NATIONAL INSTITUTE OF TECHNOLOGY MANIPUR
(An Autonomous Institute under MHRD, Govt. of India)

Database Management System Lab (CS332)
VI Semester

Submitted by:
Name: **Nongmaithem Miranda Devi**
Enrollment Number: **21103003**

Submitted to: **Merina Mam**

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
NATIONAL INSTITUTE OF TECHNOLOGY MANIPUR

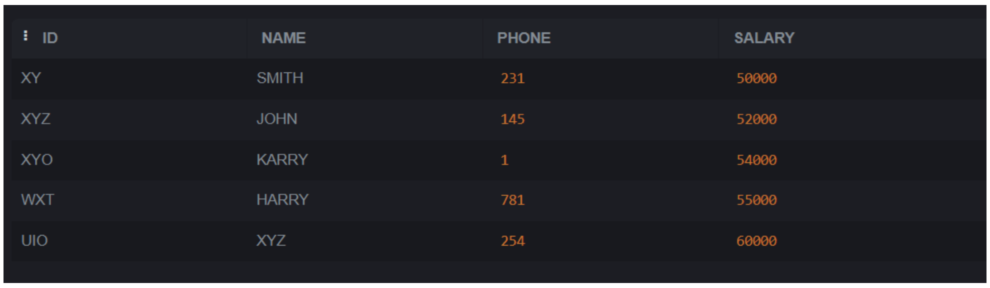
EXPERIMENT 1

AIM: To create a simple creation of an employee tables and perform queries.

TABLE CREATION CODE:

```
CREATE TABLE EMPLOYEES (ID VARCHAR (5),

NAME VARCHAR (10), PHONE INT, SALARY INT);

INSERT INTO EMPLOYEES (ID, NAME, PHONE, SALARY) VALUES('XY','SMITH',231,50000);

INSERT INTO EMPLOYEES (ID, NAME, PHONE, SALARY) VALUES('XYZ','JOHN',145,52000);

INSERT INTO EMPLOYEES (ID, NAME, PHONE, SALARY) VALUES('XYO','KARRY',001,54000);

INSERT INTO EMPLOYEES (ID, NAME, PHONE, SALARY) VALUES('WXT','HARRY',781,55000);

INSERT INTO EMPLOYEES (ID, NAME, PHONE, SALARY) VALUES('UIO','XYZ',157,60000);
```

OUTPUT:



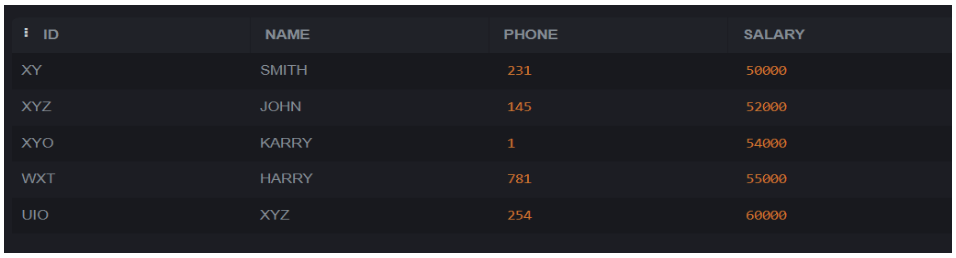
ID	NAME	PHONE	SALARY
XY	SMITH	231	50000
XYZ	JOHN	145	52000
XYO	KARRY	1	54000
WXT	HARRY	781	55000
UIO	XYZ	254	60000

QUERY 1: To display all employee's details

CODE:

```
SELECT * FROM EMPLOYEES;
```

OUTPUT:



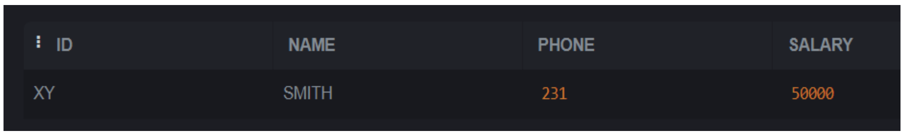
ID	NAME	PHONE	SALARY
XY	SMITH	231	50000
XYZ	JOHN	145	52000
XYO	KARRY	1	54000
WXT	HARRY	781	55000
UIO	XYZ	254	60000

QUERY 2: To display all the details of one particular employee.

CODE:

```
SELECT *

FROM EMPLOYEES WHERE ID='XY';
```

OUTPUT:



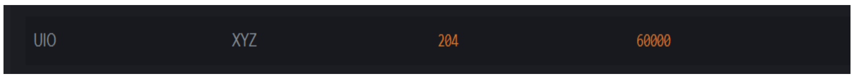
ID	NAME	PHONE	SALARY
XY	SMITH	231	50000

QUERY 3: To modify the phone number of an employee whose name is “XYZ”.

CODE:

```
SELECT * FROM EMPLOYEES

WHERE ID='XY';
```

OUTPUT:



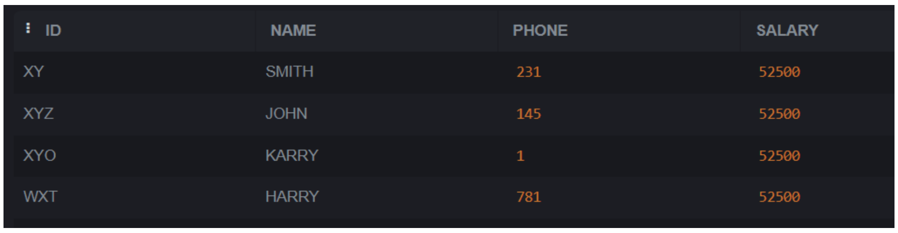
UIO	XYZ	204	60000

QUERY 4: To modify the salaries of all employees whose salary is Rs. 50000 to Rs. 55000

CODE:

```
UPDATE EMPLOYEES

SET SALARY=52500

WHERE SALARY>=50000 AND SALARY<=55000;
```

OUTPUT:



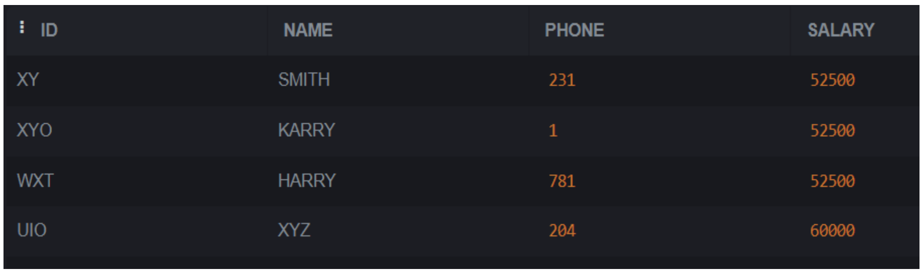
ID	NAME	PHONE	SALARY
XY	SMITH	231	52500
XYZ	JOHN	145	52500
XYO	KARRY	1	52500
WXT	HARRY	781	52500

QUERY 5: To delete the employee’s record whose id is “XYZ”.

CODE:

```
DELETE FROM EMPLOYEES

WHERE ID='XYZ';
```

OUTPUT:



ID	NAME	PHONE	SALARY
XY	SMITH	231	52500
XYO	KARRY	1	52500
WXT	HARRY	781	52500
UIO	XYZ	204	60000

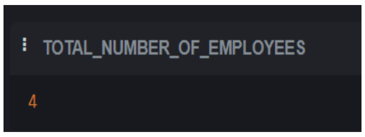
QUERY 6: To count the total number of employees.

CODE:

SELECT COUNT (*) AS TOTAL_NUMBER_OF_EMPLOYEES

FROM EMPLOYEES;

OUTPUT:



TOTAL_NUMBER_OF_EMPLOYEES
4

QUERY 7: To add another column, start_date to the table and also insert the values for each employee.

CODE:

ALTER TABLE EMPLOYEES

ADD START_DATE;

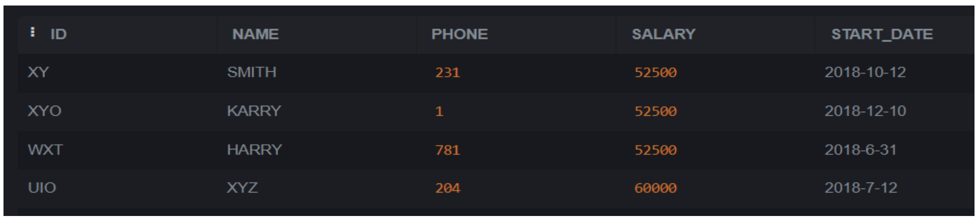
UPDATE EMPLOYEES SET START_DATE = ‘2018-10-12’ WHERE ID='XY';

UPDATE EMPLOYEES SET START_DATE = ‘2018-12-10’ WHERE ID='XYO';

UPDATE EMPLOYEES SET START_DATE = ‘2018-6-31’ WHERE ID='WXT';

UPDATE EMPLOYEES SET START_DATE = ‘2018-7-12’ WHERE ID='UIO';

OUTPUT:



ID	NAME	PHONE	SALARY	START_DATE
XY	SMITH	231	52500	2018-10-12
XYO	KARRY	1	52500	2018-12-10
WXT	HARRY	781	52500	2018-6-31
UIO	XYZ	204	60000	2018-7-12

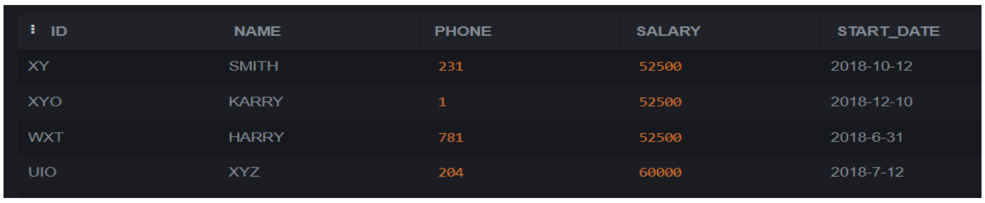
QUERY 8: To list all the employees according to increasing order of their salaries.

CODE:

SELECT * FROM EMPLOYEES

ORDER BY SALARY;

OUTPUT:



ID	NAME	PHONE	SALARY	START_DATE
XY	SMITH	231	52500	2018-10-12
XYO	KARRY	1	52500	2018-12-10
WXT	HARRY	781	52500	2018-6-31
UIO	XYZ	204	60000	2018-7-12

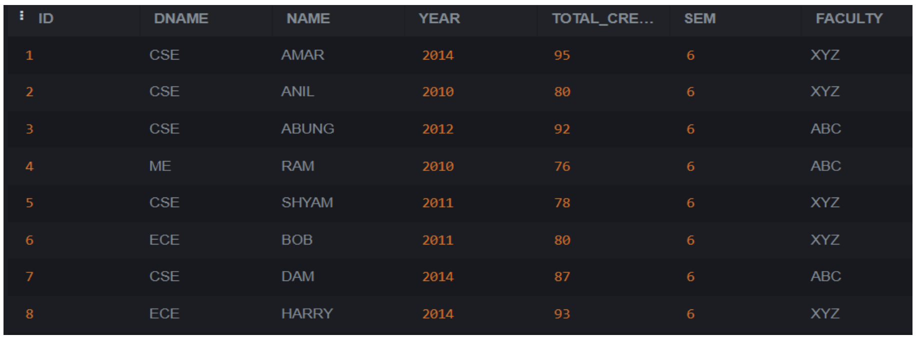
EXPERIMENT 2

AIM: To create a student record keeping system. The system should also record the details of faculties and other relevant information if any and also perform the queries.

TABLE CREATION CODE:

```
CREATE TABLE STUDENTS (ID INT PRIMARY KEY, DNAME VARCHAR(30), NAME VARCHAR(10), YEAR INT, TOTAL_CREDITS INT, SEM CHAR(1), FACULTY VARCHAR(5));

 INSERT INTO STUDENTS VALUES (01, 'CSE', 'AMAR', 2014, 95, '6', 'XYZ');

 INSERT INTO STUDENTS VALUES (02, 'CSE', 'ANIL', 2010, 80, '6', 'XYZ');

 INSERT INTO STUDENTS VALUES (03, 'CSE', 'ABUNG', 2012, 92, '6', 'ABC');

 INSERT INTO STUDENTS VALUES (04, 'ME', 'RAM', 2010, 76, '6', 'ABC');

 INSERT INTO STUDENTS VALUES (05, 'CSE', 'SHYAM', 2011, 78, '6', 'XYZ');

 INSERT INTO STUDENTS VALUES (06, 'ECE', 'BOB', 2011, 80, '6', 'XYZ');

 INSERT INTO STUDENTS VALUES (07, 'CSE', 'DAM', 2014, 87, '6', 'ABC');

 INSERT INTO STUDENTS VALUES (08, 'ECE', 'HARRY', 2014, 93, '6', 'XYZ');
```

OUTPUT:



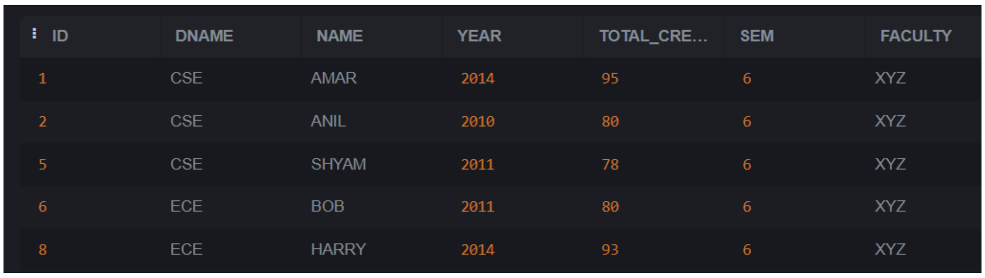
ID	DNAME	NAME	YEAR	TOTAL_CRE...	SEM	FACULTY
1	CSE	AMAR	2014	95	6	XYZ
2	CSE	ANIL	2010	80	6	XYZ
3	CSE	ABUNG	2012	92	6	ABC
4	ME	RAM	2010	76	6	ABC
5	CSE	SHYAM	2011	78	6	XYZ
6	ECE	BOB	2011	80	6	XYZ
7	CSE	DAM	2014	87	6	ABC
8	ECE	HARRY	2014	93	6	XYZ

QUERY 1: To display all the students taught by a particular faculty “XYZ”.

CODE:

```
SELECT * FROM STUDENTS

 WHERE FACULTY='XYZ';
```

OUTPUT:



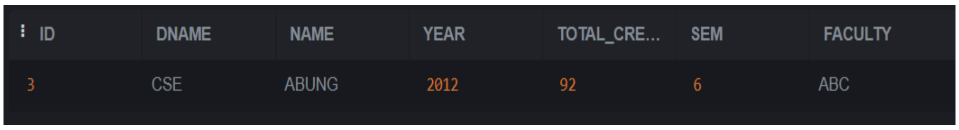
ID	DNAME	NAME	YEAR	TOTAL_CRE...	SEM	FACULTY
1	CSE	AMAR	2014	95	6	XYZ
2	CSE	ANIL	2010	80	6	XYZ
5	CSE	SHYAM	2011	78	6	XYZ
6	ECE	BOB	2011	80	6	XYZ
8	ECE	HARRY	2014	93	6	XYZ

QUERY 2: To display all the students of a particular department (e.g. “CSE”) enrolled in the year 2012.

CODE:

```
SELECT * FROM STUDENTS
 WHERE DNAME='CSE' AND YEAR=2012;
```

OUTPUT:



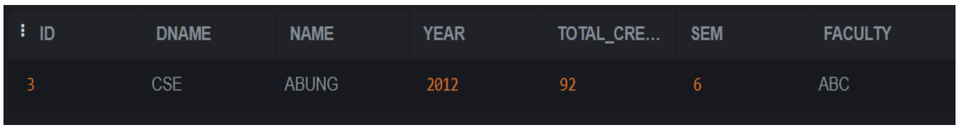
ID	DNAME	NAME	YEAR	TOTAL_CRE...	SEM	FACULTY
3	CSE	ABUNG	2012	92	6	ABC

QUERY 3: To display all the students enrolled in the year 2012 whose total credits is greater than 80.

CODE:

```
SELECT * FROM STUDENTS
 WHERE YEAR = 2012 AND TOTAL_CREDITS >= 80;
```

OUTPUT:



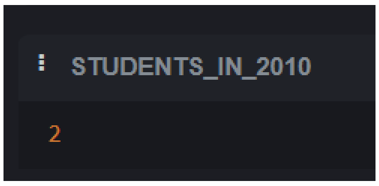
ID	DNAME	NAME	YEAR	TOTAL_CRE...	SEM	FACULTY
3	CSE	ABUNG	2012	92	6	ABC

QUERY 4: To count the number of students enrolled in 2010.

CODE:

```
SELECT COUNT (*) AS STUDENTS_IN_2010 FROM STUDENTS
 WHERE YEAR = 2010;
```

OUTPUT:



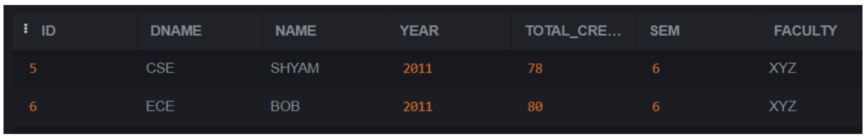
STUDENTS_IN_2010
2

QUERY 5: To display all the students’ details enrolled in 2011 and taught by a particular faculty “XYZ”.

CODE:

```
SELECT * FROM STUDENTS
  WHERE YEAR = 2011 AND FACULTY = 'XYZ';
```

OUTPUT:



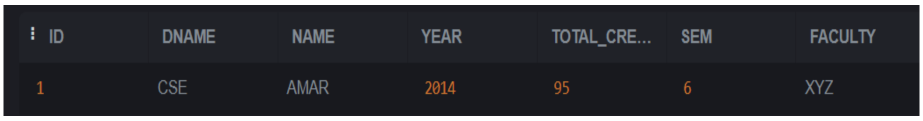
ID	DNAME	NAME	YEAR	TOTAL_CRE...	SEM	FACULTY
5	CSE	SHYAM	2011	78	6	XYZ
6	ECE	BOB	2011	80	6	XYZ

QUERY 6: To display the details of the student who topped in sixth semester in 2014.

CODE:

```
SELECT * FROM STUDENTS
  WHERE TOTAL_CREDITS = (SELECT MAX (TOTAL_CREDITS)
                        FROM STUDENTS WHERE SEM = '6') AND YEAR = 2014;
```

OUTPUT:



ID	DNAME	NAME	YEAR	TOTAL_CRE...	SEM	FACULTY
1	CSE	AMAR	2014	95	6	XYZ

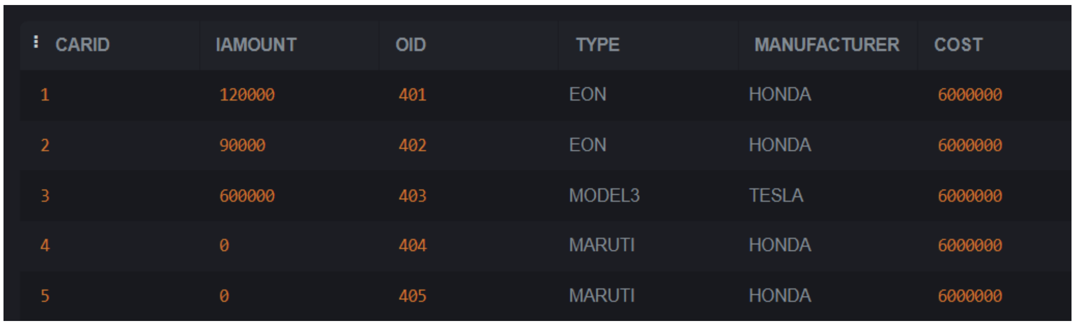
EXPERIMENT 3

AIM: To create a database for car insurance and execute the given queries.

