DAYANANDA SAGAR COLLEGE OF ENGINEERING

An Autonomous Institution
Affiliated to VTU
Approved by AICTE & UGC
Accredited by NBA
Accredited by NAAC with 'A' grade

DEPARTMENT OF COMPUTER SCIENCE AND DESIGN

DATABASE MANAGEMENT SYSTEMS LABORATORY MANUAL

DEPARTMENT OF COMPUTER SCIENCE AND DESIGN

DAYANANDA SAGAR COLLEGE OF ENGINEERING

(AN AUTONOMOUS INSTITUTE AFFILIATED TO VTU, BELAGAVI)

Shavige Malleshwara Hills, Kumaraswamy Layout, Bangalore-560078

Course Name and Course Code: DBMS Laboratory (21CSL46)

Year and Semester : II year, IV semester

Name of the Faculty : Poornima D and Harshitha H R

VISION AND MISSION OF THE INSTITUTION

INSTITUTION VISION

To impact quality technical education with a focus on Research and Innovation emphasizing on Development of Sustainable and Inclusive Technology for the benefit of society.

INSTITUTION MISSION

- To provide an environment that enhances creativity and Innovation in pursuit of Excellence.
- To nurture teamwork in order to transform individuals as responsible leaders and entrepreneurs.
- To train the students to the changing technical scenario and make them to understand the importance of Sustainable and Inclusive technologies.

VISION AND MISSION OF CSE DEPARTMENT

DEPARTMENT VISION

Computer Science and Design Engineering Department shall architect the most innovative programs to deliver competitive and sustainable solutions using cutting edge technologies and implementations, for betterment of society and research.

DEPARTMENT MISSION

- To adopt the latest industry trends in teaching learning process in order to make students competitive in the job market
- To encourage forums that enable students to develop skills in multidisciplinary areas and emerging technologies
- To encourage research and innovation among students by creating an environment of learning through active participation and presentations
- To collaborate with industry and professional bodies for the students to gauge the market trends and train accordingly.
- To create an environment which fosters ethics and human values to make students responsible citizens.

DAYANANDA SAGAR COLLEGE OF ENGINEERING
(An Autonomous Institution Affiliated to VTU, Belagavi)
DEPARTMENT OF COMPUTER SCIENCE AND DESIGN

COURSE OUTCOMES (CO)

COs	DESCRIPTION	REVISED BLOOM'S TAXONOMY (RBT)LEVEL
CO1	Identify the Entities, Attributes and different Constraints for the given Database requirements commonly used in day to day life in the different fields like Education, Banking, Business and all other fields where there is a need for data security, data updating, management and its maintenance.	L1, L2
CO2.	Design a Database schema and establish the relationships using foreign keys.	L2
CO3.	Create, Update and query on the database.	L3
CO4.	Demonstrate the working of different concepts of DBMS.	L4
CO5.	Implement, analyse and evaluate the mini project developed for an application in different sectors like Healthcare, Travel, Food etc.	L5, L6

S. No	Name of the Experiment	Course Outcome
1	**PROGRAM 1: INSURANCE DATABASE** Consider the Insurance database given below. Table names and Data types are specified. **PERSON** (driver – id #: String, name: String, address: String) **CAR** (Regno: String, model: String, year: int) **ACCIDENT** (report-number: int, date: date, location: String) **OWNS** (driver-id #: String, Regno: String) **PARTICIPATED** (driver-id: String, Regno: String, report-number: int, damage-amount: int)	CO1, CO2 and CO3

	WRITE THE SQL QUERIES TO: 1) Create the above tables by properly specifying the primary keys and the foreign keys. 2) Enter at least five tuples for each relation. 3) Demonstrate how to add a new accident to the database. 4) Find the total number of people who owned cars that involved in accidents in 2008. 5) Find the number of accidents in which cars belonging to a specific model were involved.	
2	**PROGRAM 2: COMPANY DATABASE** Consider the schema for Company Database: **EMPLOYEE**(SSN, Name, Address, Sex, Salary, SuperSSN, DNo) **DEPARTMENT**(DNo, DName, MgrSSN, MgrStartDate) **DLOCATION**(DNo,DLoc) **PROJECT**(PNo, PName, PLocation, DNo) **WORKS_ON**(SSN, PNo, Hours) WRITE THE SQL QUERIES TO: 1) Make a list of all project numbers for projects that involve an employee whose last name is 'Scott', either as a worker or as a manager of the department that controls the project. 2) Show the resulting salaries if every employee working on the 'IoT' project is given a 10 percent raise. 3) Find the sum of the salaries of all employees of the 'Accounts' department, as well as the maximum salary, the minimum salary, and the average salary in this department 4) Retrieve the name of each employee who works on all the projects controlled by department number 5 (use NOT EXISTS operator). 5) For each department that has more than five employees, retrieve the department number and the number of its employees who are making more than Rs.6,00,000.	CO1, CO2 and CO3
3	**PROGRAM 3: BANKING ENTERPRISE DATABASE** Consider the following database for a banking enterprise. **BRANCH** (branch-name: String, branch-city: String, assets: real) **ACCOUNTS** (accno: int, branch-name: String, balance: real)	CO1, CO2 and CO3

	DEPOSITOR (customer-name: String, customer-street: String, customer-city: String) **LOAN** (loan-number: int, branch-name: String, amount: real) **BORROWER** (customer-name: String, loan-number: int) WRITE THE SQL QUERIES TO: 1) Create the above tables by properly specifying the primary keys and the foreign keys. 2) Enter at least five tuples for each relation. 3) Find all the customers who have at least two accounts at the *Main* branch. 4) Find all the customers who have an account at *all* the branches located in a specific city. 5) Demonstrate how you delete all account tuples at every branch located in a specific city. 6) Generate suitable reports. 7) Create suitable front end for querying and displaying the results.	
4	**PROGRAM 4: LIBRARY DATABASE** Consider the following schema for a Library Database: **BOOK**(Book_id, Title, Publisher_Name, Pub_Year) **BOOK_AUTHORS**(Book_id, Author_Name) **PUBLISHER**(Name, Address, Phone) **BOOK_COPIES**(Book_id, Programme_id, No-of_Copies) **BOOK_LENDING**(Book_id, Programme_id, Card_No, Date_Out, Due_Date) **LIBRARY_PROGRAMME**(Programme_id, Programme_Name, Address) WRITE THE SQL QUERIES TO: 1) Retrieve details of all books in the library – id, title, name of publisher, authors, number of copies in each Programme, etc. 2) Get the particulars of borrowers who have borrowed more than 3 books, but from Jan 2017 to Jun 2017. 3) Delete a book in BOOK table. Update the contents of other tables to reflect this data manipulation operation. 4) Partition the BOOK table based on year of publication. Demonstrate its working with a simple query. 5) Create a view of all books and its number of copies that are currently available in the Library.	CO1, CO2 and CO3
5	**PROGRAM 5: ORDER PROCESSING DATABASE** Consider the following relations for an Order Processing database	CO1, CO2 and

	application in a company. **CUSTOMER** (CUST #: int, cname: String, city: String) **ORDER** (order #: int, odate: date, cust #: int, ord-Amt: int) **ITEM** (item #: int, unit-price: int) **ORDER-ITEM** (order #: int, item #: int, qty: int) **WAREHOUSE** (warehouse #: int, city: String) **SHIPMENT** (order #: int, warehouse #: int, ship-date: date) WRITE THE SQL QUERIES TO: 1. Create the above tables by properly specifying the primary keys and the foreign keys and the foreign keys. 2. Enter at least five tuples for each relation. 3. Produce a listing: CUSTNAME, #oforders, AVG_ORDER_AMT, where the middle column is the total numbers of orders by the customer and the last column is the average order amount for that customer. 4. List the order# for orders that were shipped from all warehouses that the company has in a specific city. 5. Demonstrate how you delete item# 10 from the ITEM table and make that field *null* in the ORDER_ITEM table.	CO3

PROGRAM 1: INSURANCE DATABASE

Consider the Insurance database given below. Table names and Data types are specified.

PERSON (driver – id #: String, name: String, address: String)
CAR (Regno: String, model: String, year: int)
ACCIDENT (report-number: int, date: date, location: String)
OWNS (driver-id #: String, Regno: String)
PARTICIPATED (driver-id: String, Regno: String, report-number: int, damage-amount: int)

WRITE THE SQL QUERIES TO:
1. Create the above tables by properly specifying the primary keys and the foreign keys.
2. Enter at least five tuples for each relation.
3. Demonstrate
   a) How to add a new accident to the database.
   b) Update the damage amount for the car with a specific Regno in the accident with report number 12 to 25000.
4. Find the total number of people who owned cars that involved in accidents in 2008.

5. Find the number of accidents in which cars belonging to a specific model were involved.

INTRODUCTION: This program is based on the car insurance. We maintain records of all registered cars and their owners. Details of accidents and damage amount are recorded.

1. Create the above tables by properly specifying the primary keys and the foreign keys.

```
create table person
(
Driverid varchar(10),
name varchar(20),
address varchar(30),
primary key(driverid)
);
```

Table created.

