

VIVEKANANDA INSTITUTE OF PROFESSIONAL STUDIES - TECHNICAL CAMPUS

Grade **A++** Accredited Institution by NAAC

NBA Accredited for MCA Programme; Recognized under Section 2(f) by UGC;
Affiliated to GGSIP University, Delhi; Recognized by Bar Council of India and AICTE

An ISO 9001:2015 Certified Institution

SCHOOL OF ENGINEERING & TECHNOLOGY

B. Tech Programme: AI-ML (A) (4th Semester)

**Course Title: Database Management
Systems Lab**

Course Code: AIML-254

Submitted To:

Dr. Deepika Bhatia

Submitted By:

Name: Kunsh Sabharwal

Enrolment No: 01117711623



VIVEKANANDA INSTITUTE OF PROFESSIONAL STUDIES - TECHNICAL CAMPUS

Grade A++ Accredited Institution by NAAC

NBA Accredited for MCA Programme; Recognized under Section 2(f) by UGC;

Affiliated to GGSIP University, Delhi; Recognized by Bar Council of India and AICTE

An ISO 9001:2015 Certified Institution

SCHOOL OF ENGINEERING & TECHNOLOGY

VISION OF INSTITUTE

To be an educational institute that empowers the field of engineering to build a sustainable future by providing quality education with innovative practices that supports people, planet and profit.

MISSION OF INSTITUTE

To groom the future engineers by providing value-based education and awakening students' curiosity, nurturing creativity and building capabilities to enable them to make significant contributions to the world.



VIVEKANANDA INSTITUTE OF PROFESSIONAL STUDIES - TECHNICAL CAMPUS

Grade A++ Accredited Institution by NAAC

NBA Accredited for MCA Programme; Recognized under Section 2(f) by UGC;
Affiliated to GGSIP University, Delhi; Recognized by Bar Council of India and AICTE

An ISO 9001:2015 Certified Institution

SCHOOL OF ENGINEERING & TECHNOLOGY

INDEX

S. No .	Experiment	Date	Marks			Remarks	Updated Marks	Faculty Signature
			Laboratory Assessment (15 Marks)	Class Participation (5 Marks)	Viva (5 Marks)			
1.	Study and practice various database management systems like MySQL/Oracle/ PostgreSQL/SQ L Server and others.							
2.	Implement simple queries of DDL and DML.							
3.	Implement basic queries to Create, Insert, Update, Delete and Select Statements for two different scenarios (For instance: Bank, College etc.)							
4.	Implement queries including various functions- mathematical, string, date etc.							



VIVEKANANDA INSTITUTE OF PROFESSIONAL STUDIES - TECHNICAL CAMPUS

Grade A++ Accredited Institution by NAAC

NBA Accredited for MCA Programme; Recognized under Section 2(f) by UGC;

Affiliated to GGSIP University, Delhi; Recognized by Bar Council of India and AICTE

An ISO 9001:2015 Certified Institution

SCHOOL OF ENGINEERING & TECHNOLOGY

5.	Implement queries including Sorting, Grouping and Subqueries- like any, all, exists, not exists.						
6.	Implement queries including various Set operations (Union, Intersection, Except etc.).						
7.	Implement various JOIN operations- (Inner, Outer).						
8.	Write a PL/SQL program using FOR loop to insert ten rows into a database table.						
9.	Given the table EMPLOYEE (Emp No, Name, Salary, Designation, DeptID), write a cursor to select the five highest-paid employees from the table.						

10.	Illustrate how you can embed PL/SQL in a high-level host language such as C/Java And demonstrates how a banking debit transaction might be done.							
-----	--	--	--	--	--	--	--	--


VIVEKANANDA INSTITUTE OF PROFESSIONAL STUDIES - TECHNICAL CAMPUS
Grade A++ Accredited Institution by NAAC

NBA Accredited for MCA Programme; Recognized under Section 2(f) by UGC;

Affiliated to GGSIP University, Delhi; Recognized by Bar Council of India and AICTE

An ISO 9001:2015 Certified Institution

SCHOOL OF ENGINEERING & TECHNOLOGY

Sr. No.	EXPERIMENT TITLE (Beyond Curriculum)	Date	Faculty Signature
1	Write the steps to install and implement NOSQL databases-MongoDB		
2	Study and implement basic commands of MongoDB		
3	Implement any one real-time project using MySQL/MongoDB such as Library Database Management System etc..		

SECTION 1
Database Management System Lab

GGSIPU

EXPERIMENT 1

Problem statement: Study and practice various database management systems like MySQL/Oracle/PostgreSQL/SQL Server and others.

Theory:



KUNSH SABHARWAL

ORACLE®

KUNSH SABHARWAL



KUNSH SABHARWAL



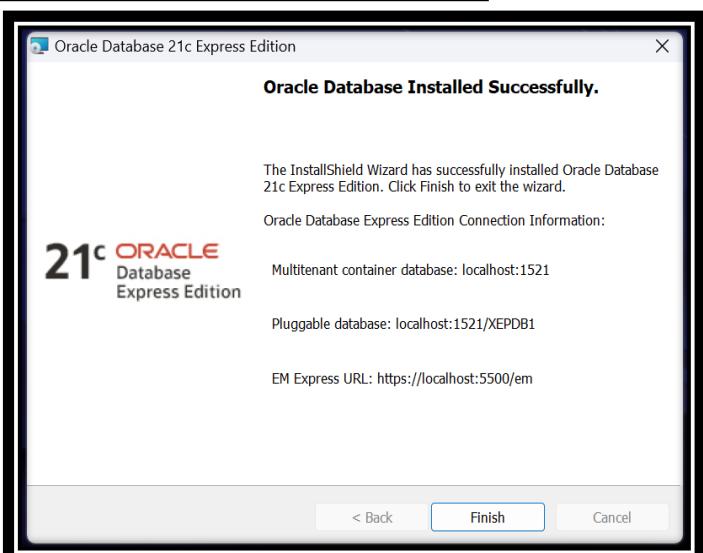
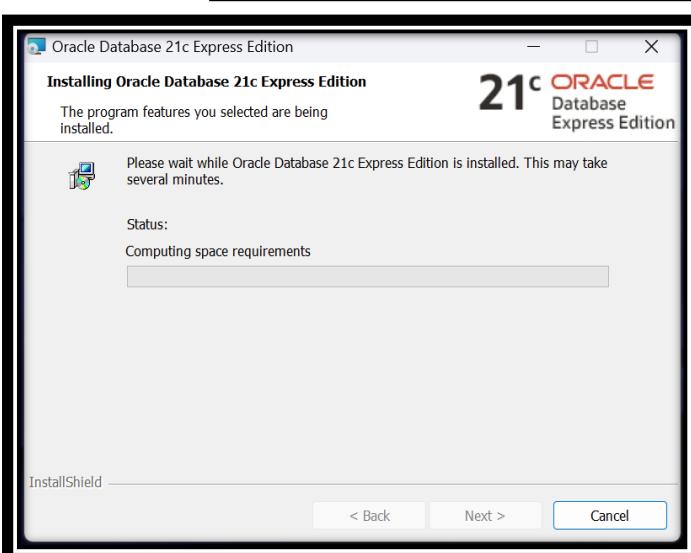
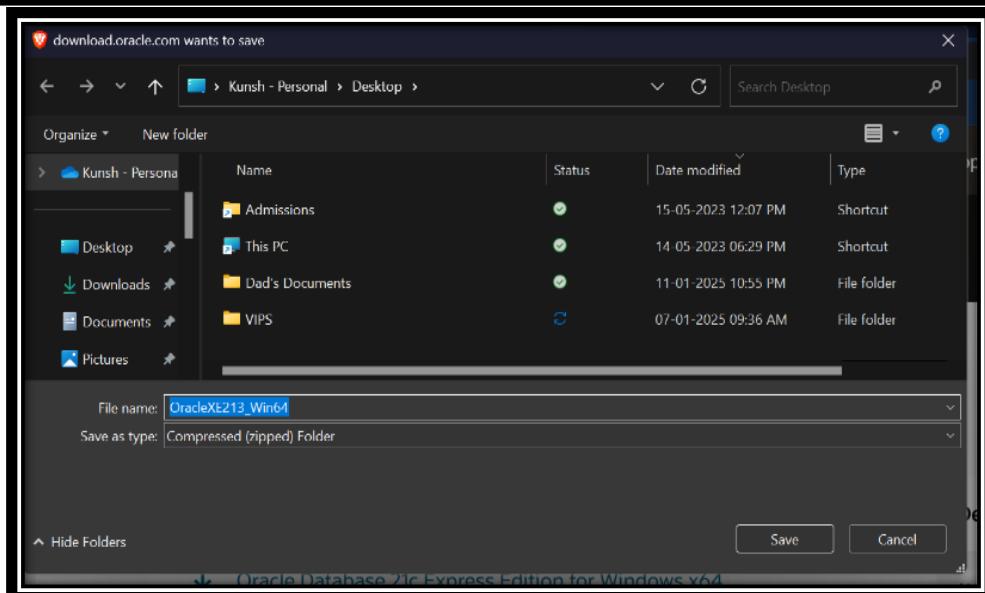
Installation screenshots of SQL:

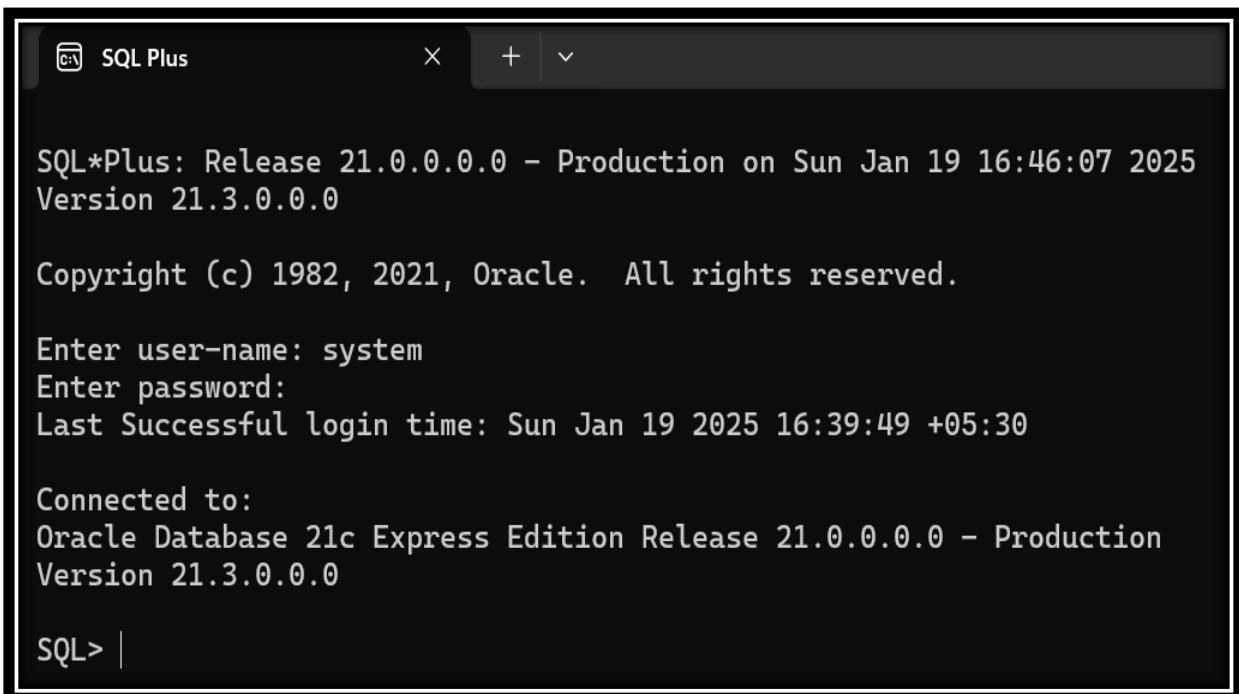
Oracle Database XE Downloads

Oracle Database 21c Express Edition

Download	Description
Oracle Database 21c Express Edition for Windows x64	(1,967,615,483 bytes - October 08, 2021) [Sha256sum: 939742c3305c466566a55f607638621b6aa7033a183175f6bcd6cffb48e6bc3f]
Oracle Database 21c Express Edition for Linux x64 (OL8)	(2,339,651,768 bytes - September 08, 2021) [Sha256sum: f8357b452de33478549a76557e8c5220ec243710ed86115c65b0c2bc]
Oracle Database 21c Express Edition for Linux x64 (OL7)	(2,339,017,432 bytes - September 08, 2021) [Sha256sum: 4c8f40a19d4d1a2f00e46df022943a04cc13fe62aed27c4c66a137e72]

Chat now **Call Sales** **+91 80-37132100** Complete list of local country numbers





SQL*Plus: Release 21.0.0.0.0 - Production on Sun Jan 19 16:46:07 2025
Version 21.3.0.0.0

Copyright (c) 1982, 2021, Oracle. All rights reserved.

Enter user-name: system
Enter password:
Last Successful login time: Sun Jan 19 2025 16:39:49 +05:30

Connected to:
Oracle Database 21c Express Edition Release 21.0.0.0.0 - Production
Version 21.3.0.0.0

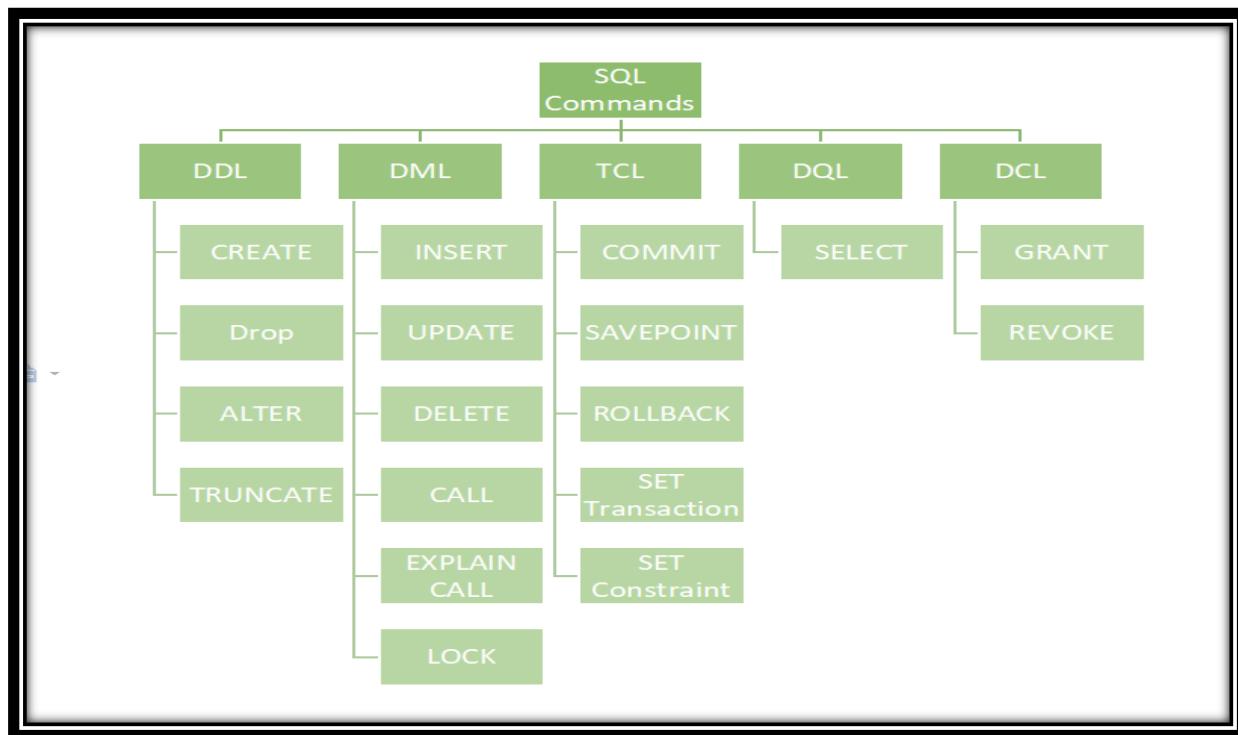
SQL> |

Learning Outcome:

EXPERIMENT 2

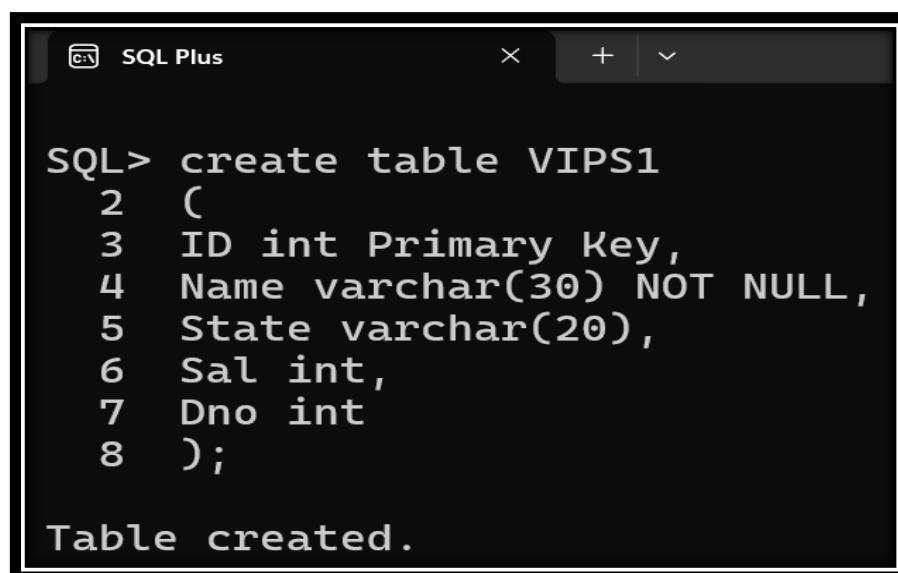
Problem statement: Implement simple queries of DDL and DML.

Theory:



SQL Commands: - (DDL Commands)

- 1) Create a table VIPS1 (ID, NAME, STATE, SAL, Dno). (Add Constraints)

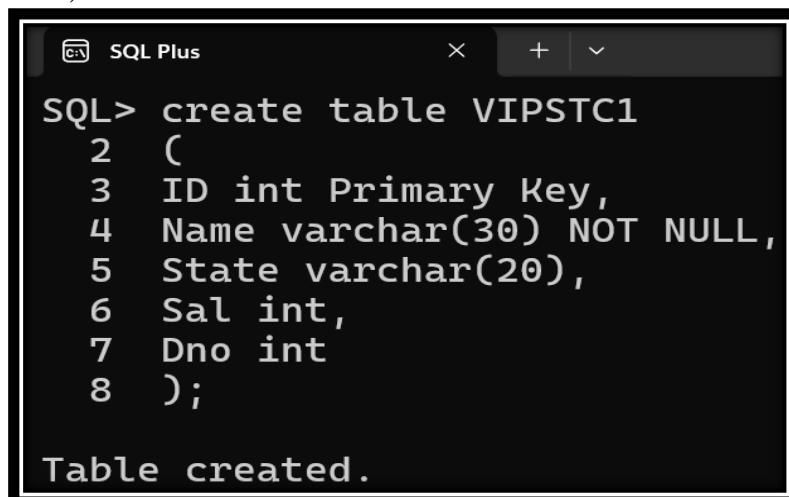


The screenshot shows a terminal window titled "SQL Plus". The command entered is:

```
SQL> create table VIPS1
  2  (
  3  ID int Primary Key,
  4  Name varchar(30) NOT NULL,
  5  State varchar(20),
  6  Sal int,
  7  Dno int
  8 );
```

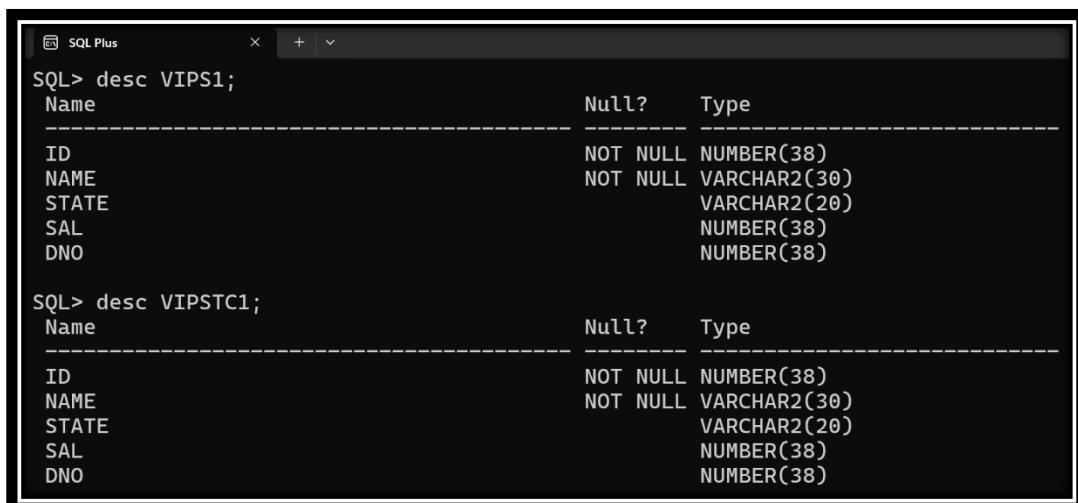
After executing the command, the response "Table created." is displayed.

- 2) Create a table VIPSTC1(ID, NAME, STATE, SAL, Dno). (Add Constraints)



```
SQL> create table VIPSTC1
  2  (
  3    ID int Primary Key,
  4    Name varchar(30) NOT NULL,
  5    State varchar(20),
  6    Sal int,
  7    Dno int
  8  );
```

Table created.

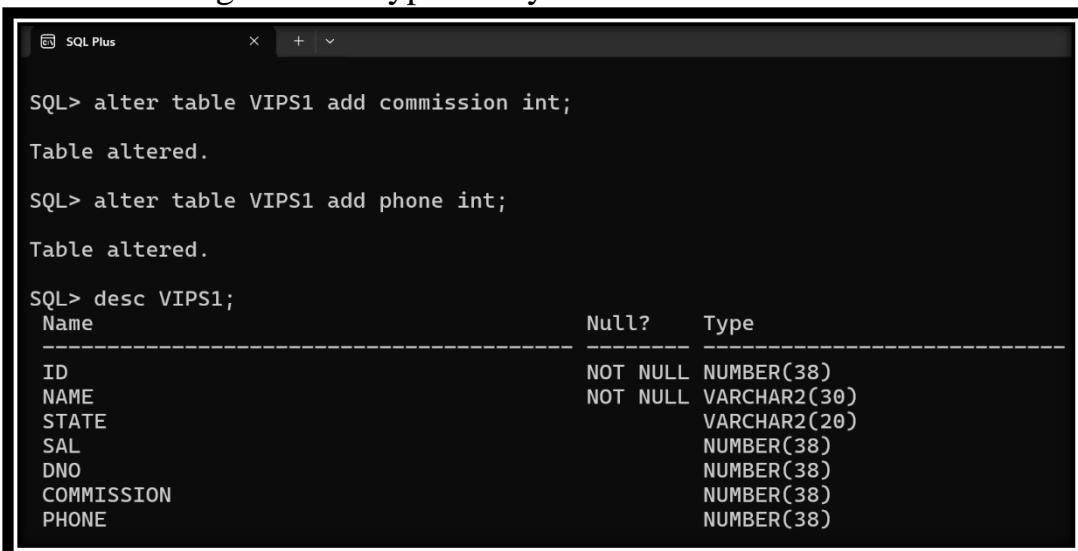


Name	Null?	Type
ID	NOT NULL	NUMBER(38)
NAME	NOT NULL	VARCHAR2(30)
STATE		VARCHAR2(20)
SAL		NUMBER(38)
DNO		NUMBER(38)

Name	Null?	Type
ID	NOT NULL	NUMBER(38)
NAME	NOT NULL	VARCHAR2(30)
STATE		VARCHAR2(20)
SAL		NUMBER(38)
DNO		NUMBER(38)

- 3) Add column Commission and Phone to table.

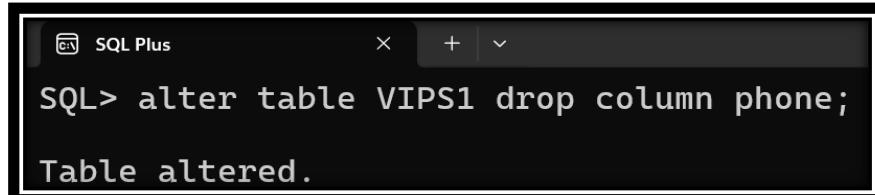
- Drop Column Phone
- Change the datatype of any of the column.



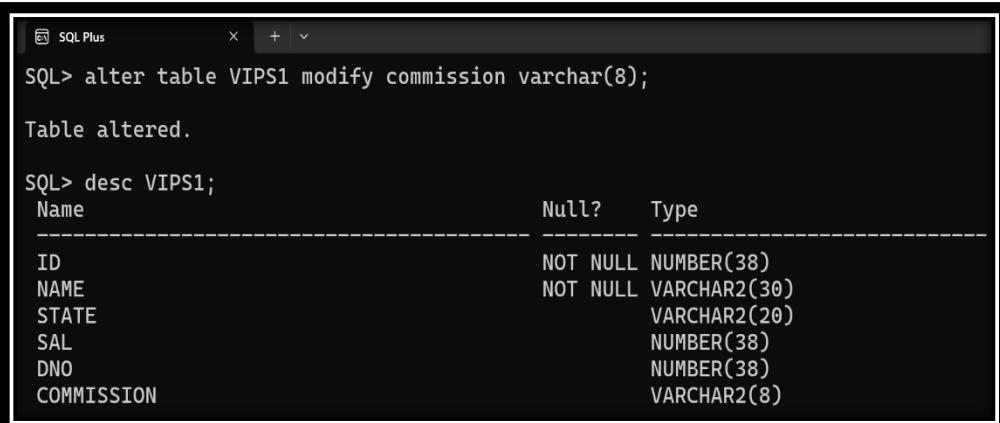
```
SQL> alter table VIPS1 add commission int;
Table altered.

SQL> alter table VIPS1 add phone int;
Table altered.

SQL> desc VIPS1;
Name          Null?    Type
-----        -----   -----
ID           NOT NULL NUMBER(38)
NAME         NOT NULL VARCHAR2(30)
STATE        VARCHAR2(20)
SAL          NUMBER(38)
DNO          NUMBER(38)
COMMISSION   NUMBER(38)
PHONE        NUMBER(38)
```



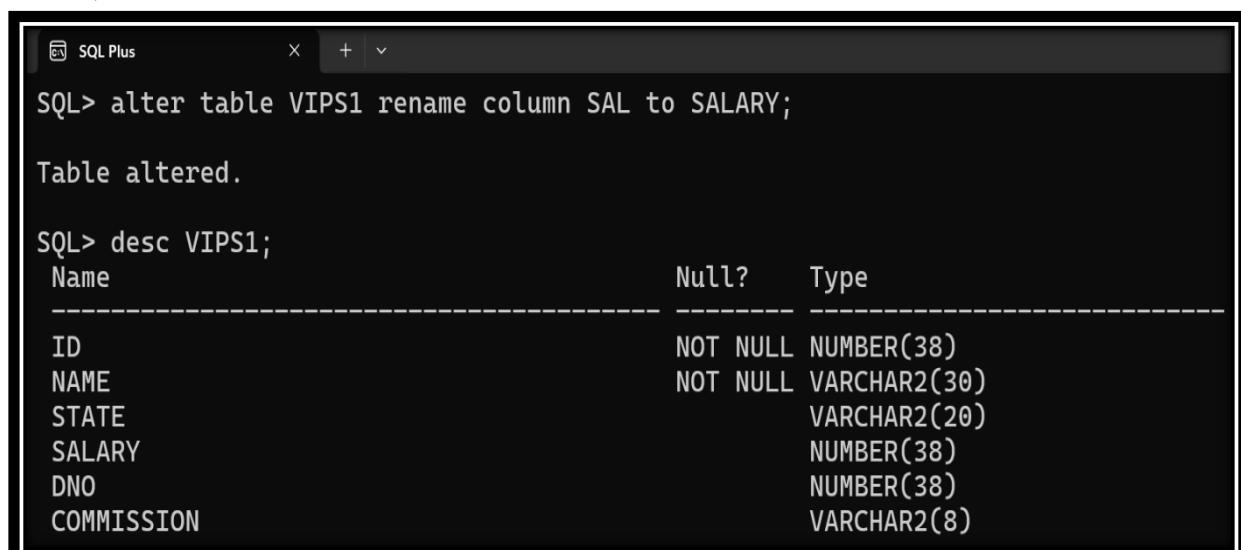
```
SQL> alter table VIPS1 drop column phone;
Table altered.
```



```
SQL> alter table VIPS1 modify commission varchar(8);
Table altered.

SQL> desc VIPS1;
Name          Null?    Type
-----        -----   -----
ID           NOT NULL NUMBER(38)
NAME          NOT NULL VARCHAR2(30)
STATE         VARCHAR2(20)
SAL            NUMBER(38)
DNO            NUMBER(38)
COMMISSION    VARCHAR2(8)
```

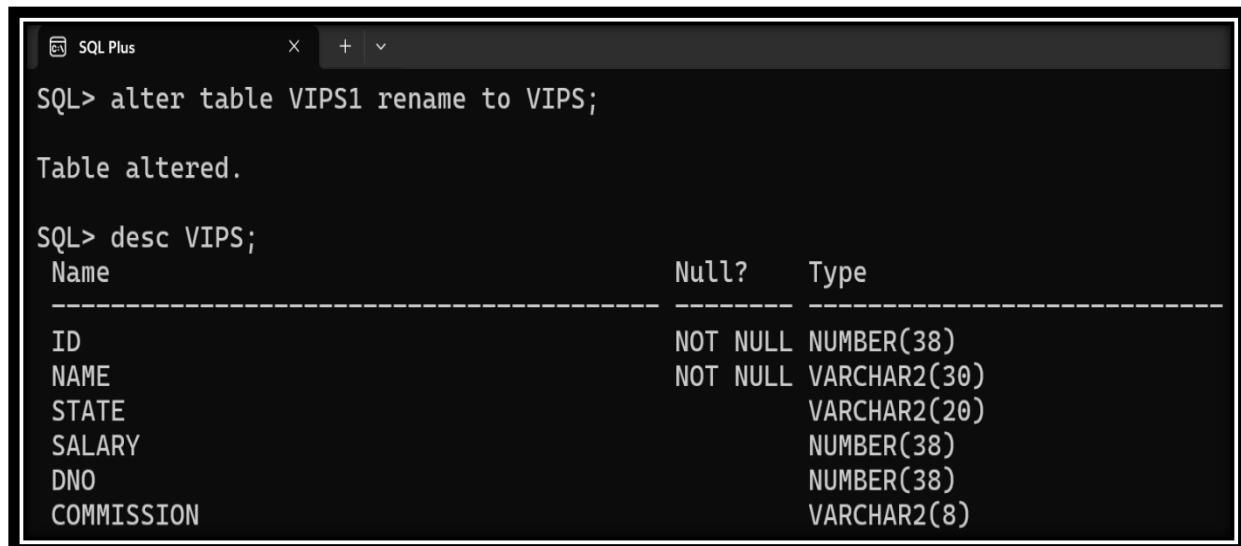
- 4) Rename one of the columns of the table.



```
SQL> alter table VIPS1 rename column SAL to SALARY;
Table altered.

SQL> desc VIPS1;
Name          Null?    Type
-----        -----   -----
ID           NOT NULL NUMBER(38)
NAME          NOT NULL VARCHAR2(30)
STATE         VARCHAR2(20)
SALARY        NUMBER(38)
DNO            NUMBER(38)
COMMISSION    VARCHAR2(8)
```

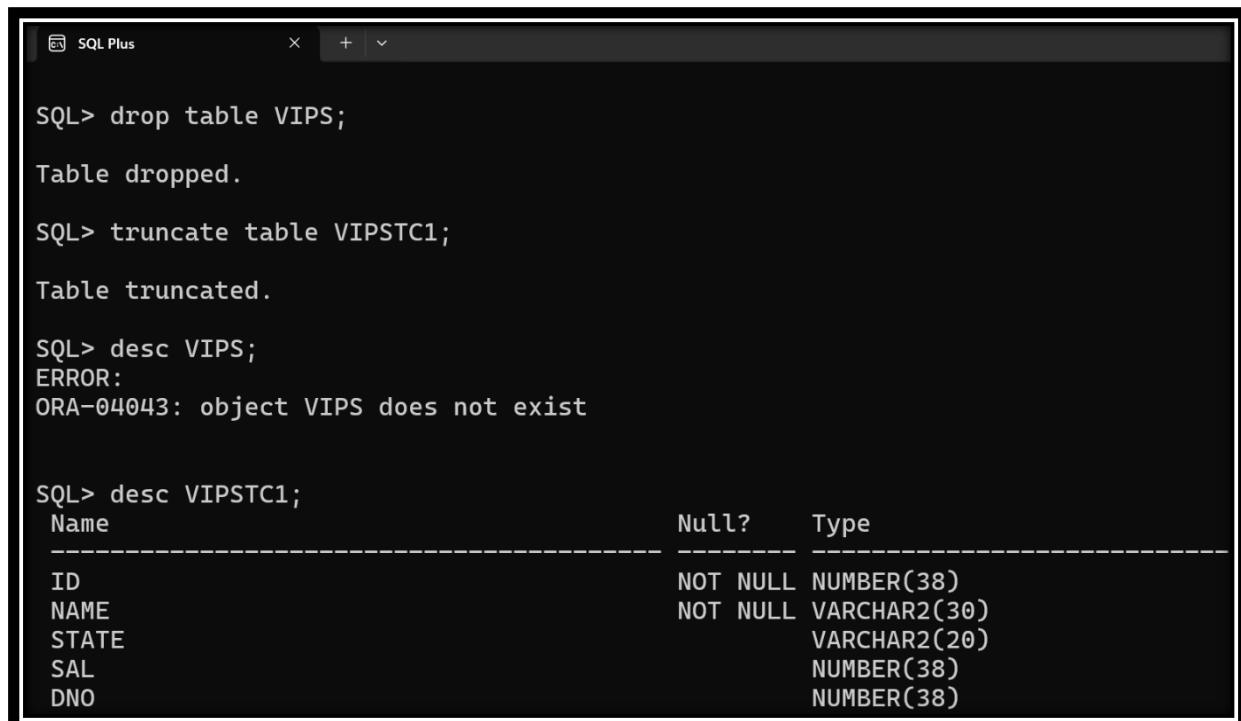
- 5) Rename the table name.



```
SQL> alter table VIPS1 rename to VIPS;
Table altered.

SQL> desc VIPS;
Name          Null?    Type
-----        -----   -----
ID           NOT NULL NUMBER(38)
NAME          NOT NULL VARCHAR2(30)
STATE         VARCHAR2(20)
SALARY        NUMBER(38)
DNO            NUMBER(38)
COMMISSION    VARCHAR2(8)
```

- 6) Drop the table VIPS1 and Truncate VIPSTC1 and show the difference.



```

SQL> drop table VIPS;
Table dropped.

SQL> truncate table VIPSTC1;
Table truncated.

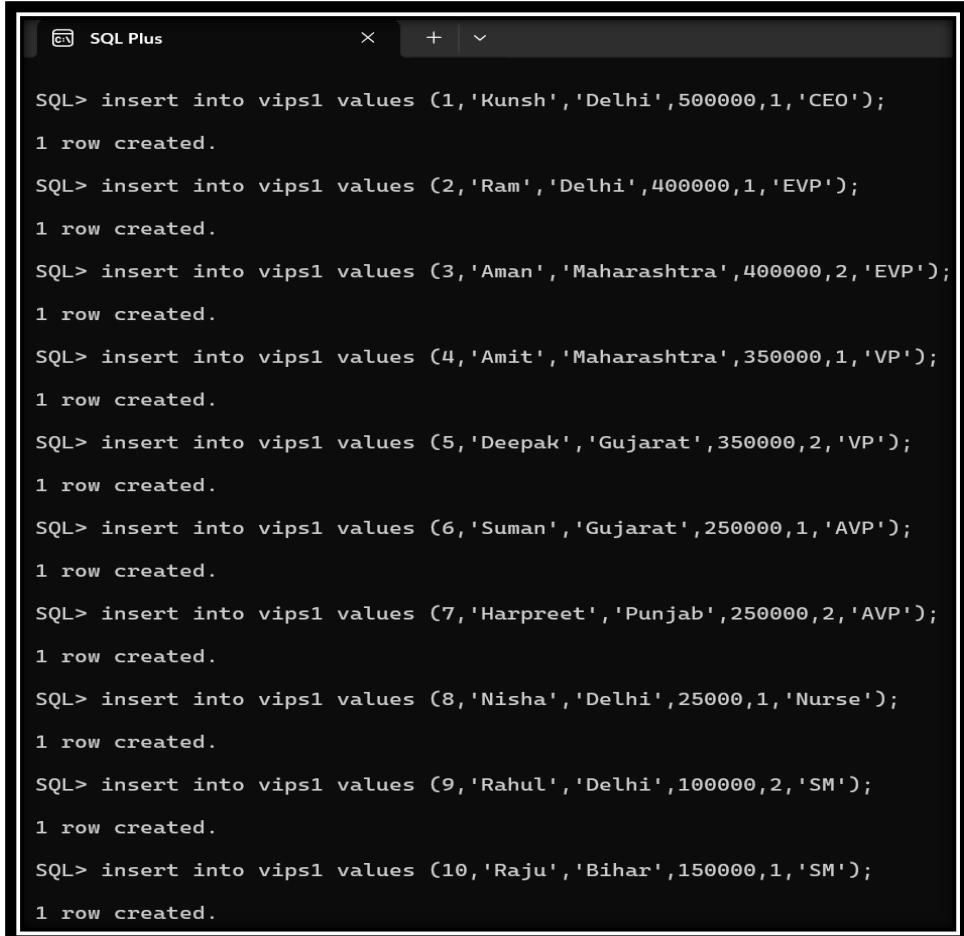
SQL> desc VIPS;
ERROR:
ORA-04043: object VIPS does not exist

SQL> desc VIPSTC1;
Name          Null?    Type
-----        -----   -----
ID           NOT NULL NUMBER(38)
NAME         NOT NULL VARCHAR2(30)
STATE        VARCHAR2(20)
SAL          NUMBER(38)
DNO          NUMBER(38)

```

DML Commands: -

- 1) Insert Values in both tables. (at least 7 rows)



```

SQL> insert into vips1 values (1,'Kunsh','Delhi',500000,1,'CEO');
1 row created.

SQL> insert into vips1 values (2,'Ram','Delhi',400000,1,'EVP');
1 row created.

SQL> insert into vips1 values (3,'Aman','Maharashtra',400000,2,'EVP');
1 row created.

SQL> insert into vips1 values (4,'Amit','Maharashtra',350000,1,'VP');
1 row created.

SQL> insert into vips1 values (5,'Deepak','Gujarat',350000,2,'VP');
1 row created.

SQL> insert into vips1 values (6,'Suman','Gujarat',250000,1,'AVP');
1 row created.

SQL> insert into vips1 values (7,'Harpreet','Punjab',250000,2,'AVP');
1 row created.

SQL> insert into vips1 values (8,'Nisha','Delhi',25000,1,'Nurse');
1 row created.

SQL> insert into vips1 values (9,'Rahul','Delhi',100000,2,'SM');
1 row created.

SQL> insert into vips1 values (10,'Raju','Bihar',150000,1,'SM');
1 row created.

```

```
SQL> insert into vipstc1 values (1,'Kunsh','Delhi',500000,1);
1 row created.

SQL> insert into vipstc1 values (2,'Ram','Delhi',400000,1);
1 row created.

SQL> insert into vipstc1 values (3,'Aman','Maharashtra',400000,2);
1 row created.

SQL> insert into vipstc1 values (4,'Amit','Maharashtra',350000,1);
1 row created.

SQL> insert into vipstc1 values (5,'Deepak','Gujarat',350000,2);
1 row created.

SQL> insert into vipstc1 values (6,'Suman','Gujarat',250000,1);
1 row created.

SQL> insert into vipstc1 values (7,'Harpreet','Punjab',250000,2);
1 row created.

SQL> insert into vipstc1 values (8,'Nisha','Delhi',25000,1);
1 row created.

SQL> insert into vipstc1 values (9,'Rahul','Delhi',100000,2);
1 row created.

SQL> insert into vipstc1 values (10,'Raju','Bihar',150000,1);
1 row created.
```

2) Show the entire tables.

```
SQL> select * from vips1;

        ID NAME          STATE      SALARY
-----+-----+-----+-----+
        DNO DESIGNATION
-----+-----+
        1 Kunsh           Delhi     500000
        1 CEO

        2 Ram             Delhi     400000
        1 EVP

        3 Aman            Maharashtra 400000
        2 EVP

        ID NAME          STATE      SALARY
-----+-----+-----+-----+
        DNO DESIGNATION
-----+-----+
        4 Amit            Maharashtra 350000
        1 VP

        5 Deepak          Gujarat   350000
        2 VP

        6 Suman            Gujarat   250000
        1 AVP

        ID NAME          STATE      SALARY
-----+-----+-----+-----+
        DNO DESIGNATION
-----+-----+
        7 Harpreet         Punjab    250000
        2 AVP

        8 Nisha            Delhi     25000
        1 Nurse

        9 Rahul            Delhi     100000
        2 SM

        ID NAME          STATE      SALARY
-----+-----+-----+-----+
        DNO DESIGNATION
-----+-----+
        10 Raju            Bihar     150000
        1 SM

10 rows selected.
```

SQL Plus

X + ▾

SQL> select * from vipstcl;

ID	NAME	STATE	SAL
<hr/>			
1	Kunsh	Delhi	500000
1			
2	Ram	Delhi	400000
1			
3	Aman	Maharashtra	400000
2			
<hr/>			
ID	NAME	STATE	SAL
<hr/>			
4	Amit	Maharashtra	350000
1			
5	Deepak	Gujarat	350000
2			
6	Suman	Gujarat	250000
1			
<hr/>			
ID	NAME	STATE	SAL
<hr/>			
7	Harpreet	Punjab	250000
2			
8	Nisha	Delhi	25000
1			
9	Rahul	Delhi	100000
2			
<hr/>			
ID	NAME	STATE	SAL
<hr/>			
10	Raju	Bihar	150000
1			

10 rows selected.

3) Select only 3 columns from the table (With and without where clause)

```
SQL> select ID, NAME, STATE from vips1;

ID NAME STATE
----- -----
1 Kunsh Delhi
2 Ram Delhi
3 Aman Maharashtra
4 Amit Maharashtra
5 Deepak Gujarat
6 Suman Gujarat
7 Harpreet Punjab
8 Nisha Delhi
9 Rahul Delhi
10 Raju Bihar

10 rows selected.

SQL> select ID, NAME, STATE from vips1 where salary<50000;

ID NAME STATE
----- -----
8 Nisha Delhi

SQL> select ID, NAME, STATE from vips1 where state='Delhi';

ID NAME STATE
----- -----
1 Kunsh Delhi
2 Ram Delhi
8 Nisha Delhi
9 Rahul Delhi
```

4) Update the salary of any one of the employees by 3000.

```
SQL> update vips1 set salary=salary+3000 where id=8;

1 row updated.

SQL> select * from vips1 where id=8;

ID NAME STATE SALARY
----- -----
DNO DESIGNATION
----- -----
8 Nisha Delhi 28000
1 Nurse
```

5) Update the salary of each employee by 100.

```

SQL> update vips1 set salary=salary+100;
10 rows updated.

SQL> select * from vips1;

      ID NAME          STATE        SALARY
-----+-----+-----+-----+
      DNO DESIGNATION
-----+-----+
      1 Kunsh           Delhi       500100
      1 CEO

      2 Ram             Delhi       400100
      1 EVP

      3 Aman            Maharashtra 400100
      2 EVP

      ID NAME          STATE        SALARY
-----+-----+-----+-----+
      DNO DESIGNATION
-----+-----+
      4 Amit            Maharashtra 350100
      1 VP

      5 Deepak          Gujarat     350100
      2 VP

      6 Suman            Gujarat     250100
      1 AVP

      ID NAME          STATE        SALARY
-----+-----+-----+-----+
      DNO DESIGNATION
-----+-----+
      7 Harpreet         Punjab      250100
      2 AVP

      8 Nisha            Delhi       28100
      1 Nurse

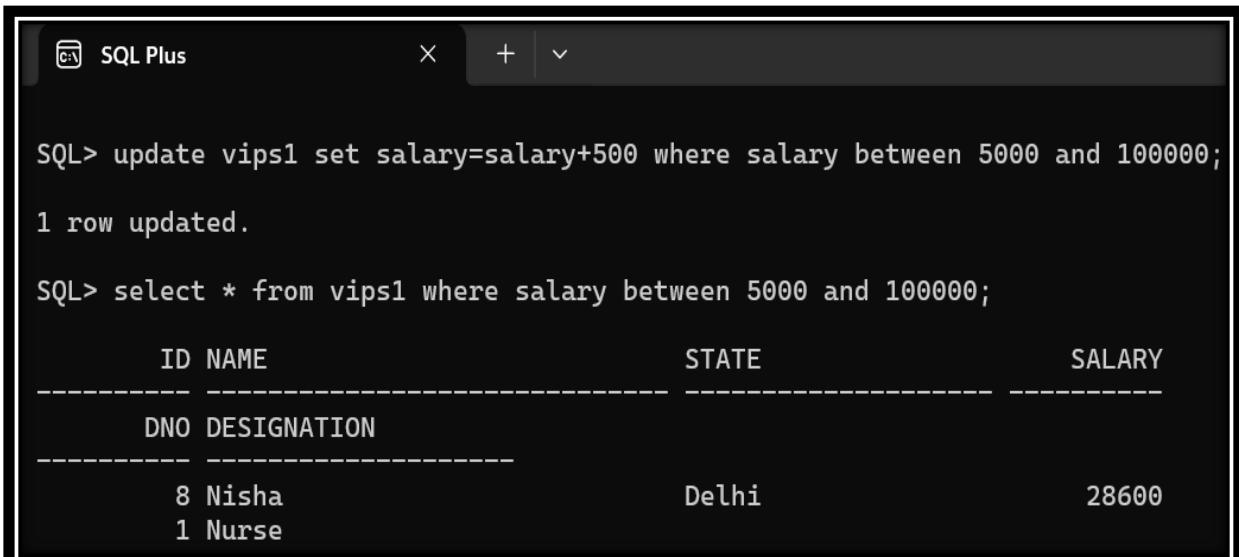
      9 Rahul            Delhi       100100
      2 SM

      ID NAME          STATE        SALARY
-----+-----+-----+-----+
      DNO DESIGNATION
-----+-----+
      10 Raju            Bihar       150100
      1 SM

10 rows selected.

```

- 6) Update the salary of the employee by 500 with salary between 5000 and 1,00,000.

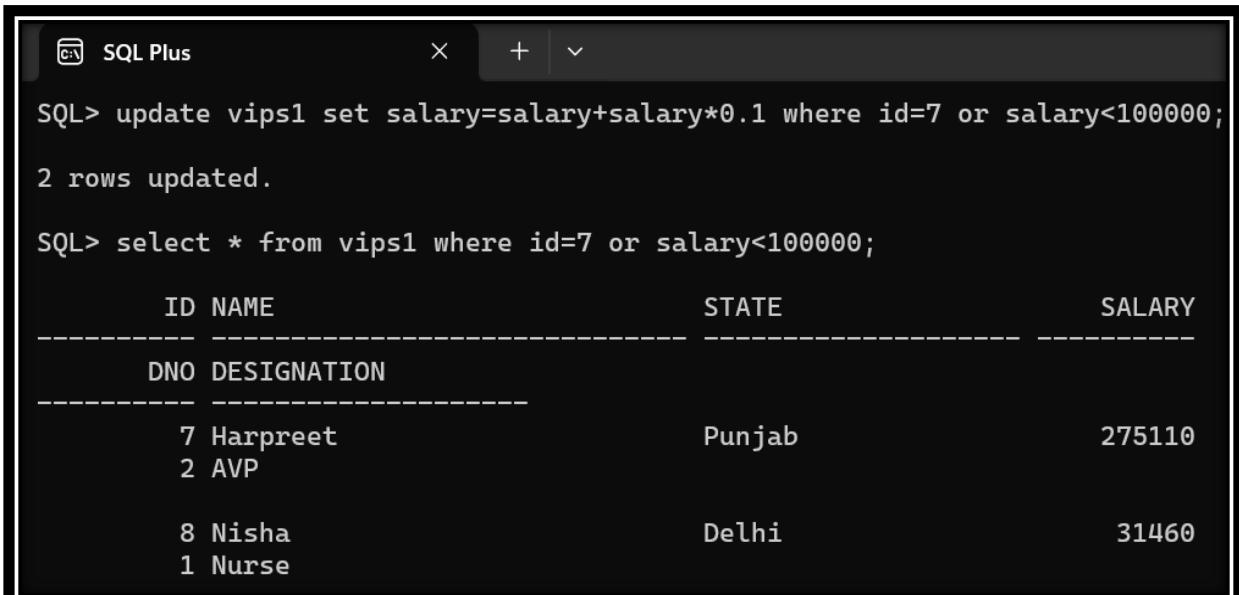


```
SQL> update vips1 set salary=salary+500 where salary between 5000 and 100000;
1 row updated.

SQL> select * from vips1 where salary between 5000 and 100000;

ID NAME          STATE      SALARY
--- ---          ---       ---
DNO DESIGNATION
--- ---
8 Nisha          Delhi     28600
1 Nurse          
```

- 7) Update the salary of the employees by 10% whose ID is 7 or salary is less than 1,00,000.



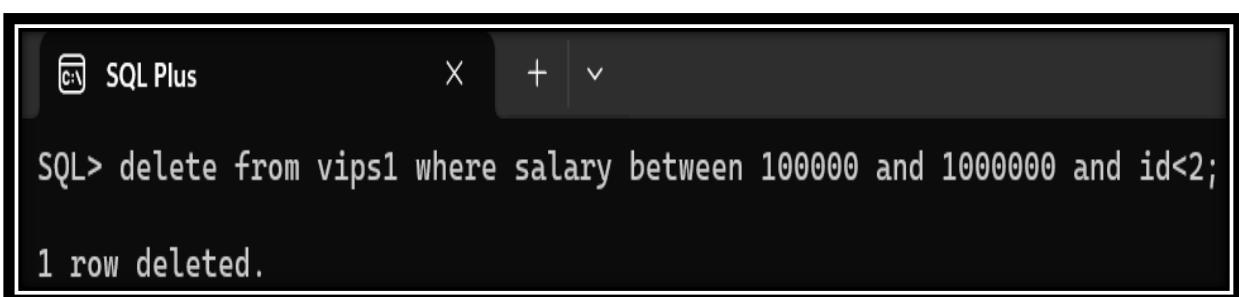
```
SQL> update vips1 set salary=salary+salary*0.1 where id=7 or salary<100000;
2 rows updated.

SQL> select * from vips1 where id=7 or salary<100000;

ID NAME          STATE      SALARY
--- ---          ---       ---
DNO DESIGNATION
--- ---
7 Harpreet       Punjab    275110
2 AVP           

8 Nisha          Delhi     31460
1 Nurse          
```

- 8) Delete the employee with salary between 1,00,000 and 1,00,00,000 and ID less than 2.



```
SQL> delete from vips1 where salary between 100000 and 1000000 and id<2;

1 row deleted. 
```

```
SQL> select * from vips1;
```

ID	NAME	STATE	SALARY
2	Ram	Delhi	400100
1	EVP		
3	Aman	Maharashtra	400100
2	EVP		
4	Amit	Maharashtra	350100
1	VP		
5	Deepak	Gujarat	350100
2	VP		
6	Suman	Gujarat	250100
1	AVP		
7	Harpreet	Punjab	275110
2	AVP		
8	Nisha	Delhi	31460
1	Nurse		
9	Rahul	Delhi	100100
2	SM		
10	Raju	Bihar	150100
1	SM		

```
9 rows selected.
```

9) Delete the employee whose designation is Nurse.

```
SQL> delete from vips1 where designation='Nurse';

1 row deleted.

SQL> select * from vips1;

ID NAME STATE SALARY
--- --- --- ---
DNO DESIGNATION
--- --- --- ---

2 Ram Delhi 400100
1 EVP

3 Aman Maharashtra 400100
2 EVP

4 Amit Maharashtra 350100
1 VP

ID NAME STATE SALARY
--- --- --- ---
DNO DESIGNATION
--- --- --- ---

5 Deepak Gujarat 350100
2 VP

6 Suman Gujarat 250100
1 AVP

7 Harpreet Punjab 275110
2 AVP

ID NAME STATE SALARY
--- --- --- ---
DNO DESIGNATION
--- --- --- ---

9 Rahul Delhi 100100
2 SM

10 Raju Bihar 150100
1 SM

8 rows selected.
```

10) Delete the employee with no salary.

```
SQL> update vips1 set salary=NULL where id=5;  
1 row updated.
```

```
SQL> delete from vips1 where salary is NULL;  
1 row deleted.  
SQL> select * from vips1;  
  
ID NAME STATE SALARY  
---  
DNO DESIGNATION  
---  
2 Ram Delhi 400100  
1 EVP  
  
3 Aman Maharashtra 400100  
2 EVP  
  
4 Amit Maharashtra 350100  
1 VP  
  
ID NAME STATE SALARY  
---  
DNO DESIGNATION  
---  
6 Suman Gujarat 250100  
1 AVP  
  
7 Harpreet Punjab 275110  
2 AVP  
  
9 Rahul Delhi 100100  
2 SM  
  
ID NAME STATE SALARY  
---  
DNO DESIGNATION  
---  
10 Raju Bihar 150100  
1 SM  
  
7 rows selected.
```

Learning Outcome:

EXPERIMENT 3

Problem statement: Implement basic queries to Create, Insert, Update, Delete and Select Statements for two different scenarios (For instance: Bank, College etc.)

Theory:

SQL Commands: - (For Bank Scenario)

- 1) Create the 5 tables given below: -
 - a. Acc (acc_no, name, phone)
 - b. Trans (acc_no, C_D, amount)
 - c. Loan (acc_no, interest, time, type)
 - d. FD (acc_no, amount, interest, time)
 - e. Locker (acc_no, l_no)

```
SQL> create table acc
  2  (
  3  acc_no int Primary Key,
  4  name varchar(10),
  5  phone int
  6  );

Table created.

SQL> create table trans
  2  (
  3  acc_no int,foreign key(acc_no) references acc(acc_no),
  4  c_d varchar(3),
  5  amount int
  6  );

Table created.

SQL>
SQL> create table loan
  2  (
  3  acc_no int,foreign key(acc_no) references acc(acc_no),
  4  interest int,
  5  time_month int,
  6  type varchar(10)
  7  );

Table created.

SQL> create table fd
  2  (
  3  acc_no int,foreign key(acc_no) references acc(acc_no),
  4  amount int,
  5  interest int,
  6  time_month int
  7  );

Table created.

SQL> create table locker
  2  (
  3  acc_no int, foreign key(acc_no) references acc(acc_no),
  4  l_no int
  5  );

Table created.
```

```
SQL> desc acc;
Name Null? Type
-----
ACC_NO NOT NULL NUMBER(38)
NAME VARCHAR2(10)
PHONE NUMBER(38)

SQL> desc trans;
Name Null? Type
-----
ACC_NO NUMBER(38)
C_D VARCHAR2(3)
AMOUNT NUMBER(38)

SQL> desc loan;
Name Null? Type
-----
ACC_NO NUMBER(38)
INTEREST NUMBER(38)
TIME_MONTH NUMBER(38)
TYPE VARCHAR2(10)

SQL> desc fd;
Name Null? Type
-----
ACC_NO NUMBER(38)
AMOUNT NUMBER(38)
INTEREST NUMBER(38)
TIME_MONTH NUMBER(38)

SQL> desc locker;
Name Null? Type
-----
ACC_NO NUMBER(38)
L_NO NUMBER(38)
```

2) Insert values in each table and show contents.

```
SQL> insert into acc values(10000, 'Sumit', 9999999999);
1 row created.

SQL> insert into acc values(10001, 'Raman', 8888888888);
1 row created.

SQL> insert into acc values(10002, 'Sita', 7777766666);
1 row created.

SQL> insert into acc values(10003, 'Kunsh', 5555555555);
1 row created.

SQL> insert into acc values(10004, 'Ketan', 4567891230);
1 row created.
```

```
SQL*Plus  
SQL> insert into trans values(10000, 'C', 50000);  
1 row created.  
  
SQL> insert into trans values(10001, 'D', 75000);  
1 row created.  
  
SQL> insert into trans values(10002, 'C', 15000);  
1 row created.  
  
SQL> insert into trans values(10003, 'D', 25000);  
1 row created.  
  
SQL> insert into trans values(10004, 'C', 10000);  
1 row created.
```

```
SQL*Plus  
SQL> insert into loan values(10000, 7, 12, 'home');  
1 row created.  
  
SQL> insert into loan values(10001, 5, 6, 'personal');  
1 row created.  
  
SQL> insert into loan values(10002, 6, 24, 'education');  
1 row created.  
  
SQL> insert into loan values(10003, 8, 9, 'home');  
1 row created.  
  
SQL> insert into loan values(10004, 12, 6, 'car');  
1 row created.
```

```
SQL*Plus  
SQL> insert into fd values(10000, 12000, 6, 12);  
1 row created.  
  
SQL> insert into fd values(10001, 24000, 4, 6);  
1 row created.  
  
SQL> insert into fd values(10002, 30000, 5, 9);  
1 row created.  
  
SQL> insert into fd values(10003, 60000, 8, 18);  
1 row created.  
  
SQL> insert into fd values(10004, 50000, 6, 12);  
1 row created.
```

```
SQL*Plus  
SQL> insert into locker values(10000, 100);  
1 row created.  
  
SQL> insert into locker values(10001, 101);  
1 row created.  
  
SQL> insert into locker values(10002, 102);  
1 row created.  
  
SQL> insert into locker values(10003, 103);  
1 row created.  
  
SQL> insert into locker values(10004, 104);  
1 row created.
```

```
SQL> select * from acc;
```

ACC_NO	NAME	PHONE
10000	Sumit	9999999999
10001	Raman	8888888888
10002	Sita	7777766666
10003	Kunsh	5555555555
10004	Ketan	4567891230

```
SQL> select * from trans;
```

ACC_NO	C_D	AMOUNT
10000	C	50000
10001	D	75000
10002	C	15000
10003	D	25000
10004	C	10000

```
SQL> select * from loan;
```

ACC_NO	INTEREST	TIME_MONTH	TYPE
10000	7	12	home
10001	5	6	personal
10002	6	24	education
10003	8	9	home
10004	12	6	car

```
SQL> select * from fd;
```

ACC_NO	AMOUNT	INTEREST	TIME_MONTH
10000	12000	6	12
10001	24000	4	6
10002	30000	5	9
10003	60000	8	18
10004	50000	6	12

```
SQL> select * from locker;
```

ACC_NO	L_NO
10000	100
10001	101
10002	102
10003	103
10004	104

3) Add a column loan_no in loan table and update values for this column.

```
SQL> alter table loan add loan_no int;
```

Table altered.

```
SQL> desc loan;
```

Name	Null?	Type
ACC_NO		NUMBER(38)
INTEREST		NUMBER(38)
TIME_MONTH		NUMBER(38)
TYPE		VARCHAR2(10)
LOAN_NO		NUMBER(38)

```

SQL> update loan set loan_no=1 where acc_no=10000;
1 row updated.

SQL> update loan set loan_no=2 where acc_no=10001;
1 row updated.

SQL> update loan set loan_no=3 where acc_no=10002;
1 row updated.

SQL> update loan set loan_no=4 where acc_no=10003;
1 row updated.

SQL> update loan set loan_no=5 where acc_no=10004;
1 row updated.

SQL> select * from loan;

```

ACC_NO	INTEREST	TIME_MONTH	TYPE	LOAN_NO
10000	7	12	home	1
10001	5	6	personal	2
10002	6	24	education	3
10003	8	9	home	4
10004	12	6	car	5

- 4) Insert a row in account table without phone number.

```

SQL> insert into acc values(10005, 'Raju', null);
1 row created.

SQL> select * from acc;

      ACC_NO    NAME          PHONE
-----  -----  -----
      10000  Sumit  9999999999
      10001  Raman  8888888888
      10002  Sita   7777766666
      10003  Kunsh  5555555555
      10004  Ketan  4567891230
      10005  Raju

6 rows selected.

```

- 5) Delete account with no phone number.

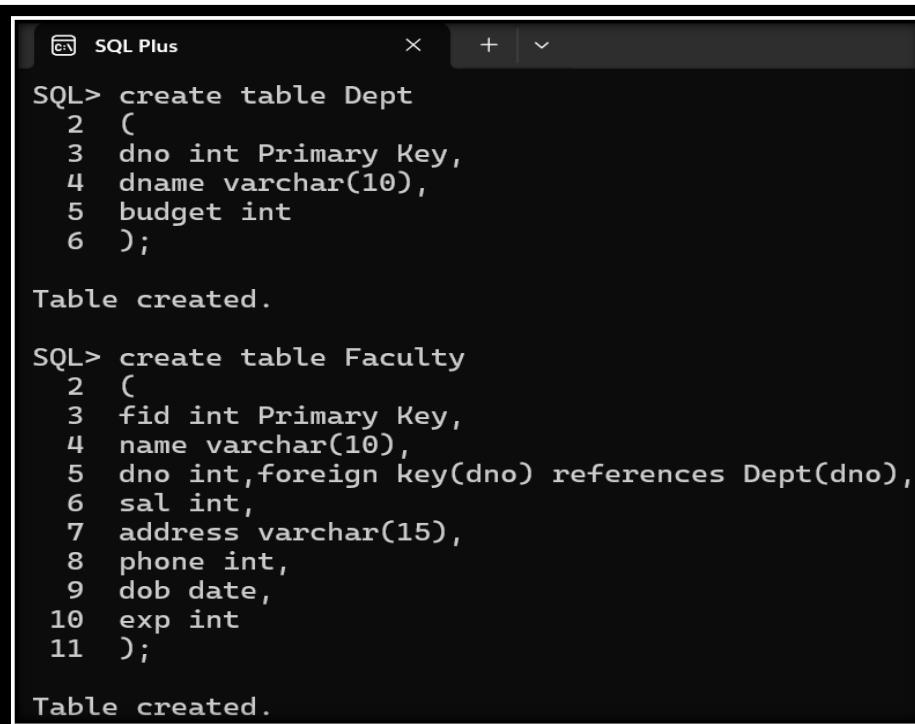
```
SQL> delete from acc where phone is null;
1 row deleted.

SQL> select * from acc;

ACC_NO NAME           PHONE
----- ----- -----
10000  Sumit          9999999999
10001  Raman          8888888888
10002  Sita           7777766666
10003  Kunsh          5555555555
10004  Ketan          4567891230
```

SQL Commands: - (For College Scenario)

- 1) Create the below mentioned tables with proper constraints: -
 - a. Faculty (fid, name, dno, sal, address, phone, dob, exp)
 - b. Dept (dno, dname, budget)
 - c. Student (sid, name, dno, phone, dob, address, sem)
 - d. Society (sc_id, name)
 - e. Std_soc (sid, sc_id)
 - f. Course (cid, cname)
 - g. Fac_course (fid, cid)



The screenshot shows the SQL Plus interface with the title bar "SQL Plus". The command window contains the following SQL code:

```
SQL> create table Dept
  2  (
  3  dno int Primary Key,
  4  dname varchar(10),
  5  budget int
  6  );
Table created.

SQL> create table Faculty
  2  (
  3  fid int Primary Key,
  4  name varchar(10),
  5  dno int,foreign key(dno) references Dept(dno),
  6  sal int,
  7  address varchar(15),
  8  phone int,
  9  dob date,
 10  exp int
 11  );
Table created.
```

```
SQL> create table student
  2  (
  3  sid int Primary Key,
  4  name varchar(10),
  5  dno int,foreign key(dno) references Dept(dno),
  6  phone int,
  7  dob date,
  8  address varchar(15),
  9  sem int
 10  );
Table created.

SQL> create table society
  2  (
  3  sc_id int Primary Key,
  4  name varchar(10)
  5  );
Table created.

SQL> create table std_soc
  2  (
  3  sid int,foreign key(sid) references Student(sid),
  4  sc_id int,foreign key(sc_id) references Society(sc_id)
  5  );
Table created.

SQL> create table course
  2  (
  3  cid int Primary Key,
  4  cname varchar(10)
  5  );
Table created.

SQL> create table Fac_course
  2  (
  3  fid int,foreign key(fid) references Faculty(fid),
  4  cid int,foreign key(cid) references Course(cid)
  5  );
Table created.
```

```

SQL*Plus

SQL> desc dept;
Name          Null?    Type
-----        -----   -----
DNO           NOT NULL NUMBER(38)
DNAME         VARCHAR2(10)
BUDGET        NUMBER(38)

SQL> desc faculty;
Name          Null?    Type
-----        -----   -----
FID           NOT NULL NUMBER(38)
NAME          VARCHAR2(10)
DNO           NUMBER(38)
SAL            NUMBER(38)
ADDRESS        VARCHAR2(15)
PHONE          NUMBER(38)
DOB             DATE
EXP            NUMBER(38)

SQL> desc student;
Name          Null?    Type
-----        -----   -----
SID           NOT NULL NUMBER(38)
NAME          VARCHAR2(10)
DNO           NUMBER(38)
PHONE          NUMBER(38)
DOB             DATE
ADDRESS        VARCHAR2(15)
SEM            NUMBER(38)

SQL> desc society;
Name          Null?    Type
-----        -----   -----
SC_ID          NOT NULL NUMBER(38)
NAME          VARCHAR2(10)

SQL> desc std_soc;
Name          Null?    Type
-----        -----   -----
SID            NUMBER(38)
SC_ID          NUMBER(38)

SQL> desc course;
Name          Null?    Type
-----        -----   -----
CID           NOT NULL NUMBER(38)
CNAME         VARCHAR2(10)

SQL> desc fac_course;
Name          Null?    Type
-----        -----   -----
FID            NUMBER(38)
CID            NUMBER(38)

```

- 2) Insert rows in each table (at least 5 records).

```

SQL> insert into dept values(1, 'English', 500000);
1 row created.

SQL> insert into dept values(2, 'CSE', 1000000);
1 row created.

SQL> insert into dept values(3, 'AIML', 800000);
1 row created.

SQL> insert into dept values(4, 'IIOT', 600000);
1 row created.

SQL> insert into dept values(5, 'Maths', 800000);
1 row created.

SQL> select * from dept;
      DNO    DNAME          BUDGET
-----  -----
      1 English        500000
      2 CSE           1000000
      3 AIML          800000
      4 IIOT          600000
      5 Maths          800000

```

SQL Plus

```

SQL> insert into faculty values(1, 'Aditi', 1, 40000, 'Rohini', 9999999999, '4-July-1995', 5);
1 row created.

SQL> insert into faculty values(2, 'Akshay', 2, 50000, 'Noida', 8888888888, '13-June-1994', 6);
1 row created.

SQL> insert into faculty values(3, 'Aman', 3, 60000, 'Dwarka', 7777777777, '2-October-1995', 5);
1 row created.

SQL> insert into faculty values(4, 'Salman', 4, 45000, 'Janakpuri', 6666666666, '1-April-1994', 6);
1 row created.

SQL> insert into faculty values(5, 'Poonam', 5, 50000, 'Rohini', 5555555555, '5-July-1996', 4);
1 row created.

SQL> select * from faculty;
      FID   NAME      DNO     SAL ADDRESS          PHONE DOB
-----  -----
      EXP
      1 Aditi      1     40000 Rohini        9999999999 04-JUL-95
      5
      2 Akshay     2     50000 Noida         8888888888 13-JUN-94
      6
      3 Aman       3     60000 Dwarka        7777777777 02-OCT-95
      5

      FID   NAME      DNO     SAL ADDRESS          PHONE DOB
-----  -----
      EXP
      4 Salman     4     45000 Janakpuri      6666666666 01-APR-94
      6
      5 Poonam     5     50000 Rohini        5555555555 05-JUL-96
      4

```

```

SQL> insert into student values(1, 'Kunsh', 3, 9990159801, '5-July-2005', 'Subhash Nagar', 4);
1 row created.

SQL> insert into student values(2, 'Raju', 1, 9990159352, '15-August-2004', 'Rohini', 6);
1 row created.

SQL> insert into student values(3, 'Geeta', 2, 9990159278, '26-January-2005', 'Pitampura', 6);
1 row created.

SQL> insert into student values(4, 'Akash', 5, 9990159972, '24-March-2004', 'Dwarka', 4);
1 row created.

SQL> insert into student values(5, 'Rachit', 4, 9990159292, '13-April-2006', 'Noida', 2);
1 row created.

SQL> select * from student;

  SID NAME          DNO    PHONE DOB        ADDRESS          SEM
-----  -----
  1 Kunsh           3 9990159801 05-JUL-05 Subhash Nagar      4
  2 Raju            1 9990159352 15-AUG-04 Rohini          6
  3 Geeta           2 9990159278 26-JAN-05 Pitampura        6
  4 Akash           5 9990159972 24-MAR-04 Dwarka          4
  5 Rachit          4 9990159292 13-APR-06 Noida          2

```

```

SQL> insert into society values(1, 'Art');

1 row created.

SQL> insert into society values(2, 'Singing');

1 row created.

SQL> insert into society values(3, 'Dancing');

1 row created.

SQL> insert into society values(4, 'Debatting');

1 row created.

SQL> insert into society values(5, 'Gaming');

1 row created.

SQL> select * from society;

  SC_ID NAME
-----  -----
    1 Art
    2 Singing
    3 Dancing
    4 Debatting
    5 Gaming

```

```

SQL> insert into std_soc values(1,1);
1 row created.

SQL> insert into std_soc values(2,2);
1 row created.

SQL> insert into std_soc values(3,3);
1 row created.

SQL> insert into std_soc values(4,4);
1 row created.

SQL> insert into std_soc values(5,5);
1 row created.

SQL> select * from std_soc;
      SID          SC_ID
-----  -----
        1              1
        2              2
        3              3
        4              4
        5              5

```

```

SQL> insert into course values(1, 'B Tech');
1 row created.

SQL> insert into course values(2, 'BJMC');
1 row created.

SQL> insert into course values(3, 'BBA');
1 row created.

SQL> insert into course values(4, 'BALLB');
1 row created.

SQL> insert into course values(5, 'B Com');
1 row created.

SQL> select * from course;
      CID          CNAME
-----  -----
        1      B Tech
        2      BJMC
        3      BBA
        4      BALLB
        5      B Com

```

```

SQL> insert into fac_course values(1,1);
1 row created.

SQL> insert into fac_course values(2,2);
1 row created.

SQL> insert into fac_course values(3,3);
1 row created.

SQL> insert into fac_course values(4,4);
1 row created.

SQL> insert into fac_course values(5,5);
1 row created.

SQL> select * from fac_course;
      FID          CID
      -----  -----
        1            1
        2            2
        3            3
        4            4
        5            5

```

- 3) Add a column no_of_awards to faculty table and update the values in the column

```

SQL> alter table faculty add no_of_awards int;
Table altered.

SQL> update faculty set no_of_awards=5 where fid=1;
1 row updated.

SQL> update faculty set no_of_awards=3 where fid=2;
1 row updated.

SQL> update faculty set no_of_awards=4 where fid=3;
1 row updated.

SQL> update faculty set no_of_awards=6 where fid=4;
1 row updated.

SQL> update faculty set no_of_awards=2 where fid=5;
1 row updated.

```

```
SQL> select * from faculty;
```

FID	NAME	DNO	SAL	ADDRESS	PHONE	DOB
EXP NO_OF_AWARDS						
1	Aditi	5	1	40000	Rohini	9999999999 04-JUL-95
2	Akshay	6	2	50000	Noida	8888888888 13-JUN-94
3	Aman	5	3	60000	Dwarka	7777777777 02-OCT-95
 EXP NO_OF_AWARDS						
4	Salman	6	4	45000	Janakpuri	6666666666 01-APR-94
5	Poonam	4	5	50000	Rohini	5555555555 05-JUL-96

4) Retrieve the faculties with exp greater than 10 years and sal<60000.

SQL> insert into faculty values(6, 'Puneet', 5, 40000, 'Rohini', 4444444444, '18-September-1986', 14, 8);						
1 row created.						
SQL> insert into faculty values(7, 'Mona', 4, 55000, 'Pitampura', 3333333333, '21-December-1987', 13, 7);						
1 row created.						
SQL> select * from faculty where exp>10 and sal<60000;						
FID	NAME	DNO	SAL	ADDRESS	PHONE	DOB
EXP NO_OF_AWARDS						
6	Puneet	5	40000	Rohini	4444444444	18-SEP-86
14		8				
7	Mona	4	55000	Pitampura	3333333333	21-DEC-87
13		7				

5) Retrieve the students enrolled in dno 2 or sem 4.

```
SQL> select * from student where dno=2 or sem=4;
```

SID	NAME	DNO	PHONE	DOB	ADDRESS	SEM
1	Kunsh	3	9990159801	05-JUL-05	Subhash Nagar	4
3	Geeta	2	9990159278	26-JAN-05	Pitampura	6
4	Akash	5	9990159972	24-MAR-04	Dwarka	4

6) Increase the faculties salary whose dno is 1.

```
SQL> update faculty set sal=sal+10000 where dno=1;
1 row updated.

SQL> select * from faculty where dno=1;

  FID NAME          DNO      SAL ADDRESS        PHONE DOB
----- -----
  EXP NO_OF_AWARDS
----- -----
    1 Aditi           1      50000 Rohini      99999999999 04-JUL-95
    5                 5
```

7) Increase the budget of AIML Department by 10000

```
SQL> update dept set budget=budget+10000 where dname='AIML';
1 row updated.

SQL> select * from dept where dname='AIML';

  DNO DNAME          BUDGET
----- -----
    3 AIML            810000
```

8) Update the phone no. of student id 21.

```
SQL> insert into student values(21, 'Rohan', 4, 9990100000, '15-May-2006', 'Moti Nagar', 2);
1 row created.

SQL> select * from student where sid=21;

  SID NAME          DNO      PHONE DOB        ADDRESS        SEM
----- -----
    21 Rohan          4 9990100000 15-MAY-06 Moti Nagar          2

SQL> update student set phone=9990159555 where sid=21;
1 row updated.

SQL> select * from student where sid=21;

  SID NAME          DNO      PHONE DOB        ADDRESS        SEM
----- -----
    21 Rohan          4 9990159555 15-MAY-06 Moti Nagar          2
```

- 9) Delete the society CODE (see if it allows (provided there are students enrolled))

```
SQL> update society set name='CODE' where sc_id=3;
1 row updated.

SQL> select * from society;
SC_ID NAME
-----
1 Art
2 Singing
3 CODE
4 Debating
5 Gaming

SQL> delete from society where name='CODE';
delete from society where name='CODE'
*
ERROR at line 1:
ORA-02292: integrity constraint (SYSTEM.SYS_C008336) violated - child record
found
```

- 10) Delete the student with no phone no.

```
SQL> insert into student values(15, 'Rahul', 4, null, '20-August-2005', 'Moti Bagh', 4);
1 row created.

SQL> select * from student where sid=15;
SID NAME          DNO      PHONE DOB      ADDRESS          SEM
-----  -----
15 Rahul           4        20-AUG-05 Moti Bagh          4

SQL> delete from student where phone is null;
1 row deleted.

SQL> select * from student;
SID NAME          DNO      PHONE DOB      ADDRESS          SEM
-----  -----
1 Kunsh            3        9990159801 05-JUL-05 Subhash Nagar    4
2 Raju             1        9990159352 15-AUG-04 Rohini          6
3 Geeta            2        9990159278 26-JAN-05 Pitampura       6
4 Akash            5        9990159972 24-MAR-04 Dwarka          4
5 Rachit           4        9990159292 13-APR-06 Noida           2
21 Rohan           4        9990159555 15-MAY-06 Moti Nagar       2

6 rows selected.
```

Learning Outcome:

EXPERIMENT 4

Problem statement: Implement queries including various functions- mathematical, string, date etc.

Theory:

KUNSH SABHARWAL

SQL Commands:

1) Mathematical Functions:

- a. SQRT()
- b. SIN() / COS() / TAN()
- c. ABS()
- d. CEIL()
- e. FLOOR()
- f. ROUND()

```
SQL*Plus

SQL> SELECT SQRT(100) "Square root" FROM DUAL;
Square root
-----
          10

SQL> SELECT SIN(30 * 3.14159265359/180) "Sine of 30 degrees" FROM DUAL;
Sine of 30 degrees
-----
           .5

SQL> SELECT COS(60 * 3.14159265359/180) "Cos of 60 degrees" FROM DUAL;
Cos of 60 degrees
-----
           .5

SQL> SELECT TAN(45 * 3.14159265359/180) "Tan of 45 degrees" FROM DUAL;
Tan of 45 degrees
-----
           1

SQL> SELECT ABS(-28) "Absolute" FROM DUAL;
Absolute
-----
          28

SQL> SELECT CEIL(9.4) from dual;
CEIL(9.4)
-----
         10

SQL> SELECT floor(9.4) from dual;
FLOOR(9.4)
-----
          9

SQL> SELECT ROUND(15.193,1) "Round" FROM DUAL;
Round
-----
       15.2

SQL> SELECT ROUND(15.193,2) "Round" FROM DUAL;
Round
-----
      15.19
```

2) String Functions:

- a. ASCII()
- b. CHR()
- c. LENGTH()
- d. CONCAT()
- e. LOWER() / UPPER()
- f. SUBSTR()

```
SQL*Plus

SQL> SELECT ASCII('A') "ASCII 'A'" FROM DUAL;
      ASCII 'A'
      -----
      65

SQL> SELECT ASCII('a') "ASCII 'a'" FROM DUAL;
      ASCII 'a'
      -----
      97

SQL> SELECT CHR(98) FROM DUAL;
      C
      -
      B

SQL> SELECT CHR(66) FROM DUAL;
      C
      -
      B

SQL> SELECT LENGTH('KUNSH') "Length in characters" FROM DUAL;
      Length in characters
      -----
      5

SQL> SELECT LENGTH('      KUNSH      ') "Length in characters" FROM DUAL;
      Length in characters
      -----
      11

SQL> select concat('Kunsh','AIML - A') from dual;
      CONCAT('KUNSH
      -----
      KunshAIML - A

SQL> select lower('KUNSH') from dual;
      LOWER
      -----
      kunsh

SQL> select upper('kunsh') from dual;
      UPPER
      -----
      KUNSH

SQL> select substr('ABCDEF',3,4) from dual;
      SUBS
      -----
      CDEF
```

3) Date Functions:

- a. CURRENT_DATE()
- b. CURRENT_TIMESTAMP()
- c. MONTHS_BETWEEN()
- d. SYSDATE()
- e. LAST_DAY()
- f. NEXT_DAY()

```
SQL*Plus X + v

SQL> SELECT CURRENT_DATE FROM DUAL;

CURRENT_DATE
-----
10-FEB-2025 00:27:06

SQL> SELECT CURRENT_TIMESTAMP FROM DUAL;

CURRENT_TIMESTAMP
-----
10-FEB-25 12.27.08.727000 AM +05:30

SQL> SELECT MONTHS_BETWEEN(TO_DATE('02-01-2025','MM-DD-YYYY'),(TO_DATE('01-01-2025','MM-DD-YYYY'))) FROM DUAL;
MONTHS_BETWEEN(TO_DATE('02-01-2025','MM-DD-YYYY'),(TO_DATE('01-01-2025','MM-DD-Y
-----
1

SQL> SELECT SYSDATE FROM DUAL;

SYSDATE
-----
10-FEB-2025 00:27:17

SQL> SELECT LAST_DAY(SYSDATE) FROM DUAL;

LAST_DAY(SYSDATE)
-----
28-FEB-2025 00:27:48

SQL> SELECT NEXT_DAY('10-FEB-2025','FRIDAY') "NEXT DAY" FROM DUAL;

NEXT DAY
-----
14-FEB-2025 00:00:00
```

4) Aggregate Functions:

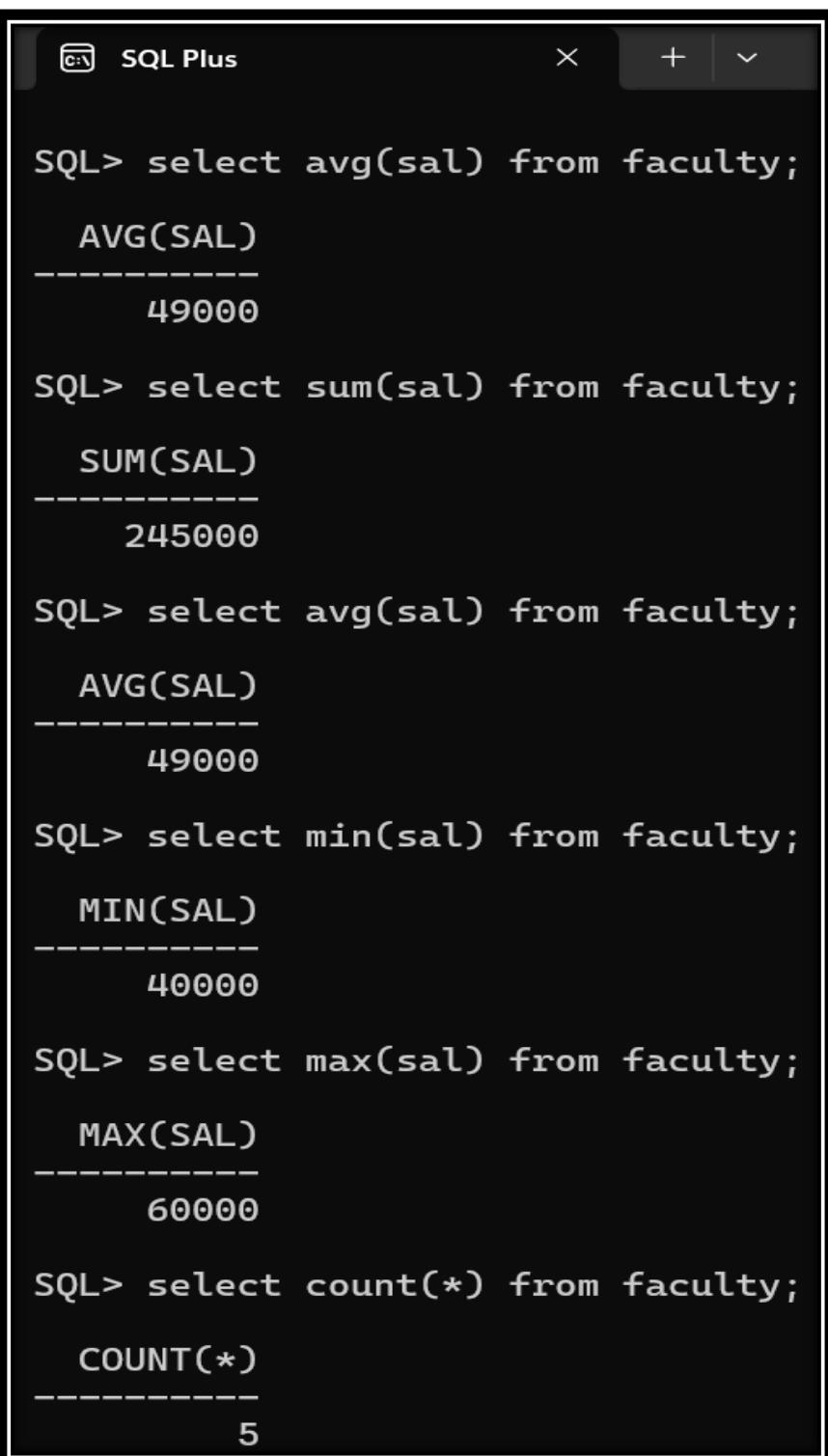
- a. SUM()
- b. AVG()
- c. MIN()
- d. MAX()
- e. COUNT(*)

```
SQL> select * from faculty;

      FID NAME          DNO      SAL ADDRESS        PHONE DOB
-----+-----+-----+-----+-----+-----+-----+-----+
      EXP NO_OF_AWARDS
-----+-----+-----+-----+-----+-----+-----+-----+
      1 Aditi           1     40000 Rohini       9999999999 04-JUL-95
      5
      2 Akshay          2     50000 Noida       8888888888 13-JUN-94
      6
      3 Aman            3     60000 Dwarka      7777777777 02-OCT-95
      5

      FID NAME          DNO      SAL ADDRESS        PHONE DOB
-----+-----+-----+-----+-----+-----+-----+-----+
      EXP NO_OF_AWARDS
-----+-----+-----+-----+-----+-----+-----+-----+
      4 Salman          4     45000 Janakpuri   6666666666 01-APR-94
      6
      5 Poonam          5     50000 Rohini       5555555555 05-JUL-96
      4

SQL> desc faculty;
      Name          Null?    Type
-----+-----+-----+-----+
      FID          NOT NULL NUMBER(38)
      NAME         VARCHAR2(10)
      DNO          NUMBER(38)
      SAL          NUMBER(38)
      ADDRESS      VARCHAR2(15)
      PHONE        NUMBER(38)
      DOB          DATE
      EXP          NUMBER(38)
      NO_OF_AWARDS NUMBER(38)
```



The screenshot shows a terminal window titled "SQL Plus" displaying several SQL queries and their results. The queries are as follows:

```
SQL> select avg(sal) from faculty;
      AVG(SAL)
      -----
      49000

SQL> select sum(sal) from faculty;
      SUM(SAL)
      -----
      245000

SQL> select avg(sal) from faculty;
      AVG(SAL)
      -----
      49000

SQL> select min(sal) from faculty;
      MIN(SAL)
      -----
      40000

SQL> select max(sal) from faculty;
      MAX(SAL)
      -----
      60000

SQL> select count(*) from faculty;
      COUNT(*)
      -----
      5
```

Learning Outcome:

EXPERIMENT 5

Problem statement: Implement queries including Sorting, Grouping and Subqueries- like any, all, exists, not exists.

Theory:

SQL Commands:

- 1) Create the below mentioned tables with proper constraints: -
 - a. Faculty (fid, name, *dno*, sal, address, phone, dob, exp)
 - b. Dept (dno, dname, budget)

```
SQL> create table Dept
  2  (
  3   dno int Primary Key,
  4   dname varchar(10),
  5   budget int
  6  );

Table created.

SQL> create table Faculty
  2  (
  3   fid int Primary Key,
  4   name varchar(10),
  5   dno int,foreign key(dno) references Dept(dno),
  6   sal int,
  7   address varchar(15),
  8   phone int,
  9   dob date,
 10  exp int
 11  );

Table created.
```

```
SQL> insert into dept values(&1, '&2', &3);
Enter value for 1: 10
Enter value for 2: AIML
Enter value for 3: 5000
old    1: insert into dept values(&1, '&2', &3)
new    1: insert into dept values(10, 'AIML', 5000)

1 row created.

SQL> insert into dept values(&1, '&2', &3);
Enter value for 1: 20
Enter value for 2: AIDS
Enter value for 3: 7000
old    1: insert into dept values(&1, '&2', &3)
new    1: insert into dept values(20, 'AIDS', 7000)

1 row created.
```

```
SQL> insert into dept values(&1, '&2', &3);
Enter value for 1: 30
Enter value for 2: IIOT
Enter value for 3: 8000
old    1: insert into dept values(&1, '&2', &3)
new    1: insert into dept values(30, 'IIOT', 8000)

1 row created.

SQL> insert into dept values(&1, '&2', &3);
Enter value for 1: 40
Enter value for 2: CSE
Enter value for 3: 9000
old    1: insert into dept values(&1, '&2', &3)
new    1: insert into dept values(40, 'CSE', 9000)

1 row created.

SQL> insert into dept values(&1, '&2', &3);
Enter value for 1: 50
Enter value for 2: IT
Enter value for 3: 12000
old    1: insert into dept values(&1, '&2', &3)
new    1: insert into dept values(50 , 'IT', 12000)

1 row created.
```

```
SQL> insert into faculty values(&1,'&2',&3,&4,'&5',&6,'&7',&8);
Enter value for 1: 1
Enter value for 2: Puneet
Enter value for 3: 10
Enter value for 4: 5000
Enter value for 5: Paschim Vihar
Enter value for 6: 98123
Enter value for 7: 01-March-85
Enter value for 8: 14
old    1: insert into faculty values(&1,'&2',&3,&4,'&5',&6,'&7',&8)
new    1: insert into faculty values(1,'Puneet',10,5000,'Paschim Vihar',98123,'01-March-85',14)

1 row created.
```

```
SQL> insert into faculty values(&1,'&2',&3,&4,'&5',&6,'&7',&8);
Enter value for 1: 2
Enter value for 2: Ashish
Enter value for 3: 10
Enter value for 4: 5000
Enter value for 5: Tilak Nagar
Enter value for 6: 98124
Enter value for 7: 07-November-85
Enter value for 8: 13
old  1: insert into faculty values(&1,'&2',&3,&4,'&5',&6,'&7',&8)
new  1: insert into faculty values(2,'Ashish',10,5000,'Tilak Nagar',98124,'07-November-85',13)

1 row created.

SQL> insert into faculty values(&1,'&2',&3,&4,'&5',&6,'&7',&8);
Enter value for 1: 3
Enter value for 2: Ishleen
Enter value for 3: 20
Enter value for 4: 6000
Enter value for 5: Rajouri
Enter value for 6: 98174
Enter value for 7: 08-December-85
Enter value for 8: 5
old  1: insert into faculty values(&1,'&2',&3,&4,'&5',&6,'&7',&8)
new  1: insert into faculty values(3,'Ishleen',20,6000,'Rajouri',98174,'08-December-85',5)

1 row created.

SQL> insert into faculty values(&1,'&2',&3,&4,'&5',&6,'&7',&8);
Enter value for 1: 4
Enter value for 2: Sonakshi
Enter value for 3: 30
Enter value for 4: 2000
Enter value for 5: Pitampura
Enter value for 6: NULL
Enter value for 7: 23-June-85
Enter value for 8: 8
old  1: insert into faculty values(&1,'&2',&3,&4,'&5',&6,'&7',&8)
new  1: insert into faculty values(4,'Sonakshi',30,2000,'Pitampura',NULL,'23-June-85',8)

1 row created.
```

```

SQL> insert into faculty values(&1,'&2',&3,&4,'&5',&6,'&7',&8);
Enter value for 1: 5
Enter value for 2: Suman
Enter value for 3: 20
Enter value for 4: 1500
Enter value for 5: ITO
Enter value for 6: 98764
Enter value for 7: 19-April-85
Enter value for 8: 19
old  1: insert into faculty values(&1,'&2',&3,&4,'&5',&6,'&7',&8)
new  1: insert into faculty values(5,'Suman',20,1500,'ITO',98764,'19-April-85',19)

1 row created.

SQL> insert into faculty values(&1,'&2',&3,&4,'&5',&6,'&7',&8);
Enter value for 1: 6
Enter value for 2: Gopal
Enter value for 3: 40
Enter value for 4: 9000
Enter value for 5: Vikaspuri
Enter value for 6: 98960
Enter value for 7: 10-March-85
Enter value for 8: 12
old  1: insert into faculty values(&1,'&2',&3,&4,'&5',&6,'&7',&8)
new  1: insert into faculty values(6,'Gopal',40,9000,'Vikaspuri',98960,'10-March-85',12)

1 row created.

SQL> insert into faculty values(&1,'&2',&3,&4,'&5',&6,'&7',&8);
Enter value for 1: 7
Enter value for 2: Archa
Enter value for 3: 40
Enter value for 4: 6000
Enter value for 5: Hari Nagar
Enter value for 6: NULL
Enter value for 7: 07-March-85
Enter value for 8: 9
old  1: insert into faculty values(&1,'&2',&3,&4,'&5',&6,'&7',&8)
new  1: insert into faculty values(7,'Archa',40,6000,'Hari Nagar',NULL,'07-March-85',9)

1 row created.

```

DNO	DNAME	BUDGET
10	AIML	5000
20	AIDS	7000
30	IIOT	8000
40	CSE	9000
50	IT	12000

SQL Plus

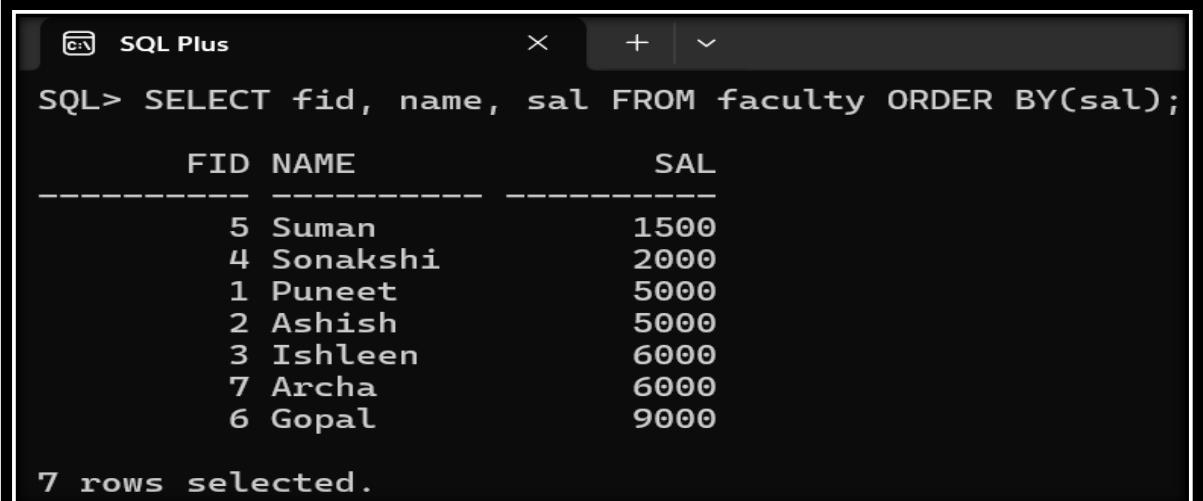
X + ▾

SQL> select * from faculty;

FID	NAME	DNO	SAL	ADDRESS	PHONE	DOB
EXP						
1	Puneet	10	5000	Paschim Vihar	98123	01-MAR-85
14						
2	Ashish	10	5000	Tilak Nagar	98124	07-NOV-85
13						
3	Ishleen	20	6000	Rajouri	98174	08-DEC-85
5						
FID	NAME	DNO	SAL	ADDRESS	PHONE	DOB
EXP						
4	Sonakshi	30	2000	Pitampura	23-JUN-85	
8						
5	Suman	20	1500	ITO	98764	19-APR-85
19						
6	Gopal	40	9000	Vikaspuri	98960	10-MAR-85
12						
FID	NAME	DNO	SAL	ADDRESS	PHONE	DOB
EXP						
7	Archa	40	6000	Hari Nagar	07-MAR-85	
9						

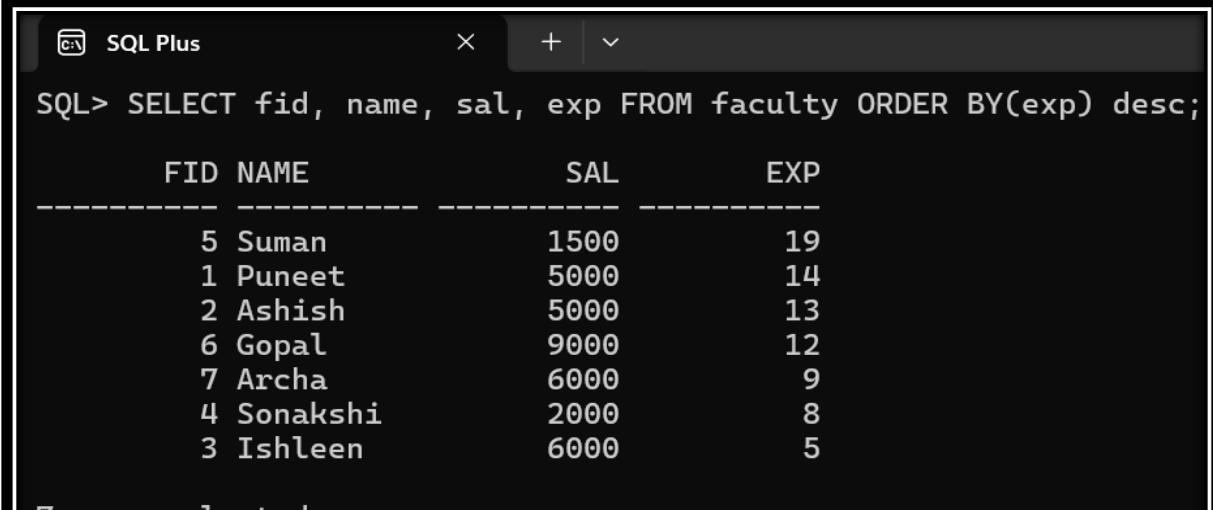
7 rows selected.

- 2) Retrieve the ID, Name and Salary of Faculty in increasing order of Salary.



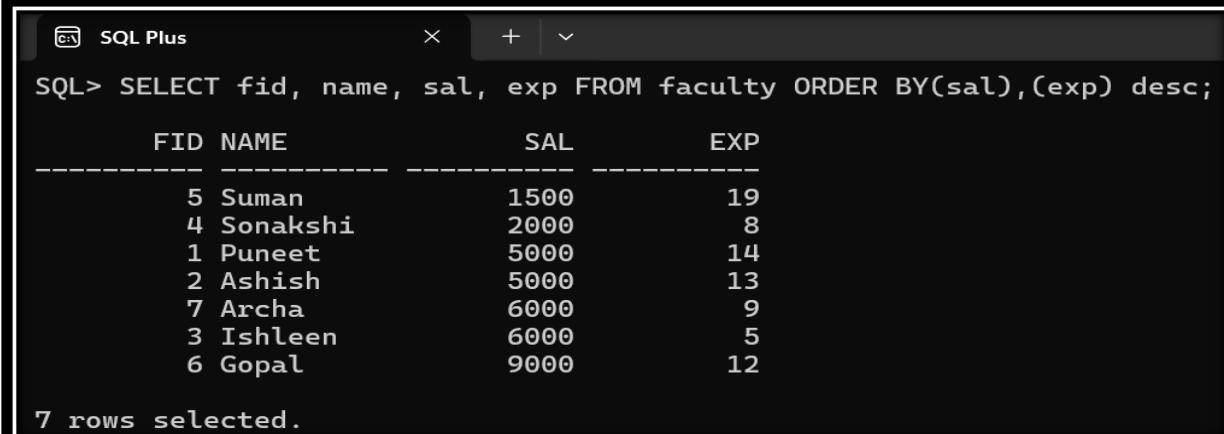
```
SQL> SELECT fid, name, sal FROM faculty ORDER BY(sal);
      FID NAME          SAL
-----  -----
      5  Suman        1500
      4  Sonakshi    2000
      1  Puneet       5000
      2  Ashish       5000
      3  Ishleen      6000
      7  Archa        6000
      6  Gopal        9000
7 rows selected.
```

- 3) Retrieve the ID, Name, Salary and Experience of Faculty in decreasing order of Experience.



```
SQL> SELECT fid, name, sal, exp FROM faculty ORDER BY(exp) desc;
      FID NAME          SAL      EXP
-----  -----
      5  Suman        1500     19
      1  Puneet       5000     14
      2  Ashish       5000     13
      6  Gopal        9000     12
      7  Archa        6000      9
      4  Sonakshi    2000      8
      3  Ishleen      6000      5
7 rows selected.
```

- 4) Retrieve the ID, Name, Salary and Experience of Faculty in increasing order of Salary and Decreasing order of Experience.



```
SQL> SELECT fid, name, sal, exp FROM faculty ORDER BY(sal), (exp) desc;
      FID NAME          SAL      EXP
-----  -----
      5  Suman        1500     19
      4  Sonakshi    2000      8
      1  Puneet       5000     14
      2  Ashish       5000     13
      7  Archa        6000      9
      3  Ishleen      6000      5
      6  Gopal        9000     12
7 rows selected.
```

5) Show the average experience department wise

```
SQL> SELECT dno, AVG(exp) FROM faculty GROUP BY(dno);

      DNO      AVG(EXP)
----- -----
        10       13.5
        20        12
        30         8
        40      10.5
```

6) Retrieve the maximum salary department wise.

```
SQL> SELECT dno, MAX(sal) FROM faculty GROUP BY(dno);

      DNO      MAX(SAL)
----- -----
        10      5000
        20      6000
        30      2000
        40      9000
```

7) Retrieve the average salary department wise having average salary greater than 5000.

```
SQL> SELECT dno, AVG(sal) FROM faculty GROUP BY(dno) HAVING AVG(sal)>5000

      DNO      AVG(SAL)
----- -----
        40      7500
```

8) Retrieve faculties details where name (table should have names accordingly): -

a) Starting with 'S'

```
SQL> SELECT * FROM faculty WHERE name LIKE 'S%';

      FID NAME          DNO      SAL ADDRESS           PHONE DOB
----- -----
      EXP
----- -----
        4 Sonakshi        30     2000 Pitampura      23-JUN-85
        8
----- -----
        5 Suman           20     1500 ITO          98764 19-APR-85
        19
```

b) Ending with 'n'

SQL> SELECT * FROM faculty WHERE name LIKE '%n';						
FID	NAME	DNO	SAL	ADDRESS	PHONE	DOB
<hr/>						
EXP						
3	Ishleen	20	6000	Rajouri	98174	08-DEC-85
5						
5	Suman	20	1500	ITO	98764	19-APR-85
19						

c) Starting with 'a' and ending with 'h'

SQL> SELECT * FROM faculty WHERE name LIKE 'A%h';						
FID	NAME	DNO	SAL	ADDRESS	PHONE	DOB
<hr/>						
EXP						
2	Ashish	10	5000	Tilak Nagar	98124	07-NOV-85
13						

d) Must contain 'a'

SQL> SELECT * FROM faculty WHERE name LIKE '%a%';						
FID	NAME	DNO	SAL	ADDRESS	PHONE	DOB
<hr/>						
EXP						
4	Sonakshi	30	2000	Pitampura	23-JUN-85	
8						
5	Suman	20	1500	ITO	98764	19-APR-85
19						
6	Gopal	40	9000	Vikaspuri	98960	10-MAR-85
12						
FID	NAME	DNO	SAL	ADDRESS	PHONE	DOB
EXP						
7	Archa	40	6000	Hari Nagar	07-MAR-85	
9						

e) Should have 5 letters only starting with 'g'

SQL> SELECT * FROM faculty WHERE name LIKE 'G____';						
FID	NAME	DNO	SAL	ADDRESS	PHONE	DOB
<hr/>						
EXP						
6	Gopal	40	9000	Vikaspuri	98960	10-MAR-85
12						

9) Show the use of ANY/ALL clauses.

```
SQL> SELECT * FROM dept WHERE dno = ANY (SELECT dno FROM faculty WHERE dno<30);
```

DNO	DNAME	BUDGET
10	AIML	5000
20	AIDS	7000

```
SQL> SELECT * FROM faculty WHERE dno = ANY (SELECT dno FROM dept WHERE dno>20);
```

FID	NAME	DNO	SAL	ADDRESS	PHONE	DOB
	EXP					
4	Sonakshi	30	2000	Pitampura		23-JUN-85
8						
6	Gopal	40	9000	Vikaspuri	98960	10-MAR-85
12						
7	Archa	40	6000	Hari Nagar		07-MAR-85
9						

```
SQL> SELECT * FROM dept WHERE dno > ALL (SELECT dno FROM faculty WHERE dno<30);
```

DNO	DNAME	BUDGET
30	IIOT	8000
40	CSE	9000
50	IT	12000

```
SQL> SELECT * FROM faculty WHERE dno < ALL (SELECT dno FROM dept WHERE dno>20);
```

FID	NAME	DNO	SAL	ADDRESS	PHONE	DOB
	EXP					
5	Suman	20	1500	IT0	98764	19-APR-85
19						
3	Ishleen	20	6000	Rajouri	98174	08-DEC-85
5						
2	Ashish	10	5000	Tilak Nagar	98124	07-NOV-85
13						

FID	NAME	DNO	SAL	ADDRESS	PHONE	DOB
	EXP					
1	Puneet	10	5000	Paschim Vihar	98123	01-MAR-85
14						

10) Show the use of EXISTS/NOT EXISTS clauses

```
SQL> SELECT fid, name, dno FROM faculty WHERE EXISTS(SELECT dno FROM dept WHERE dno>40);

    FID NAME          DNO
----- -----
    1  Puneet        10
    2  Ashish        10
    3  Ishleen       20
    4  Sonakshi      30
    5  Suman         20
    6  Gopal          40
    7  Archa          40

7 rows selected.

SQL> SELECT fid, name, dno FROM faculty WHERE EXISTS(SELECT dno FROM dept WHERE dno>50);

no rows selected
```

```
SQL> SELECT * FROM dept WHERE NOT EXISTS(SELECT dno FROM faculty WHERE dno<40);

no rows selected

SQL> SELECT * FROM dept WHERE NOT EXISTS(SELECT dno FROM faculty WHERE dno>40);

    DNO DNAME          BUDGET
----- -----
    10  AIML           5000
    20  AIDS           7000
    30  IIOT           8000
    40  CSE            9000
    50  IT             12000
```

11) Find the names of Faculty who work in department AIML (using subquery)

```
SQL> SELECT name FROM faculty WHERE dno IN (SELECT dno FROM dept where dname='AIML');

NAME
-----
Puneet
Ashish
```

Learning Outcome:

EXPERIMENT 6

Problem statement: Implement queries including various Set operations (Union, Intersection, Except etc.).

Theory:

SQL Commands:

- 1) Create a table named section with the following columns and add appropriate values to it:
 - a. Section (course_id, sec_id, semester, year, building, room_number, time_slot_id)

```
SQL> CREATE TABLE Section(
  2  course_id varchar(10),
  3  sec_id int,
  4  semester varchar(10),
  5  year int,
  6  building varchar(10),
  7  room_number int,
  8  time_slot_id varchar(1)
  9 );
```

```
SQL> desc section;
Name          Null?    Type
----- -----
COURSE_ID           VARCHAR2(10)
SEC_ID             NUMBER(38)
SEMESTER          VARCHAR2(10)
YEAR              NUMBER(38)
BUILDING          VARCHAR2(10)
ROOM_NUMBER        NUMBER(38)
TIME_SLOT_ID       VARCHAR2(1)
```

```
SQL*Plus  
X +   
  
SQL> INSERT INTO section VALUES('BIO-101', 1, 'Summer', 2017, 'Painter', 514, 'B');  
1 row created.  
  
SQL> INSERT INTO section VALUES('BIO-301', 1, 'Summer', 2018, 'Painter', 514, 'A');  
1 row created.  
  
SQL> INSERT INTO section VALUES('CS-101', 1, 'Fall', 2017, 'Packard', 101, 'H');  
1 row created.  
  
SQL> INSERT INTO section VALUES('CS-101', 1, 'Spring', 2018, 'Packard', 101, 'F');  
1 row created.  
  
SQL> INSERT INTO section VALUES('CS-190', 1, 'Spring', 2017, 'Taylor', 3128, 'E');  
1 row created.  
  
SQL> INSERT INTO section VALUES('CS-190', 2, 'Spring', 2017, 'Taylor', 3128, 'A');  
1 row created.  
  
SQL> INSERT INTO section VALUES('CS-315', 1, 'Spring', 2018, 'Watson', 120, 'D');  
1 row created.  
  
SQL> INSERT INTO section VALUES('CS-319', 1, 'Spring', 2018, 'Watson', 100, 'B');  
1 row created.  
  
SQL> INSERT INTO section VALUES('CS-319', 2, 'Spring', 2018, 'Taylor', 3128, 'C');  
1 row created.  
  
SQL> INSERT INTO section VALUES('CS-347', 1, 'Fall', 2017, 'Taylor', 3128, 'A');  
1 row created.  
  
SQL> INSERT INTO section VALUES('EE-181', 1, 'Spring', 2017, 'Taylor', 3128, 'C');  
1 row created.  
  
SQL> INSERT INTO section VALUES('FIN-201', 1, 'Spring', 2018, 'Packard', 101, 'B');  
1 row created.  
  
SQL> INSERT INTO section VALUES('HIS-351', 1, 'Spring', 2018, 'Painter', 514, 'C');  
1 row created.  
  
SQL> INSERT INTO section VALUES('MU-199', 1, 'Spring', 2018, 'Packard', 101, 'D');  
1 row created.  
  
SQL> INSERT INTO section VALUES('PHY-101', 1, 'Fall', 2017, 'Watson', 100, 'A');  
1 row created.
```

```
SQL*Plus
```

```
SQL> select * from section;
```

COURSE_ID	SEC_ID	SEMESTER	YEAR	BUILDING	ROOM_NUMBER	T
BIO-101	1	Summer	2017	Painter	514	B
BIO-301	1	Summer	2018	Painter	514	A
CS-101	1	Fall	2017	Packard	101	H
CS-101	1	Spring	2018	Packard	101	F
CS-190	1	Spring	2017	Taylor	3128	E
CS-190	2	Spring	2017	Taylor	3128	A
CS-315	1	Spring	2018	Watson	120	D
CS-319	1	Spring	2018	Watson	100	B
CS-319	2	Spring	2018	Taylor	3128	C
CS-347	1	Fall	2017	Taylor	3128	A
EE-181	1	Spring	2017	Taylor	3128	C

COURSE_ID	SEC_ID	SEMESTER	YEAR	BUILDING	ROOM_NUMBER	T
FIN-201	1	Spring	2018	Packard	101	B
HIS-351	1	Spring	2018	Painter	514	C
MU-199	1	Spring	2018	Packard	101	D
PHY-101	1	Fall	2017	Watson	100	A

```
15 rows selected.
```

2) Find courses that ran in Fall 2017.

```
SQL> SELECT course_id FROM section WHERE semester='Fall' AND year=2017;
```

COURSE_ID
CS-101
CS-347
PHY-101

3) Find the courses that ran in Spring 2018.

```
SQL> SELECT course_id FROM section WHERE semester='Spring' AND year=2018;
```

COURSE_ID
CS-101
CS-315
CS-319
CS-319
FIN-201
HIS-351
MU-199

```
7 rows selected.
```

4) Find courses that ran in Fall 2017 or in Spring 2018.

```
SQL> SELECT course_id FROM section WHERE semester='Fall' AND year=2017 UNION SELECT course_id FROM section WHERE semester='Spring' AND year=2018;

COURSE_ID
-----
CS-101
CS-347
PHY-101
CS-315
CS-319
FIN-201
HIS-351
MU-199

8 rows selected.
```

5) Find courses that ran in Fall 2017 and in Spring 2018.

```
SQL> SELECT course_id FROM section WHERE semester='Fall' AND year=2017 INTERSECT SELECT course_id FROM section WHERE semester='Spring' AND year=2018;

COURSE_ID
-----
CS-101
```

6) Find courses that ran in Fall 2017 but not in Spring 2018.

```
SQL> SELECT course_id FROM section WHERE semester='Fall' AND year=2017 MINUS SELECT course_id FROM section WHERE semester='Spring' AND year=2018;

COURSE_ID
-----
CS-347
PHY-101
```

7) Find courses that ran in Spring 2018 but not in Fall 2017.

```
SQL> SELECT course_id FROM section WHERE semester='Spring' AND year=2018 MINUS SELECT course_id FROM section WHERE semester='Fall' AND year=2017;

COURSE_ID
-----
CS-315
CS-319
FIN-201
HIS-351
MU-199
```

8) Create 2 different tables with same number of columns and make use of Union, Union all, Intersect and Except.

- a. Building (id, name)
- b. Construction (id, name)

```
SQL> CREATE TABLE building(
  2      id int,
  3      name varchar(10)
  4 );
```

Table created.

```
SQL> CREATE TABLE construction(
  2      id int,
  3      name varchar(10)
  4 );
```

Table created.

```
SQL> INSERT INTO building VALUES(1, 'Emporio');
```

1 row created.

```
SQL> INSERT INTO building VALUES(2, 'Unity-one');
```

1 row created.

```
SQL> INSERT INTO building VALUES(3, 'Pacific');
```

1 row created.

```
SQL> INSERT INTO building VALUES(4, 'Promenade');
```

1 row created.

```
SQL> INSERT INTO building VALUES(5, 'D-Mall');
```

1 row created.

```
SQL> select * from building;
```

ID	NAME
1	Emporio
2	Unity-one
3	Pacific
4	Promenade
5	D-Mall

```
SQL> INSERT INTO construction VALUES(2,'Unity-one');

1 row created.

SQL> INSERT INTO construction VALUES(5,'D-Mall');

1 row created.

SQL> INSERT INTO construction VALUES(6,'DLF');

1 row created.

SQL> INSERT INTO construction VALUES(7,'Mohan');

1 row created.

SQL> INSERT INTO construction VALUES(8,'DLF');

1 row created.

SQL> select * from construction;

      ID NAME
----- -----
        2 Unity-one
        5 D-Mall
        6 DLF
        7 Mohan
        8 DLF
```

```
SQL> SELECT * FROM building UNION SELECT * FROM construction;

      ID NAME
----- -----
        1 Emporio
        2 Unity-one
        3 Pacific
        4 Promenade
        5 D-Mall
        6 DLF
        7 Mohan
        8 DLF

8 rows selected.
```

```
SQL> SELECT * FROM building UNION ALL SELECT * FROM construction;
```

ID	NAME
1	Emporio
2	Unity-one
3	Pacific
4	Promenade
5	D-Mall
2	Unity-one
5	D-Mall
6	DLF
7	Mohan
8	DLF

```
10 rows selected.
```

```
SQL> SELECT * FROM building INTERSECT SELECT * FROM construction;
```

ID	NAME
2	Unity-one
5	D-Mall

```
SQL> SELECT * FROM building MINUS SELECT * FROM construction;
```

ID	NAME
1	Emporio
3	Pacific
4	Promenade

Learning Outcome:

EXPERIMENT 7

Problem statement: Implement various JOIN operations- (Inner, Outer).

Theory:

SQL Commands:

- 1) Create the following tables
 - a. Employee – (*empid*, *empname*, *deptid*)
 - b. Department – (*deptid*, *deptname*)

```
SQL> CREATE TABLE Employee(  
 2      empid int,  
 3      empname varchar(10),  
 4      deptid int PRIMARY KEY  
 5 );  
  
Table created.  
  
SQL> CREATE TABLE Department(  
 2      deptid int, FOREIGN KEY (deptid) REFERENCES Employee(deptid),  
 3      deptname varchar(10)  
 4 );  
  
Table created.
```

- 2) Insert values in the above tables.

```
SQL> INSERT INTO Employee VALUES(101, 'Ram', 1);  
  
1 row created.  
  
SQL> INSERT INTO Employee VALUES(102, 'Raju', 2);  
  
1 row created.  
  
SQL> INSERT INTO Employee VALUES(103, 'Shyam', 3);  
  
1 row created.  
  
SQL> INSERT INTO Employee VALUES(104, 'Krishna', 4);  
  
1 row created.  
  
SQL> INSERT INTO Employee VALUES(105, 'Rajat', 5);  
  
1 row created.
```

```
SQL> INSERT INTO Department VALUES(1, 'AIML');
1 row created.

SQL> INSERT INTO Department VALUES(2, 'AIDS');
1 row created.

SQL> INSERT INTO Department VALUES(3, 'CSE');
1 row created.

SQL> INSERT INTO Department VALUES(4, 'VLSI');
1 row created.

SQL> INSERT INTO Department VALUES(5, 'IIOT');
1 row created.
```

```
SQL> select * from employee;
```

EMPID	EMPNAME	DEPTID
101	Ram	1
102	Raju	2
103	Shyam	3
104	Krishna	4
105	Rajat	5

```
SQL> select * from department;
```

DEPTID	DEPTNAME
1	AIML
2	AIDS
3	CSE
4	VLSI
5	IIOT

3) Implement INNER JOIN on the tables.

```
SQL> SELECT employee.empname,department.deptname FROM employee INNER JOIN department ON employee.deptid=department.deptid;
```

EMPNAME	DEPTNAME
Ram	AIML
Raju	AIDS
Shyam	CSE
Krishna	VLSI
Rajat	IIOT

```
SQL> SELECT e.empname,d.deptname FROM employee e INNER JOIN department d ON e.deptid=d.deptid;
```

EMPNAME	DEPTNAME
Ram	AIML
Raju	AIDS
Shyam	CSE
Krishna	VLSI
Rajat	IIOT

4) Implement LEFT JOIN (OUTER JOIN) on the tables.

```
SQL> INSERT INTO Employee (empname, deptid) VALUES('Sakshi', 6);
```

1 row created.

```
SQL> INSERT INTO Employee (empname, deptid) VALUES('Rakesh', 7);
```

1 row created.

```
SQL> INSERT INTO Employee (empname, deptid) VALUES('Suraj', 8);
```

1 row created.

```
SQL> select * from employee;
```

EMPID	EMPNAME	DEPTID
101	Ram	1
102	Raju	2
103	Shyam	3
104	Krishna	4
105	Rajat	5
	Sakshi	6
	Rakesh	7
	Suraj	8

```
SQL> INSERT INTO Department (deptid) VALUES(6);
```

1 row created.

```
SQL> INSERT INTO Department (deptid) VALUES(7);
```

1 row created.

```
SQL> select * from department;
```

DEPTID	DEPTNAME
1	AIML
2	AIDS
3	CSE
4	VLSI
5	IIOT
6	
7	

```
SQL> SELECT e.empid,e.empname,d.deptname,d.deptid FROM employee e LEFT JOIN department d ON e.deptid=d.deptid;
```

EMPID	EMPNAME	DEPTNAME	DEPTID
101	Ram	AIML	1
102	Raju	AIDS	2
103	Shyam	CSE	3
104	Krishna	VLSI	4
105	Rajat	IIOT	5
	Sakshi		6
	Rakesh		7
	Suraj		

8 rows selected.

5) Implement RIGHT JOIN (OUTER JOIN) on the tables.

```
SQL> SELECT e.empid,e.empname,d.deptname,d.deptid FROM employee e RIGHT JOIN department d ON e.deptid=d.deptid;
```

EMPID	EMPNAME	DEPTNAME	DEPTID
101	Ram	AIML	1
102	Raju	AIDS	2
103	Shyam	CSE	3
104	Krishna	VLSI	4
105	Rajat	IIOT	5
	Sakshi		6
	Rakesh		7

7 rows selected.

6) Implement FULL JOIN (OUTER JOIN) on the tables.

```
SQL> SELECT e.empid,e.empname,d.deptname,d.deptid FROM employee e FULL JOIN department d ON e.deptid=d.deptid;
```

EMPID	EMPNAME	DEPTNAME	DEPTID
101	Ram	AIML	1
102	Raju	AIDS	2
103	Shyam	CSE	3
104	Krishna	VLSI	4
105	Rajat	IIOT	5
	Sakshi		6
	Rakesh		7
	Suraj		

8 rows selected.

7) Implement THETA and EQUI Join (INNER JOIN).

```
SQL> CREATE TABLE car(
  2 CarModel varchar(10),
  3 CarPrice int
  4 );
```

Table created.

```
SQL> CREATE TABLE boat(
  2 BoatModel varchar(10),
  3 BoatPrice int
  4 );
```

Table created.

```
SQL> SELECT * FROM car;
```

CARMODEL	CARPRICE
CarA	20000
CarB	30000
CarC	50000

```
SQL> SELECT * FROM boat;
```

BOATMODEL	BOATPRICE
Boat1	10000
Boat2	40000
Boat3	60000
Boat4	30000

```
SQL> INSERT INTO car values('CarA',20000);
```

1 row created.

```
SQL> INSERT INTO car values('CarB',30000);
```

1 row created.

```
SQL> INSERT INTO car values('CarC',50000);
```

1 row created.

```
SQL> INSERT INTO boat values('Boat1',10000);
1 row created.

SQL> INSERT INTO boat values('Boat2',40000);
1 row created.

SQL> INSERT INTO boat values('Boat3',60000);
1 row created.

SQL> INSERT INTO boat values('Boat4',30000);
1 row created.
```

```
SQL> SELECT * FROM car, boat WHERE CarPrice > BoatPrice;
```

CARMODEL	CARPRICE	BOATMODEL	BOATPRICE
CarA	20000	Boat1	10000
CarB	30000	Boat1	10000
CarC	50000	Boat1	10000
CarC	50000	Boat2	40000
CarC	50000	Boat4	30000

```
SQL> SELECT * FROM car, boat WHERE CarPrice = BoatPrice;
```

CARMODEL	CARPRICE	BOATMODEL	BOATPRICE
CarB	30000	Boat4	30000

Learning Outcome:

EXPERIMENT 8

Problem statement: Write a PL/SQL program using FOR loop to insert ten rows into a database table.

Theory:

SQL Commands:

- 1) Create table classes with only one column (class_id int) to display 10 classes.

SQL> CREATE TABLE classes
2 (3 class_id int4);
Table created.
SQL> desc classes;
Name Null? Type

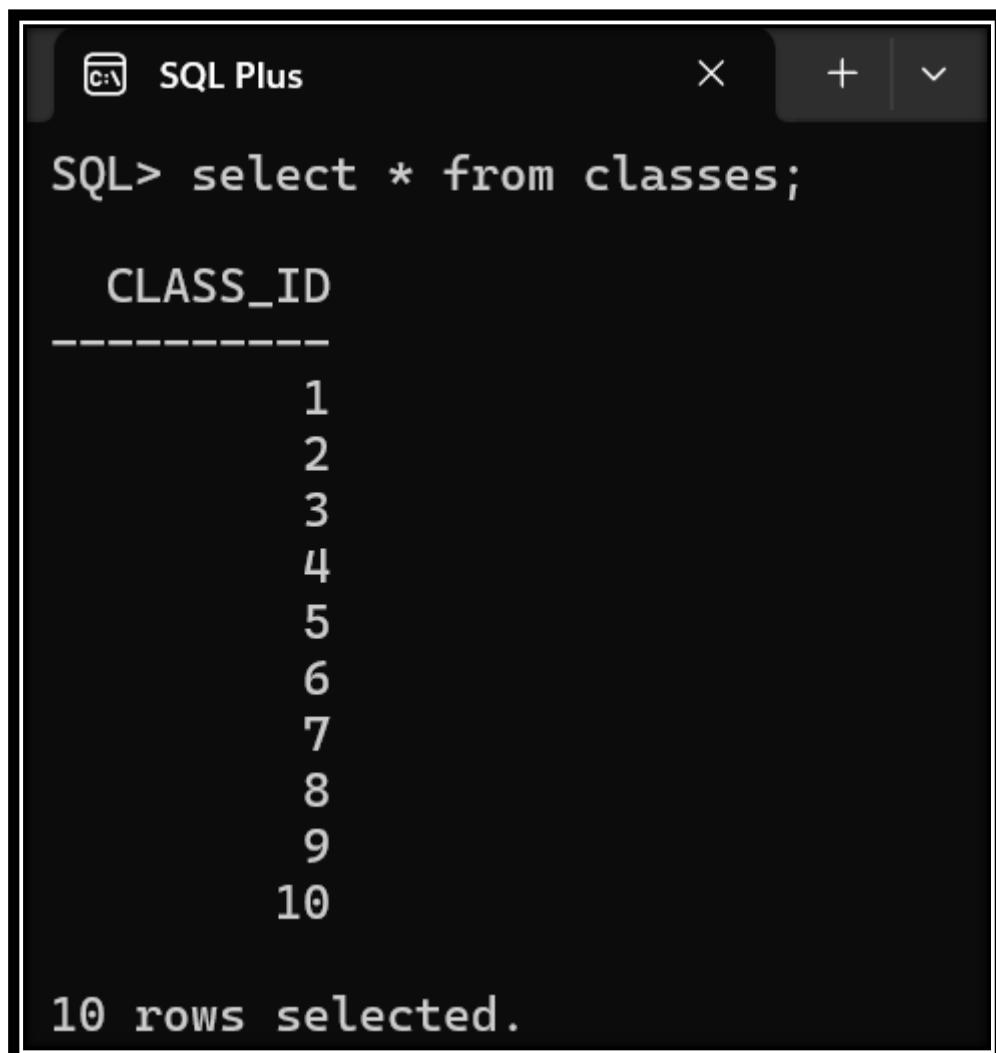
CLASS_ID NUMBER(38)

- 2) Create the PL/SQL command to DECLARE a variable v_counter starting from 1 and in BEGIN make a FOR LOOP to run from 1 – 10, incrementing value of v_counter each time. Add these values to the table each time the for loop iterates.

```
SQL> DECLARE
 2  v_counter NUMBER := 1;
 3  BEGIN
 4  FOR i in 1..10 LOOP
 5  INSERT INTO classes(class_id) VALUES (v_counter);
 6  v_counter := v_counter + 1;
 7  END LOOP;
 8  COMMIT;
 9  dbms_output.put_line('Rows inserted successfully');
10 EXCEPTION
11 WHEN OTHERS THEN
12 dbms_output.put_line('Error:' ||SQLERRM);
13 END;
14 /
```

PL/SQL procedure successfully completed.

3) Print the full table.



The screenshot shows a terminal window titled "SQL Plus". The command "select * from classes;" is entered. The output displays 10 rows of data, each containing a single value for "CLASS_ID": 1, 2, 3, 4, 5, 6, 7, 8, 9, and 10. A message at the bottom indicates "10 rows selected."

CLASS_ID
1
2
3
4
5
6
7
8
9
10

10 rows selected.

Learning Outcome:

EXPERIMENT 9

Problem statement: Given the table EMPLOYEE (Emp No, Name, Salary, Designation, DeptID), write a cursor to select the five highest-paid employees from the table.

Theory:

SQL Commands:

- 1) Create table employee with columns (empno int, name varchar (10), salary int, designation varchar (10), deptid int).

```
SQL> CREATE TABLE employee(
  2  empno int,
  3  name varchar(10),
  4  salary int,
  5  designation varchar(15),
  6  deptid int);
```

Table created.

Name	Null?	Type
EMPNO		NUMBER(38)
NAME		VARCHAR2(10)
SALARY		NUMBER(38)
DESIGNATION		VARCHAR2(15)
DEPTID		NUMBER(38)

- 2) Insert appropriate values into the above table with employee number, names, salary, designation, and department ID.

```
SQL> INSERT INTO employee values(1, 'Ram', 1000, 'Clerk', 101);
1 row created.

SQL> INSERT INTO employee values(2, 'Mohan', 2000, 'AVP', 102);
1 row created.

SQL> INSERT INTO employee values(3, 'Raju', 3000, 'VP', 103);
1 row created.

SQL> INSERT INTO employee values(4, 'Rajesh', 4000, 'EVP', 104);
1 row created.

SQL> INSERT INTO employee values(5, 'Rakesh', 5000, 'TL', 105);
1 row created.

SQL> INSERT INTO employee values(6, 'Raman', 6000, 'Manager', 106);
1 row created.
```

- 3) Create the required cursor to select only the 5 highest paid employees from the table.

```
SQL> SET SERVEROUTPUT ON;
SQL> DECLARE
2  CURSOR c1 is SELECT empno, name, salary, designation, deptid FROM employee ORDER BY(salary) desc FETCH FIRST 5 ROWS ONLY;
3  counter int;
4  e_no int;
5  e_name varchar(10);
6  sal int;
7  des varchar(15);
8  did int;
9  BEGIN
10 OPEN c1;
11 LOOP
12   FETCH c1 into e_no, e_name, sal, des, did;
13   EXIT WHEN c1%notfound;
14   dbms_output.put_line(e_no || ' ' || e_name || ' ' || sal || ' ' || des || ' ' || did);
15 END LOOP;
16 CLOSE c1;
17 END;
18 /
6 Raman 6000 Manager 106
5 Rakesh 5000 TL 105
4 Rajesh 4000 EVP 104
3 Raju 3000 VP 103
2 Mohan 2000 AVP 102
PL/SQL procedure successfully completed.
```

Learning Outcome:

EXPERIMENT 10

Problem statement: Illustrate how you can embed PL/SQL in a high-level host language such as C/Java And demonstrates how a banking debit transaction might be done.

Theory:

SQL Commands: -

- (a) Create a table in SQL Plus and add values to it and display the table.

```
SQL*Plus: Release 11.2.0.1.0 Production on Fri Jul 10 11:08:20 2015
Copyright (c) 1982, 2009, Oracle.  All rights reserved.

SQL> CREATE TABLE employee (
  2  empid int PRIMARY KEY,
  3  empname varchar(10),
  4  salary int);

Table created.

SQL> DESC employee;
      Name          Null?    Type
-----  -----
EMPID           NOT NULL NUMBER(38)
EMPNAME        VARCHAR2(10)
SALARY          NUMBER(38)

SQL> INSERT INTO employee VALUES (1, 'Ram', 10000);

1 row created.

SQL> INSERT INTO employee VALUES (2, 'Shyam', 5000);

1 row created.

SQL> INSERT INTO employee VALUES (3, 'Kunsh', 4000);

1 row created.

SQL> INSERT INTO employee VALUES (4, 'Rakesh', 7000);

1 row created.

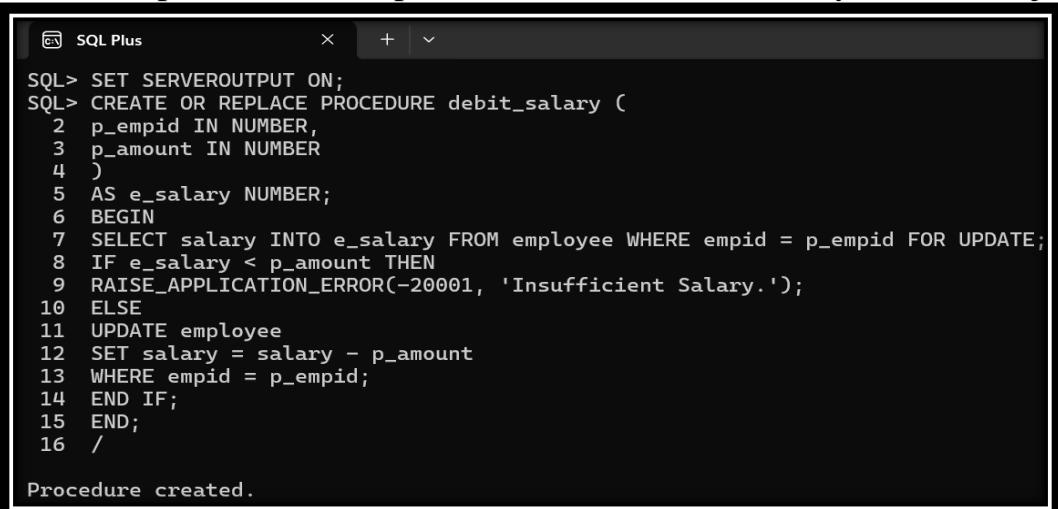
SQL> INSERT INTO employee VALUES (5, 'Rajeev', 3000);

1 row created.

SQL> SELECT * FROM employee;

      EMPID    EMPNAME      SALARY
-----  -----
          1      Ram      10000
          2     Shyam       5000
          3    Kunsh       4000
          4   Rakesh       7000
          5   Rajeev       3000
```

(b) Create a procedure to implement when the connectivity is done via java.



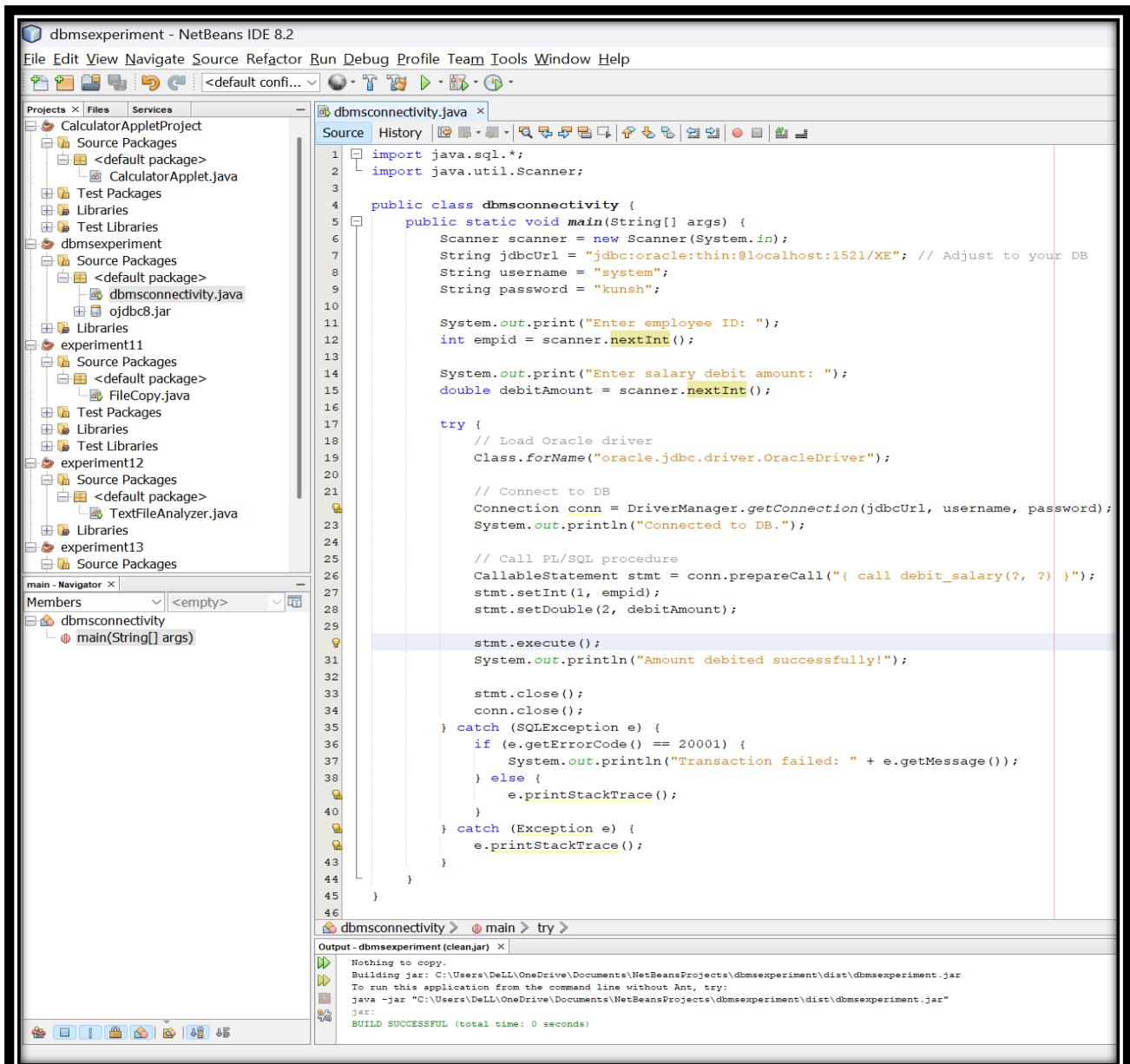
```

SQL> SET SERVEROUTPUT ON;
SQL> CREATE OR REPLACE PROCEDURE debit_salary (
  2   p_empid IN NUMBER,
  3   p_amount IN NUMBER
  4 )
  5   AS e_salary NUMBER;
  6   BEGIN
  7     SELECT salary INTO e_salary FROM employee WHERE empid = p_empid FOR UPDATE;
  8     IF e_salary < p_amount THEN
  9       RAISE_APPLICATION_ERROR(-20001, 'Insufficient Salary.');
 10    ELSE
 11      UPDATE employee
 12        SET salary = salary - p_amount
 13      WHERE empid = p_empid;
 14    END IF;
 15  END;
 16 /

```

Procedure created.

(c) Write the connectivity code in any IDE for Java.



```

import java.sql.*;
import java.util.Scanner;

public class dbmsconnectivity {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        String jdbcUrl = "jdbc:oracle:thin:@localhost:1521/XE"; // Adjust to your DB
        String username = "system";
        String password = "kunsh";

        System.out.print("Enter employee ID: ");
        int empid = scanner.nextInt();

        System.out.print("Enter salary debit amount: ");
        double debitAmount = scanner.nextInt();

        try {
            // Load Oracle driver
            Class.forName("oracle.jdbc.driver.OracleDriver");

            // Connect to DB
            Connection conn = DriverManager.getConnection(jdbcUrl, username, password);
            System.out.println("Connected to DB.");

            // Call PL/SQL procedure
            CallableStatement stmt = conn.prepareCall("{ call debit_salary(?, ?) }");
            stmt.setInt(1, empid);
            stmt.setDouble(2, debitAmount);

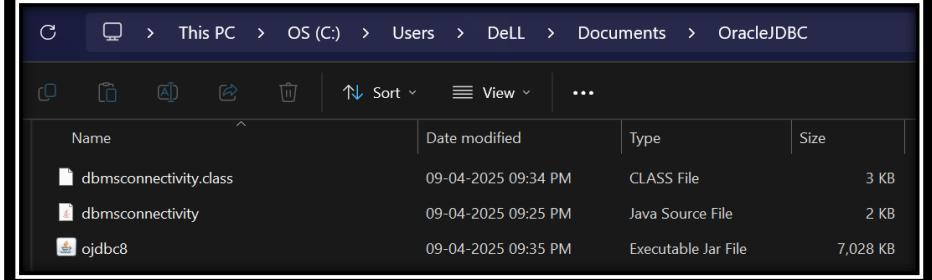
            stmt.execute();
            System.out.println("Amount debited successfully!");

            stmt.close();
            conn.close();
        } catch (SQLException e) {
            if (e.getErrorCode() == 20001) {
                System.out.println("Transaction failed: " + e.getMessage());
            } else {
                e.printStackTrace();
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}

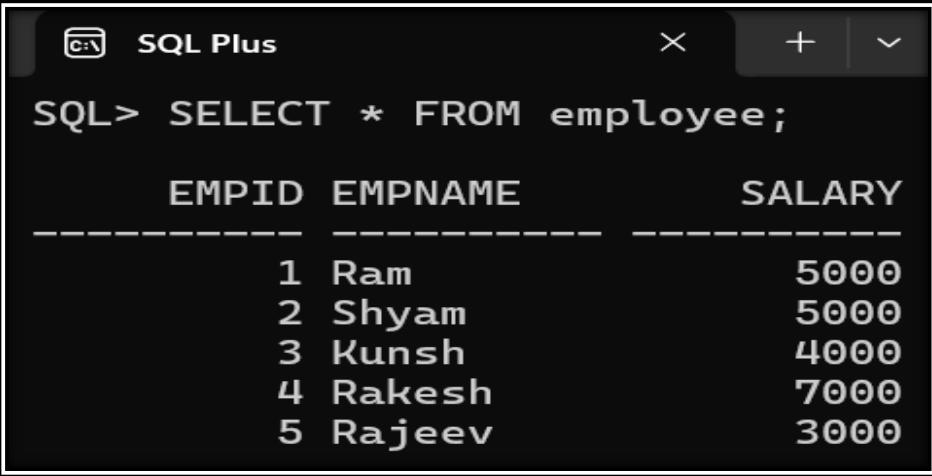
```

The screenshot shows the NetBeans IDE interface with the dbmsconnectivity.java file open in the editor. The code implements a main method that connects to an Oracle database using JDBC and calls a stored procedure named debit_salary with parameters empid and debitAmount. The Java code uses standard JDBC API like Class.forName, DriverManager.getConnection, CallableStatement, and SQLException. The output window at the bottom shows the build log indicating a successful build.

- (d) Open command prompt and navigate to the folder containing the .class file, .java file and ojdbc8.jar file for the connectivity.



Name	Date modified	Type	Size
dbmsconnectivity.class	09-04-2025 09:34 PM	CLASS File	3 KB
dbmsconnectivity	09-04-2025 09:25 PM	Java Source File	2 KB
ojdbc8	09-04-2025 09:35 PM	Executable Jar File	7,028 KB

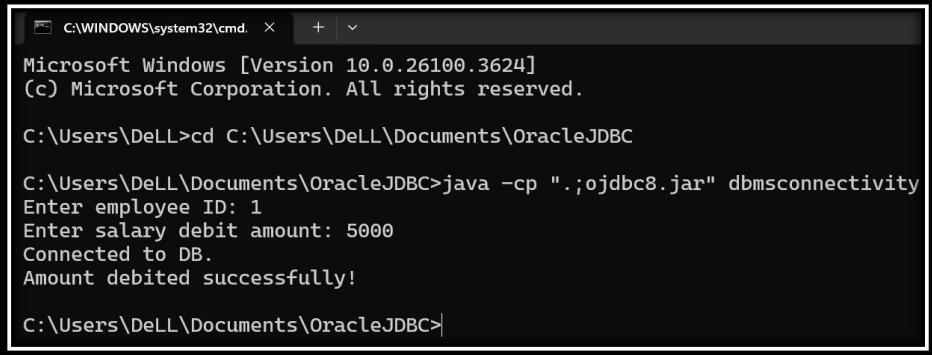


```

SQL> SELECT * FROM employee;

      EMPID    EMPNAME          SALARY
-----  -----
        1    Ram            5000
        2   Shyam            5000
        3   Kunsh            4000
        4  Rakesh            7000
        5  Rajeev            3000
  
```

- (e) Print the table again in SQL to confirm the debit of salary.



```

C:\WINDOWS\system32\cmd. > + ^
Microsoft Windows [Version 10.0.26100.3624]
(c) Microsoft Corporation. All rights reserved.

C:\Users\DeLL>cd C:\Users\DeLL\Documents\OracleJDBC

C:\Users\DeLL\Documents\OracleJDBC>java -cp ".;ojdbc8.jar" dbmsconnectivity
Enter employee ID: 1
Enter salary debit amount: 5000
Connected to DB.
Amount debited successfully!

C:\Users\DeLL\Documents\OracleJDBC>
  
```

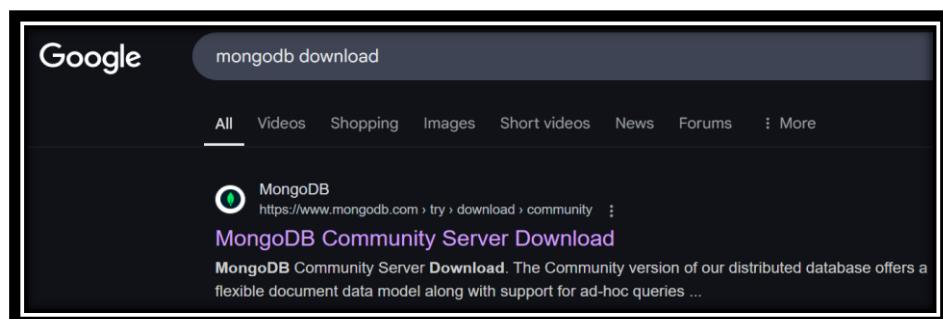
Learning Outcome:

Beyond Curriculum **EXPERIMENT 1**

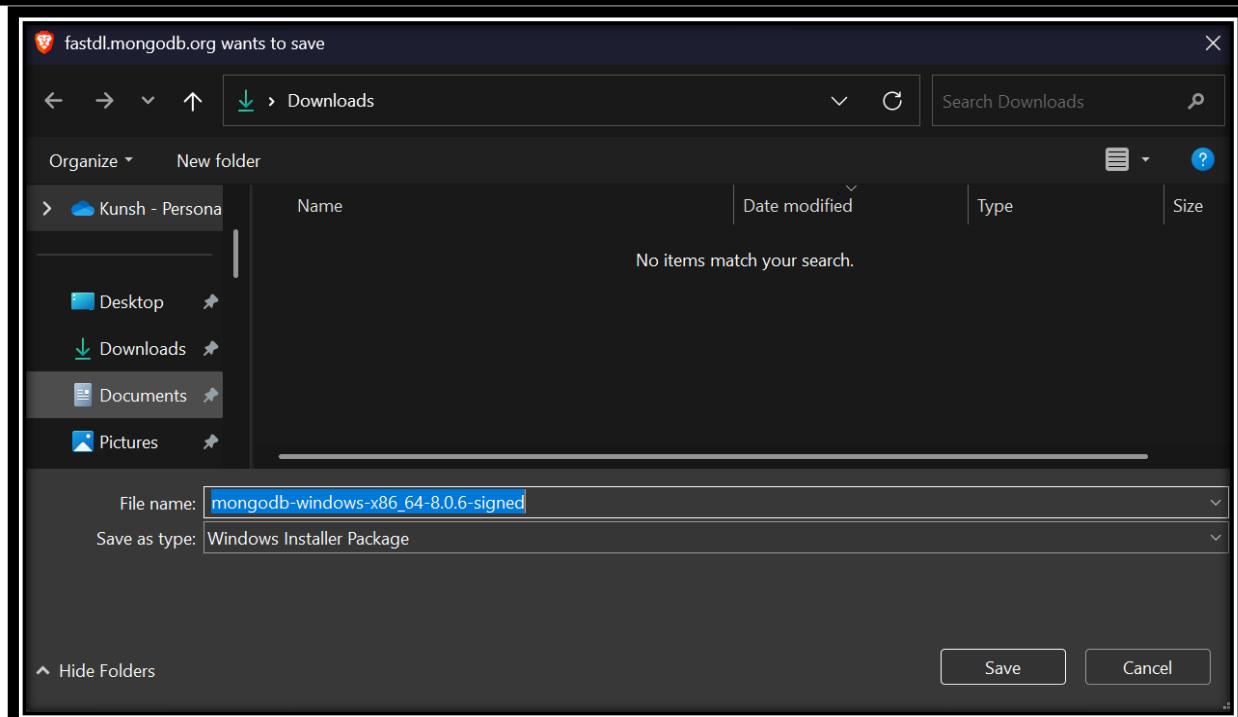
Problem statement: Write the steps to install and implement NOSQL databases-MongoDB.

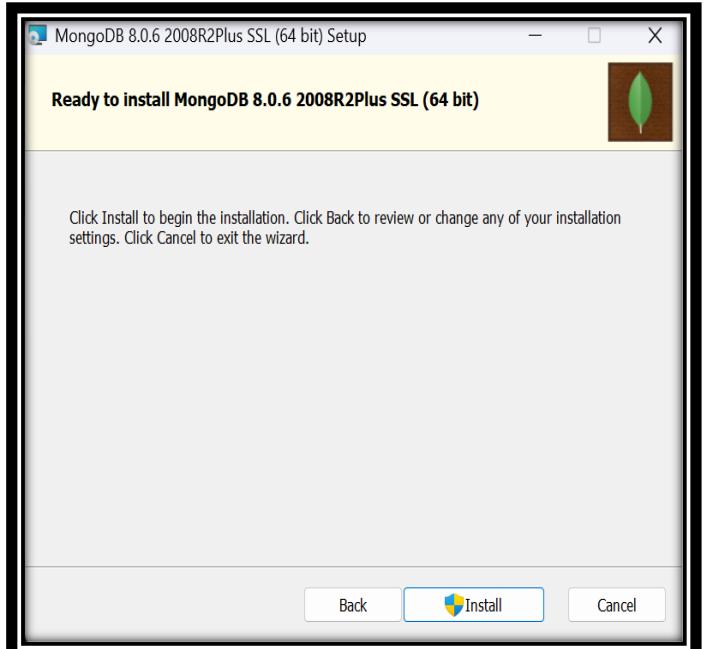
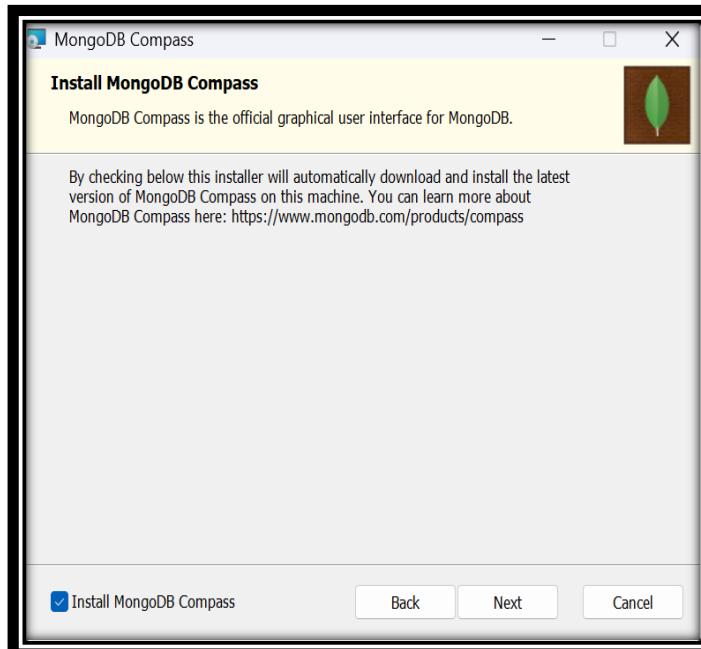
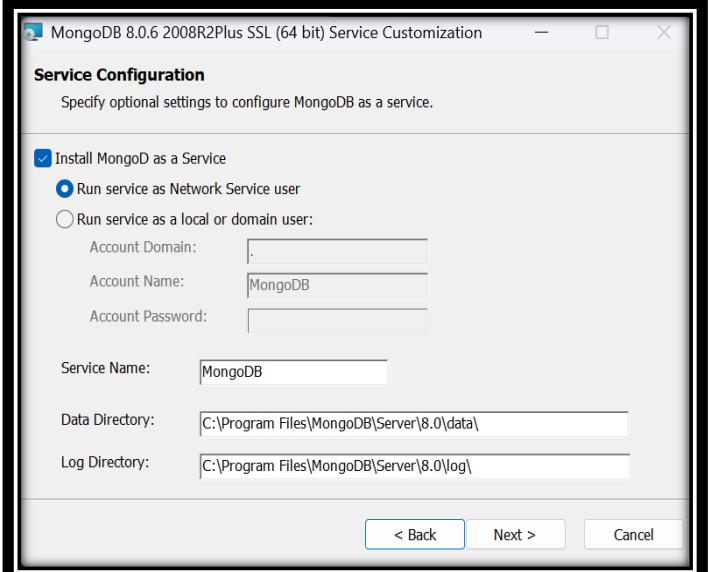
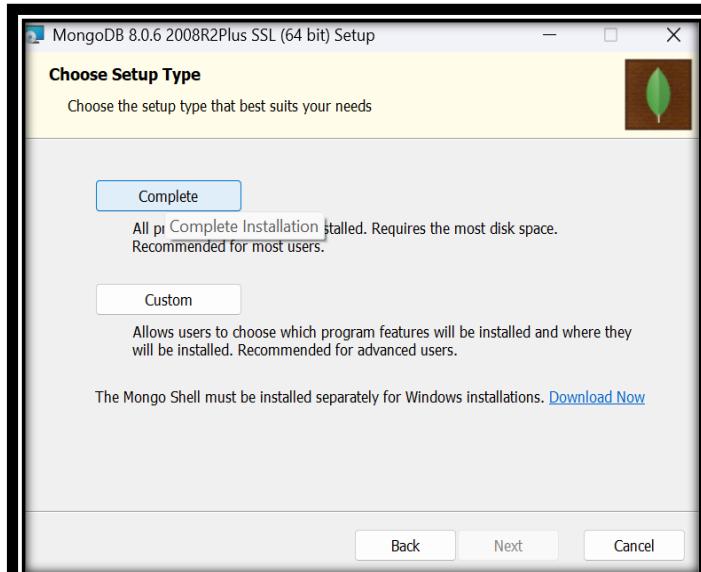
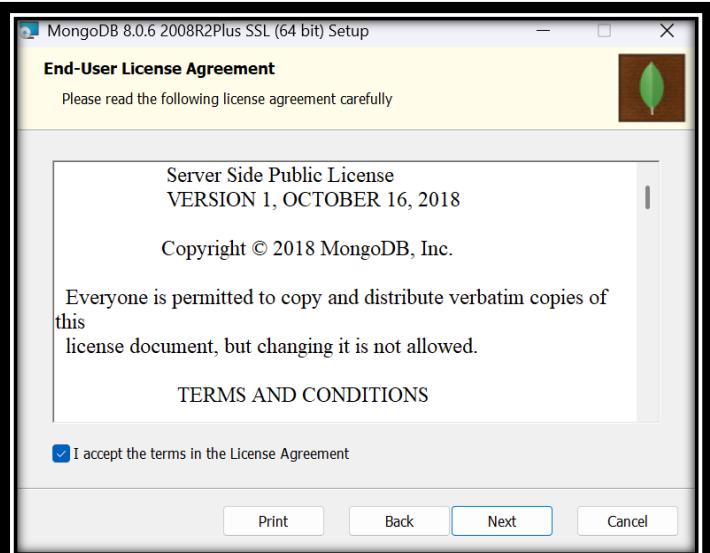
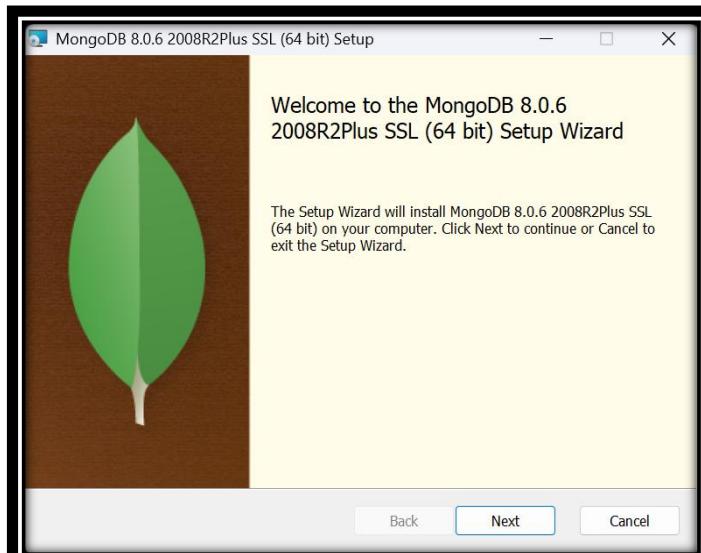
Theory:

Installation Screenshots:

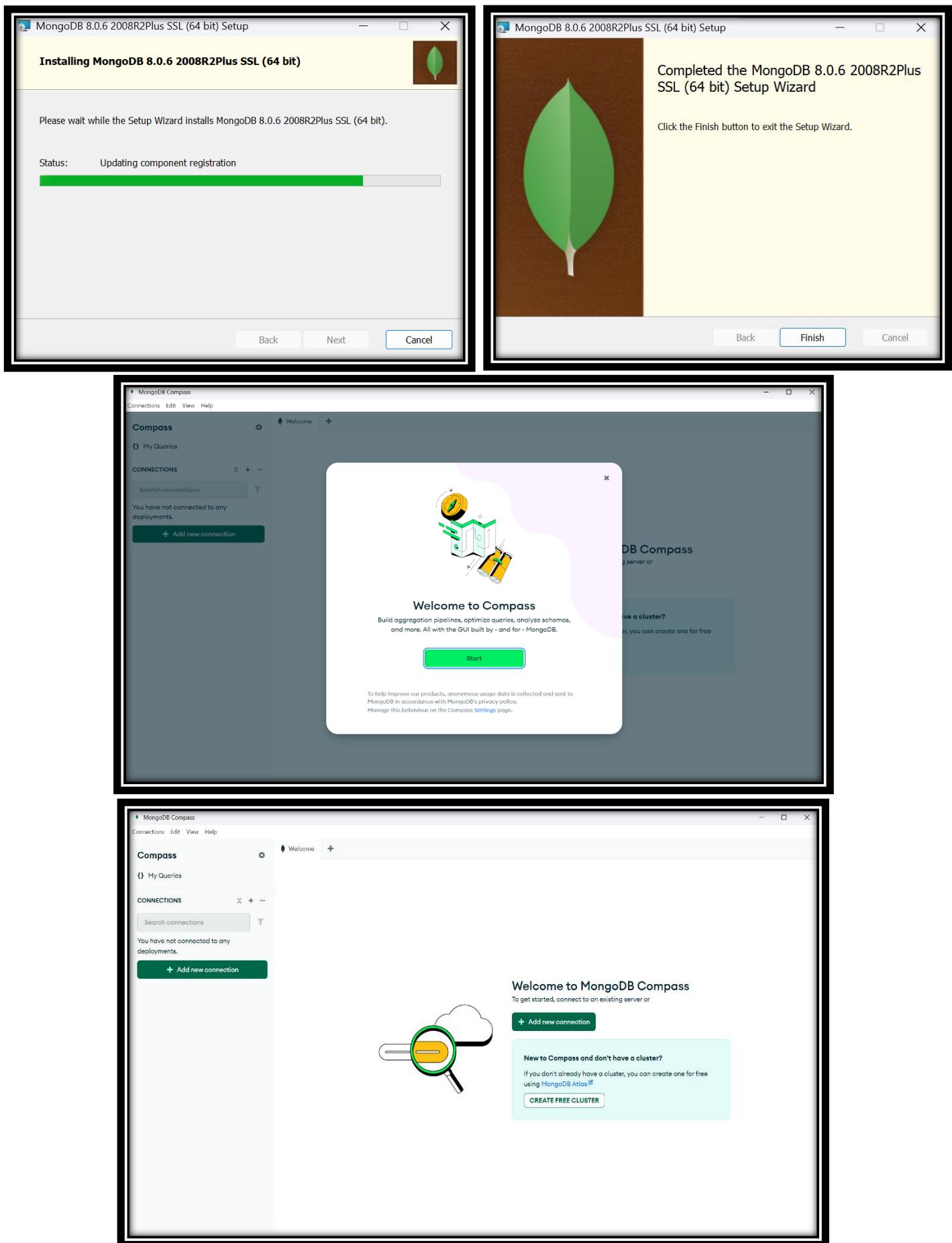


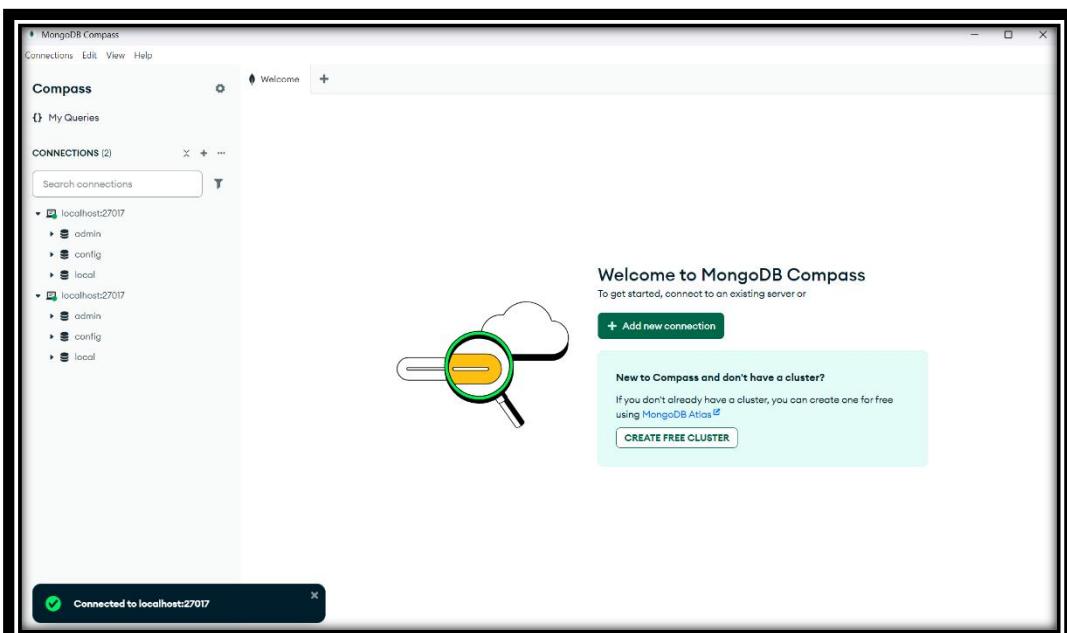
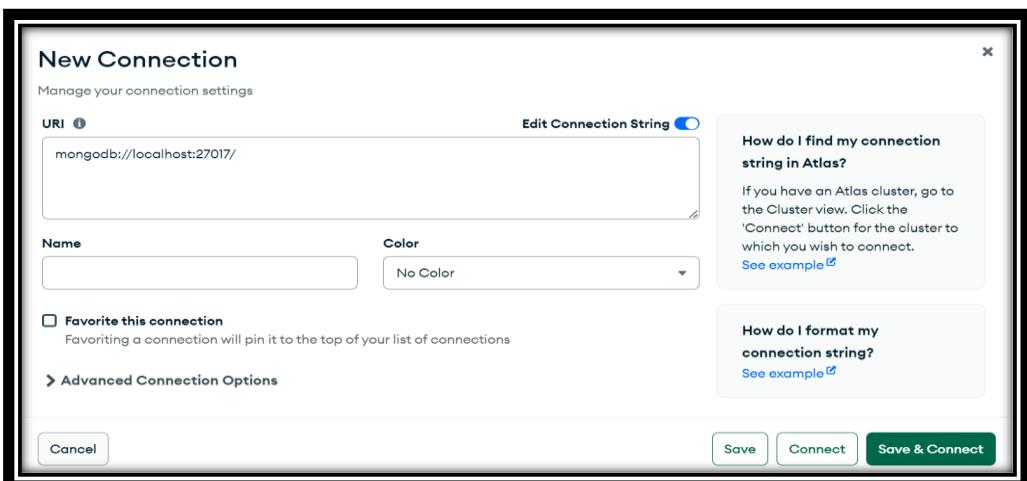
A screenshot of the MongoDB download page for the MongoDB Community Server. The page shows various download options on the left, including MongoDB Atlas, MongoDB Enterprise Advanced, MongoDB Community Edition, and MongoDB Community Server. On the right, there is a section titled "Give it a try with a free, highly-available 512 MB cluster, or get started from your terminal with the following two commands:" which contains the command "\$ brew install mongodb-atlas" and "\$ atlas setup". Below this are dropdown menus for "Version" (set to 8.0.6 (current)), "Platform" (set to Windows x64), and "Package" (set to msi). At the bottom are "Download" and "Copy link" buttons, and a "More Options" button.





KUNSH SABHARWAL





```

mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+2.5.0
MongoDB shell version: 2.5.0
connecting to: mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+2.5.0
For mongosh info see: https://www.mongodb.com/docs/mongodb-shell/
To help improve our products, anonymous usage data is collected and sent to MongoDB periodically (https://www.mongodb.com/legal/privacy-policy).
You can opt-out by running the disableTelemetry() command.

-----
The server generated these startup warnings when booting
2025-04-10T14:47:44.256+05:30: Access control is not enabled for the database. Read and write access to data and configuration is unrestricted
-----

test> show dbs
admin 40.00 KiB
config 60.00 KiB
local 40.00 KiB
test>

```

Learning Outcome:

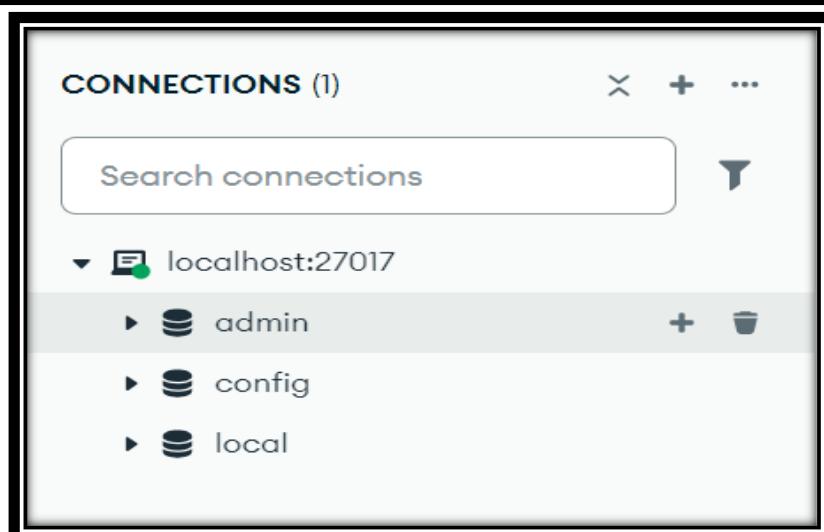
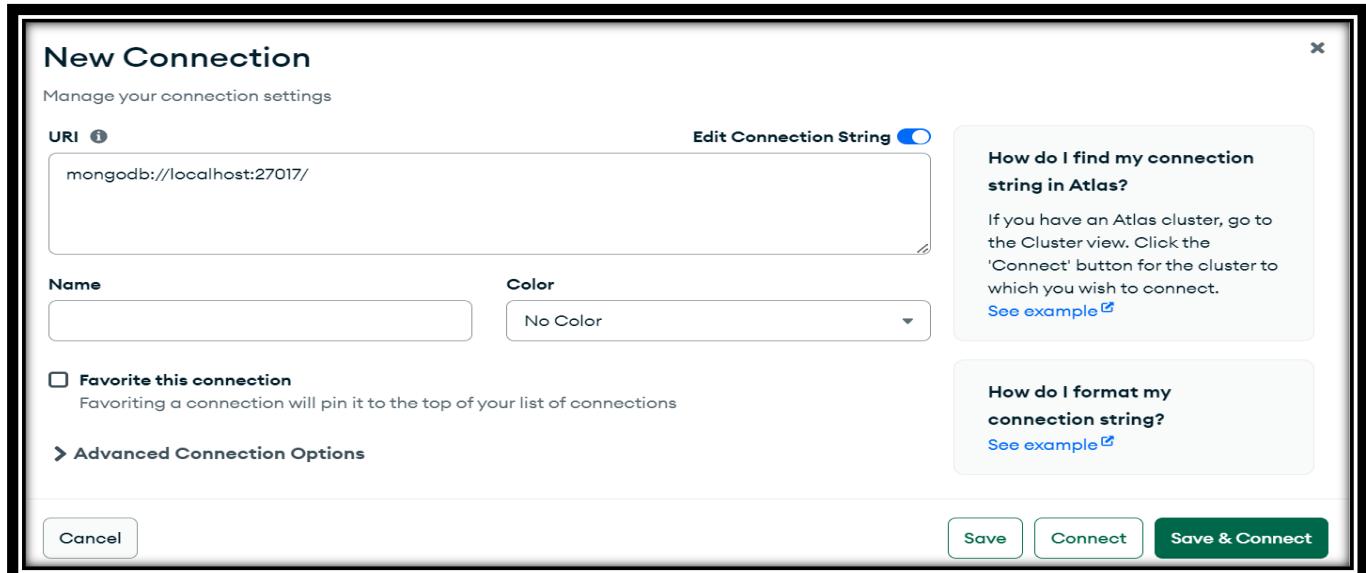
Beyond Curriculum **EXPERIMENT 2**

Problem statement: Study and implement basic commands of MongoDB

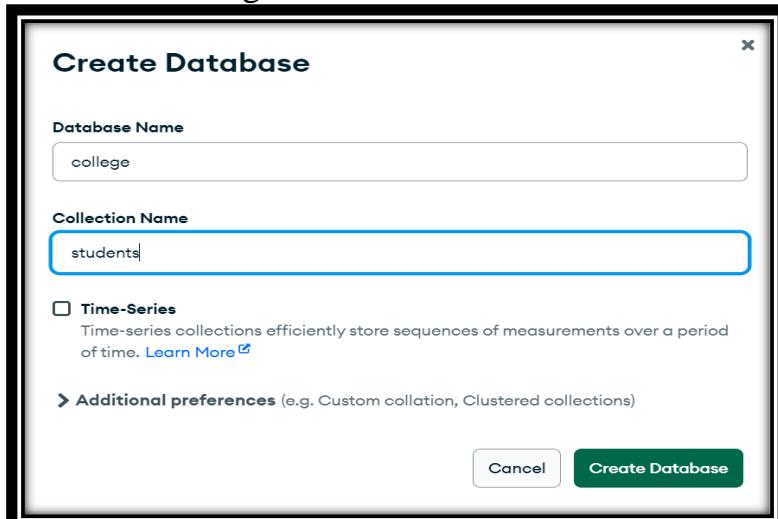
Theory:

MongoDB Compass: -

(a) Connect to the runtime: -localhost:27017.



(b) We see three existing databases named – admin, config and local. Create a new database with name college and in it a collection named students.



(c) View all the databases in compass's GUI by clicking on the connections name i.e. localhost:27017

Database	Storage size	Collections	Indexes
admin	20.48 kB	0	1
college	20.48 kB	1	1
config	12.29 kB	0	2
local	36.86 kB	1	1

(d) On visiting the college database and the collection student we enter the documents tab which is empty by default.

This collection has no data
It only takes a few seconds to import data from a JSON or CSV file.

Import Data

(e) Insert values in the documents tab by clicking on add data → insert document and keep changing the value you want to insert there and click insert after each update to add that value.

```

1 /**
2  * Paste one or more documents here
3 */
4 [
5   {
6     "name": "Kunsh",
7     "roll_no": 101,
8     "course": "BTech",
9     "year": 2
10 }
11 ]

```

```

1 /**
2  * Paste one or more documents here
3 */
4 [
5   {
6     "name": "Riya",
7     "roll_no": 102,
8     "course": "BBA",
9     "year": 2
10 }
11 ]

```

```

1 /**
2  * Paste one or more documents here
3 */
4 [
5   {
6     "name": "Neha",
7     "roll_no": 103,
8     "course": "BCA",
9     "year": 1
10 }
11 ]

```

MongoDB Compass - localhost:27017/college.students

Connections Edit View Collection Help

Compass

My Queries

CONNECTIONS (1)

- localhost:27017
- admin
- college
- students**
- config
- local
- startup_log

localhost:27017 > college > students

Documents 3 Aggregations Schema Indexes 1 Validation

Type a query: { field: 'value' } or [Generate query](#)

[EXPLAIN](#) [RESET](#) [FIND](#) [OPTIONS](#)

[ADD DATA](#) [EXPORT DATA](#) [UPDATE](#) [DELETE](#)

_id	name	roll_no	course	year
<code>_id: ObjectId('67fcda1ccc6bc9c002a347e')</code>	Kunsh	101	BTech	2
<code>_id: ObjectId('67fcda25dccc6bc9c002a3480')</code>	Riya	102	BBA	2
<code>_id: ObjectId('67fcda274ccc6bc9c002a3482')</code>	Neha	103	BCA	1

(f) After inserting the values, we can click on the button in the top right of the MongoDB Compass GUI – [Open MongoDB shell](#). After clicking, a navy-blue colour screen appears where we may type and execute some basic queries related to the data that we just inserted.

Commands in MongoDB shell: -

```
>_MONGOSH
> show dbs;
< admin      40.00 KiB
  college    72.00 KiB
  config     108.00 KiB
  local      72.00 KiB
> use college;
< switched to db college
> show collections;
< students
```

```
>_MONGOSH
> db["students"].find()
< [
  {
    _id: ObjectId('67fcfd241ccc6bc9c002a347e'),
    name: 'Kunsh',
    roll_no: 101,
    course: 'BTech',
    year: 2
  },
  {
    _id: ObjectId('67fcfd25dccc6bc9c002a3480'),
    name: 'Riya',
    roll_no: 102,
    course: 'BBA',
    year: 2
  },
  {
    _id: ObjectId('67fcfd274ccc6bc9c002a3482'),
    name: 'Neha',
    roll_no: 103,
    course: 'BCA',
    year: 1
  }
]
```

```
>_MONGOSH
> db["students"].find().pretty();
< [
  {
    _id: ObjectId('67fcfd241ccc6bc9c002a347e'),
    name: 'Kunsh',
    roll_no: 101,
    course: 'BTech',
    year: 2
  },
  {
    _id: ObjectId('67fcfd25dccc6bc9c002a3480'),
    name: 'Riya',
    roll_no: 102,
    course: 'BBA',
    year: 2
  },
  {
    _id: ObjectId('67fcfd274ccc6bc9c002a3482'),
    name: 'Neha',
    roll_no: 103,
    course: 'BCA',
    year: 1
  }
]
```

```
>_MONGOSH
> db.students.insertOne({
  "name": "Amit",
  "roll_no": 104,
  "course": "MBA",
  "year": 1
});
< {
  acknowledged: true,
  insertedId: ObjectId('67fc3e60422e72b0fdedc')
}
> db.students.updateOne(
  { "name": "Kunsh" },
  { $set: { "course": "MTech" } }
);
< {
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
> db.students.deleteOne({ "name": "Neha" });
< {
  acknowledged: true,
  deletedCount: 1
}
> db.students.countDocuments();
< 3
```

```
>_MONGOSH
> db.students.find({ "year": 2 }).pretty();
< [
  {
    _id: ObjectId('67fc241ccc6bc9c002a347e'),
    name: 'Kunsh',
    roll_no: 101,
    course: 'MTech',
    year: 2
  },
  {
    _id: ObjectId('67fc25dccc6bc9c002a3480'),
    name: 'Riya',
    roll_no: 102,
    course: 'BBA',
    year: 2
  }
]
```

```
>_MONGOSH
> db.students.find().sort({ "roll_no": 1 }).pretty();
< [
  {
    _id: ObjectId('67fc241ccc6bc9c002a347e'),
    name: 'Kunsh',
    roll_no: 101,
    course: 'MTech',
    year: 2
  },
  {
    _id: ObjectId('67fc25dccc6bc9c002a3480'),
    name: 'Riya',
    roll_no: 102,
    course: 'BBA',
    year: 2
  },
  {
    _id: ObjectId('67fc3e60422e72b0fdedc'),
    name: 'Amit',
    roll_no: 104,
    course: 'MBA',
    year: 1
  }
]
```

```
> db.students.find();
< [
  {
    _id: ObjectId('67fcfd241ccc6bc9c002a347e'),
    name: 'Kunsh',
    roll_no: 101,
    course: 'MTech',
    year: 2
  },
  {
    _id: ObjectId('67fcfd25dccc6bc9c002a3480'),
    name: 'Riya',
    roll_no: 102,
    course: 'BBA',
    year: 2
  },
  {
    _id: ObjectId('67fcfd3eec60422e72b0fdedc'),
    name: 'Amit',
    roll_no: 104,
    course: 'MBA',
    year: 1
  }
]
college>
```

Learning Outcome: -