EXPERIMENT 1

1. Create an employee table 'EMP' with following fields :

EMPNO	NUMBER(4)
ENAME	VARCHAR2(25)
JOB	VARCHAR2(12)
SALARY	NUMBER(10,2)
COMMISSION	NUMBER(7,2)
DEPTNO	NUMBER(2)

SQL> create table EMP(empno NUMBER(4), ename VARCHAR2(25), job VARCHAR2(12),salary NUMBER(10,2),commission NUMBER(7,2), deptno NUMBER(2));

Table created.

2. Display the structure of 'EMP'

SQL> desc EMP

Name	Null?	Type
EMPNO		NUMBER(4)
ENAME		VARCHAR2(25)
JOB		VARCHAR2(12)
SALARY		NUMBER(10,2)
COMMISSION		NUMBER(7,2)
DEPTNO		NUMBER(2)

3. Insert the following record into 'EMP'

EMPNO	ENAME	JOB	SAL	COMM	DEPTNO
7369	SMITH	CLERK	2000	800	20

SQL> insert into EMP values(7369,'SMITH','CLERK',2000,800,20);

1 row created.

4. Insert the rest of records using substitution variable.

EMPNO	ENAME	JOB	SAL	COMM	DEPTNO
7499	ALLEN	SALESMAN	1600	300	30
7521	WARD	SALESMAN	1250	500	30
7566	JONES	MANAGER	2975		20
7654	MARTIN	SALESMAN	1250	1400	30
7698	BLAKE	MANAGER	2850		30
7782	CLARK	MANAGER	2450		10
7788	SCOTT	ANALYST	3000		20
7839	KING	PRESIDENT	5000		10
7844	TURNER	SALESMAN	1500		30
7876	ADAMS	CLERK	1100		20
7900	JAMES	NULL	950		30
7902	FORD	ANALYST	3000		20
7934	MILLER	CLERK	1300		10

SQL> insert into EMP values(7499,'ALLEN','SALESMAN',1600,300,30);
1 row created.

SQL> insert into EMP values(7521,'WARD','SALESMAN',1250,500,30);

1 row created.

SQL> insert into EMP(empno,ename,job,salary,deptno)
values(7566,'JONES','MANAGER',2975,20);

1 row created.

SQL> insert into EMP(empno,ename,job,salary,commission,deptno)
values(7654,'MARTIN','SALESMAN',1250,1400,30);

1 row created.

SQL> insert into EMP(empno,ename,job,salary,deptno)
values(7698,'BLAKE','MANAGER',2850,30);

1 row created.

SQL> insert into EMP(empno,ename,job,salary,deptno)
values(7782,'CLARK','MANAGER',2450,10);

1 row created.

SQL> insert into EMP(empno,ename,job,salary,deptno)
values(7788,'SCOTT','ANALYST',3000,20);

1 row created

SQL> insert into EMP(empno,ename,job,salary,deptno)
values(7839,'KING','PRESIDENT',5000,10);
1 row created.

SQL> insert into EMP(empno,ename,job,salary,deptno)
values(7844,'TURNER','SALESMAN',1500,30);

1 row created.

SQL> insert into EMP(empno,ename,job,salary,deptno)
values(7876,'ADAMS','CLERK',1100,20);

1 row created.

SQL> insert into EMP(empno,ename,job,salary,deptno)
values(7900,'JAMES','NULL',950,30);

1 row created.

SQL> insert into EMP(empno,ename,job,salary,deptno)
values(7902,'FORD','ANALYST',3000,20);

1 row created.

SQL> insert into EMP(empno,ename,job,salary,deptno)
values(7934,'MILLER','CLERK',1300,10);
1 row created.

SQL> SET LINESIZE 100;
SQL> SELECT * FROM EMP;

EMPNO	ENAME	JOB	SALARY	COMMISSION	DEPTNO
7369	SMITH	CLERK	2000	800	20
7499	ALLEN	SALESMAN	1600	300	30
7521	WARD	SALESMAN	1250	500	30
7566	JONES	MANAGER	2975		20
7654	MARTIN	SALESMAN	1250	1400	30
7698	BLAKE	MANAGER	2850		30
7782	CLARK	MANAGER	2450		10
7788	SCOTT	ANALYST	3000		20
7839	KING	PRESIDENT	5000		10
7844	TURNER	SALESMAN	1500		30
7876	ADAMS	CLERK	1100		20
7900	JAMES	NULL	950		30
7902	FORD	ANALYST	3000		20
7934	MILLER	CLERK	1300		10

14 rows selected.

5. Insert job as 'CLERK' for all 'NULL' job types.

SQL> UPDATE EMP SET JOB='CLERK' WHERE JOB='NULL';

1 row updated.

SQL> select * from EMP;

EMPNO	ENAME	JOB	SALARY	COMMISSION	DEPTNO
7369	SMITH	CLERK	2000	800	20
7499	ALLEN	SALESMAN	1600	300	30
7521	WARD	SALESMAN	1250	500	30
7566	JONES	MANAGER	2975		20
7654	MARTIN	SALESMAN	1250	1400	30
7698	BLAKE	MANAGER	2850		30
7782	CLARK	MANAGER	2450		10
7788	SCOTT	ANALYST	3000		20
7839	KING	PRESIDENT	5000		10
7844	TURNER	SALESMAN	1500		30
7876	ADAMS	CLERK	1100		20
7900	JAMES	CLERK	950		30
7902	FORD	ANALYST	3000		20
7934	MILLER	CLERK	1300		10

14 rows selected.

6. Add a new field 'date_join' with following values

date_join
17-DEC-80
20-FEB-81
22-FEB-81
02-APR-81
28-SEP-81
01-MAY-81
09-JUN-81
19-APR-87
17-NOV-81
08-SEP-81

SQL> alter table EMP ADD date_join date;

Table altered.

SQL> update EMP set date_join='17-DEC-80' where empno=7369;
1 row updated.

SQL> update EMP set date_join='20-FEB-81' where empno=7499;
1 row updated.

SQL> update EMP set date_join='22-FEB-81' where empno=7521;

1 row updated.

SQL> update EMP set date_join='02-APR-81' where empno=7566;

1 row updated.

SQL> update EMP set date_join='28-SEP-81' where empno=7654;

1 row updated.

SQL> update EMP set date_join='01-MAY-81' where empno=7698;

1 row updated.

SQL> update EMP set date_join='09-JUN-81' where empno=7782;

1 row updated.

SQL> update EMP set date_join='19-APR-87' where empno=7788;

1 row updated.

SQL> update EMP set date_join='17-NOV-81' where empno=7839;

1 row updated.

SQL> update EMP set date_join='08-SEP-81' where empno=7844;

1 row updated.

SQL> update EMP set date_join='23-MAY-87' where empno=7876;

1 row updated.

SQL> update EMP set date_join='03-DEC-81' where empno=7900;

1 row updated.

SQL> update EMP set date_join='03-DEC-81' where empno=7902;

1 row updated.

SQL> update EMP set date_join='23-JAN-82' where empno=7934;

1 row updated.

7. Display details of all employees

SQL> select * from EMP;

EMPNO	ENAME	JOB	SALARY	COMMISSION	DEPTNO	DATE_JOIN
7369	SMITH	CLERK	2000	800	20	17-DEC-80
7499	ALLEN	SALESMAN	1600	300	30	20-FEB-81
7521	WARD	SALESMAN	1250	500	30	22-FEB-81
7566	JONES	MANAGER	2975		20	02-APR-81
7654	MARTIN	SALESMAN	1250	1400	30	28-SEP-81
7698	BLAKE	MANAGER	2850		30	01-MAY-81
7782	CLARK	MANAGER	2450		10	09-JUN-81
7788	SCOTT	ANALYST	3000		20	19-APR-87
7839	KING	PRESIDENT	5000		10	17-NOV-81
7844	TURNER	SALESMAN	1500		30	08-SEP-81
7876	ADAMS	CLERK	1100		20	23-MAY-87
7900	JAMES	CLERK	950		30	03-DEC-81
7902	FORD	ANALYST	3000		20	03-DEC-81
7934	MILLER	CLERK	1300		10	23-JAN-82

14 rows selected.

8. Display all the distinct job types in 'EMP'.

SQL> select distinct job from EMP;

```
JOB
------------
CLERK
SALESMAN
MANAGER
ANALYST
PRESIDENT
```

9. Display names of all employees in dept 20 and 30

SQL> select ename from EMP where deptno in (20,30);

```
ENAME
------------------------
SMITH
ALLEN
WARD
JONES
MARTIN
BLAKE
SCOTT
```

TURNER
ADAMS
JAMES
FORD

11 rows selected.

10. List name and Total of salary i.e sal+commission

SQL> select ename,sum(salary + commission) from EMP GROUP by ename;

ENAME	SUM(SALARY+COMMISSION)
SMITH	2800
ALLEN	1900
WARD	1750
JONES	2975
MARTIN	2650
BLAKE	2850
CLARK	2450
SCOTT	3000
KING	5000
TURNER	1500
ADAMS	1100
JAMES	950
FORD	3000
MILLER	1300

14 rows selected.

11. List name and Annual Salary i.e sal*12

SQL> select ename,sum(salary*12) from EMP group by ename;

ENAME	SUM(SALARY*12)
SMITH	24000
ALLEN	19200
WARD	15000
JONES	35700
MARTIN	15000
BLAKE	34200
CLARK	29400
SCOTT	36000
KING	60000
TURNER	18000
ADAMS	13200
JAMES	11400

FORD   36000
MILLER   15600

14 rows selected.

12. List the employee who joined in the date '03-DEC-81'

SQL> select ename from EMP where date_join='03-DEC-81';

ENAME

JAMES
FORD

13. Display the total salary of 'Miller'.

SQL> select salary from EMP where ename='MILLER';

  SALARY

    1300

14. Delete the employee 'Miller' from'EMP'

SQL> DELETE FROM EMP WHERE ENAME='MILLER';

1 row deleted.

15. Display name and deptno of all employees.

SQL> SELECT ENAME,DEPTNO FROM EMP;

ENAME   DEPTNO
------------------------ ----------
SMITH   20
ALLEN   30
WARD   30
JONES   20
MARTIN   30
BLAKE   30
CLARK   10
SCOTT   20
KING   10
TURNER   30

ADAMS	20
JAMES	30
FORD	20

13 rows selected.

16. Remove the field 'commission' fom'EMP' after updating salary with total salary, i.e sal+commission

SQL> UPDATE EMP SET SALARY=SALARY+COMMISSION;

13 rows updated.

SQL> ALTER TABLE EMP DROP(COMMISSION);

Table altered.

SQL> SELECT * FROM EMP;

EMPNO	ENAME	JOB	SALARY	DEPTNO	DATE_JOIN
7369	SMITH	CLERK	2800	20	17-DEC-80
7499	ALLEN	SALESMAN	1900	30	20-FEB-81
7521	WARD	SALESMAN	1750	30	22-FEB-81
7566	JONES	MANAGER	2975	20	02-APR-81
7654	MARTIN	SALESMAN	2650	30	28-SEP-81
7698	BLAKE	MANAGER	2850	30	01-MAY-81
7782	CLARK	MANAGER	2450	10	09-JUN-81
7788	SCOTT	ANALYST	3000	20	19-APR-87
7839	KING	PRESIDENT	5000	10	17-NOV-81
7844	TURNER	SALESMAN	1500	30	08-SEP-81
7876	ADAMS	CLERK	1100	20	23-MAY-87
7900	JAMES	CLERK	950	30	03-DEC-81
7902	FORD	ANALYST	3000	20	03-DEC-81

13 rows selected.

17. Display the name of employees having the same amount of salary (don't use subqueries)

SQL> SELECT ENAME,SALARY FROM EMP WHERE SALARY IN (SELECT salary FROM EMP e WHERE EMP.EMPNO < > e.EMPNO);

ENAME	SALARY
FORD	3000
SCOTT	3000

18. Display the name and employee no as 'name' and 'emp_id'

SQL> SELECT ENAME,EMPNO FROM EMP;

```
ENAME                    EMPNO
------------------------ ----------
SMITH                     7369
ALLEN                     7499
WARD                      7521
JONES                     7566
MARTIN                    7654
BLAKE                     7698
CLARK                     7782
SCOTT                     7788
KING                      7839
TURNER                    7844
ADAMS                     7876
JAMES                     7900
FORD                      7902
```

13 rows selected.

19. Rename table 'EMP' to 'EMPLOYEE'

SQL> ALTER TABLE EMP RENAME TO EMPLOYEE;

Table altered.

20. Create a new table 'EMP_TAB' from table 'EMPLOYEE'

SQL> CREATE TABLE EMP_TAB AS (SELECT * FROM EMPLOYEE);

Table created.

21. List the details of 'EMPLOYEE' and 'EMPTAB'

SQL> SELECT * FROM EMP;

EMPNO	ENAME	JOB	SALARY	DEPTNO	DATE_JOIN
7369	SMITH	CLERK	2800	20	17-DEC-80
7499	ALLEN	SALESMAN	1900	30	20-FEB-81
7521	WARD	SALESMAN	1750	30	22-FEB-81
7566	JONES	MANAGER	2975	20	02-APR-81
7654	MARTIN	SALESMAN	2650	30	28-SEP-81
7698	BLAKE	MANAGER	2850	30	01-MAY-81

7782	CLARK	MANAGER	2450	10	09-JUN-81
7788	SCOTT	ANALYST	3000	20	19-APR-87
7839	KING	PRESIDENT	5000	10	17-NOV-81
7844	TURNER	SALESMAN	1500	30	08-SEP-81
7876	ADAMS	CLERK	1100	20	23-MAY-87
7900	JAMES	CLERK	950	30	03-DEC-81
7902	FORD	ANALYST	3000	20	03-DEC-81

13 rows selected.

SQL> SELECT * FROM EMP_TAB;

EMPNO	ENAME	JOB	SALARY	DEPTNO	DATE_JOIN
7369	SMITH	CLERK	2800	20	17-DEC-80
7499	ALLEN	SALESMAN	1900	30	20-FEB-81
7521	WARD	SALESMAN	1750	30	22-FEB-81
7566	JONES	MANAGER	2975	20	02-APR-81
7654	MARTIN	SALESMAN	2650	30	28-SEP-81
7698	BLAKE	MANAGER	2850	30	01-MAY-81
7782	CLARK	MANAGER	2450	10	09-JUN-81
7788	SCOTT	ANALYST	3000	20	19-APR-87
7839	KING	PRESIDENT	5000	10	17-NOV-81
7844	TURNER	SALESMAN	1500	30	08-SEP-81
7876	ADAMS	CLERK	1100	20	23-MAY-87
7900	JAMES	CLERK	950	30	03-DEC-81
7902	FORD	ANALYST	3000	20	03-DEC-81

13 rows selected.

22. Delete all records from 'EMP'

SQL> DELETE FROM EMP_TAB;

13 rows deleted.

23. Delete the table 'EMP'

SQL> DROP TABLE EMP_TAB;

Table dropped.

EXPERIMENT 2

Create the following tables and execute the queries given below

SAILORS

sid	sname	rating	age
22	Dustin	7	45
29	Brutas	1	33
31	Lubber	8	55
32	Andy	8	25
58	Rusty	10	35
64	Horatio	7	35
71	Zorba	10	16
74	Horatio	9	35
85	Art	3	26
95	Bob	3	64

BOATS

Bid	bname	color
101	Interlake	Blue
102	Interlake	Red
103	Clipper	Green
104	Marine	Red

RESERVES

sid	bid	day
22	101	10/10/98
22	102	10/10/98
22	103	10/8/98
22	104	10/7/98
31	102	11/10/98

31	103	11/6/98
31	104	11/12/98
64	101	9/5/98
64	102	9/8/98
74	103	9/8/98

SQL> CREATE TABLE SAILORS(sid NUMBER(10) PRIMARY KEY, sname VARCHAR2(25),rating NUMBER(10,2),age NUMBER(2));

Table created.

SQL> INSERT INTO SAILORS VALUES(22,'Dustin',7,45);
1 row created.

SQL> INSERT INTO SAILORS VALUES(29,'Brutas',1,33);
1 row created.

SQL> INSERT INTO SAILORS VALUES(31,'Lubber',8,55);
1 row created.

SQL> INSERT INTO SAILORS VALUES(32,'Andy',8,25);
1 row created.

SQL> INSERT INTO SAILORS VALUES(58,'Rusty',10,35);
1 row created.

SQL> INSERT INTO SAILORS VALUES(64,'Horatio',7,35);
1 row created.

SQL> INSERT INTO SAILORS VALUES(71,'Zorba',10,16);
1 row created.

SQL> INSERT INTO SAILORS VALUES(74,'Horatio',9,35);
1 row created.

SQL> INSERT INTO SAILORS VALUES(85,'Art',3,26);
1 row created.

SQL> INSERT INTO SAILORS VALUES(95,'Bob',3,64);
1 row created.

SQL> SELECT * FROM SAILORS;

```
    SID   SNAME              RATING      AGE
 ---------- ---------------------- ---------- ----------
    22   Dustin                7          45
```

29	Brutas	1	33
31	Lubber	8	55
32	Andy	8	25
58	Rusty	10	35
64	Horatio	7	35
71	Zorba	10	16
74	Horatio	9	35
85	Art	3	26
95	Bob	3	64

10 rows selected.

SQL> CREATE TABLE BOATS(bid NUMBER(10) PRIMARY KEY, bname VARCHAR2(25),color VARCHAR2(25));

Table created.

SQL> INSERT INTO BOATS VALUES(101,'Interlake','Blue');
1 row created.

SQL> INSERT INTO BOATS VALUES(102,'Interlake','Red');
1 row created.

SQL> INSERT INTO BOATS VALUES(103,'Clipper','Green');
1 row created.

SQL> INSERT INTO BOATS VALUES(104,'Marine','Red');
1 row created.

SQL> SELECT * FROM BOATS;

BID	BNAME	COLOR
101	Interlake	Blue
102	Interlake	Red
103	Clipper	Green
104	Marine	Red

SQL> CREATE TABLE RESERVES(sid INT REFERENCES SAILORS(sid), bid INT REFERENCES BOATS(bid),day DATE);

Table created.

SQL> INSERT INTO RESERVES VALUES(22,101,'10/oct/98');
1 row created.

SQL> INSERT INTO RESERVES VALUES(22,102,'10/oct/98');
1 row created.

SQL> INSERT INTO RESERVES VALUES(22,103,'10/aug/98');
1 row created.

SQL> INSERT INTO RESERVES VALUES(22,104,'10/jul/98');
1 row created.

SQL> INSERT INTO RESERVES VALUES(31,102,'11/oct/98');
1 row created.

SQL> INSERT INTO RESERVES VALUES(31,103,'11/jun/98');
1 row created.

SQL> INSERT INTO RESERVES VALUES(31,104,'11/dec/98');
1 row created.

SQL> INSERT INTO RESERVES VALUES(64,101,'09/may/98');
1 row created.

SQL> INSERT INTO RESERVES VALUES(64,102,'09/aug/98');
1 row created.

SQL> INSERT INTO RESERVES VALUES(74,103,'09/aug/98');
1 row created.

```
SQL> SELECT * FROM RESERVES
    SID     BID       DAY
 ---------- ----------  ---------
     22     101      10-OCT-98
     22     102      10-OCT-98
     22     103      10-AUG-98
     22     104      10-JUL-98
     31     102      11-OCT-98
     31     103      11-JUN-98
     31     104      11-DEC-98
     64     101      09-MAY-98
     64     102      09-AUG-98
     74     103      09-AUG-98
```

10 rows selected.

1. Find the names and ages of all sailors

SQL> SELECT sname,age FROM SAILORS;

```
SNAME                 AGE
------------------------  ----------
Dustin                 45
Brutas                 33
```

Lubber	55
Andy	25
Rusty	35
Horatio	35
Zorba	16
Horatio	35
Art	26
Bob	64

10 rows selected.

2. Find all information of sailors who have reserved boat number 101.

SQL> SET LINESIZE 100;
SQL> SELECT * FROM SAILORS S,RESERVES R WHERE S.sid=R.sid AND R.bid=101;

SID	SNAME	RATING	AGE	SID	BID	DAY
22	Dustin	7	45	22	101	10-OCT-98
64	Horatio	7	35	64	101	09-MAY-98

3. Find all sailors with rating above 7.

SQL> SELECT * FROM SAILORS WHERE rating>7;

SID	SNAME	RATING	AGE
31	Lubber	8	55
32	Andy	8	25
58	Rusty	10	35
71	Zorba	10	16
74	Horatio	9	35

4. Find the names of sailors who have reserved boat no 103.

SQL> SELECT S.sname FROM SAILORS S,RESERVES R WHERE S.sid=R.sid AND R.bid=103;

SNAME

Dustin
Lubber
Horatio

5. Find the names of sailors who have reserved a red boat, and list in the order of age.

SQL> select distinct s.sname,s.age from SAILORS s,RESERVES r,BOATS b where s.sid=r.sid and r.Bid=b.Bid and b.color='Red'order by s.age;

```
SNAME                   AGE
----------------------- ----------
Horatio                 35
Dustin                   45
Lubber                   55
```

6. Find the names of sailors who have reserved either a red or green boat.

SQL> select distinct s.sname from sailors s,reserves r,boats b where s.sid=r.sid and r.bid=b.bid and (b.color='Red' or b.color='Green');

```
SNAME
------------------------
Dustin
Lubber
Horatio
```

7. Find the colors of boats reserved by "Lubber".

SQL> select distinct b.color from sailors s,reserves r,boats b where s.sid=r.sid and r.bid=b.bid and s.sname='Lubber';

```
COLOR
------------------------
Red
Green
```

8. Find the names of sailors who have reserved both red and green boats.

SQL> select s.sname from SAILORS s,BOATS b,RESERVES r where s.sid=r.sid and r.Bid=b.Bid and b.color='Red' intersect select s.sname from SAILORS s,BOATS b,RESERVES r WHERE s.sid=r.sid and r.Bid=b.Bid and b.color='Green';

```
SNAME
------------------------
Dustin
Lubber
Horatio
```

9. Find the names of sailors who have reserved at least one boat

SQL> SELECT DISTINCT s.sname FROM SAILORS s, RESERVES r WHERE s.sid = r.sid;

SNAME

Dustin
Lubber
Horatio

10. Find the ids and names of sailors who have reserved two different boats on the same day.

SQL> SELECT DISTINCT s.sid,s.sname FROM SAILORS s,RESERVES r1,RESERVES r2 WHERE s.sid=r1.sid AND s.sid=r2.sid AND r1.day=r2.day AND r1.Bid<>r2.Bid;

     SID   SNAME
   ---------- ------------------------
     22   Dustin

11. Find the name and the age of the youngest sailor.

SQL> select s.sname,s.age from sailors s where s.age<=all(select age from sailors);

SNAME   AGE
------------------------ ----------
Zorba   16

12. Find the names and ratings of a sailor whose rating is better than some sailor called Horatio

SQL> select s.sname,s.rating from sailors s where s.rating>any(select s2.rating from sailors s2 where s2.sname='Horatio');

SNAME   RATING
------------------------ ----------
Rusty   10
Zorba   10
Horatio   9
Lubber   8
Andy   8

13. Find the names of sailors who have reserved all boats

SQL> select s.sname from sailors s where NOT EXISTS (select b.bid from boats b where NOT EXISTS (select r.bid from reserves r where r.bid = b.bid and r.sid = s.sid));

SNAME

Dustin

14. Count the number of different sailor names.

SQL> select count(distinct s.sname)from sailors s;

COUNT(DISTINCTS.SNAME)

        9

15. Calculate the average age of all sailors.

SQL> SELECT AVG(s.age) FROM SAILORS S;

AVG(S.AGE)

 36.9

16. Find the average age of sailors for each rating level.

SQL> select s.rating,avg(s.age)as avg_age from SAILORS s group by s.rating;

    RATING   AVG_AGE
---------- ----------
        7   40
        1   33
        8   40
       10   25.5
        9   35
        3   45

6 rows selected.

17. Find the average age of sailors for each rating level that has at least two sailors.

SQL> select s.rating,avg(s.age)as avg_age from SAILORS s group by s.rating having count(*)>1;

    RATING   AVG_AGE
  ---------- ----------
        7   40
        8   40
       10   25.5
        3   4

EXPERIMENT 3

Consider the following schema for OrderDatabase:

SALESMAN (Salesman_id, Name, City, Commission)

CUSTOMER (Customer_id, Cust_Name, City, Grade,Salesman_id)

ORDERS (Ord_No, Purchase_Amt, Ord_Date, Customer_id,Salesman_id)

Write SQL queries to

SALESMAN_ID	NAME	CITY	COMMISSION
1000	JOHN	BANGALORE	25
2000	RAVI	BANGALORE	20
3000	KUMAR	MYSORE	15
4000	SMITH	DELHI	30
5000	HARSHA	HYDRABAD	15

CUSTOMER_ID	CUST_NAME	CITY	GRADE	SALESMAN_ID
10	PREETHI	BANGALORE	100	1000
11	VIVEK	MANGALORE	300	1000
12	BHASKAR	CHENNAI	400	2000
13	CHETHAN	BANGALORE	200	2000
14	MAMATHA	BANGALORE	400	3000

ORD_NO	PURCHASE_AMT	ORD_DATE	CUSTOMER_ID	SALESMAN_ID
50	5000	04-MAY-17	10	1000
51	450	20-JAN-17	10	2000
52	1000	24-FEB-17	13	2000
53	3500	13-APR-17	14	3000
54	550	09-MAR-17	12	2000

SQL> CREATE TABLE SALESMAN (Salesman_id NUMBER(10) PRIMARY KEY, Name VARCHAR (20), City VARCHAR (20), Commission number);

Table created.

SQL> INSERT INTO SALESMAN VALUES(1000,'JOHN','BANGALORE',25);
1 row created.

SQL> INSERT INTO SALESMAN VALUES(2000,'RAVI','BANGALORE',20);
1 row created.

SQL> INSERT INTO SALESMAN VALUES(3000,'KUMAR','MYSORE',15);
1 row created.

SQL> INSERT INTO SALESMAN VALUES(4000,'SMITH','DELHI',30);
1 row created.

SQL> INSERT INTO SALESMAN VALUES(5000,'HARSHA','HYDRABAD',15);
1 row created.

SQL> SELECT * FROM SALESMAN;

SALESMAN_ID	NAME	CITY	COMMISSION
1000	JOHN	BANGALORE	25
2000	RAVI	BANGALORE	20
3000	KUMAR	MYSORE	15
4000	SMITH	DELHI	30
5000	HARSHA	HYDRABAD	15

SQL> CREATE TABLE CUSTOMER(Customer_id NUMBER(10) PRIMARY KEY, Cust_Name VARCHAR(20), City VARCHAR(20), Grade NUMBER,Salesman_id INT REFERENCES SALESMAN(SALESMAN_ID));

Table created.

SQL> INSERT INTO CUSTOMER VALUES(10,'PREETHI','BANGALORE',100,1000);
1 row created.

SQL> INSERT INTO CUSTOMER VALUES(11,'VIVEK','MANGALORE',300,1000);
1 row created.

SQL> INSERT INTO CUSTOMER VALUES(12,'BHASKAR','CHENNAI',400,2000);
1 row created.

SQL> INSERT INTO CUSTOMER VALUES(13,'CHETHAN','BANGALORE',200,2000);
1 row created.

SQL> INSERT INTO CUSTOMER VALUES(14,'MAMATHA','BANGALORE',400,3000);
1 row created.

SQL> SELECT * FROM CUSTOMER;

CUSTOMER_ID	CUST_NAME	CITY	GRADE	SALESMAN_ID
10	PREETHI	BANGALORE	100	1000
11	VIVEK	MANGALORE	300	1000
12	BHASKAR	CHENNAI	400	2000
13	CHETHAN	BANGALORE	200	2000
14	MAMATHA	BANGALORE	400	3000

SQL> CREATE TABLE ORDERS(Ord_No NUMBER, Purchase_Amt NUMBER, Ord_Date DATE, Customer_id INT REFERENCES

CUSTOMER(CUSTOMER_ID),Salesman_id INT REFERENCES SALESMAN(SALESMAN_ID));

Table created.

SQL> INSERT INTO ORDERS VALUES(50,5000,'04-MAY-17',10,1000);
1 row created.

SQL> INSERT INTO ORDERS VALUES(51,450,'20-JAN-17',10,2000);
1 row created.

SQL> INSERT INTO ORDERS VALUES(52,1000,'24-FEB-17',13,2000);
1 row created.

SQL> INSERT INTO ORDERS VALUES(53,3500,'13-APR-17',14,3000);
1 row created.

SQL> INSERT INTO ORDERS VALUES(54,550,'09-MAR-17',12,2000);
1 row created.

SQL> SELECT * FROM ORDERS;

ORD_NO	PURCHASE_AMT	ORD_DATE	CUSTOMER_ID	SALESMAN_ID
50	5000	04-MAY-17	10	1000
51	450	20-JAN-17	10	2000
52	1000	24-FEB-17	13	2000
53	3500	13-APR-17	14	3000
54	550	09-MAR-17	12	2000

1. Count the customers with grades above Bangalore's Average.

SQL> SELECT GRADE, COUNT (DISTINCT CUSTOMER_ID) FROM CUSTOMER GROUP BY GRADE HAVING GRADE > (SELECT AVG(GRADE) FROM CUSTOMER WHERE CITY='BANGALORE');

GRADE	COUNT(DISTINCTCUSTOMER_ID)
300	1
400	2

2. Find the name and numbers of all salesmen who had more than one customer

SQL> SELECT SALESMAN_ID, NAME FROM SALESMAN A WHERE 1 < (SELECT COUNT (*) FROM CUSTOMER WHERE SALESMAN_ID=A.SALESMAN_ID);

SALESMAN_ID	NAME
100	JOHN
2000	RAVI

3. List all salesmen and indicate those who have and don't have customers in their cities (Use UNION operation.)

SQL> SELECT SALESMAN.SALESMAN_ID, NAME, CUST_NAME,COMMISSION FROM SALESMAN, CUSTOMER WHERE SALESMAN.CITY = CUSTOMER.CITY UNION SELECT SALESMAN_ID, NAME, 'NO MATCH', COMMISSION FROM SALESMAN WHERE NOT CITY = ANY (SELECT CITY FROM CUSTOMER) ORDER BY 2 DESC;

SALESMAN_ID	NAME	CUST_NAME	COMMISSION
4000	SMITH	NO MATCH	30
2000	RAVI	CHETHAN	20
2000	RAVI	MAMATHA	20
2000	RAVI	PREETHI	20
3000	KUMAR	NO MATCH	15
1000	JOHN	CHETHAN	25
1000	JOHN	MAMATHA	25
1000	JOHN	PREETHI	25
5000	HARSHA	NO MATCH	15

9 rows selected.

4. Create a view that finds the salesman who has the customer with the highest order of the day.

SQL> CREATE VIEW A AS SELECT B.ORD_DATE, A.SALESMAN_ID, A.NAME FROM SALESMAN A, ORDERS B WHERE A.SALESMAN_ID = B.SALESMAN_ID AND B.PURCHASE_AMT=(SELECT MAX (PURCHASE_AMT) FROM ORDERS C WHERE C.ORD_DATE = B.ORD_DATE);

View created.
SQL> SELECT * FROM A;

ORD_DATE	SALESMAN_ID	NAME
04-MAY-17	1000	JOHN
20-JAN-17	2000	RAVI
24-FEB-17	2000	RAVI
13-APR-17	3000	KUMAR
09-MAR-17	2000	RAVI

5. Demonstrate the DELETE operation by removing salesmen with id 1000. All his orders must also be deleted.
SQL> DELETE FROM SALESMAN WHERE SALESMAN_ID=1000;
1 row deleted

EXPERIMENT 4

DCL AND TCL

SQL> CREATE TABLE STUDENT(ROLLNO NUMBER(5),FIRSTNAME VARCHAR(5),LASTNAME VARCHAR(20));

Table created.

SQL> INSERT INTO STUDENT VALUES(60,'TOM','EMPHREM');

1 row created.

SQL> INSERT INTO STUDENT VALUES (18,'ANJU','SAJI');

1 row created.

SQL> INSERT INTO STUDENT VALUES (10,'AMMU','RAJU');

1 row created.

1. DCL

SQL> CREATE USER C##TOM IDENTIFIED BY 123;
User created.

GRANT
SQL> GRANT CREATE SESSION TO C##TOM;
Grant succeeded.

SQL> GRANT SELECT,DELETE ON STUDENT TO C##TOM;
Grant succeeded.

SQL> CONNECT;
Enter user-name: C##TOM
Enter password:
Connected.
SQL> SELECT * FROM SYSTEM.STUDENT;

   ROLLNO FIRST LASTNAME
---------- ----- --------------------
       60 TOM   EMPHREM
       18 ANJU SAJI
       10 AMMU RAJU

SQL> DELETE FROM SYSTEM.STUDENT WHERE ROLLNO=10;
1 row deleted.

REVOKE
SQL> CONNECT;
Enter user-name: SYSTEM
Enter password:
Connected.

SQL> REVOKE DELETE ON STUDENT FROM C##TOM;
Revoke succeeded.

SQL> CONNECT;
Enter user-name: C##TOM
Enter password:
Connected.
SQL> DELETE FROM SYSTEM.STUDENT WHERE ROLLNO=18;
DELETE FROM SYSTEM.STUDENT WHERE ROLLNO=18
              *
ERROR at line 1:
ORA-01031: insufficient privileges

SQL> CONNECT;
Enter user-name: SYSTEM
Enter password:
Connected.
SQL> DROP USER C##TOM;
User dropped.

2. TCL

SQL> SELECT * FROM STUDENT;

```
   ROLLNO FIRST LASTNAME
---------- ----- --------------------
    60 TOM   EMPHREM
    18 ANJU  SAJI
```

COMMIT
SQL> INSERT INTO STUDENT VALUES (10,'AMMU','RAJU');
1 row created.

SQL> SELECT * FROM STUDENT;
```
   ROLLNO FIRST LASTNAME
---------- ----- --------------------
    60 TOM   EMPHREM
    18 ANJU  SAJI
    10 AMMU  RAJU
```


SQL> COMMIT;
Commit complete.

SAVEPOINT
SQL> INSERT INTO STUDENT VALUES (11,'TOBIN','MATHEW');
1 row created.

SQL> SAVEPOINT ROLLNO;
Savepoint created.

ROLLBACK
SQL> INSERT INTO STUDENT VALUES (15,'JOSE','KJ');
1 row created.

SQL> SELECT * FROM STUDENT;
  ROLLNO FIRST LASTNAME
---------- ----- --------------------
    60 TOM   EMPHREM
    18 ANJU SAJI
    15 JOSE KJ
    10 AMMU RAJU
    11 TOBIN MATHEW

SQL> ROLLBACK WORK TO ROLLNO;
Rollback complete.

SQL> SELECT * FROM STUDENT;
  ROLLNO FIRST LASTNAME
---------- ----- --------------------
    60 TOM   EMPHREM
    18 ANJU SAJI
    10 AMMU RAJU
    11 TOBIN MATHEW

EXPERIMENT 5

VIEWS

SQL> CREATE TABLE employee(SSN VARCHAR2(20),FNAME VARCHAR2(20),LNAME VARCHAR2(20),ADDRESS VARCHAR2(20),SEX VARCHAR(1),SALARY NUMBER(38));

Table created.

SQL> insert into employee values('abc','Amrutha','biju','abc','F',25000); 1 row created.

SQL> insert into employee values('dbc','Anite','jose','jjjk','F',25000); 1 row created.

SQL> insert into employee values('cbc','Anna','maria','asd','F',25000); 1 row created.

SQL> insert into employee values(' bbc','Bharathi','S','sss','F',25000); 1 row created.

1. Creating a views (with and without check option)
SQL> CREATE VIEW sales_staff AS SELECT FNAME,SSN FROM employee; View

created.

2. Selecting from a view.
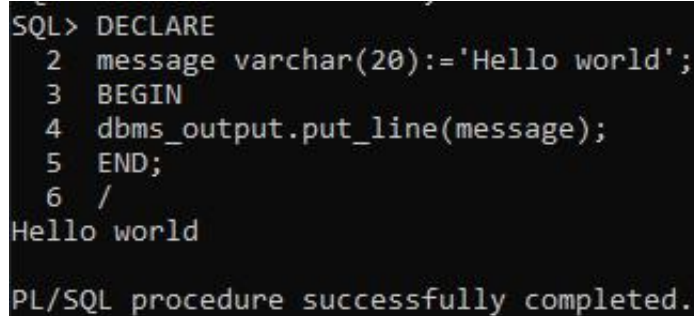
SQL> select * from sales_staff;

FNAME   SSN

-------------------- --------------------

Amrutha   abc

Anite   dbc

Anna   cbc

Bharathi   bdc

EXPERIMENT 6

PL/SQL Programs

1. Write a Pl/SQL program to print Hello world

SQL> SET SERVEROUTPUT ON;

SQL> DECLARE

2 message varchar(20):='Hello world';

3 BEGIN

4 dbms_output.put_line(message);

5 END;

6 /

Hello world

PL/SQL procedure successfully completed.

```
SQL> DECLARE
  2  message varchar(20):='Hello world';
  3  BEGIN
  4  dbms_output.put_line(message);
  5  END;
  6  /
Hello world

PL/SQL procedure successfully completed.
```