SQL> **desc person**

```
Name                      Null?                Type
------------------------- -------- ------------------------------
DRIVERID                  NOT NULL             VARCHAR2(10)
NAME                      VARCHAR2(20)
ADDRESS                   VARCHAR2
```

SQL> **create table car**

```
(
regno varchar(10),
model varchar(10),
year int,
primary key(regno)
);
```

Table created.

SQL> **desc car**

```
Name                      Null?                          Type
------------------------- -------- ---------------------------------
REGNO                     NOT NULL                       VARCHAR2(10)
MODEL                     VARCHAR2(10)
YEAR                      NUMBER(38)
```

SQL> **create table accident**

```
(
reportno int,
```

```
        date date,
        location varchar(20),
        primary key(reportno)
        );

        Table created.
SQL> desc accident

Name                      Null?                Type
------------------------- -------- ---------------------------
REPORTNO                   NOT NULL             NUMBER(38)
DATE                      DATE
LOCATION                                       VARCHAR2(20)

SQL> create table owns
        (
        driverid varchar(10),
        regno varchar(10),
        primary key(driverid,regno),
        foreign key(driverid) references person(driverid),
        foreign key(regno) references car(regno)
        );

Table created.

SQL> desc owns

Name                       Null?                Type
------------------------------ -------- ------------------------------
DRIVERID                  NOT NULL             VARCHAR2(10)
REGNO                     NOT NULL             VARCHAR2(10)

SQL>create table participated
        (
        Driverid varchar(10),
        regno varchar(10),
        reportno int,
        damageamt,
        int, primary key(driver-id,regno,reportno),
        foreign key(driver-id) references person(driverid),
        foreign key(regno) references car(regno),
        foreign key(report-no) references accident(reportno)
        );
Table created.

SQL> desc participated

Name                       Null?                Type
------------------------------ ---------------- ----------------------------
DRIVERID                  NOT NULL             VARCHAR2(10)
REGNO                     NOT NULL             VARCHAR2(10)
REPORTNO                  NOT NULL              NUMBER(38)
```

DAMAGEAMT NUMBER(38)

QUERY 2: Enter at least five tuples for each relation

SQL> insert into person values('&driverid','&name','&address');

Enter value for driverid: A01
Enter value for name: Richard
Enter value for address: Srinivas Nagar
old 1: insert into person values('&driverid','&name','&address')
new 1: insert into person values('A01','Richard','Srinivas Nagar')

1 row created.
SQL> /
Enter value for driverid: A02
Enter value for name: Pradeep
Enter value for address: Rajajinagar
old 1: insert into person values('&driverid','&name','&address')
new 1: insert into person values('A02','Pradeep','Rajajinagar')
1 row created.
SQL> **commit;**
Commit complete.

SQL> select * from person;

DRIVERID	NAME	ADDRESS
A01	Richard	Srinivas Nagar
A02	Pradeep	Rajajinagar
A03	Smith	Ashoknagar
A04	Venu	N.R.Colony
A05	John	Hanumanth Nagar

SQL> insert into car values('&regno','&model', &year);

Enter value for regno: KA052250
Enter value for model: Indica
Enter value for year: 1990
old 1: insert into car values('&regno','&model', &year)
new 1: insert into car values('KA052250','Indica', 1990)
1 row created.

SQL> /
Enter value for regno: KA031181
Enter value for model: Lancer
Enter value for year: 1957
old 1: insert into car values('&regno','&model',&year)
new 1: insert into car values('KA031181','Lancer', 1957)
1 row created.

SQL> **commit;**
Commit complete.

SQL> select * from car;

REGNO	MODEL	YEAR
KA052250	Indica	1990
KA031181	Lancer	1957
KA095477	Toyota	1998
KA053408	Honda	2008
KA041702	Audi	2005

SQL> insert into accident values(&reportno,'&adate','&location');

Enter value for reportno: 11
Enter value for adate: 01-JAN-03
Enter value for location: Mysore Road
old 1: insert into accident values(&reportno,'&adate','&location')
new 1: insert into accident values(111,'01-JAN-03','Mysore Road')
1 row created.

SQL> **commit;**
Commit complete.
SQL> select * from accident;

REPORTNO	DATE	LOCATION
11	01-JAN-03	Mysore Road
12	02-FEB-04	Southend Circle
13	21-JAN-03	Bulltemple Road
14	17-FEB-08	Mysore Road
15	04-MAR-05	Kanakpura Road

SQL> insert into owns values ('&driverid','&regno');

Enter value for driverid: A01
Enter value for regno: KA052250
old 1: insert into owns values('&driverid','&regno')
new 1: insert into owns values('A01','KA052250')
1 row created.
SQL> **commit;**
Commit complete.

SQL> select * from owns;

DRIVERID	REGNO
A01	KA052250
A02	KA053408
A04	KA031181
A03	KA095477
A05	KA041702

SQL> insert into participated values ('&driverid','&regno',&reportno, &damt);

Enter value for driverid: A01

Enter value for regno: KA052250
Enter value for reportno: 11
Enter value for damt: 10000
old 1: insert into participated values ('&driverid','&regno',&reportno,&damt)
new 1: insert into participated values('A01','KA052250',11,10000)
1 row created.

Enter value for driverid: A02
Enter value for regno: KA053408
Enter value for reportno: 12
Enter value for damt: 50000
old 1: insert into participated values ('&driverid','&regno', &reportno,&damt)
new 1: insert into participated values('A02','KA053408',12,50000)
1 row created.
SQL> **commit;**
Commit complete.

SQL> select * from participated;

DRIVERID	REGNO	REPORTNO	DAMAGEAMT
A01	KA052250	11	10000
A02	KA053408	12	50000
A03	KA095477	13	25000
A04	KA031181	14	3000
A05	KA041702	15	5000

QUERY 3:

a) Update the damage amount for the car with a specific Regno in the accident with report number 12 to 25000.

SQL> update participated
set damageamt=25000
where regno='KA053408' and reportno=12;
1 row updated.
SQL> **commit;**
Commit complete.

SQL> **select * from participated;**

DRIVERID	REGNO	REPORTNO	DAMAGEAMT
A01	KA052250	11	10000
A02	KA053408	12	25000
A03	KA095477	13	25000
A04	KA031181	14	3000
A05	KA041702	15	5000

b) Add a new accident to the database.

SQL> **insert into accident values(16,'15-MAR-08','Domlur');**

1 row created.

SQL> **select * from accident;**

REPORTNO	DATE	LOCATION
11	01-JAN-03	Mysore Road
12	02-FEB-04	Southend Circle
13	21-JAN-03	Bulltemple Road
14	17-FEB-08	Mysore Road
15	04-MAR-05	Kanakpura Road
16	15-MAR-08	Domlur

6 rows selected.

QUERY 4: Find the total number of people who owned cars that were involved in accidents in 2008.

```
SQL> select count(distinct driver-id) CNT
        from participated a, accident b
        where a.reportno=b.reportno and b.adate like '%08';
        CNT
        ----------
          1
```

QUERY 5: Find the number of accidents in which cars belonging to a specific model were involved.

```
SQL> select count(reportno) CNT
         from car c,
        participated p
        where c.regno=p.regno and model='Lancer';
        CNT
        ----------
           1
```

PROGRAM 2: COMPANY DATABASE

Consider the schema for Company Database:

EMPLOYEE(SSN, Name, Address, Gender, Salary, SuperSSN, DNo)
DEPARTMENT(DNo, DName, MgrSSN, MgrStartDate)
DLOCATION(DNo,DLoc)
PROJECT(PNo, PName, PLocation, DNo)
WORKS_ON(SSN, PNo, Hours)

WRITE THE SQL QUERIES TO:

1. Make a list of all project numbers for projects that involve an employee whose last name is ‘Scott’, either as a worker or as a manager of the department that controls the project.
2. Show the resulting salaries if every employee working on the ‘IoT’ project is given a 10 percent raise.
3. Find the sum of the salaries of all employees of the ‘Accounts’ department, as well as the maximum salary, the minimum salary, and the average salary in this department

4. Retrieve the name of each employee who works on all the projects controlled by department number 5 (use NOT EXISTS operator).
5. For each department that has more than five employees, retrieve the department number and the number of its employees who are making more than Rs.6,00,000.