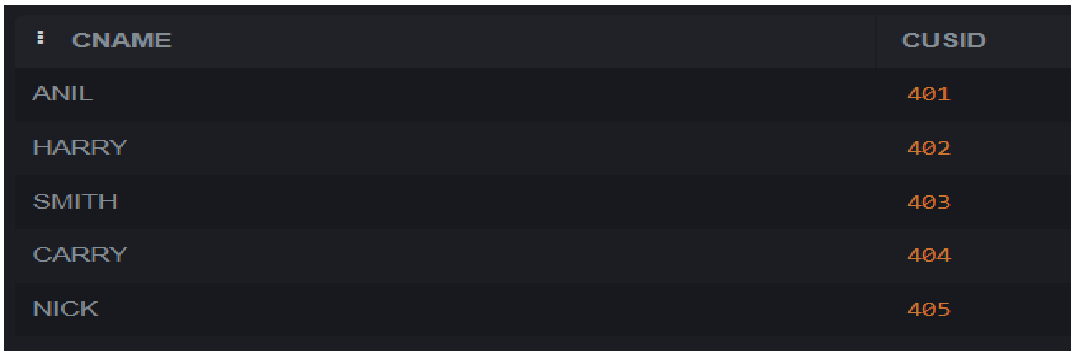
TABLE CREATION CODE:

```
CREATE TABLE CAR (CARID INT PRIMARY KEY, IAMOUNT INT, OID INT, TYPE VARCHAR(10),
MANUFACTURER VARCHAR(10), COST INT );
INSERT INTO CAR VALUES(01, 120000, 401, 'EON', 'HONDA', 6000000);
INSERT INTO CAR VALUES(02, 90000, 402, 'EON', 'HONDA', 6000000);
INSERT INTO CAR VALUES(03, 600000, 403, 'MODEL3', 'TESLA', 6000000);
INSERT INTO CAR VALUES(04, 0, 404, 'MARUTI', 'HONDA', 6000000);
INSERT INTO CAR VALUES(05, 0, 405, 'MARUTI', 'HONDA', 6000000);

CREATE TABLE CUSTOMER (CNAME VARCHAR(10), CUSID INT PRIMARY KEY);
INSERT INTO CUSTOMER VALUES('ANIL', 401);
INSERT INTO CUSTOMER VALUES('HARRY', 402);
INSERT INTO CUSTOMER VALUES('SMITH', 403);
INSERT INTO CUSTOMER VALUES('CARRY', 404);
INSERT INTO CUSTOMER VALUES('NICK', 405);
```

OUTPUT:



CARID	IAMOUNT	OID	TYPE	MANUFACTURER	COST
1	120000	401	EON	HONDA	6000000
2	90000	402	EON	HONDA	6000000
3	600000	403	MODEL3	TESLA	6000000
4	0	404	MARUTI	HONDA	6000000
5	0	405	MARUTI	HONDA	6000000



CNAME	CUSID
ANIL	401
HARRY	402
SMITH	403
CARRY	404
NICK	405

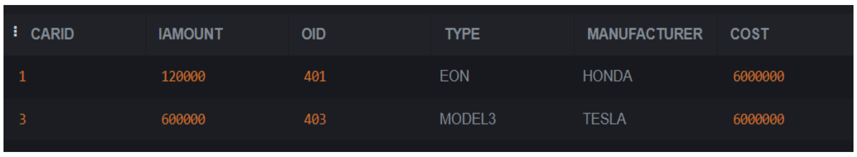
QUERY 1: To display all the cars having insurance amount greater than Rs. 100000.

CODE:

SELECT * FROM CAR

WHERE IAMOUNT > 100000;

OUTPUT:



CARID	IAMOUNT	OID	TYPE	MANUFACTURER	COST
1	120000	401	EON	HONDA	6000000
3	600000	403	MODEL3	TESLA	6000000

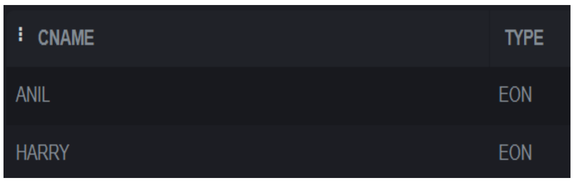
QUERY 2: To display all the customers having same cars of a particular type (e.g. “Eon”).

CODE:

SELECT C.CNAME, S.TYPE FROM CUSTOMER C JOIN CAR S

ON C.CUSID=S.OID WHERE S.TYPE = 'EON';

OUTPUT:



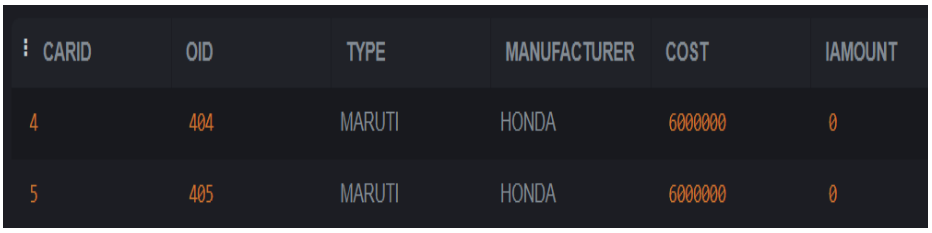
CNAME	TYPE
ANIL	EON
HARRY	EON

QUERY 3: To list all the cars which are not insured by their owners.

CODE:

SELECT S.CARID, S.OID, S.TYPE, S.MANUFACTURER, S.COST, S.IAMOUNT FROM CUSTOMER C JOIN CAR S ON C.CUSID=S.OID WHERE S.IAMOUNT = 0;

OUTPUT:



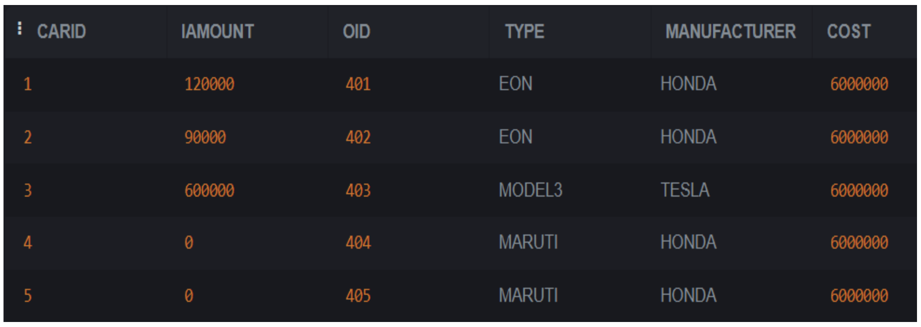
CARID	OID	TYPE	MANUFACTURER	COST	IAMOUNT
4	404	MARUTI	HONDA	6000000	0
5	405	MARUTI	HONDA	6000000	0

QUERY 4: To display all cars details in increasing order of cost.

CODE:

SELECT * FROM CAR ORDER BY COST;

OUTPUT:



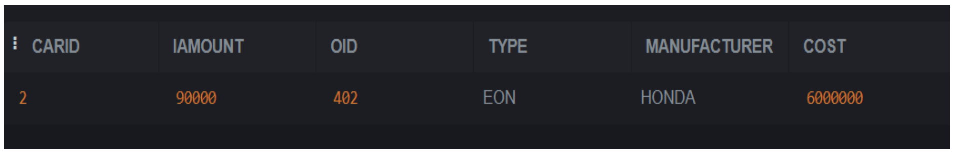
CARID	IAMOUNT	OID	TYPE	MANUFACTURER	COST
1	120000	401	EON	HONDA	6000000
2	90000	402	EON	HONDA	6000000
3	600000	403	MODEL3	TESLA	6000000
4	0	404	MARUTI	HONDA	6000000
5	0	405	MARUTI	HONDA	6000000

QUERY 5: To display all the cars manufactured by a particular company (e.g. Honda) and which are insured.

CODE:

SELECT * FROM CAR WHERE MANUFACTURER = 'HONDA' AND IAMOUNT > 0;

OUTPUT:



CARID	IAMOUNT	OID	TYPE	MANUFACTURER	COST
2	90000	402	EON	HONDA	6000000

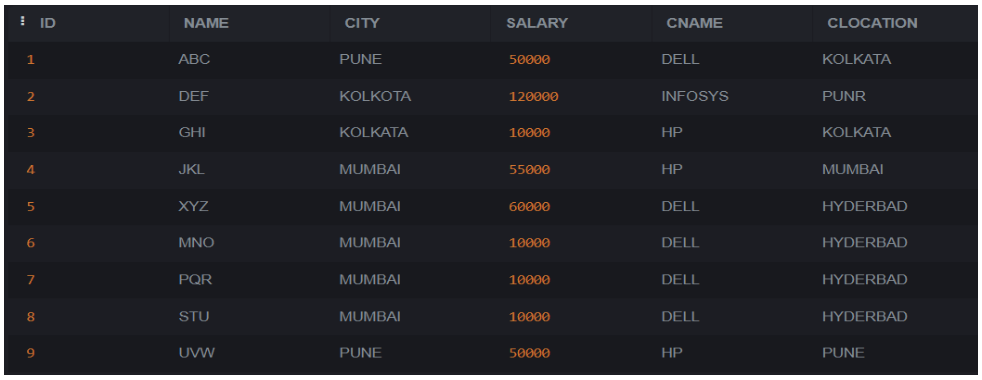
EXPERIMENT 4

AIM: To create a database for recording employee's details working in different companies and perform the given queries.

TABLE CREATION CODE:

```
CREATE TABLE EMPLOYEE( ID INT PRIMARY KEY, NAME VARCHAR(10), CITY VARCHAR(10), SALARY INT,

CNAME VARCHAR(10), CLOCATION VARCHAR(10));

INSERT INTO EMPLOYEE VALUES(01,'ABC','PUNE',50000,'DELL','KOLKATA');

INSERT INTO EMPLOYEE VALUES(02,'DEF','KOLKOTA',120000,'INFOSYS','PUNR');

INSERT INTO EMPLOYEE VALUES(03,'GHI','KOLKATA',10000,'HP','KOLKATA');

INSERT INTO EMPLOYEE VALUES(04,'JKL','MUMBAI',55000,'HP','MUMBAI');

INSERT INTO EMPLOYEE VALUES(05,'XYZ','MUMBAI',60000,'DELL','HYDERBAD');

INSERT INTO EMPLOYEE VALUES(06,'MNO','MUMBAI',10000,'DELL','HYDERBAD');

INSERT INTO EMPLOYEE VALUES(07,'PQR','MUMBAI',10000,'DELL','HYDERBAD');

INSERT INTO EMPLOYEE VALUES(08,'STU','MUMBAI',10000,'DELL','HYDERBAD');

INSERT INTO EMPLOYEE VALUES(09,'UVW','PUNE',50000,'HP','PUNE');
```

OUTPUT:



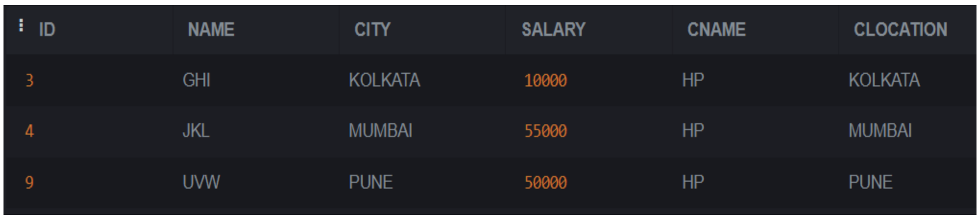
ID	NAME	CITY	SALARY	CNAME	CLOCATION
1	ABC	PUNE	50000	DELL	KOLKATA
2	DEF	KOLKOTA	120000	INFOSYS	PUNR
3	GHI	KOLKATA	10000	HP	KOLKATA
4	JKL	MUMBAI	55000	HP	MUMBAI
5	XYZ	MUMBAI	60000	DELL	HYDERBAD
6	MNO	MUMBAI	10000	DELL	HYDERBAD
7	PQR	MUMBAI	10000	DELL	HYDERBAD
8	STU	MUMBAI	10000	DELL	HYDERBAD
9	UVW	PUNE	50000	HP	PUNE

QUERY 1: To display all employees who work for a particular company (e.g. Hp)

CODE:

```
SELECT * FROM EMPLOYEE WHERE CNAME = 'HP';
```

OUTPUT:



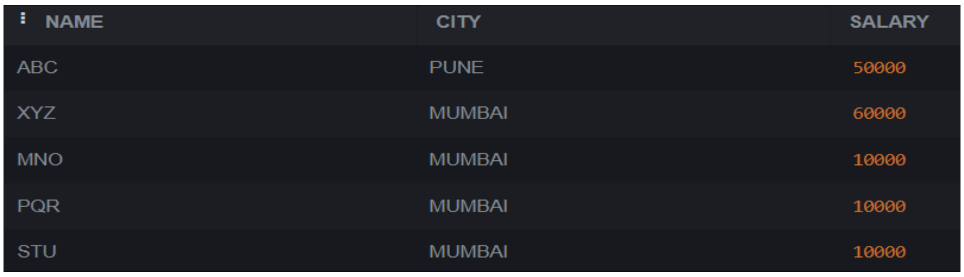
ID	NAME	CITY	SALARY	CNAME	CLOCATION
3	GHI	KOLKATA	10000	HP	KOLKATA
4	JKL	MUMBAI	55000	HP	MUMBAI
9	UVW	PUNE	50000	HP	PUNE

QUERY 2: To display the names, cities and salaries of all employees who work for a particular company (e.g. Dell)

CODE:

SELECT NAME, CITY, SALARY FROM EMPLOYEE WHERE CNAME = 'DELL';

OUTPUT:



NAME	CITY	SALARY
ABC	PUNE	50000
XYZ	MUMBAI	60000
MNO	MUMBAI	10000
PQR	MUMBAI	10000
STU	MUMBAI	10000

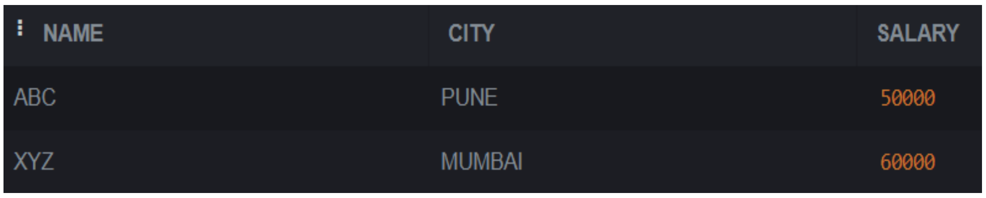
QUERY 3: To find the employees names, cities and salaries of all employees who work for a particular company and are more than 50 thousand

CODE:

SELECT NAME, CITY, SALARY FROM EMPLOYEE

WHERE CNAME = 'DELL' AND SALARY >= 50000;

OUTPUT:



NAME	CITY	SALARY
ABC	PUNE	50000
XYZ	MUMBAI	60000

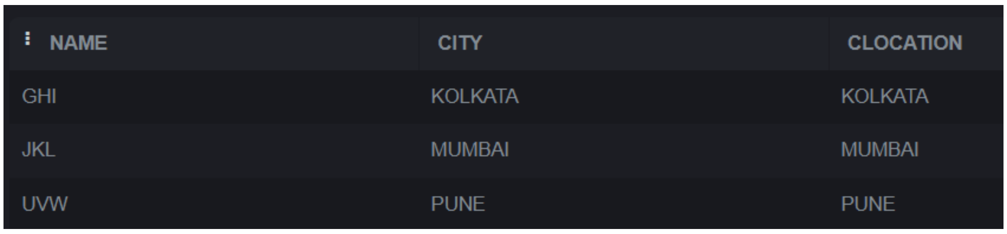
QUERY 4: To find all employees in the db who live in the same city as the companies's for which they work.

CODE:

SELECT M.NAME, M.CITY, M.CLOCATION FROM EMPLOYEE M JOIN EMPLOYEE E

ON M.ID=E.ID WHERE M.CITY = E.CLOCATION;

OUTPUT:



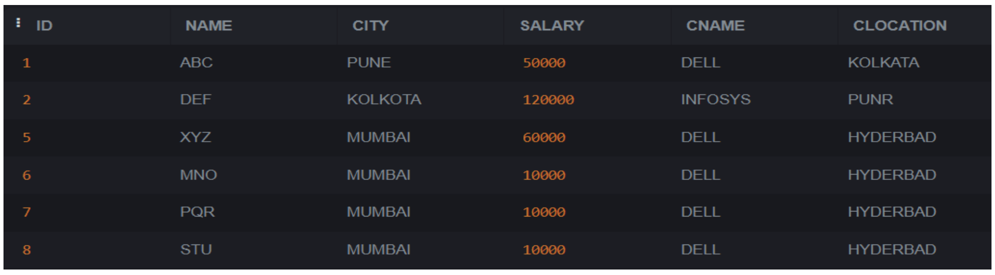
NAME	CITY	CLOCATION
GHI	KOLKATA	KOLKATA
JKL	MUMBAI	MUMBAI
UVW	PUNE	PUNE

QUERY 5: To find all the employees in the db who do not work for a particular company.

CODE:

SELECT * FROM EMPLOYEE WHERE CNAME != 'HP';

OUTPUT:



ID	NAME	CITY	SALARY	CNAME	CLOCATION
1	ABC	PUNE	50000	DELL	KOLKATA
2	DEF	KOLKOTA	120000	INFOSYS	PUNR
5	XYZ	MUMBAI	60000	DELL	HYDERBAD
6	MNO	MUMBAI	10000	DELL	HYDERBAD
7	PQR	MUMBAI	10000	DELL	HYDERBAD
8	STU	MUMBAI	10000	DELL	HYDERBAD

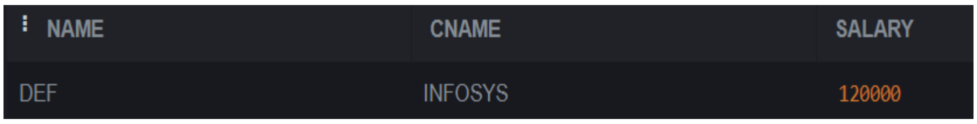
QUERY 6: To find all the employees in the db earning more than every other employee of some other company (e.g. dell).

CODE:

SELECT NAME, CNAME,SALARY FROM EMPLOYEE

WHERE SALARY > (SELECT MAX(SALARY) FROM EMPLOYEE WHERE CNAME = 'DELL');

OUTPUT:



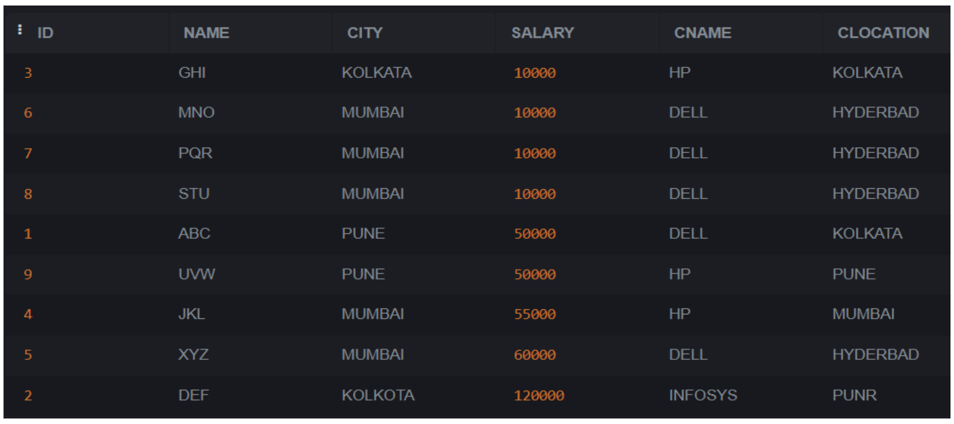
NAME	CNAME	SALARY
DEF	INFOSYS	120000

QUERY 7: To display all employees in increasing order of their salaries.

CODE:

SELECT * from EMPLOYEE ORDER by SALARY;

OUTPUT:



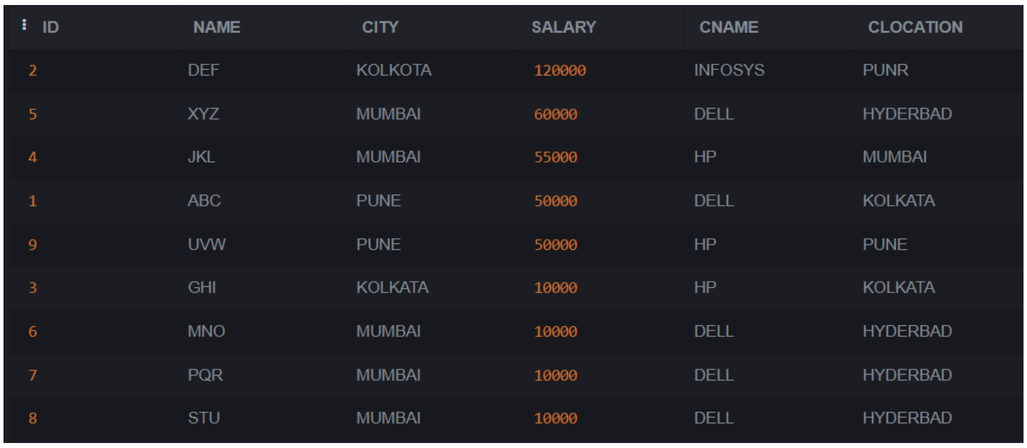
ID	NAME	CITY	SALARY	CNAME	CLOCATION
3	GHI	KOLKATA	10000	HP	KOLKATA
6	MNO	MUMBAI	10000	DELL	HYDERBAD
7	PQR	MUMBAI	10000	DELL	HYDERBAD
8	STU	MUMBAI	10000	DELL	HYDERBAD
1	ABC	PUNE	50000	DELL	KOLKATA
9	UVW	PUNE	50000	HP	PUNE
4	JKL	MUMBAI	55000	HP	MUMBAI
5	XYZ	MUMBAI	60000	DELL	HYDERBAD
2	DEF	KOLKOTA	120000	INFOSYS	PUNR

QUERY 8: To display all employees in decreasing order of their salaries.