2. Write a PL/SQL block to find the maximum number from the given three numbers.

SQL>DECLARE

a number;

b number;

c number;

BEGIN

a:=&a;

b:=&b;

c:=&c;

if(a>b and a>c)then

dbms_output.put_line('a is maximum and the value is '||a);

elsif(b>a and b>c)then

dbms_output.put_line('b is maximum and the value is '||b);

else

dbms_output.put_line('c is maximum and the value is '||c);

end if;

END;

/

PL/SQL procedure successfully completed.

```
Enter value for a: 10
old    6: a:=&a;
new    6: a:=10;
Enter value for b: 20
old    7: b:=&b;
new    7: b:=20;
Enter value for c: 5
old    8: c:=&c;
new    8: c:=5;
b is maximum and the value is 20

PL/SQL procedure successfully completed.
```

3. Write a Pl/SQL program to print integers from 1 to 10 by using PL/SQL FOR loop.

SQL>DECLARE

n NUMBER:=10;

BEGIN

FOR i in 1..n LOOP

dbms_output.put_line(i);

END LOOP;

END;

/

```
1
2
3
4
5
6
7
8
9
10

PL/SQL procedure successfully completed.
```

4. Write a program to accept a number and find the sum of the digits .

```
SQL>declare
n number(5):=&n;
s number:=0;
r number(2):=0;
begin
while n!=0
loop
r:=mod(n,10);
s:=s+r;
n:=trunc(n/10);
end loop;
dbms_output.put_line('sum of digits of given number is '||s);
end;
/
```

```
Enter value for n: 243
old    2: n number(5):=&n;
new    2: n number(5):=243;
sum of digits of given number is 9

PL/SQL procedure successfully completed.

SQL>
```

5. Find the greatest number of inputs from the console.

```
SQL>declare
a number(2) :=&value_of_a;
b number(2) :=&value_of_b;
Begin
if a>b then
dbms_output.put_line(' Greatest Value is '||a);
elsif a<b then
dbms_output.put_line(' Greatest Value is '||b);
else
```

dbms_output.put_line(' Both no. are equal ');

end if;

END;

/

```
Enter value for value_of_a: 45
old    2: a number(2) :=&value_of_a;
new    2: a number(2) :=45;
Enter value for value_of_b: 78
old    3: b number(2) :=&value_of_b;
new    3: b number(2) :=78;
Greatest Value is 78

PL/SQL procedure successfully completed.

SQL>
```

6. Reading the values from EMployee table.

SQL>declare

efname employee.fname%type;

elname employee.lname%type;

esalary employee.salary%type;

begin

select fname,lname,salary

into efname,elname,esalary

from employee

where ssn=12;

dbms_output.put_line(efname||' '||elname||' '||esalary);

end;

/
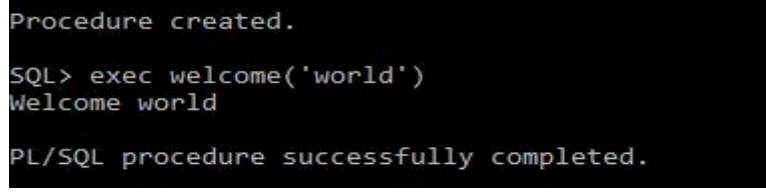
```
ajo jojo 25000

PL/SQL procedure successfully completed.
```

EXPERIMENT 7

PL/SQL Procedure and Functions

1.Procedure

SQL>SET SERVEROUTPUT ON;

SQL> create or replace procedure welcome(pname in varchar2)

is

begin

dbms_output.put_line('Welcome '||pname);

end;

/

```
Procedure created.

SQL> exec welcome('world')
Welcome world

PL/SQL procedure successfully completed.
```

2.Procedure

SQL> CREATE OR REPLACE PROCEDURE welcome_msg (p_name IN
       VARCHAR2,salary out number)

2   IS

3   BEGIN   salary:=10000;

4   dbms_output.put_line ('Welcome ' || p_name);

5   END;

6   /

Procedure created.

SQL> var sal number;

SQL> EXEC welcome_msg ('Amrutha',:sal);
Welcome Amrutha

PL/SQL procedure successfully completed. Print sal;

3. Function

SQL> CREATE OR REPLACE FUNCTION welcome_msg_func (p_name IN VARCHAR2) RETURN VARCHAR2

2   IS

3   BEGIN

4   RETURN ('Welcome '|| p_name);   END;

5   /

Function created.

SQL> DECLARE

    lv_msg VARCHAR2(250);

    BEGIN

    lv_msg:=welcome_msg_func('Amrutha');
    dbms_output.put_line(lv_msg);

    END;

    /

Welcome Amrutha

PL/SQL procedure successfully completed.
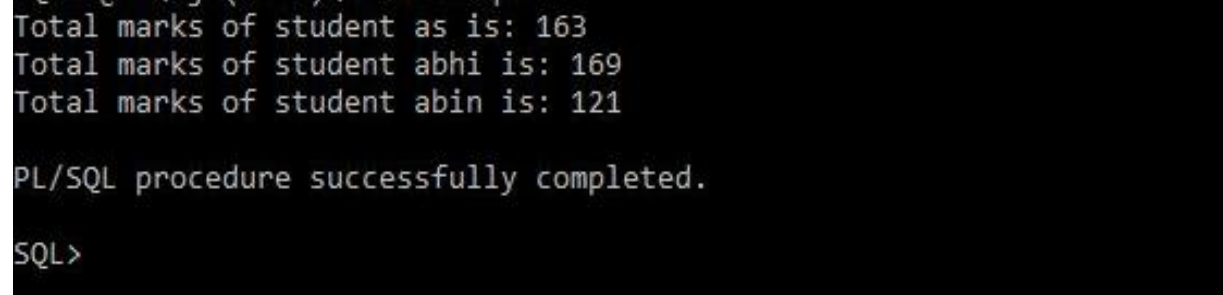
SQL> SELECT welcome_msg_func('Amrutha') FROM DUAL;

WELCOME_MSG_FUNC('Amrutha')

 Welcome Amrutha

EXPERIMENT 8

PL/SQL Cursor, Trigger

1.Cursor

```
declare
cursor stud_cursor is select * from stud_file;
stud_rec stud_cursor%rowtype;
total number:=0;
begin
open stud_cursor;
loop
fetch stud_cursor into stud_rec;
exit when stud_cursor%notfound ;
total:=stud_rec.m1+stud_rec.m2;
dbms_output.put_line('Total marks of student '||stud_rec.name||' is: '||total);
end loop;
end;
/
```

```
Total marks of student as is: 163
Total marks of student abhi is: 169
Total marks of student abin is: 121

PL/SQL procedure successfully completed.

SQL>
```

2.Trigger

create or replace trigger stud_trig after insert on stud_file

for each row

declare

tot number:=0;

begin

tot:=:new.m1+:new.m2;

insert into stud_mark values(:new.sid,tot);

DBMS_OUTPUT.PUT_LINE('AFTER INSERT trigger activated:');

end;

/

insert into stud_file values(5,'rani',40,45);

select * from stud_mark;

```
SQL> insert into stud_file values(5,'rani',40,45);
AFTER INSERT trigger activated:

1 row created.

SQL> select * from stud_mark;

      SID      TOTAL
---------- ----------
        5         85
```

EXPERIMENT 9

SQL OPERATIONS ON STUDENT TABLE

Student(rollno,name,date_of_birth,course_id,city,fees_paid,marks)
Course(course_id,course_sdesc,duration,course_fees)

1. Create above table with proper constraints(Enter atleast 5 valid records).

SQL> create table course(course_id varchar(5) primary key,course_desc
varchar(10),duration varchar(10),course_fees number(6));

Table created.

SQL> create table student(rollno number(2),name varchar(20),date_of_birth
date,course_id varchar(5)references course(course_id),city varchar(20),fees_paid
number(5),marks number(3));

Table created.

SQL> insert into course values('co1','bca','3year',100000);

1 row created.

SQL> insert into course values('co2','bba','3year',50000);

1 row created.

SQL> insert into course values('co3','mca','2year',200000);

1 row created.

SQL> insert into course values('co4','bcom','3year',80000);

1 row created.

SQL> insert into course values('co5','btech','4year',300000);

1 row created.

SQL> insert into student values(01,'ammu','15-aug-87','co1','pala',10000,75);

1 row created.

SQL> insert into student values(02,'anu','16-dec-86','co2','pala',5000,60);

1 row created.

SQL> insert into student values(03,'manu','15-aug-87','co3','kottayam',20000,45);

1 row created.

SQL> insert into student values(04,'vinu','12-dec-99','co4','idukki',15000,55);

1 row created.

SQL> insert into student values(05,'maya','11-jan-91','co5','kottayam',0,35);

1 row created.

SQL> select *from student;

ROLLNO	NAME	DOB	COURSE	CITY	FEES_PAID	MARKS
1	ammu	15-AUG-87	co1	pala	10000	75
2	anu	16-DEC-86	co2	pala	5000	60
3	manu	15-AUG-87	co3	kottayam	20000	45
4	vinu	12-DEC-99	co4	idukki	15000	55
5	maya	11-JAN-91	co5	kottayam	0	35

SQL> select *from course;

COURSE	COURSE_ DES	DURATION	COURSE_FEES
co1	bca	3year	100000
co2	bba	3year	50000
co3	mca	2year	200000
co4	bcom	3year	80000
co5	btech	4year	300000

2. List details of student whose birth date is 15th august 87.

SQL> select *from student where date_of_birth='15-aug-87';

ROLLNO	NAME	DATE_OF_B	COURSE	CITY	FEES_PAID	MARKS
1	ammu	15-AUG-87	co1	pala	10000	75
3	manu	15-AUG-87	co3	kottayam	20000	45

3. Display details of students whose marks are less than 50 and not paid a fee.

SQL> select * from student where fees_paid=0 AND marks<50;

ROLLNO	NAME	DATE_OF_B	COURSE	CITY	FEES_PAID	MARKS
5	maya	11-JAN-91	co5	kottayam	0	35

4.Display city wise count of students.

SQL> select city,count(city)from student group by city;

CITY   COUNT(CITY)

----------   ---------------------

kottayam   2

pala   2

idukki   1

5. Display total fees paid.

SQL> select sum(fees_paid)as total_fees from student;

TOTAL_FEES

50000

6. write PL/SQL block to display the name and mark of the top student.

SQL> set serveroutput on

SQL> declare

2 name student.name%type;

3 marks student.marks%type;

4 begin

5 select name,marks into name,marks from student where marks=(select

max(marks)from student);

6 dbms_output.put_line('name'||name||'marks'||marks);

7 end;

8 /

output

name ammu marks 75

PL/SQL procedure successfully completed.

EXPERIMENT 10

MONGODB CRUD OPERATIONS

1. Student Database

Create database, Create collection, insert data, find, find one, sort , limit, skip, distinct, projection.

Create a student database with the fields: (SRN, Sname , Degree, Sem , CGPA)

//create database
employee> use Sreyas
switched to db Sreyas

//create collection
Sreyas> db.createCollection('stud1col1');
{ ok: 1 }

>db.stud1col1.insert({srn:110,sname:"Rahul",degree:"BCA",sem:6, CGPA:7.9}) OR

> doc1=({srn:110,sname:"Rahul",degree:"BCA",sem:6,CGPA:7.9})
>db.studcol1.insert (doc1)

Note:

insert 10 documents.

2. display all the documents

>db.studcol1.find()

3. Display all the students in BCA

>db.studcol1.find({degree:"BCA"})

4. Display all the students in ascending order

>db. studcol1.find().sort({sname:1})

5. Display first 5 students

>db. studcol1.find().limit(5)

6. display students 5,6,7

>db. studcol1.find().skip(4).limit(3)

7. list the degree of student "Rahul"

>db. studcol1.find({degree:1, sname:"Rahul"})

8. Display students details of 5,6,7 in descending order of percentage

>db. studcol1.find().sort({CGPA:-1}).skip(4).limit(3)

9. Display the number of students in BCA

>db. studcol1.find({degree:"BCA"}).count()

10. Display all the degrees without _id

>db. studcol1.find({},{_id:0})

11. Display all the distinct degrees

>db. studcol1.distinct("degree")

12. Display all the BCA students with CGPA greater than 6, but less than 7.5

>db. studcol1.find(degree:"BCA",{CGPA:{$gt:6, $lt:7.5}})

13. Display all the students in BCA and in 6th Sem

>db. studcol1.find({$and:[{degree:"BCA"},{sem:6}]})

2. Employee Database

Update modifiers ($set, $unset, $inc, $push, $pushAll, $pull, $pullAll, $addToSet)

Create an employee database with the fields: {eid, ename, dept, desig,salary, yoj,

address{ dno, street, locality, city}}

> use empdb9 switched to db empdb9

> doc1 = {eid:001, ename:"Rahul", dept:"production", desig:"developer", salary:30000,

yoj:2015, address:{dno:397, street:2, locality:"rmnagar", city:"bangalore"} } {

"eid" : 1,

"ename" : "Rahul",

"dept" : "production",

"desig" : "developer",

"salary" : 30000,

"yoj" : 2015,

"address" : {

"dno" : 397,

"street" : 2,

"locality" : "rmnagar",

"city" : "bangalore"

} }

>db.emp09.insert(doc1)

WriteResult({ "nInserted" : 1 }) Note:

insert 10 documents.

1. Display all the employees with salary in range (50000, 75000)

>db.emp09.find({salary:{$gt:50000, $lt:75000}})

2. Display all the employees with desig developer

>db.emp09.find({desig:"developer"})

3. Display the Salary of "Rahul"

>db.emp09.find({ename:"Rahul"},{salary:1})

4. Display the city of employee "Rahul"

>db.emp09.find({ename:"Rahul"},{"address.city":1})

5. Update the salary of developers by 5000 increment

>db.emp09.update({desig:"developer"},{$inc:{"salary":5000}})

6. Add field age to employee "Rahul"

>db.emp09.update({ename:"Rahul"},{$set:{age:"22"}})

7. Remove YOJ from "Rahul"

>db.emp09.update({ename:"Rahul"},{$unset:{yoj:1}})

8. Add an array field project to "Rahul"

>db.emp09.update({ename:"Rahul"},{$push:{projects:"p1"}})

9. Add p2 and p3 project to "Rahul"

>db.emp09.update({ename:"Rahul"},{$pushAll:{projects:["p2","p3"]}})

10. Remove p3 from "Rahul"

>db.emp09.update({ename:"Rahul"},{$pull:{projects:"p3"}})

11. Add a new embedded object "contacts" with "email" and "phone" as array objects to "Rahul"

>db.emp09.update({ename:"Rahul"},{$push:{contacts:{phone:"9036240380",
email:"abc@gmail.com"}}})

12. Add two phone numbers to "Rahul"

>db.emp09.update({ename:"Rahul"},{$addToSet:{"contact.phone":[9738751143,9880730784]}})

EXPERIMENT 11

AGGREGATE FUNCTIONS

//USE DATABASE

test> use sreyas

switched to db sreyas

//CREATE COLLECTION WEBSITE

sreyas> db.createCollection('website')
{ ok: 1 }

//INSERT VALUES IN WEBSITE

sreyas> db.website.insertOne({Rollno:1,Name:'Harsh',Amount:10000,url:'www.yahoo.com'})
{

  acknowledged: true,

  insertedId: ObjectId('662624aa86aa3e9f2e9f990a')

}

sreyas> db.website.insertOne({Rollno:2,Name:'Jitesh',Amount:20000,url:'www.yahoo.com'})

{

  acknowledged: true,

  insertedId: ObjectId('662624d286aa3e9f2e9f990b')

}

sreyas> db.website.insertOne({Rollno:3,Name:'Rina',Amount:30000,url:'www.google.com'})

{

  acknowledged: true,

  insertedId: ObjectId('662624ed86aa3e9f2e9f990c')

}

sreyas> db.website.insertOne({Rollno:4,Name:'Ash',Amount:40000,url:'www.gmail.com'})

{

  acknowledged: true,

  insertedId: ObjectId('6626251a86aa3e9f2e9f990d')

}

sreyas> db.website.insertOne({Rollno:5,Name:'Sreyas',Amount:10000,url:'www.pvg.com'})

{

  acknowledged: true,

```
  insertedId: ObjectId('6626253186aa3e9f2e9f990e')

}

sreyas> db.website.insertOne({Rollno:6,Name:'Ash',Amount:20000,url:'www.gmail.com'})

{

  acknowledged: true,

  insertedId: ObjectId('6626280586aa3e9f2e9f990f')

}
```

//SUM AGGREGATE

```
sreyas> db.website.aggregate({$group:{_id:'$Name','total':{$sum:'$Amount'}}})

[

  { _id: 'Harsh', total: 10000 },

  { _id: 'Rina', total: 30000 },

  { _id: 'Jitesh', total: 20000 },

  { _id: 'Ash', total: 60000 },

  { _id: 'Sreyas', total: 10000 }

]
```

//AVG AGGREGATE

```
sreyas> db.website.aggregate({$group:{_id:'$Name','Total':{$avg:'$Amount'}}})

[

  { _id: 'Rina', Total: 30000 },

  { _id: 'Jitesh', Total: 20000 },

  { _id: 'Sreyas', Total: 10000 },

  { _id: 'Harsh', Total: 10000 },

  { _id: 'Ash', Total: 30000 }

]
```

//MIN AGGREGATION

```
sreyas> db.website.aggregate({$group:{_id:'$Name','Total':{$min:'$Amount'}}})

[

  { _id: 'Jitesh', Total: 20000 },

  { _id: 'Rina', Total: 30000 },

  { _id: 'Sreyas', Total: 10000 },

  { _id: 'Harsh', Total: 10000 },

  { _id: 'Ash', Total: 20000 }]
```

//MAX AGGREGATION

sreyas> db.website.aggregate({$group:{_id:'$Name','Total':{$max:'$Amount'}}})

[

  { _id: 'Jitesh', Total: 20000 },

  { _id: 'Rina', Total: 30000 },

  { _id: 'Sreyas', Total: 10000 },

  { _id: 'Harsh', Total: 10000 },

  { _id: 'Ash', Total: 40000 }

]

//FIRST AGGREGATION

sreyas> db.website.aggregate({$group:{_id:'$Name',Total:{$first:'$Amount'}}})

[

  { _id: 'Harsh', Total: 10000 },

  { _id: 'Rina', Total: 30000 },

  { _id: 'Jitesh', Total: 20000 },

  { _id: 'Ash', Total: 40000 },

  { _id: 'Sreyas', Total: 10000 }

]

//LAST AGGREGATION

sreyas> db.website.aggregate({$group:{_id:'$Name',Total:{$last:'$Amount'}}})

[

  { _id: 'Jitesh', Total: 20000 },

  { _id: 'Rina', Total: 30000 },

  { _id: 'Sreyas', Total: 10000 },

  { _id: 'Harsh', Total: 10000 },

  { _id: 'Ash', Total: 20000 }

]

//PUSH AGGREGATION

sreyas> db.website.aggregate({$group:{_id:'$Name',Total:{$push:'$Amount'}}})

[

  { _id: 'Rina', Total: [30000] },

  { _id: 'Jitesh', Total: [20000] },

  { _id: 'Sreyas', Total: [10000] },

  { _id: 'Harsh', Total: [10000] },

  { _id: 'Ash', Total: [40000, 20000] }

]

//COUNT AGGREGATION

sreyas> db.website.aggregate({$group:{_id:'$Name',Total:{$sum:1}}})

[

  { _id: 'Harsh', Total: 1 },

  { _id: 'Jitesh', Total: 1 },

  { _id: 'Rina', Total: 1 },

  { _id: 'Ash', Total: 2 },

  { _id: 'Sreyas', Total: 1 }

]

//ADDTOSET AGGREGATE

sreyas> db.website.aggregate({$group:{_id:'$Name',Total:{$addToSet:'$Amount'}}})

[

  { _id: 'Harsh', Total: [10000] },

  { _id: 'Jitesh', Total: [20000] },

  { _id: 'Rina', Total: [30000] },

  { _id: 'Ash', Total: [40000, 20000] },

  { _id: 'Sreyas', Total: [10000] }

]

EXPERIMENT 12

PYTHON MONGODB CONNECTION

PyMongo library, which is a Python driver for MongoDB, a popular NoSQL database.

1. Importing the necessary modules: import pymongo

2. Creating a MongoDB client and connecting to a database:

   myclient = pymongo.MongoClient("mongodb://localhost:27017/")

3. Accessing a specific database:

   mydb = myclient["mydatabase"]

4. Accessing a specific collection within the database:

   mycol = mydb["customers"]

EXPERIMENT 13

PYTHON MONGODB INSERTION

1. **Inserting a single document into the collection:**

myd = {"name": "Divya","address":"highway37"}

q = mycol.insert_one(myd) print(q.inserted_id)

2. **Inserting multiple documents into the collection:**

mydict = [

{"name": "John", "address": "highway 37"},

{"name": "Aby", "address": "Cross 30"},

{"name": "Jerry", "address": "River Road 45"}

]

x = mycol.insert_many(mydict) print(x.inserted_ids)

EXPERIMENT 14

PYTHON MONGODB DISPLAY

1. Retrieving a single document from the collection:

y = mycol.find_one() print(y)

2. Retrieving all documents from the collection:

for z in mycol.find():

print(z)

3. Retrieving documents while excluding the "name" field:

for a in mycol.find({}, {"name": 0}):

print(a)

4. Deleting a document from the collection:

myquery = {"name": "John"} mycol.delete_one(myquery)

5. Deleting multiple documents from the collection:

c = mycol.delete_many({})

print(c.deleted_count, "documents/rows deleted")

EXPERIMENT 15

PYTHON MONGODB UPDATION

1.   Updating a document in the collection:

myquery = {"address": "Canyon 123"}

newval = {"$set": {"address": "highway37"}}

f = mycol.update_one(myquery, newval)

print(f.modified_count, "Document updated")

2. Updating multiple documents :

myquery = {"address": "highway 37"}

newval={"$set":{"address":"Canyon123"}}

f = mycol.update_many(myquery, newval)

print(f.modified_count, "Document updated")

3. Limiting the number of retrieved documents to 2:

for p in mycol.find().limit(2):

print(p)

4. Sorting the retrieved documents by the "name" field:

mydoc = mycol.find().sort("name")

for l in mydoc: print(l)

EXPERIMENT 16

PYTHON MONGODB DELETION

1. Dropping the collection (deleting all documents within it):

 mycol.drop()