Table Creation

```
CREATE TABLE DEPARTMENT
(
DNO VARCHAR2 (20) PRIMARY KEY,
DNAME VARCHAR2 (20),
MGRSTARTDATE DATE
);

CREATE TABLE EMPLOYEE
(
SSN VARCHAR2 (20) PRIMARY KEY,
FNAME VARCHAR2 (20),
LNAME VARCHAR2 (20),
ADDRESS VARCHAR2 (20),
GENDER CHAR (1),
SALARY INTEGER,
SUPERSSN REFERENCES EMPLOYEE (SSN),
DNO REFERENCES DEPARTMENT (DNO)
);
```

NOTE: Once DEPARTMENT and EMPLOYEE tables are created we must alter department table to add foreign constraint MGRSSN using sql command

```
ALTER TABLE DEPARTMENT
ADD MGRSSN REFERENCES EMPLOYEE (SSN);

CREATE TABLE DLOCATION
(
DLOC VARCHAR2 (20),
DNO REFERENCES DEPARTMENT (DNO),
PRIMARY KEY (DNO, DLOC)
);

CREATE TABLE PROJECT
(
PNO INTEGER PRIMARY KEY,
PNAME VARCHAR2 (20),
PLOCATION VARCHAR2 (20),
DNO REFERENCES DEPARTMENT (DNO)
);
```

```
CREATE TABLE WORKS_ON
(
HOURS NUMBER (2),
SSN REFERENCES EMPLOYEE (SSN),
PNO REFERENCES PROJECT(PNO),
PRIMARY KEY (SSN, PNO)
);
```

Table Descriptions

DESC EMPLOYEE;

SQL> DESC EMPLOYEE

Name

SSN
NAME
ADDRESS
GENDER
SAL
SUPERSSN
DNO

DESC DEPARTMENT

```
SQL> DESC DEPARTMENT;
 Name
 -----------------------------------------------------
 DNO
 DNAME
 MGRSTARTDATE
 MGRSSN
```

DESC DLOCATION

```
SQL> DESC DLOCATION;
 Name
 --------------------------
 DLOC
 DNO
```

DESC PROJECT

```
SQL> DESC PROJECT;
 Name
 --------------------------
 PNO
 PNAME
 PLOCATION
 DNO
```

DESC WORKS_ON

```
SQL> DESC WORKS_ON;
 Name
 -----------------------------------------------
 HOURS
 SSN
 PNO
```

Insertion of values to tables:

INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, GENDER, SALARY)
VALUES (‘RNSECE01’,’JOHN’,’SCOTT’,’BANGALORE’,’M’, 450000);

INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, GENDER, SALARY)
VALUES (‘RNSCSE01’,’JAMES’,’SMITH’,’BANGALORE’,’M’, 500000);

INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, GENDER, SALARY)
VALUES (‘RNSCSE02’,’HEARN’,’BAKER’,’BANGALORE’,’M’, 700000);

INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, GENDER, SALARY)
VALUES (‘RNSCSE03’,’EDWARD’,’SCOTT’,’MYSORE’,’M’, 500000);

INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, GENDER, SALARY)
VALUES (‘RNSCSE04’,’PAVAN’,’HEGDE’,’MANGALORE’,’M’, 650000);

INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, GENDER, SALARY)
VALUES (‘RNSCSE05’,’GIRISH’,’MALYA’,’MYSORE’,’M’, 450000);

INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, GENDER, SALARY)
VALUES (‘RNSCSE06’,’NEHA’,’SN’,’BANGALORE’,’F’, 800000);

INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, GENDER, SALARY)
VALUES (‘RNSACC01’,’AHANA’,’K’,’MANGALORE’,’F’, 350000);

INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, GENDER, SALARY)
VALUES (‘RNSACC02’,’SANTHOSH’,’KUMAR’,’MANGALORE’,’M’, 300000);

INSERT INTO EMPLOYEE (SSN, FNAME, LNAME, ADDRESS, GENDER, SALARY)
VALUES (‘RNSISE01’,’VEENA’,’M’,’MYSORE’,’M’, 600000);

INSERT INTO DEPARTMENT VALUES (‘1’,’ACCOUNTS’,’01-JAN-01’,’RNSACC02’);
INSERT INTO DEPARTMENT VALUES (‘2’,’IT’,’01-AUG-16’,’SKIT01’);
INSERT INTO DEPARTMENT VALUES (‘3’,’ECE’,’01-JUN-08’,’RNSECE01’);
INSERT INTO DEPARTMENT VALUES (‘4’,’ISE’,’01-AUG-15’,’RNSISE01’);
INSERT INTO DEPARTMENT VALUES (‘5’,’CSE’,’01-JUN-02’,’RNSCSE05’);

Note: update entries of employee table to fill missing fields SUPERSSN and DNO

UPDATE EMPLOYEE SET
SUPERSSN=NULL, DNO=’3’

```
WHERE SSN='RNSECE01';

UPDATE EMPLOYEE SET
SUPERSSN='RNSCSE02', DNO='5'
WHERE SSN='RNSCSE01';

UPDATE EMPLOYEE SET
SUPERSSN='RNSCSE03', DNO='5'
WHERE SSN='RNSCSE02';

UPDATE EMPLOYEE SET
SUPERSSN='RNSCSE04', DNO='5'
WHERE SSN='RNSCSE03';

UPDATE EMPLOYEE SET
DNO='5', SUPERSSN='RNSCSE05'
WHERE SSN='RNSCSE04';

UPDATE EMPLOYEE SET
DNO='5', SUPERSSN='RNSCSE06'
WHERE SSN='RNSCSE05';

UPDATE EMPLOYEE SET
DNO='5', SUPERSSN=NULL
WHERE SSN='RNSCSE06';

UPDATE EMPLOYEE SET
DNO='1', SUPERSSN='RNSACC02'
WHERE SSN='RNSACC01';


INSERT INTO DLOCATION VALUES ('BANGALORE', '1');
INSERT INTO DLOCATION VALUES ('BANGALORE', '2');
INSERT INTO DLOCATION VALUES ('BANGALORE', '3');
INSERT INTO DLOCATION VALUES ('MANGALORE', '4');
INSERT INTO DLOCATION VALUES ('MANGALORE', '5');

INSERT INTO PROJECT VALUES (100,'IOT','BANGALORE','5');
INSERT INTO PROJECT VALUES (101,'CLOUD','BANGALORE','5');
INSERT INTO PROJECT VALUES (102,'BIGDATA','BANGALORE','5');
INSERT INTO PROJECT VALUES (103,'SENSORS','BANGALORE','3');
INSERT INTO PROJECT VALUES (104,'BANK MANAGEMENT','BANGALORE','1');
INSERT INTO PROJECT VALUES (105,'SALARY MANAGEMENT','BANGALORE','1');
INSERT INTO PROJECT VALUES (106,'OPENSTACK','BANGALORE','4'); INSERT INTO
PROJECT VALUES (107,'SMART CITY','BANGALORE','2');

INSERT INTO WORKS_ON VALUES (4, 'RNSCSE01', 100);
INSERT INTO WORKS_ON VALUES (6, 'RNSCSE01', 101);
INSERT INTO WORKS_ON VALUES (8, 'RNSCSE01', 102);
INSERT INTO WORKS_ON VALUES (10, 'RNSCSE02', 100);
INSERT INTO WORKS_ON VALUES (3, 'RNSCSE04', 100);
```

INSERT INTO WORKS_ON VALUES (4, ‘RNSCSE05’, 101);
INSERT INTO WORKS_ON VALUES (5, ‘RNSCSE06’, 102);
INSERT INTO WORKS_ON VALUES (6, ‘RNSCSE03’, 102);
INSERT INTO WORKS_ON VALUES (7, ‘RNSECE01’, 103);
INSERT INTO WORKS_ON VALUES (5, ‘RNSACC01’, 104);
INSERT INTO WORKS_ON VALUES (6, ‘RNSACC02’, 105);
INSERT INTO WORKS_ON VALUES (4, ‘RNSISE01’, 106);
INSERT INTO WORKS_ON VALUES (10, ‘SKIT01’, 107);

SELECT * FROM EMPLOYEE;

SSN	FNAME	LNAME	ADDRESS	S	SALARY	SUPERSSN	DNO
RNSECE01	JOHN	SCOTT	BANGALORE	M	450000		3
RNSCSE01	JAMES	SMITH	BANGALORE	M	500000	RNSCSE02	5
RNSCSE02	HEARN	BAKER	BANGALORE	M	700000	RNSCSE03	5
RNSCSE03	EDWARD	SCOTT	MYSORE	M	500000	RNSCSE04	5
RNSCSE04	PAVAN	HEGDE	MANGALORE	M	650000	RNSCSE05	5
RNSCSE05	GIRISH	MALYA	MYSORE	M	450000	RNSCSE06	5
RNSCSE06	NEHA	SN	BANGALORE	F	800000		5
RNSACC01	AHANA	K	MANGALORE	F	350000	RNSACC02	1
RNSACC02	SANTHOSH	KUMAR	MANGALORE	M	300000		1
RNSISE01	VEENA	M	MYSORE	M	600000		4
RNSIT01	NAGESH	HR	BANGALORE	M	500000		2

SELECT * FROM DEPARTMENT;

```
SQL> SELECT * FROM DEPARTMENT;
```

DNO	DNAME	MGRSTARTD	MGRSSN
1	ACCOUNTS	01-JAN-01	RNSACC02
2	IT	01-AUG-16	RNSIT01
3	ECE	01-JUN-08	RNSECE01
4	ISE	01-AUG-15	RNSISE01
5	CSE	01-JUN-02	RNSCSE05

SELECT * FROM WORKSON;

HOURS	SSN	PNO
4	RNSCSE01	100
6	RNSCSE01	101
8	RNSCSE01	102
10	RNSCSE02	100
3	RNSCSE04	100
4	RNSCSE05	101
5	RNSCSE06	102
6	RNSCSE03	102
7	RNSECE01	103
5	RNSACC01	104
6	RNSACC02	105
4	RNSISE01	106
10	RNSIT01	107

SELECT * FROM DLOCATION;