CODE:

```
SELECT * FROM EMPLOYEE ORDER by SALARY DESC;
```

OUTPUT:



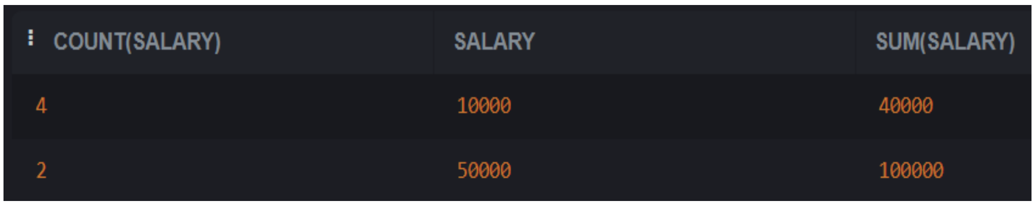
ID	NAME	CITY	SALARY	CNAME	CLOCATION
2	DEF	KOLKOTA	120000	INFOSYS	PUNR
5	XYZ	MUMBAI	60000	DELL	HYDERBAD
4	JKL	MUMBAI	55000	HP	MUMBAI
1	ABC	PUNE	50000	DELL	KOLKATA
9	UVW	PUNE	50000	HP	PUNE
3	GHI	KOLKATA	10000	HP	KOLKATA
6	MNO	MUMBAI	10000	DELL	HYDERBAD
7	PQR	MUMBAI	10000	DELL	HYDERBAD
8	STU	MUMBAI	10000	DELL	HYDERBAD

QUERY 9: To display the sum of salaries of employees getting same salary greater than ₹100000.

CODE:

```
SELECT COUNT(SALARY), SALARY, SUM(SALARY) FROM EMPLOYEE

GROUP BY SALARY HAVING COUNT(SALARY)>1;
```

OUTPUT:



COUNT(SALARY)	SALARY	SUM(SALARY)
4	10000	40000
2	50000	100000

EXPERIMENT 5

AIM: To create a university database and perform given queries. The database records the details of faculties, departments, students, courses, prerequisite for different courses, section, grade report of students and other relevant information.

TABLE CREATION CODE:

CREATE TABLE STUDENT(ID INT PRIMARY KEY, NAME VARCHAR(5), DEPT VARCHAR(3), GRADE CHAR(1), SEM CHAR(1), YEAR INT, CNO INT);

INSERT INTO STUDENT VALUES(01, 'ABC', 'CSE', 'A', '6', 2014, 1);

INSERT INTO STUDENT VALUES(02, 'XYZ', 'CSE', 'B', '6', 2014, 1);

INSERT INTO STUDENT VALUES(03, 'DEF', 'CE', 'A', '6', 2014, 2);

INSERT INTO STUDENT VALUES(04, 'GHI', 'CE', 'A', '6', 2014, 2);

INSERT INTO STUDENT VALUES(05, 'JKL', 'ECE', 'B', '6', 2014, 3);

INSERT INTO STUDENT VALUES(06, 'MNO', 'ECE', 'B', '6', 2014, 3);

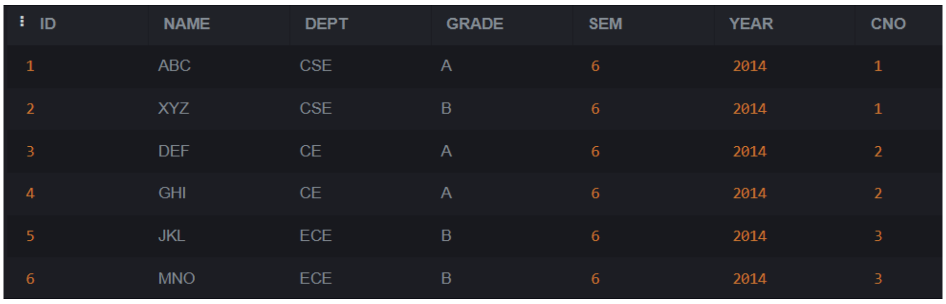
CREATE TABLE FACULTY(FID INT PRIMARY KEY, CNO INT, FNAME VARCHAR(5), DEPENDENT VARCHAR(2), SECTION CHAR(1), SALARY INT, CNAME VARCHAR(10));

INSERT INTO FACULTY VALUES(001,1,'ABC', 'D1', 'A', 50000, 'YUWAO');

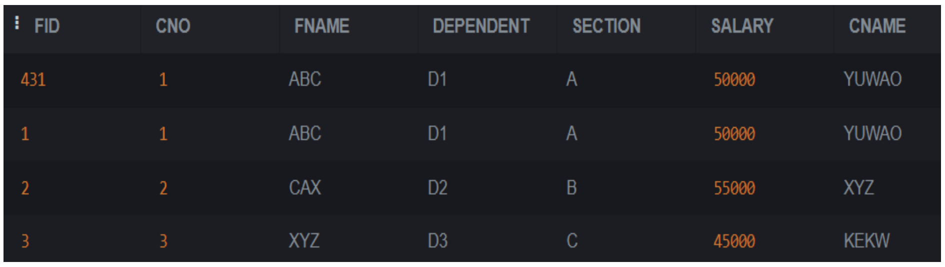
INSERT INTO FACULTY VALUES(002,2,'CAX', 'D2', 'B', 55000, 'XYZ');

INSERT INTO FACULTY VALUES(003,3,'XYZ', 'D3', 'C', 45000, 'KEKW');

OUTPUT:



ID	NAME	DEPT	GRADE	SEM	YEAR	CNO
1	ABC	CSE	A	6	2014	1
2	XYZ	CSE	B	6	2014	1
3	DEF	CE	A	6	2014	2
4	GHI	CE	A	6	2014	2
5	JKL	ECE	B	6	2014	3
6	MNO	ECE	B	6	2014	3



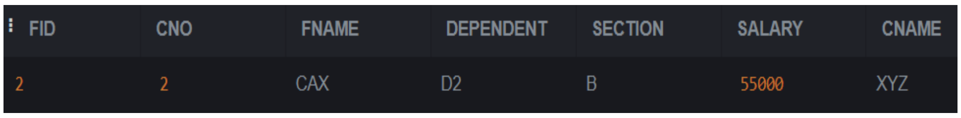
FID	CNO	FNAME	DEPENDENT	SECTION	SALARY	CNAME
431	1	ABC	D1	A	50000	YUWAO
1	1	ABC	D1	A	50000	YUWAO
2	2	CAX	D2	B	55000	XYZ
3	3	XYZ	D3	C	45000	KEKW

QUERY 1: To find the highest paid faculty.

CODE:

SELECT * FROM FACULTY WHERE SALARY IN (SELECT MAX(SALARY) FROM FACULTY);

OUTPUT:



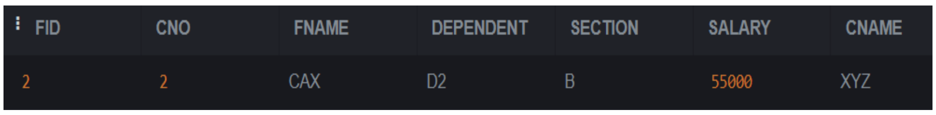
FID	CNO	FNAME	DEPENDENT	SECTION	SALARY	CNAME
2	2	CAX	D2	B	55000	XYZ

QUERY 2: To display all the faculties whose salaries are greater than the highest paid faculty of a particular department (e.g. CSE).

CODE:

SELECT * FROM FACULTY WHERE SALARY > (SELECT MAX(SALARY) FROM FACULTY F JOIN STUDENT S ON F.CNO=S.CNO WHERE S.DEPT = 'CSE');

OUTPUT:



FID	CNO	FNAME	DEPENDENT	SECTION	SALARY	CNAME
2	2	CAX	D2	B	55000	XYZ

QUERY 3: To display all the students who got “A” grade in a particular course “XYZ” of sixth semester for the year 2014.

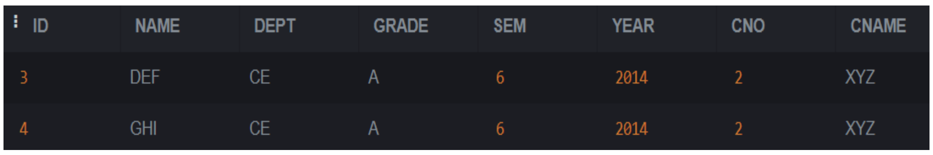
CODE:

SELECT S.ID, S.NAME, S.DEPT, S.GRADE, S.SEM, S.YEAR, S.CNO, F.CNAME

FROM FACULTY F JOIN STUDENT S ON F.CNO=S.CNO

WHERE S.GRADE = 'A' AND F.CNAME = 'XYZ' AND S.SEM='6' AND YEAR=2014;

OUTPUT:



ID	NAME	DEPT	GRADE	SEM	YEAR	CNO	CNAME
3	DEF	CE	A	6	2014	2	XYZ
4	GHI	CE	A	6	2014	2	XYZ

QUERY 4: To list all courses taught by faculty “XYZ” for 6th semester for 2014.

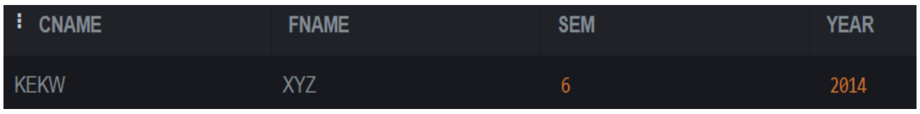
CODE:

SELECT F.CNAME, F.FNAME, S.SEM,S.YEAR

FROM FACULTY F JOIN STUDENT S

WHERE F.FNAME = 'XYZ' AND S.SEM = '6' AND S.YEAR= 2014;

OUTPUT:



CNAME	FNAME	SEM	YEAR
KEKW	XYZ	6	2014

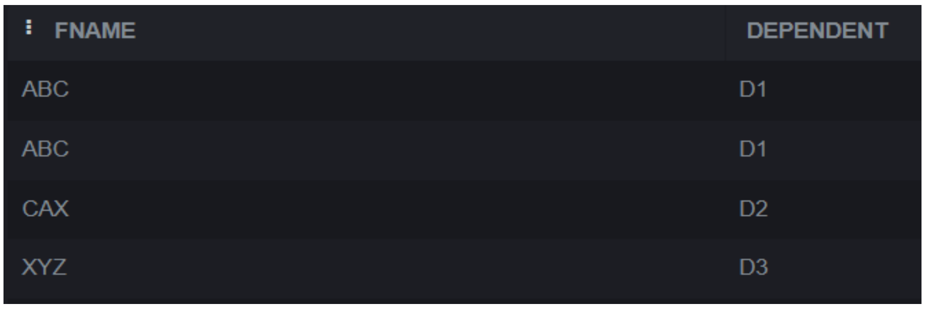
QUERY 5: To display all faculties along with their dependents.

CODE:

SELECT FNAME, DEPENDENT

FROM FACULTY;

OUTPUT:



FNAME	DEPENDENT
ABC	D1
ABC	D1
CAX	D2
XYZ	D3

QUERY 6: To display the details of faculties teaching courses along with the section details.

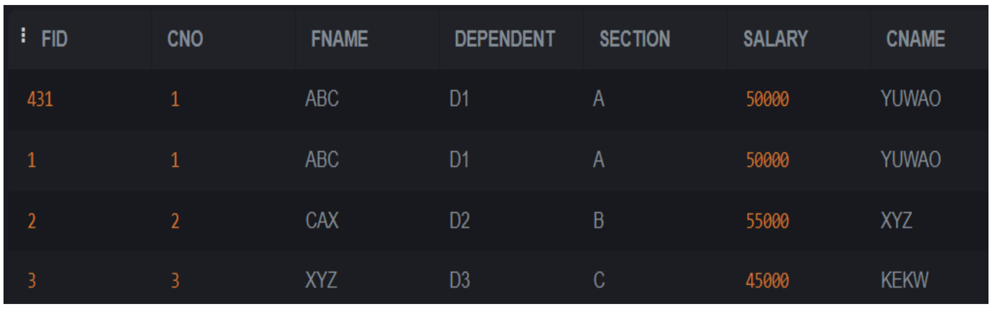
CODE:

SELECT *

FROM FACULTY

WHERE CNAME>0;

OUTPUT:



FID	CNO	FNAME	DEPENDENT	SECTION	SALARY	CNAME
431	1	ABC	D1	A	50000	YUWAO
1	1	ABC	D1	A	50000	YUWAO
2	2	CAX	D2	B	55000	XYZ
3	3	XYZ	D3	C	45000	KEKW

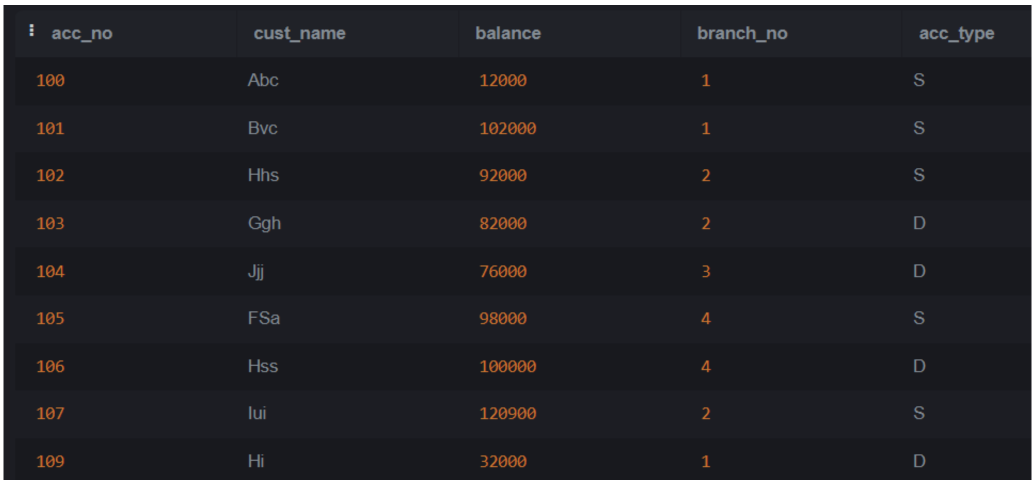
EXPERIMENT 6

AIM: To create a bank database and perform queries. The databes is organised into many branches. A customer can open different kinds of accounts in different branches. The account holder can enquire about the balance in his account. The database keeps track of a customer by his ID, name and address. Accounts (identified by account number) having a starting date and balance. The database keeps track of every transaction with details information about it.

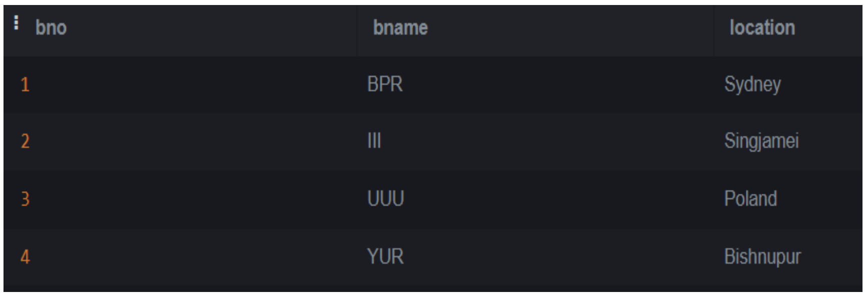
TABLE CREATION CODE:

```
CREATE TABLE Account(acc_no int PRIMARY KEY,cust_name varchar(10),
        balance int,branch_no int ,acc_type char(1));
INSERT INTO Account VALUES(100,'Abc',12000,1,'S');
INSERT INTO Account VALUES(101,'Bvc',102000,1,'S');
INSERT INTO Account VALUES(102,'Hhs',92000,2,'S');
INSERT INTO Account VALUES(103,'Ggh',82000,2,'D');
INSERT INTO Account VALUES(104,'Jjj',76000,3,'D');
INSERT INTO Account VALUES(105,'FSa',98000,4,'S');
INSERT INTO Account VALUES(106,'Hss',100000,4,'D');
INSERT INTO Account VALUES(107,'Iui',120900,2,'S');
INSERT INTO Account VALUES(109,'Hi',32000,1,'D');

CREATE TABLE Branch(bno int PRIMARY KEY, bname varchar(10),location varchar(10));
INSERT INTO Branch VALUES(1,'BPR','Sydney');
INSERT INTO Branch VALUES(2,'III','Singjamei');
INSERT INTO Branch VALUES(3,'UUU','Poland');
INSERT INTO Branch VALUES(4,'YUR','Bishnupur');

CREATE TABLE Transactions1(acc_no int,t_date varchar(10),PRIMARY KEY(acc_no,t_date));
INSERT into Transactions1 VALUES(101,'14-02-24');
INSERT into Transactions1 VALUES(102,'15-02-24');
INSERT into Transactions1 VALUES(103,'16-03-24');
INSERT into Transactions1 VALUES(104,'14-04-24');
INSERT into Transactions1 VALUES(105,'17-12-24');
INSERT into Transactions1 VALUES(106,'01-62-24');
```

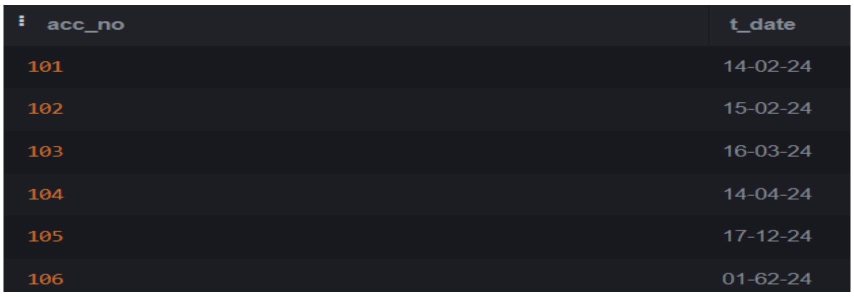
OUTPUT:



acc_no	cust_name	balance	branch_no	acc_type
100	Abc	12000	1	S
101	Bvc	102000	1	S
102	Hhs	92000	2	S
103	Ggh	82000	2	D
104	Jjj	76000	3	D
105	FSa	98000	4	S
106	Hss	100000	4	D
107	Iui	120900	2	S
109	Hi	32000	1	D



bno	bname	location
1	BPR	Sydney
2	III	Singjamei
3	UUU	Poland
4	YUR	Bishnupur



acc_no	t_date
101	14-02-24
102	15-02-24
103	16-03-24
104	14-04-24
105	17-12-24
106	01-62-24

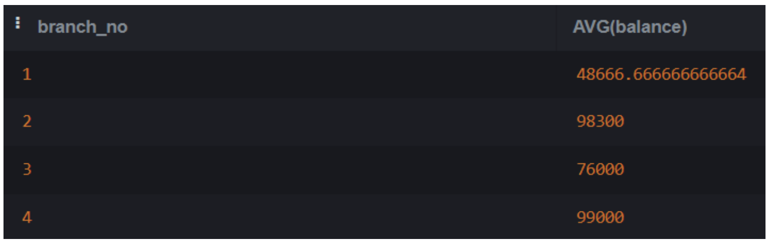
QUERY 1: To retrieve branch details with its average balance only if it is greater than 10000.

CODE:

SELECT branch_no, AVG(balance) FROM Account

GROUP by branch_no HAVING balance>10000;

OUTPUT:



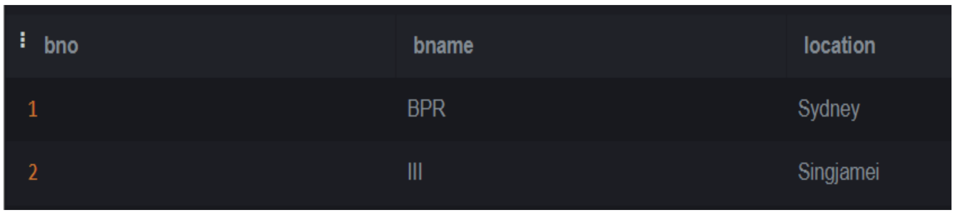
branch_no	AVG(balance)
1	48666.666666666664
2	98300
3	76000
4	99000

QUERY 2: To display the branch details located in a city starting with the letter 'S'.

