXkFqcGdeQXVyNzQzNzQxNzI@._V1_SX677_AL_.jpg',
    title: 'The Great Train Robbery',
    fullplot: "Among the earliest existing films in American cinema - notable as the first film that presented a narrative story to tell - it depicts a group of cowboy outlaws who hold up a train and rob the passengers. They are then pursued by a Sheriff's posse. Several scenes have color included - all hand tinted.",
    languages: [ 'English' ],
    released: ISODate('1903-12-01T00:00:00.000Z'),
    directors: [ 'Edwin S. Porter' ],
    rated: 'TV-G',
    awards: { wins: 1, nominations: 0, text: '1 win.' },
    lastupdated: '2015-08-13 00:27:59.177000000',
    year: 1903,
    imdb: { rating: 7.4, votes: 9847, id: 439 },
    countries: [ 'USA' ],
    type: 'movie',
    tomatoes: {
        viewer: { rating: 3.7, numReviews: 2559, meter: 75 },
        fresh: 6,
        critic: { rating: 7.6, numReviews: 6, meter: 100 },
        rotten: 0,
        lastUpdated: ISODate('2015-08-08T19:16:10.000Z')
    }
}
]
Blessyabidha_47> |
```

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:

```
Blessyabidha_47> db.movies.find({ 'tomatoes.viewer.rating': { $gt: 4 } })
Blessyabidha_47> |
```

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

QUERY:

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

OUTPUT:

```
Blessyabidha_47> db.movies.find({ 'awards.wins': { $gt: 0 } })
[ {
    _id: ObjectId('573a1390f29313caabcd42e8'),
    plot: 'A group of bandits stage a brazen train hold-up, only to find a determined posse hot on their heels.',
    genres: [ 'Short', 'Western' ],
    runtime: 11,
    cast: [
        'A.C. Abadie',
        "Gilbert M. 'Broncho Billy' Anderson",
        'George Barnes',
        'Justus D. Barnes'
    ],
    poster: 'https://m.media-amazon.com/images/M/MV5BMTU3NjE5NzYtYTlyNS00MDVmLWIwYjgtMmYwYWIxZDYyNzU2XkEyXkFqcGdeQXVyNzQzNzQxNzI@._V1_SY1000_SX677_AL_.jpg',
    title: 'The Great Train Robbery',
    fullplot: "Among the earliest existing films in American cinema - notable as the first film that presented a narrative story to tell - it depicts a group of cowboy outlaws who hold up a train and rob the passengers. They are then pursued by a Sheriff's posse. Several scenes have color included - all hand tinted.",
    languages: [ 'English' ],
    released: ISODate('1903-12-01T00:00:00.000Z'),
    directors: [ 'Edwin S. Porter' ],
    rated: 'TV-G',
    awards: { wins: 1, nominations: 0, text: '1 win.' },
    lastupdated: '2015-08-13 00:27:59.177000000',
    year: 1903,
    imdb: { rating: 7.4, votes: 9847, id: 439 },
    countries: [ 'USA' ],
    type: 'movie',
    tomatoes: {
        viewer: { rating: 3.7, numReviews: 2559, meter: 75 },
        fresh: 6,
        critic: { rating: 7.6, numReviews: 6, meter: 100 },
        rotten: 0,
        lastUpdated: ISODate('2015-08-08T19:16:10.000Z')
    }
}
]
Blessyabidha_47> |
```

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:

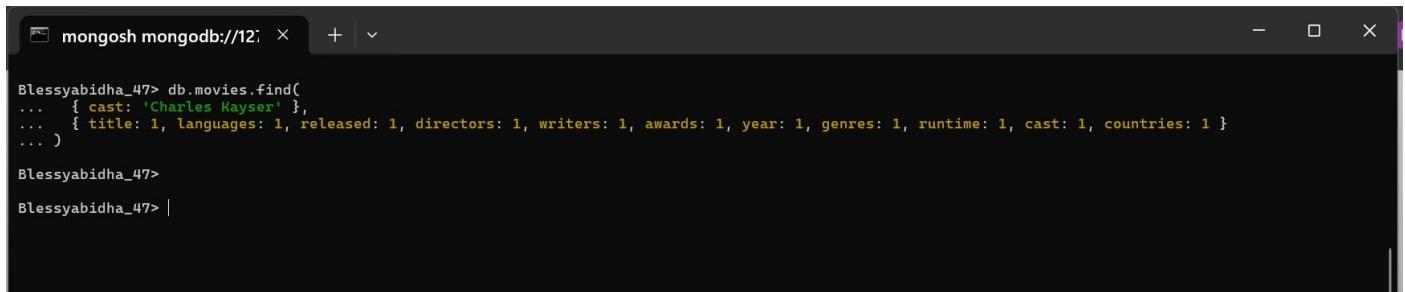
```
Blessyabidha_47> db.movies.find(
...   { 'awards.nominations': { $gt: 0 } },
...   { title: 1, languages: 1, released: 1, directors: 1, writers: 1, awards: 1, year: 1, genres: 1, runtime: 1, cast: 1, countries: 1 }
... )
Blessyabidha_47>
Blessyabidha_47> |
```

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:



```
Blessyabidha_47> 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 }
... )
Blessyabidha_47>
Blessyabidha_47> |
```

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:



```
Blessyabidha_47> db.movies.find(
...   { released: ISODate("1893-05-09T00:00:00.000Z") },
...   { title: 1, languages: 1, released: 1, directors: 1, writers: 1, countries: 1 }
... )
Blessyabidha_47>
Blessyabidha_47> |
```

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:



```
Blessyabidha_47> db.movies.find(
...   { title: /scene/i },
...   { title: 1, languages: 1, released: 1, directors: 1, writers: 1, countries: 1 }
... )
Blessyabidha_47> |
```

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

RESULT: