B.M.S. EDUCATION TRUST

B.M.S.COLLEGE OF ENGINEERING, BANGALORE-19

(Autonomous College under VTU)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

DATABASE MANAGEMENT SYSTEM LABORATORY MANUAL 19CS4PCDBM

PROGRAM: BACHELOR OF ENGINEERING
SEMESTER: IV
SESSION:2020
COURSE CODE: 19CS4PCDBM
COURSE TITLE: DATABASE MANAGEMENT SYSTEM
CREDITS: 4

PREFACE

This laboratory manual is prepared by the Department of Computer Science and engineering for Database Management Systems Laboratory (19CS4PCDBM). This lab manual can be used as instructional book for students, staff and instructors to assist in performing and understanding the experiments. In this manual, experiments as per syllabus are described.

DATABASE MANAGEMENT SYSTEM LABORATORY MANUAL

DBMS Lab List

Proposed Lab Plan

Instructions to Students to be followed in each DBMS lab:

1. Each Student should write down the work carried out and the outputs in the observation book and get it evaluated by the respective lab faculty in-charge.

2. Each Student should bring the lab record with the programs and output written for the programs completed in their respective previous week and gets it evaluated by the lab faculty in-charge.

Writing SQL Queries using Oracle for the following database systems

.

Experiment #	Name of Experiment
1	Insurance Database
2	Banking Enterprise Database
3	Supplier Database
4	Student Faculty Database
5	Airline Flight Database
6	Order Processing Database
7	Book dealer Database
8	Student Enrolment Database
9	Movie Database
10	College Database

PROGRAM 1: INSURANCE DATABASE

Consider the Insurance database given below. The data types are specified.

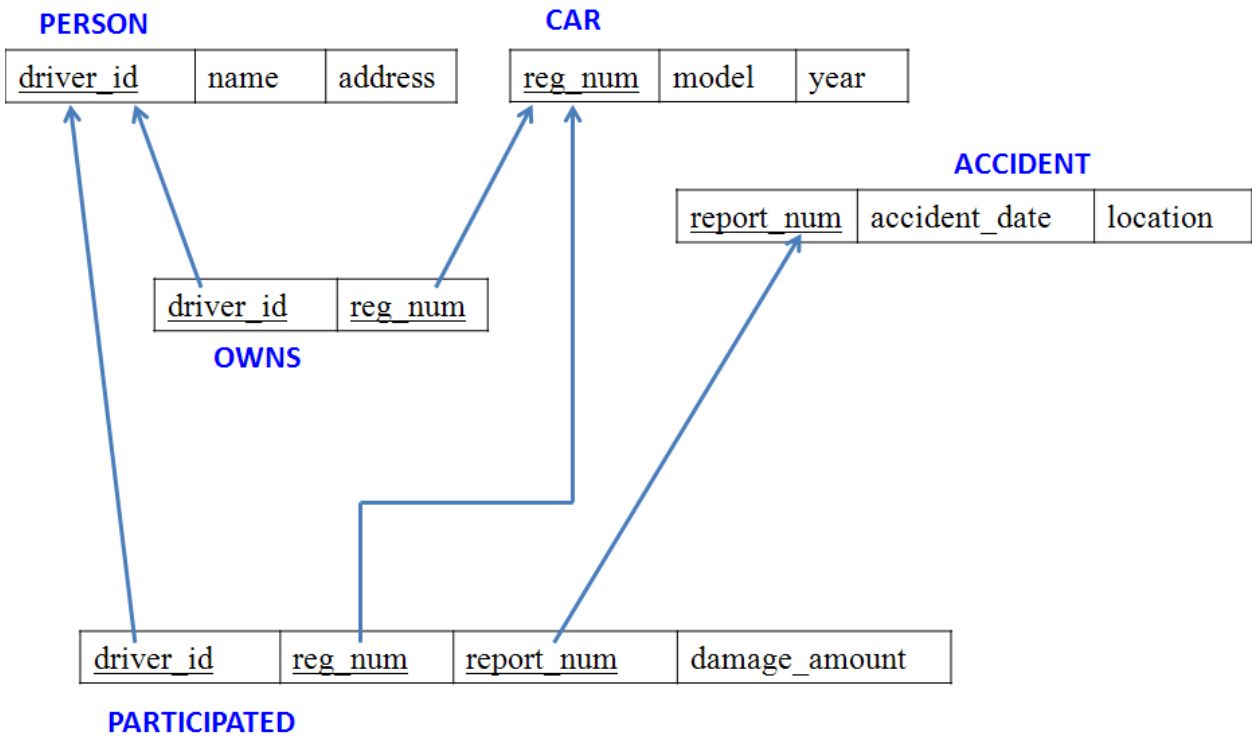
PERSON (driver_id: String, name: String, address: String)
CAR (reg_num: String, model: String, year: int)
ACCIDENT (report_num: int, accident_date: date, location: String)
OWNS (driver_id: String, reg_num: String)
PARTICIPATED (driver_id: String,reg_num: String, report_num: int, damage_amount: int)

i) Create the above tables by properly specifying the primary keys and the foreign keys.
ii)Enter at least five tuples for each relation.
iii)Demonstrate how you

a.Update the damage amount to 25000 for the car with a specific reg-num(example 'K A053408') for which the accident report number was 12.

b.Add a new accident to the database.
iv)Find the total number of people who owned cars that involved in accidents in 2008.
v)Find the number of accidents in which cars belonging to a specific model (example)were involved.

Schema diagram



Tables

PERSON

driver_id	name	address
A01	Richard	Srinivas nagar
A02	Pradeep	Rajaji nagar
A03	Smith	Ashok nagar
A04	Venu	N R Colony
A05	Jhon	Hanumanth nagar

CAR

reg_num	model	year
KA052250	Indica	1990
KA031181	Lancer	1957
KA095477	Toyota	1998
KA053408	Honda	2008
KA041702	Audi	2005

OWNS

driver_id	reg_num
A01	KA052250
A02	KA053408
A03	KA031181
A04	KA095477
A05	KA041702

ACCIDENT

report_num	accident_date	location
11	01-JAN-03	Mysore Road
12	02-FEB-04	South end Circle
13	21-JAN-03	Bull temple Road
14	17-FEB-08	Mysore Road
15	04-MAR-05	Kanakpura Road

PARTICIPATED

driver_id	reg_num	report_num	damage_amount
A01	KA052250	11	10000
A02	KA053408	12	50000
A03	KA095477	13	25000
A04	KA031181	14	3000
A05	KA041702	15	5000

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

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

SQL>**create table person (driver_id varchar(10),**
name varchar(20),
address varchar(30),
primary key(driver_id));
Table created.

SQL>**desc person**

Name	Null?	Type
DRIVER_ID	NOT NULL	VARCHAR2(10)
NAME		VARCHAR2(20)
ADDRESS		VARCHAR2(30)

SQL>**create table car(reg_num varchar(10),model varchar(10),year int,primary key(reg_num));**
Table created.

SQL>**desc car**

Name	Null?	Type
REG_NUM	NOT NULL	VARCHAR2(10)
MODEL		VARCHAR2(10)
YEAR		NUMBER(38)

SQL>**create table accident(report_num int,accident_date date,location varchar(20),primary key(report_num));**

Table created.

SQL>**desc accident**

Name	Null?	Type
REPORT_NUM	NOT NULL	NUMBER(38)
ACCIDENT_DATE		DATE
LOCATION		VARCHAR2(20)

SQL>**create table owns(driver_id varchar(10),reg_num varchar(10),**
primary key(driver_id,reg_num),
foreign key(driver_id) referencesperson(driver_id),
foreign key(reg_num) references car(reg_num));

Table created.

SQL>**desc owns**

```
Name                          Null?                    Type
------------------------------ -------- ------------------------------
DRIVER_ID NOT NULL VARCHAR2(10)
REG_NUM                       NOT NULL VARCHAR2(10)
```

SQL>**create table participated(driver_id varchar(10), reg_num varchar(10),**
report_num int, damage_amount int,
primary key(driver_id,reg_num,report_num),
foreign key(driver_id) references person(driver_id),
foreign key(reg_num) references car(reg_num),
foreign key(report_num) references accident(report_num));
Table created.
SQL>**desc participated**

```
Name                          Null?            Type
------------------------------ -------- ----------------------------------------------
DRIVER_ID                     NOT NULL       VARCHAR2(10)
REG_NUM                         NOT NULL     VARCHAR2(10)
REPORT_NUM                    NOT NULL     NUMBER(38)
DAMAGE_AMOUNT                          NUMBER(38)
```

QUERY 2: Enter at least five tuples for each relation

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

1 row created.

SQL>/
Enter value for driver_id: A02
Enter value for name: Pradeep
Enter value for address: Rajajinagar
old 1: insert into person values('&driver_id','&name','&address')
new 1: insert into person values('A02','Pradeep','Rajajinagar')

1 row created.

SQL>**commit;**
Commit complete.

SQL> select * from person;

```
DRIVER_ID  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('&reg_num','&model', &year);

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

1 row created.

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

1 row created.

SQL>**commit;**

Commit complete.

SQL> select * from car;

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

SQL> insert into accident values(&report_num,'&accident_date','&location');

Enter value for report_num: 11
Enter value for accident_date: 01-JAN-03
Enter value for location: Mysore Road
old 1: insert into accident values(&report_num,'&accident_date','&location')
new 1: insert into accident values(111,'01-JAN-03','Mysore Road')

1 row created.

SQL>**commit;**

Commit complete.
SQL> select * from accident;

REPORT_NUM	ACCIDENT_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 ('&driver_id','&reg_num');

Enter value for driver_id: A01
Enter value for reg_num: KA052250
old 1: insert into owns values('&driver_id','&reg_num')
new 1: insert into owns values('A01','KA052250')
1 row created.

SQL>**commit;**
Commit complete.

SQL> select * from owns;

DRIVER_ID	REG_NUM
A01	KA052250
A02	KA053408
A04	KA031181
A03	KA095477
A05	KA041702

SQL> insert into participated values ('&driver_id','&reg_num',&report_num, &damage_amount);

Enter value for driver_id: A01
Enter value for reg_num: KA052250
Enter value for report_num: 11
Enter value for damage_amount: 10000
old 1: insert into participated values ('&driver_id','&reg_num',&report_num,&damage_amount)
new 1: insert into participated values('A01','KA052250',11,10000)

1 row created.

SQL>/
Enter value for driver_id: A02
Enter value for reg_num: KA053408
Enter value for report_num: 12
Enter value for damage_amount: 50000
old 1: insert into participated values ('&driver_id','&reg_num', &report_num,&
damage_amount)
new 1: insert into participated values('A02','KA053408',12,50000)

1 row created.

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

DRIVER_ID	REG_NUM	REPORT_NUM	DAMAGE_AMOUNT
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 to 25000 for the car with a specific reg_num (example 'K A053408') for which the accident report number was 12.

SQL> update participated set damage_amount=25000 where reg_num='KA053408' and report_num=12;

1 row updated.

SQL>**commit;**

Commit complete.

SQL>**select * from participated;**

DRIVER_ID	REG_NUM	REPORTNUM	DAMAGE_AMOUNT
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;**

REPORT_NUM	ACCIDENT_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.report_num=b.report_num and b.accident_date like '%08';**

CNT
1

QUERY 5: Find the number of accidents in which cars belonging to a specific model (example 'Lancer') were involved.

SQL> select count(report_num) CNT from car c,participated p where c.reg_num=p.reg_num and model='Lancer';

CNT
1

ADDITIONAL QUERIES:

1) LIST THE ENTIRE PARTICIPATED RELATION IN THE DESCENDING ORDER OF DAMAGE AMOUNT.

SQL> SELECT * FROM PARTICIPATED ORDER BY DAMAGE_AMOUNTT DESC;

2) FIND THE AVERAGE DAMAGE AMOUNT

SQL> SELECT AVG(DAMAGE_AMOUNTT) FROM PARTICIPATED;

3) DELETE THE TUPLE WHOSE DAMAGE AMOUNT IS BELOW THE AVERAGE DAMAGE AMOUNT

```
SQL> DELETE FROM PARTICIPATED WHERE
     DAMAGE_AMOUNTT<(SELECT AVG (DAMAGE_AMOUNT) FROM PARTICIPATED);
```

4) LIST THE NAME OF DRIVERS WHOSE DAMAGE IS GREATER THAN THE AVERAGE DAMAGE AMOUNT.

```
SQL> SELECT NAME FROM PERSON A, PARTICIPATED B WHERE A.DRIVER_ID = B.DRIVER_ID AND
DAMAGE_AMOUNT>(SELECT AVG(DAMAGE_AMOUNT) FROM PARTICIPATED);
```

5) FIND MAXIMUM DAMAGE AMOUNT.

```
SQL>SELECT MAX(DAMAGE_AMOUNT) FROM PARTICIPATED;
```

PROGRAM 2: BANKING ENTERPRISE DATABASE

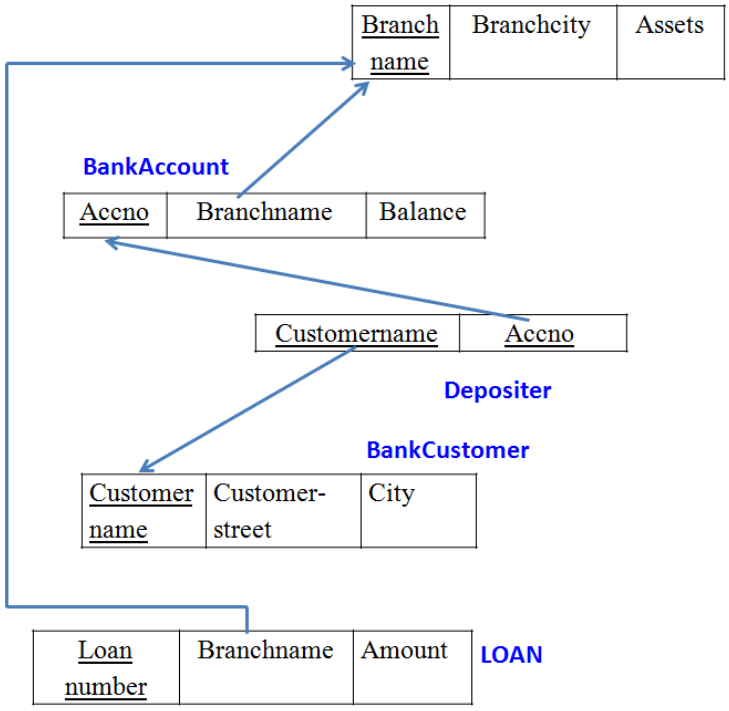
Consider the following database for a banking enterprise.

Branch (branch-name: String, branch-city: String, assets: real)
BankAccount(accno: int, branch-name: String, balance: real)
BankCustomer (customer-name: String, customer-street: String,customer-city: String)
Depositer(customer-name: String, accno: int)
Loan(loan-number: int, branch-name: String, amount: real)

i. Create the above tables by properly specifying the primary keys and the foreign keys.
ii. Enter at least five tuples for each relation.
iii. Find all the customers who have at least two accounts at the *Main* branch (ex. SBI_ResidencyRoad).
iv. Find all the customers who have an account at *all* the branches located in a specific city (Ex. Delhi).
v. Demonstrate how you delete all account tuples at every branch located in a specific city (Ex. Bombay).

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 depositors of the corresponding branches are maintained.

Schema Diagram



Sample Table data

Branch

BRANCHNAME	BRANCHCITY	ASSESTS
SBI_Chamrajpet	Bangalore	50000
SBI_ResidencyRoad	Bangalore	10000
SBI_ShivajiRoad	Bombay	20000
SBI_ParlimentRoad	Delhi	10000
SBI_Jantarmantar	Delhi	20000

BankAccount

ACCNO	BRANCHNAME	BALANCE
1	SBI_Chamrajpet	2000
2	SBI_ResidencyRoad	5000
3	SBI_ShivajiRoad	6000
4	SBI_ParlimentRoad	9000
5	SBI_Jantarmantar	8000
6	SBI_ShivajiRoad	4000
8	SBI_ResidencyRoad	4000
9	SBI_ParlimentRoad	3000
10	SBI_ResidencyRoad	5000
11	SBI_Jantarmantar	2000

BankCustomer

CUSTOMERNAME	CUSTOMERSTREET	CUSTOMERCITY
Avinash	Bull_Temple_Road	Bangalore
Dinesh	Bannergatta_Road	Bangalore
Mohan	NationalCollege_Road	Bangalore
Nikil	Akbar_Road	Delhi
Ravi	Prithviraj_Road	Delhi

Depositer

CUSTOMERNAME	ACCNO
Avinash	1
Dinesh	2
Nikil	4
Ravi	5
Avinash	8
Nikil	9
Dinesh	10
Nikil	11

Loan

LOANNUMBER	BRANCHNAME	AMOUNT
1	SBI_Chamrajpet	1000
2	SBI_ResidencyRoad	2000
3	SBI_ShivajiRoad	3000
4	SBI_ParlimentRoad	4000
5	SBI_Jantarmantar	5000

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

```
SQL> create table Branch(branchname varchar(30),branchcity varchar(30),assests r
eal, primary key(branchname));
SQL> desc Branch
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------

 BRANCHNAME                                NOT NULL VARCHAR2(30)
 BRANCHCITY                                         VARCHAR2(30)
 ASSESTS                                            FLOAT(63)
```

```
SQL> create table BankAccount(accno integer,branchname varchar(30), balance real
,primary key (accno),foreign key (branchname) references Branch(branchname));
SQL> desc BankAccount
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------

 ACCNO                                     NOT NULL NUMBER(38)
 BRANCHNAME                                         VARCHAR2(30)
 BALANCE                                            FLOAT(63)

SQL> create table BankCustomer(customername varchar(30),customerstreet varchar(3
),customercity varchar(30),primary key(customername));

Table created.

SQL> desc BankCustomer
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------

 CUSTOMERNAME                              NOT NULL VARCHAR2(30)
 CUSTOMERSTREET                                     VARCHAR2(3)
 CUSTOMERCITY                                       VARCHAR2(30)
SQL> create table Depositer(customername varchar(30),accno integer,primary key(c
ustomername,accno),foreign key(customername) references BankCustomer(customernam
e), foreign key(accno) references BankAccount(accno));

Table created.

SQL> desc Depositer;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------

 CUSTOMERNAME                              NOT NULL VARCHAR2(30)
 ACCNO                                     NOT NULL NUMBER(38)

SQL> create table Loan (loannumber int,branchname varchar(30),amount real,primar
y key (loannumber), foreign key (branchname) references Branch(branchname));
```

QUERY 2: Enter at least five tuples for each relation

```
SQL> insert into Branch values('SBI_Chamrajpet','Bangalore',50000);

1 row created.

SQL> insert into Branch values('SBI_ResidencyRoad','Bangalore',10000);

1 row created.

SQL> insert into Branch values('SBI_ShivajiRoad','Bombay',20000);

1 row created.

SQL> insert into Branch values('SBI_ParlimentRoad','Delhi',10000);

1 row created.

SQL> insert into Branch values('SBI_Jantarmantar','Delhi',20000);

1 row created.
SQL> select * from Branch;

BRANCHNAME                     BRANCHCITY                          ASSESTS
------------------------------ ------------------------------ ----------
SBI_Chamrajpet                 Bangalore                           50000
SBI_ResidencyRoad              Bangalore                           10000
SBI_ShivajiRoad                Bombay                              20000
SBI_ParlimentRoad              Delhi                               10000
SBI_Jantarmantar               Delhi                               20000
```

Insert records for Loan

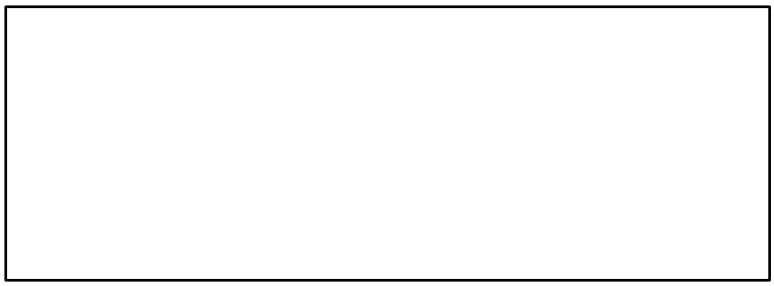
```
SQL> insert into Loan values(2,'SBI_ResidencyRoad',2000);
SQL> insert into Loan values(1,'SBI_Chamrajpet',1000);
SQL> insert into Loan values(3,'SBI_ShivajiRoad',3000);
SQL> insert into Loan values(4,'SBI_ParlimentRoad',4000);
SQL> insert into Loan values(5,'SBI_Jantarmantar',5000);

SQL> select * from Loan;

LOANNUMBER BRANCHNAME                     AMOUNT
---------- ------------------------------ ----------
         1 SBI_Chamrajpet                       1000
         2 SBI_ResidencyRoad                    2000
         3 SBI_ShivajiRoad                      3000
         4 SBI_ParlimentRoad                    4000
         5 SBI_Jantarmantar                     5000
```

Similarly insert records for BankAccount, Depositer and BankCustomer

```
SQL> insert into BanKAccount values(11,'SBI_Jantarmantar',2000);

1 row created.

SQL> commit;

Commit complete.

SQL> select * from BankAccount;

     ACCNO BRANCHNAME                         BALANCE
---------- ------------------------------ ----------
         1 SBI_Chamrajpet                       2000
         2 SBI_ResidencyRoad                    5000
         3 SBI_ShivajiRoad                      6000
         4 SBI_ParlimentRoad                    9000
         5 SBI_Jantarmantar                     8000
         6 SBI_ShivajiRoad                      4000
         8 SBI_ResidencyRoad                    4000
         9 SBI_ParlimentRoad                    3000
        10 SBI_ResidencyRoad                    5000
        11 SBI_Jantarmantar                     2000
```

BRANCHNAME	BRANCHCITY	ASSESTS
SBI_Chamrajpet	Bangalore	50000
SBI_ResidencyRoad	Bangalore	10000
SBI_ShivajiRoad	Bombay	20000
SBI_ParlimentRoad	Delhi	10000
SBI_Jantarmantar	Delhi	20000

ACCNO	BRANCHNAME	BALANCE
1	SBI_Chamrajpet	2000
2	SBI_ResidencyRoad	5000
3	SBI_ShivajiRoad	6000
4	SBI_ParlimentRoad	9000
5	SBI_Jantarmantar	8000
6	SBI_ShivajiRoad	4000
8	SBI_ResidencyRoad	4000
9	SBI_ParlimentRoad	3000
10	SBI_ResidencyRoad	5000
11	SBI_Jantarmantar	2000



CUSTOMERNAME	ACCNO
Avinash	1
Dinesh	2
Nikil	4
Ravi	5
Avinash	8
Nikil	9
Dinesh	10
Nikil	11

LOANNUMBER	BRANCHNAME	AMOUNT
1	SBI_Chamrajpet	1000
2	SBI_ResidencyRoad	2000
3	SBI_ShivajiRoad	3000
4	SBI_ParlimentRoad	4000
5	SBI_Jantarmantar	5000

SQL> commit;

Commit complete.

QUERY 3: Find Find all the customers who have at least two deposits at the same branch (Ex. 'SBI_ResidencyRoad').

