

Creating and Managing Tables

EX_NO:1

DATE: 18/2/2024

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

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

QUERY:

```
Create table DEPT01(ID int,Name varchar(20), Dept_name varchar(20) ,Location_ip int);
```

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 ('chithra k chithu') are also present. The main area is titled 'SQL Commands' with a 'Schema' dropdown set to 'WKSP_CHITHU'. The code input field contains the command: 'create table DEPT01(ID int,NAME varchar(20),DEPT_NAME varchar(20),LOCATION_IP int);'. The results tab shows the output: 'Table created.' and '0.03 seconds' execution time.

```
create table DEPT01(ID int,NAME varchar(20),DEPT_NAME varchar(20),LOCATION_IP int);
```

Table created.
0.03 seconds

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 EMPA03(Employee_id number(9), first_name varchar(20),last_name varchar(20),Dept_id int);
```

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. In the SQL Commands pane, the following SQL command is entered:

```
1 create table EMPA03(EMP_ID int, LAST_NAME varchar(20),First_Name varchar(20),DEPT_ID int);
```

In the Results pane, the output is:

```
Table created.  
0.03 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 EMPA01 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, and Gallery. On the right, there's a search bar, a user icon for 'chithra k chithu', and a schema dropdown set to 'WKSP_CHITHU'. The main area is titled 'SQL Commands' with a language dropdown set to 'SQL'. Below it, there are buttons for 'Rows' (set to 10), 'Clear Command', and 'Find Tables'. The SQL editor contains the command: '1 ALTER TABLE EMPA01 modify (LAST_NAME varchar(50));'. The results tab is selected, showing the output: 'Table altered.' and '0.06 seconds'. Other tabs include Explain, Describe, Saved SQL, and History.

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 employees02(id number(7),first_name varchar2(25),Last_name varchar2(25),Salary int not null,Dept_id number(7));
```

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 'chithra k chithu', and a schema dropdown set to 'WKSP_CHITHU'. The main area is titled 'SQL Commands' with a language dropdown set to 'SQL'. Below it, there are buttons for 'Rows' (set to 10), 'Clear Command', and 'Find Tables'. The SQL editor contains the command: '1 create table EMPLOYEES02(EMP_ID int,FIRST_NAME varchar(20),LAST_NAME varchar(20), SALARY int, DEPT_ID int);'. The results tab is selected, showing the output: 'Table created.' and '0.04 seconds'. Other tabs include Explain, Describe, Saved SQL, and History.

5.Drop the EMP table.

QUERY:

Drop table EMPA01;

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 'chithra k chithu', and a 'Run' button. The main area is titled 'SQL Commands'. It shows a single line of code: '1 DROP TABLE EMPA01;'. Below the code, the 'Results' tab is selected, showing the output: 'Table dropped.' and '0.08 seconds'. Other tabs include 'Explain', 'Describe', 'Saved SQL', and 'History'.

6.Rename the EMPLOYEES2 table as EMP.

QUERY:

Rename employees2 to EMP01;

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 'chithra k chithu', and a 'Run' button. The main area is titled 'SQL Commands'. It shows a single line of code: '1 RENAME EMPLOYEES02 TO EMP01;'. Below the code, the 'Results' tab is selected, showing the output: 'Statement processed.' and '0.14 seconds'. Other tabs include 'Explain', 'Describe', 'Saved SQL', and 'History'.

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

QUERY:

```
comment on table dept01 is 'Department info';
comment on table empa is Employee info';
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. In the SQL Commands pane, the following command is entered and executed:

```
1 COMMENT on table DEPT01 is 'Department info';
1 COMMENT on table empa is Employee info';
```

The Results pane displays the output:

```
Statement processed.
0.01 seconds
```

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

QUERY:

```
Alter table EMPA03 drop column first_name;
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. In the SQL Commands pane, the following command is entered and executed:

```
1 alter table EMPA03 drop column First_Name;
```

The Results pane displays the output:

```
Table altered.
0.09 seconds
```

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

RESULT:

Thus the Query for creating and managing tables are executed successfully.

MANIPULATING DATA

EX_NO:2

DATE:22/2/2024

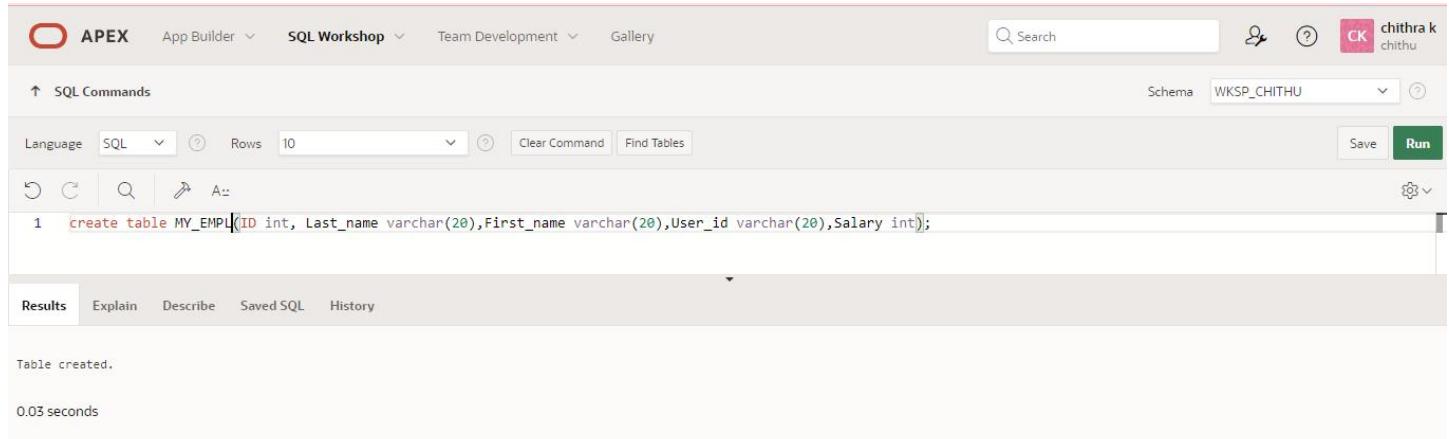
1.Create MY_EMPLOYEE table with the following structure:-

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_empl (id int not null,last_name varchar(20),first_name varchar(30),user_id varchar(30),Salary int not null);
```

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 'chithra k chithu', and a schema dropdown set to 'WKSP_CHITHU'. The main workspace is titled 'SQL Commands' and contains a single line of SQL code: 'create table MY_EMPL(id int, Last_name varchar(20),First_name varchar(20),User_id varchar(20),Salary int);'. Below the code, the 'Results' tab is selected, showing the output: 'Table created.' and '0.03 seconds'. Other tabs available include Explain, Describe, Saved SQL, and History.

```
create table MY_EMPL(id int, Last_name varchar(20),First_name varchar(20),User_id varchar(20),Salary int);
```

Table created.
0.03 seconds

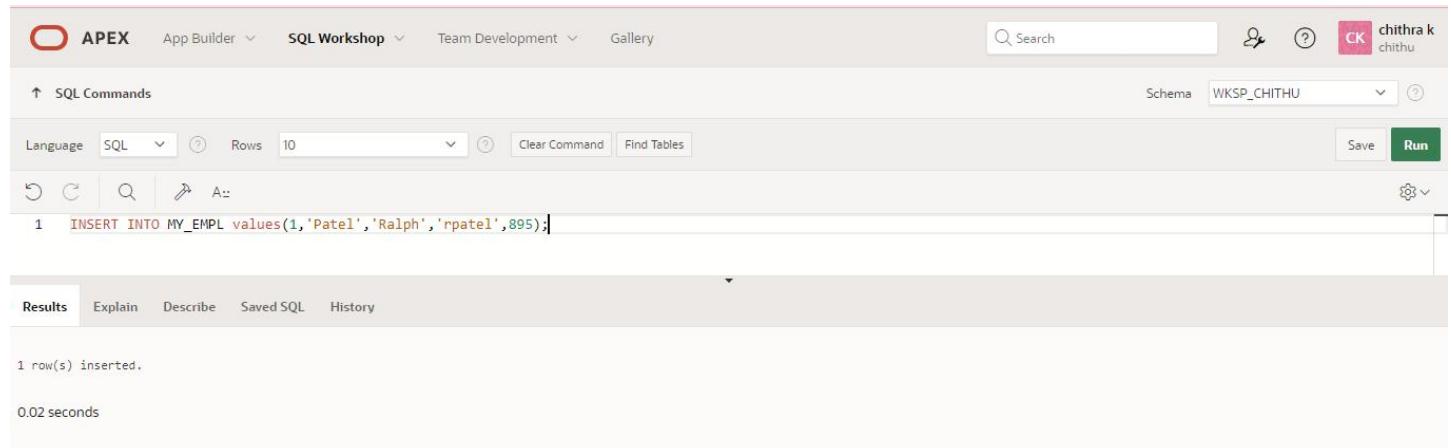
2.Add the first and second rows data to MY_EMPLOYEE table from the following sample data.

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_empL values (1,'patel','ralph','rpatel',895);
Insert into my_empL values(2,'Dances','Betty','Bdancs',890);
```

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, and the schema name WKSP_CHITHU. The main workspace is titled 'SQL Commands' and contains a single line of SQL code: '1 INSERT INTO MY_EMPL values(1,'Patel','Ralph','rpatel',895);'. Below the code, the results tab is selected, showing the output: '1 row(s) inserted.' and '0.02 seconds'. Other tabs available are Explain, Describe, Saved SQL, and History.

```
1 INSERT INTO MY_EMPL values(1,'Patel','Ralph','rpatel',895);
```

Results

1 row(s) inserted.
0.02 seconds

3. Display the table with values.

QUERY:

Select * from my_empl order by 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 search bar, a user icon for 'chithra k', and a schema dropdown set to 'WKSP_CHITHU'. The main area has tabs for Language (SQL selected), Rows (10), Clear Command, and Find Tables. Below is a command input field containing '1 Select * from MY_EMPL order by ID;'. The results tab is active, displaying a table with columns: ID, LAST_NAME, FIRST_NAME, USER_ID, and SALARY. Two rows are shown: ID 1 with LAST_NAME 'Patel', FIRST_NAME 'Ralph', USER_ID 'rpatel', and SALARY 895; ID 2 with LAST_NAME 'Dancs', FIRST_NAME 'Betty', USER_ID 'bdancs', and SALARY 860. A note at the bottom says '2 rows returned in 0.01 seconds'.

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

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

QUERY:

Insert into my_empl values (3,'Biri','Ben','BBiri',1100);
Insert into my_empl values (4,'Newman','Chad','Cnewman',790);

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 'chithra k', and a schema dropdown set to 'WKSP_CHITHU'. The main area has tabs for Language (SQL selected), Rows (10), Clear Command, and Find Tables. Below is a command input field containing '1 INSERT INTO MY_EMPL values(3,'Biri','Ben','bbiri',1100);'. The results tab is active, displaying the message '1 row(s) inserted.' and '0.00 seconds'.

5. Make the data additions permanent.

QUERY:

Select * from my_empl order by id;

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The SQL command entered is "select * from MY_EMPL order by ID;". The results pane displays a table with four rows of employee data:

ID	LAST_NAME	FIRST_NAME	USER_ID	SALARY
1	Patel	Ralph	rpatel	895
2	Dancs	Betty	bdancs	860
3	Biri	Ben	bbiri	1100
4	Newman	Chad	Cnewman	750

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

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

QUERY:

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

OUTPUT:

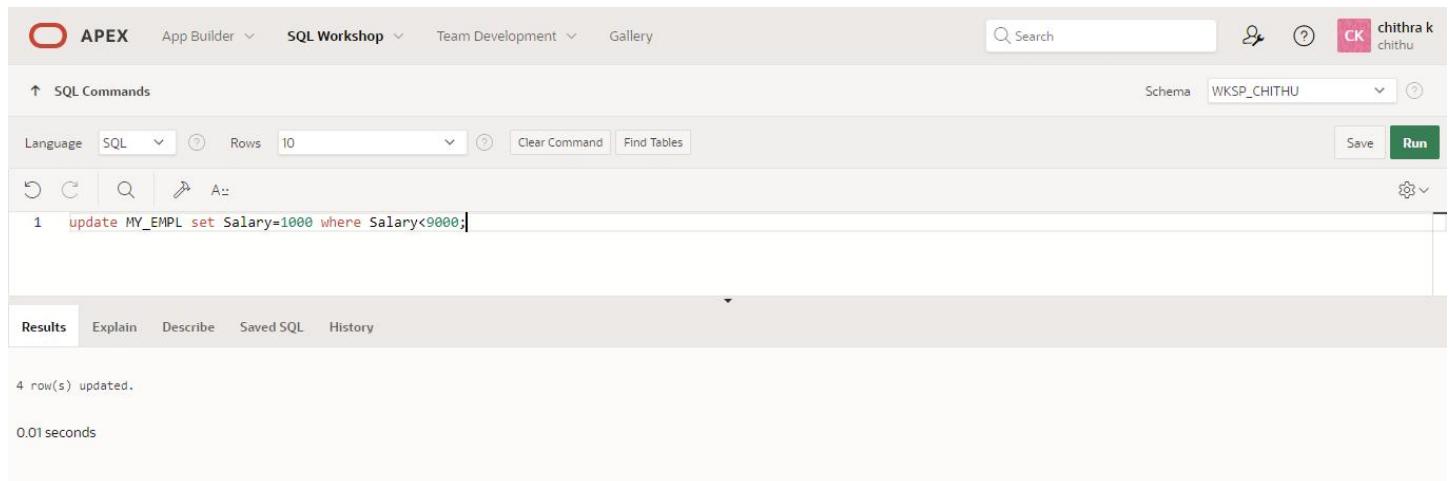
The screenshot shows the Oracle SQL Workshop interface. The SQL command entered is "update MY_EMPL set Last_name='Drexler' where ID=3;". The results pane shows the message "1 row(s) updated." and "0.01 seconds".

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

QUERY:

Update my_empl set salary=1000 where salary<=900;

OUTPUT:



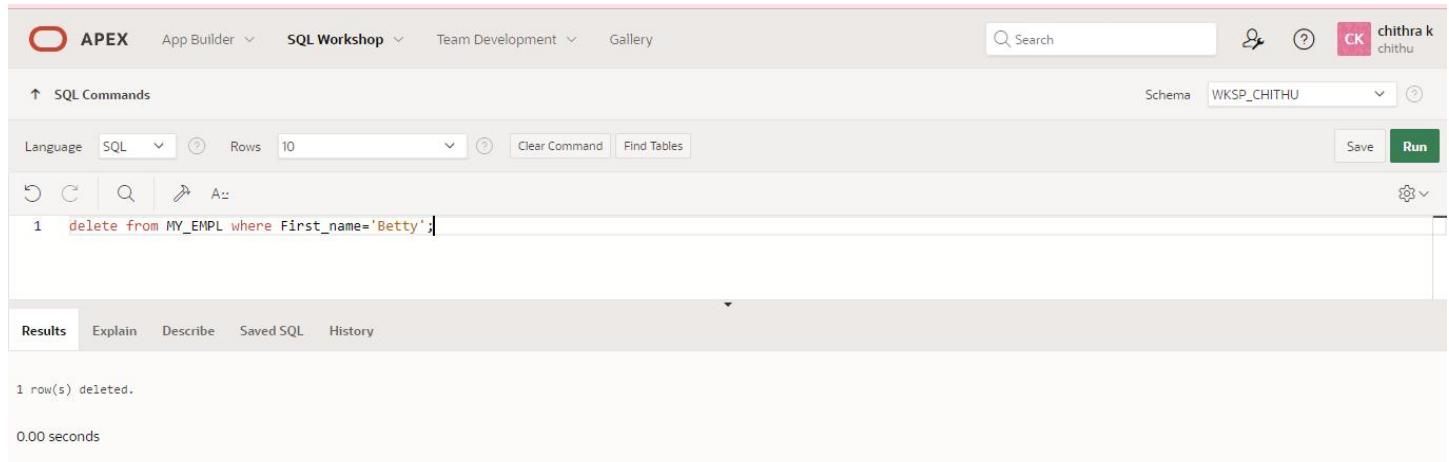
The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there's a search bar, a user icon for 'chithra k chithu', and a 'Run' button. The main area is titled 'SQL Commands'. It has dropdowns for 'Language' (set to SQL) and 'Rows' (set to 10). Buttons for 'Clear Command' and 'Find Tables' are visible. The SQL command entered is: `1 update MY_EMPL set Salary=1000 where Salary<9000;`. Below the command, the results tab is selected, showing the output: `4 row(s) updated.` and `0.01 seconds`.

8.Delete Betty dancs from MY_EMPLOYEE table.

QUERY:

Delete from my_empl where first_name='Betty';

OUTPUT:



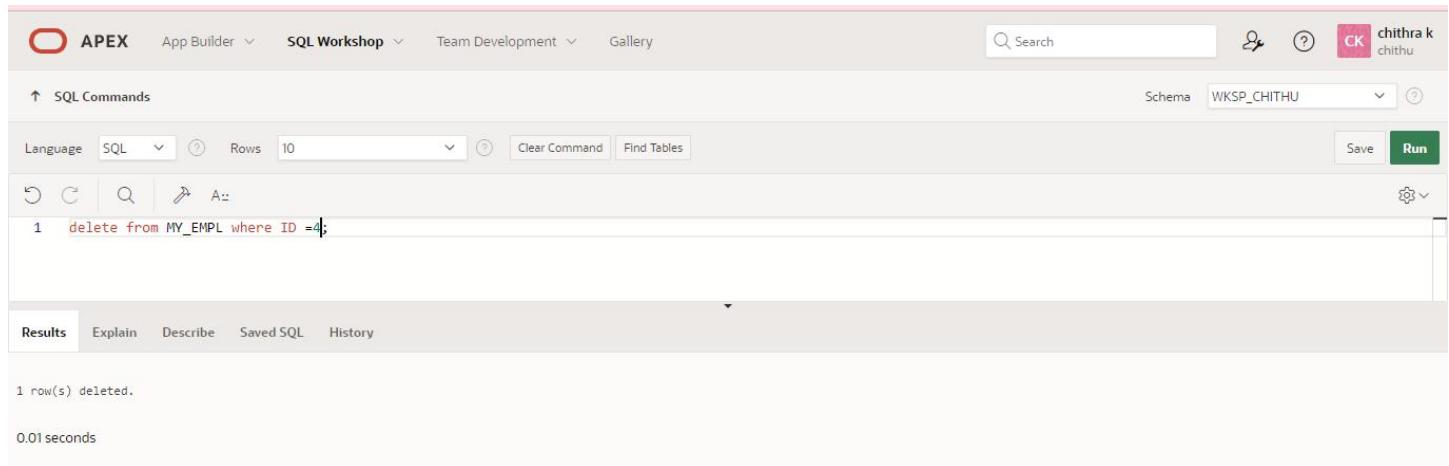
The screenshot shows the Oracle SQL Workshop interface, identical to the previous one but with a different query. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there's a search bar, a user icon for 'chithra k chithu', and a 'Run' button. The main area is titled 'SQL Commands'. It has dropdowns for 'Language' (set to SQL) and 'Rows' (set to 10). Buttons for 'Clear Command' and 'Find Tables' are visible. The SQL command entered is: `1 delete from MY_EMPL where First_name='Betty';`. Below the command, the results tab is selected, showing the output: `1 row(s) deleted.` and `0.00 seconds`.

9.Empty the fourth row of the emp table.

QUERY:

Delete from my_empl where id=4;

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 'chithra k', and session information for 'WKSP_CHITHU'. The main workspace is titled 'SQL Commands' with a 'Language' dropdown set to 'SQL'. Below it, a command line shows the SQL statement: '1 delete from MY_EMPL where ID =4;'. The results tab is selected, displaying the output: '1 row(s) deleted.' and '0.01 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:

Thus the Queries for manipulating data are executed successfully.

INCLUDING CONSTRAINTS

EX_NO:3

DATE:29/2/2024

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.

Create table emploabc (Id number(6), last_name varchar(30),email varchar(30),salary number(8,2),constraint my_emp_id_pk Primary Key(ID));

OUTPUT:

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

```
1 create table emploabc(Id number(4), LAST_NAME varchar(15), FIRST_NAME varchar(15),
2 CONSTRAINT my_emp_id_pk PRIMARY KEY(Id));
```

In the Results tab, the output is:

```
Table created.  
0.05 seconds
```

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

QUERY:

Create table depart (Id number(6),last_name varchar(30),first_name varchar(30),email varchar(30),constraint my_dept_id_pk ,Primary key((Id));

OUTPUT:

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

```
1 create table depart(Id number(5), Last_name varchar(15), First_name varchar(15),
2 constraint my_dept_id_pk primary key(Id));
```

In the Results tab, the output is:

```
Table created.  
0.06 seconds
```

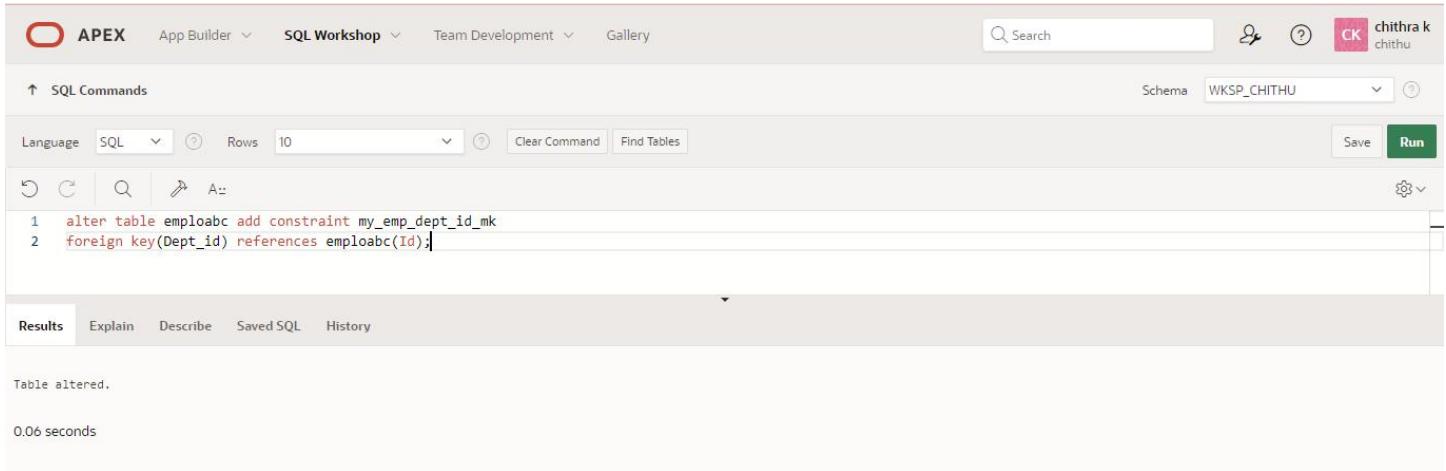
3.Add a column DEPT_ID to the EMP table. Add a foreign key reference on the EMP table that ensures that the employee is not assigned to nonexistent department. Name the constraint my_emp_dept_id_fk.

QUERY:

Alter table emploabc

Add constraint my_dept_id_mk Foreign key(dept_id) references emploabc(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 search bar, a user icon for 'chithra k chithu', and a 'Run' button. The main area is titled 'SQL Commands'. It shows the following SQL code:

```
1 alter table emploabc add constraint my_emp_dept_id_mk
2 foreign key(Dept_id) references emploabc(Id);
```

Below the code, the 'Results' tab is selected, showing the output: 'Table altered.' and '0.06 seconds'. Other tabs include Explain, Describe, Saved SQL, and History.

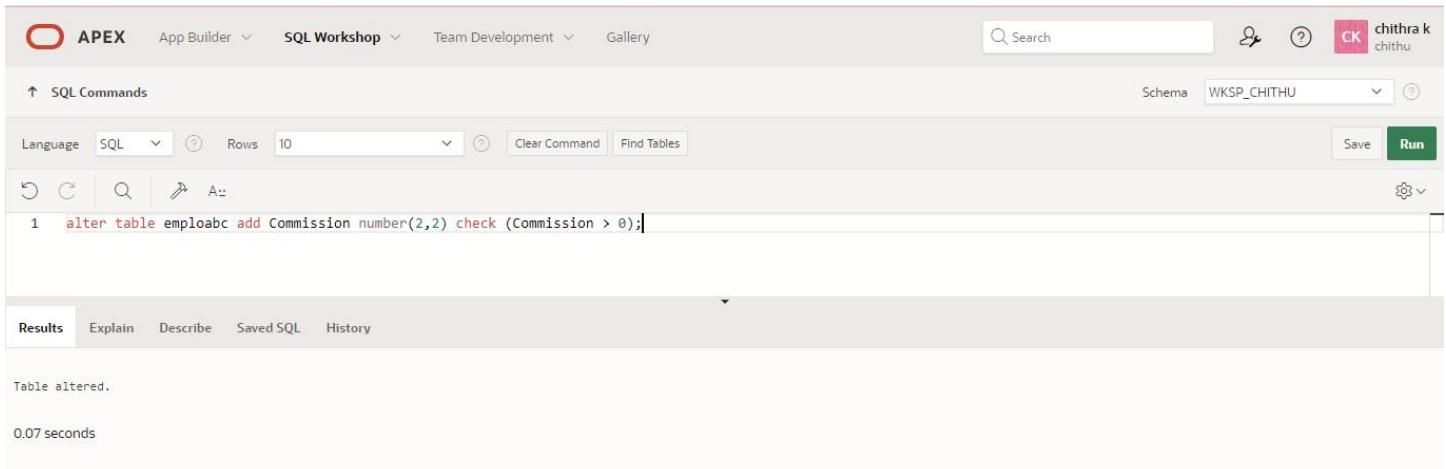
4.Modify the EMP table. Add a COMMISSION column of NUMBER data type, precision 2, scale 2. Add a constraint to the commission column that ensures that a commission value is greater than zero.

QUERY:

Alter table emploabc

Add commissions number(2,2) check (commissions >0);

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface, identical to the previous one but with a different query. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there's a search bar, a user icon for 'chithra k chithu', and a 'Run' button. The main area is titled 'SQL Commands'. It shows the following SQL code:

```
1 alter table emploabc add Commission number(2,2) check (Commission > 0);
```

Below the code, the 'Results' tab is selected, showing the output: 'Table altered.' and '0.07 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:

Thus the queries for including constraints are executed successfully.

Writing Basic SQL SELECT Statements

EX_NO:4

DATE:02/03/2024

1.The following statement executes successfully.

Identify the Errors

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

QUERY:

```
Select employee_id ,last_name ,sal*12 as annual salary from employees;
```

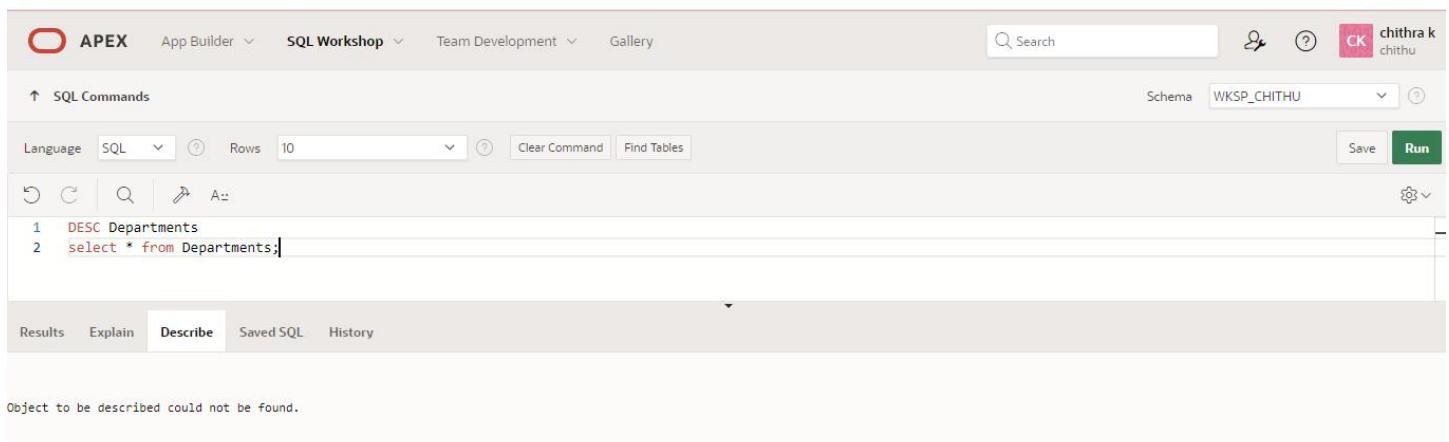
2.Show the structure of departments the table. Select all the data from it.

QUERY:

```
Desc departments
```

```
Select * from departments;
```

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 side, there's a search bar, a user icon for 'chithra k chithu', and a 'Run' button. The main workspace is titled 'SQL Commands'. It has a toolbar with icons for Undo, Redo, Find, Replace, and a dropdown menu. Below the toolbar, the schema is set to 'WKSP_CHITHU'. The SQL editor contains the following code:

```
1 DESC Departments
2 select * from Departments;
```

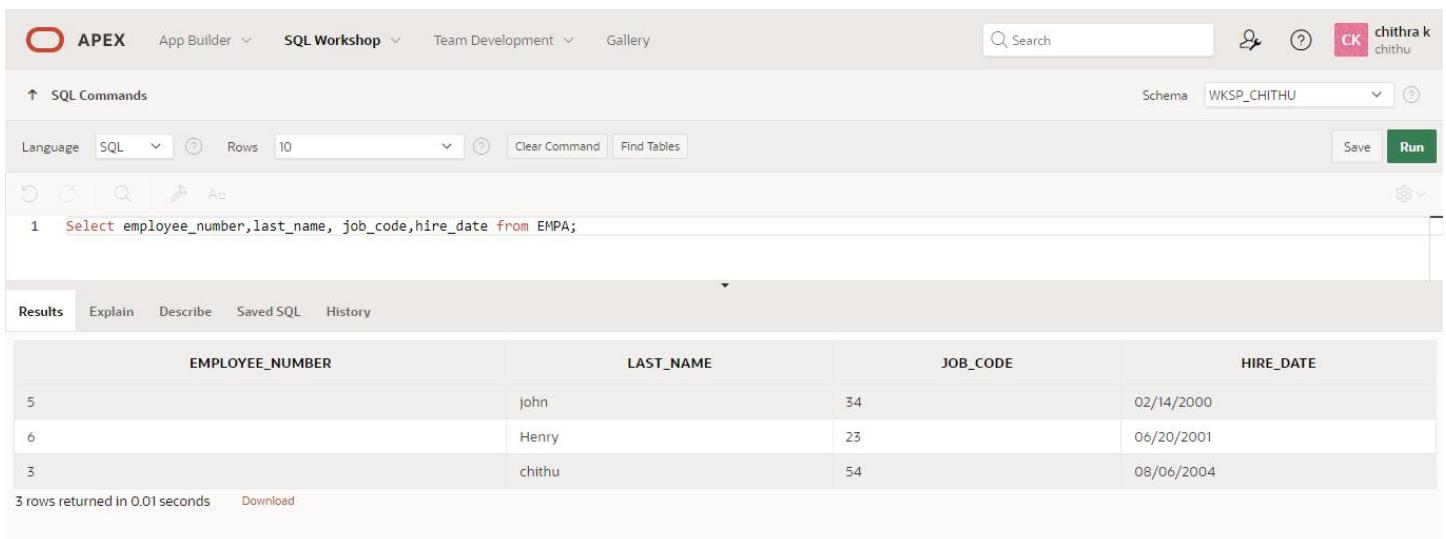
At the bottom of the editor, there are tabs for Results, Explain, Describe (which is selected), Saved SQL, and History. The results pane displays the message: "Object to be described could not be found."

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

QUERY:

Select employee_number, last_name, job_code, hire_date from employee;

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 'chithra k chithu'. The main area has a search bar and a schema dropdown set to 'WKSP_CHITHU'. Below is a toolbar with icons for SQL, rows, clear command, and find tables, followed by a 'Run' button. The SQL command entered is: `1 Select employee_number, last_name, job_code, hire_date from EMPA;`. The results tab is selected, displaying a table with four columns: EMPLOYEE_NUMBER, LAST_NAME, JOB_CODE, and HIRE_DATE. The data returned is:

EMPLOYEE_NUMBER	LAST_NAME	JOB_CODE	HIRE_DATE
5	john	34	02/14/2000
6	Henry	23	06/20/2001
3	chithu	54	08/06/2004

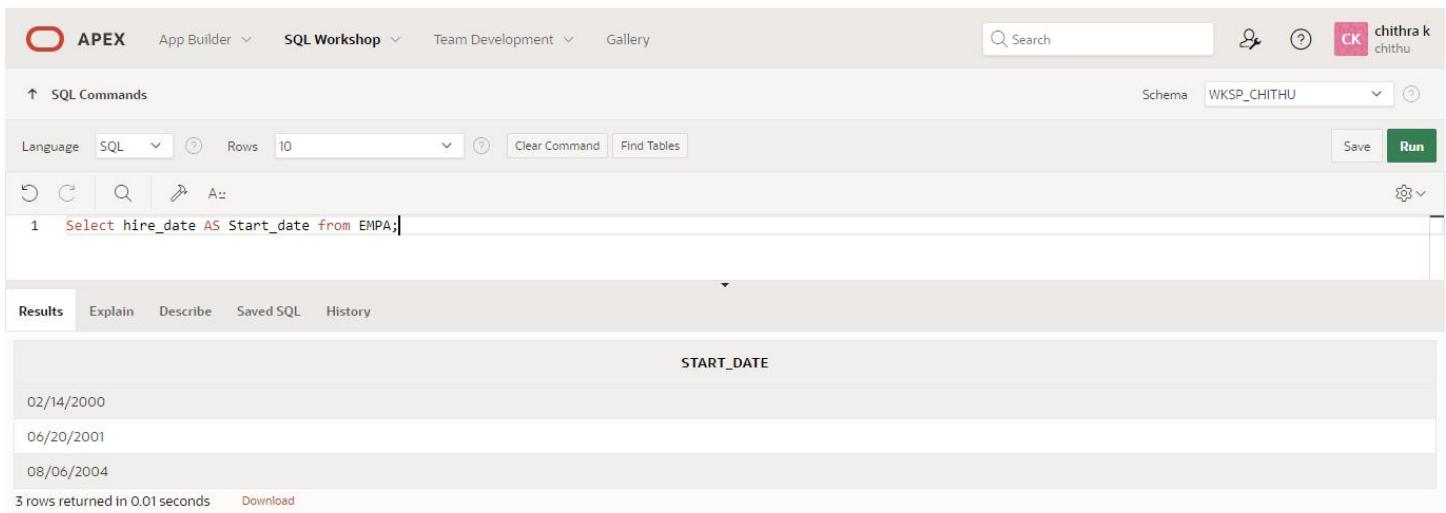
3 rows returned in 0.01 seconds [Download](#)

4.Provide an alias STARTDATE for the hire date.

QUERY:

Select hire_date as startDate from EMPA;

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface, identical to the previous one but with a different query. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows a user profile for 'chithra k chithu'. The main area has a search bar and a schema dropdown set to 'WKSP_CHITHU'. Below is a toolbar with icons for SQL, rows, clear command, and find tables, followed by a 'Run' button. The SQL command entered is: `1 Select hire_date AS Start_date from EMPA;`. The results tab is selected, displaying a table with one column: START_DATE. The data returned is:

START_DATE
02/14/2000
06/20/2001
08/06/2004

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

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

QUERY:

Select distinct job_code from EMPA;

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 'chithra k' with a pink icon. The main area has a search bar and a schema dropdown set to 'WKSP_CHITHU'. The SQL Commands panel shows the query 'Select DISTINCT job_code from EMPA;'. The Results tab displays the output:

JOB_CODE
23
34
54

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

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

QUERY:

Select lastname||','|| job_code as "employee and title" from EMPA;

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 'chithra k' with a pink icon. The main area has a search bar and a schema dropdown set to 'WKSP_CHITHU'. The SQL Commands panel shows the query 'Select lastname||','|| job_code AS "Employee and Title" From EMPA;'. The Results tab displays the output:

Employee and Title
john,34
Henry,23
chithu,54

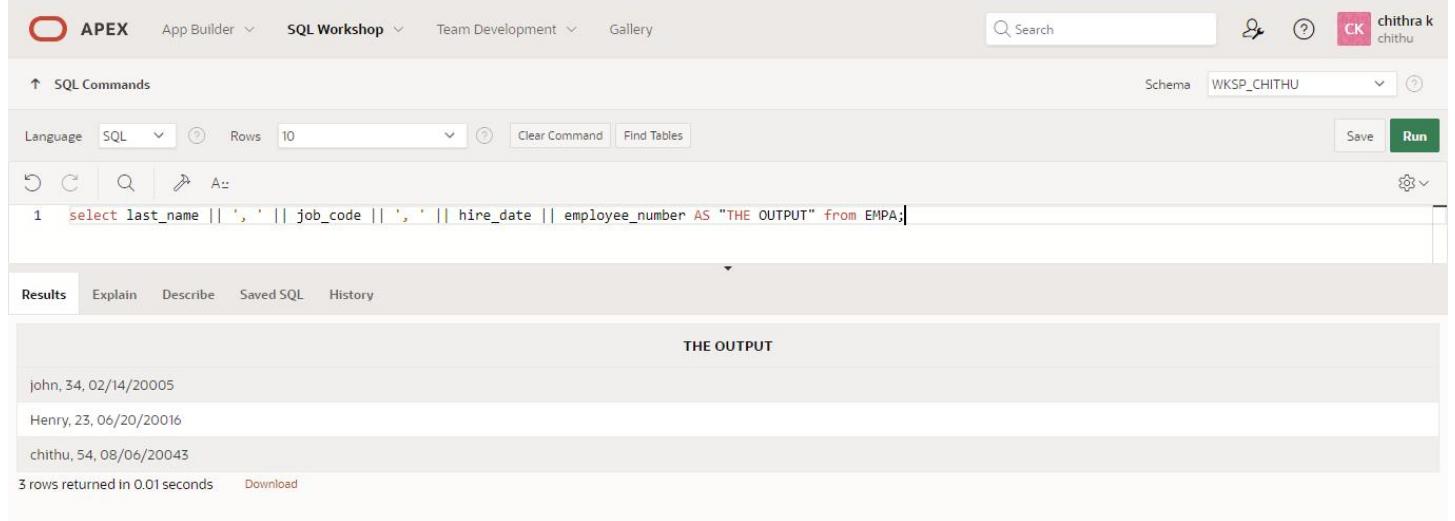
3 rows returned in 0.01 seconds [Download](#)

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 last_name ||','|| job_code||','|| employee_number ||','|| hire_date as "the output" from EMPA;

OUTPUT:



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

```
1 select last_name ||','|| job_code||','|| hire_date || employee_number AS "THE OUTPUT" from EMPA;
```

Below the command window, the results tab is selected, displaying the output:

THE OUTPUT	
john, 34, 02/14/20005	
Henry, 23, 06/20/20016	
chithu, 54, 08/06/20043	

At the bottom left, it says "3 rows returned in 0.01 seconds".

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

RESULT:

Thus the queries for basic SQL statement are executed successfully.

RESTRICTING AND SORTING DATA

EX.NO.5

DATE: 07/03/2024

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 empo2 where salary > 12000;

OUTPUT:



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

```
1 select last_name,salary from empo2 where salary >12000;
```

The results table displays two rows:

LAST_NAME	SALARY
sree	120000
mohanraj	15000

2 rows returned in 0.01 seconds

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

QUERY:

Select last_name, dept_id from empo2 where e_id=176;

OUTPUT:



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

```
1 select last_name,dept_num from empo2 where e_num=176;
```

The results table displays one row:

LAST_NAME	DEPT_NUM
mohanraj	32

1 rows returned in 0.01 seconds

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

QUERY:

SELECT last_name , dept_id from empo2

Where Salary NOT BETWEEN 5000 AND 12000

OUTPUT:

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

```
1 select last_name,salary from empo2 where salary not between 5000 and 12000;
```

The results table displays two rows:

LAST_NAME	SALARY
sree	120000
mohanraj	15000

2 rows returned in 0.000 seconds

4. Display the employee last name, job ID, and start date of employees hired between

February 20,1998 and May 1,1998.order the query in ascending order by start date.(hints: between)

QUERY:

Select last_name, Job_id, hire_date from empo2

where hire_date between 'February 20,1998' AND 'May 1,1998' order by hire_date ASC;

OUTPUT:

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

```
1 Select last_name,job_id,Start_date from empo2
2 where Start_date between '02/20/1998' AND '05/01/1998'
3 order by Start_date ASC;
```

The results table displays three rows:

LAST_NAME	JOB_ID	START_DATE
mohanraj	2	02/23/1998
sree	1	03/24/1998
mtchel	4	04/14/1998

3 rows returned in 0.000 seconds

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,Salary from empo2 where (Salary BETWEEN 5000 AND 12000) AND (dept_id IN(20,50)) order by last_name;

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The query entered is: `1 Select last_name,dept_num from empo2 where dept_num IN(20,50) order by last_name;`. The results table has two columns: LAST_NAME and DEPT_NUM. The data returned is:

LAST_NAME	DEPT_NUM
Henry	50
sree	20

2 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 empo2 where (Salary BETWEEN 5000 AND 12000) AND (dept_id IN(20,50)) order by last_name ;

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The query entered is: `1 Select last_name AS "EMPLOYEE",salary AS "MONTHLY SALARY" from empo2 where (salary between 5000 AND 12000) AND (dept_num IN(20,50)) order by last_name;`. The results table has two columns: EMPLOYEE and MONTHLY SALARY. The data returned is:

EMPLOYEE	MONTHLY SALARY
Henry	10000

1 rows returned in 0.00 seconds

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

QUERY:

Select last_name ,hire_date from empo2 where hire_date like '%1994';

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The query entered is: `1 Select last_name,Start_date from empo2 where Start_date LIKE '%1994%';`. The results table has two columns: LAST_NAME and START_DATE. One row is returned, showing Henry as the last name and 06/15/1994 as the start date.

LAST_NAME	START_DATE
Henry	06/15/1994

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

QUERY:

Select last_name ,job_title from empo2 where manager_id is NULL;

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The query entered is: `1 Select last_name, job_name from empo2 where manager_id IS NULL;`. The results table has two columns: LAST_NAME and JOB_NAME. One row is returned, showing SIEE as the last name and president as the job name.

LAST_NAME	JOB_NAME
SIEE	president

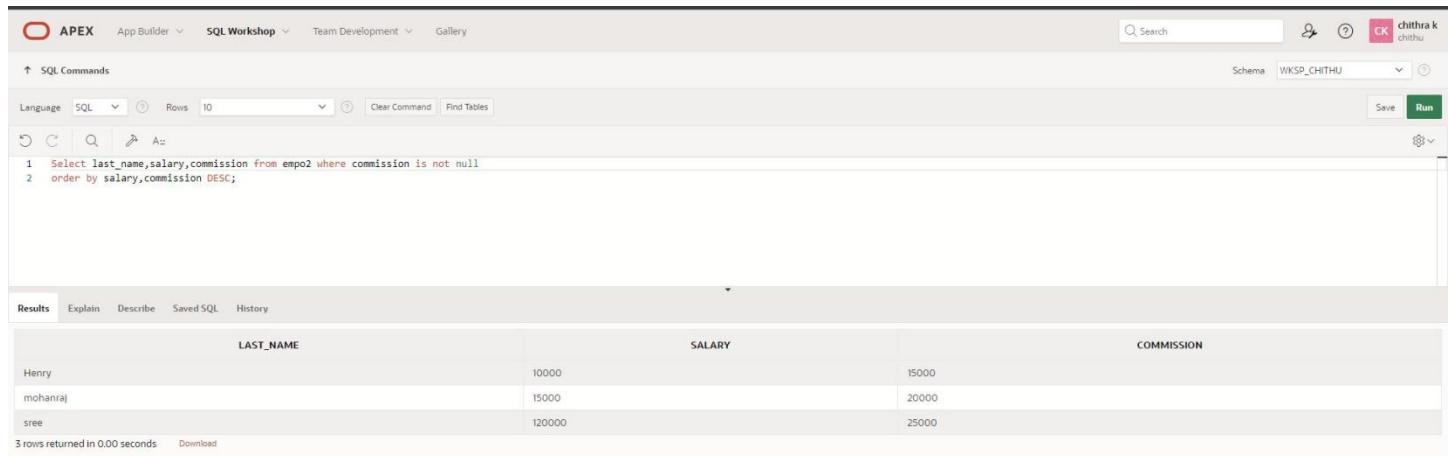
9. Display the last name, salary, and commission for all employees who earn commissions.

Sort data in descending order of salary and commissions.(hints: is not null,orderby)

QUERY:

```
Select last_name ,Salary,commission from empo2 where commission is not null  
order by Salary,commission DESC;
```

OUTPUT:



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

```
1 Select last_name,salary,commission from empo2 where commission is not null  
2 order by salary,commission DESC;
```

The results section displays the following data:

LAST_NAME	SALARY	COMMISSION
Henry	10000	15000
mohanraj	15000	20000
sree	120000	25000

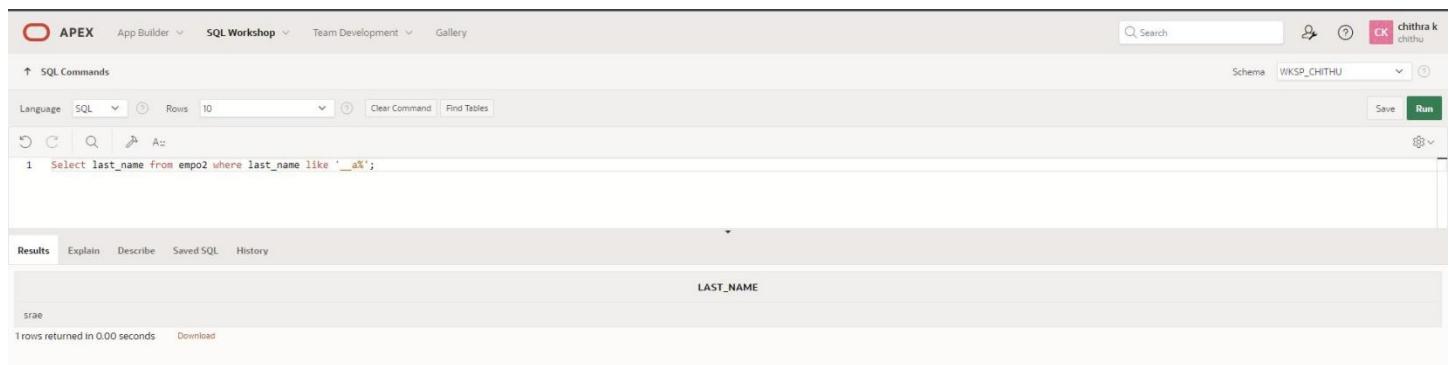
3 rows returned in 0.00 seconds

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

QUERY:

```
Select last_name from empo2 where last_name LIKE '__a%';
```

OUTPUT:



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

```
1 Select last_name from empo2 where last_name like '__a%';
```

The results section displays the following data:

LAST_NAME
SRAK

1 rows returned in 0.00 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 ,job_title ,Salary from empo2 where last_name LIKE '%a%' AND  
last_name LIKE '%e%';
```

OUTPUT:

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

```
1 Select last_name from empo2 where last_name like '%a%';  
2 Select last_name from empo2 where last_name like '%e%';
```

The results pane displays two rows:

LAST_NAME
srae
Haery

2 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,job_name,salary from empo2 where job_name IN('sales_rep','stock_clerk')  
AND salary NOT IN(2500,3500,7000) ;
```

OUTPUT:

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

```
1 SELECT last_name, job_name, salary FROM empo2 WHERE job_name IN ('sales_rep', 'stock_clerk') AND salary NOT IN (2500, 3500, 7000);
```

The results pane displays two rows:

LAST_NAME	JOB_NAME	SALARY
mohanraj	sales_rep	15000
michel	stock_clerk	8000

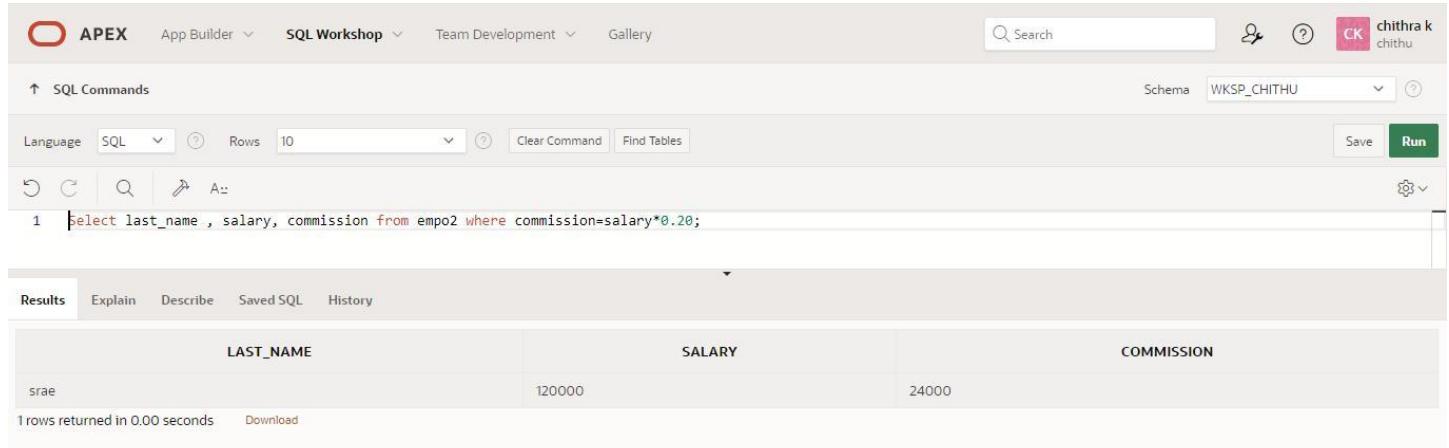
2 rows returned in 0.00 seconds

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 empo2 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, and Gallery. On the right, there's a search bar, a user icon for 'chithra k chithu', and a schema dropdown set to 'WKSP_CHITHU'. The main workspace is titled 'SQL Commands' with a language dropdown set to 'SQL'. It shows a single query line: '1 select last_name , salary, commission from empo2 where commission=salary*0.20;'. Below the query, the results tab is selected, showing a table with three columns: LAST_NAME, SALARY, and COMMISSION. One row is returned, showing 'srae' in the LAST_NAME column, '120000' in the SALARY column, and '24000' in the COMMISSION column. A note at the bottom says '1 rows returned in 0.00 seconds'.

Evaluation Procedure	Marks awarded
Practice Evaluation(5)	
Viva	
Total(10)	
Faculty Signature	

RESULT:

Thus the queries for restricting and sorting data are executed successfully.

SINGLE ROW FUNCTIONS

EX.NO.6

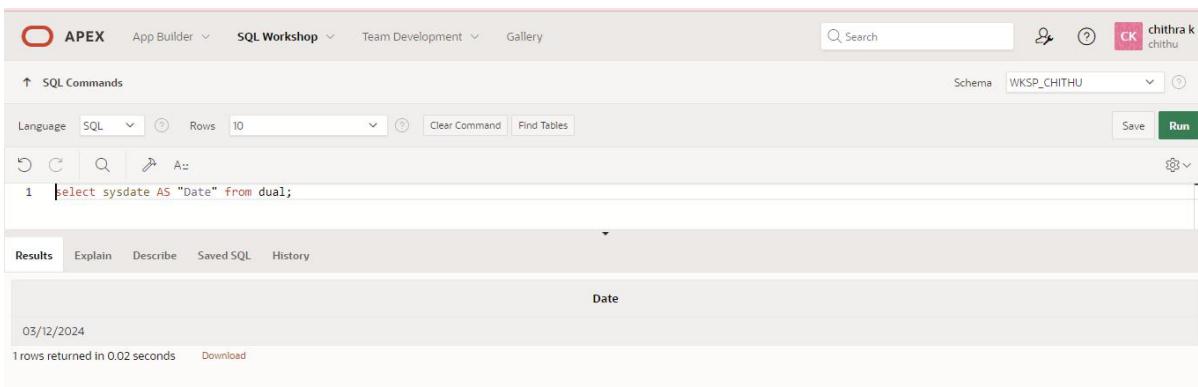
DATE:09/03/2024

Find the Solution for the following:

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

SELECT SYSDATE AS "DATE" FROM DUAL;

OUTPUT:



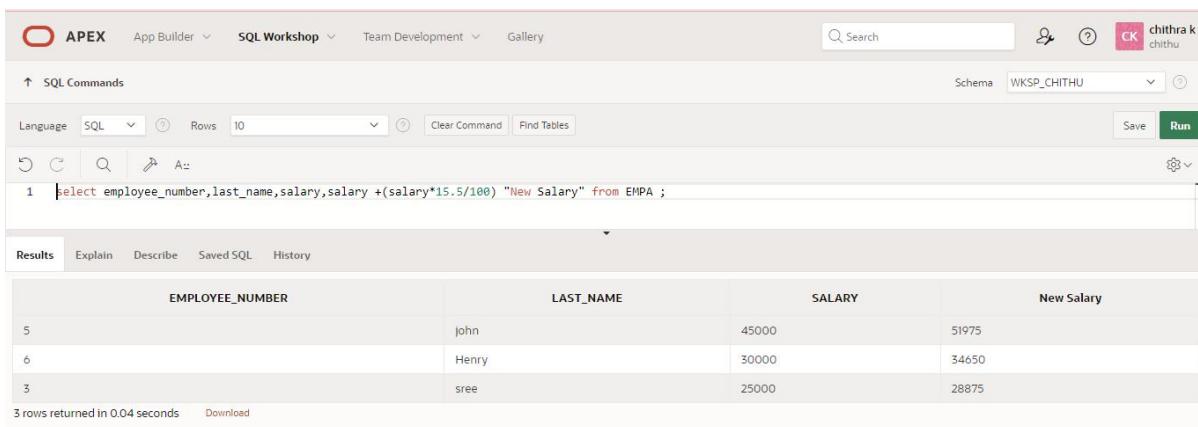
The screenshot shows the Oracle SQL Workshop interface. In the SQL Commands pane, the query `select sysdate AS "Date" from dual;` is entered. The Results pane displays the output: a single row with one column named "Date" containing the value `03/12/2024`. The status bar at the bottom indicates "1 rows returned in 0.02 seconds".

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 emplo_num, last_name, Salary, Salary+(15.5/100*Salary) "NEW_SALARY"
From EMPA;**

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. In the SQL Commands pane, the query `select employee_number, last_name, salary, salary +(salary*15.5/100) "New Salary" from EMPA ;` is entered. The Results pane displays the output: three rows with columns `EMPLOYEE_NUMBER`, `LAST_NAME`, `SALARY`, and `New Salary`. The data is as follows:

EMPLOYEE_NUMBER	LAST_NAME	SALARY	New Salary
5	john	45000	51975
6	Henry	30000	34650
3	sree	25000	28875

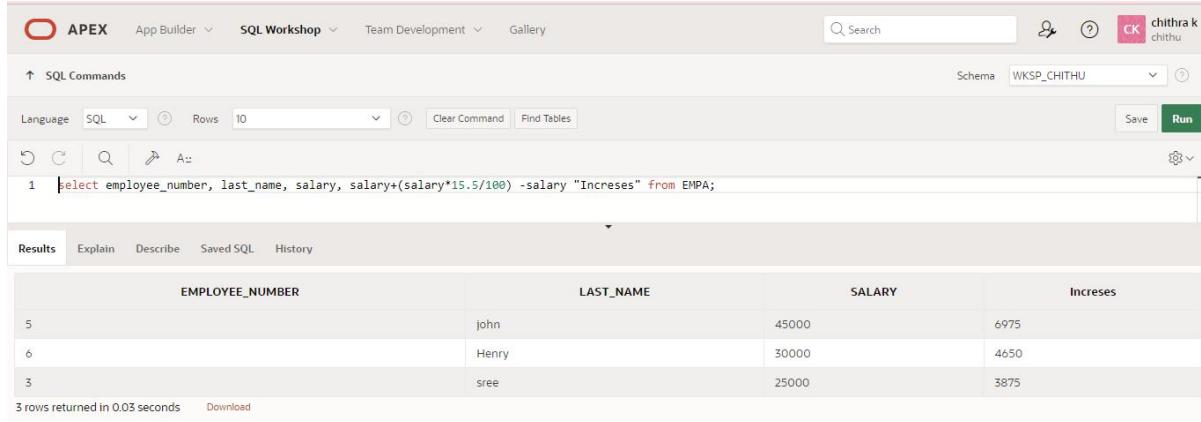
The status bar at the bottom indicates "3 rows returned in 0.04 seconds".

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 EMPA;
```

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 'chithra k'. The main area has tabs for SQL Commands, Results, Explain, Describe, Saved SQL, and History. The SQL Commands tab contains the following query:

```
1 select employee_number, last_name, salary, salary+(salary*15.5/100) -salary "Increases" from EMPA;
```

The Results tab displays the output in a table:

EMPLOYEE_NUMBER	LAST_NAME	SALARY	Increases
5	john	45000	6975
6	Henry	30000	4650
3	sree	25000	3875

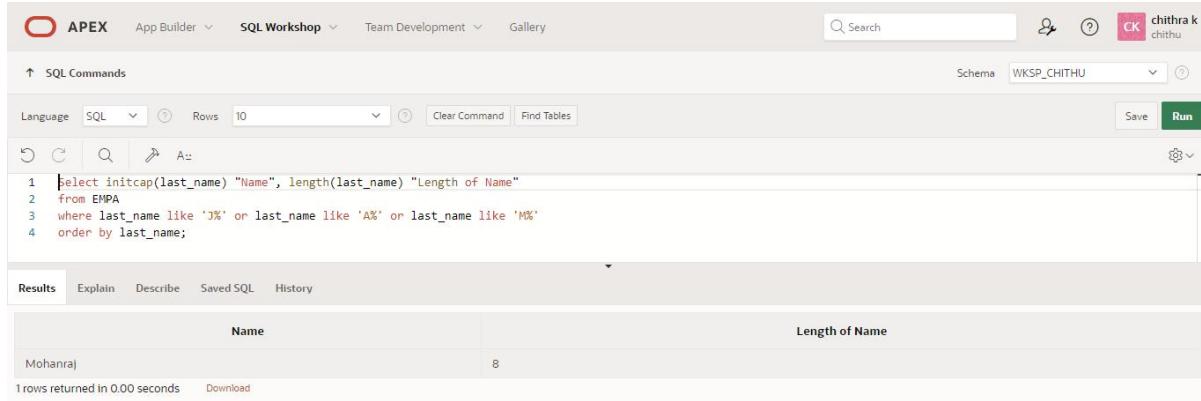
Below the table, it says '3 rows returned in 0.03 seconds' and there is a 'Download' link.

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 EMPA 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 top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows a user profile for 'chithra k'. The main area has tabs for SQL Commands, Results, Explain, Describe, Saved SQL, and History. The SQL Commands tab contains the following query:

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

The Results tab displays the output in a table:

Name	Length of Name
Mohanraj	8

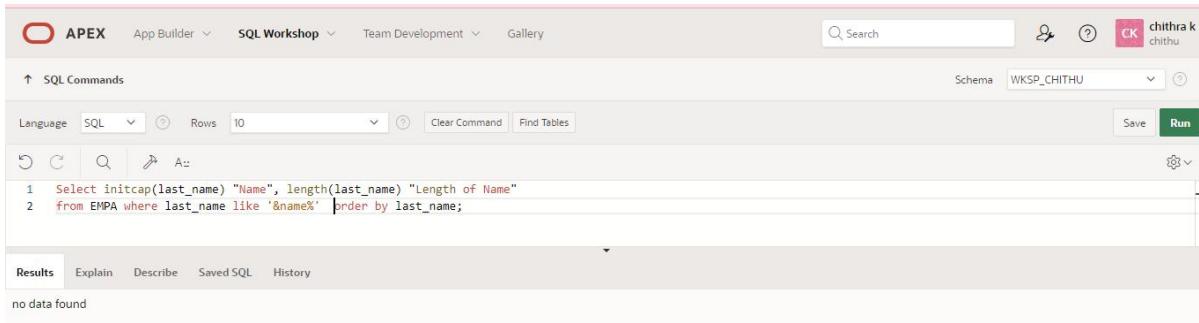
Below the table, it says '1 rows returned in 0.00 seconds' and there is a 'Download' link.

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 EMPA  
where last_name like '&name%'  
order by last_name;
```

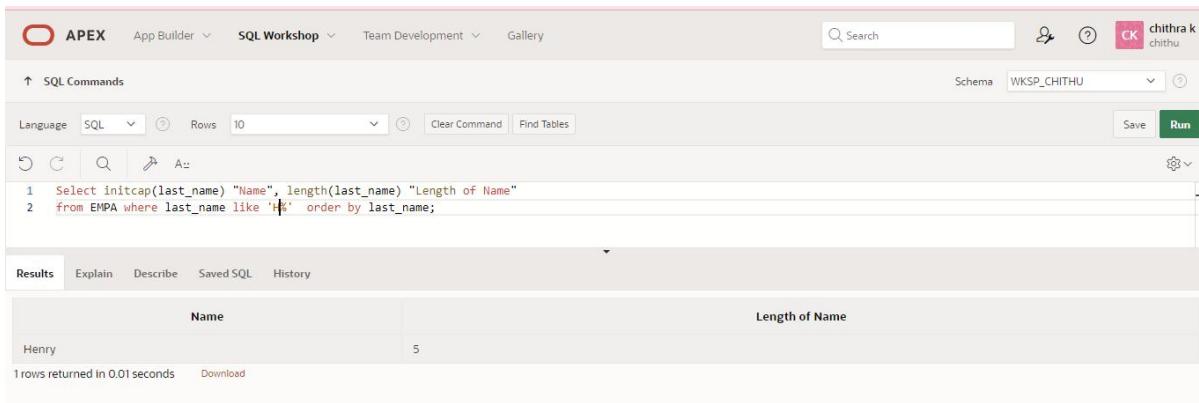
OUTPUT:



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 'chithra k chithu' are on the right. The main area has tabs for 'SQL Commands', 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History'. Under 'SQL Commands', the schema is set to 'WKSP_CHITHU'. The query entered is:

```
1 Select initcap(last_name) "Name", length(last_name) "Length of Name"  
2 from EMPA where last_name like '&name%' |order by last_name;
```

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



This screenshot shows the same SQL Workshop interface after running the query. The results table displays one row:

Name	Length of Name
Henry	5

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

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 EMPA order by 2;
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, and Gallery. The right side shows a user profile for 'chithra k' with a CK icon. The main area has tabs for SQL Commands, Results, Explain, Describe, Saved SQL, and History. The SQL Commands tab is active, showing the following command:

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

The Results tab displays the output:

LAST_NAME	MONTHS_WORKED
Mohanraj	109
Henry	240
sree	290

3 rows returned in 0.00 seconds. There is a 'Download' link at the bottom.

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 EMPA;
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface, similar to the previous one. The top navigation bar and user profile are identical. The SQL Commands tab is active, showing the following command:

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

The Results tab displays the output:

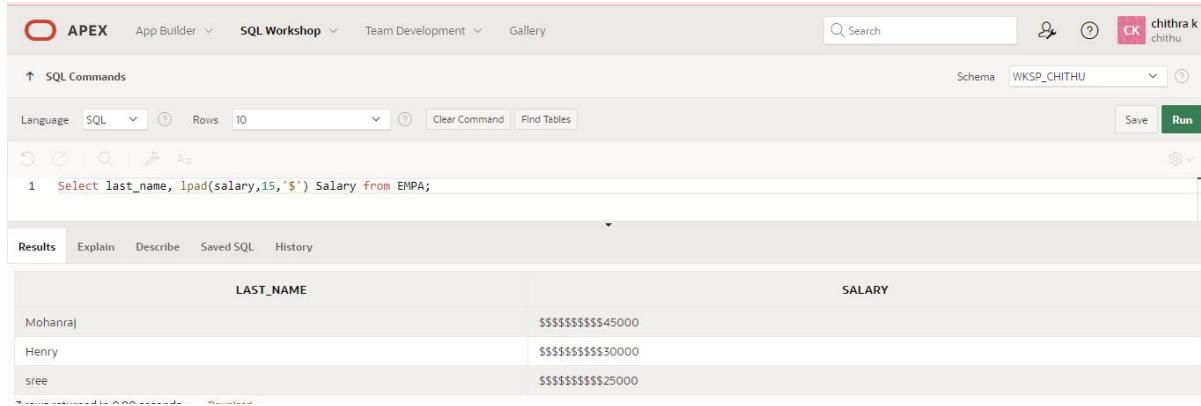
Dream Salary
Mohanraj earns \$45000 monthly but wants \$135000
Henry earns \$30000 monthly but wants \$90000
sree earns \$25000 monthly but wants \$75000

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 EMPA;
```

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, and Gallery. The right side shows a user profile for 'chithra k' (chithu). The main area is titled 'SQL Commands' with a search bar and a schema dropdown set to 'WKSP_CHITHU'. Below the search bar are buttons for Save and Run. The SQL command entered is: 'Select last_name, lpad(salary,15,'\$') Salary from EMPA;'. The results tab is selected, displaying the output:

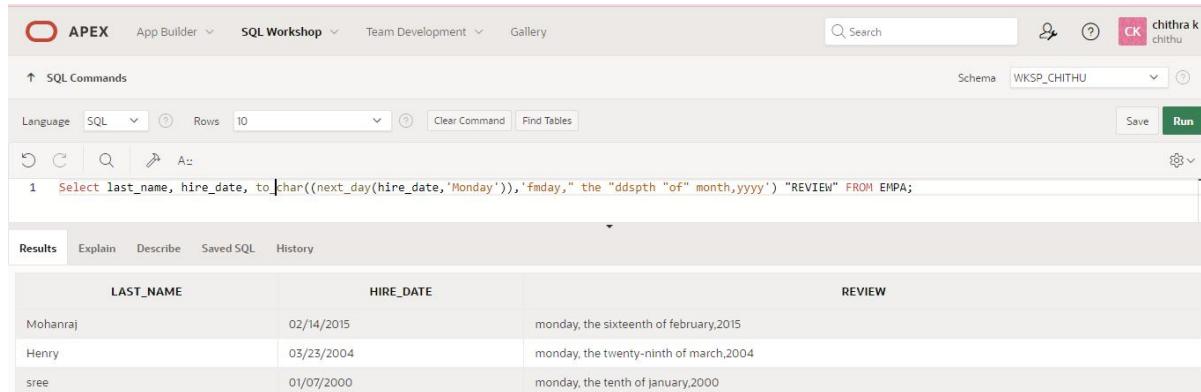
LAST_NAME	SALARY
Mohanraj	\$\$\$\$\$\$\$\$\$\$45000
Henry	\$\$\$\$\$\$\$\$\$\$30000
sree	\$\$\$\$\$\$\$\$\$\$25000

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 EMPA;
```

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, and Gallery. The right side shows a user profile for 'chithra k' (chithu). The main area is titled 'SQL Commands' with a search bar and a schema dropdown set to 'WKSP_CHITHU'. Below the search bar are buttons for Save and Run. The SQL command entered is: 'Select last_name, hire_date, to_char((next_day(hire_date,'Monday')),'fmday," the "ddspth "of"
month,yyyy') "REVIEW" from EMPA;'. The results tab is selected, displaying the output:

LAST_NAME	HIRE_DATE	REVIEW
Mohanraj	02/14/2015	monday, the sixteenth of february,2015
Henry	03/23/2004	monday, the twenty-ninth of march,2004
sree	01/07/2000	monday, the tenth of january,2000

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

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The query is executed successfully, returning three rows of data:

LAST_NAME	HIRE_DATE	Day
Henry	03/23/2004	Tuesday
sree	01/07/2000	Friday
Mohanraj	02/14/2015	Saturday

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

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

RESULT:

Thus the queries for single row function are executed successfully.

DISPLAYING DATA FROM MULTIPLE TABLES

EX_NO:7

DATE:14/03/2024

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_id from empa e,dept d where e.department_id=d.dept_id;
```

OUTPUT:

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

```
1 Select e.last_name,e.department_id,d.department_id from empa e,dept d where e.department_id=d.department_id;
```

The results section displays the following data:

LAST_NAME	DEPARTMENT_ID	DEPARTMENT_ID
john	20	20
Mohanraj	50	50
Henry	30	30
sree	10	10

4 rows returned in 0.00 seconds

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

QUERY:

```
select distinct job_id,loc_id from empa e,dept d where e.department_id=d.dept_id and  
e.department_id=80;
```

OUTPUT:

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

```
1 select distinct job_id,loc_id from empa e,dept d where e.department_id=d.department_id and  
2 e.department_id=80;
```

The results section displays the following data:

JOB_ID	LOC_ID
st_clerk	1800

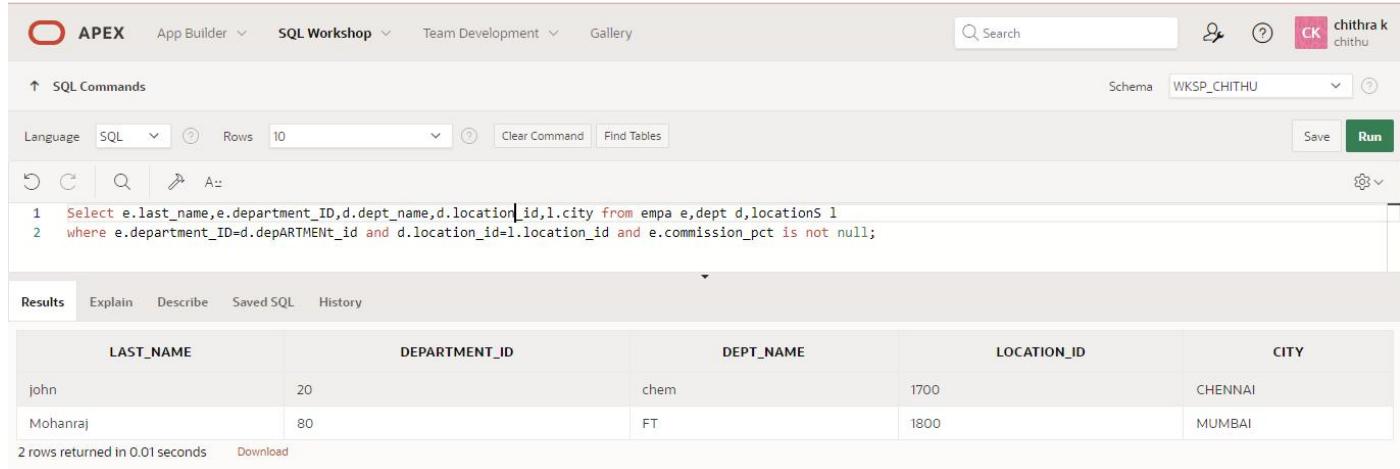
1 rows returned in 0.01 seconds

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

QUERY:

```
Select e.last_name,e.department_id,d.dept_name,d.loc_id,l.city from emp a,e,dept d,locations l where  
e.department_id=d.dept_id and d.loc_id=l.location_id and e.commission_pct is not null;
```

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, and Gallery. On the right, there's a search bar, a user icon for 'chithra k chithu', and a 'Run' button. The main area has tabs for SQL Commands, Results, Explain, Describe, Saved SQL, and History. The SQL Commands tab contains the following SQL code:

```
1 Select e.last_name,e.department_ID,d.dept_name,d.location_id,l.city from emp a,e,dept d,locations l  
2 where e.department_ID=d.dePARTMENT_id and d.location_id=l.location_id and e.commission_pct is not null;
```

The Results tab displays the output in a grid format:

LAST_NAME	DEPARTMENT_ID	DEPT_NAME	LOCATION_ID	CITY
john	20	chem	1700	CHENNAI
Mohanraj	80	FT	1800	MUMBAI

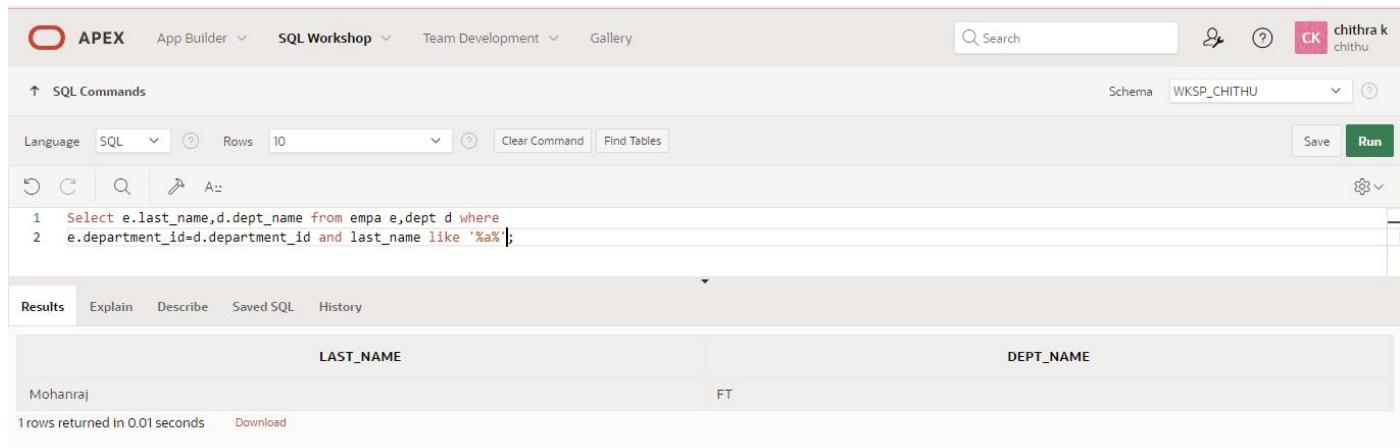
Below the grid, it says "2 rows returned in 0.01 seconds" and has a "Download" link.

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

QUERY:

```
Select emp021.last_name,dept23.dept_name from emp a,dept d where emp.a.department_id=dept.d.department_id and  
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, and Gallery. On the right, there's a search bar, a user icon for 'chithra k chithu', and a 'Run' button. The main area has tabs for SQL Commands, Results, Explain, Describe, Saved SQL, and History. The SQL Commands tab contains the following SQL code:

```
1 Select e.last_name,d.dept_name from emp a,e,dept d where  
2 e.department_id=d.department_id and last_name like '%a%';
```

The Results tab displays the output in a grid format:

LAST_NAME	DEPT_NAME
Mohanraj	FT

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

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

QUERY:

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

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 'chithra k chithu'. The main area is titled 'SQL Commands' with a schema dropdown set to 'WKSP_CHITHU'. The code editor contains the following SQL query:

```
1 Select e.last_name,e.department_id,e.job_id,d.dept_name from empa e join dept d  
2 on(e.department_id=d.department_id) join locations on (d.location_id=locations.location_id) | where  
3 lower(locations.city)='toronto';
```

Below the code editor, the 'Results' tab is selected. The output table has columns: LAST_NAME, DEPARTMENT_ID, JOB_ID, and DEPT_NAME. One row is displayed:

LAST_NAME	DEPARTMENT_ID	JOB_ID	DEPT_NAME
Mohanraj	80	80	IT

At the bottom left, it says '1 rows returned in 0.03 seconds'. There are 'Download' and 'Run' buttons at the bottom right.

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

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

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. On the right, there's a user profile for 'chithra k chithu'. The main area is titled 'SQL Commands' with a schema dropdown set to 'WKSP_DHANATB12'. The code editor contains the following SQL query:

```
1 Select w.last_name "Employee",w.emp_id "emp#",m.last_name "manager" ,m.emp_id "Mgr#" from emp90 w JOIN emp90 m on (w.manager_id=m.emp_id);
```

Below the code editor, the 'Results' tab is selected. The output table has columns: LAST_NAME, DEPT_NAME. One row is displayed:

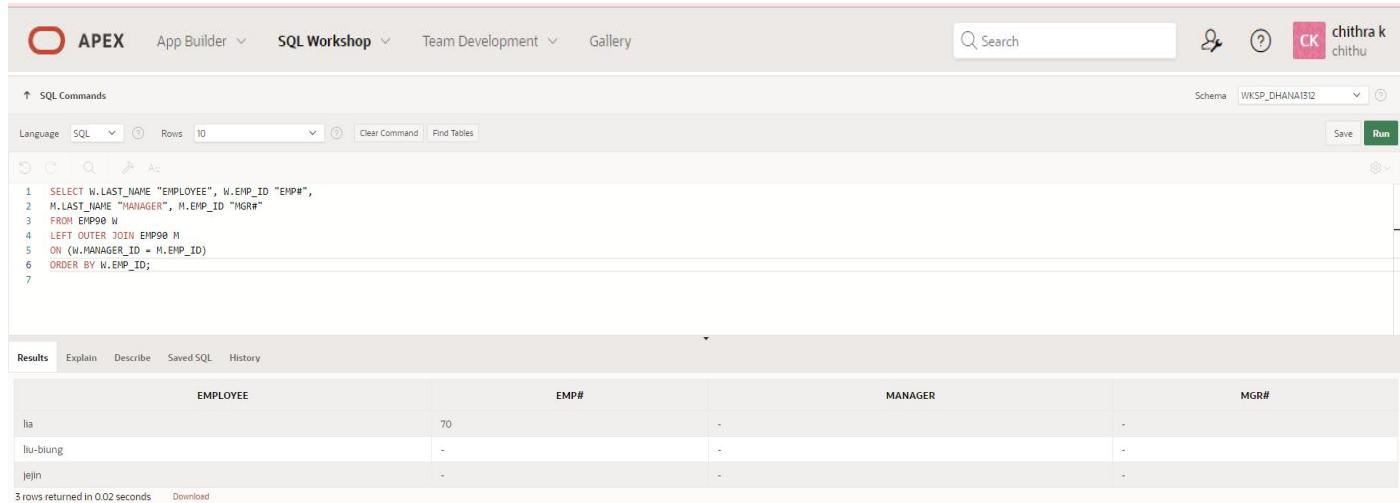
LAST_NAME	DEPT_NAME
Ila	IT

At the bottom left, it says '1 rows returned in 0.01 seconds'. There are 'Download' and 'Run' buttons at the bottom right.

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

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

OUTPUT:



The screenshot shows the Oracle APEX SQL Workshop interface. The query window contains the following SQL code:

```
1 SELECT W.LAST_NAME "EMPLOYEE", W.EMP_ID "EMP#",  
2 M.LAST_NAME "MANAGER", M.EMP_ID "MGR#"  
3 FROM EMP90 N  
4 LEFT OUTER JOIN EMP90 M  
5 ON (W.MANAGER_ID = M.EMP_ID)  
6 ORDER BY W.EMP_ID;  
7
```

The results section displays the following data:

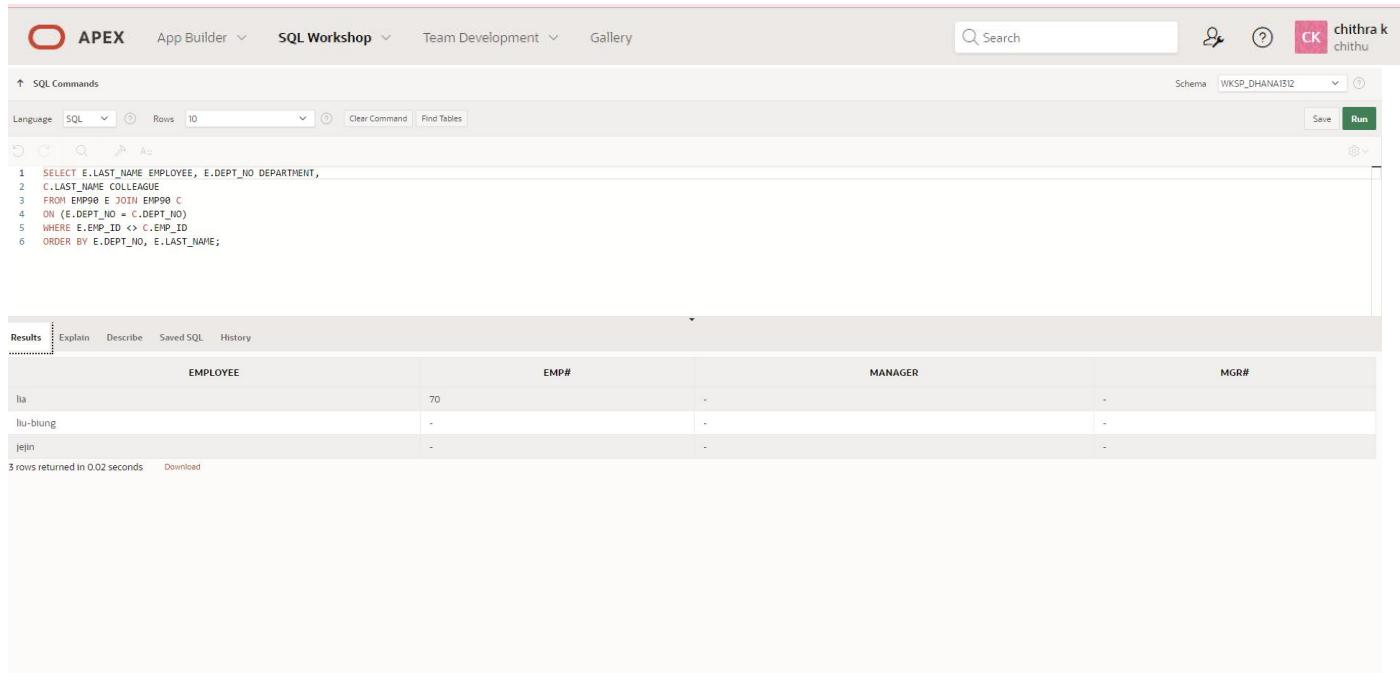
EMPLOYEE	EMP#	MANAGER	MGR#
lia	70	-	-
liu-biung	-	-	-
jejin	-	-	-

3 rows returned in 0.02 seconds

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

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

OUTPUT:



The screenshot shows the Oracle APEX SQL Workshop interface. The query window contains the following SQL code:

```
1 SELECT E.LAST_NAME EMPLOYEE, E.DEPT_NO DEPARTMENT,  
2 C.LAST_NAME COLLEAGUE  
3 FROM EMP90 E JOIN EMP90 C  
4 ON (E.DEPT_NO = C.DEPT_NO)  
5 WHERE E.EMP_ID <> C.EMP_ID  
6 ORDER BY E.DEPT_NO, E.LAST_NAME;
```

The results section displays the following data:

EMPLOYEE	DEPARTMENT	COLLEAGUE
lia	70	-
liu-biung	-	-
jejin	-	-

3 rows returned in 0.02 seconds

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 emp a JOIN dept d
ON (e.dept_id = d.dept_id)
JOIN job_grade j
ON (e.salary BETWEEN j.lowest_sal AND j.highest_sal);
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. The top right shows a user profile for 'chithra k chithu'. 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 shows three rows of employee data:

EMPLOYEE	EMP#	MANAGER	MGR#
Ila	70	-	-
Hu-bung	-	-	-
Jolin	-	-	-

Below the table, it says '3 rows returned in 0.02 seconds' and 'Download'.

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 emp a, emp Davies
WHERE Davies.last_name = 'Davies'
AND Davies.hire_date < e.hire_date;
```

OUTPUT:

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. The top right shows a user profile for 'chithra k chithu'. 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 shows one row of employee data:

LAST_NAME	DEPT_NAME
Ila	IT

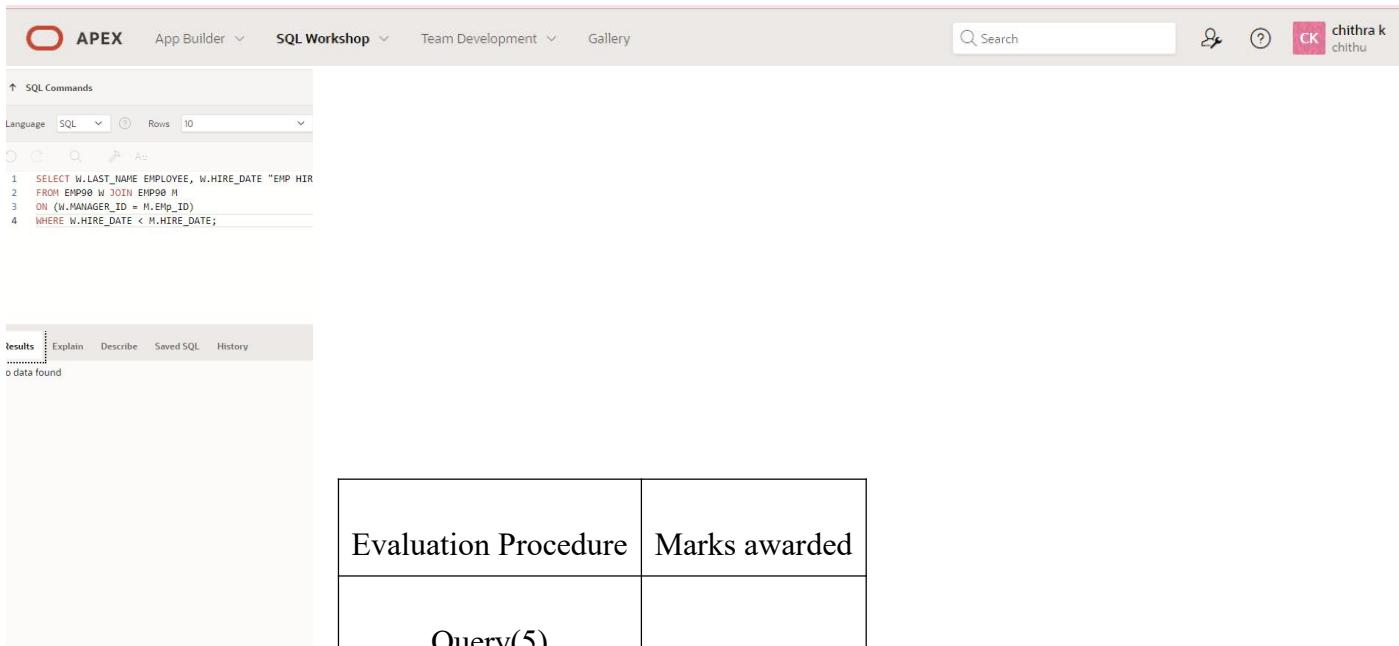
Below the table, it says '1 rows returned in 0.04 seconds' and 'Download'.

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

QUERY:

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

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. At the top, there are tabs for App Builder, SQL Workshop (which is selected), Team Development, and Gallery. On the right, there is a search bar, a refresh icon, and a user profile for 'chithra k chithu'. Below the tabs, there is a toolbar with icons for back, forward, search, and refresh. A dropdown menu shows 'SQL Commands' is selected. Underneath, there is a language dropdown set to 'SQL', a rows dropdown set to '10', and a toolbar with icons for copy, paste, and search. The main area contains the following SQL code:

```
1 SELECT W.LAST_NAME EMPLOYEE, W.HIRE_DATE "EMP HIR  
2 FROM EMP90 W JOIN EMP90 M  
3 ON (W.MANAGER_ID = M.Emp_ID)  
4 WHERE W.HIRE_DATE < M.HIRE_DATE;
```

Below the code, there is a results panel with tabs for 'Results' (selected), 'Explain', 'Describe', 'Saved SQL', and 'History'. The 'Results' tab shows the message '0 data found'.

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

RESULT:

Thus the queries for displaying datas from multiple tables are executed successfully.

AGGREGATING DATA USING GROUP FUNCTIONS

EX_NO : 8

DATE: 24/03/2024

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

TRUE

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

FALSE

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

FALSE

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

QUERY:

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

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 'chithra k chithu' and a search bar. The main workspace is titled 'SQL Commands'. It shows a code editor with the following SQL query:

```
1 | SELECT ROUND(MAX(salary),0) "Maximum",ROUND(MIN(salary),0) "Minimum",ROUND(SUM(salary),0) "Sum",ROUND(AVG(salary),0) "Average"FROM EMPA;
```

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

Maximum	Minimum	Sum	Average
45000	25000	100000	33333

At the bottom left, it says '1 rows returned in 0.03 seconds'. There are also 'Download' and 'Run' buttons.

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

QUERY:

```
select job_id ,Round(MAX(salary),0) "MAXIMUM",Round (Min(salary),0)"Minimum",Round  
(SUM(Salary),0)"sum" ,Round (AVg (salary),0)"average" from EMPA 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 search bar, user profile 'chithra k chithu', and a 'Run' button. The main area has tabs for 'SQL Commands' and 'Results'. Under 'SQL Commands', the query is displayed:

```
1 | SELECT job_id, ROUND(MAX(salary),0) "Maximum",ROUND(MIN(salary),0) "Minimum",ROUND(SUM(salary),0) "Sum",ROUND(AVG(salary),0) "Average"FROM EMPA GROUP BY job_id;
```

Under 'Results', the output is a table:

JOB_ID	Maximum	Minimum	Sum	Average
44	30000	30000	30000	30000
46	45000	45000	45000	45000
47	25000	25000	25000	25000

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

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

QUERY:

```
select job_id, count(*) from EMPA group by job_id ;  
select job_id, count(*) from EMPA where job_id='47' 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 search bar, user profile 'chithra k chithu', and a 'Run' button. The main area has tabs for 'SQL Commands' and 'Results'. Under 'SQL Commands', two queries are listed:

```
1 | SELECT job_id, COUNT(*)FROM EMPA GROUP BY job_id;  
2 |
```

Under 'Results', the output is a table:

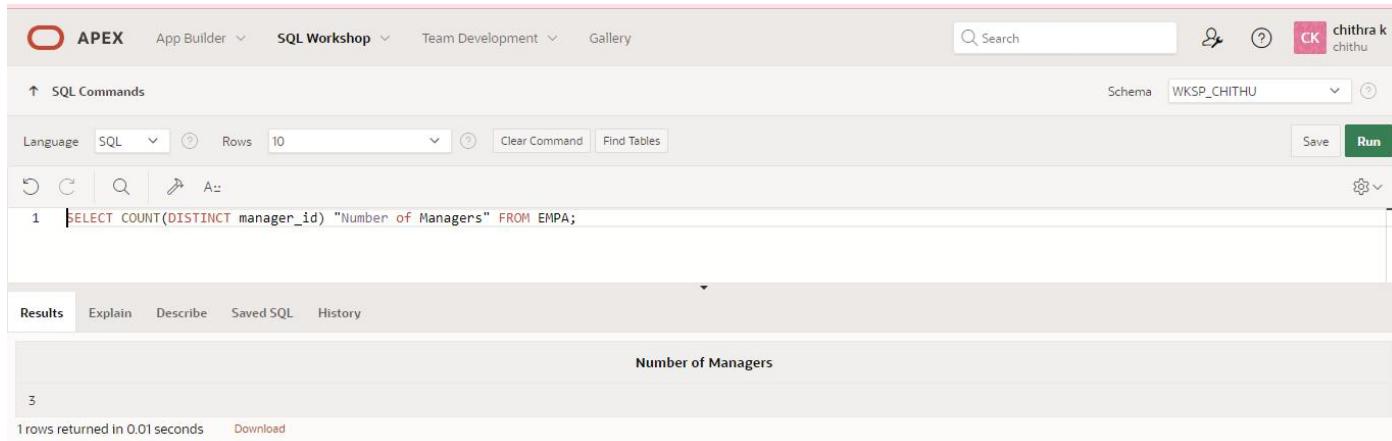
JOB_ID	COUNT(*)
44	1
46	1
47	1

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 empa;
```

OUTPUT:



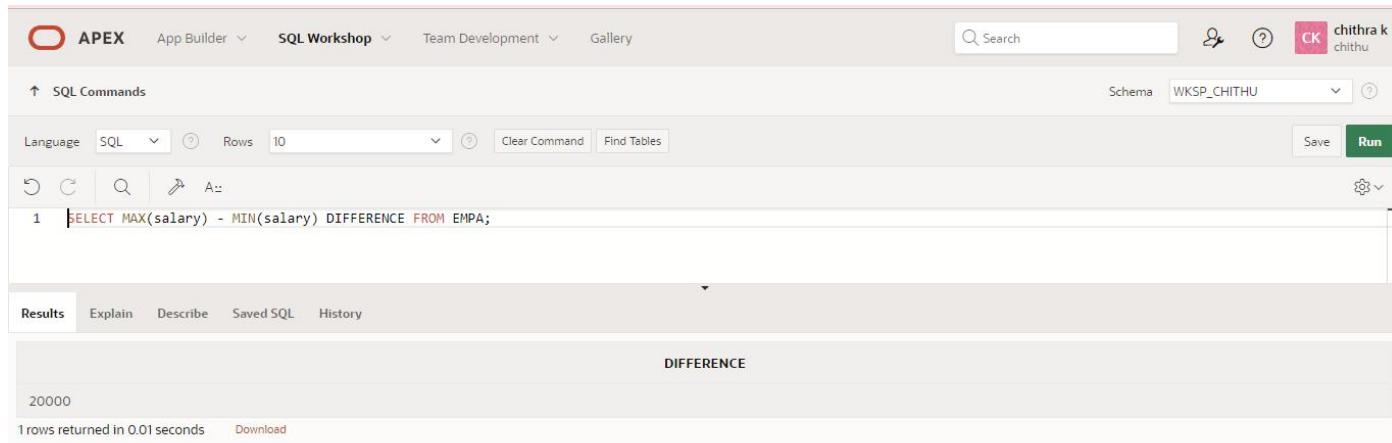
The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, and Gallery. On the right, there's a search bar, user profile 'chithra k chithu', and a schema dropdown set to 'WKSP_CHITHU'. The main workspace has tabs for SQL Commands, Results, Explain, Describe, Saved SQL, and History. The SQL tab is active, showing the query: 'SELECT COUNT(DISTINCT manager_id) "Number of Managers" FROM EMPA;'. The Results tab displays the output: 'Number of Managers' with a value of 3. Below the results, it says '1 rows returned in 0.01 seconds' and has a 'Download' link.

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

QUERY:

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

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface, identical to the previous one but with a different query. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, and Gallery. On the right, there's a search bar, user profile 'chithra k chithu', and a schema dropdown set to 'WKSP_CHITHU'. The main workspace has tabs for SQL Commands, Results, Explain, Describe, Saved SQL, and History. The SQL tab is active, showing the query: 'SELECT MAX(salary) - MIN(salary) DIFFERENCE FROM EMPA;'. The Results tab displays the output: 'DIFFERENCE' with a value of 20000. Below the results, it says '1 rows returned in 0.01 seconds' and has a 'Download' link.

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

QUERY:

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

OUTPUT:

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

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

The Results tab is selected, displaying the output:

MANAGER_ID	MIN(SALARY)
5	45000
6	30000
3	25000

3 rows returned in 0.01 seconds

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

QUERY:

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

OUTPUT:

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

```
1 | SELECT COUNT(*) total,SUM(DECODE(TO_CHAR(hire_date, 'YYYY'),1995,1,0))"1995",SUM(DECODE(TO_CHAR(hire_date, 'YYYY'),1996,1,0))"1996",
2 | SUM(DECODE(TO_CHAR(hire_date, 'YYYY'),1997,1,0))"1997",SUM(DECODE(TO_CHAR(hire_date, 'YYYY'),1998,1,0))"1998" FROM EMPA;
```

The Results tab is selected, displaying the output:

TOTAL	1995	1996	1997	1998
3	1	0	1	0

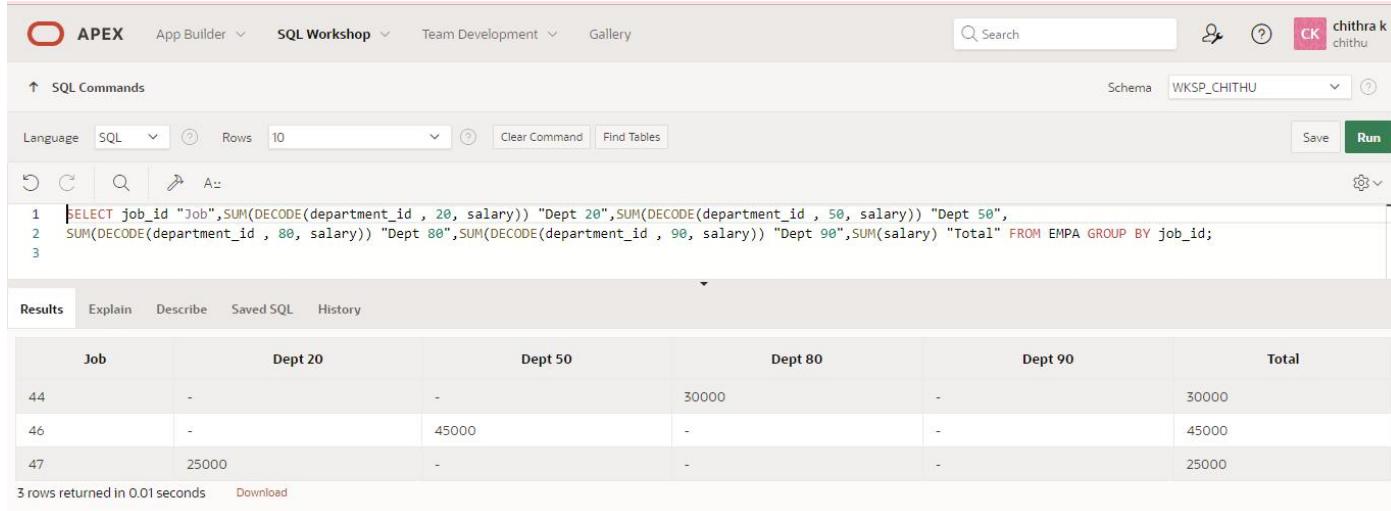
1 rows returned in 0.01 seconds

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

QUERY:

```
select job_id "job", sum(decode(dept_id,20,salary))"Dept20",sum (decode(dept_id ,50, salary)) "dept50",sum (decode(dept_id ,80, salary)) "dept80",sum (decode(dept_id ,90, salary)) "dept90",sum(salary) "TOTAL" from empa 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 icon for 'chithra k' and a search bar. The main area has tabs for 'SQL Commands' and 'Results'. Under 'SQL Commands', the schema is set to 'WKSP_CHITHU'. The code entered is:

```
1 SELECT job_id "Job",SUM(DECODE(department_id , 20, salary)) "Dept 20",SUM(DECODE(department_id , 50, salary)) "Dept 50",
2 SUM(DECODE(department_id , 80, salary)) "Dept 80",SUM(DECODE(department_id , 90, salary)) "Dept 90",SUM(salary) "Total" FROM EMPA GROUP BY job_id;
3
```

The 'Results' tab displays the output in a grid format:

Job	Dept 20	Dept 50	Dept 80	Dept 90	Total
44	-	-	30000	-	30000
46	-	45000	-	-	45000
47	25000	-	-	-	25000

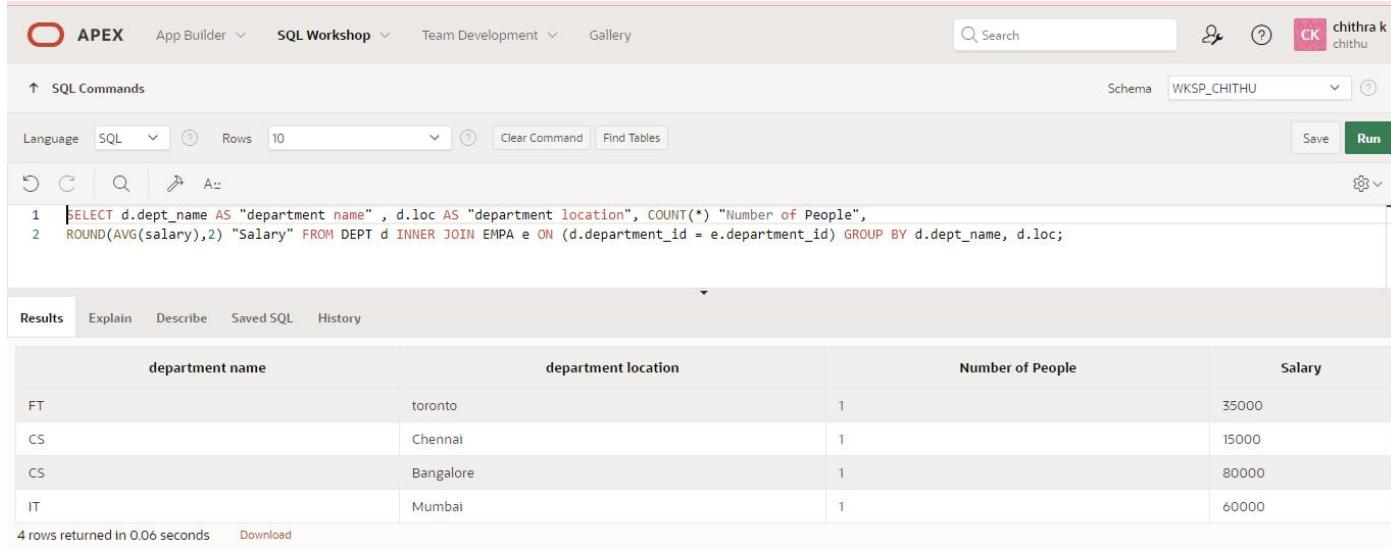
Below the table, it says '3 rows returned in 0.01 seconds' and has a 'Download' link.

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

QUERY:

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

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 icon for 'chithra k' and a search bar. The main area has tabs for 'SQL Commands' and 'Results'. Under 'SQL Commands', the schema is set to 'WKSP_CHITHU'. The code entered is:

```
1 SELECT d.dept_name AS "department name" , d.loc AS "department location", COUNT(*) "Number of People",
2 ROUND(AVG(salary),2) "Salary" FROM DEPT d INNER JOIN EMPA e ON (d.department_id = e.department_id) GROUP BY d.dept_name, d.loc;
```

The 'Results' tab displays the output in a grid format:

department name	department location	Number of People	Salary
FT	toronto	1	35000
CS	Chennai	1	15000
CS	Bangalore	1	80000
IT	Mumbai	1	60000

Below the table, it says '4 rows returned in 0.06 seconds' and has a 'Download' link.

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

RESULT:

Thus the queries for aggregating data using group functions are executed successfully.

SUB-QUERIES

EX.NO:9

DATE:06/04/2024

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 dept_id =
(select dept_id from employees where last_name like 'Zlotkey')and last_name <>
'Zlotkey';
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. In the SQL Commands pane, the following code is entered:

```
1 select last_name, hire_date
2   from EMP01
3  where dept_id =
4    (select dept_id from EMP01 where last_name like 'Zlotkey')and last_name<>
5      'Zlotkey';
```

In the Results pane, the output is displayed as a table:

LAST_NAME	HIRE_DATE
King	02/03/1998

Below the table, it says "1 rows returned in 0.00 seconds".

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

QUERY:

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

OUTPUT:

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

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

The results section displays the following data:

EMPLOYEE_ID	LAST_NAME	SALARY
9	king	100000
2	sree	120000
5	bhuvana	150000

3 rows returned in 0.01 seconds

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

QUERY:

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

OUTPUT:

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

```
1 select employee_id, last_name  
2   from empol  
3  where dept_id in (select dept_id from employees where last_name like '%u%');  
4
```

The results section displays the following data:

EMPLOYEE_ID	LAST_NAME
5	bhuvana

1 rows returned in 0.01 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, dept_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 SQL command is:

```
1 select last_name, dept_id, job_id  
2 from emp01  
3 where dept_id in (select dept_id from dept where location_id =1700);  
4
```

The results table has columns LAST_NAME, DEPT_ID, and JOB_ID. One row is shown:

LAST_NAME	DEPT_ID	JOB_ID
Sue	25	34

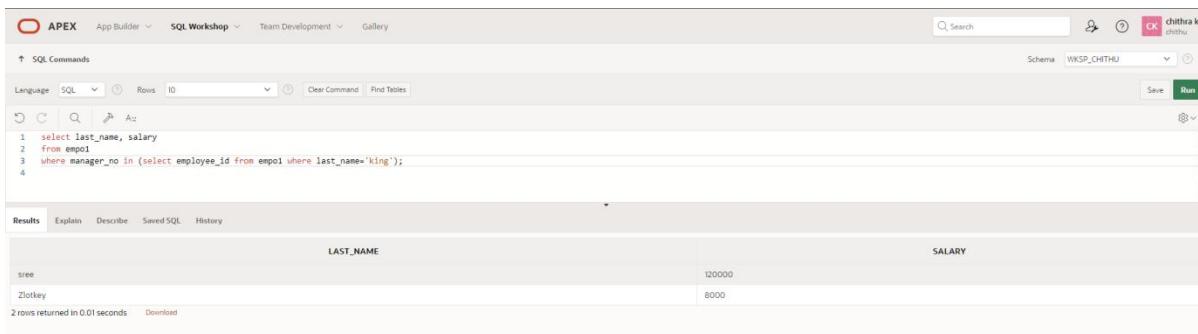
Rows returned in 0.02 seconds

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:



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

```
1 select last_name, salary  
2 from emp01  
3 where manager_no in (select employee_id from employees where last_name='King');  
4
```

The results table has columns LAST_NAME and SALARY. Two rows are shown:

LAST_NAME	SALARY
Zlotkey	120000
Sue	8000

2 rows returned in 0.01 seconds

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:

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

```
1 select dept_id, last_name, job_id  
2 from emp01  
3 where dept_id in (select dept_id from department where dept_name = 'Executive');
```

The results table shows the following data:

DEPT_ID	LAST_NAME	JOB_ID
25	sree	54

1 rows returned in 0.01 seconds

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 SQL command entered is:

```
1 select employee_id, last_name, salary  
2 from emp01  
3 where salary > (select avg(salary) from emp01) and  
4 dept_id in (select dept_id from employees where last_name like '%u%');
```

The results table shows the following data:

EMPLOYEE_ID	LAST_NAME	SALARY
2	bhuvaneshwari	120000
5	bhuvana	150000

2 rows returned in 0.02 seconds

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

RESULT:

Thus the sub-queries are executed successfully.

USING THE SET OPERATORS

EX.NO:10

DATE:16/04/2024

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 dept_id FROM department MINUS  
SELECT dept_id FROM employees WHERE job_id = 'st_clerk';
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The SQL command window contains the following code:

```
1 SELECT department_id  
2   FROM dept  
3  MINUS  
4 SELECT department_id  
5   FROM emp  
6  WHERE job_id = 'st_clerk';
```

The results pane shows the output:

DEPARTMENT_ID
25
34

2 rows returned in 0.02 seconds

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

QUERY:

```
SELECT country_id,country_name FROM countries MINUS  
SELECT l.country_id,c.country_name FROM locations l, countries c WHERE l.country_id = c.country_id;
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The SQL command window contains the following code:

```
1 SELECT country_id,country_name  
2   FROM countries  
3  MINUS  
4 SELECT l.country_id,c.country_name  
5   FROM locations l, countries c  
6  WHERE l.country_id = c.country_id;
```

The results pane shows the output:

COUNTRY_ID	COUNTRY_NAME
173	TORONTO

1 rows returned in 0.00 seconds

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 job_no, dept_id  
FROM employees  
WHERE dept_id = 10
```

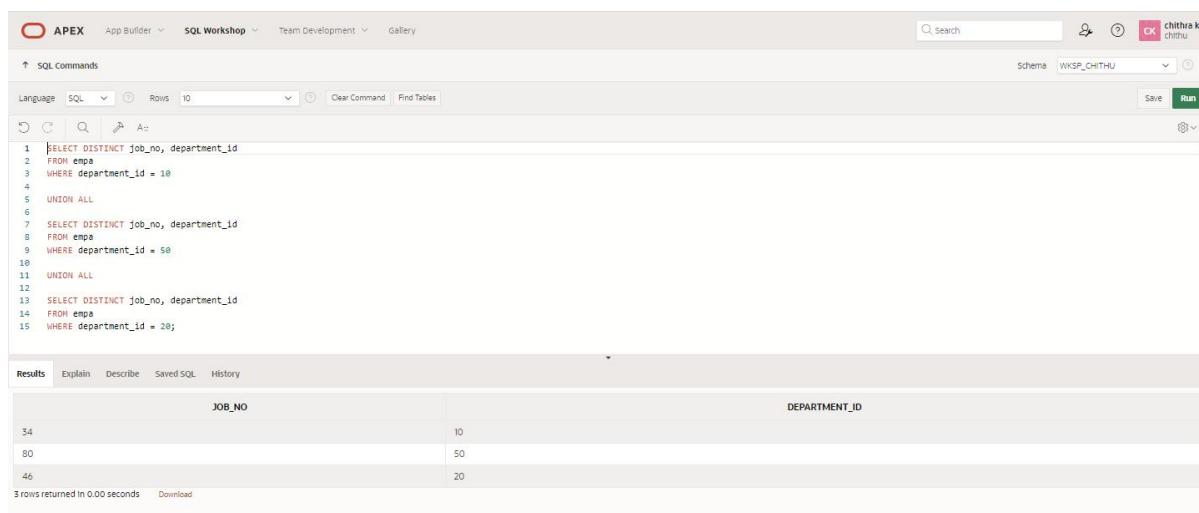
UNION ALL

```
SELECT DISTINCT job_no, dept_id  
FROM employees  
WHERE dept_id = 50
```

UNION ALL

```
SELECT DISTINCT job_no, dept_id  
FROM employees  
WHERE dept_id = 20;
```

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The SQL command window contains the following code:

```
1 SELECT DISTINCT job_no, department_id  
2 FROM emp  
3 WHERE department_id = 10  
4 UNION ALL  
5  
6 SELECT DISTINCT job_no, department_id  
7 FROM emp  
8 WHERE department_id = 50  
9 UNION ALL  
10  
11 SELECT DISTINCT job_no, department_id  
12 FROM emp  
13 WHERE department_id = 20;
```

The results section displays the output:

JOB_NO	DEPARTMENT_ID
54	10
80	50
46	20

3 rows returned in 0.00 seconds

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 employee_id,job_no FROM employees INTERSECT  
SELECT employee_id,job_no FROM job_history;
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The SQL Commands tab contains the following code:

```
1 SELECT employee_id, job_no  
2 FROM emp  
INTERSECT  
3 SELECT employee_id, job_no  
4 FROM job_history;  
5  
6  
7
```

The Results tab displays the output:

EMPLOYEE_ID	JOB_NO
24	46
26	80

2 rows returned in 0.02 seconds. A Download button is also present.

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 last_name,dept_id,TO_CHAR(null) FROM employees UNION  
SELECT TO_CHAR(null),dept_id,dept_name FROM department;
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The SQL Commands tab contains the following code:

```
1 SELECT last_name,department_id,TO_CHAR(null)  
2 FROM emp  
3 UNION  
4 SELECT TO_CHAR(null),department_id,dept_name  
5 FROM dept;
```

The Results tab displays the output:

LAST_NAME	DEPARTMENT_ID	TO_CHAR(NULL)
Henry	30	-
Mohanal	50	-
John	20	-
Sree	10	-
-	25	IT
-	34	CS
-	46	executive
-	80	FT

8 rows returned in 0.01 seconds. A Download button is also present.

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

RESULT:

Thus the queries for set operations are executed successfully.

CREATING VIEWS

EX.NO.11

DATE:20/04/2024

Find the Solution for the following:

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

QUERY:

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

OUTPUT:



A screenshot of 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 icon for 'chithra k chithu' and a search bar. The main area shows a SQL command line with the following code:

```
1 create view EMPLOYEE_VU AS select employee_id,employee_names AS EMPLOYEE,department_id from employees;
```

The results tab shows the output of the command:

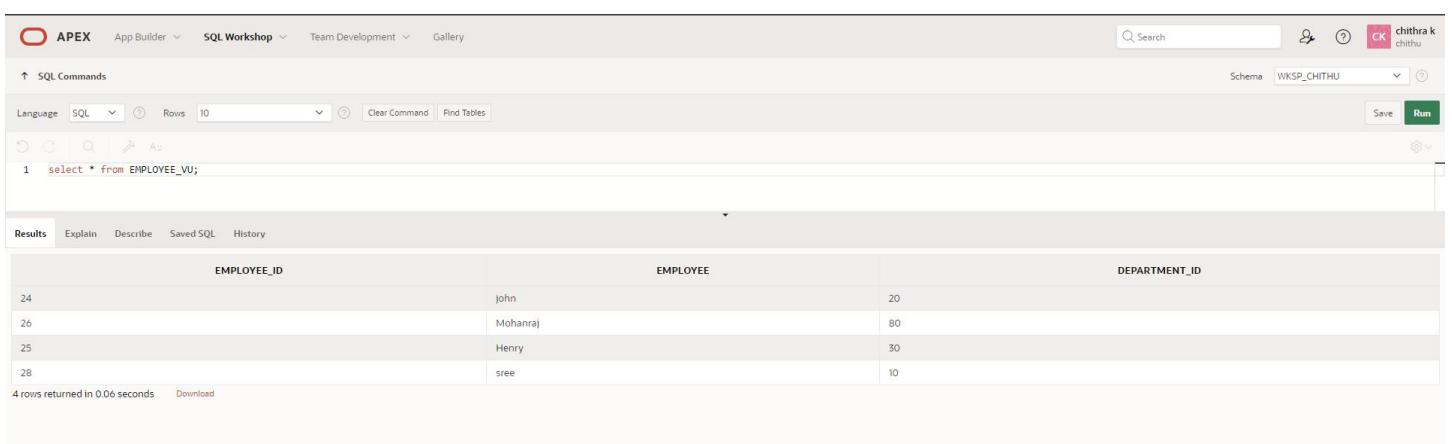
```
View created.  
0.04 seconds
```

2. Display the contents of the EMPLOYEES_VU view.

QUERY:

```
select * from employees_vu;
```

OUTPUT:



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

```
1 select * from EMPLOYEE_VU;
```

The results tab shows the output of the command, which is a table with three columns: EMPLOYEE_ID, EMPLOYEE, and DEPARTMENT_ID. The data is as follows:

EMPLOYEE_ID	EMPLOYEE	DEPARTMENT_ID
24	john	20
26	Mohanraj	80
25	Henry	30
28	sree	10

At the bottom, it says '4 rows returned in 0.06 seconds' and has 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:



EMPLOYEE	DEPARTMENT_ID
john	20
Mohanraj	80
Henry	30
sree	10

4 rows returned in 0.01 seconds Download

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

QUERY:

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

OUTPUT:



```
CREATE VIEW DEPART50 AS  
  SELECT employee_id EMPNO, employee_names EMPLOYEE,  
         department_id DEPTNO  
    FROM employees  
   WHERE department_id = 50  
     WITH CHECK OPTION CONSTRAINT emp_dept_50;
```

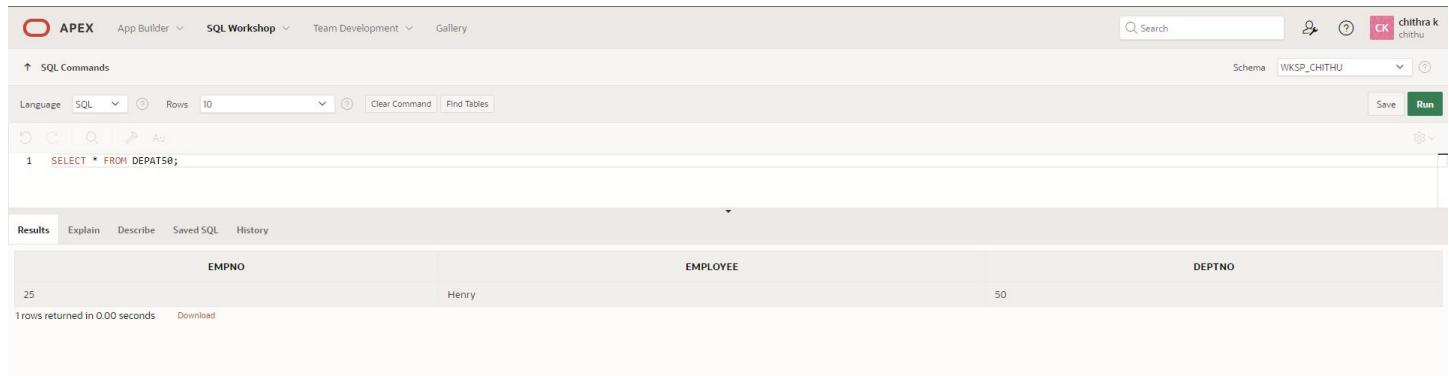
View created.
0.05 seconds

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

QUERY:

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

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. In the top navigation bar, 'APEX' is selected. Below it, 'SQL Workshop' is active. The main area shows the SQL command: '1 SELECT * FROM DEPAT50;'. The results section displays one row from the view:

EMPNO	EMPLOYEE	DEPTNO
25	Henry	50

1 rows returned in 0.00 seconds

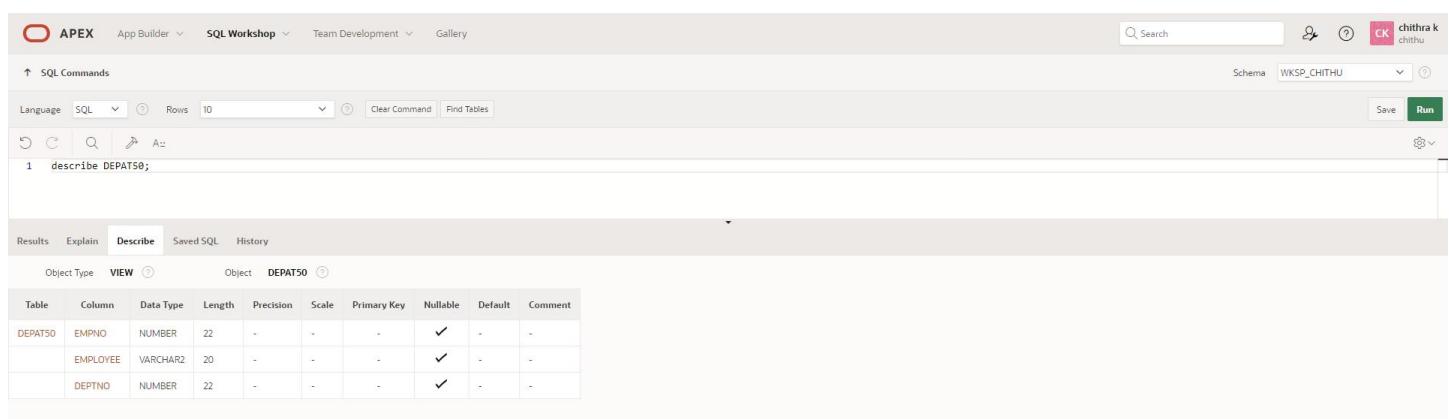
6. Display the structure and contents of the DEPT50 view.

QUERY:

```
Describe dept50;
```

```
SELECT * from dept50;
```

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface with 'Describe' selected in the tabs. The command '1 describe DEPAT50;' is shown in the SQL editor. The results section shows the structure of the view:

Object Type	VIEW	Object:	DEPAT50						
Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
DEPAT50	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';
```

OUTPUT:

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

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

In the Results tab, the output shows:

```
ORA-01402: view WITH CHECK OPTION where-clause violation
```

Execution time: 0.04 seconds.

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

QUERY:

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

OUTPUT:

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

```
1 CREATE OR REPLACE VIEW SALARY_VU AS
2   SELECT e.employee_names "Employees",
3         d.dept_name "Department",
4         e.salary "Salary",
5         j.grade_level "Grade"
6   FROM employees e,
7        dept d,
8        job_grade j
9  WHERE e.department_id = d.department_id
10    AND e.salary BETWEEN j.lowest_salary AND j.highest_salary;
```

In the Results tab, the output shows:

```
View created.

0.06 seconds
```

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

RESULT:

Thus the queries for creating views are executed successfully.

EXERCISE 12

PRACTICE QUESTIONS

Ex.no:12

DATE:04/05/2024

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	

RESULT:

Thus the queries for intro to constraints are executed successfully.

Creating views

EX.no:13

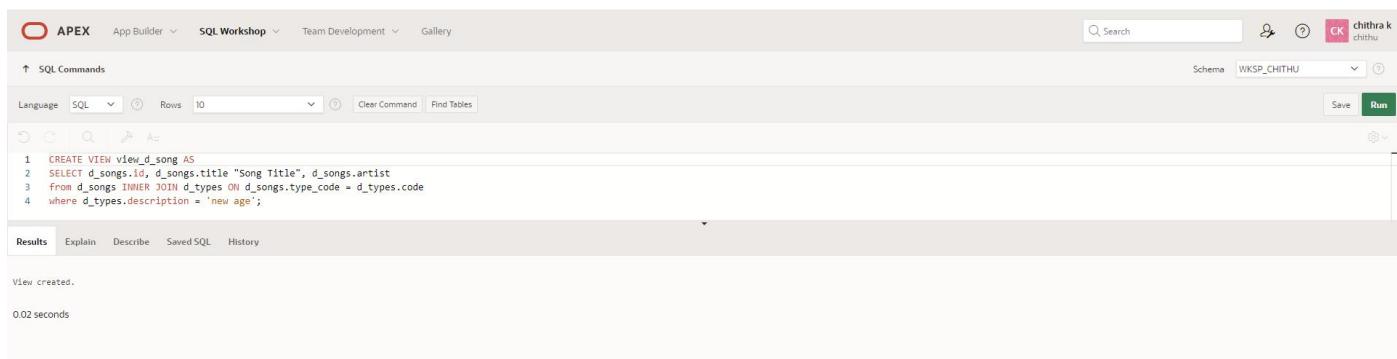
DATE:12/5/2024

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';
```



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery'. On the right, there is a user profile for 'chithra k chithu'. The main area is titled 'SQL Commands' with a search bar and a schema dropdown set to 'WKSP_CHITHU'. Below the search bar are buttons for 'Save' and 'Run'. The SQL command for creating the 'view_d_songs' is pasted into the editor. The command is:

```
1 CREATE VIEW view_d_song AS
2 SELECT d_songs.id, d_songs.title "Song Title", d_songs.artist
3 from d_songs INNER JOIN d_types ON d_songs.type_code = d_types.code
4 where d_types.description = 'new age';
```

The results tab shows the message 'View created.' and a execution time of '0.02 seconds'.

3.SELECT * FROM view_d_songs. What was returned?



The screenshot shows the results of the query 'SELECT * FROM view_d_songs'. The results tab is active, showing two rows of data. The table has three columns: 'ID', 'Song Title', and 'ARTIST'. The data is as follows:

ID	Song Title	ARTIST
47	Hurrah for Today	The Jubilant Trio
49	Lets Celebrate	The Celebrants

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

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';
```



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 'chithra k chithru', and a 'Run' button. The main area is titled 'SQL Commands' with tabs for 'Language' (set to 'SQL'), 'Rows' (set to 10), 'Clear Command', and 'Find Tables'. Below these are icons for Undo, Redo, Search, and Run. The SQL command is pasted into the editor:

```
1 CREATE OR REPLACE VIEW view_d_song AS
2 SELECT d_songs.id, d_songs.title "Song Title", d_songs.artist, d_songs.type_code
3 from d_songs INNER JOIN d_types ON d_songs.type_code = d_types.code
4 where d_types.description = 'new age';
```

Below the command, the results show 'View created.' and a execution time of '0.03 seconds'. There are tabs for 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History'.

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);
```



The screenshot shows the Oracle SQL Workshop interface, identical to the previous one but with a different SQL command. The top navigation bar and schema selection are the same. The SQL command is:

```
1 CREATE OR REPLACE VIEW view_d_events_pkgs AS
2 SELECT evt.name "Name of Event", TO_CHAR(evt.event_date, 'dd-Month-yyyy') "Event date", thm.description "Theme description"
3 FROM d_events evt INNER JOIN d_themes thm ON evt.theme_code = thm.code
4 WHERE evt.event_date <= ADD_MONTHS(SYSDATE,1);
5
```

The results show 'View created.' and a execution time of '0.03 seconds'. The interface includes tabs for 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History'.

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);
```



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 'chithra k chithru', and a 'Run' button. The main workspace has tabs for Language (SQL selected), Rows (10), Clear Command, and Find Tables. Below these are icons for Undo, Redo, Save, and Run. The SQL command area contains the code provided above. The results tab is active, showing the message 'View created.' and a execution time of '0.03 seconds'.

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';
```

The screenshot shows the Oracle SQL Workshop interface with the 'SQL Commands' tab selected. The schema is set to 'WKSP_CHITHU'. The code entered is:

```
1 SELECT owner, table_name, column_name, updatable,insertable, deletable
2 FROM user_updatable_columns WHERE LOWER(table_name) = 'copy_d_songs';
3
```

The results table has the following columns: OWNER, TABLE_NAME, COLUMN_NAME, UPDATABLE, INSERTABLE, and DELETABLE. The data is:

OWNER	TABLE_NAME	COLUMN_NAME	UPDATABLE	INSERTABLE	DELETABLE
WKSP_CHITHU	COPY_D_SONGS	ID	YES	YES	YES
WKSP_CHITHU	COPY_D_SONGS	TITLE	YES	YES	YES
WKSP_CHITHU	COPY_D_SONGS	DURATION	YES	YES	YES
WKSP_CHITHU	COPY_D_SONGS	ARTIST	YES	YES	YES
WKSP_CHITHU	COPY_D_SONGS	TYPE_CODE	YES	YES	YES

The screenshot shows the Oracle SQL Workshop interface with the 'SQL Commands' tab selected. The schema is set to 'WKSP_CHITHU'. The code entered is:

```
1 SELECT owner, table_name, column_name, updatable,insertable, deletable
2 FROM user_updatable_columns WHERE LOWER(table_name) = 'copy_d_events';
3
```

The results table has the following columns: OWNER, TABLE_NAME, COLUMN_NAME, UPDATABLE, INSERTABLE, and DELETABLE. The data is:

OWNER	TABLE_NAME	COLUMN_NAME	UPDATABLE	INSERTABLE	DELETABLE
WKSP_CHITHU	COPY_D_EVENTS	ID	YES	YES	YES
WKSP_CHITHU	COPY_D_EVENTS	NAME	YES	YES	YES
WKSP_CHITHU	COPY_D_EVENTS	EVENT_DATE	YES	YES	YES

3 rows returned in 0.04 seconds

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;
```

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 'chithra k chithu', and a 'Run' button. The main area is titled 'SQL Commands' with a 'Language' dropdown set to 'SQL'. It shows the following code:

```
1 CREATE OR REPLACE VIEW view_copy_d_songs AS
2 SELECT *
3 FROM copy_d_songs;
```

Below the code, the 'Results' tab is selected, showing the message 'View created.' and a execution time of '0.03 seconds'.

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 , 'The What',4);
```

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 'chithra k chithu', and a 'Run' button. The main area is titled 'SQL Commands' with a 'Language' dropdown set to 'SQL'. It shows the following code:

```
1 INSERT INTO view_copy_d_songs(id,title,duration,artist,type_code)
2 VALUES(88,'Mello Jello',2 , 'The What',4);
3
```

Below the code, the 'Results' tab is selected, showing the message '1 row(s) inserted.' and a execution time of '0.02 seconds'.

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;
```

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 'chithra k chithu', and a 'Run' button. The main area is titled 'SQL Commands' with a schema dropdown set to 'WKSP_CHITHU'. Below the title are buttons for Language (SQL), Rows (10), Clear Command, and Find Tables. The SQL editor contains the following code:

```
1 CREATE OR REPLACE VIEW read_copy_d_cds AS
2 SELECT *
3 FROM copy_d_cds
4 WHERE year = '2000'
5 WITH READ ONLY;
```

Below the editor, tabs for 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History' are visible. The results section displays the message 'View created.' and '0.04 seconds' execution time.

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

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

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

The results section shows a yellow box containing the error message:

Error at line 1/13: ORA-42399: cannot perform a DML operation on a read-only view

1. DELETE FROM read_copy_d_cds WHERE cd_number = 90;

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;
```

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 'chithra k chithu', and a schema dropdown set to 'WKSP_CHITHU'. The main area is titled 'SQL Commands' with tabs for Language (set to SQL), Rows (set to 10), Clear Command, and Find Tables. Below these are standard toolbar icons for copy, paste, search, and refresh. The code editor contains the following SQL:

```
1 CREATE OR REPLACE VIEW read_copy_d_cds AS  
2 SELECT *  
3 FROM copy_d_cds  
4 WHERE year = '2000'  
5 WITH CHECK OPTION CONSTRAINT ck_read_copy_d_cds;  
6
```

Below the code editor, there are tabs for Results, Explain, Describe, Saved SQL, and History. The results pane displays the message "View created." and a execution time of "0.03 seconds".

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';
```

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 'chithra k chithu', and a schema dropdown set to 'WKSP_CHITHU'. The main area is titled 'SQL Commands' with tabs for Language (set to SQL), Rows (set to 10), Clear Command, and Find Tables. Below these are standard toolbar icons for copy, paste, search, and refresh. The code editor contains the following SQL:

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

Below the code editor, there are tabs for Results, Explain, Describe, Saved SQL, and History. The results pane displays the message "1 row(s) deleted." and a execution time of "0.01 seconds".

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;
```

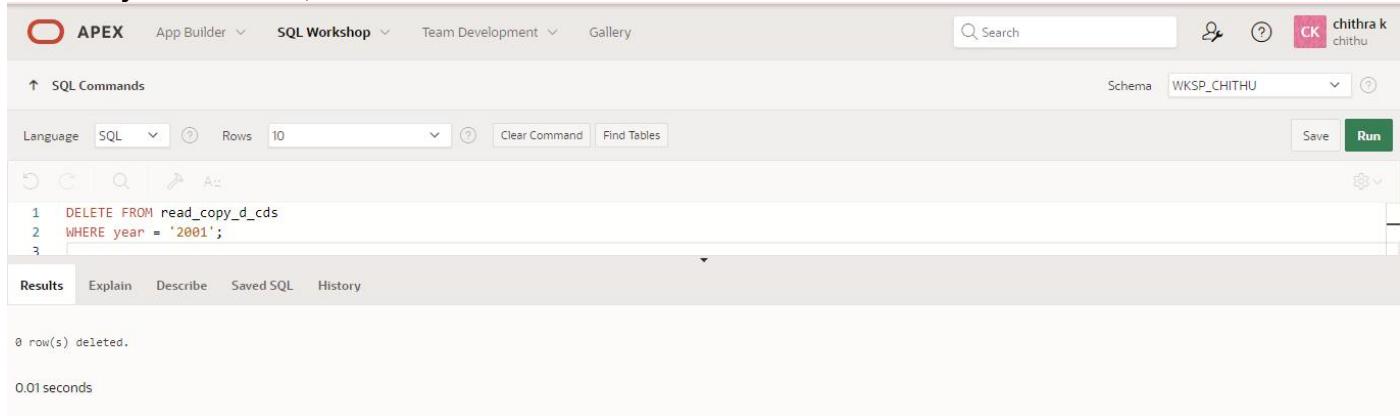
The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there's a search bar, a user icon for 'chithra k chithu', and a schema dropdown set to 'WKSP_CHITHU'. The main area is titled 'SQL Commands' with tabs for Language (set to SQL), Rows (set to 10), Clear Command, and Find Tables. Below these are standard toolbar icons for copy, paste, search, and refresh. The code editor contains the following SQL:

```
1 DELETE FROM read_copy_d_cds  
2 WHERE cd_number = 90;  
3
```

Below the code editor, there are tabs for Results, Explain, Describe, Saved SQL, and History. The results pane displays the message "0 row(s) deleted." and a execution time of "0.01 seconds".

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

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



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 'chithra k chithu', and a schema dropdown set to 'WKSP_CHITHU'. The main workspace is titled 'SQL Commands' and contains a code editor with the following SQL command:

```
1 DELETE FROM read_copy_d_cds
2 WHERE year = '2001';
3
```

Below the code editor, the results tab is selected, showing the output: '0 row(s) deleted.' and '0.01 seconds'. There are also tabs for Explain, Describe, Saved SQL, and History.

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;
```

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, user profile 'chithra k chithu', and a 'Run' button. The main area is titled 'SQL Commands' with a sub-section 'View created.' and a note '0.03 seconds'. The code editor contains the following SQL:

```
1 CREATE OR REPLACE VIEW view_copy_d_songs AS
2 SELECT title, artist
3 FROM copy_d_songs;
4
```

```
SELECT * FROM view_copy_d_songs;
```

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, user profile 'chithra k chithu', and a 'Run' button. The main area is titled 'SQL Commands' with a sub-section '1 rows returned in 0.01 seconds'. The code editor contains the following SQL:

```
1 SELECT * FROM view_copy_d_songs;
```

The results table below shows the output:

TITLE	ARTIST
Mello Jello	The What

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;
```

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, user profile 'chithra k chithu', and a 'Run' button. The main area is titled 'SQL Commands' with a sub-section 'View dropped.' and a note '0.05 seconds'. The code editor contains the following SQL:

```
1 DROP VIEW view_copy_d_songs;
2
3
```

A screenshot of 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 search bar, user profile 'chithra k chithu', and a schema dropdown set to 'WKSP_CHITHU'. The main area shows a SQL command window with the following content:

```
1 SELECT * FROM view_copy_d_songs;
```

The 'Results' tab is selected. Below it, an error message is displayed: "Error at line 1/15: ORA-00942: table or view does not exist".

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;
```

A screenshot of the Oracle SQL Workshop interface, identical to the first one but showing the results of the executed query. The results are displayed in a table:

LAST_NAME	SALARY
Mohanraj	100000
sree	60000
Matos	50000

Below the table, the message "3 rows returned in 0.02 seconds" is shown, along with a "Download" link.

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;
```

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 'chithra k chithu', and a 'Run' button. The main area has tabs for SQL Commands, Language (SQL selected), Rows (10), Clear Command, Find Tables, Schema (WKSP_CHITHU), Save, and Run. The SQL command window contains the query from step 4. The Results tab is selected, displaying a table with three columns: LAST_NAME, SALARY, and DEPT_ID. The data shows two rows: bhuvaneshwari (120000, 420) and Zlotkey (8000, 200). A note at the bottom says '2 rows returned in 0.02 seconds'. There are also Explain, Describe, Saved SQL, and History tabs.

LAST_NAME	SALARY	DEPT_ID
bhuvaneshwari	120000	420
Zlotkey	8000	200

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?

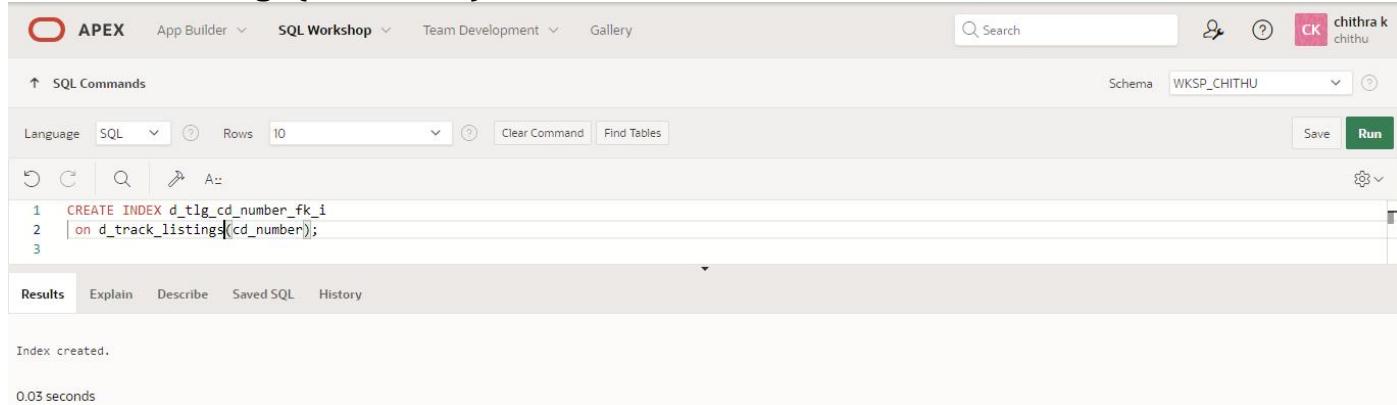
Indexes use ROWID's (base 64 string representation of the row address containing block identifier, row location in the block and the database file identifier) which is the fastest way to access any particular row.

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);**



The screenshot shows the Oracle Application Express 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 'chithra k' and the schema 'WKSP_CHITHU'. The main area has tabs for SQL Commands, Results, Explain, Describe, Saved SQL, and History. The SQL tab is active, showing the command:

```
1 CREATE INDEX d_tlg_cd_number_fk_i
2   | on d_track_listings(cd_number);
3
```

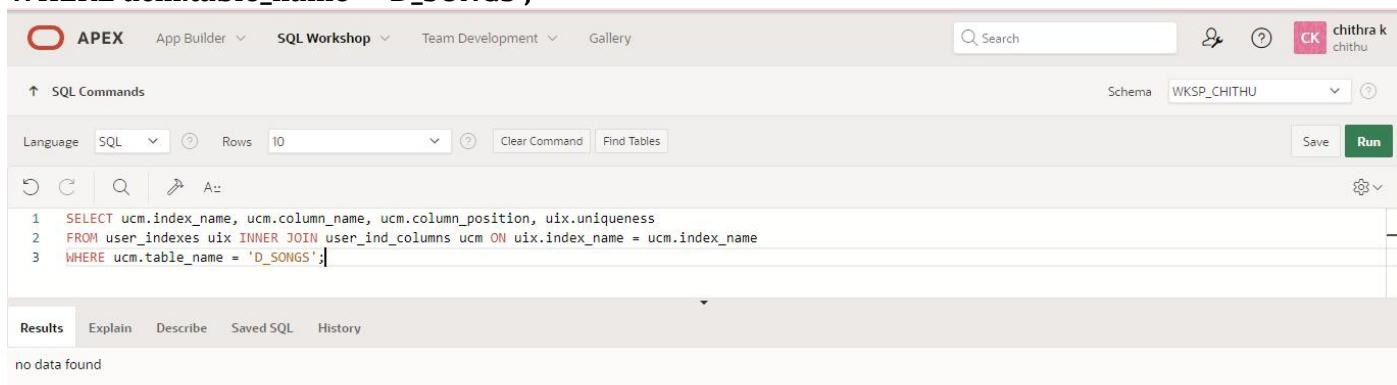
The Results tab shows the output:

```
Index created.
```

Execution time: 0.03 seconds.

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';**



The screenshot shows the Oracle Application Express 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 'chithra k' and the schema 'WKSP_CHITHU'. The main area has tabs for SQL Commands, Results, Explain, Describe, Saved SQL, and History. The SQL tab is active, showing the command:

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

The Results tab shows the output:

```
no data found
```

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';

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, and Gallery. The right side shows a user profile for 'chithra k chithu'. The main area is titled 'SQL Commands' with a search bar and a schema dropdown set to 'WKSP_CHITHU'. The code editor contains the following SQL command:

```
1 | SELECT index_name, table_name,uniqueness FROM user_indexes where table_name = 'D_EVENTS';
2 | 
```

The results section shows a table with three columns: INDEX_NAME, TABLE_NAME, and UNIQUENESS. The data is as follows:

INDEX_NAME	TABLE_NAME	UNIQUENESS
MY_D_EVENTS_PK	D_EVENTS	NONUNIQUE

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

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;

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, and Gallery. The right side shows a user profile for 'chithra k chithu'. The main area is titled 'SQL Commands' with a search bar and a schema dropdown set to 'WKSP_CHITHU'. The code editor contains the following SQL command:

```
1 | CREATE SYNONYM dj_tracks FOR d_track_listings;|
```

The results section shows the message 'Synonym created.' and a time of '0.02 seconds'.

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));**

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, and Gallery. The right side shows a user profile for 'chithra k chithu'. The main area is titled 'SQL Commands' with a search bar and a schema dropdown set to 'WKSP_CHITHU'. The code editor contains the following SQL command:

```
1 | CREATE INDEX d_ptr_last_name_idx
2 | ON d_partners(LOWER(last_name));
3 | 
```

The results section shows the message 'Index created.' and a time of '0.03 seconds'.

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');

The screenshot shows the Oracle SQL Workshop interface. In the top navigation bar, 'APEX' is selected. The main area is titled 'SQL Commands'. A single line of SQL code is entered: 'CREATE SYNONYM dj_tracks2 FOR d_track_listings;'. Below the command, the message 'Synonym created.' is displayed. The results tab shows the execution time as '0.02 seconds'.

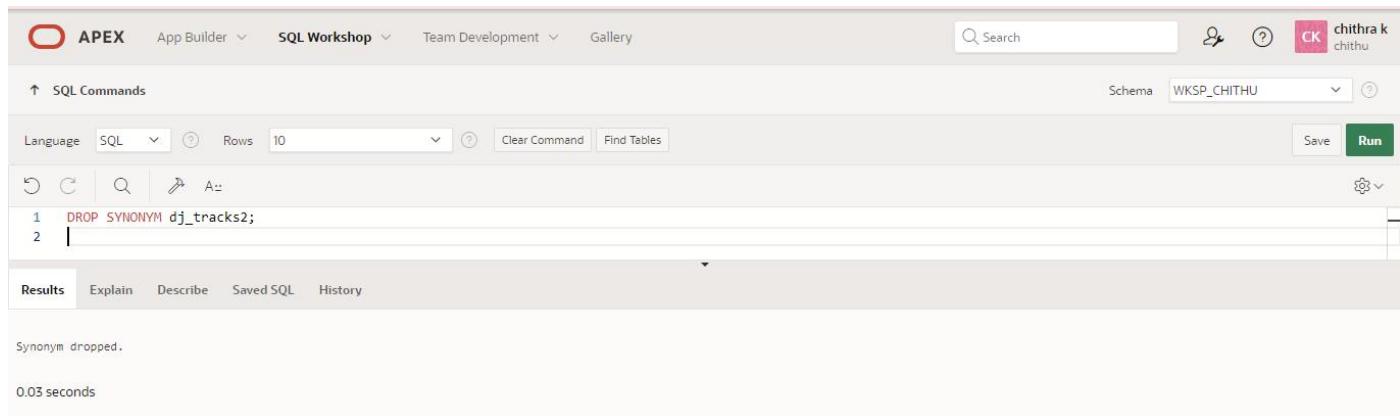
The screenshot shows the Oracle SQL Workshop interface. In the top navigation bar, 'APEX' is selected. The main area is titled 'SQL Commands'. A single line of SQL code is entered: 'SELECT * FROM user_synonyms WHERE table_NAME = UPPER('d_track_listings');'. The results tab displays a table with two rows. The table has columns: SYNONYM_NAME, TABLE_OWNER, TABLE_NAME, DB_LINK, and ORIGIN_CON_ID. The data is as follows:

SYNONYM_NAME	TABLE_OWNER	TABLE_NAME	DB_LINK	ORIGIN_CON_ID
DJ_TRACKS	WKSP_CHITHU	D_TRACK_LISTINGS	-	0
DJ_TRACKS2	WKSP_CHITHU	D_TRACK_LISTINGS	-	0

Below the table, it says '2 rows returned in 0.03 seconds'.

10.Drop the synonym that you created in question

DROP SYNONYM dj_tracks2;



The screenshot shows the Oracle SQL Workshop interface. In the top navigation bar, 'APEX' is selected. The main area is titled 'SQL Commands'. A single line of code is entered: 'DROP SYNONYM dj_tracks2;'. Below the code, the output shows 'Synonym dropped.' and a execution time of '0.03 seconds'. The 'Run' button at the top right is highlighted.

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

RESULT:

Thus the queries for creating views are executed successfully.

OTHER DATABASE OBJECTS

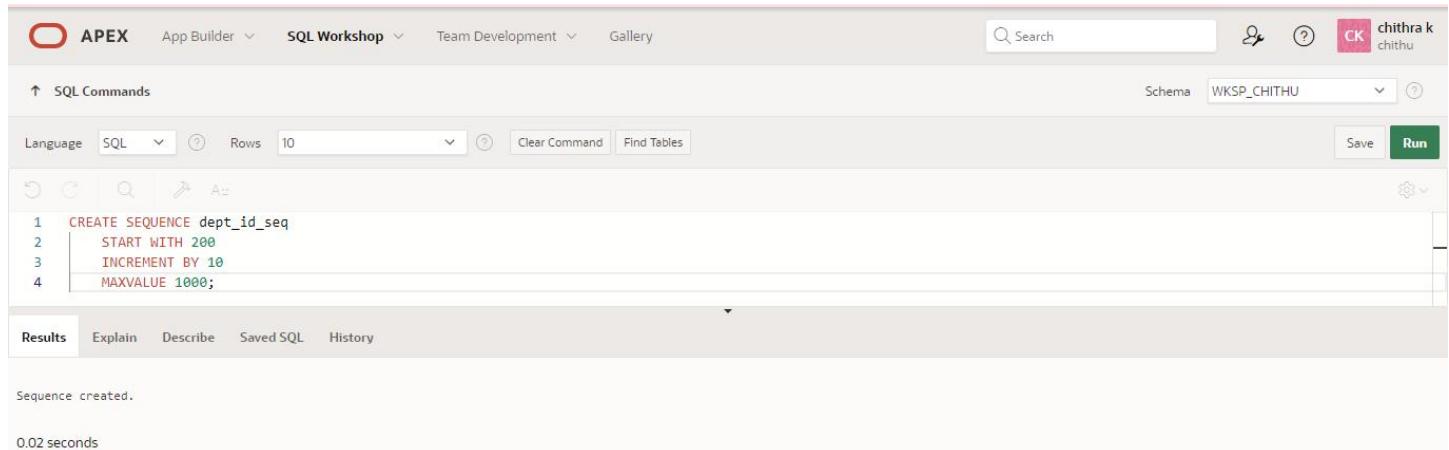
EX_NO:14

DATE:16/05/2024

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

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

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. In the SQL Commands tab, the following SQL code is run:

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

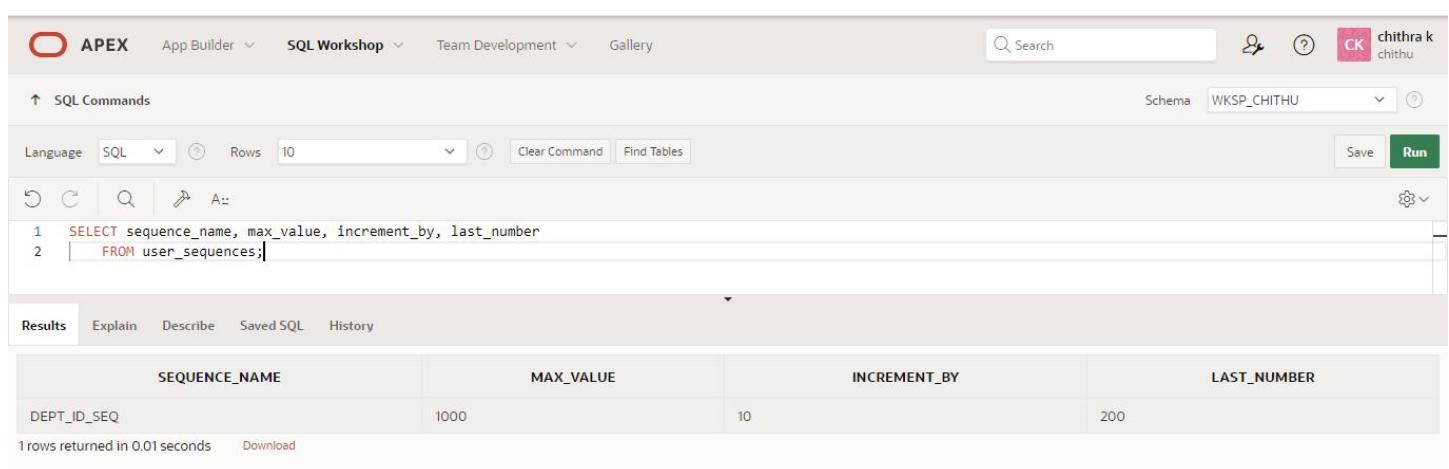
The results show the message "Sequence created." and a execution time of "0.02 seconds".

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. In the SQL Commands tab, the following SQL code is run:

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

The results show a table with the following data:

SEQUENCE_NAME	MAX_VALUE	INCREMENT_BY	LAST_NUMBER
DEPT_ID_SEQ	1000	10	200

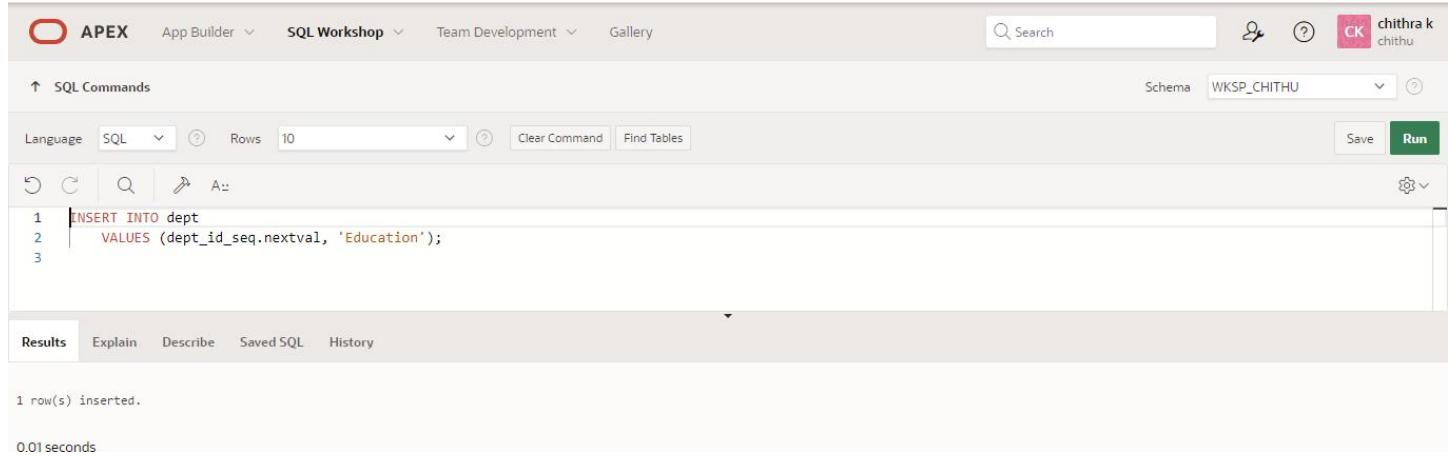
1 rows returned in 0.01 seconds

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

QUERY:

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

OUTPUT:

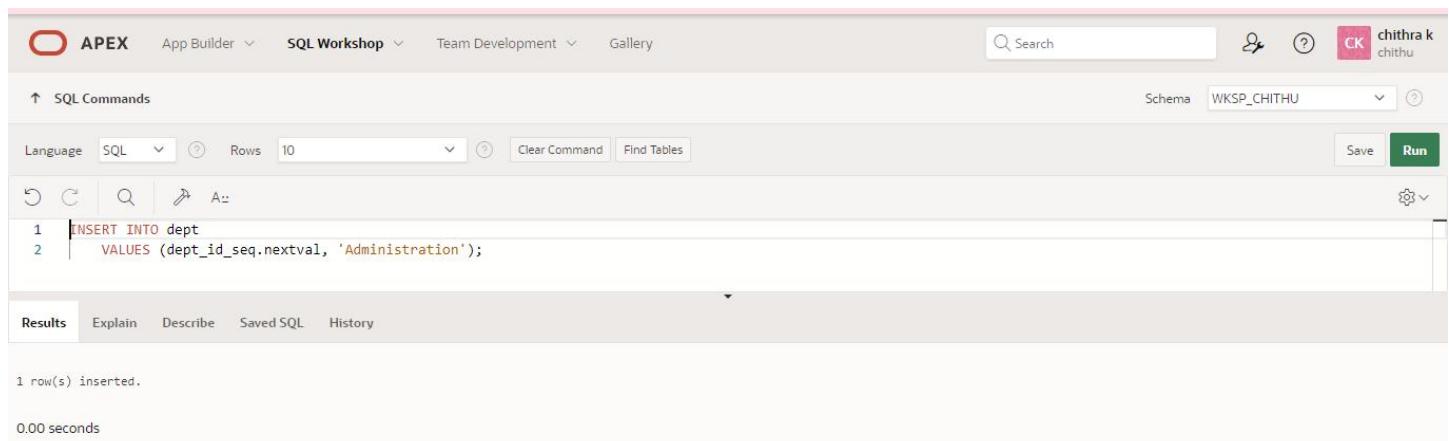


Screenshot of Oracle SQL Workshop showing the first insert command. The interface includes a top navigation bar with APEX, App Builder, SQL Workshop, Team Development, and Gallery. The SQL Workshop tab is selected. On the left, there's a toolbar with icons for Undo, Redo, Search, and Find Tables. The main area shows the SQL command:

```
1 | INSERT INTO dept
2 |   VALUES (dept_id_seq.nextval, 'Education');
3 |
```

The results section below shows the output:

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



Screenshot of Oracle SQL Workshop showing the second insert command. The interface is identical to the previous screenshot. The SQL command is:

```
1 | INSERT INTO dept
2 |   VALUES (dept_id_seq.nextval, 'Administration');
```

The results section shows:

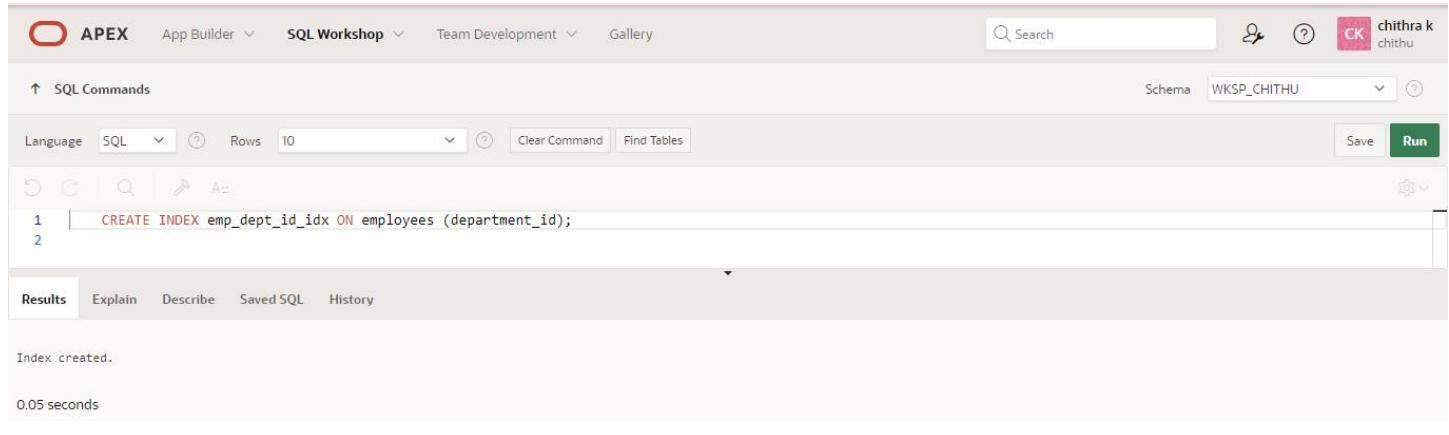
```
1 row(s) inserted.
0.00 seconds
```

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

QUERY:

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

OUTPUT:



Screenshot of Oracle SQL Workshop showing the index creation command. The interface is identical to the previous screenshots. The SQL command is:

```
1 | CREATE INDEX emp_dept_id_idx ON employees (department_id);
2 |
```

The results section shows:

```
Index created.
0.05 seconds
```

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

QUERY:

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

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, and Gallery. On the right, there's a search bar, a user icon for 'chithra k chithu', and a schema dropdown set to 'WKSP_CHITHU'. The main area has tabs for SQL Commands, Results, Explain, Describe, Saved SQL, and History. The SQL Commands tab contains the following query:

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

The Results tab displays the output:

INDEX_NAME	TABLE_NAME	UNIQUENESS
EMP_DEPT_ID_IDX	EMPLOYEES	NONUNIQUE

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

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

RESULT:

Thus the queries for other database objects are executed successfully.

CONTROLLING USER ACCESS

EX_NO:15

DATE:16/05/2024

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

The CREATE SESSION system privilege

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

The CREATE TABLE privilege

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

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

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

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

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

The ALTER USER statement

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

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

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

7. Query all the rows in your DEPARTMENTS table.

SELECT * FROM departments;

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

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

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

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

SELECT table_name FROM user_tables;

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

Team 1 revokes the privilege.

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

Team 2 revokes the privilege.

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

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

Team 1 executes this INSERT statement.

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

Team 2 executes this INSERT statement.

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

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

RESULT:

Thus the queries for controlling user access are executed successfully.

PL/SQL

CONTROL STRUCTURES

EX_NO:16

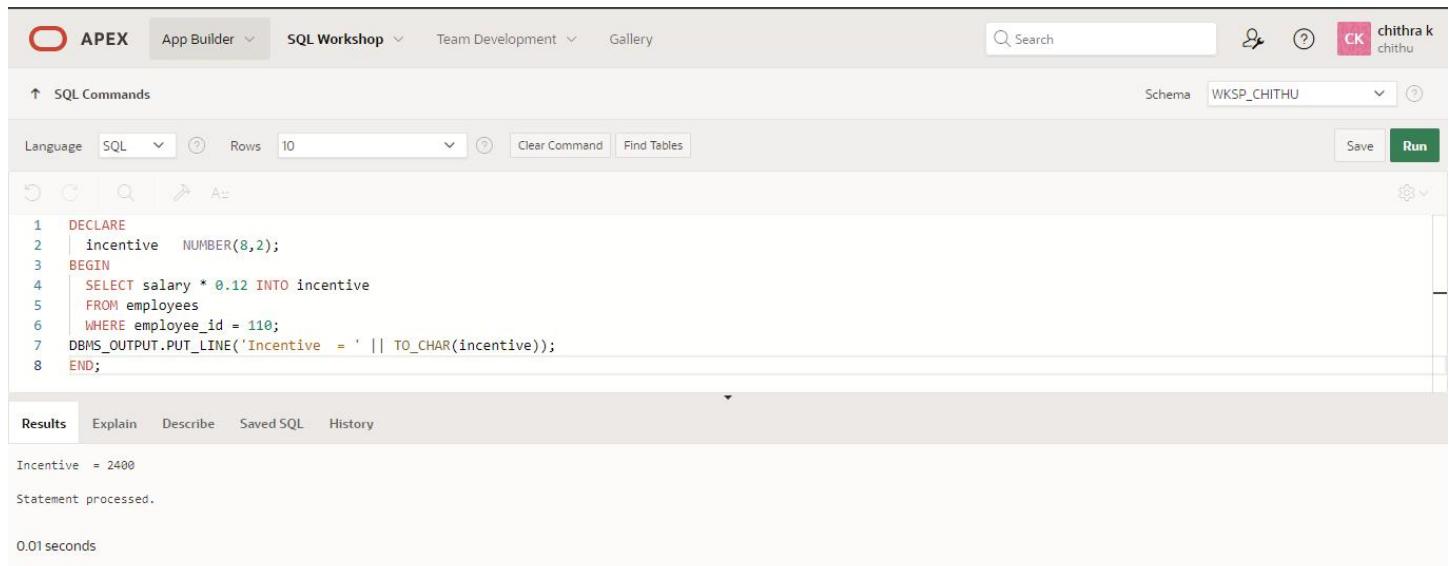
DATE:18/05/2024

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, and Gallery. On the right, there's a search bar, a user icon for 'chithra k chithu', and a 'Run' button. The main workspace has tabs for SQL Commands, Explain, Describe, Saved SQL, and History. The SQL Commands tab is active, displaying the following PL/SQL code:

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

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

```
Incentive = 2400
Statement processed.

0.01 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;
/
```

DECLARE

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

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. In the SQL Commands pane, there is a code editor with the following PL/SQL block:

```
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 /
```

Below the code editor, the Results pane displays the execution output. A yellow box highlights 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.INV_DBMS_SQL_APEX_230200", line 801
ORA-06550: line 4, column 3:
PL/SQL: Statement ignored
```

Underneath the error message, the original code is shown again:

```
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 screenshot shows the Oracle SQL Workshop interface. In the SQL Commands pane, there is a code editor with the following PL/SQL block:

```
1 DECLARE
2   WELCOME varchar2(10) := 'welcome'; -- identifier without quotation
3 BEGIN
4   DBMS_Output.Put_Line("Welcome"); --reference to the identifier with quotation and different case
5 END;
6 /
```

Below the code editor, the Results pane displays the execution output. A yellow box highlights 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.INV_DBMS_SQL_APEX_230200", line 801
ORA-06550: line 4, column 3:
PL/SQL: Statement ignored
```

Underneath the error message, the original code is shown again:

```
2.   WELCOME varchar2(10) := 'welcome'; -- identifier without quotation
3. BEGIN
4.   DBMS_Output.Put_Line("Welcome"); --reference to the identifier with quotation and different case
5. END;
6. /
```

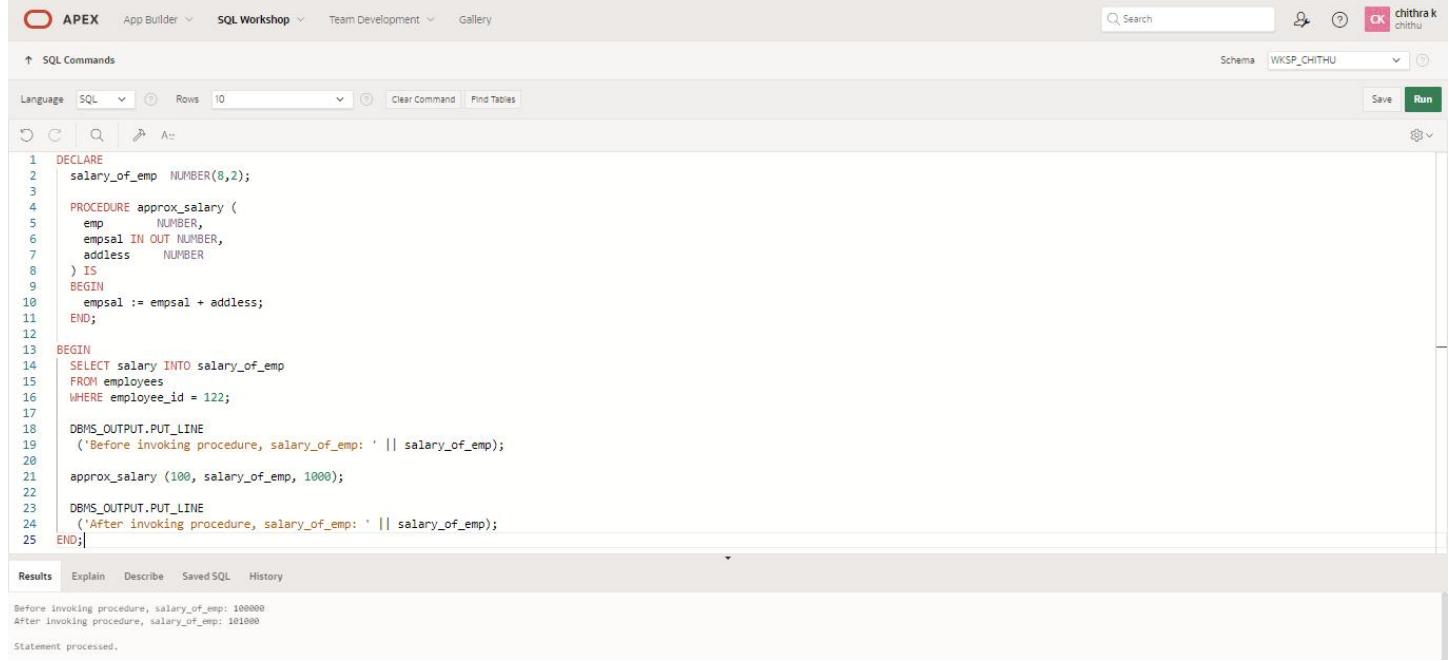
3.) Write a PL/SQL block to adjust the salary of the employee whose ID 122.

QUERY:

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

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

/OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (selected), 'Team Development', and 'Gallery'. The top right shows the user 'chithra k' and the schema 'WKSP_CHITHU'. The main area is titled 'SQL Commands' with a 'Language' dropdown set to 'SQL'. The code area contains the PL/SQL block provided above. The results section at the bottom displays the output of the DBMS_OUTPUT.PUT_LINE statements:

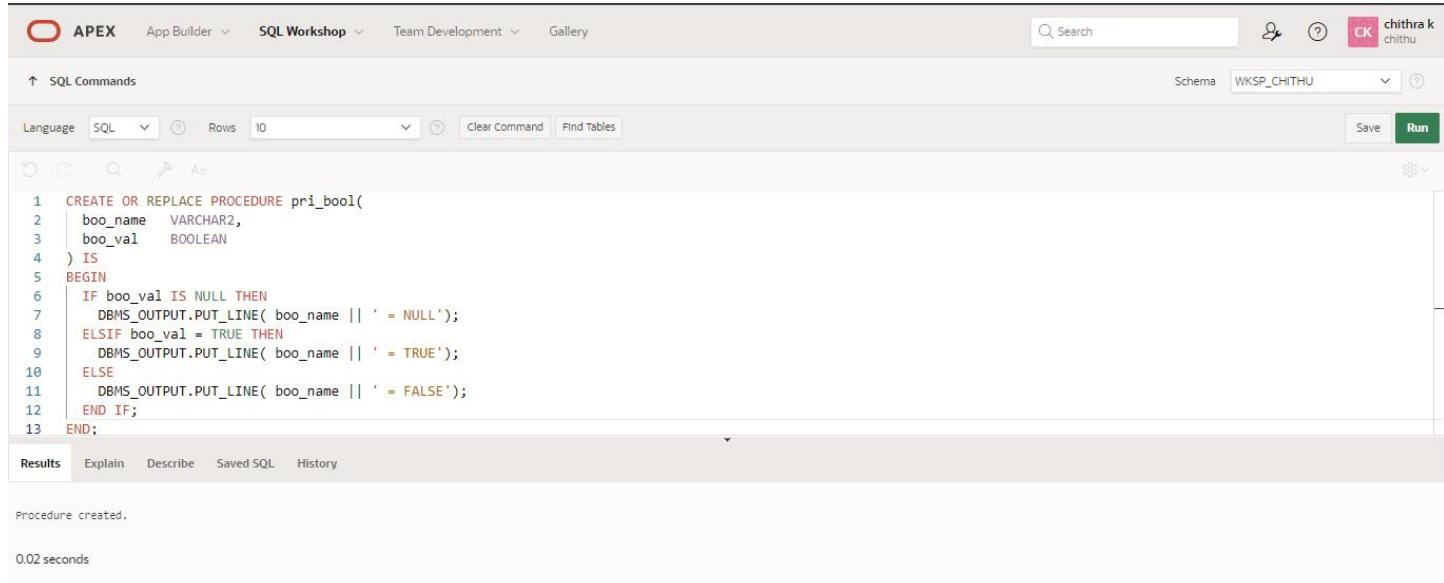
```
Before invoking procedure, salary_of_emp: 100000
After invoking procedure, salary_of_emp: 101000
Statement processed.
```

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

QUERY:

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

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, and Gallery. The user is logged in as 'chithra k chithu'. The main workspace is titled 'SQL Commands' and contains the PL/SQL code for the 'pri_bool' procedure. The code uses the IS NULL operator to check if a boolean value is null, and the AND operator (represented by the IS operator) to ensure both boolean values are true. The code is numbered from 1 to 13. Below the code, the 'Results' tab is selected, showing the message 'Procedure created.' and a execution time of '0.02 seconds'.

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

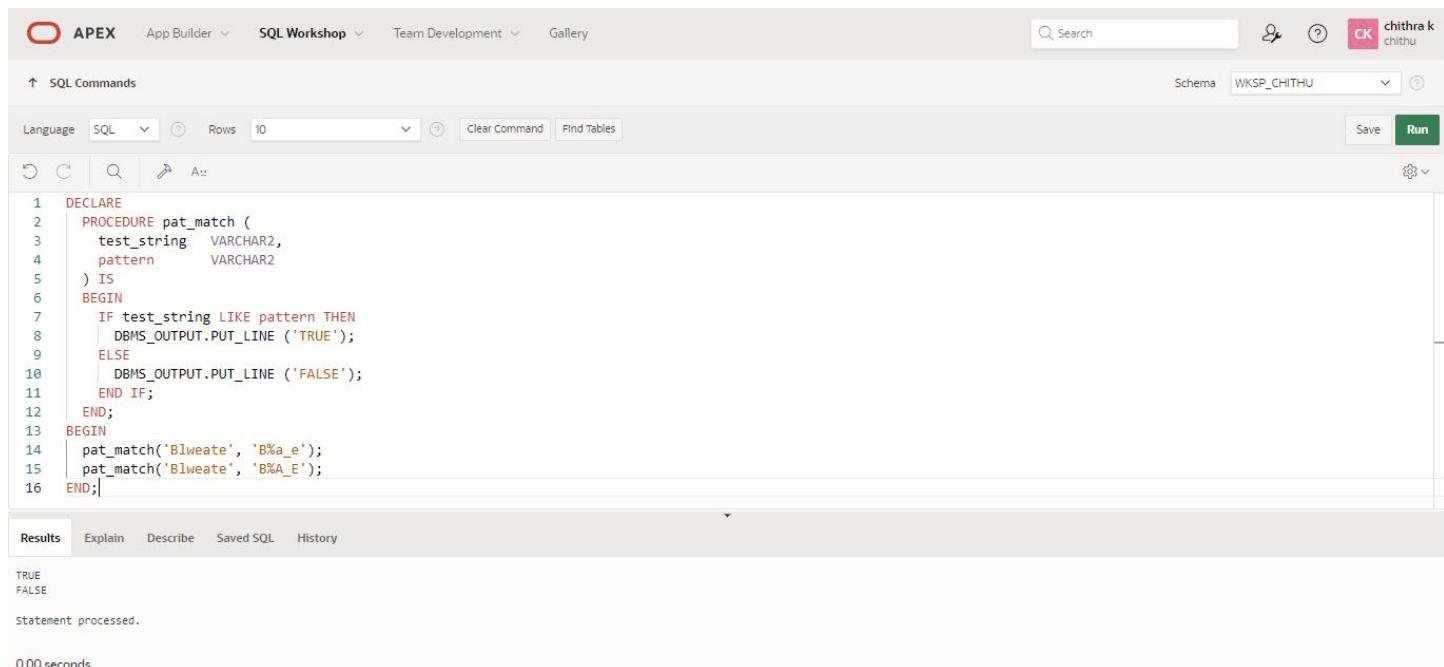
Procedure created.
0.02 seconds

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

QUERY:

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

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows a user profile for 'chithra k chithu'. The main area is titled 'SQL Commands' and contains the PL/SQL code from the previous section. The code is numbered 1 through 16. Lines 13 and 14 are highlighted in blue, indicating they are the statements being executed. The results tab at the bottom shows the output: 'TRUE' and 'FALSE', followed by the message 'Statement processed.' and a timestamp of '0.00 seconds'.

```
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;|
```

Results	Explain	Describe	Saved SQL	History
TRUE FALSE Statement processed. 0.00 seconds				

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

QUERY:

DECLARE

num_small NUMBER := 8;

num_large NUMBER := 5;

num_temp NUMBER;

BEGIN

IF num_small > num_large THEN

num_temp := num_small;

num_small := num_large;

num_large := num_temp;

END IF;

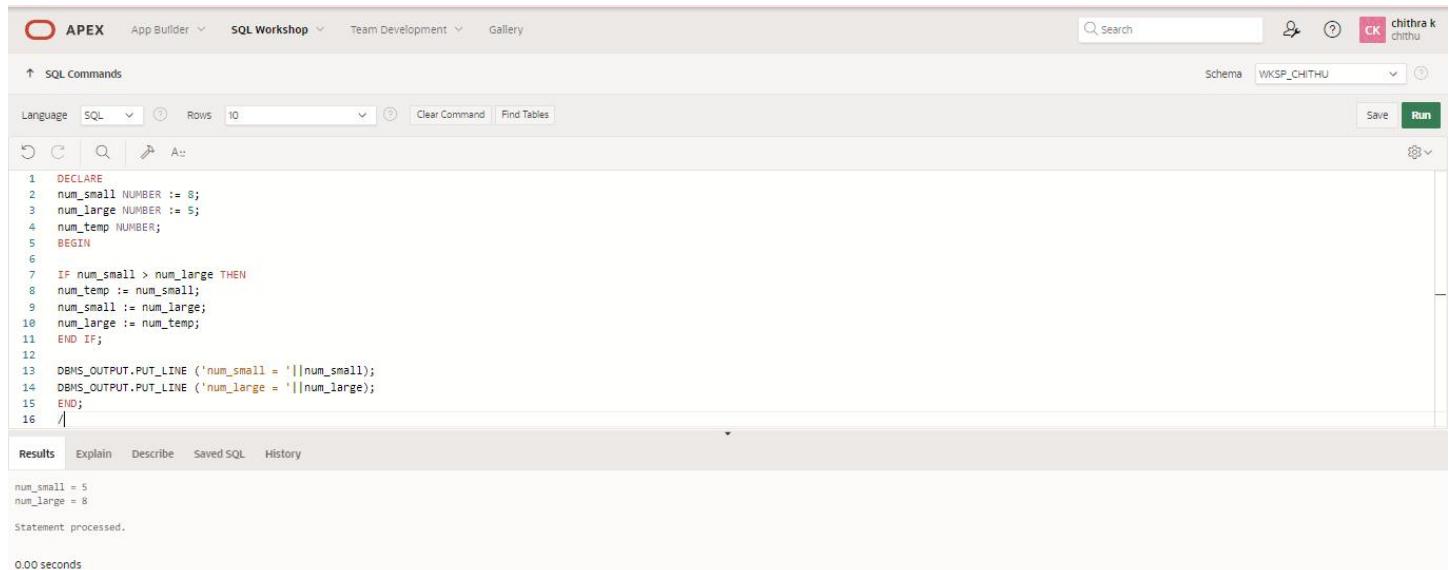
DBMS_OUTPUT.PUT_LINE ('num_small = '||num_small);

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

END;

/

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows a user profile for 'chithra k chithu'. The main area is titled 'SQL Commands' with a schema dropdown set to 'WKSP_CHITHU'. The code editor contains the PL/SQL block provided in the question. The results tab at the bottom shows the output: 'num_small = 5' and 'num_large = 8', followed by 'Statement processed.' and a timestamp of '0.00 seconds'.

```
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;
16 /
```

Results Explain Describe Saved SQL History

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;
/
```

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, and Gallery. The SQL Workshop tab is selected. The schema dropdown is set to WKSP_CHITHU. The main area displays the following PL/SQL code:

```
1  DECLARE
2      PROCEDURE test1 (
3          sal_achieve NUMBER,
4          target_qty NUMBER,
5          emp_id NUMBER
6      )
7      IS
8          incentive NUMBER := 0;
9          updated VARCHAR2(3) := 'No';
10     BEGIN
11         IF sal_achieve > (target_qty + 200) THEN
12             incentive := (sal_achieve - target_qty)/4;
```

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

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

Execution time: 0.02 seconds.

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop, Team Development, and Gallery. The SQL Workshop tab is selected. The schema dropdown is set to WKSP_CHITHU. The main area displays the following PL/SQL block:

```
14    SET salary = salary + incentive
15    WHERE employee_id = emp_id;
16    updated := 'Yes';
17    END IF;
18    DBMS_OUTPUT.PUT_LINE (
19        'Table updated? ' || updated || ', ' ||
20        'incentive = ' || incentive || '.';
21    );
22    END test1;
23 BEGIN
24     test1(2300, 2000, 144);
25     test1(3600, 3000, 145);
26 END;
27 /
```

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

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

Execution time: 0.02 seconds.

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

QUERY:

DECLARE

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

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery'. The right side of the header shows the user 'chithra k' and the schema 'WKSP_CHITHU'. The main area is titled 'SQL Commands' with a 'Language' dropdown set to 'SQL'. Below the code editor, there are buttons for 'Run' and 'Save'. The code itself is the PL/SQL procedure defined above. In the 'Results' tab at the bottom, the output of the procedure is displayed: 'Sale achieved : 45000, incentive : 1800.', 'Sale achieved : 36000, incentive : 800.', and 'Sale achieved : 28000, incentive : 500.'. The bottom status bar shows the session ID '220701054@rajalakshmi.edu.in', the user 'chithra', and the page number '1 of 1'. The footer includes copyright information for Oracle and the text 'Oracle APEX 23.2.0'.

```
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;
```

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

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

QUERY:

```
SET SERVEROUTPUT ON
```

```
DECLARE
```

```
    tot_emp NUMBER;
```

```
get_dep_id NUMBER;
```

```
BEGIN
```

```
    get_dep_id := 80;
```

```
    SELECT Count(*)
```

```
    INTO tot_emp
```

```
    FROM employees e
```

```
        join departments d
```

```
            ON e.department_id = d.department_id
```

```
    WHERE e.department_id = get_dep_id;
```

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

```
        ||To_char(tot_emp));
```

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

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

```
    ELSE
```

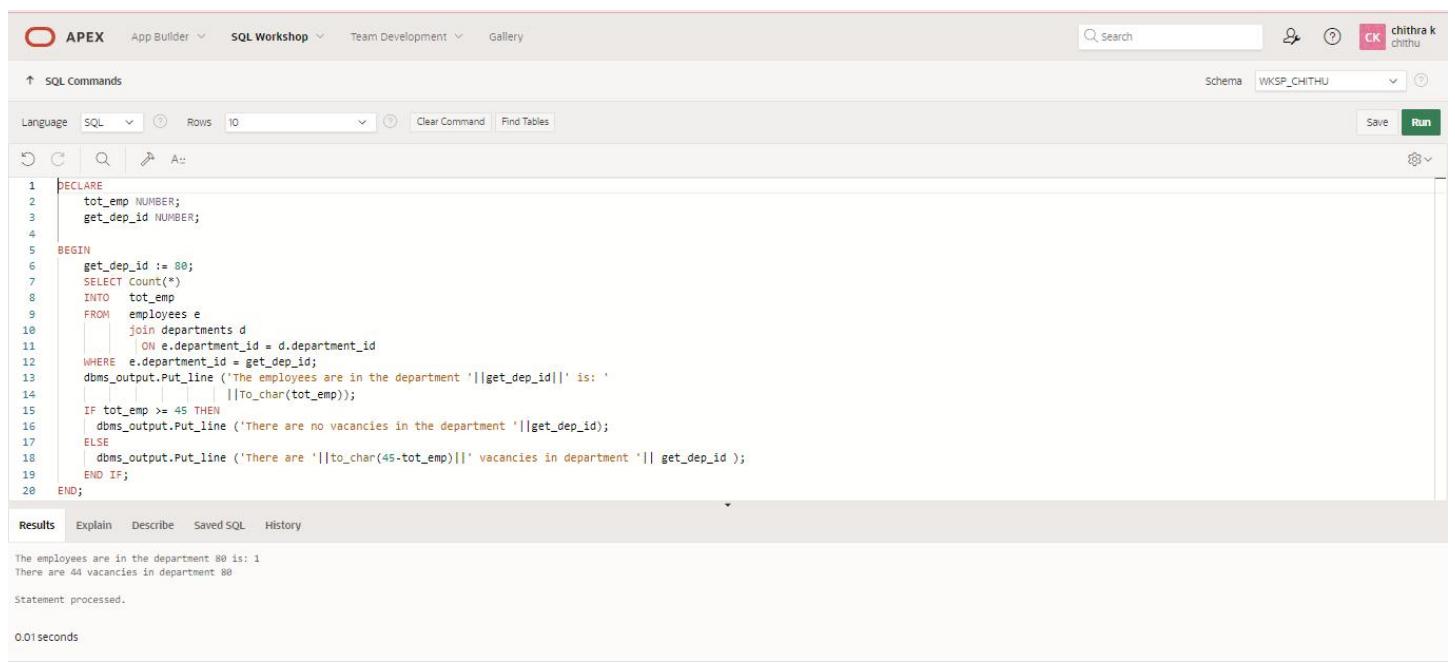
```
        dbms_output.Put_line ('There are'||to_char(45-tot_emp)||' vacancies in department'||
```

```
get_dep_id );
```

```
    END IF;
```

```
END;
```

/OUTPUT:



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

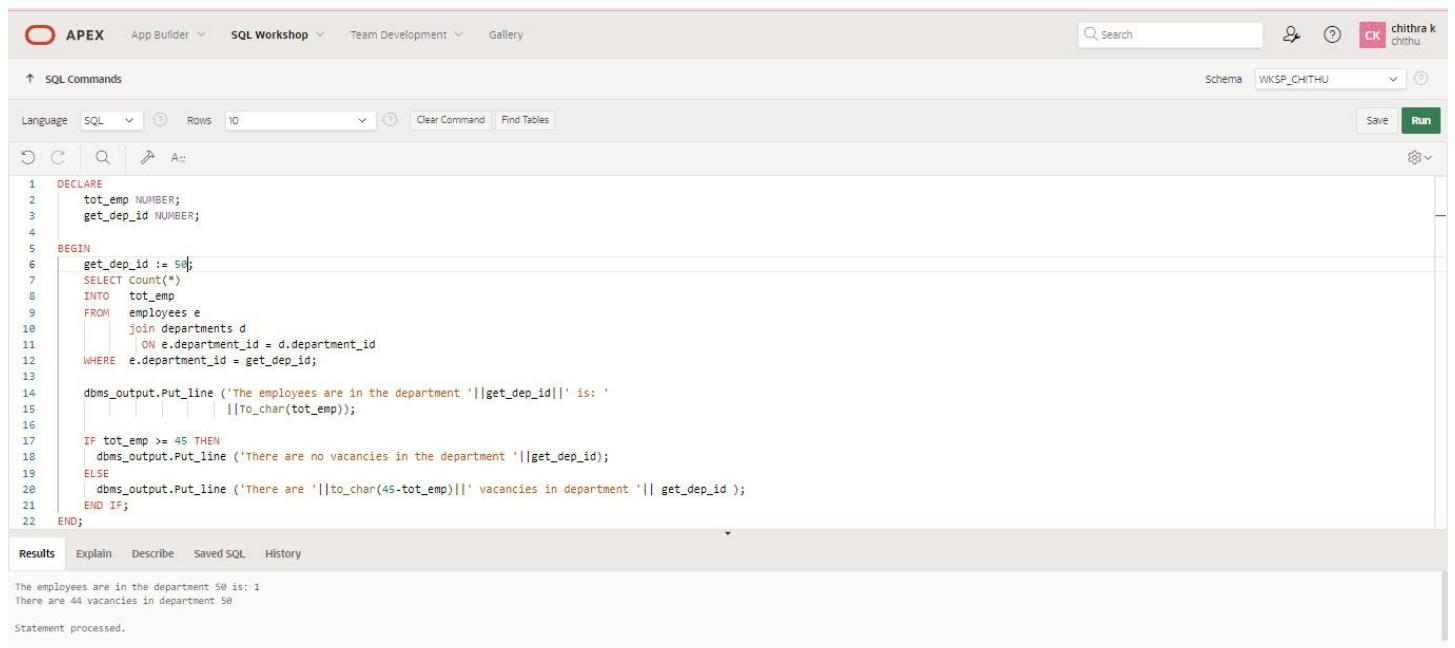
- Toolbar:** APEX, App Builder, SQL Workshop, Team Development, Gallery.
- Search Bar:** Search, Schema: WKSP_CHITHU.
- Language Selection:** Language: SQL, Rows: 10.
- Code Area:** The code block is pasted into the SQL Commands area. It contains a PL/SQL block that counts employees in department 80 and checks for vacancies. The output shows the count is 44, indicating 45 vacancies.
- Results Tab:** The Results tab is selected, showing the output of the query: "The employees are in the department 80 is: 1" and "There are 44 vacancies in department 80".
- Log:** Statement processed. 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:



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 'chithra k chithu'. The main area is titled 'SQL Commands' with tabs for Language (set to SQL), Rows (10), and various buttons like Save and Run. The code area contains the PL/SQL block from above. The results tab at the bottom displays the output of the program: 'The employees are in the department 80 is: 44' and 'There are 44 vacancies in department 80'. A note at the bottom says 'Statement processed.'

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

QUERY:

DECLARE

```
v_employee_id employees.employee_id%TYPE;
v_full_name employees.first_name%TYPE;
v_job_id employees.job_id%TYPE;
v_hire_date employees.hire_date%TYPE;
v_salary employees.salary%TYPE;
```

CURSOR c_employees IS

```
SELECT employee_id, first_name || ' ' || last_name AS full_name, job_id, hire_date, salary
FROM employees;
```

BEGIN

```
DBMS_OUTPUT.PUT_LINE('Employee ID | Full Name | Job Title | Hire Date | Salary');
```

```
DBMS_OUTPUT.PUT_LINE('-----');
```

```
OPEN c_employees;
```

```
FETCH c_employees INTO v_employee_id, v_full_name, v_job_id, v_hire_date, v_salary;
```

```
WHILE c_employees%FOUND LOOP
```

```
DBMS_OUTPUT.PUT_LINE(v_employee_id || ' ' || v_full_name || ' ' || v_job_id || ' ' ||
v_hire_date || ' ' || v_salary);
```

```
FETCH c_employees INTO v_employee_id, v_full_name, v_job_id, v_hire_date, v_salary;
```

```
END LOOP;
```

```
CLOSE c_employees;
```

```
END;
```

/OUTPUT:

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

Employee ID | Full Name | Job Title | Hire Date | Salary

118	henry john 46 03/23/1990 20000
122	bhuva Mohanraj 88 02/14/1995 100000

Statement processed.

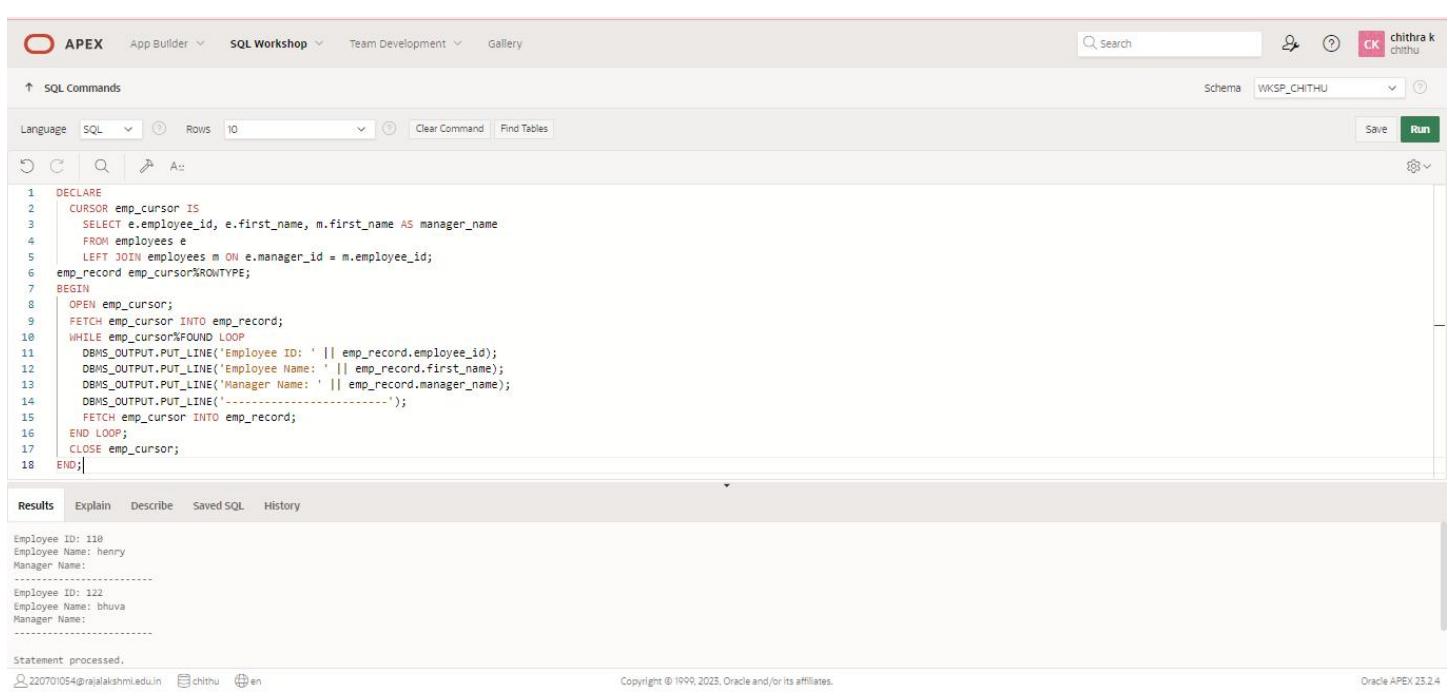
12.) Write a PL/SQL program to display the employee IDs, names, and department names of all employees.

QUERY:

DECLARE

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

/OUTPUT:



The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. The SQL Workshop tab is selected. The schema dropdown is set to WKSP_CHITHU. The main area displays the PL/SQL code for listing employees and their managers. Below the code, the Results tab is active, showing the output of the executed query. The output lists two employees: Henry (Employee ID 110) and Bhuvan (Employee ID 122), along with their respective manager names. At the bottom of the results, a message indicates the statement was processed.

```
Employee ID: 110
Employee Name: henry
Manager Name:
-----
Employee ID: 122
Employee Name: bhuvan
Manager Name:
```

Statement processed.

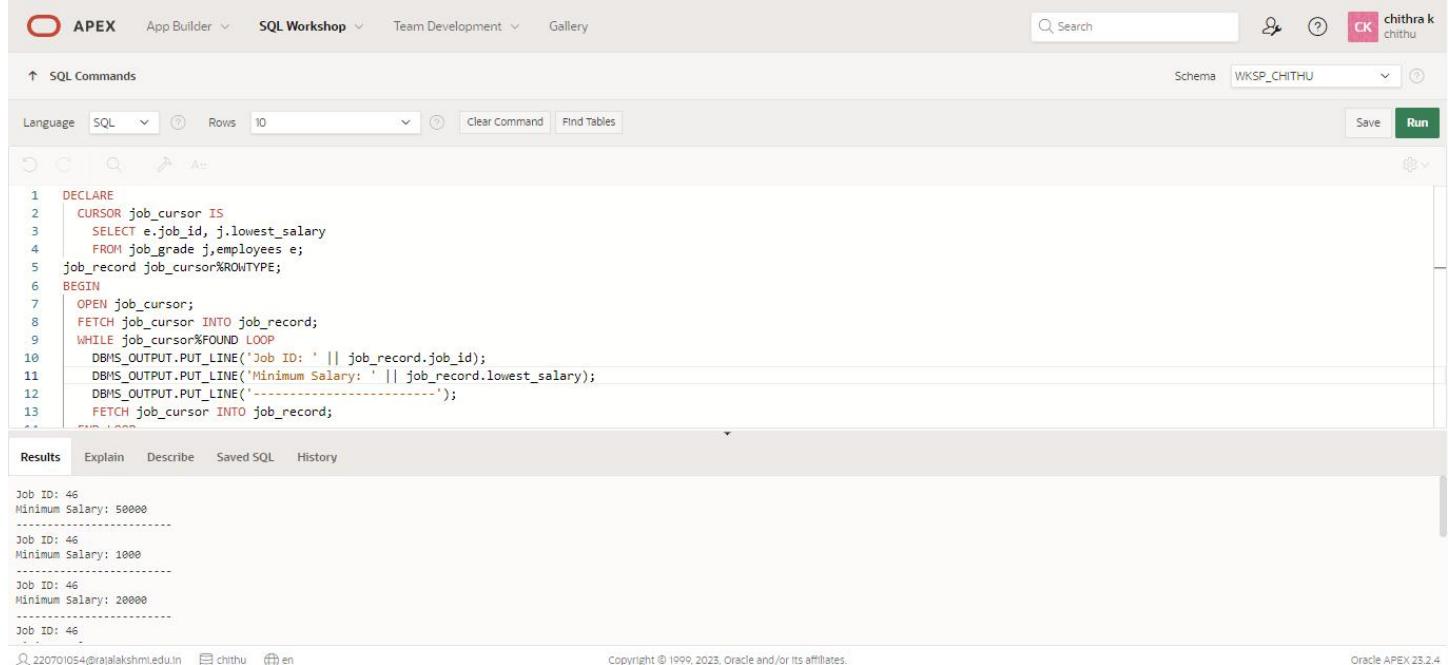
13.) Write a PL/SQL program to display the job IDs, titles, and minimum salaries of all jobs

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:



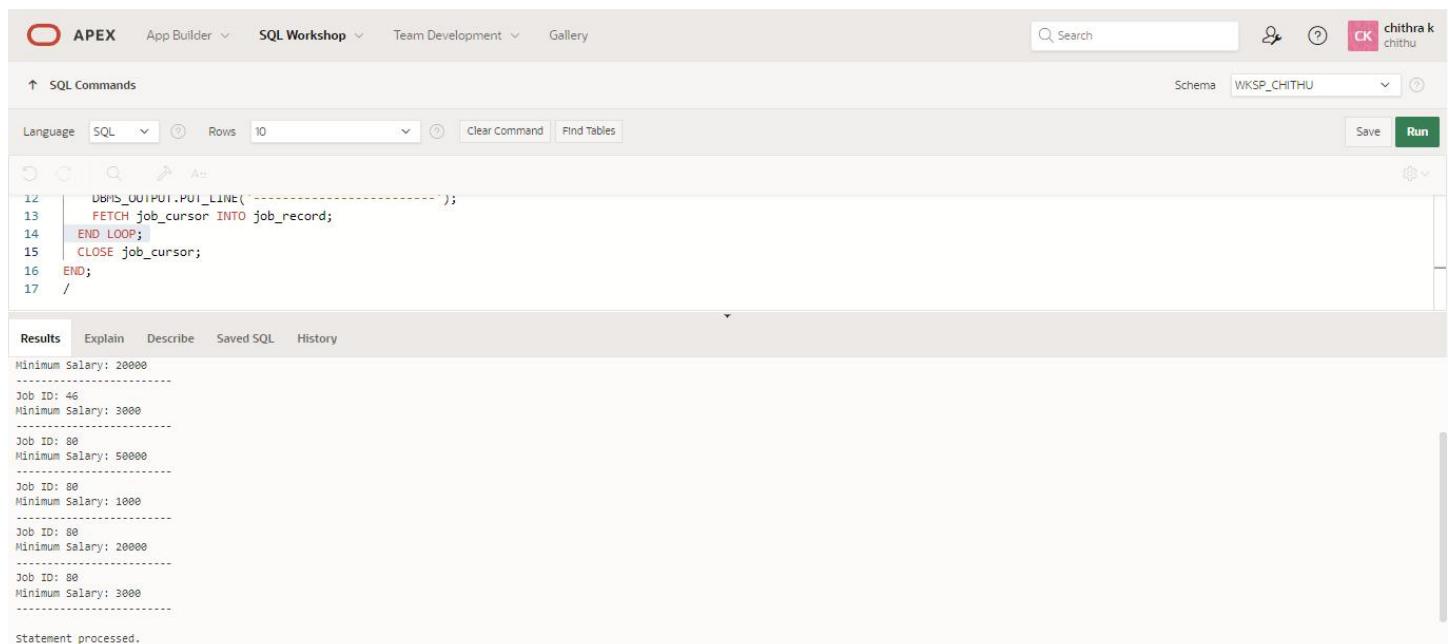
The screenshot shows the Oracle APEX interface with the SQL Workshop tab selected. The code in the editor is:

```
1  DECLARE
2      CURSOR job_cursor IS
3          SELECT e.job_id, j.lowest_salary
4          FROM job_grade j,employees e;
5      job_record job_cursor%ROWTYPE;
6  BEGIN
7      OPEN job_cursor;
8      FETCH job_cursor INTO job_record;
9      WHILE job_cursor%FOUND LOOP
10          DBMS_OUTPUT.PUT_LINE('Job ID: ' || job_record.job_id);
11          DBMS_OUTPUT.PUT_LINE('Minimum Salary: ' || job_record.lowest_salary);
12          DBMS_OUTPUT.PUT_LINE('-----');
13      FETCH job_cursor INTO job_record;
14  END LOOP;
15  CLOSE job_cursor;
16 END;
17 /
```

The results pane displays the output of the PL/SQL code:

```
Job ID: 46
Minimum Salary: 50000
-----
Job ID: 46
Minimum Salary: 1000
-----
Job ID: 46
Minimum Salary: 20000
-----
Job ID: 46
```

At the bottom, the status bar shows the user is chithu and the session ID is 220701054@rajalakshmi.edu.in. The copyright notice indicates Oracle APEX 23.2.4.