CODE:

SELECT * FROM Branch WHERE location LIKE 'S%';

OUTPUT:



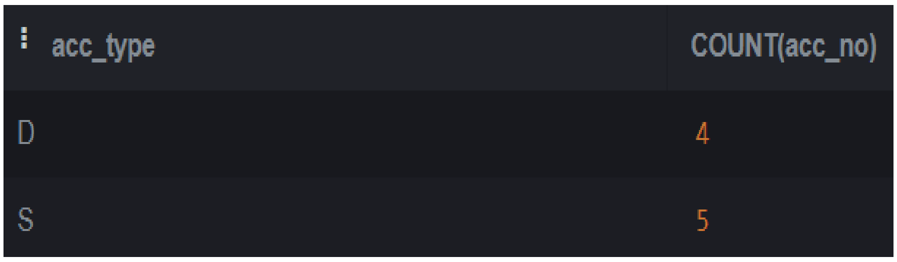
bno	bname	location
1	BPR	Sydney
2	III	Singjamei

QUERY 3: To retrieve the number of depositors in each branch.

CODE:

SELECT acc_type, COUNT(acc_no) FROM Account GROUP by acc_type;

OUTPUT:



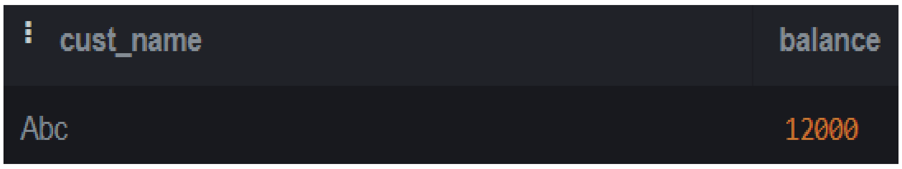
acc_type	COUNT(acc_no)
D	4
S	5

QUERY 4: To display the total account balance of the given customer name 'Abc.

CODE:

SELECT cust_name, balance FROM Account WHERE cust_name='Abc';

OUTPUT:



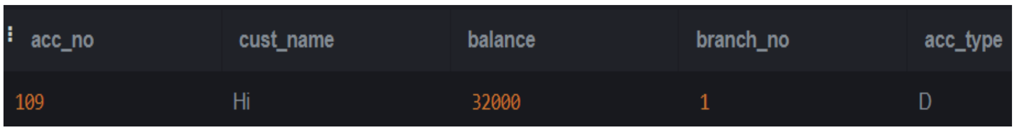
cust_name	balance
Abc	12000

QUERY 5: To display the details of all the customers whose account balance is between 30000 and 40000.

CODE:

SELECT * FROM Account WHERE balance>=30000 AND balance<=40000;

OUTPUT:



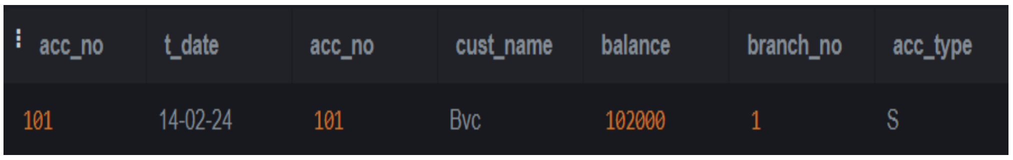
acc_no	cust_name	balance	branch_no	acc_type
109	Hi	32000	1	D

QUERY 6: To retrieve customer details who did transaction on 14th february 2017 along with the details of these transactions.

CODE:

```
SELECT * from Transactions1 T JOIN Account A

on T.acc_no=A.acc_no WHERE t_date='14-02-24';
```

OUTPUT:



acc_no	t_date	acc_no	cust_name	balance	branch_no	acc_type
101	14-02-24	101	Bvc	102000	1	S

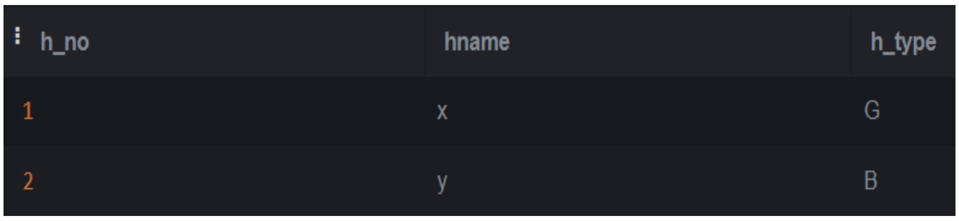
EXPERIMENT 7

AIM: To create a hostel mess database and perform given queries. The database keeps track of all the available hostel, mess menu, warden and student details. Each hostel has a unique number, name and type of hostel which gives information about whether the hostel is of girls or boys. Keeps track of the mess menu of each hostel to record in which day what special dishes are given for breakfast, lunch and dinner.

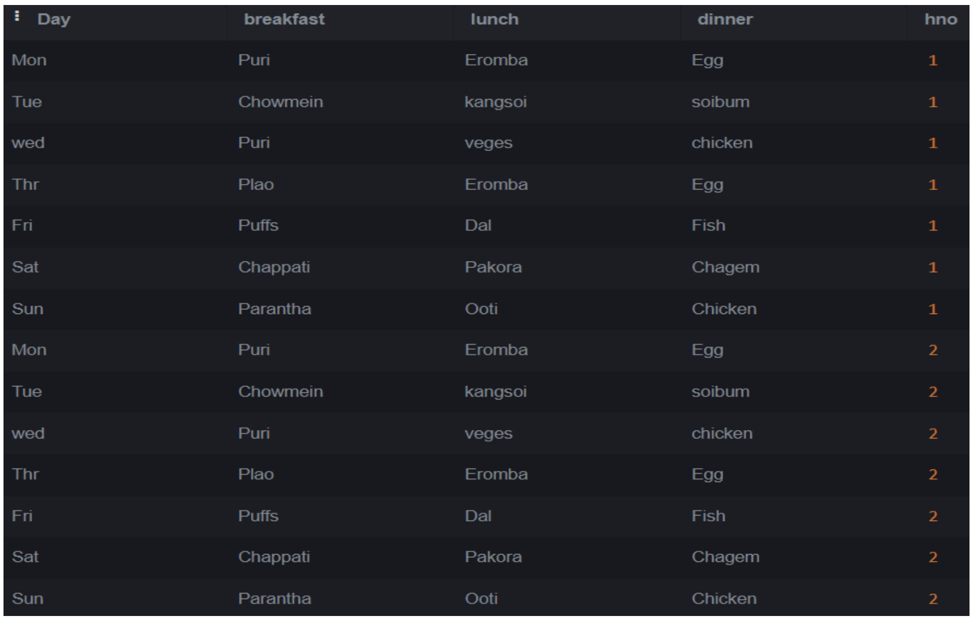
TABLE CREATION CODE:

```
CREATE TABLE Hostel(h_no int PRIMARY KEY,hname varchar(5),h_type char(1));

CREATE TABLE Mess(Day char(3),breakfast varchar(10),lunch varchar(10),dinner varchar(10),hno int );

CREATE TABLE Warden(wno int PRIMARY KEY,wname varchar(5),hno int);

CREATE TABLE Student(roll int PRIMARY KEY ,name varchar(5),gender char(1),hno int);


INSERT INTO Hostel VALUES(01,'x','G');

INSERT INTO Hostel VALUES(02,'y','B');


INSERT INTO Mess VALUES('Mon','Puri','Eromba','Egg',01);

INSERT INTO Mess VALUES('Tue','Chowmein','kangsoi','soibum',01);

INSERT INTO Mess VALUES('wed','Puri','veges','chicken',01);

INSERT INTO Mess VALUES('Thr','Plao','Eromba','Egg',01);

INSERT INTO Mess VALUES('Fri','Puffs','Dal','Fish',01);

INSERT INTO Mess VALUES('Sat','Chappati','Pakora','Chagem',01);

INSERT INTO Mess VALUES('Sun','Parantha','Ooti','Chicken',01);

INSERT INTO Mess VALUES('Mon','Puri','Eromba','Egg',02);

INSERT INTO Mess VALUES('Tue','Chowmein','kangsoi','soibum',02);

INSERT INTO Mess VALUES('wed','Puri','veges','chicken',02);

INSERT INTO Mess VALUES('Thr','Plao','Eromba','Egg',02);

INSERT INTO Mess VALUES('Fri','Puffs','Dal','Fish',02);

INSERT INTO Mess VALUES('Sat','Chappati','Pakora','Chagem',02);

INSERT INTO Mess VALUES('Sun','Parantha','Ooti','Chicken',02);

INSERT INTO Warden VALUES(420,'abc',01);

INSERT INTO Warden VALUES(421,'Cic',01);

INSERT INTO Warden VALUES(422,'Gih',02);
INSERT into Student VALUES(101,'AA','F',01);

INSERT into Student VALUES(102,'BB','F',01);

INSERT into Student VALUES(103,'CC','F',01);

INSERT into Student VALUES(104,'DD','F',01);

INSERT into Student VALUES(105,'EE','F',01);

INSERT into Student VALUES(106,'FF','M',02);

INSERT into Student VALUES(107,'GG','M',02);

INSERT into Student VALUES(108,'HH','M',02);

INSERT into Student VALUES(109,'II','M',02);

INSERT into Student VALUES(110,'JJ','M',02);
```

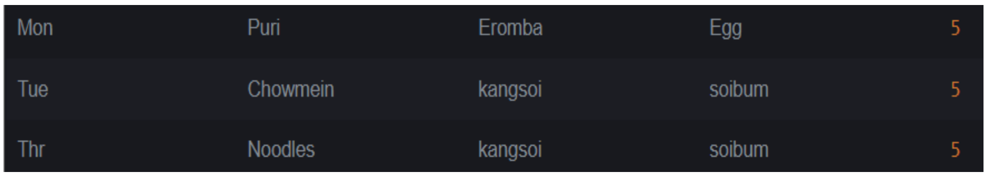
OUTPUT:



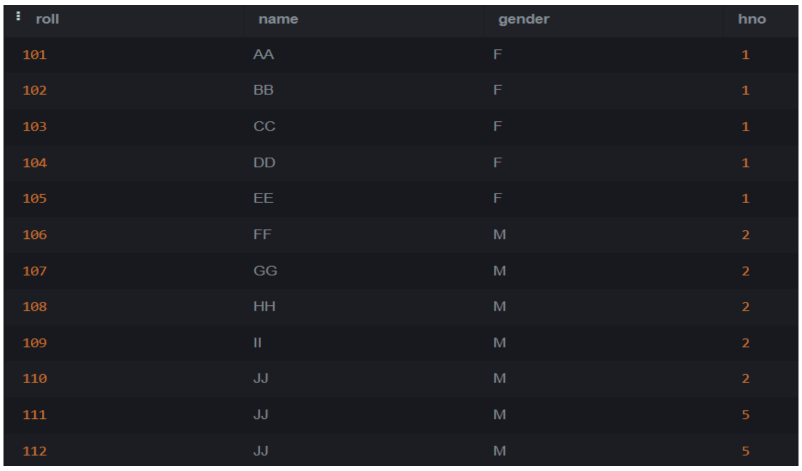
h_no	hname	h_type
1	x	G
2	y	B



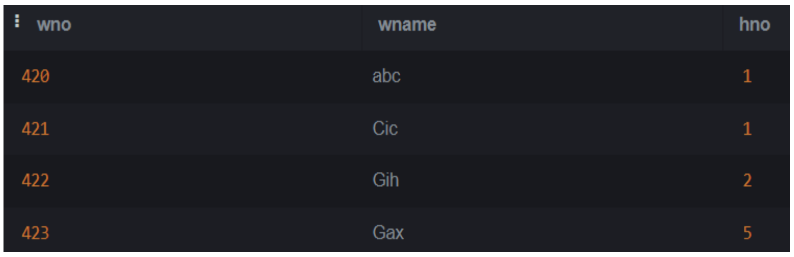
Day	breakfast	lunch	dinner	hno
Mon	Puri	Eromba	Egg	1
Tue	Chowmein	kangsoi	soibum	1
wed	Puri	veges	chicken	1
Thr	Plao	Eromba	Egg	1
Fri	Puffs	Dal	Fish	1
Sat	Chappati	Pakora	Chagem	1
Sun	Parantha	Ooti	Chicken	1
Mon	Puri	Eromba	Egg	2
Tue	Chowmein	kangsoi	soibum	2
wed	Puri	veges	chicken	2
Thr	Plao	Eromba	Egg	2
Fri	Puffs	Dal	Fish	2
Sat	Chappati	Pakora	Chagem	2
Sun	Parantha	Ooti	Chicken	2



Mon	Puri	Eromba	Egg	5
Tue	Chowmein	kangsoi	soibum	5
Thr	Noodles	kangsoi	soibum	5



roll	name	gender	hno
101	AA	F	1
102	BB	F	1
103	CC	F	1
104	DD	F	1
105	EE	F	1
106	FF	M	2
107	GG	M	2
108	HH	M	2
109	II	M	2
110	JJ	M	2
111	JJ	M	5
112	JJ	M	5



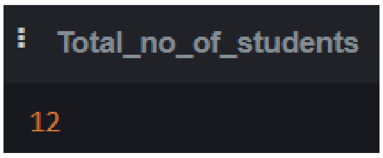
wno	wname	hno
420	abc	1
421	Cic	1
422	Gih	2
423	Gax	5

QUERY 1: To display the total number of girls and boys in the college.

CODE:

SELECT COUNT(*) as Total_no_of_students FROM Student;

OUTPUT:



Total_no_of_students
12

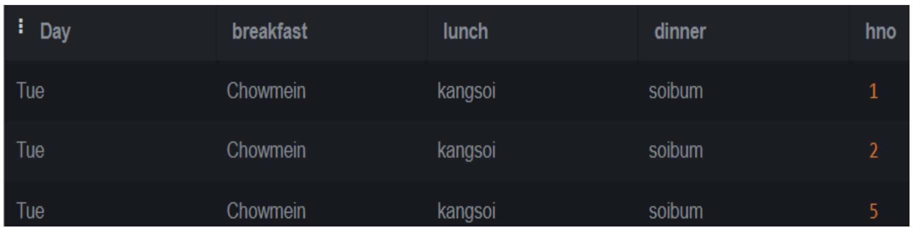
QUERY 2: To display the menu in the hostel ‘x’ on Tuesday.

CODE:

SELECT * FROM Mess WHERE hno=(SELECT hno FROM Hostel

WHERE hname='x') AND day='Tue';

OUTPUT:



Day	breakfast	lunch	dinner	hno
Tue	Chowmein	kangsoi	soibum	1
Tue	Chowmein	kangsoi	soibum	2
Tue	Chowmein	kangsoi	soibum	5

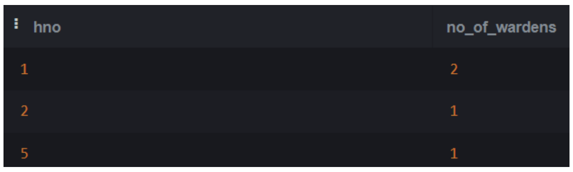
QUERY 3: To retrieve the number of wardens for each hostel.

CODE:

SELECT hno, COUNT (*) as no_of_wardens FROM Warden

GROUP by hno;

OUTPUT:



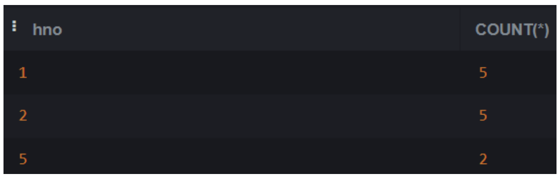
hno	no_of_wardens
1	2
2	1
5	1

QUERY 4: To retrieve the total number of students residing in a particular hostel (say hostel 'xyz').

CODE:

SELECT hno, COUNT (*) FROM Student GROUP by hno;

OUTPUT:



hno	COUNT(*)
1	5
2	5
5	2

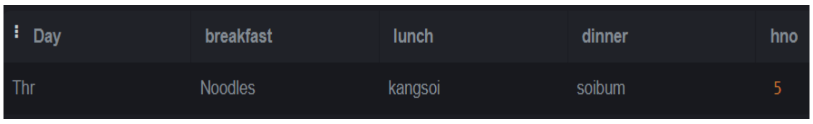
QUERY 5: To change breakfast item given on Thursday of hostel number 5 to 'Noodles'.

CODE:

UPDATE Mess SEt breakfast='Noodles' WHERE day='Thr' AND hno=05;

SELECT * FROM Mess WHERE day='Thr' AND hno=05;

OUTPUT:



Day	breakfast	lunch	dinner	hno
Thr	Noodles	kangsoi	soibum	5

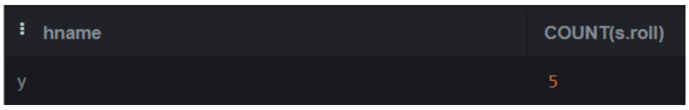
QUERY 6: To display the name of all the hostels that is having more number of students than hostel 'x'.

CODE:

SELECT h.hname,COUNT(s.roll) FROM Student s JOIN Hostel h

ON s.hno=h.h_no GROUP by h.hname HAVING COUNT(s.roll) >(SELECT COUNT(roll) from Student WHERE hname='x');

OUTPUT:



hname	COUNT(s.roll)
y	5

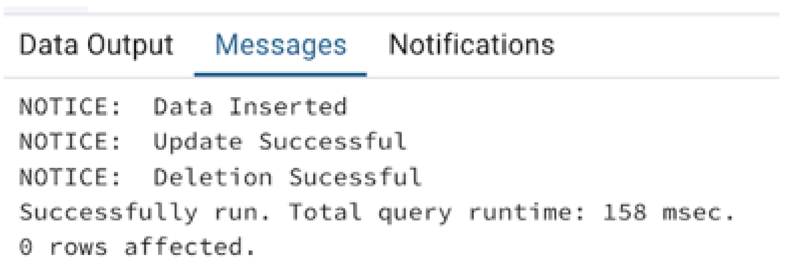
EXPERIMENT 8

AIM: To write a PL/SQL program to create a table and show insertion deletion and updation.

CODE:

```
CREATE TABLE CUSTOMER ( CUSID INT NOT NULL, CUSNAME VARCHAR(20) NOT NULL, AGE INT NOT NULL, SALARY DECIMAL(18,2), ADDRESS CHAR(25), PRIMARY KEY(CUSID));

DECLARE

BEGIN

INSERT INTO CUSTOMER (CUSID, CUSNAME, AGE, SALARY, ADDRESS)

VALUES(2101, ‘ANAND’, 18, 25000.00, ‘ARARAI’);

Dbms_output.put_line(‘Data Inserted’);

END;

/

DECLARE

BEGIN

UPDATE CUSTOMER SET AGE = 28 WHERE CUSID = 2101;

Dbms_output.put_line(‘Update Successful’);

END;

/

DECLARE

BEGIN

DELETE CUSTOMER WHERE CUSID = 2101;

Dbms_output.put_line(‘Deletion Successful’);

END;

/
```

OUTPUT:



```
NOTICE:  Data Inserted
NOTICE:  Update Successful
NOTICE:  Deletion Sucessful
Successfully run. Total query runtime: 158 msec.
0 rows affected.
```

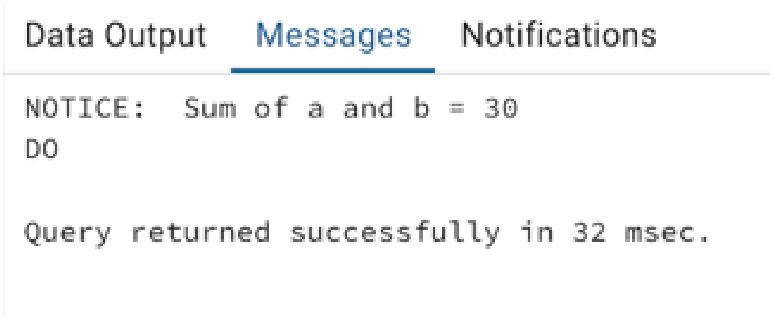
EXPERIMENT 9

AIM: To write a PL/SQL program for addition of two numbers.

CODE:

```
Declare
a int;
b int;
c int;
begin
a:=10;
b:=20;
c:=a+b;
dbms_output.put_line(‘sum of a and b=’ ||c);
end;
/
```

OUTPUT:



```
NOTICE:  Sum of a and b = 30
DO

Query returned successfully in 32 msec.
```