```
select C.customername
from BankCustomer C
where exists (
select D.customername, count(D.customername)
fromdepositer D, BankAccount BA
where
D.accno= BA.accno AND
C.customername= D.customernameAND
BA.branchname= 'SBI_ResidencyRoad'
group by D. customername
havingcount(D.customername)>=2;
                );
```

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

```
selectBC.customername
fromBankCustomer BC
where not exists  (
selectbracnhname from Branch where
branchcity='Delhi'
 minus
                        (selectBA.branchname from Depositer
D, BankAccount BA
whereD.accno=BA.accno and
BC.customername=D.customername)
                        );
```

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

```
delete  from BankAccount
wherebranchname IN (
selectbranchname
from Branch
wherebranchcity='BOMBAY'
                    );
```

ADDITIONAL QUERIES:

1. **LIST THE ENTIRE LOAN RELATION IN THE DESCENDING ORDER OF AMOUNT.**

SQL> SELECT * FROM LOAN ORDER BY AMOUNT DESC;

2. **FIND ALL CUSTOMERS HAVING A LAON, AN ACCOUNT OR BOTH AT THE BANK**

SQL> (SELECT CUSTOMER_NAME FROM DEPOSITOR) UNION (SELECT CUSTOMER_NAME FROM BORROWER);

3. **CREATE A VIEW WHICH GIVES EACH BRANCH THE SUM OF THE AMOUNT OF ALL THE LOANS AT THE BRANCH.**

SQL> CREATE VIEW BRANCH_TOTAL_LOAN (BRANCH_NAME, TOTAL_LOAN) AS SELECT BRANCH_NAME, SUM(AMOUNT) FROM LOAN GROUP BY BRANCH_NAME;

4. **THE ANNUAL INTEREST PAYMENTS ARE MADE AND ALL BRANCHES ARE TO BE INCREASED BY 5%.**

SQL> UPDATE ACCOUNT SET BALANCE=BALANCE *1.05;

PROGRAM 3: SUPPLIER DATABASE

Consider the following schema:

SUPPLIERS(sid: integer, sname: string, address: string)

PARTS(pid: integer, pname: string, color: string)

CATALOG(sid: integer, pid: integer, cost: real)

The Catalog relation lists the prices charged for parts by Suppliers.

Write the following queries in SQL:

i) Find the pnames of parts for which there is some supplier.

ii) Find the snames of suppliers who supply every part.

iii) Find the snames of suppliers who supply every red part.

iv) Find the pnames of parts supplied by Acme Widget Suppliers and by no one else.

v) Find the sids of suppliers who charge more for some part than the average cost of that part (averaged over all the suppliers who supply that part).

vi) For each part, find the sname of the supplier who charges the most for that part.

Schema Diagram

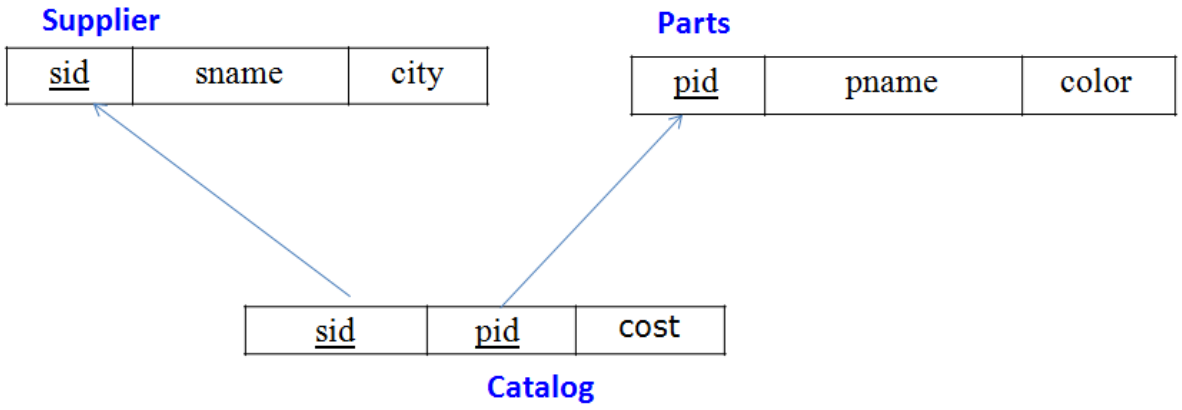


Table Data

SUPPLIERS

SID	SNAME	CITY
10001	Acme Widget	Bangalore
10002	Johns	Kolkata
10003	Vimal	Mumbai
10004	Reliance	Delhi

PARTS

PID	PNAME	COLOR
20001	Book	Red
20002	Pen	Red
20003	Pencil	Green
20004	Mobile	Green
20005	Charger	Black

CATALOG

SID	PID	COST
10001	20001	10
10001	20002	10
10001	20003	30
10001	20004	10
10001	20005	10
10002	20001	10
10002	20002	20
10003	20003	30
10004	20003	40

CREATION of Tables:

SQL> create table SUPPLIERS(sid number(5) primary key, sname varchar(20), city varchar(20));

Table created.

SQL> desc SUPPLIERS;

Name	Null?	Type
SID	NOT NULL	NUMBER(5)
SNAME		VARCHAR2(20)
CITY		VARCHAR2(20)

SQL> create table PARTS(pid number(5) primary key, pname varchar(20), color varchar(10));

Table created.

SQL> desc PARTS;

Name	Null?	Type
PID	NOT NULL	NUMBER(5)

PNAME	VARCHAR2(20)
COLOR	VARCHAR2(10)

SQL> create table CATALOG(sid number(5), pid number(5), foreign key(sid) references SUPPLIERS(sid), foreign key(pid) references PARTS(pid), cost float(6), primary key(sid, pid));

Table created.

SQL> desc CATALOG;

Name	Null?	Type
SID	NOT NULL	NUMBER(5)
PID	NOT NULL	NUMBER(5)
COST		FLOAT(6)

INSERTION OF DATA:

SQL> insert into suppliers values(&sid, '&sname','&city');

Enter value for sid: 10001

Enter value for sname: Acme Widget

Enter value for address: Bangalore

old 1: insert into suppliers values(&sid, '&sname','&city')

new 1: insert into suppliers values(10001, 'Acme Widget','Bangalore')

1 row created.

SQL> /

Enter value for sid: 10002

Enter value for sname: Johns

Enter value for address: Kolkata

old 1: insert into suppliers values(&sid, '&sname','&city')
new 1: insert into suppliers values(10002, 'Johns','Kolkata')

1 row created.

SQL> /
Enter value for sid: 10003
Enter value for sname: Vimal
Enter value for address: Mumbai
old 1: insert into suppliers values(&sid, '&sname','&city')
new 1: insert into suppliers values(10003, 'Vimal','Mumbai')

1 row created.

SQL> /
Enter value for sid: 10004
Enter value for sname: Reliance
Enter value for address: Delhi
old 1: insert into suppliers values(&sid, '&sname','&city')
new 1: insert into suppliers values(10004, 'Reliance','Delhi')

1 row created.

SQL> /
Enter value for sid: 10005
Enter value for sname: Mahindra
Enter value for address: Mumbai
old 1: insert into suppliers values(&sid, '&sname','&city')
new 1: insert into suppliers values(10005, 'Mahindra','Mumbai')

1 row created.

SQL> select * from SUPPLIERS;

```
      SID SNAME              CITY
---------- -------------------- --------------------
    10001 Acme Widget         Bangalore
    10002 Johns               Kolkata
    10003 Vimal               Mumbai
    10004 Reliance            Delhi
```

SQL> commit;

Commit complete.

SQL> insert into PARTS values(&pid, '&pname','&color');

Enter value for pid: 20001

Enter value for pname: Book

Enter value for color: Red

old 1: insert into PARTS values(&pid, '&pname','&color')

new 1: insert into PARTS values(20001, 'Book','Red')

1 row created.

SQL> /

Enter value for pid: 20002

Enter value for pname: Pen

Enter value for color: Red

old 1: insert into PARTS values(&pid, '&pname','&color')

new 1: insert into PARTS values(20002, 'Pen','Red')

1 row created.

SQL> /
Enter value for pid: 20003
Enter value for pname: Pencil
Enter value for color: Green
old 1: insert into PARTS values(&pid, '&pname','&color')
new 1: insert into PARTS values(20003, 'Pencil','Green')

1 row created.

SQL> /
Enter value for pid: 20004
Enter value for pname: Mobile
Enter value for color: Green
old 1: insert into PARTS values(&pid, '&pname','&color')
new 1: insert into PARTS values(20004, 'Mobile','Green')

1 row created.

SQL> /
Enter value for pid: 20005
Enter value for pname: Charger
Enter value for color: Black
old 1: insert into PARTS values(&pid, '&pname','&color')
new 1: insert into PARTS values(20005, 'Charger','Black')

1 row created.

SQL> select * from PARTS;

```
       PID PNAME               COLOR
---------- -------------------- ----------
     20001 Book                 Red
     20002 Pen                  Red
     20003 Pencil               Green
     20004 Mobile               Green
     20005 Charger              Black
```

SQL> commit;

Commit complete.

SQL> insert into CATALOG values(&sid, '&pid','&cost');

Enter value for sid: 10001

Enter value for pid: 20001

Enter value for cost: 10

old 1: insert into CATALOG values(&sid, '&pid','&cost')

new 1: insert into CATALOG values(10001, '20001','10')

1 row created.

SQL> /

Enter value for sid: 10001

Enter value for pid: 20002

Enter value for cost: 10

old 1: insert into CATALOG values(&sid, '&pid','&cost')

new 1: insert into CATALOG values(10001, '20002','10')

1 row created.

SQL> /

```
Enter value for sid: 10001
Enter value for pid: 20003
Enter value for cost: 30
old   1: insert into CATALOG values(&sid, '&pid','&cost')
new   1: insert into CATALOG values(10001, '20003','30')

1 row created.

SQL> /
Enter value for sid: 10001
Enter value for pid: 20004
Enter value for cost: 10
old   1: insert into CATALOG values(&sid, '&pid','&cost')
new   1: insert into CATALOG values(10001, '20004','10')

1 row created.

SQL> /
Enter value for sid: 10001
Enter value for pid: 20005
Enter value for cost: 10
old   1: insert into CATALOG values(&sid, '&pid','&cost')
new   1: insert into CATALOG values(10001, '20005','10')

1 row created.

SQL> /
Enter value for sid: 10002
Enter value for pid: 20001
Enter value for cost: 10
old   1: insert into CATALOG values(&sid, '&pid','&cost')
```

new 1: insert into CATALOG values(10002, '20001','10')

1 row created.

SQL> /
Enter value for sid: 10002
Enter value for pid: 20002
Enter value for cost: 20
old 1: insert into CATALOG values(&sid, '&pid','&cost')
new 1: insert into CATALOG values(10002, '20002','20')

1 row created.

SQL> /
Enter value for sid: 10003
Enter value for pid: 20003
Enter value for cost: 30
old 1: insert into CATALOG values(&sid, '&pid','&cost')
new 1: insert into CATALOG values(10003, '20003','30')

1 row created.

SQL> /
Enter value for sid: 10004
Enter value for pid: 20003
Enter value for cost: 40
old 1: insert into CATALOG values(&sid, '&pid','&cost')
new 1: insert into CATALOG values(10004, '20003','40')

1 row created.

SQL> select * from CATALOG;

SID	PID	COST
10001	20001	10
10001	20002	10
10001	20003	30
10001	20004	10
10001	20005	10
10002	20001	10
10002	20002	20
10003	20003	30
10004	20003	40

9 rows selected.

i) <u>Find the pnames of parts for which there is some supplier.</u>

SQL> SELECT DISTINCT P.pname
2 FROM Parts P, Catalog C
3 WHERE P.pid = C.pid;

PNAME

Book
Charger
Mobile
Pen
Pencil

ii) Find the snames of suppliers who supply every part.

```
SQL> SELECT S.sname
2  FROM Suppliers S
3  WHERE NOT EXISTS ((SELECT P.pid  FROM Parts P)
 4        MINUS (SELECT C.pid FROM Catalog C
 5        WHERE C.sid = S.sid));
```

SNAME

Acme Widget

iii) Find the snames of suppliers who supply every red part.

```
SQL>SELECT S.sname
FROM Suppliers S
WHERE NOT EXISTS (( SELECT P.pid
                    FROM Parts P
                    WHERE P.color = 'Red' )
                    MINUS
                    ( SELECT C.pid
                    FROM Catalog C, Parts P
                    WHERE C.sid = S.sid AND
                    C.pid = P.pid AND P.color = 'Red' ));
```

SNAME

Acme Widget

Johns

iv) Find the pnames of parts supplied by Acme Widget Suppliers and by no one else.