The screenshot shows the Oracle APEX interface with the SQL Workshop tab selected. The code in the editor is identical to the first output:

```
1  DECLARE
2      CURSOR job_cursor IS
3          SELECT e.job_id, j.lowest_salary
4          FROM job_grade j,employees e;
5      job_record job_cursor%ROWTYPE;
6  BEGIN
7      OPEN job_cursor;
8      FETCH job_cursor INTO job_record;
9      WHILE job_cursor%FOUND LOOP
10          DBMS_OUTPUT.PUT_LINE('-----');
11          DBMS_OUTPUT.PUT_LINE('Job ID: ' || job_record.job_id);
12          DBMS_OUTPUT.PUT_LINE('Minimum Salary: ' || job_record.lowest_salary);
13          DBMS_OUTPUT.PUT_LINE('-----');
14      FETCH job_cursor INTO job_record;
15  END LOOP;
16  CLOSE job_cursor;
17 END;
18 /
```

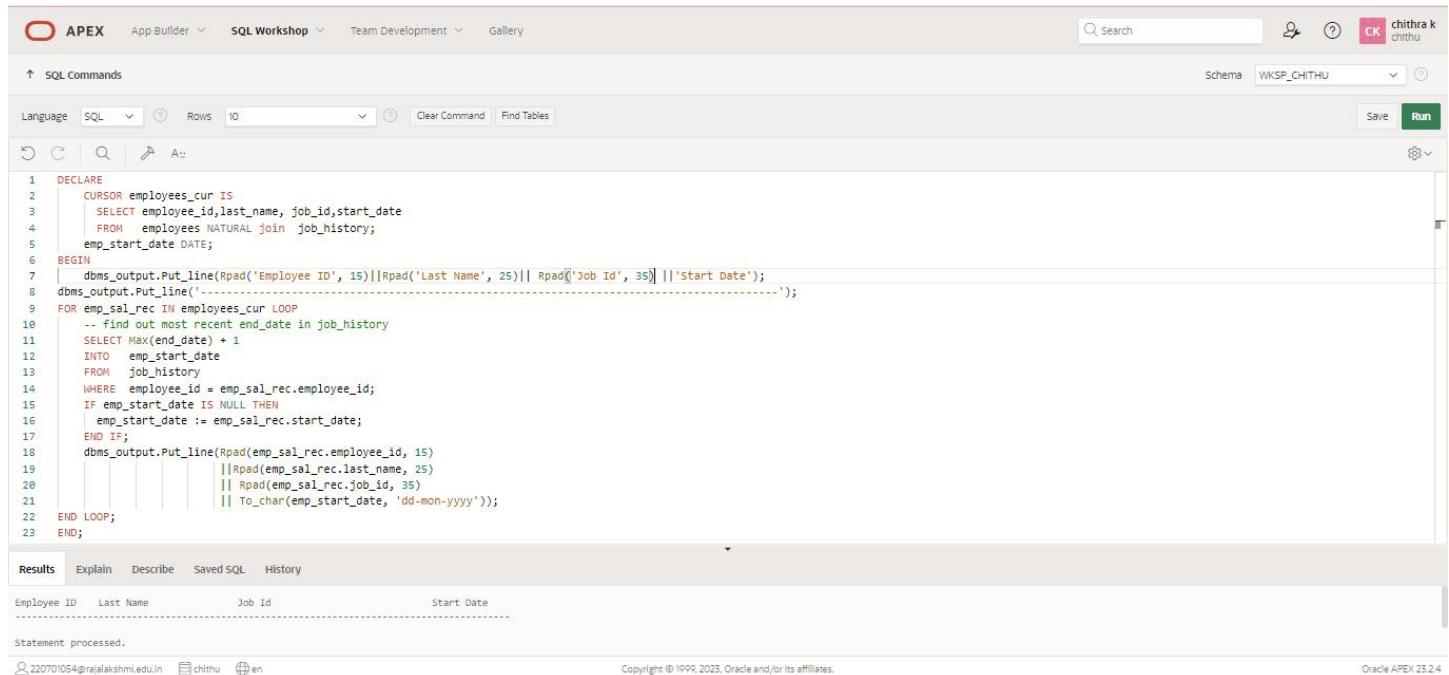
The results pane displays the output of the PL/SQL code:

```
Minimum Salary: 20000
-----
Job ID: 46
Minimum Salary: 3000
-----
Job ID: 80
Minimum Salary: 50000
-----
Job ID: 80
Minimum Salary: 1000
-----
Job ID: 80
Minimum Salary: 20000
-----
Job ID: 80
Minimum Salary: 3000
-----
Statement processed.
```

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

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

OUTPUT:



The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery. The right side of the header shows the user 'chithra k' and the schema 'WKSP_CHITHU'. The main workspace is titled 'SQL Commands' and contains the PL/SQL code from the previous block. Below the code, there are tabs for 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History'. The results tab displays the output of the query, which consists of four columns: Employee ID, Last Name, Job Id, and Start Date. The data is presented in a single row. At the bottom of the results table, it says 'Statement processed.' and shows the session details '220701054@rajalakshmi.edu.in chithru'. The footer of the page includes copyright information for Oracle and the text 'Oracle APEX 23.2.4'.

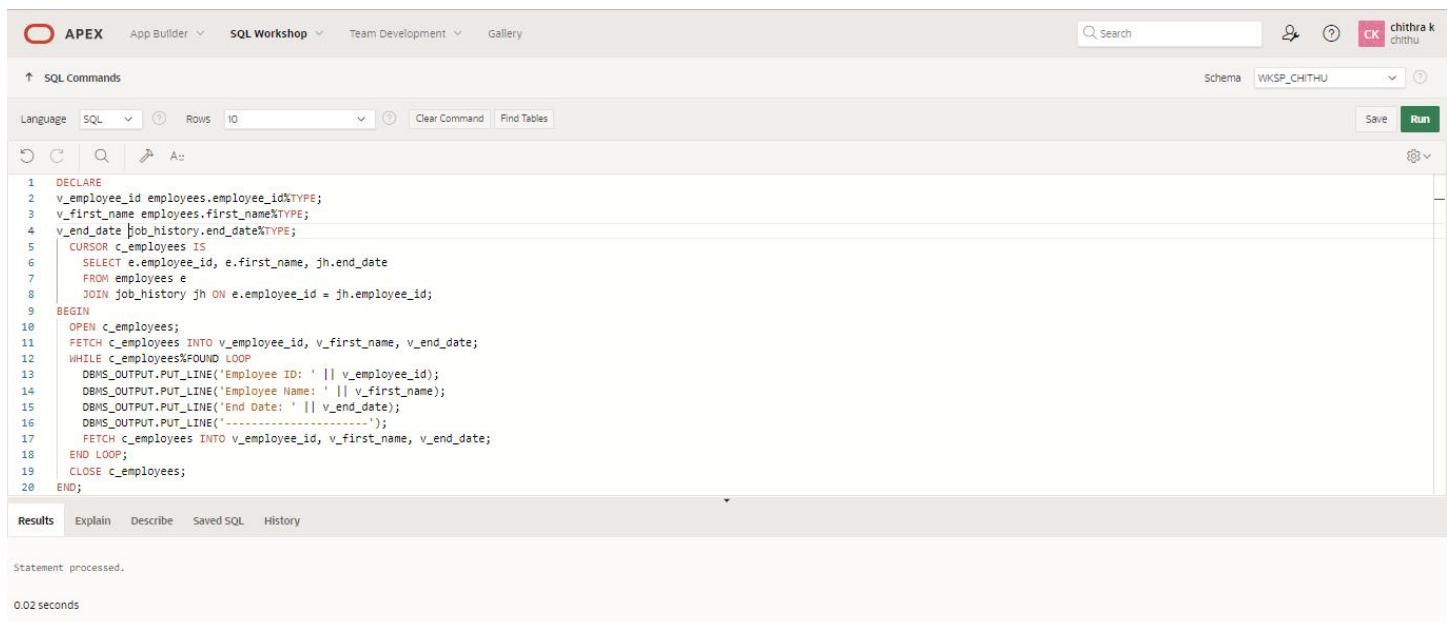
Employee ID	Last Name	Job Id	Start Date
101	King	CE001	01-JAN-1981

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

QUERY:

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

OUTPUT:



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. On the right, there's a search bar, a user icon for 'chithra k chithru', and a 'Run' button. The main workspace is titled 'SQL Commands'. It displays the PL/SQL code from the previous block. Below the code, there are tabs for 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History'. The status bar at the bottom indicates 'Statement processed.' and '0.02 seconds'.

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

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

RESULT:

Thus the queries for pl/sql control structures are executed successfully.

PROCEDURES AND FUNCTIONS

EX_NO: 17

DATE:18/05/2024

1.)Factorial of a number using function.

QUERY:

DECLARE

 fac NUMBER := 1;

 n NUMBER := :1;

BEGIN

 WHILE n > 0 LOOP

 fac := n * fac;

 n := n - 1;

 END LOOP;

 DBMS_OUTPUT.PUT_LINE(fac);

END;

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows a user profile for 'chithra k chithu'. The main area is a SQL editor with the following code:

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

The code is executed, and the results tab shows the output:

```
6
Statement processed.
```

Execution time: 0.01 seconds.

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

QUERY:

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

    p_title := p_title || ' - Retrieved';

    EXCEPTION
        WHEN NO_DATA_FOUND THEN
            p_title := NULL;
            p_author := NULL;
            p_year_published := NULL;
END;

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

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

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

OUTPUT:

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows a user profile for 'chithra k chithu'. 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 the PL/SQL code for creating a stored procedure named GET_BOOK_INFO. The code initializes output parameters p_title, p_author, p_publisher, p_year_published, and p_status, then fetches book information from the BOOKS table where BOOK_ID matches the input parameter p_book_id. It handles the NO_DATA_FOUND exception by setting p_status to 'NOT FOUND'. The results tab shows the message 'Procedure created.' and a execution time of '0.05 seconds'.

```
CREATE OR REPLACE PROCEDURE GET_BOOK_INFO (  
    p_book_id IN NUMBER,  
    p_title OUT VARCHAR2,  
    p_author OUT VARCHAR2,  
    p_publisher OUT VARCHAR2,  
    p_year_published OUT NUMBER,  
    p_status IN OUT VARCHAR2  
) AS  
BEGIN  
    -- Initialize the output parameters  
    p_title := NULL;  
    p_author := NULL;  
    p_publisher := NULL;  
    p_year_published := NULL;  
    p_status := 'NOT FOUND';  
    -- Fetch the book information  
    SELECT TITLE, AUTHOR, PUBLISHER, YEAR_PUBLISHED  
    INTO p_title, p_author, p_publisher, p_year_published  
    FROM BOOKS  
    WHERE BOOK_ID = p_book_id;  
    p_status := 'FOUND';  
EXCEPTION  
    WHEN NO_DATA_FOUND THEN  
        -- If no book is found, the OUT parameters remain NULL and status is 'NOT FOUND'  
        p_status := "NOT FOUND";  
END;  
/
```

Procedure created.
0.05 seconds

The screenshot shows the Oracle SQL Workshop interface, identical to the previous one but with a different code block. The code is identical to the one in the first screenshot, but it is enclosed in a BEGIN...END block. The results tab shows the message 'Procedure created.' and a execution time of '0.05 seconds'.

```
-- Fetch the book information  
BEGIN  
    SELECT TITLE, AUTHOR, PUBLISHER, YEAR_PUBLISHED  
    INTO p_title, p_author, p_publisher, p_year_published  
    FROM BOOKS  
    WHERE BOOK_ID = p_book_id;  
    p_status := 'FOUND';  
EXCEPTION  
    WHEN NO_DATA_FOUND THEN  
        -- If no book is found, the OUT parameters remain NULL and status is 'NOT FOUND'  
        p_status := "NOT FOUND";  
END;  
/
```

Procedure created.
0.05 seconds

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

RESULT:

Thus the queries for procedures and functions are executed successfully.

TRIGGER

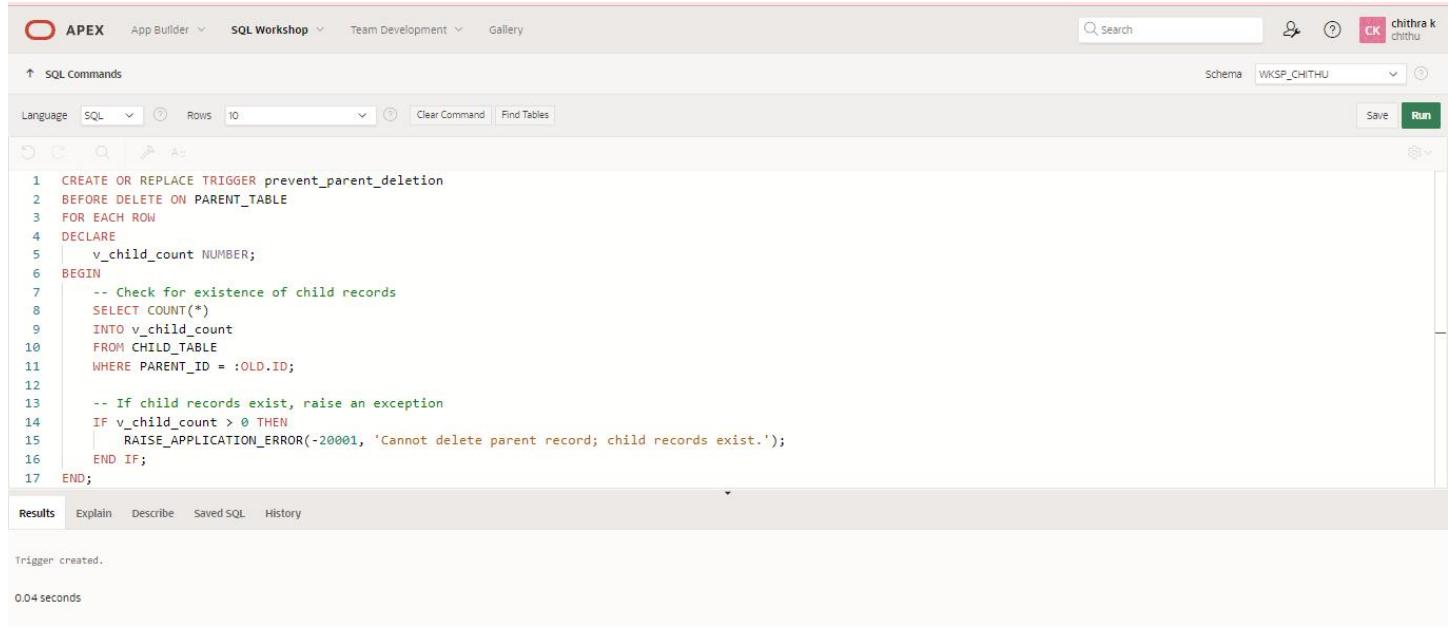
EX_NO: 18

DATE: 18/05/2024

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

```
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. The main area displays the SQL command for creating the trigger:

```
1 CREATE OR REPLACE TRIGGER prevent_parent_deletion
2 BEFORE DELETE ON PARENT_TABLE
3 FOR EACH ROW
4 DECLARE
5     v_child_count NUMBER;
6 BEGIN
7     -- Check for existence of child records
8     SELECT COUNT(*)
9     INTO v_child_count
10    FROM CHILD_TABLE
11   WHERE PARENT_ID = :OLD.ID;
12
13   -- If child records exist, raise an exception
14   IF v_child_count > 0 THEN
15       RAISE_APPLICATION_ERROR(-20001, 'Cannot delete parent record; child records exist.');
16   END IF;
17 END;
```

The 'Results' tab at the bottom shows the output: "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. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows a user profile for 'chithra k chithu'. The main area is titled 'SQL Commands' with a schema dropdown set to 'WKSP_CHITHU'. The code editor contains the following PL/SQL trigger creation script:

```
1 CREATE OR REPLACE TRIGGER check_duplicate_email
2 BEFORE INSERT OR UPDATE ON EMPLOYEES
3 FOR EACH ROW
4 DECLARE
5     v_count NUMBER;
6 BEGIN
7     -- Check for duplicate email on insert
8     IF INSERTING THEN
9         SELECT COUNT(*)
10            INTO v_count
11           FROM EMPLOYEES
12          WHERE EMAIL = :NEW.EMAIL;
13
14     IF v_count > 0 THEN
15         RAISE_APPLICATION_ERROR(-20002, 'Duplicate email found: ' || :NEW.EMAIL);
16     END IF;
17
18     -- Check for duplicate email on update, excluding the current record
19     ELSIF UPDATING THEN
20         SELECT COUNT(*)
21            INTO v_count
22               FROM EMPLOYEES
23              WHERE EMAIL = :NEW.EMAIL
24              AND EMPLOYEE_ID != :OLD.EMPLOYEE_ID;
25
26     IF v_count > 0 THEN
27         RAISE_APPLICATION_ERROR(-20002, 'Duplicate email found: ' || :NEW.EMAIL);
28     END IF;
29
30 END;
```

The 'Results' tab is selected, showing the output: 'Trigger created.' and '0.03 seconds' execution time.

The screenshot shows the Oracle SQL Workshop interface with the same setup as the first one. The code editor now contains a modified version of the trigger script, specifically addressing the update logic:

```
14 IF v_count > 0 THEN
15     RAISE_APPLICATION_ERROR(-20002, 'Duplicate email found: ' || :NEW.EMAIL);
16 END IF;
17
18 -- Check for duplicate email on update, excluding the current record
19 ELSIF UPDATING THEN
20     SELECT COUNT(*)
21        INTO v_count
22           FROM EMPLOYEES
23          WHERE EMAIL = :NEW.EMAIL
24          AND EMPLOYEE_ID != :OLD.EMPLOYEE_ID;
25
26     IF v_count > 0 THEN
27         RAISE_APPLICATION_ERROR(-20002, 'Duplicate email found: ' || :NEW.EMAIL);
28     END IF;
29
30 END;
```

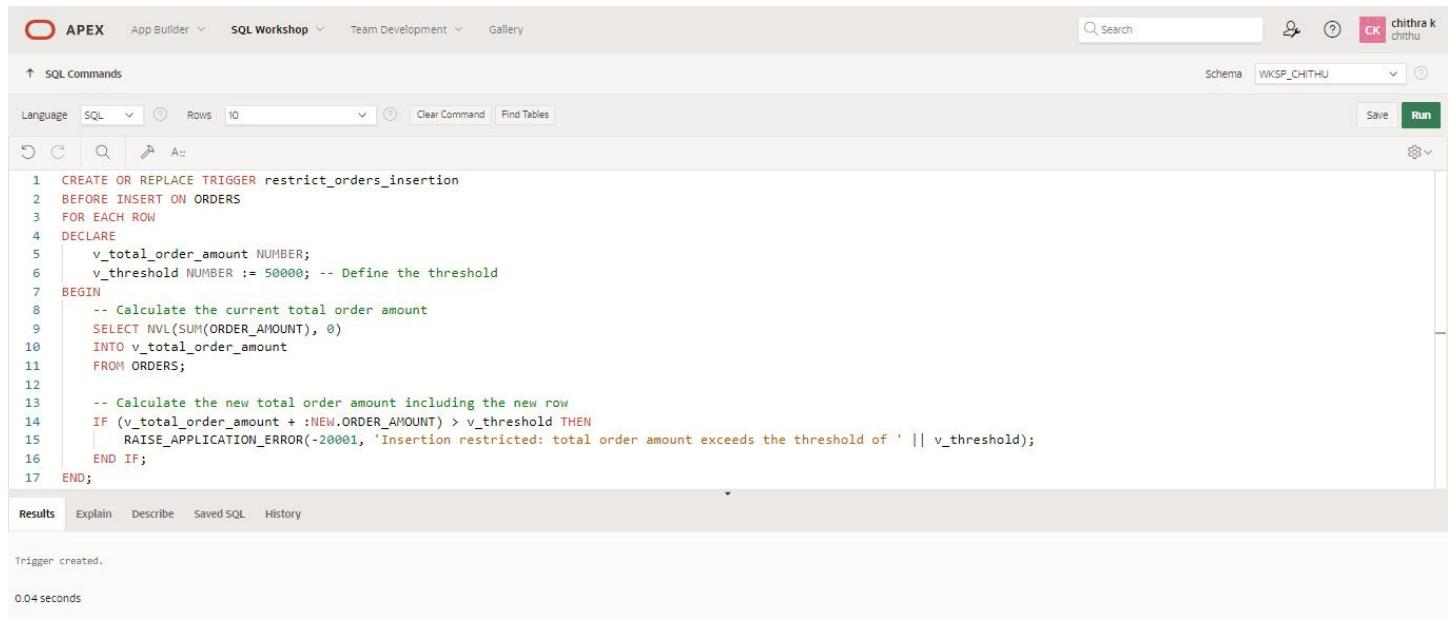
The 'Results' tab is selected, showing the output: 'Trigger created.' and '0.03 seconds' execution time.

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. In the top navigation bar, 'APEX' is selected. The main area displays the SQL command for creating a trigger:

```
1 CREATE OR REPLACE TRIGGER restrict_orders_insertion
2 BEFORE INSERT ON ORDERS
3 FOR EACH ROW
4 DECLARE
5     v_total_order_amount NUMBER;
6     v_threshold NUMBER := 50000; -- Define the threshold
7 BEGIN
8     -- Calculate the current total order amount
9     SELECT NVL(SUM(ORDER_AMOUNT), 0)
10    INTO v_total_order_amount
11   FROM ORDERS;
12
13     -- Calculate the new total order amount including the new row
14     IF (v_total_order_amount + :NEW.ORDER_AMOUNT) > v_threshold THEN
15         RAISE_APPLICATION_ERROR(-20001, 'Insertion restricted: total order amount exceeds the threshold of ' || v_threshold);
16     END IF;
17 END;
```

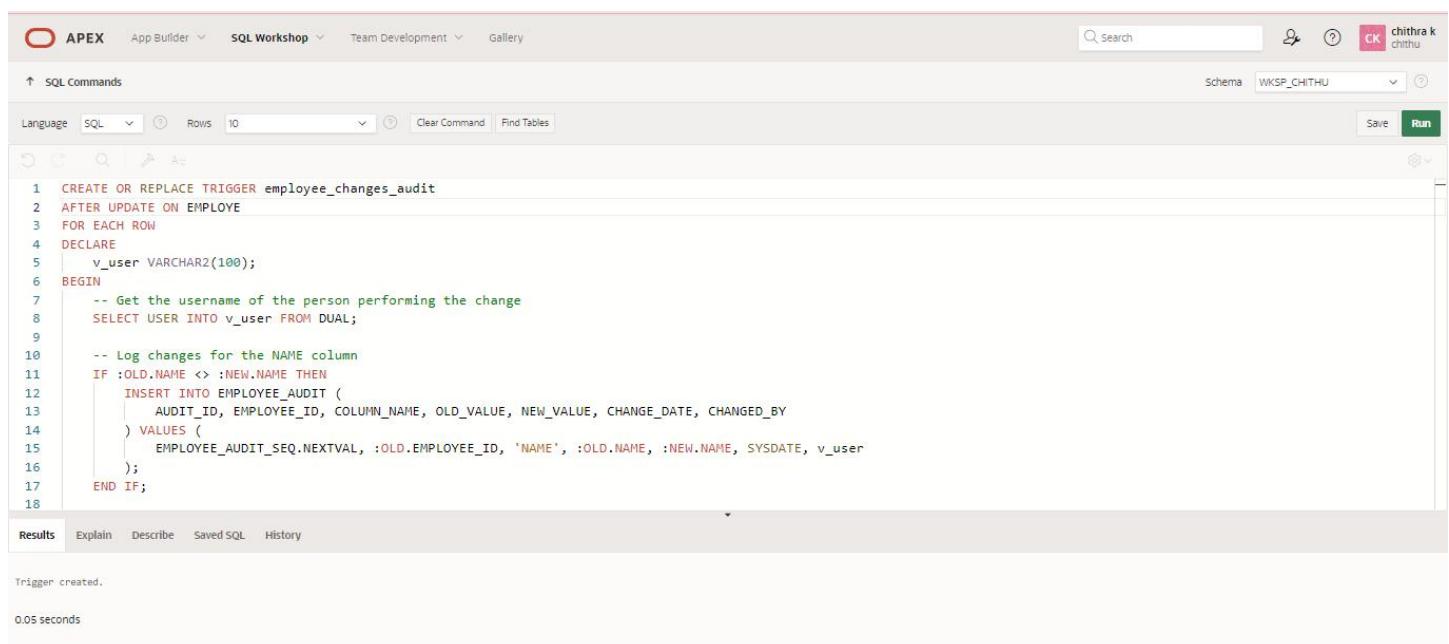
The 'Results' tab at the bottom shows the output: "Trigger created." and "0.04 seconds".

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. In the top navigation bar, 'APEX' is selected, followed by 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery'. On the right side, there's a user profile for 'chithra k chithu'. The main area is titled 'SQL Commands'. The schema dropdown is set to 'WKSP_CHITHU'. Below the title, there are buttons for 'Language' (set to 'SQL'), 'Rows' (set to 10), and 'Run'. There are also 'Clear Command' and 'Find Tables' buttons. The SQL command window contains the code for the 'log_changes' trigger. The code is numbered from 1 to 18. Lines 1 through 17 are visible, while line 18 is partially cut off at the bottom. The code creates a trigger on the 'main_table' that logs changes to specific columns into an 'audit_table'. It uses bind variables like :OLD and :NEW, and the audit sequence 'audit_seq'. The trigger body includes a SELECT statement to get the current user performing the change and an INSERT statement into the audit table. The 'Results' tab is selected at the bottom, showing the message 'Trigger created.' and '0.05 seconds' execution time.

```
1 CREATE OR REPLACE TRIGGER employee_changes_audit
2 AFTER UPDATE ON EMPLOYEE
3 FOR EACH ROW
4 DECLARE
5     v_user VARCHAR2(100);
6 BEGIN
7     -- Get the username of the person performing the change
8     SELECT USER INTO v_user FROM DUAL;
9
10    -- Log changes for the NAME column
11    IF :OLD.NAME <> :NEW.NAME THEN
12        INSERT INTO EMPLOYEE_AUDIT (
13            AUDIT_ID, EMPLOYEE_ID, COLUMN_NAME, OLD_VALUE, NEW_VALUE, CHANGE_DATE, CHANGED_BY
14        ) VALUES (
15            EMPLOYEE_AUDIT_SEQ.NEXTVAL, :OLD.EMPLOYEE_ID, 'NAME', :OLD.NAME, :NEW.NAME, SYSDATE, v_user
16        );
17    END IF;
18
```

APEX App Builder SQL Workshop Team Development Gallery

SQL Commands

Language SQL Rows 10 Clear Command Find Tables

Schema WKSP_CHITHU Save Run

```
-- Log changes for the DEPARTMENT column
19 IF :OLD.DEPARTMENT <> :NEW.DEPARTMENT THEN
20   INSERT INTO EMPLOYEE_AUDIT (
21     AUDIT_ID, EMPLOYEE_ID, COLUMN_NAME, OLD_VALUE, NEW_VALUE, CHANGE_DATE, CHANGED_BY
22   ) VALUES (
23     EMPLOYEE_AUDIT_SEQ.NEXTVAL, :OLD.EMPLOYEE_ID, 'DEPARTMENT', :OLD.DEPARTMENT, :NEW.DEPARTMENT, SYSDATE, v_user
24   );
25 END IF;
26
27
28 -- Log changes for the SALARY column
29 IF :OLD.SALARY <> :NEW.SALARY THEN
30   INSERT INTO EMPLOYEE_AUDIT (
31     AUDIT_ID, EMPLOYEE_ID, COLUMN_NAME, OLD_VALUE, NEW_VALUE, CHANGE_DATE, CHANGED_BY
32   ) VALUES (
33     EMPLOYEE_AUDIT_SEQ.NEXTVAL, :OLD.EMPLOYEE_ID, 'SALARY', TO_CHAR(:OLD.SALARY), TO_CHAR(:NEW.SALARY), SYSDATE, v_user
34   );
35 END IF;
36
```

Results Explain Describe Saved SQL History

Trigger created.

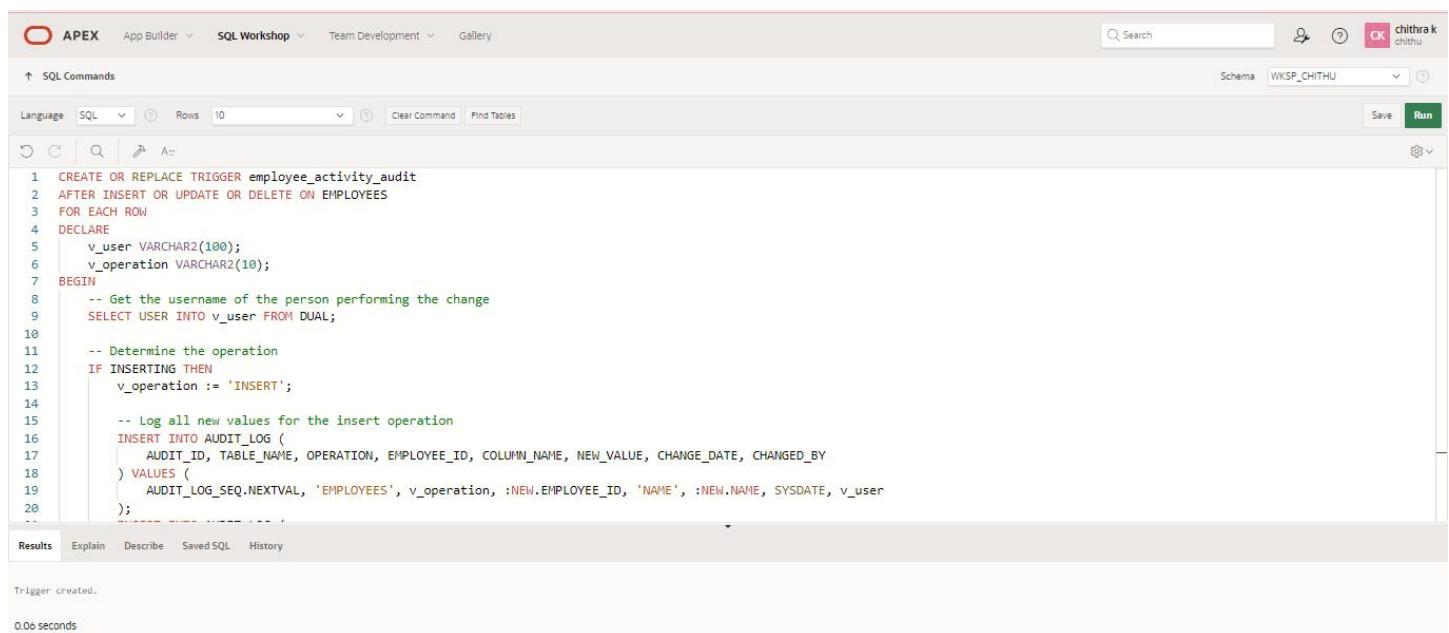
0.05 seconds

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's a user icon for 'chithra k' and a schema dropdown for 'WKSP_CHITHU'. The main workspace displays the PL/SQL code for the trigger. Below the code, the 'Results' tab is selected, showing the output: 'Trigger created.' and '0.06 seconds'. The bottom of the window has tabs for Explain, Describe, Saved SQL, and History.