DLOC	DNO
BANGALORE	1
BANGALORE	2
BANGALORE	3
MANGALORE	4
MANGALORE	5

SELECT * FROM PROJECT;

PNO	PNAME	PLOCATION	DNO
100	IOT	BANGALORE	5
101	CLOUD	BANGALORE	5
102	BIGDATA	BANGALORE	5
103	SENSORS	BANGALORE	3
104	BANK MANAGEMENT	BANGALORE	1
105	SALARY MANAGEMENT	BANGALORE	1
106	OPENSTACK	BANGALORE	4
107	SMART CITY	BANGALORE	2

Queries:

1. Make a list of all project numbers for projects that involve an employee whose last
name is 'Scott', either as a worker or as a manager of the department that controls the project.

```
(SELECT DISTINCT P.PNO
FROM PROJECT P, DEPARTMENT D, EMPLOYEE
E WHERE E.DNO=D.DNO
AND D.MGRSSN=E.SSN
AND E.LNAME='SCOTT')
UNION
(SELECT DISTINCT P1.PNO
FROM PROJECT P1, WORKS_ON W, EMPLOYEE E1
WHERE P1.PNO=W.PNO
AND E1.SSN=W.SSN
AND E1.LNAME='SCOTT');
```

2. Show the resulting salaries if every employee working on the 'IoT' project is given a
10 percent raise.

```
SELECT E.FNAME, E.LNAME, 1.1*E.SALARY AS INCR_SAL
FROM EMPLOYEE E, WORKS_ON W, PROJECT P WHERE
E.SSN=W.SSN
AND W.PNO=P.PNO
AND P.PNAME='IOT';
```

FNAME	LNAME	INCR_SAL
JAMES	SMITH	550000
HEARN	BAKER	770000
PAVAN	HEGDE	715000

3. Find the sum of the salaries of all employees of the 'Accounts' department, as well as the maximum salary, the minimum salary, and the average salary in this department

```
SELECT SUM (E.SALARY), MAX (E.SALARY), MIN (E.SALARY),
AVG (E.SALARY)
```

```
FROM EMPLOYEE E, DEPARTMENT D
WHERE E.DNO=D.DNO
AND D.DNAME='ACCOUNTS';
```

SUM(E.SALARY)	MAX(E.SALARY)	MIN(E.SALARY)	AVG(E.SALARY)
650000	350000	300000	325000

4. Retrieve the name of each employee who works on all the projects Controlled by department number 5 (use NOT EXISTS operator).

```
SELECT E.FNAME, E.LNAME
FROM EMPLOYEE E
WHERE NOT EXISTS((SELECT PNO
FROM PROJECT
WHERE DNO='5')
MINUS (SELECT PNO
FROM WORKS_ON
WHERE E.SSN=SSN));
```

FNAME	LNAME
JAMES	SMITH

5. For each department that has more than five employees, retrieve the department number and the number of its employees who are making more than Rs. 6, 00,000.

```
SELECT D.DNO, COUNT (*)
FROM DEPARTMENT D, EMPLOYEE E
WHERE D.DNO=E.DNO AND E.SALARY>600000
AND D.DNO IN (SELECT E1.DNO FROM EMPLOYEE E1
GROUP BY E1.DNO HAVING COUNT (*)>5) GROUP BY D.DNO;
```

DNO	COUNT(*)
5	3

PROGRAM 3: BANKING ENTERPRISE DATABASE

Consider the following database for a banking enterprise.

BRANCH (branch-name: String, branch-city: String, assets: real)
ACCOUNTS (accno: int, branch-name: String, balance: real)
DEPOSITOR (customer-name: String, customer-street: String, customer-city: String)
LOAN (loan-number: int, branch-name: String, amount: real)
BORROWER (customer-name: String, loan-number: int)

WRITE THE SQL QUERIES TO:

1. Create the above tables by properly specifying the primary keys and the foreign keys.
2. Enter at least five tuples for each relation.
3. Find all the customers who have at least two accounts at the *Main* branch.
4. Find all the customers who have an account at *all* the branches located in a specific city.
5. Demonstrate how you delete all account tuples at every branch located in a specific city.
6. Generate suitable reports.
7. Create suitable front end for querying and displaying the results.

INTRODUCTION: This database is developed for supporting banking facilities. Details of the branch along with the accounts and loans handled by them are recorded. Also details of the borrowers and depositors of the corresponding branches are maintained.

QUERY 1: Create the above tables by properly specifying the primary keys and the foreign keys.

```
SQL> create table branch
     (
      branch_name varchar(20)
      primary key,
      branch_city varchar(10),assets real
      );
```

Table created.

```
SQL> desc branch;
```

Name	Null?	Type
BRANCH_NAME	NOT NULL	VARCHAR2(20)
BRANCH_CITY		VARCHAR2(10)
ASSETS		FLOAT(63)

```
SQL> create table account
     (
     accno int primary key,
     branch_name varchar(20),
     balance real,foreign key(branch_name) references branch(branch_name)
     );
Table created.
```

```
SQL> desc account;
```

Name	Null?	Type
ACCNO	NOT NULL	NUMBER(38)
BRANCH_NAME		VARCHAR2(20)
BALANCE		FLOAT(63)

```
SQL> create table customer(
customer_name varchar(20) primary key,
```

customer_street varchar(20),
cust_city varchar(20)
);
Table created.

SQL> desc customer;

Name	Null?	Type
CUSTOMER_NAME	NOT NULL	VARCHAR2(20)
CUSTOMER_STREET		VARCHAR2(20)
CUST_CITY		VARCHAR2(20)

SQL> create table depositor
(
customer_name varchar(20),
accno int,
foreign key(customer_name) references customer(customer_name),
foreign key(accno) references account(accno)
);

Table created.

SQL> desc depositor;

Name	Null?	Type
CUSTOMER_NAME		VARCHAR2(20)
ACCNO		NUMBER(38)

SQL> create table loan
(
loan_no int primary key,
branch_name varchar(20),
amount real,
foreign key(branch_name) references branch(branch_name)
);

Table created.

SQL> desc loan;

Name	Null?	Type
LOAN_NO	NOT NULL	NUMBER(38)
BRANCH_NAME		VARCHAR2(20)
AMOUNT		FLOAT(63)

SQL> create table borrower
(
customer_name varchar(20),

loan_no int,
foreign key(customer_name) references customer(customer_name),
foreign key(loan_no) references loan(loan_no)
);

Table created.

SQL> desc borrower;

Name	Null?	Type
CUSTOMER_NAME		VARCHAR2(20)
LOAN_NO		NUMBER(38)

QUERY 2: Enter at least five tuples for each relation

SQL> insert into branch values('&bname','&bcity',&assets);
Enter value for bname: SBI PD NAGAR
Enter value for bcity: BANGALORE
Enter value for assets: 200000
old 1: insert into branch values('&bname','&bcity',&assets)
new 1: insert into branch values('SBI PD NAGAR','BANGALORE',200000)
1 row created.

SQL> commit;
Commit complete.

SQL> select * from branch;

BRANCH_NAME	BRANCH_CIT	ASSETS
SBI PD NAGAR	BANGALORE	200000
SBI RAJAJI NAGAR	BANGALORE	500000
SBI JAYANAGAR	BANGALORE	660000
SBI VIJAY NAGAR	BANGALORE	870000
SBI HOSAKEREHALLI	BANGALORE	550000

SQL> insert into account values(&accno,'&bname',&balance);
Enter value for accno: 1234602
Enter value for bname: SBI HOSAKEREHALLI
Enter value for balance: 5000
old 1: insert into account values(&accno,'&bname',&balance)
new 1: insert into account values(1234602,'SBI HOSAKEREHALLI',5000)
1 row created.
SQL> /
Enter value for accno: 1234603
Enter value for bname: SBI VIJAY NAGAR
Enter value for balance: 5000
old 1: insert into account values(&accno,'&bname',&balance)
new 1: insert into account values(1234603,'SBI VIJAY NAGAR',5000)
1 row created.
SQL> commit;
Commit complete.

SQL> select * from account;

ACCNO	BRANCH_NAME	BALANCE
1234602	SBI HOSAKEREHALLI	5000
1234603	SBI VIJAY NAGAR	5000
1234604	SBI JAYANAGAR	5000
1234605	SBI RAJAJI NAGAR	10000
1234503	SBI VIJAY NAGAR	40000
1234504	SBI PD NAGAR	4000

6 rows selected.

SQL> insert into customer values('&cname','&cstreet','&ccity');
Enter value for cname: KEZAR
Enter value for cstreet: M G ROAD
Enter value for ccity: BANGALORE
old 1: insert into customer values('&cname','&cstreet','&ccity')
new 1: insert into customer values('KEZAR','M G ROAD','BANGALORE')
1 row created.

SQL> commit;
Commit complete

SQL> select * from customer;

CUSTOMER_NAME	CUSTOMER_STREET	CUST_CITY
KEZAR	M G ROAD	BANGALORE
LAL KRISHNA	ST MKS ROAD	BANGALORE
RAHUL	AUGSTEN ROAD	BANGALORE
LALLU	V S ROAD	BANGALORE
FAIZAL	RESEDENCY ROAD	BANGALORE
RAJEEV	DICKNSN ROAD	BANGALORE

6 rows selected.