```
SQL> select pname from parts where pid in (select pid from cataloge where sid =(
select sid from suppliers where sname='Acme widget') minus select pid from catal
oge where sid in (select sid from suppliers where sname <>'Acme widget'));

PNAME
---------------
Mobile
Charger
```

PNAME

Mobile

Charger

v) **Find the sids of suppliers who charge more for some part than the average cost of that part (averaged over all the suppliers who supply that part).**

```
SQL> SELECT DISTINCT C.sid FROM Catalog C
2  WHERE C.cost > ( SELECT AVG (C1.cost)
3  FROM Catalog C1
4  WHERE C1.pid = C.pid );
```

SID

10002

10004

vi) **For each part, find the sname of the supplier who charges the most for that part.**

```
SQL>SELECT P.pid, S.sname
FROM Parts P, Suppliers S, Catalog C
WHERE C.pid = P.pid
AND C.sid = S.sid
AND C.cost = (SELECT MAX (C1.cost)
              FROM Catalog C1
              WHERE C1.pid = P.pid);
```

PID SNAME

--- ---

20001 Acme Widget

20004 Acme Widget

20005 Acme Widget

20001 Johns

20002 Johns

20003 Reliance

PROGRAM 4: STUDENT FACULTY DATABASE

Consider the following database for student enrollment for course :

STUDENT(snum: integer, sname: string, major: string, lvl: string, age: integer)

CLASS(cname: string, meets at: time, room: string, fid: integer)

ENROLLED(snum: integer, cname: string)

FACULTY(fid: integer, fname: string, deptid: integer)

The meaning of these relations is straightforward; for example, Enrolled has one record per student-class pair such that the student is enrolled in the class. Level(lvl) is a two character code with 4 different values (example: Junior: JR etc)

Write the following queries in SQL. No duplicates should be printed in any of the answers.

i. Find the names of all Juniors (level = JR) who are enrolled in a class taught by

ii. Find the names of all classes that either meet in room R128 or have five or more Students enrolled.

iii. Find the names of all students who are enrolled in two classes that meet at the same time.

iv. Find the names of faculty members who teach in every room in which some class is taught.

v. Find the names of faculty members for whom the combined enrollment of the courses that they teach is less than five.

vi. Find the names of students who are not enrolled in any class.

vii. For each age value that appears in Students, find the level value that appears most often. For example, if there are more FR level students aged 18 than SR, JR, or SO students aged 18, you should print the pair (18, FR).

```
SQL> CREATE TABLE student(
  2  snum INT,
  3  sname VARCHAR(10),
  4  major VARCHAR(2),
  5  lvl VARCHAR(2),
  6  age INT, primary key(snum));
```

Table created.

SQL> desc student;

Name	Null?	Type
SNUM	NOT NULL	NUMBER(38)
SNAME		VARCHAR2(10)
MAJOR		VARCHAR2(2)
LVL		VARCHAR2(2)
AGE		NUMBER(38)

```
SQL> CREATE TABLE faculty(
  2  fid INT,fname VARCHAR(20),
  3  deptid INT,
  4  PRIMARY KEY(fid));
```

Table created.

SQL> desc faculty;

Name	Null?	Type
FID	NOT NULL	NUMBER(38)
FNAME		VARCHAR2(20)
DEPTID		NUMBER(38)

```
SQL>  CREATE TABLE class(
  2  cname VARCHAR(20),
  3  metts_at TIMESTAMP,
  4  room VARCHAR(10),
5  fid INT,
  6  PRIMARY KEY(cname),
  7  FOREIGN KEY(fid) REFERENCES faculty(fid));
```

Table created.

SQL> DESC class;

Name	Null?	Type
CNAME	NOT NULL	VARCHAR2(20)
METTS_AT		TIMESTAMP(6)
ROOM		VARCHAR2(10)
FID		NUMBER(38)

```
SQL> CREATE TABLE enrolled(
  2  snum INT,
  3  cname VARCHAR(20),
  4  PRIMARY KEY(snum,cname),
  5  FOREIGN KEY(snum) REFERENCES student(snum),
```

6 FOREIGN KEY(cname) REFERENCES class(cname));

Table created.

SQL> desc enrolled;

Name	Null?	Type
SNUM	NOT NULL	NUMBER(38)
CNAME	NOT NULL	VARCHAR2(20)

SQL> commit;

Commit complete.

INSERTION OF VALUES:

SQL> INSERT INTO STUDENT VALUES(&snum, '&sname', '&major', '&lvl', &age);
Enter value for snum: 1
Enter value for sname: jhon
Enter value for major: CS
Enter value for lvl: Sr
Enter value for age: 19
old 1: INSERT INTO STUDENT VALUES(&snum, '&sname', '&major', '&lvl', &age)
new 1: INSERT INTO STUDENT VALUES(1, 'jhon', 'CS', 'Sr', 19)

1 row created.

SQL> /
Enter value for snum: 2
Enter value for sname: Smith
Enter value for major: CS
Enter value for lvl: Jr
Enter value for age: 20
old 1: INSERT INTO STUDENT VALUES(&snum, '&sname', '&major', '&lvl', &age)
new 1: INSERT INTO STUDENT VALUES(2, 'Smith', 'CS', 'Jr', 20)

1 row created.

SQL> /
Enter value for snum: 3
Enter value for sname: Jacob
Enter value for major: CV
Enter value for lvl: Sr

```
Enter value for age: 20
old   1: INSERT INTO STUDENT VALUES(&snum, '&sname', '&major', '&lvl', &age)
new   1: INSERT INTO STUDENT VALUES(3 , 'Jacob', 'CV', 'Sr', 20)

1 row created.

SQL> /
Enter value for snum: 4
Enter value for sname: Tom
Enter value for major: CS
Enter value for lvl: Jr
Enter value for age: 20
old   1: INSERT INTO STUDENT VALUES(&snum, '&sname', '&major', '&lvl', &age)
new   1: INSERT INTO STUDENT VALUES(4, 'Tom ', 'CS', 'Jr', 20)

1 row created.

SQL> /
Enter value for snum: 5
Enter value for sname: Rahul
Enter value for major: CS
Enter value for lvl: Jr
Enter value for age: 20
old   1: INSERT INTO STUDENT VALUES(&snum, '&sname', '&major', '&lvl', &age)
new   1: INSERT INTO STUDENT VALUES(5, 'Rahul', 'CS', 'Jr', 20)

1 row created.

SQL> /
Enter value for snum: 6
Enter value for sname: Rita
Enter value for major: CS
Enter value for lvl: Sr
Enter value for age: 21
old   1: INSERT INTO STUDENT VALUES(&snum, '&sname', '&major', '&lvl', &age)
new   1: INSERT INTO STUDENT VALUES(6, 'Rita', 'CS', 'Sr', 21)

1 row created.
```

SQL> select * from student;

```
      SNUM SNAME      MA LV        AGE
---------- ---------- -- -- ----------
         1 jhon       CS Sr         19
         2 Smith      CS Jr         20
         3 Jacob      CV Sr         20
```

```
4 Tom       CS Jr       20
5 Rahul     CS Jr       20
6 Rita      CS Sr       21
```

6 rows selected.

SQL> INSERT INTO FACULTY VALUES(&FID, '&FNAME', &DEPTID);

```
Enter value for fid: 11
Enter value for fname: Harish
Enter value for deptid: 1000
old   1: INSERT INTO FACULTY VALUES(&FID, '&FNAME', &DEPTID)
new   1: INSERT INTO FACULTY VALUES(11, 'Harish', 1000)
```

1 row created.

```
SQL> /
Enter value for fid: 12
Enter value for fname: MV
Enter value for deptid: 1000
old   1: INSERT INTO FACULTY VALUES(&FID, '&FNAME', &DEPTID)
new   1: INSERT INTO FACULTY VALUES(12, 'MV', 1000)
```

1 row created.

```
SQL> /
Enter value for fid: 13
Enter value for fname: Mira
Enter value for deptid: 1001
old   1: INSERT INTO FACULTY VALUES(&FID, '&FNAME', &DEPTID)
new   1: INSERT INTO FACULTY VALUES(13 , 'Mira', 1001)
```

1 row created.

```
SQL> /
Enter value for fid: 14
Enter value for fname: Shiva
Enter value for deptid: 1002
old   1: INSERT INTO FACULTY VALUES(&FID, '&FNAME', &DEPTID)
new   1: INSERT INTO FACULTY VALUES(14, 'Shiva', 1002)
```

1 row created.

```
SQL> /
Enter value for fid: 15
Enter value for fname: Nupur
Enter value for deptid: 1000
```

old 1: INSERT INTO FACULTY VALUES(&FID, '&FNAME', &DEPTID)
new 1: INSERT INTO FACULTY VALUES(15, 'Nupur', 1000)

1 row created.

SQL> commit;

Commit complete.

SQL> select * from faculty;

FID	FNAME	DEPTID
11	Harish	1000
12	MV	1000
13	Mira	1001
14	Shiva	1002
15	Nupur	1000

SQL> commit;

Commit complete.

SQL> alter session set nls_timestamp_format = 'RR/MM/DD HH24:MI:SSXFF';

Session altered.

SQL> alter session set nls_date_language ='ENGLISH';

Session altered.

SQL> insert into class values('&cname', '&meets_at', '&room', &fid);
Enter value for cname: class1
Enter value for meets_at: 12/11/15 10:15:16
Enter value for room: R1
Enter value for fid: 14
old 1: insert into class values('&cname', '&meets_at', '&room', &fid)
new 1: insert into class values('class1', '12/11/15 10:15:16', 'R1', 14)

1 row created.

Enter value for cname: class10
Enter value for meets_at: 12/11/15 10:15:16
Enter value for room: R128
Enter value for fid: 14

```
old   1: insert into class values('&cname', '&meets_at', '&room', &fid)
new   1: insert into class values('class10', '12/11/15 10:15:16', 'R128', 14)

1 row created.

SQL> /
Enter value for cname: class2
Enter value for meets_at: 12/11/15 10:15:20
Enter value for room: R2
Enter value for fid: 12
old   1: insert into class values('&cname', '&meets_at', '&room', &fid)
new   1: insert into class values('class2', '12/11/15 10:15:20', 'R2', 12)

1 row created.

SQL> insert into class values('&cname', '&meets_at', '&room', &fid);
Enter value for cname: class3
Enter value for meets_at: 12/11/15 10:15:25
Enter value for room: R3
Enter value for fid: 11
old   1: insert into class values('&cname', '&meets_at', '&room', &fid)
new   1: insert into class values('class3', '12/11/15 10:15:25', 'R3', 12)

1 row created.

SQL> /
Enter value for cname: class4
Enter value for meets_at: 12/11/15 20:15:20
Enter value for room: R4
Enter value for fid: 14
old   1: insert into class values('&cname', '&meets_at', '&room', &fid)
new   1: insert into class values('class4', '12/11/15 20:15:20', 'R4', 14)

1 row created.

SQL> /
Enter value for cname: class5
Enter value for meets_at: 12/11/15 20:15:20
Enter value for room: R3
Enter value for fid: 15
old   1: insert into class values('&cname', '&meets_at', '&room', &fid)
new   1: insert into class values('class5', '12/11/15 20:15:20', 'R3', 15)

1 row created.

SQL> /
```

```
Enter value for cname: class6
Enter value for meets_at: 12/11/15 13:20:20
Enter value for room: R2
Enter value for fid: 14
old   1: insert into class values('&cname', '&meets_at', '&room', &fid)
new   1: insert into class values('class6', '12/11/15 13:20:20', 'R2', 14)

1 row created.

SQL> /
Enter value for cname: class7
Enter value for meets_at: 12/11/15 10:10:10
Enter value for room: R3
Enter value for fid: 14
old   1: insert into class values('&cname', '&meets_at', '&room', &fid)
new   1: insert into class values('class7', '12/11/15 10:10:10', 'R3', 14)

1 row created.