```
1 CREATE OR REPLACE TRIGGER employee_activity_audit
2 AFTER INSERT OR UPDATE OR DELETE ON EMPLOYEES
3 FOR EACH ROW
4 DECLARE
5   v_user VARCHAR2(100);
6   v_operation VARCHAR2(10);
7 BEGIN
8   -- Get the username of the person performing the change
9   SELECT USER INTO v_user FROM DUAL;
10
11  -- Determine the operation
12  IF INSERTING THEN
13    v_operation := 'INSERT';
14
15  -- Log all new values for the insert operation
16  INSERT INTO AUDIT_LOG (
17    AUDIT_ID, TABLE_NAME, OPERATION, EMPLOYEE_ID, COLUMN_NAME, NEW_VALUE, CHANGE_DATE, CHANGED_BY
18  ) VALUES (
19    AUDIT_LOG_SEQ.NEXTVAL, 'EMPLOYEES', v_operation, :NEW.EMPLOYEE_ID, 'NAME', :NEW.NAME, SYSDATE, v_user
20  );

```

APEX App Builder SQL Workshop Team Development Gallery

↑ SQL Commands

Language SQL Rows 10 Clear Command Find Tables

Schema WKSP_CHITHU Save Run

```
42
43
44    -- Log changes for the DEPARTMENT column
45    IF :OLD.DEPARTMENT <> :NEW.DEPARTMENT THEN
46        INSERT INTO AUDIT_LOG (
47            |   AUDIT_ID, TABLE_NAME, OPERATION, EMPLOYEE_ID, COLUMN_NAME, OLD_VALUE, NEW_VALUE, CHANGE_DATE, CHANGED_BY
48        ) VALUES (
49            |   AUDIT_LOG_SEQ.NEXTVAL, 'EMPLOYEES', v_operation, :OLD.EMPLOYEE_ID, 'DEPARTMENT', :OLD.DEPARTMENT, :NEW.DEPARTMENT, SYSDATE, v_user
50        );
51    END IF;
52
53    -- Log changes for the SALARY column
54    IF :OLD.SALARY <> :NEW.SALARY THEN
55        INSERT INTO AUDIT_LOG (
56            |   AUDIT_ID, TABLE_NAME, OPERATION, EMPLOYEE_ID, COLUMN_NAME, OLD_VALUE, NEW_VALUE, CHANGE_DATE, CHANGED_BY
57        ) VALUES (
58            |   AUDIT_LOG_SEQ.NEXTVAL, 'EMPLOYEES', v_operation, :OLD.EMPLOYEE_ID, 'SALARY', TO_CHAR(:OLD.SALARY), TO_CHAR(:NEW.SALARY), SYSDATE, v_user
59        );
60    END IF;
61
```

Results Explain Describe Saved SQL History

Trigger created.

0.06 seconds

APEX App Builder SQL Workshop Team Development Gallery

↑ SQL Commands

Language SQL Rows 10 Clear Command Find Tables

Schema WKSP_CHITHU Save Run

```
63
64
65    v_operation := 'DELETE';
66
67    -- Log all old values for the delete operation
68    INSERT INTO AUDIT_LOG (
69        |   AUDIT_ID, TABLE_NAME, OPERATION, EMPLOYEE_ID, COLUMN_NAME, OLD_VALUE, CHANGE_DATE, CHANGED_BY
70    ) VALUES (
71        |   AUDIT_LOG_SEQ.NEXTVAL, 'EMPLOYEES', v_operation, :OLD.EMPLOYEE_ID, 'NAME', :OLD.NAME, SYSDATE, v_user
72    );
73    INSERT INTO AUDIT_LOG (
74        |   AUDIT_ID, TABLE_NAME, OPERATION, EMPLOYEE_ID, COLUMN_NAME, OLD_VALUE, CHANGE_DATE, CHANGED_BY
75    ) VALUES (
76        |   AUDIT_LOG_SEQ.NEXTVAL, 'EMPLOYEES', v_operation, :OLD.EMPLOYEE_ID, 'DEPARTMENT', :OLD.DEPARTMENT, SYSDATE, v_user
77    );
78    INSERT INTO AUDIT_LOG (
79        |   AUDIT_ID, TABLE_NAME, OPERATION, EMPLOYEE_ID, COLUMN_NAME, OLD_VALUE, CHANGE_DATE, CHANGED_BY
80    ) VALUES (
81        |   AUDIT_LOG_SEQ.NEXTVAL, 'EMPLOYEES', v_operation, :OLD.EMPLOYEE_ID, 'SALARY', TO_CHAR(:OLD.SALARY), SYSDATE, v_user
82    );
83    END IF;
84
```

Results Explain Describe Saved SQL History

Trigger created.

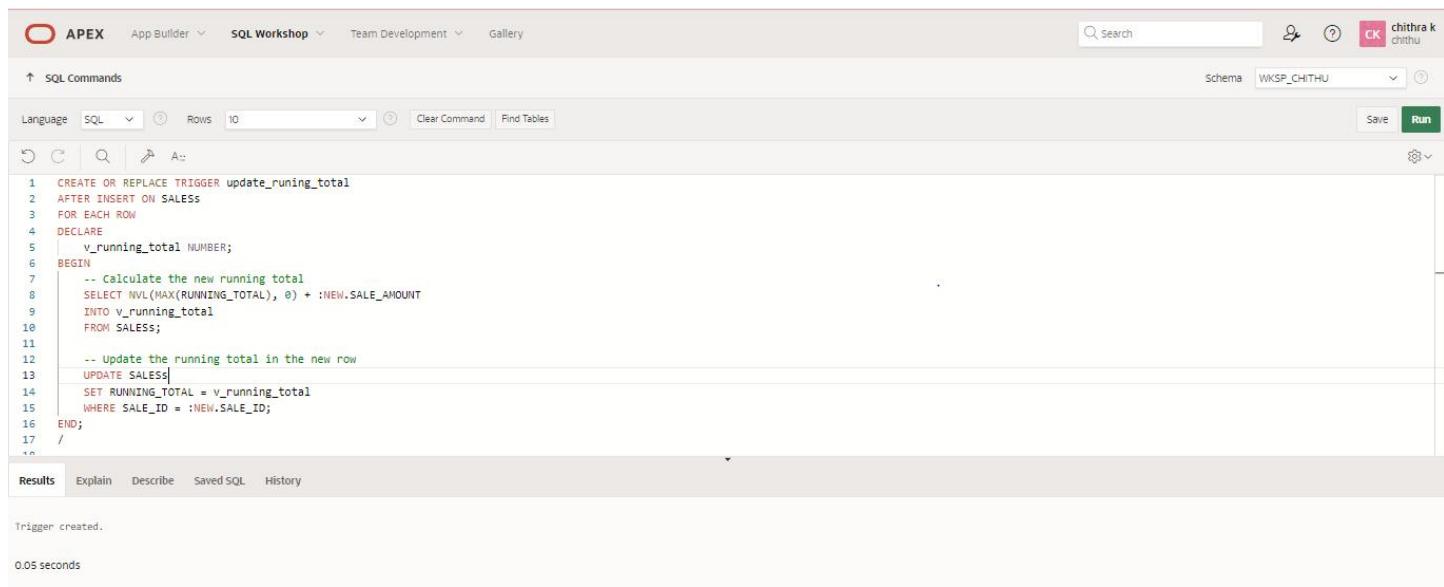
0.06 seconds

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, Team Development, and Gallery. The user is signed in as 'chithra k chithu'. The main area is titled 'SQL Commands' with tabs for Language (set to SQL), Rows (set to 10), and various buttons like Clear Command and Find Tables. The schema is set to 'WKSP_CHITHU'. The SQL editor contains the PL/SQL code for the trigger, which is then executed. The results tab shows the message 'Trigger created.' and a execution time of '0.05 seconds'.

```
1 CREATE OR REPLACE TRIGGER update_running_total
2 AFTER INSERT ON SALESS
3 FOR EACH ROW
4 DECLARE
5     v_running_total NUMBER;
6 BEGIN
7     -- Calculate the new running total
8     SELECT NVL(MAX(RUNNING_TOTAL), 0) + :NEW.SALE_AMOUNT
9     INTO v_running_total
10    FROM SALESS;
11
12    -- Update the running total in the new row
13    UPDATE SALESS|
14    SET RUNNING_TOTAL = v_running_total
15    WHERE SALE_ID = :NEW.SALE_ID;
16 END;
17 /
```

Results Explain Describe Saved SQL History

Trigger created.
0.05 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;

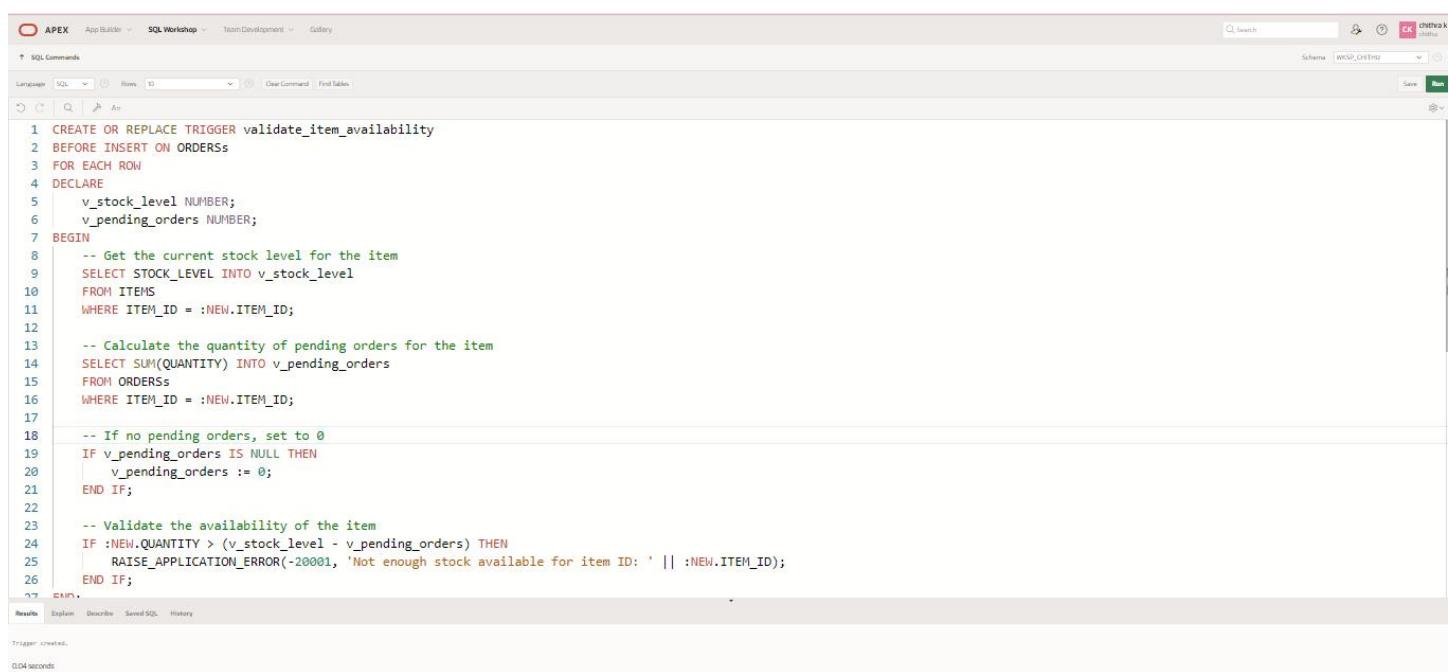
EXCEPTION

WHEN insufficient_stock THEN

RAISE_APPLICATION_ERROR(-20004, 'Insufficient stock for the item.');

END;

OUTPUT:



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

- Toolbar:** APEX, App Builder, SQL Workshop, Team Development, Gallery.
- Language:** SQL.
- Schema:** WISD_CHTU.
- Code Area:** The code for the trigger is pasted here. It includes declarations for stock_level and pending_orders, a BEGIN block with a SELECT statement to get current stock level, a calculation of pending orders, and a validation step where it checks if the new quantity is greater than the available stock (stock_level - pending_orders). If so, it raises an application error.
- Status Bar:** Trigger created. 0.04 seconds.

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

RESULT:

Thus the queries for triggers are executed successfully.

MONGO DB

EX_NO: 19

DATE:23/05/2024

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:

```
chithra_54> db.restaurants.find( { $or: [ { name: /^Wil/ }, { cuisine: { $nin: ['American', 'Chinese'] } } ] } , { restaurant_id: 1, name: 1, borough: 1, cuisine: 1 } )
[
  {
    _id: ObjectId('564c2d949eb21ad392f1d6de'),
    borough: 'Manhattan',
    cuisine: 'Other',
    name: '',
    restaurant_id: '50017887'
  },
  {
    _id: ObjectId('564c2d949eb21ad392f1d6ec'),
    borough: 'Brooklyn',
    cuisine: 'Other',
    name: '',
    restaurant_id: '50017910'
  },
  {
    _id: ObjectId('564c2d949eb21ad392f1d6ed'),
    borough: 'Manhattan',
    cuisine: 'Other',
    name: '',
    restaurant_id: '50017912'
  },
  {
    _id: ObjectId('564c2d949eb21ad392f1d6f5'),
    borough: 'Brooklyn',
    cuisine: 'Other',
    name: '',
    restaurant_id: '50017925'
  }
]
chithra_54>
```

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:
```

```
chithra_54> db.restaurants.find( { grades: { $elemMatch: { grade: "A", score: 11, date: ISODate("2014-08-11T00:00:00Z") } } }, { restaurant_id: 1, name: 1, grades: 1 } );
chithra_54>
```

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

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

OUTPUT:

```
chithra_54> 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 } )
```

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:

```
chithra_54> db.restaurants.find({$and : [{"address.coord.1": {$gt : 42}}, {"address.coord.1": {$lte : 52}}]}, { _id:0, restaurant_id:1, name:1, address:1 })
```

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:

```
chithra_54> db.restaurants.find( {}, { _id: 0 } ).sort( { name: 1 } );
[
  {
    address: {
      building: '154',
      coord: [ -73.9189064, 40.8654529 ],
      street: 'Post Ave',
      zipcode: '10034'
    },
    borough: 'Manhattan',
    cuisine: 'Other',
    grades: [],
    name: '',
    restaurant_id: '50017887'
  },
  {
    address: {
      building: '508',
      coord: [ -73.999813, 40.683876 ],
      street: 'Henry St',
      zipcode: '11231'
    },
    borough: 'Brooklyn',
    cuisine: 'Other',
    grades: [],
    name: '',
    restaurant_id: '50017910'
  }
]
```

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

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

```
chithra_54> db.restaurants.find({}, {_id: 0 }).sort({ name: -1 })
[
  {
    address: {
      building: '154',
      coord: [ -73.9189064, 40.8654529 ],
      street: 'Post Ave',
      zipcode: '10034'
    },
    borough: 'Manhattan',
    cuisine: 'Other',
    grades: [],
    name: '',
    restaurant_id: '50017887'
  },
  {
    address: {
      building: '508',
      coord: [ -73.999813, 40.683876 ],
      street: 'Henry St',
      zipcode: '11231'
    },
    ...]
```

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.

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

OUTPUT:

```
chithra_54> db.restaurants.find({}, {_id: 0 }).sort({ cuisine: 1, borough: -1 })
[
  {
    address: {
      building: '154',
      coord: [ -73.9189064, 40.8654529 ],
      street: 'Post Ave',
      zipcode: '10034'
    },
    borough: 'Manhattan',
    cuisine: 'Other',
    grades: [],
    name: '',
    restaurant_id: '50017887'
  },
  {
    address: {
      building: '15',
      coord: [ -73.9966882, 40.7139264 ],
      street: 'Division St',
      zipcode: '10002'
    },
    ...]
```

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

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

OUTPUT:

```
chithra_54> db.restaurants.find({ "address.street": { $exists: true, $ne: "" } })
[
  {
    _id: ObjectId('564c2d949eb21ad392f1d6de'),
    address: {
      building: '154',
      coord: [ -73.9189064, 40.8654529 ],
      street: 'Post Ave',
      zipcode: '10034'
    },
    borough: 'Manhattan',
    cuisine: 'Other',
    grades: [],
    name: '',
    restaurant_id: '50017887'
  },
  {
    _id: ObjectId('564c2d949eb21ad392f1d6ec'),
    address: {
      building: '508',
      coord: [ -73.999813, 40.683876 ],
      street: 'Henry St',
      zipcode: '11231'
    },
  }
]
```

9.) Write a MongoDB query which will select all documents in the restaurants collection where the coord field value is Double.

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

OUTPUT:

```
chithra_54> db.restaurants.find({ "address.coord": { $elemMatch: { $type: "double" } } })
[
  {
    _id: ObjectId('564c2d949eb21ad392f1d6de'),
    address: {
      building: '154',
      coord: [ -73.9189064, 40.8654529 ],
      street: 'Post Ave',
      zipcode: '10034'
    },
    borough: 'Manhattan',
    cuisine: 'Other',
    grades: [],
    name: '',
    restaurant_id: '50017887'
  },
  {
    _id: ObjectId('564c2d949eb21ad392f1d6ec'),
    address: {
      building: '508',
      coord: [ -73.999813, 40.683876 ],
      street: 'Henry St',
      zipcode: '11231'
    },
  }
]
```

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.

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

OUTPUT:

```
chithra_54> db.restaurants.find({ "grades.score": { $mod: [7, 0] } }, { restaurant_id: 1, name: 1, grades: 1 });
chithra_54>
```

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:

```
chithra_54> db.restaurants.find({ name: /mon/i }, { name: 1, borough: 1, "address.coord": 1, cuisine: 1 });
chithra_54>
```

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 })
```

```
chithra_54> db.restaurants.find({ name: /^Mad/i }, { name: 1, borough: 1, "address.coord": 1, cuisine: 1 });
chithra_54>
```

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 } } } })

```
chithra_54> db.restaurants.find({ "grades": { $elemMatch: { "score": { $lt: 5 } } } });  
chithra_54>
```

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" })
```

```
chithra_54> db.restaurants.find({ "grades": { $elemMatch: { "score": { $lt: 5 } } }, "borough": "Manhattan" });  
chithra_54>
```

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:

```
chithra_54> db.restaurants.find({ "grades": { $elemMatch: { "score": { $lt: 5 } } }, $or: [{ "borough": "Manhattan" }, { "borough": "Brooklyn" }] })  
chithra_54>
```

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:

```
chithra_54> db.restaurants.find({ "grades": { $elemMatch: { "score": { $lt: 5 } } }, $or: [{ "borough": "Manhattan" }, { "borough": "Brooklyn" }], "cuisine": { $ne: "American" } })  
chithra_54>
```

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:

```
chithra_54> db.restaurants.find({ "grades": { $elemMatch: { "score": { $lt: 5 } } }, $or: [{ "borough": "Manhattan" }, { "borough": "Brooklyn" }], "cuisine": { $nin: ["American", "Chinese"] } })  
chithra_54>
```

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:

```
chithra_54> db.restaurants.find({ $and: [{ "grades.grade": "A", "grades.score": 2 }, { "grades.grade": "A", "grades.score": 6 }] })  
chithra_54>
```

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:

```
chithra_54> db.restaurants.find({ $and: [{ "grades.grade": "A", "grades.score": 2 }, { "grades.grade": "A", "grades.score": 6 }], "borough": "Manhattan" })
chithra_54>
```

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:

```
chithra_54> db.restaurants.find({ $and: [{ "grades.grade": "A", "grades.score": 2 }, { "grades.grade": "A", "grades.score": 6 }], $or: [{ "borough": "Manhattan" }, { "borough": "Brooklyn" }] })
chithra_54>
```

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" } })
```

```
chithra_54> db.restaurants.find({ $and: [{ "grades.grade": "A", "grades.score": 2 }, { "grades.grade": "A", "grades.score": 6 }], $or: [{ "borough": "Manhattan" }, { "borough": "Brooklyn" }], "cuisine": { $ne: "American" } })
chithra_54>
```

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"] } })

```
chithra_54> 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"] } })
```

chithra_54>

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 }] })
```

```
chithra_54> db.restaurants.find({ $or: [{ "grades.score": 2 }, { "grades.score": 6 }] })
```

chithra_54>

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

RESULT:

Thus the mongo db queries for restaurants collections are executed successfully.

MONGO DB

EX_NO: 20

DATE:23/05/2024

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

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

```
chithra_54> db.movies.find({ year: 1893 });
[ {
  _id: ObjectId('573a1390f29313caabcd4135'),
  plot: 'Three men hammer on an anvil and pass a bottle of beer around.',
  genres: [ 'Short' ],
  runtime: 1,
  cast: [ 'Charles Kayser', 'John Ott' ],
  num_mflix_comments: 1,
  title: 'Blacksmith Scene',
  fullplot: 'A stationary camera looks at a large anvil with a blacksmith behind it and one on either side. The smith in the middle draws a heated metal rod from the fire, places it on the anvil, and all three begin a rhythmic hammering. After several blows, the metal goes back in the fire. One smith pulls out a bottle of beer, and they each take a swig. Then, out comes the glowing metal and the hammering resumes.',
  countries: [ 'USA' ],
  released: ISODate('1893-05-09T00:00:00.000Z'),
  directors: [ 'William K.L. Dickson' ],
  rated: 'UNRATED',
  awards: { wins: 1, nominations: 0, text: '1 win.' },
  lastupdated: '2015-08-26 00:03:50.133000000',
  year: 1893,
  imdb: { rating: 6.2, votes: 1189, id: 5 },
  type: 'movie',
  tomatoes: {
    viewer: { rating: 3, numReviews: 184, meter: 32 },
    lastUpdated: ISODate('2015-06-28T18:34:09.000Z')
  }
}
]
chithra_54>
```

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 } })

```
chithra_54> db.movies.find({ runtime: { $gt: 120 } });
[ {
  _id: ObjectId('573a1390f22313caabcd5967'),
  plot: 'An intrepid reporter and his loyal friend battle a bizarre secret society of criminals known as The Vampires.',
  genres: [ 'Action', 'Adventure', 'Crime' ],
  runtime: 399,
  rated: 'NOT RATED',
  cast: [ 'Musidora', 'ÉdouardMathè', 'Marcel Lèvesque', 'Jean Aymé' ],
  poster: 'https://m.media-amazon.com/images/M/MVSBMTc1NTY3NDIzN15BM15BanBnXkFtZTgwNTIyODg5MTE@._V1_SY1000_SX677_AL_.jpg',
  title: 'Les vampires',
  fullplot: 'An intrepid reporter and his loyal friend battle a bizarre secret society of criminals known as The Vampires.',
  languages: [ 'French' ],
  released: ISODate('1916-11-23T00:00:00.000Z'),
  directors: [ 'Louis Feuillade' ],
  writers: [ 'Louis Feuillade' ],
  awards: { wins: 0, nominations: 1, text: '1 nomination.' },
  lastupdated: '2015-09-02 00:24:27.333000000',
  year: 1915,
  imdb: { rating: 6.8, votes: 2878, id: 6206 },
  countries: [ 'France' ],
  type: 'movie',
  tomatoes: {
    viewer: { rating: 3.8, numReviews: 2118, meter: 82 },
    dvd: ISODate('2000-05-16T00:00:00.000Z'),
    critic: { rating: 8.8, numReviews: 13, meter: 100 },
    lastUpdated: ISODate('2015-09-15T17:02:33.000Z'),
    rotten: 0,
    fresh: 13
  }
}
]
```

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

QUERY:

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

OUTPUT:

```
chithra_54> db.movies.find({ genres: 'Short' });
[ {
  _id: ObjectId('573a1390f29313caabcd4135'),
  plot: 'Three men hammer on an anvil and pass a bottle of beer around.',
  genres: [ 'Short' ],
  runtime: 1,
  cast: [ 'Charles Kaysen', 'John Ott' ],
  num_mflix_comments: 1,
  title: 'Blacksmith Scene',
  fullplot: 'A stationary camera looks at a large anvil with a blacksmith behind it and one on either side. The smith in the middle draws a heated metal rod from the fire, places it on the anvil, and all three begin a rhythmic hammering. After several blows, the metal goes back in the fire. One smith pulls out a bottle of beer, and they each take a swig. Then, out comes the glowing metal and the hammering resumes.',
  countries: [ 'USA' ],
  released: ISODate('1893-05-09T00:00:00.000Z'),
  directors: [ 'William K.L. Dickson' ],
  rated: 'UNRATED',
  awards: { wins: 1, nominations: 0, text: '1 win.' },
  lastupdated: '2015-08-26 00:03:50.133000000',
  year: 1893,
  imdb: { rating: 6.2, votes: 1189, id: 5 },
  type: 'movie',
  tomatoes: {
    viewer: { rating: 3, numReviews: 184, meter: 32 },
    lastUpdated: ISODate('2015-06-28T18:34:09.000Z')
  }
} ]
```

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' })
```

```
chithra_54> db.movies.find({ directors: 'William K.L. Dickson' });
[ {
  _id: ObjectId('573a1390f29313caabcd4135'),
  plot: 'Three men hammer on an anvil and pass a bottle of beer around.',
  genres: [ 'Short' ],
  runtime: 1,
  cast: [ 'Charles Kaysen', 'John Ott' ],
  num_mflix_comments: 1,
  title: 'Blacksmith Scene',
  fullplot: 'A stationary camera looks at a large anvil with a blacksmith behind it and one on either side. The smith in the middle draws a heated metal rod from the fire, places it on the anvil, and all three begin a rhythmic hammering. After several blows, the metal goes back in the fire. One smith pulls out a bottle of beer, and they each take a swig. Then, out comes the glowing metal and the hammering resumes.',
  countries: [ 'USA' ],
  released: ISODate('1893-05-09T00:00:00.000Z'),
  directors: [ 'William K.L. Dickson' ],
  rated: 'UNRATED',
  awards: { wins: 1, nominations: 0, text: '1 win.' },
  lastupdated: '2015-08-26 00:03:50.133000000',
  year: 1893,
  imdb: { rating: 6.2, votes: 1189, id: 5 },
  type: 'movie',
  tomatoes: {
    viewer: { rating: 3, numReviews: 184, meter: 32 },
    lastUpdated: ISODate('2015-06-28T18:34:09.000Z')
  }
} ]
```

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:

```
chithra_54> db.movies.find({ countries: 'USA' });
[
  {
    _id: ObjectId('573a1390f29313caabcd4135'),
    plot: 'Three men hammer on an anvil and pass a bottle of beer around.',
    genres: [ 'Short' ],
    runtime: 1,
    cast: [ 'Charles Kayser', 'John Ott' ],
    num_mflix_comments: 1,
    title: 'Blacksmith Scene',
    fullplot: 'A stationary camera looks at a large anvil with a blacksmith behind it and one on either side. The smith in the middle draws a heated metal rod from the fire, places it on the anvil, and all three begin a rhythmic hammering. After several blows, the metal goes back in the fire. One smith pulls out a bottle of beer, and they each take a swig. Then, out comes the glowing metal and the hammering resumes.',
    countries: [ 'USA' ],
    released: ISODate('1893-05-09T00:00:00.000Z'),
    directors: [ 'William K.L. Dickson' ],
    rated: 'UNRATED',
    awards: { wins: 1, nominations: 0, text: '1 win.' },
    lastupdated: '2015-08-26 00:03:50.133000000',
    year: 1893,
    imdb: { rating: 6.2, votes: 1189, id: 5 },
    type: 'movie',
    tomatoes: {
      viewer: { rating: 3, numReviews: 184, meter: 32 },
      lastUpdated: ISODate('2015-06-28T18:34:09.000Z')
    }
  }
]
chithra_54>
```

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

QUERY:

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

OUTPUT:

```
chithra_54> db.movies.find({ rated: 'UNRATED' });
[
  {
    _id: ObjectId('573a1390f29313caabcd4135'),
    plot: 'Three men hammer on an anvil and pass a bottle of beer around.',
    genres: [ 'Short' ],
    runtime: 1,
    cast: [ 'Charles Kayser', 'John Ott' ],
    num_mflix_comments: 1,
    title: 'Blacksmith Scene',
    fullplot: 'A stationary camera looks at a large anvil with a blacksmith behind it and one on either side. The smith in the middle draws a heated metal rod from the fire, places it on the anvil, and all three begin a rhythmic hammering. After several blows, the metal goes back in the fire. One smith pulls out a bottle of beer, and they each take a swig. Then, out comes the glowing metal and the hammering resumes.',
    countries: [ 'USA' ],
    released: ISODate('1893-05-09T00:00:00.000Z'),
    directors: [ 'William K.L. Dickson' ],
    rated: 'UNRATED',
    awards: { wins: 1, nominations: 0, text: '1 win.' },
    lastupdated: '2015-08-26 00:03:50.133000000',
    year: 1893,
    imdb: { rating: 6.2, votes: 1189, id: 5 },
    type: 'movie',
    tomatoes: {
      viewer: { rating: 3, numReviews: 184, meter: 32 },
      lastUpdated: ISODate('2015-06-28T18:34:09.000Z')
    }
  }
]
chithra_54>
```

7.) Retrieve all movies from the 'movies' collection that have complete information and have received more than 1000 votes on IMDb.

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

```
chithra_54> db.movies.find({ 'imdb.votes': { $gt: 1000 } });
[ {
  _id: ObjectId('573a1390f22313caabacd5907'),
  plot: 'An intrepid reporter and his loyal friend battle a bizarre secret society of criminals known as The Vampires.',
  genres: [ 'Action', 'Adventure', 'Crime' ],
  runtime: 399,
  rated: 'NOT RATED',
  cast: [ 'Musidora', 'édouardMathè', 'Marcel Lèvesque', 'Jean Aymè' ],
  poster: 'https://m.media-amazon.com/images/M/MVSBMTc1NTY3NDIzN15Bn8nXkFtZTgwNTiyODg5MTE@.V1_SY1000_SX677_AL_.jpg',
  title: 'Les vampires',
  fullplot: 'An intrepid reporter and his loyal friend battle a bizarre secret society of criminals known as The Vampires.',
  languages: [ 'French' ],
  released: ISODate('1916-11-23T00:00:00.000Z'),
  directors: [ 'Louis Feuillade' ],
  writers: [ 'Louis Feuillade' ],
  awards: { wins: 0, nominations: 1, text: '1 nomination.' },
  lastupdated: '2015-09-02 00:24:27.333000000',
  year: 1915,
  imdb: { rating: 6.8, votes: 2878, id: 6206 },
  countries: [ 'France' ],
  type: 'movie',
  tomatoes: {
    viewer: { rating: 3.8, numReviews: 2118, meter: 82 },
    dvd: ISODate('2000-05-16T00:00:00Z'),
    critic: { rating: 8.8, numReviews: 13, meter: 100 },
    lastUpdated: ISODate('2015-09-15T17:02:33.000Z'),
    rotten: 0,
    fresh: 13
  }
},
```

8.) Retrieve all movies from the 'movies' collection that have complete information and have an IMDb rating higher than 7.

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

```
chithra_54> db.movies.find({ 'imdb.rating': { $gt: 7 } });
chithra_54>
```

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 } })
```

```
chithra_54> db.movies.find({ 'tomatoes.viewer.rating': { $gt: 4 } });
chithra_54>
```

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

QUERY:

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

OUTPUT:

```
chithra_54> db.movies.find({ 'awards.wins': { $gt: 0 } });
[
  {
    _id: ObjectId('573a1390f29313caabcd4135'),
    plot: 'Three men hammer on an anvil and pass a bottle of beer around.',
    genres: [ 'Short' ],
    runtime: 1,
    cast: [ 'Charles Kayser', 'John Ott' ],
    num_mflix_comments: 1,
    title: 'Blacksmith Scene',
    fullplot: 'A stationary camera looks at a large anvil with a blacksmith behind it and one on either side. The smith in the middle draws a heated metal rod from the fire, places it on the anvil, and all three begin a rhythmic hammering. After several blows, the metal goes back in the fire. One smith pulls out a bottle of beer, and they each take a swig. Then, out comes the glowing metal and the hammering resumes.',
    countries: [ 'USA' ],
    released: ISODate('1893-05-09T00:00:00.000Z'),
    directors: [ 'William K.L. Dickson' ],
    rated: 'UNRATED',
    awards: { wins: 1, nominations: 0, text: '1 win.' },
    lastupdated: '2015-08-26 00:03:50.133000000',
    year: 1893,
    imdb: { rating: 6.2, votes: 1189, id: 5 },
    type: 'movie',
    tomatoes: {
      viewer: { rating: 3, numReviews: 184, meter: 32 },
      lastUpdated: ISODate('2015-06-28T18:34:09.000Z')
    }
  }
]
chithra_54>
```

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.

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

OUTPUT:

```
chithra_54> db.movies.find( { 'awards.nominations': { $gt: 0 } }, { title: 1, languages: 1, released: 1, directors: 1, writers: 1, awards: 1, year: 1, genres: 1, runt
runtime: 1, cast: 1, countries: 1 } );
[
  {
    _id: ObjectId('573a1390f22313caabcd5967'),
    genres: [ 'Action', 'Adventure', 'Crime' ],
    runtime: 399,
    cast: [ 'Musidora', 'éduardMathé', 'Marcel Lèvesque', 'Jean Aymé' ],
    title: 'Les vampires',
    languages: [ 'French' ],
    released: ISODate('1916-11-23T00:00:00.000Z'),
    directors: [ 'Louis Feuillade' ],
    writers: [ 'Louis Feuillade' ],
    awards: { wins: 0, nominations: 1, text: '1 nomination.' },
    year: 1915,
    countries: [ 'France' ]
  }
]
chithra_54>
```

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:

```
chithra_54> 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 } );
[
  {
    _id: ObjectId('573a1390f29313caabcd4135'),
    genres: [ 'Short' ],
    runtime: 1,
    cast: [ 'Charles Kayser', 'John Ott' ],
    title: 'Blacksmith Scene',
    countries: [ 'USA' ],
    released: ISODate('1893-05-09T00:00:00.000Z'),
    directors: [ 'William K.L. Dickson' ],
    awards: { wins: 1, nominations: 0, text: '1 win.' },
    year: 1893
  }
]
chithra_54>
```

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:

```
chithra_54> db.movies.find( { released: ISODate("1893-05-09T00:00:00.000Z") }, { title: 1, languages: 1, released: 1, directors: 1, writers: 1, countries: 1 } );
[
  {
    _id: ObjectId('573a1390f29313caabcd4135'),
    title: 'Blacksmith Scene',
    countries: [ 'USA' ],
    released: ISODate('1893-05-09T00:00:00.000Z'),
    directors: [ 'William K.L. Dickson' ]
  }
]
```

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:

```
chithra_54> db.movies.find( { title: /scene/i }, { title: 1, languages: 1, released: 1, directors: 1, writers: 1, countries: 1 });
[
  {
    _id: ObjectId('573a1390f29313caabcd4135'),
    title: 'Blacksmith Scene',
    countries: [ 'USA' ],
    released: ISODate('1893-05-09T00:00:00.000Z'),
    directors: [ 'William K.L. Dickson' ]
  }
]
chithra_54>
```

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

RESULT:

Thus the mongo db queries for movies collection are executed successfully.