SQL> insert into depositor values('&cname',&accno);
Enter value for cname: KEZAR
Enter value for accno: 1234602
old 1: insert into depositor values('&cname',&accno)
new 1: insert into depositor values('KEZAR',1234602)

1 row created.
SQL> commit;
Commit complete.
SQL> select * from depositor;

CUSTOMER_NAME	ACCNO
KEZAR	1234602
LAL KRISHNA	1234603
RAHUL	1234604
LALLU	1234605
LAL KRISHNA	234503

RAJEEV 1234504

6 rows selected.

SQL> insert into loan values(&loanno,'&bname',&amount);

Enter value for loanno: 10011
Enter value for bname: SBI JAYANAGAR
Enter value for amount: 10000
old 1: insert into loan values(&loanno,'&bname',&amount)
new 1: insert into loan values(10011,'SBI JAYANAGAR',10000)
1 row created.

SQL> /

Enter value for loanno: 10012
Enter value for bname: SBI VIJAY NAGAR
Enter value for amount: 5000
old 1: insert into loan values(&loanno,'&bname',&amount)
new 1: insert into loan values(10012,'SBI VIJAY NAGAR',5000)
1 row created.

SQL> commit;

Commit complete.

SQL> select * from loan;

LOAN_NO	BRANCH_NAME	AMOUNT
10011	SBI JAYANAGAR	10000
10012	SBI VIJAY NAGAR	5000
10013	SBI HOSAKEREHALLI	20000
10014	SBI PD NAGAR	15000
10015	SBI RAJAJI NAGAR	25000

SQL> insert into borrower values('&cname',&loanno);

Enter value for cname: KEZAR
Enter value for loanno: 10011
old 1: insert into borrower values('&cname',&loanno)
new 1: insert into borrower values('KEZAR',10011)
1 row created.

SQL> /

Enter value for cname: LAL KRISHNA
Enter value for loanno: 10012

SQL> commit;

Commit complete.

SQL> select * from borrower;

CUSTOMER_NAME	LOAN_NO
KEZAR	10011
LAL KRISHNA	10012
RAHUL	10013
LALLU	10014
LAL KRISHNA	10015

QUERY 3: Find all the customers who have at least two accounts at the Main branch.

```
SQL> select customer_name from depositor where accno in
        (
        select accno from depositor where accno in
         (
         select accno from account where branch_name in
          (
         select branch_name from account
         where branch_name='SBI VIJAY NAGAR'
          group by branch_name
          having count(*) > 1)))
        group by customer_name
        having count(*) > 1;

CUSTOMER_NAME
--------------------
LAL KRISHNA
```

QUERY 4: Find all the customers who have an account at *all* the branches located in a specific city.

```
SQL> select customer_name,accno
from depositor
where accno in(
select accno from account where branch_name in(
select branch_name from branch where branch_city='BANGALORE'));

CUSTOMER_NAME                ACCNO
-------------------- ----------------------------------------------------------------
KEZAR                        1234602
LAL KRISHNA                  1234603
RAHUL                        1234604
LALLU                        1234605
LAL KRISHNA                  1234503
RAJEEV                       1234504

6 rows selected.
```

QUERY 5: Demonstrate how you delete all account tuples at every branch located in a specific city.

```
SQL> delete from account
where branch_name=(select branch_name
from branch where branch_city='&city');

Enter value for city: BANGALORE
old 2: where branch_city='&city'
new 6: where brach_city='BANGALORE'
1 row deleted.
```

PROGRAM 4: LIBRARY DATABASE

Consider the following schema for a Library Database:

BOOK(Book_id, Title, Publisher_Name, Pub_Year)
BOOK_AUTHORS(Book_id, Author_Name)
PUBLISHER(Name, Address, Phone)
BOOK_COPIES(Book_id, Programme_id, No-of_Copies)
BOOK_LENDING(Book_id, Programme_id, Card_No, Date_Out, Due_Date)
LIBRARY_BRANCH (*Branch_id, Branch_Name, Address*)

WRITE THE SQL QUERIES TO:

1. Retrieve details of all books in the library – id, title, name of publisher, authors, number of copies in each Programme, etc.
2. Get the particulars of borrowers who have borrowed more than 3 books, but from Jan 2017 to Jun 2017.
3. Delete a book in BOOK table. Update the contents of other tables to reflect this data manipulation operation.
4. Partition the BOOK table based on year of publication. Demonstrate its working with a simple query.
5. Create a view of all books and its number of copies that are currently available in the Library.

Table Creation

```
CREATE TABLE PUBLISHER
(
NAME VARCHAR2 (20) PRIMARY KEY,
PHONE INTEGER,
ADDRESS VARCHAR2 (20)
);

CREATE TABLE BOOK
(
BOOK_ID INTEGER PRIMARY KEY,
TITLE VARCHAR2 (20),
PUB_YEAR VARCHAR2 (20),
PUBLISHER_NAME REFERENCES PUBLISHER (NAME) ON DELETE
CASCADE
);


CREATE TABLE BOOK_AUTHORS
(
```

```
AUTHOR_NAME VARCHAR2 (20),
BOOK_ID REFERENCES BOOK (BOOK_ID) ON DELETE
CASCADE, PRIMARY KEY (BOOK_ID, AUTHOR_NAME)
);

CREATE TABLE LIBRARY_BRANCH
(
BRANCH_ID INTEGER PRIMARY KEY,
BRANCH_NAME VARCHAR2 (50),
ADDRESS VARCHAR2 (50)
);

CREATE TABLE BOOK_COPIES
(
NO_OF_COPIES INTEGER,
BOOK_ID REFERENCES BOOK (BOOK_ID) ON DELETE CASCADE,
BRANCH_ID REFERENCES LIBRARY_BRANCH (BRANCH_ID) ON DELETE
CASCADE,
PRIMARY KEY (BOOK_ID, BRANCH_ID)
);

CREATE TABLE CARD
(
CARD_NO INTEGER PRIMARY KEY);


CREATE TABLE BOOK_LENDING
(
DATE_OUT DATE,
DUE_DATE DATE,
BOOK_ID REFERENCES BOOK (BOOK_ID) ON DELETE CASCADE,
BRANCH_ID REFERENCES LIBRARY_BRANCH (BRANCH_ID) ON DELETE
CASCADE,
CARD_NO REFERENCES CARD (CARD_NO) ON DELETE CASCADE,
PRIMARY KEY (BOOK_ID, BRANCH_ID, CARD_NO)
);
```

Table Descriptions

DESC PUBLISHER;

```
SQL> desc publisher;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 NAME                                      NOT NULL VARCHAR2(20)
 PHONE                                              NUMBER(38)
 ADDRESS                                            VARCHAR2(20)
```

DESC BOOK;

DESC BOOK_AUTHORS;

```
SQL> DESC BOOK_AUTHORS;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 AUTHOR_NAME                               NOT NULL VARCHAR2(20)
 BOOK_ID                                   NOT NULL NUMBER(38)
```

DESC LIBRARY_BRANCH;

```
SQL> DESC LIBRARY_BRANCH;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 BRANCH_ID                                 NOT NULL NUMBER(38)
 BRANCH_NAME                                        VARCHAR2(50)
 ADDRESS                                            VARCHAR2(50)
```

DESC BOOK_COPIES;

```
SQL> DESC BOOK_COPIES;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 NO_OF_COPIES                                       NUMBER(38)
 BOOK_ID                                   NOT NULL NUMBER(38)
 BRANCH_ID                                 NOT NULL NUMBER(38)
```

DESC CARD;

```
SQL> DESC CARD;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 CARD_NO                                   NOT NULL NUMBER(38)
```

DESC BOOK_LENDING;

```
SQL> desc book_lending;
 Name
 ----------------------------------------------------------------------------------------------------
 DATE_OUT
 DUE_DATE
 BOOK_ID
 BRANCH_ID
 CARD_NO
```

Insertion of Values to Tables

INSERT INTO PUBLISHER VALUES (‘MCGRAW-HILL’, 9989076587, ‘BANGALORE’);
INSERT INTO PUBLISHER VALUES (‘PEARSON’, 9889076565, ‘NEWDELHI’);
INSERT INTO PUBLISHER VALUES (‘RANDOM HOUSE’, 7455679345, ‘HYDRABAD’);
INSERT INTO PUBLISHER VALUES (‘HACHETTE LIVRE’, 8970862340, ‘CHENAI’);
INSERT INTO PUBLISHER VALUES (‘GRUPO PLANETA’, 7756120238, ‘BANGALORE’);

INSERT INTO BOOK VALUES (1,’DBMS’,’JAN-2017’, ‘MCGRAW-HILL’);
INSERT INTO BOOK VALUES (2,’ADBMS’,’JUN-2016’, ‘MCGRAW-HILL’);
INSERT INTO BOOK VALUES (3,’CN’,’SEP-2016’, ‘PEARSON’);
INSERT INTO BOOK VALUES (4,’CG’,’SEP-2015’, ‘GRUPO PLANETA’);
INSERT INTO BOOK VALUES (5,’OS’,’MAY-2016’, ‘PEARSON’);

INSERT INTO BOOK_AUTHORS VALUES (’NAVATHE’, 1);
INSERT INTO BOOK_AUTHORS VALUES (’NAVATHE’, 2);
INSERT INTO BOOK_AUTHORS VALUES (’TANENBAUM’, 3);
INSERT INTO BOOK_AUTHORS VALUES (’EDWARD ANGEL’,
4); INSERT INTO BOOK_AUTHORS VALUES (’GALVIN’, 5);

INSERT INTO LIBRARY_BRANCH VALUES (10,’RR NAGAR’,’BANGALORE’);

```
INSERT INTO LIBRARY_BRANCH VALUES (11,'SKIT','BANGALORE');
INSERT INTO LIBRARY_BRANCH VALUES (12,'RAJAJI NAGAR', 'BANGALORE');
INSERT INTO LIBRARY_BRANCH VALUES (13,'NITTE','MANGALORE');
INSERT INTO LIBRARY_BRANCH VALUES (14,'MANIPAL','UDUPI');

INSERT INTO BOOK_COPIES VALUES (10, 1, 10);
INSERT INTO BOOK_COPIES VALUES (5, 1, 11);
INSERT INTO BOOK_COPIES VALUES (2, 2, 12);
INSERT INTO BOOK_COPIES VALUES (5, 2, 13);
INSERT INTO BOOK_COPIES VALUES (7, 3, 14);
INSERT INTO BOOK_COPIES VALUES (1, 5, 10);
INSERT INTO BOOK_COPIES VALUES (3, 4, 11);


INSERT INTO CARD VALUES (100);
INSERT INTO CARD VALUES (101);
INSERT INTO CARD VALUES (102);
INSERT INTO CARD VALUES (103);
INSERT INTO CARD VALUES (104);

INSERT INTO BOOK_LENDING VALUES ('01-JAN-17','01-JUN-17', 1, 10, 101);
INSERT INTO BOOK_LENDING VALUES ('11-JAN-17','11-MAR-17', 3, 14, 101);
INSERT INTO BOOK_LENDING VALUES ('21-FEB-17','21-APR-17', 2, 13, 101);
INSERT INTO BOOK_LENDING VALUES ('15-MAR-17','15-JUL-17', 4, 11, 101);
INSERT INTO BOOK_LENDING VALUES ('12-APR-17','12-MAY-17', 1, 11, 104);
```

SELECT * FROM PUBLISHER;

```
SQL> select * from publisher;

NAME                      PHONE ADDRESS
-------------------- ---------- --------------------
MCGRAW-HILL          9989076587 BANGALORE
PEARSON              9889076565 NEWDELHI
RANDOM HOUSE         7455679345 HYDRABAD
HACHETTE LIVRE       8970862340 CHENAI
GRUPO PLANETA        7756120238 BANGALORE
```

SELECT * FROM BOOK;

```
SQL> SELECT * FROM BOOK;

   BOOK_ID TITLE                PUB_YEAR             PUBLISHER_NAME
---------- -------------------- -------------------- --------------------
         1 DBMS                 JAN-2017             MCGRAW-HILL
         2 ADBMS                JUN-2016             MCGRAW-HILL
         3 CN                   SEP-2016             PEARSON
         4 CG                   SEP-2015             GRUPO PLANETA
         5 OS                   MAY-2016             PEARSON
```

SELECT * FROM BOOK_AUTHORS;

```
SQL> SELECT * FROM BOOK_AUTHORS;

AUTHOR_NAME             BOOK_ID
-------------------- ----------
NAVATHE                       1
NAVATHE                       2
TANENBAUM                     3
EDWARD ANGEL                  4
GALVIN                        5
```

SELECT * FROM LIBRARY_BRANCH;

```
SQL> SELECT * FROM LIBRARY_BRANCH;

 BRANCH_ID BRANCH_NAME                                        ADDRESS
---------- -------------------------------------------------- --------------------------------------
        10 RR NAGAR                                          BANGALORE
        11 RNSIT                                             BANGALORE
        12 RAJAJI NAGAR                                      BANGALORE
        13 NITTE                                             MANGALORE
        14 MANIPAL                                           UDUPI
```

SELECT * FROM BOOK_COPIES;
SELECT * FROM CARD

```
SQL> SELECT * FROM CARD;

   CARD_NO
----------
       100
       101
       102
       103
       104
```

SELECT * FROM BOOK_LENDING;

```
SQL> select * from book_lending;

DATE_OUT  DUE_DATE     BOOK_ID  BRANCH_ID    CARD_NO
--------- --------- ---------- ---------- ----------
01-JAN-17 01-JUN-17          1         10        101
11-JAN-17 11-MAR-17          3         14        101
21-FEB-17 21-APR-17          2         13        101
15-MAR-17 15-JUL-17          4         11        101
12-APR-17 12-MAY-17          1         11        104
```

Queries:

1. Retrieve details of all books in the library – id, title, name of publisher, authors, number of copies in each branch, etc.

SELECT B.BOOK_ID, B.TITLE, B.PUBLISHER_NAME, A.AUTHOR_NAME,
C.NO_OF_COPIES, L.BRANCH_ID
FROM BOOK B, BOOK_AUTHORS A, BOOK_COPIES C, LIBRARY_BRANCH L
WHERE B.BOOK_ID=A.BOOK_ID
AND B.BOOK_ID=C.BOOK_ID
AND L.BRANCH_ID=C.BRANCH_ID;

2. Get the particulars of borrowers who have borrowed more than 3 books, but from Jan 2017 to Jun 2017.

SELECT CARD_NO FROM
BOOK_LENDING
WHERE DATE_OUT BETWEEN ’01-JAN-2017’ AND ’01-JUL-2017’
GROUP BY CARD_NO
HAVING COUNT (*)>3;

```
   CARD_NO
----------
       101
```

3. Delete a book in BOOK table. Update the contents of other tables to reflect this · data manipulation operation.

DELETE FROM BOOK

WHERE BOOK_ID=3;

```
SQL> DELETE FROM BOOK
  2  WHERE BOOK_ID=3;

1 row deleted.

SQL> SELECT * FROM BOOK;

   BOOK_ID TITLE                PUB_YEAR             PUBLISHER_NAME
---------- -------------------- -------------------- --------------------
         1 DBMS                 JAN-2017             MCGRAW-HILL
         2 ADBMS                JUN-2016             MCGRAW-HILL
         4 CG                   SEP-2015             GRUPO PLANETA
         5 OS                   MAY-2016             PEARSON
```

4. Partition the BOOK table based on year of publication. Demonstrate its working with a simple query.

CREATE VIEW V_PUBLICATION AS
SELECT PUB_YEAR
FROM BOOK;

5. Create a view of all books and its number of copies that are currently available in the Library.

CREATE VIEW V_BOOKS AS
SELECT B.BOOK_ID, B.TITLE, C.NO_OF_COPIES
FROM BOOK B, BOOK_COPIES C, LIBRARY_BRANCH L
WHERE B.BOOK_ID=C.BOOK_ID
AND C.BRANCH_ID=L.BRANCH_ID;

```
   BOOK_ID TITLE                NO_OF_COPIES
---------- -------------------- ------------
         1 DBMS                           10
         1 DBMS                            5
         2 ADBMS                           2
         2 ADBMS                           5
         3 CN                              7
         5 OS                              1
         4 CG                              3
```

PROGRAM 5: ORDER PROCESSING DATABASE

Consider the following relations for an Order Processing database application in a company.

CUSTOMER (CUST #: int, cname: String, city: String)
ORDER (order #: int, odate: date, cust #: int, ord-Amt: int)
ITEM (item #: int, unit-price: int)
ORDER-ITEM (order #: int, item #: int, qty: int)
WAREHOUSE (warehouse #: int, city: String)
SHIPMENT (order #: int, warehouse #: int, ship-date: date)

WRITE THE SQL QUERIES TO:

6. Create the above tables by properly specifying the primary keys and the foreign keys and the foreign keys.
7. Enter at least five tuples for each relation.

8. Produce a listing: CUSTNAME, #oforders, AVG_ORDER_AMT, where the middle column is the total numbers of orders by the customer and the last column is the average order amount for that customer.
9. List the order# for orders that were shipped from all warehouses that the company has in a specific city.
10. Demonstrate how you delete item# 10 from the ITEM table and make that field *null* in the ORDER_ITEM table.