SQL> select * from class;

CNAME
--------------------
METTS_AT
---------------------------------------------------------------------------
ROOM            FID
---------- ----------
class1
12/11/15 10:15:16.000000
R1              14

class10
12/11/15 10:15:16.000000
R128            14

CNAME
--------------------
METTS_AT
---------------------------------------------------------------------------
ROOM            FID
---------- ----------

class2
12/11/15 10:15:20.000000
R2              12
```

class3
12/11/15 10:15:25.000000

CNAME

METTS_AT

ROOM FID
---------- ----------
R3 11

class4
12/11/15 20:15:20.000000
R4 14

class5

CNAME

METTS_AT

ROOM FID
---------- ----------
12/11/15 20:15:20.000000
R3 15

class6
12/11/15 13:20:20.000000
R2 14

CNAME

METTS_AT

ROOM FID
---------- ----------
class7
12/11/15 10:10:10.000000
R3 14

8 rows selected.

SQL> commit;

```
Commit complete.

SQL> insert into enrolled values(&snum, '&cname');
Enter value for snum: 1
Enter value for cname: class1
old   1: insert into enrolled values(&snum, '&cname')
new   1: insert into enrolled values(1, 'class1')

1 row created.

SQL> /
Enter value for snum: 2
Enter value for cname: class1
old   1: insert into enrolled values(&snum, '&cname')
new   1: insert into enrolled values(2, 'class1')

1 row created.

SQL> /
Enter value for snum: 3
Enter value for cname: class3
old   1: insert into enrolled values(&snum, '&cname')
new   1: insert into enrolled values(3, 'class3')

1 row created.

SQL> /
Enter value for snum: 4
Enter value for cname: class3
old   1: insert into enrolled values(&snum, '&cname')
new   1: insert into enrolled values(4, 'class3')

1 row created.

SQL> /
Enter value for snum: 5
Enter value for cname: class4
old   1: insert into enrolled values(&snum, '&cname')
new   1: insert into enrolled values(5, 'class4')

1 row created.

SQL> /
Enter value for snum: 1
Enter value for cname: class5
old   1: insert into enrolled values(&snum, '&cname')
```

```
new   1: insert into enrolled values(1, 'class5')

1 row created.

SQL> /
Enter value for snum: 2
Enter value for cname: class5
old   1: insert into enrolled values(&snum, '&cname')
new   1: insert into enrolled values(2, 'class5')

1 row created.

SQL> /
Enter value for snum: 3
Enter value for cname: class5
old   1: insert into enrolled values(&snum, '&cname')
new   1: insert into enrolled values(3, 'class5')

1 row created.

SQL> /
Enter value for snum: 4
Enter value for cname: class5
old   1: insert into enrolled values(&snum, '&cname')
new   1: insert into enrolled values(4, 'class5')

1 row created.

SQL> /
Enter value for snum: 5
Enter value for cname: class5
old   1: insert into enrolled values(&snum, '&cname')
new   1: insert into enrolled values(5, 'class5')

1 row created.
```

SQL> select * from enrolled;

```
      SNUM CNAME
---------- --------------------
         1 class1
         2 class1
         3 class3
         4 class3
         5 class4
```

1 class5
2 class5
3 class5
4 class5
5 class5

10 rows selected.

viii. Find the names of all Juniors (level(lvl) = Jr) who are enrolled in a class taught by Harish.

```
SELECT DISTINCT S.Sname
FROM Student S, Class C, Enrolled E, Faculty F
WHERE S.snum = E.snum AND E.cname = C.cname AND C.fid = F.fid AND
F.fname = 'Harish' AND S.lvl = 'Jr';
```

SNAME

Tom

ix. Find the names of all classes that either meet in room R128 or have five or more Students enrolled.

```
SQL>SELECT C.cname
FROM Class C
WHERE C.room = 'R128'
OR C.cname IN (SELECT E.cname
        FROM Enrolled E
        GROUP BY E.cname
        HAVING COUNT (*) >= 5);
```

CNAME

---

class10

class5

x. Find the names of all students who are enrolled in two classes that meet at the same time.

```
SQL>SELECT DISTINCT S.sname
FROM Student S
WHERE S.snum IN (SELECT E1.snum
                 FROM Enrolled E1, Enrolled E2, Class C1, Class C2
                 WHERE E1.snum = E2.snum AND E1.cname <> E2.cname
                 AND E1.cname = C1.cname
                 AND E2.cname = C2.cname AND C1.meets_at = C2.meets_at);
```

SNAME

Rahul

xi. Find the names of faculty members who teach in every room in which some class is taught.

```
SELECT DISTINCT F.fname
FROM Faculty F
WHERE NOT EXISTS ((SELECT C.roomFROM Class C )
                  MINUS
                  (SELECTC1.room
                  FROM Class C1
                  WHERE C1.fid = F.fid ));
```

FNAME

Shiva

xii. Find the names of faculty members for whom the combined enrollment of the courses that they teach is less than five.

```
SQL>SELECT DISTINCT F.fname
FROM Faculty F
WHERE 5 > (SELECT COUNT (E.snum)
           FROM Class C, Enrolled E
           WHERE C.cname = E.cname
           AND C.fid = F.fid)
```

```
FNAME
--------------------
Harish
MV
Mira
Shiva
```

xiii. Find the names of students who are not enrolled in any class.

```
SELECT DISTINCT S.sname
FROM Student S
WHERE S.snum NOT IN (SELECT E.snum
                     FROM Enrolled E );
```

```
SNAME

----------

Rita
```

xiv. For each age value that appears in Students, find the level value that appears most often. For example, if there are more FR level students aged 18 than SR, JR, or SO students aged 18, you should print the pair (18, FR).

```
SELECT S.age, S.lvl
FROM Student S
```

```
GROUP BY S.age, S.lvl
HAVING S.lvl IN (SELECT S1.lvl FROM Student S1
                 WHERE S1.age = S.age
                 GROUP BY S1.lvl, S1.age
                 HAVING COUNT (*) >= ALL (SELECT COUNT (*)
                                          FROM Student S2
                                          WHERE s1.age = S2.age
                                          GROUP BY S2.lvl, S2.age));
```

```
       AGE LV
---------- --
        19 Sr
        20 Jr
        21 Sr
```

PROGRAM 5: AIRLINE FLIGHT DATABASE

Consider the following database that keeps track of airline flight information:

FLIGHTS(flno: integer, from: string, to: string, distance: integer, departs: time, arrives: time, price: integer)

AIRCRAFT(aid: integer, aname: string, cruisingrange: integer)

CERTIFIED(eid: integer, aid: integer)

EMPLOYEES(eid: integer, ename: string, salary: integer)

Note that the Employees relation describes pilots and other kinds of employees as well; Every pilot is certified for some aircraft, and only pilots are certified to fly.

Write each of the following queries in SQL.

i. Find the names of aircraft such that all pilots certified to operate them have salaries more than Rs.80,000.

ii. For each pilot who is certified for more than three aircrafts, find the eid and the maximum cruisingrange of the aircraft for which she or he is certified.

iii. Find the names of pilots whose salary is less than the price of the cheapest route from Bengaluru to Frankfurt.

iv. For all aircraft with cruisingrange over 1000 Kms, find the name of the aircraft and the average salary of all pilots certified for this aircraft.

v. Find the names of pilots certified for some Boeing aircraft.

vi. Find the aids of all aircraft that can be used on routes from Bengaluru to New Delhi.

vii. A customer wants to travel from Madison to New York with no more than two changes of flight. List the choice of departure times from Madison if the customer wants to arrive in New York by 6 p.m.

CREATION OF TABLES:

```
SQL> CREATE TABLE FLIGHTS
2  (FLNO INTEGER PRIMARY KEY,
3  FFROM VARCHAR(15) NOT NULL,
4  TTO VARCHAR(15) NOT NULL,
5  DISTANCE INTEGER,
```

```
6  DEPARTS TIMESTAMP,
7  ARRIVES TIMESTAMP,
8  PRICE NUMBER(10,2));
```

Table created.

```
SQL> DESC FLIGHTS;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 FLNO                                      NOT NULL NUMBER(38)
 FFROM                                     NOT NULL VARCHAR2(15)
 TTO                                       NOT NULL VARCHAR2(15)
 DISTANCE                                           NUMBER(38)
 DEPARTS                                            TIMESTAMP(6)
 ARRIVES                                            TIMESTAMP(6)
 PRICE                                              NUMBER(10,2)
```

```
SQL> CREATE TABLE AIRCRAFT
2  (AID INTEGER PRIMARY KEY,
3  ANAME VARCHAR(10),
4  CRUISINGRANGE INTEGER);
```

Table created.

```
SQL> DESC AIRCRAFT;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 AID                                       NOT NULL NUMBER(38)
 ANAME                                              VARCHAR2(10)
 CRUISINGRANGE                                      NUMBER(38)
```

```
SQL> CREATE TABLE EMPLOYEES
2  (EID INTEGER PRIMARY KEY,
3  ENAME VARCHAR(15),
4  SALARY NUMBER(10,2));
```

Table created.

```
SQL> DESC EMPLOYEES;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 EID                                       NOT NULL NUMBER(38)
 ENAME                                              VARCHAR2(15)
 SALARY                                             NUMBER(10,2)
```

```
SQL> CREATE TABLE CERTIFIED
```

2 (EID INTEGER NOT NULL,
3 AID INTEGER NOT NULL,
4 PRIMARY KEY (EID, AID),
5 FOREIGN KEY (EID) REFERENCES EMPLOYEES (EID),
6 FOREIGN KEY (AID) REFERENCES AIRCRAFT (AID));

Table created.

SQL> DESC CERTIFIED;

Name	Null?	Type
EID	NOT NULL	NUMBER(38)
AID	NOT NULL	NUMBER(38)

SQL> COMMIT;

Commit complete.

INSERTION OF VALUES:

INSERT IN TO AIRCRAFT VALUES::

SQL> insert into aircraft values(101,'747',3000);

1 row created.

SQL> insert into aircraft values(102,'Boeing',900);

1 row created.

SQL> insert into aircraft values(103,'647',800);

1 row created.

SQL> insert into aircraft values(104,'Dreamliner',10000);

1 row created.

SQL> insert into aircraft values(105,'Boeing',3500);

1 row created.

SQL> insert into aircraft values(106,'707',1500);

1 row created.
SQL> insert into aircraft values(107,'Dream', 120000);

1 row created.

INSERT INTO EMPLOYEES TABLE:

SQL> insert into employees values(701,'A',50000);

1 row created.

SQL> insert into employees values(702,'B',100000);

1 row created.

SQL> insert into employees values(703,'C',150000);

1 row created.

SQL> insert into employees values(704,'D',90000);

1 row created.

SQL> insert into employees values(705,'E',40000);

1 row created.

SQL> insert into employees values(706,'F',60000);

1 row created.

SQL> insert into employees values(707,'G',90000);

1 row created.

INSERT INTO CERTIFIED TABLE:

SQL> insert into certified values(701,101);

1 row created.

SQL> insert into certified values(701,102);

1 row created.

SQL> insert into certified values(701,106);

1 row created.

SQL> insert into certified values(701,105);

1 row created.

SQL> insert into certified values(702,104);

1 row created.

SQL> insert into certified values(703,104);

1 row created.

SQL> insert into certified values(704,104);

1 row created.

SQL> insert into certified values(702,107);

1 row created.

SQL> insert into certified values(703,107);

1 row created.

SQL> insert into certified values(704,107);

1 row created.

SQL> insert into certified values(702,101);

1 row created.

SQL> insert into certified values(703,105);

1 row created.

SQL> insert into certified values(704,105);

1 row created.

SQL> insert into certified values(705,103);

1 row created.

SQL> alter session set nls_timestamp_format = 'RR/MM/DD HH24:MI:SSXFF';

Session altered.

SQL> alter session set nls_date_language ='ENGLISH';

Session altered.

INSERT INTO FLIGHTS Table:

SQL> insert into flights values(101,'Bangalore','Delhi'**,2500,TIMESTAMP '2005-05-13 07:15:31',TIMESTAMP '2005-05-13 17:15:31',5000);**

1 row created.

SQL> insert into flights values(102,'Bangalore','Lucknow',3000,TIMESTAMP '2005-05-13 07:15:31',TIMESTAMP '2005-05-13 11:15:31',6000);

1 row created.

SQL> insert into flights values(103,'Lucknow','Delhi'**,500,TIMESTAMP '2005-05-13 12:15:31',TIMESTAMP ' 2005-05-13 17:15:31',3000);**

1 row created.

SQL> insert into flights values(107,'Bangalore','Frankfurt',8000,TIMESTAMP '2005-05-13 07:15:31',TIMESTAMP '2005-05-13 22:15:31',60000);

1 row created.

SQL> insert into flights values(104,'Bangalore','Frankfurt',8500,TIMESTAMP '2005-05-13 07:15:31',TIMESTAMP '2005-05-13 23:15:31',75000);

1 row created.

SQL> insert into flights values(105,'Kolkata','Delhi'**,3400,TIMESTAMP '2005-05-13 07:15:31',TIMESTAMP '2005-05-13 09:15:31',7000);**

1 row created.

SQL> select * from Flights;

```
     FLNO FFROM          TTO             DISTANCE
---------- --------------- --------------- ----------
DEPARTS
```

ARRIVES

PRICE
---
101 Bangalore Delhi 2500
13-MAY-05 07.15.31.000000 AM
13-MAY-05 07.15.31.000000 AM
5000

FLNO FFROM TTO DISTANCE
--- --- --- ---
DEPARTS

ARRIVES

PRICE
---
102 Bangalore Lucknow 3000
13-MAY-05 07.15.
FLNO FFROM TTO DISTANCE
--- --- --- ---
DEPARTS

ARRIVES

PRICE
---
101 Bangalore Delhi 2500
05/05/13 07:15:31.000000
05/05/13 17:15:31.000000
5000

FLNO FFROM TTO DISTANCE
--- --- --- ---
DEPARTS

ARRIVES

PRICE
---
102 Bangalore Lucknow 3000
05/05/13 07:15:31.000000
05/05/13 11:15:31.000000

```
      6000


     FLNO FFROM          TTO             DISTANCE
---------- --------------- --------------- ----------
DEPARTS
-------------------------------------------------------------------------------
ARRIVES
-------------------------------------------------------------------------------
    PRICE
----------
       103 Lucknow         Delhi                500
05/05/13 12:15:31.000000
05/05/13 17:15:31.000000
      3000


     FLNO FFROM          TTO             DISTANCE
---------- --------------- --------------- ----------
DEPARTS
-------------------------------------------------------------------------------
ARRIVES
-------------------------------------------------------------------------------
    PRICE
----------
       107 Bangalore       Frankfurt            8000
05/05/13 07:15:31.000000
05/05/13 22:15:31.000000
     60000


     FLNO FFROM          TTO             DISTANCE
---------- --------------- --------------- ----------
DEPARTS
-------------------------------------------------------------------------------
ARRIVES
-------------------------------------------------------------------------------
    PRICE
----------
       104 Bangalore       Frankfurt            8500
05/05/13 07:15:31.000000
05/05/13 23:15:31.000000
     75000


     FLNO FFROM          TTO             DISTANCE
```

---------- --------------- --------------- ----------
DEPARTS

ARRIVES

PRICE
---
105 Kolkata Delhi 3400
05/05/13 07:15:31.000000
05/05/13 09:15:31.000000
7000

6 rows selected.

SQL> select * from Aircraft;

AID ANAME CRUISINGRANGE
---------- ---------- -------------
101 747 3000
102 Boeing 900
103 647 800
104 Dreamliner 10000
105 Boeing 3500
106 707 1500
107 Dream 120000

7 rows selected.

SQL> select * from Certified;

EID AID
---------- ----------
701 101
701 102
701 106
701 105
702 104
703 104
704 104
702 107
703 107
704 107
702 101

EID AID

```
---------- ----------
       703        105
       704        105
       705        103
```

14 rows selected.

SQL> select * from Employees;

```
       EID ENAME            SALARY
---------- --------------- ----------
       701 A                50000
       702 B               100000
       703 C               150000
       704 D                90000
       705 E                40000
       706 F                60000
       707 G                90000
```

7 rows selected.

viii. Find the names of aircraft such that all pilots certified to operate them have salaries more than Rs.80,000.

```
SQL>SELECT DISTINCT A.aname
FROM Aircraft A
WHERE A.Aid IN (SELECT C.aid
 FROM Certified C, Employees E
 WHERE C.eid = E.eid AND
 NOT EXISTS ( SELECT *
 FROM Employees E1
 WHERE E1.eid = E.eid AND E1.salary <80000 ));
```

```
ANAME
----------
747
Boeing
Dream
Dreamliner
```

ix. For each pilot who is certified for more than three aircrafts, find the eid and the maximum cruising range of the aircraft for which she or he is certified.

```
SQL>SELECT C.eid, MAX (A.cruisingrange)
FROM Certified C, Aircraft A
WHERE C.aid = A.aid
```

```
GROUP BY C.eid
HAVING COUNT (*) > 3;
```

```
     EID MAX(A.CRUISINGRANGE)
---------- --------------------
     701              3500
```

x. Find the names of pilots whose salary is less than the price of the cheapest route from Bangalore to Frankfurt.

```
SELECT DISTINCT E.ename
FROM Employees E
WHERE E.salary <( SELECT MIN(F.price)
                  FROM Flights F
                  WHERE F.ffrom = 'Bangalore' AND F.tto = 'Frankfurt' );
```

```
ENAME
---------------
A
E
```

xi. For all aircraft with cruising range over 1000 Kms, find the name of the aircraft and the average salary of all pilots certified for this aircraft.

```
SELECT Temp.name, Temp.AvgSalary
FROM ( SELECT A.aid, A.aname AS name, AVG (E.salary) AS AvgSalary
 FROM Aircraft A, Certified C, Employees E
 WHERE A.aid = C.aid AND C.eid = E.eid AND A.cruisingrange > 1000
 GROUP BY A.aid, A.aname )  Temp;
```

```
NAME       AVGSALARY
---------- ----------
747           75000
Dreamliner 113333.333
Boeing     96666.6667
707           50000
Dream      113333.333
```

xii. Find the names of pilots certified for some Boeing aircraft.

```
SELECT DISTINCT E.ename
FROM Employees E, Certified C, Aircraft A
WHERE E.eid = C.eid AND C.aid = A.aid AND A.aname LIKE 'Boeing%';
```

```
ENAME
---------------
A
C
D
```

xiii. Find the aids of all aircraft that can be used on routes from Bangalore to Frankfurt.

```
SELECT A.aid
FROM Aircraft A
WHERE A.cruisingrange >( SELECT MIN (F.distance)
                         FROM Flights F
                         WHERE F.ffrom = 'Bangalore' AND F.tto = 'Frankfurt' );
```

```
    AID
----------
    104
    107
```

xiv. A customer wants to travel from Bangalore to Delhi with no more than two changes of flight. List the choice of departure times from Bangalore if the customer wants to arrive in Delhi by 6 p.m.

```
SELECT F.departs
FROM Flights F
WHERE F.flno IN ( ( SELECT F0.flno
 FROM Flights F0
 WHERE F0.ffrom = 'Bangalore' AND F0.tto = 'Delhi'
 AND extract(hour from F0.arrives) < 18 )
 UNION
( SELECT F0.flno
 FROM Flights F0, Flights F1
 WHERE F0.ffrom = 'Bangalore' AND F0.tto <> 'Delhi'
 AND F0.tto = F1.ffrom AND F1.tto = 'Delhi'
 AND F1.departs > F0.arrives
 AND extract(hour from F1.arrives) < 18)
 UNION
( SELECT F0.flno
 FROM Flights F0, Flights F1, Flights F2
 WHERE F0.ffrom = 'Bangalore'
 AND F0.tto = F1.ffrom
 AND F1.tto = F2.ffrom
 AND F2.tto = 'Delhi'
 AND F0.tto <> 'Delhi'
 AND F1.tto <> 'Delhi'
```

```
 AND F1.departs > F0.arrives
 AND F2.departs > F1.arrives
 AND extract(hour from F2.arrives) < 18));
```

```
DEPARTS
---------------------------------------------------------------------------
05/05/13 07:15:31.000000
05/05/13 07:15:31.000000
```

xv. Print the name and salary of every non-pilot whose salary is more than the average salary for pilots.

```
SELECT E.ename, E.salary
FROM Employees E
WHERE E.eid NOT IN ( SELECT DISTINCT C.eid
 FROM Certified C )
AND E.salary >( SELECT AVG (E1.salary)
 FROM Employees E1
 WHERE E1.eid IN
( SELECT DISTINCT C1.eid
 FROM Certified C1 ) );
```

```
ENAME           SALARY
        --------------- ----------
        G                90000
```

Program 6 : Order Database

Consider the following schema for Order Database:

SALESMAN (*Salesman_id, Name, City, Commission*)
CUSTOMER (*Customer_id, Cust_Name, City, Grade, Salesman_id*)
ORDERS (*Ord_No, Purchase_Amt, Ord_Date, Customer_id, Salesman_id*)
Write SQL queries to

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

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

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

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

5. Demonstrate the DELETE operation by removing salesman with id 1000. All his orders must also be deleted.

Schema Diagram

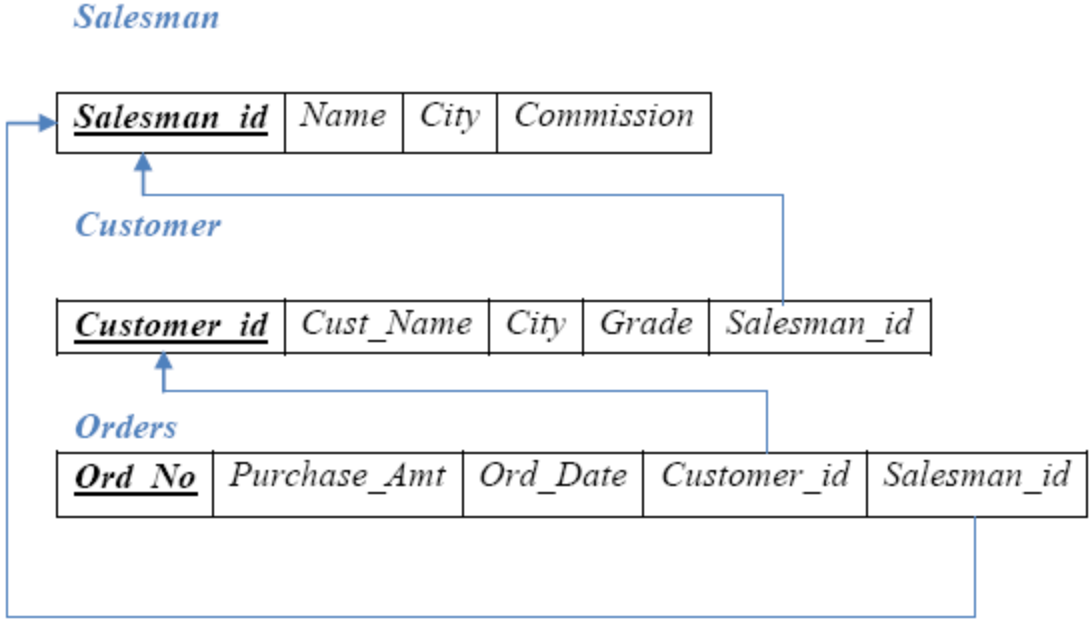


Table Creation

```
CREATE TABLE SALESMAN
(SALESMAN_ID NUMBER (4),
NAME VARCHAR2 (20),
CITY VARCHAR2 (20),
COMMISSION VARCHAR2 (20),
PRIMARY KEY (SALESMAN_ID));

CREATE TABLE CUSTOMER1
(CUSTOMER_ID NUMBER (4),
CUST_NAME VARCHAR2 (20),
CITY VARCHAR2 (20),
GRADE NUMBER (3),
PRIMARY KEY (CUSTOMER_ID),
SALESMAN_ID REFERENCES SALESMAN (SALESMAN_ID) ON DELETE SET NULL);



CREATE TABLE ORDERS
(ORD_NO NUMBER (5),
PURCHASE_AMT NUMBER (10, 2),
ORD_DATE DATE,
PRIMARY KEY (ORD_NO),
CUSTOMER_ID REFERENCES CUSTOMER1 (CUSTOMER_ID) ON DELETE CASCADE,
SALESMAN_ID REFERENCES SALESMAN (SALESMAN_ID) ON DELETE CASCADE);
```

Table Descriptions