QUERY 1: Create the above tables by properly specifying the primary keys and the foreign keys.

```
SQL> create table customer
(
custno int,
cname varchar(20),
city varchar(20),
primary key(custno)
);
```
Table created.

```
SQL> desc customer
```
Name	Null?	Type
CUSTNO	NOT NULL	NUMBER(38)
CNAME		VARCHAR2(20)
CITY		VARCHAR2(20)

```
SQL> create table order
(
orderno int,
odate date,
custno int,ordamt int,
primary key(orderno),
foreign key(custno) references customer(custno)
);
```
Table created.

```
SQL> desc order
```
Name	Null?	Type
ORDERNO	NOT NULL	NUMBER(38)
ODATE		DATE
CUSTNO		NUMBER(38)
ORDAMT		NUMBER(38)

```
SQL> create table item
(
itemno int,
unitprice int,
```

```
primary key(itemno)
);
```

Table created.

```
SQL> desc item
Name                                      Null?                    Type
----------------------------- ---------- -------------------------------- ------------------------
ITEMNO                                    NOT NULL                 NUMBER(38)
UNITPRICE                                                          NUMBER(38)
```

```
SQL> create table order_item
(
orderno int,
itemno int,
qty int,
primary key(orderno),
 foreign key(orderno) references order (orderno),
foreign key(itemno) references item (itemno) on delete set NULL);
```

```
SQL> desc order_item
Name                                      Null?                    Type
------------------------------ -------- ---- ------------------------------------------------------
ORDERNO                                   NOT NULL
NUMBER(38)
ITEMNO                                    NOT NULL                 NUMBER(38)
QTY                                                                NUMBER(38)
```

```
SQL>create table warehouse
(
warehouseno int,
city varchar(20),
primary key(warehouseno)
);
```

Table created.

```
SQL>desc warehouse
Name                                      Null?                    Type
------------------------------ -------- ---- ------------------------------------------------------
WAREHOUSENO                               NOT NULL                 NUMBER(38)
CITY                                                               VARCHAR2(20)
```

Table created.

```
SQL> create table shipment
(
orderno int,warehouseno int,shipdate date,
primary key(orderno,warehouseno),
foreign key(orderno) references order(orderno),
foreign key(warehouseno) references warehouse(warehouseno));
```

Table created.

```
SQL> desc shipment
```

Name	Null?	Type
ORDERNO	NOT NULL	NUMBER(38)
WAREHOUSENO	NOT NULL	NUMBER(38)
SHIPDATE		DATE

QUERY 2: Enter at least five tuples for each relation

SQL> insert into customer values(&custno,'&cname','&city');
Enter value for custno: 771
Enter value for cname: PUSHPA K
Enter value for city: BANGALORE
old 1: insert into customer values(&custno,'&cname','&city')
new 1: insert into customer values(771,'PUSHPA K','BANGALORE')
1 row created.
SQL> /
Enter value for custno: 772
Enter value for cname: SUMAN
Enter value for city

old 1: insert into customer values(&custno,'&cname','&city')
new 1: insert into customer values(772,'SUMAN','MUMBAI')
1 row created.

SQL> /
Enter value for custno: 773
Enter value for cname: SOURAV
Enter value for city: CALICUT
old 1: insert into customer values(&custno,'&cname','&city')
new 1: insert into customer values(773,'SOURAV','CALICUT')
1 row created.
SQL> /
Enter value for custno: 774
Enter value for cname: LAILA
Enter value for city: HYDERABAD
old 1: insert into customer values(&custno,'&cname','&city')
new 1: insert into customer values(774,'LAILA','HYDERABAD')
1 row created.
SQL> /
Enter value for custno: 775
Enter value for cname: FAIZAL
Enter value for city: BANGALORE
old 1: insert into customer values(&custno,'&cname','&city')
new 1: insert into customer values(775,'FAIZAL','BANGALORE')
1 row created.

SQL> commit;
Commit complete.

SQL> select * from customer;

CUSTNO	CNAME	CITY
771	PUSHPA K	BANGALORE
772	SUMAN	MUMBAI
773	SOURAV	CALICUT
774	LAILA	HYDERABAD
775	FAIZAL	BANGALORE

SQL> insert into order values(&ordid,'&odate', &custno,&ordamt);
Enter value for ordid: 111
Enter value for odate: 22-JAN-02
Enter value for custno: 771
Enter value for ordamt: 18000
old 1: insert into order values(&ordid,'&odate',&custno, &ordamt)
new 1: insert into order values(111,'22-JAN-02',771,18000)
1 row created.
SQL> /
Enter value for ordid: 112
Enter value for odate: 30-JUL-02
Enter value for custno: 774
Enter value for ordamt: 6000
old 1: insert into order values(&ordid,'&odate', &custno, &ordamt)
new 1: insert into order values(112,'30-JUL-02',774,6000)
1 row created.
SQL> /
Enter value for ordid: 113
Enter value for odate: 03-APR-03
Enter value for custno: 775
Enter value for ordamt: 9000
old 1: insert into order values(&ordid,'&odate',&custno,&ordamt)
new 1: insert into order values(113,'03-APR-03',775,9000)
1 row created.
SQL> /
Enter value for ordid: 114
Enter value for odate: 03-NOV-03
Enter value for custno: 775
Enter value for ordamt: 29000
old 1: insert into order values(&ordid,'&odate', &custno, &ordamt)
new 1: insert into order values(114,'03-NOV-03',775,29000)
1 row created.
SQL> /
Enter value for ordid: 115
Enter value for odate: 10-DEC-03
Enter value for custno: 773
Enter value for ordamt: 29000.
old 1: insert into order values(&ordid,'&odate', &custno, &ordamt)
new 1: insert into order values(115,'10-DEC-03',773,29000.)
1 row created.
SQL> /
Enter value for ordid: 116
Enter value for odate:

Enter value for custno: 772
Enter value for ordamt: 56000
old 1: insert into order values(&ordid,'&odate',&custno,&ordamt)
new 1: insert into order values(116,'19-AUG-04',772,56000)
1 row created.
SQL> /
Enter value for ordid: 117
Enter value for odate: 10-SEP-04
Enter value for custno: 771
Enter value for ordamt: 20000
old 1: insert into order values(&ordid,'&odate',&custno,&ordamt)
new 1: insert into order values(117,'10-SEP-04',771,20000)
1 row created.
SQL> /
Enter value for ordid: 118
Enter value for odate: 20-NOV-04
Enter value for custno: 775
Enter value for ordamt: 29000
old 1: insert into order values(&ordid,'&odate',&custno,&ordamt)
new 1: insert into order values(118,'20-NOV-04',775,29000)
1 row created.
SQL> /
Enter value for ordid: 119
Enter value for odate: 13-FEB-05
Enter value for custno: 774
Enter value for ordamt: 29000
old 1: insert into order values(&ordid,'&odate',&custno,&ordamt)
new 1: insert into order values(119,'13-FEB-05',775,29000)
1 row created.
SQL> /
Enter value for ordid: 120
Enter value for odate: 13-OCT-05
Enter value for custno: 775
Enter value for ordamt: 29000
old 1: insert into order values(&ordid,'&odate',&custno,&ordamt)
new 1: insert into order values(120,'13-OCT-05',775,29000)
1 row created.
SQL> commit;
Commit complete.

SQL> select * from order;

ORDERNO	ODATE	CUSTNO	ORDAMT
111	22-JAN-02	771	18000
112	30-JUL-02	774	6000
113	03-APR-03	775	9000
114	03-NOV-03	775	29000
115	10-DEC-03	773	29000
116	19-AUG-04	772	56000

06 rows selected.

SQL> insert into item values(&itemno,&unitprice);
Enter value for itemno: 5001
Enter value for unitprice: 503
old 1: insert into item values(&itemno,&unitprice)
new 1: insert into item values(5001,503)
1 row created.
SQL> /
Enter value for itemno: 5002
Enter value for unitprice: 750
old 1: insert into item values(&itemno,&unitprice)
new 1: insert into item values(5002,750)
1 row created.
SQL> /
Enter value for itemno: 5003
Enter value for unitprice: 150
old 1: insert into item values(&itemno,&unitprice)
new 1: insert into item values(5003,150)
1 row created.
SQL> /
Enter value for itemno: 5004
Enter value for unitprice: 600
old 1: insert into item values(&itemno,&unitprice)
new 1: insert into item values(5004,600)
1 row created.
SQL> /
Enter value for itemno: 5005
Enter value for unitprice: 890
old 1: insert into item values(&itemno,&unitprice)
new 1: insert into item values(5005,890)
1 row created.

SQL> commit;
Commit complete.

SQL> select * from item;

ITEMNO	UNITPRICE
5001	503
5002	750
5003	150
5004	600
5005	890

SQL> insert into order_item values(&orderno,&itemno,&qty);
Enter value for orderno: 111
Enter value for itemno: 5001
Enter value for qty: 50
old 1: insert into order_item values(&orderno,&itemno,&qty)
new 1: insert into order_item values(111,5001,50)

1 row created.
SQL> /
Enter value for orderno: 112
Enter value for itemno: 5003
Enter value for qty: 20
old 1: insert into order_item values(&orderno,&itemno,&qty)
new 1: insert into order_item values(112,5003,20)
1 row created.
SQL> /
Enter value for orderno: 113
Enter value for itemno: 5002
Enter value for qty: 50
old 1: insert into order_item values(&orderno,&itemno,&qty)
new 1: insert into order_item values(113,5002,50)
1 row created.
SQL> /
Enter value for orderno: 114
Enter value for itemno: 5005
Enter value for qty: 60
old 1: insert into order_item values(&orderno,&itemno,&qty)
new 1: insert into order_item values(114,5005,60)
1 row created.
SQL> /
Enter value for orderno: 115
Enter value for itemno: 5004
Enter value for qty: 90
old 1: insert into order_item values(&orderno,&itemno,&qty)
new 1: insert into order_item values(115,5004,90)
1 row created.
SQL> /
Enter value for orderno: 116
Enter value for itemno: 5001
Enter value for qty: 10
old 1: insert into order_item values(&orderno,&itemno,&qty)
new 1: insert into order_item values(116,5001,10)
1 row created.
SQL> /
Enter value for orderno: 117
Enter value for itemno: 5003
Enter value for qty: 80
old 1: insert into order_item values(&orderno,&itemno,&qty)
new 1: insert into order_item values(117,5003,80)
1 row created.
SQL> /
Enter value for orderno: 118
Enter value for itemno: 5005
Enter value for qty: 50
old 1: insert into order_item values(&orderno,&itemno,&qty)
new 1: insert into order_item values(118,5005,50
1 row created.
SQL> /
Enter value for orderno: 119

Enter value for itemno: 5002
Enter value for qty: 10
old 1: insert into order_item values(&orderno,&itemno,&qty)
new 1: insert into order_item values(119,5002,10)
1 row created.
SQL> /
Enter value for orderno: 120
Enter value for itemno: 5004
Enter value for qty: 45
old 1: insert into order_item values(&orderno,&itemno,&qty)
new 1: insert into order_item values(120,5004,45)
1 row created.
SQL> commit;
Commit complete.