```
SQL> DESC SALESMAN;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 SALESMAN_ID                               NOT NULL NUMBER(4)
 NAME                                               VARCHAR2(15)
 CITY                                               VARCHAR2(15)
 COMMISSION                                         NUMBER(3,2)


SQL> DESC CUSTOMER1;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 CUSTOMER_ID                               NOT NULL NUMBER(4)
 CUST_NAME                                          VARCHAR2(15)
 CITY                                               VARCHAR2(15)
 GRADE                                              NUMBER(3)
 SALESMAN_ID                                        NUMBER(4)
```

```
SQL> DESC ORDERS;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 ORD_NO                                    NOT NULL NUMBER(5)
 PURCHASE_AMT                                       NUMBER(10,2)
 ORD_DATE                                           DATE
 CUSTOMER_ID                                        NUMBER(4)
 SALESMAN_ID                                        NUMBER(4)
```

Insertion of Values to Tables

INSERT INTO SALESMAN VALUES (1000, _JOHN‘,‘BANGALORE‘,‘25 %‘);
INSERT INTO SALESMAN VALUES (2000, _RAVI‘,‘BANGALORE‘,‘20 %‘);
INSERT INTO SALESMAN VALUES (3000, _KUMAR‘,‘MYSORE‘,‘15 %‘);
INSERT INTO SALESMAN VALUES (4000, _SMITH‘,‘DELHI‘,‘30 %‘);
INSERT INTO SALESMAN VALUES (5000, _HARSHA‘,‘HYDRABAD‘,‘15 %‘);

INSERT INTO CUSTOMER1 VALUES (10, _PREETHI‘,‘BANGALORE‘, 100, 1000);
INSERT INTO CUSTOMER1 VALUES (11, _VIVEK‘,‘MANGALORE‘, 300, 1000);
INSERT INTO CUSTOMER1 VALUES (12, _BHASKAR‘,‘CHENNAI‘, 400, 2000);
INSERT INTO CUSTOMER1 VALUES (13, _CHETHAN‘,‘BANGALORE‘, 200, 2000);
INSERT INTO CUSTOMER1 VALUES (14, _MAMATHA‘,‘BANGALORE‘, 400, 3000);

INSERT INTO ORDERS VALUES (50, 5000, _04-MAY-17‘, 10, 1000);
INSERT INTO ORDERS VALUES (51, 450, _20-JAN-17‘, 10, 2000);
INSERT INTO ORDERS VALUES (52, 1000, _24-FEB-17‘, 13, 2000);
INSERT INTO ORDERS VALUES (53, 3500, _13-APR-17‘, 14, 3000);
INSERT INTO ORDERS VALUES (54, 550, _09-MAR-17‘, 12, 2000);

SELECT * FROM SALESMAN;

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

SELECT * FROM CUSTOMER1;

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

SELECT * FROM ORDERS;

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

Queries:

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

SELECT GRADE, COUNT (DISTINCT CUSTOMER_ID)
FROM CUSTOMER1
GROUP BY GRADE
HAVING GRADE > (SELECT AVG(GRADE)
FROM CUSTOMER1
WHERE CITY='BANGALORE');

GRADE	COUNT(DISTINCTCUSTOMER_ID)
300	1
400	2

2. Find the name and numbers of all salesmen who had more than one customer.
SELECT SALESMAN_ID, NAME
FROM SALESMAN A
WHERE 1 < (SELECT COUNT (*)
FROM CUSTOMER1
WHERE SALESMAN_ID=A.SALESMAN_ID);

SALESMAN_ID	NAME
1000	JOHN
2000	RAVI

3. List all salesmen and indicate those who have and don't have customers in their cities (Use UNION operation.)
SELECT SALESMAN.SALESMAN_ID, NAME, CUST_NAME, COMMISSION
FROM SALESMAN, CUSTOMER1
WHERE SALESMAN.CITY = CUSTOMER1.CITY
UNION

```
SELECT SALESMAN_ID, NAME, 'NO MATCH', COMMISSION
FROM SALESMAN
WHERE NOT CITY = ANY
(SELECT CITY
FROM CUSTOMER1)
ORDER BY 2 DESC;
```

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

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

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

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

5. Demonstrate the DELETE operation by removing salesman with id 1000. All his orders must also be deleted.

Use ON DELETE CASCADE at the end of foreign key definitions while creating child table orders and then execute the following:

Use ON DELETE SET NULL at the end of foreign key definitions while creating child table customers and then executes the following:

```
DELETE FROM SALESMAN
WHERE SALESMAN_ID=1000;
```

```
SQL> DELETE FROM SALESMAN
  2  WHERE SALESMAN_ID=1000;

1 row deleted.

SQL> SELECT * FROM SALESMAN;

SALESMAN_ID NAME                 CITY                 COMMISSION
----------- -------------------- -------------------- ----------------
       2000 RAVI                 BANGALORE            20 %
       3000 KUMAR                MYSORE               15 %
       4000 SMITH                DELHI                30 %
       5000 HARSHA               HYDRABAD             15 %
```

Program 7 : Book Database

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

Write SQL queries to

1. Retrieve details of all books in the library – id, title, name of publisher, authors, number of copies in each branch, 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 asimple query.
5. Create a view of all books and its number of copies that are currently available in the Library.

Schema Diagram

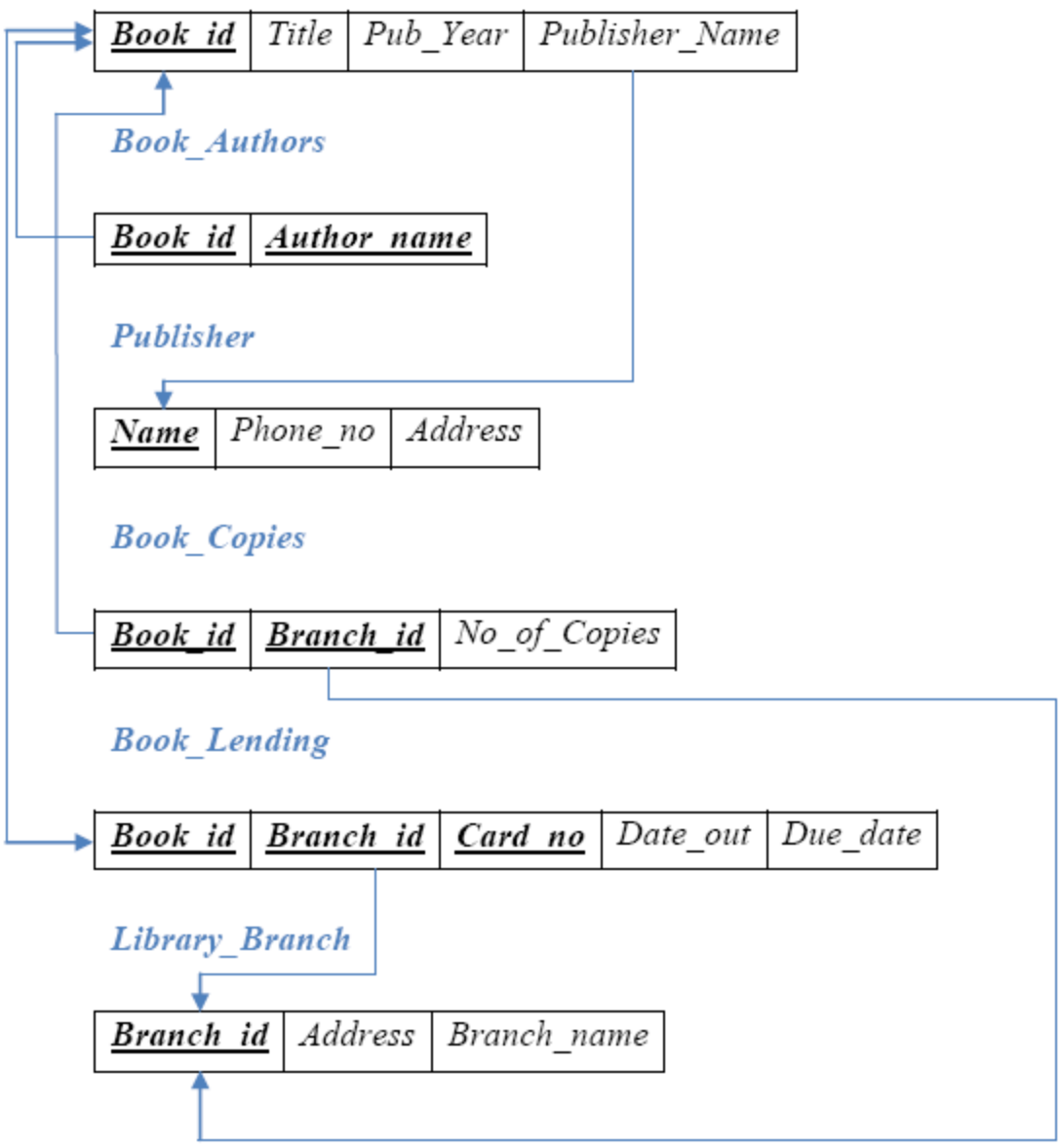


Table Creation

```
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));
```

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,‘RNSIT‘,‘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);


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
```

```
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


SQL> SELECT * FROM BOOK_AUTHORS;

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


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


SQL> SELECT * FROM BOOK_COPIES;

NO_OF_COPIES    BOOK_ID  BRANCH_ID
------------ ---------- ----------
          10          1         10
           5          1         11
           2          2         12
           5          2         13
           7          3         14
           1          5         10
           3          4         11

SQL> SELECT * FROM CARD;

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

```
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;
```

```
   BOOK_ID TITLE               PUBLISHER_NAME       AUTHOR_NAME          NO_OF_COPIES  BRANCH_ID
---------- ------------------- -------------------- -------------------- ------------ ----------
         1 DBMS                MCGRAW-HILL          NAVATHE                        10         10
         1 DBMS                MCGRAW-HILL          NAVATHE                         5         11
         2 ADBMS               MCGRAW-HILL          NAVATHE                         2         12
         2 ADBMS               MCGRAW-HILL          NAVATHE                         5         13
         3 CN                  PEARSON              TANENBAUM                       7         14
         5 OS                  PEARSON              GALVIN                          1         10
         4 CG                  GRUPO PLANETA        EDWARD ANGEL                    3         11
```

1. 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
```

2. 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
```

.

3. 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;

```
PUB_YEAR
-------------
JAN-2017
JUN-2016
SEP-2016
SEP-2015
MAY-2016
```

4. 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 8:

Consider the following database of student enrollment in courses & books adopted for each course.
STUDENT (regno: string, name: string, major: string, bdate:date)
COURSE (course #:int, cname:string, dept:string)
ENROLL (regno:string, course#:int, sem:int, marks:int)
BOOK _ ADOPTION (course# :int, sem:int, book-ISBN:int)
TEXT (book-ISBN:int, book-title:string, publisher:string, author:string)

Database applications laboratory GCEM DEPARTMENT OF CSE Page - 5 - 5th semester i. Create the above tables by properly specifying the primary keys and the foreign keys.
ii. Enter at least five tuples for each relation.
iii. Demonstrate how you add a new text book to the database and make this book be adopted by some department.
iv. Produce a list of text books (include Course #, Book-ISBN, Book-title) in the alphabetical order for courses offered by the 'CS' department that use more than two books.
v. List any department that has all its adopted books published by a specific publisher.
vi. Generate suitable reports.
vii. Create suitable front end for querying and displaying the results.