SQL> select * from order_item;

ORDERNO	ITEMNO	QTY
111	5001	50
112	5003	20
113	5002	50
114	5005	60
115	5004	90
116	5001	10

06 rows selected.

SQL> insert into warehouse values(&warehouseno,'&city');
Enter value for warehouseno: 1
Enter value for city: DELHI
old 1: insert into warehouse values(&warehouseno,'&city')
new 1: insert into warehouse values(1,'DELHI')
1 row created.
SQL> /
Enter value for warehouseno: 2
Enter value for city: BOMBAY
old 1: insert into warehouse values(&warehouseno,'&city')
new 1: insert into warehouse values(2,'BOMBAY')
1 row created.
SQL> /
Enter value for warehouseno: 3
Enter value for city: CHENNAI
old 1: insert into warehouse values(&warehouseno,'&city')
new 1: insert into warehouse values(3,'CHENNAI')
1 row created.
SQL> /
Enter value for warehouseno: 4
Enter value for city: BANGALORE
old 1: insert into warehouse values(&warehouseno,'&city')
new 1: insert into warehouse values(4,'BANGALORE')
1 row created.
SQL> /
Enter value for warehouseno: 5

Enter value for city: BANGALORE
old 1: insert into warehouse values(&warehouseno,'&city')
new 1: insert into warehouse values(5,'BANGALORE')
1 row created.
SQL> /
Enter value for warehouseno: 6
Enter value for city: DELHI
old 1: insert into warehouse values(&warehouseno,'&city')
new 1: insert into warehouse values(6,'DELHI')
1 row created.
SQL> /
Enter value for warehouseno: 7
Enter value for city: BOMBAY
old 1: insert into warehouse values(&warehouseno,'&city')
new 1: insert into warehouse values(7,'BOMBAY')
1 row created.

SQL> /
Enter value for warehouseno: 8
Enter value for city: CHENNAI
old 1: insert into warehouse values(&warehouseno,'&city')
new 1: insert into warehouse values(8,'CHENNAI')
1 row created.
SQL> /
Enter value for warehouseno: 9
Enter value for city: DELHI
old 1: insert into warehouse values(&warehouseno,'&city')
new 1: insert into warehouse values(9,'DELHI')
1 row created.
SQL> /
Enter value for warehouseno: 10
Enter value for city: BANGALORE
old 1: insert into warehouse values(&warehouseno,'&city')
new 1: insert into warehouse values(10,'BANGALORE')
1 row created.
SQL> commit;
Commit complete.

SQL> select * from warehouse;

WAREHOUSENO	CITY
1	DELHI
2	BOMBAY
3	CHENNAI
4	BANGALORE
5	BANGALORE
6	DELHI
7	BOMBAY
8	CHENNAI
9	DELHI
10	BANGALORE

10 rows selected.

SQL> insert into shipment values(&orderno,&warehouseno,'&shipdate');
Enter value for orderno: 111
Enter value for warehouseno: 1
Enter value for shipdate: 10-FEB-02
old 1: insert into shipment values(&orderno,&warehouseno,'&shipdate')
new 1: insert into shipment values(111,1,'10-FEB-02')
1 row created.
SQL> /
Enter value for orderno: 112
Enter value for warehouseno: 5
Enter value for shipdate: 10-SEP-02
old 1: insert into shipment values(&orderno,&warehouseno,'&shipdate')
new 1: insert into shipment values(112,5,'10-SEP-02')
1 row created.
SQL> /
Enter value for orderno: 113
Enter value for warehouseno: 8
Enter value for shipdate: 10-FEB-03
old 1: insert into shipment values(&orderno,&warehouseno,'&shipdate')
new 1: insert into shipment values(113,8,'10-FEB-03')
1 row created.
SQL> /
Enter value for orderno: 114
Enter value for warehouseno: 3
Enter value for shipdate: 10-DEC-03
old 1: insert into shipment values(&orderno,&warehouseno,'&shipdate')
new 1: insert into shipment values(114,3,'10-DEC-03')
1 row created.
SQL> /
Enter value for orderno: 115
Enter value for warehouseno: 9
Enter value for shipdate: 19-JAN-04
old 1: insert into shipment values(&orderno,&warehouseno,'&shipdate')
new 1: insert into shipment values(115,9,'19-JAN-04')
1 row created.
SQL> /
Enter value for orderno: 116
Enter value for warehouseno: 1
Enter value for shipdate: 20-SEP-04
old 1: insert into shipment values(&orderno,&warehouseno,'&shipdate')
new 1: insert into shipment values(116,1,'20-SEP-04')
1 row created.
SQL> /
Enter value for orderno: 117
Enter value for warehouseno: 5
Enter value for shipdate: 10-SEP-04
old 1: insert into shipment values(&orderno,&warehouseno,'&shipdate')
new 1: insert into shipment values(117,5,'10-SEP-04')
1 row created.
SQL> /
Enter value for orderno: 118

Enter value for warehouseno: 7
Enter value for shipdate: 30-NOV-04
old 1: insert into shipment values(&orderno,&warehouseno,'&shipdate')
new 1: insert into shipment values(118,7,'30-NOV-04')
1 row created.
SQL> /
Enter value for orderno: 119
Enter value for warehouseno: 7
Enter value for shipdate: 30-APR-05
old 1: insert into shipment values(&orderno,&warehouseno,'&shipdate')
new 1: insert into shipment values(119,7,'30-APR-05')
1 row created.
SQL> /
Enter value for orderno: 120
Enter value for warehouseno: 6
Enter value for shipdate: 21-DEC-05
old 1: insert into shipment values(&orderno,&warehouseno,'&shipdate')
new 1: insert into shipment values(120,6,'21-DEC-05')
1 row created.
SQL> commit;
Commit complete.
SQL> select * from shipment;

ORDERNO	WAREHOUSENO	SHIPDATE
111	1	10-FEB-02
112	5	10-SEP-02
113	8	10-FEB-03
114	3	10-DEC-03
115	9	19-JAN-04
116	1	20-SEP-04
117	5	10-SEP-04
118	7	30-NOV-04
119	7	30-APR-05
120	6	21-DEC-05

10 rows selected.

QUERY 3: Produce a listing: CUSTNAME, #of orders, AVG_ORDER_AMT, where the middle column is the total numbers of orders by the customer and the last column is the average order amount for that customer.

SQL> select cname CUSTNAME, count(orderno) NOOFORDERS, avg(ordamt) AVGORDAMT from customer a,order b where a.custno=b.custno group by cname;

CUSTNAME	NOOFORDERS	AVGORDAMT
FAIZAL	4	24000
LAILA	2	17500
PUSHPA K	2	19000
SOURAV	1	29000
SUMAN	1	56000

QUERY 4: List the order# for orders that were shipped from all warehouses that the company has in a specific city.

```
SQL> select * from order_cust where orderno in(
2 select orderno from shipment where warehouseno in(
3 select warehouseno from warehouse where city='CHENNAI'));
```

ORDERNO	ODATE	CUSTNO	ORDAMT
113	03-APR-03	775	9000
114	03-NOV-03	775	29000

QUERY 5: Demonstrate how you delete item # 10 from ITEM table and make *null* in the ORDER_ITEM table.

```
SQL> delete from item1 where itemno=5001;
1 row deleted.
```

PART B: Mini project

For any problem selected, make sure that the application should have **five or more** tables. Indicative areas include: Organization, health care, Ecommerce etc.

- Students can pick one experiment from the questions lot of PART A with an equal choice to all the students in a batch. For PART B, the project group (Maximum of 4 students per batch) should demonstrate the mini-project.
- Weightage of marks for PART A is 60% and for PART B is 40%.
- Change of experiment is allowed only once and Marks allotted to the procedure part to be made zero (Not allowed for Part B).
- Mini project can be done using any DBMS – for back end and any Programming language for the front end as per the choice of students.
- Mini-Project report should be submitted in the form of Hard copy, spiral binding and it should be as per the department standards and format.