```
mysql> CREATE DATABASE books;
Query OK, 1 row affected (0.01 sec)
mysql> USE books;
Database changed
mysql> CREATE TABLE student(
regno VARCHAR(15),
name VARCHAR(20),
major VARCHAR(20),
bdate DATE,
     PRIMARY KEY (regno) );
```

```
CREATE TABLE course(
courseno INT,
cname VARCHAR(20),
dept VARCHAR(20),
     PRIMARY KEY (courseno) );
CREATE TABLE enroll(
regno VARCHAR(15),
courseno INT,
sem INT(3),
marks INT(4),
     PRIMARY KEY (regno,courseno),
```

```
     FOREIGN KEY (regno) REFERENCES student (regno),
     FOREIGN KEY (courseno) REFERENCES course (courseno) );

CREATE TABLE text(
book_isbn INT(5),
book_title VARCHAR(20),
publisher VARCHAR(20),
author VARCHAR(20),
     PRIMARY KEY (book_isbn) );


CREATE TABLE book_adoption(
courseno INT,
sem INT(3),
book_isbn INT(5),
     PRIMARY KEY (courseno,book_isbn),
     FOREIGN KEY (courseno) REFERENCES course (courseno),
     FOREIGN KEY (book_isbn) REFERENCES text(book_isbn) );

INSERT INTO student (regno,name,major,bdate) VALUES
     ('1pe11cs002','b','sr','19930924'),
     ('1pe11cs003','c','sr','19931127'),
     ('1pe11cs004','d','sr','19930413'),
     ('1pe11cs005','e','jr','19940824');



INSERT INTO course VALUES (111,'OS','CSE'),
     (112,'EC','CSE'),
     (113,'SS','ISE'),
     (114,'DBMS','CSE'),
     (115,'SIGNALS','ECE');

INSERT INTO text VALUES (book_isbn,book_title,publisher,author)
     (10,'DATABASE SYSTEMS','PEARSON','SCHIELD'),
     (900,'OPERATING SYS','PEARSON','LELAND'),
     (901,'CIRCUITS','HALL INDIA','BOB'),
     (902,'SYSTEM SOFTWARE','PETERSON','JACOB'),
     (903,'SCHEDULING','PEARSON','PATIL'),
     (904,'DATABASE SYSTEMS','PEARSON','JACOB'),
     (905,'DATABASE MANAGER','PEARSON','BOB'),
     (906,'SIGNALS','HALL INDIA','SUMIT');


INSERT INTO enroll (regno,courseno,sem,marks) VALUES
('1pe11cs001',115,3,100),
     ('1pe11cs002',114,5,100),
     ('1pe11cs003',113,5,100),
     ('1pe11cs004',111,5,100),
     ('1pe11cs005',112,3,100);
```

```
INSERT INTO book_adoption (courseno,sem,book_isbn) VALUES
(111,5,900),
(111,5,903),
(111,5,904),
(112,3,901),
(113,3,10),
(114,5,905),
(113,5,902),
(115,3,906);
```

Queries:

4. Produce a list of text books (include Course #, Book-ISBN, Book-title) in the alphabetical order for courses offered by the 'CS' department that use more than two books.

```
SELECT c.courseno,t.book_isbn,t.book_title
     FROM course c,book_adoption ba,text t
     WHERE c.courseno=ba.courseno
     AND ba.book_isbn=t.book_isbn
     AND c.dept='CSE'
     AND 2<(
     SELECT COUNT(book_isbn)
     FROM book_adoption b
     WHERE c.courseno=b.courseno)
     ORDER BY t.book_title;
```

```
----------+-----------+------------------+
| courseno | book_isbn | book_title       |
+----------+-----------+------------------+
|      111 |       904 | DATABASE SYSTEMS |
|      111 |       900 | OPERATING SYS    |
|      111 |       903 | SCHEDULING       |
+----------+-----------+----------------
```

5. List any department that has all its adopted books published by a specific publisher.

```
SELECT DISTINCT c.dept
     FROM course c
     WHERE c.dept IN
( SELECT c.dept
     FROM course c,book_adoption b,text t
     WHERE c.courseno=b.courseno
     AND t.book_isbn=b.book_isbn
     AND t.publisher='PEARSON')
     AND c.dept NOT IN
     (SELECT c.dept
     FROM course c,book_adoption b,text t
     WHERE c.courseno=b.courseno
     AND t.book_isbn=b.book_isbn
     AND t.publisher != 'PEARSON');
```

```
+------+
| dept |
+------+
| CSE  |
+------+
```

Program 9: Movie database

Consider the schema for Movie Database:

ACTOR (*Act_id, Act_Name, Act_Gender*)
DIRECTOR (*Dir_id, Dir_Name, Dir_Phone*)
MOVIES (*Mov_id, Mov_Title, Mov_Year, Mov_Lang, Dir_id*)
MOVIE_CAST (*Act_id, Mov_id, Role*)
RATING (*Mov_id, Rev_Stars*)
Write SQL queries to

1. List the titles of all movies directed by 'Hitchcock'.

2. Find the movie names where one or more actors acted in two or more movies.

3. List all actors who acted in a movie before 2000 and also in a movie after

2015 (use JOIN operation).

4. Find the title of movies and number of stars for each movie that has at least one rating and find the highest number of stars that movie received. Sort the result by movie title.

5. Update rating of all movies directed by 'Steven Spielberg' to 5.

Schema Diagram

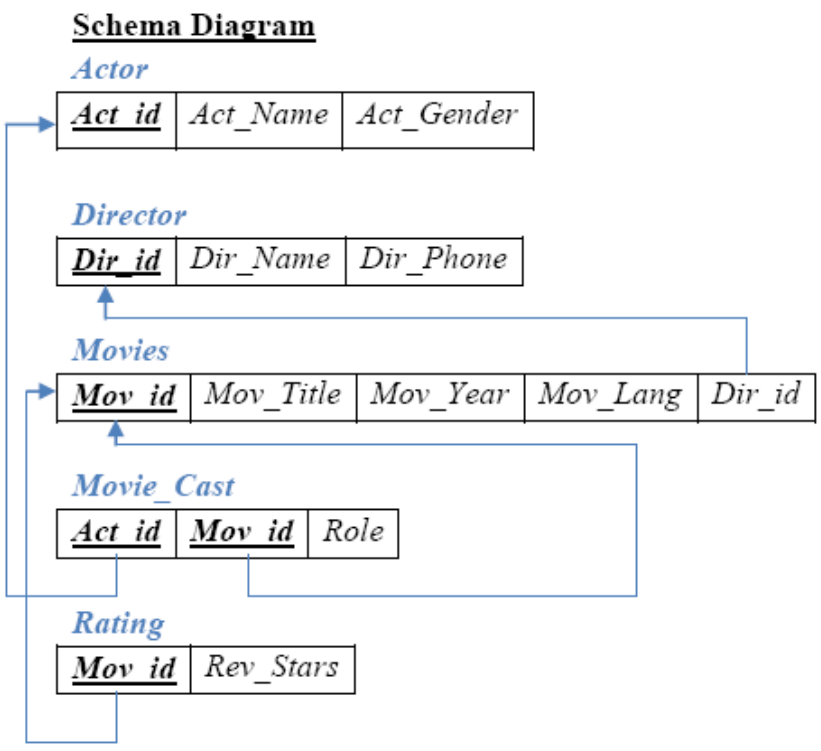


Table Creation

```
CREATE TABLE ACTOR (
ACT_ID NUMBER (3),
ACT_NAME VARCHAR (20),
ACT_GENDER CHAR (1),
PRIMARY KEY (ACT_ID));

CREATE TABLE DIRECTOR (
DIR_ID NUMBER (3),
DIR_NAME VARCHAR (20),
DIR_PHONE NUMBER (10),
```

```
PRIMARY KEY (DIR_ID));

CREATE TABLE MOVIES (
MOV_ID NUMBER (4),
MOV_TITLE VARCHAR (25),
MOV_YEAR NUMBER (4),
MOV_LANG VARCHAR (12),
DIR_ID NUMBER (3),
PRIMARY KEY (MOV_ID),
FOREIGN KEY (DIR_ID) REFERENCES DIRECTOR (DIR_ID));

CREATE TABLE MOVIE_CAST (
ACT_ID NUMBER (3),
MOV_ID NUMBER (4),
ROLE VARCHAR (10),
PRIMARY KEY (ACT_ID, MOV_ID),
FOREIGN KEY (ACT_ID) REFERENCES ACTOR (ACT_ID),
FOREIGN KEY (MOV_ID) REFERENCES MOVIES (MOV_ID));

CREATE TABLE RATING (
MOV_ID NUMBER (4),
REV_STARS VARCHAR (25),
PRIMARY KEY (MOV_ID),
FOREIGN KEY (MOV_ID) REFERENCES MOVIES (MOV_ID));
```

Table Descriptions

```
SQL> DESC ACTOR;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 ACT_ID                                    NOT NULL NUMBER(3)
 ACT_NAME                                           VARCHAR2(20)
 ACT_GENDER                                         CHAR(1)
```

```
SQL> DESC DIRECTOR;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 DIR_ID                                    NOT NULL NUMBER(3)
 DIR_NAME                                           VARCHAR2(20)
 DIR_PHONE                                          NUMBER(10)
```

```
SQL> DESC MOVIES;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 MOV_ID                                    NOT NULL NUMBER(4)
 MOV_TITLE                                          VARCHAR2(25)
 MOV_YEAR                                           NUMBER(4)
 MOV_LANG                                           VARCHAR2(12)
 DIR_ID                                             NUMBER(3)


SQL> DESC MOVIE_CAST;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 ACT_ID                                    NOT NULL NUMBER(3)
 MOV_ID                                    NOT NULL NUMBER(4)
 ROLE                                               VARCHAR2(10)


SQL> DESC RATING;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 MOV_ID                                    NOT NULL NUMBER(4)
 REV_STARS                                          VARCHAR2(25)
```

Insertion of Values to Tables

INSERT INTO ACTOR VALUES (301,‘ANUSHKA‘,‘F‘);
INSERT INTO ACTOR VALUES (302,‘PRABHAS‘,‘M‘);
INSERT INTO ACTOR VALUES (303,‘PUNITH‘,‘M‘);
INSERT INTO ACTOR VALUES (304,‘JERMY‘,‘M‘);

INSERT INTO DIRECTOR VALUES (60,‘RAJAMOULI‘, 8751611001);
INSERT INTO DIRECTOR VALUES (61,‘HITCHCOCK‘, 7766138911);
INSERT INTO DIRECTOR VALUES (62,‘FARAN‘, 9986776531);
INSERT INTO DIRECTOR VALUES (63,‘STEVEN SPIELBERG‘, 8989776530);

INSERT INTO MOVIES VALUES (1001,‘BAHUBALI-2‘, 2017, _TELAGU‘, 60);
INSERT INTO MOVIES VALUES (1002,‘BAHUBALI-1‘, 2015, _TELAGU‘, 60);
INSERT INTO MOVIES VALUES (1003,‘AKASH‘, 2008, _KANNADA‘, 61);
INSERT INTO MOVIES VALUES (1004,‘WAR HORSE‘, 2011, _ENGLISH‘, 63);

INSERT INTO MOVIE_CAST VALUES (301, 1002, _HEROINE‘);
INSERT INTO MOVIE_CAST VALUES (301, 1001, _HEROINE‘);
INSERT INTO MOVIE_CAST VALUES (303, 1003, _HERO‘);
INSERT INTO MOVIE_CAST VALUES (303, 1002, _GUEST‘);
INSERT INTO MOVIE_CAST VALUES (304, 1004, _HERO‘);

INSERT INTO RATING VALUES (1001, 4);
INSERT INTO RATING VALUES (1002, 2);

INSERT INTO RATING VALUES (1003, 5);
INSERT INTO RATING VALUES (1004, 4);

```
SQL> SELECT * FROM ACTOR;

    ACT_ID ACT_NAME             A
---------- -------------------- -
       301 ANUSHKA              F
       302 PRABHAS              M
       303 PUNITH               M
       304 JERMY                M
SQL> SELECT * FROM DIRECTOR;

    DIR_ID DIR_NAME              DIR_PHONE
---------- -------------------- ----------
        60 RAJAMOULI            8751611001
        61 HITCHCOCK            7766138911
        62 FARAN                9986776531
        63 STEVEN SPIELBERG     8989776530



SQL> SELECT * FROM MOVIES;

    MOV_ID MOV_TITLE                MOV_YEAR MOV_LANG         DIR_ID
---------- ---------------------- ---------- ------------ ----------
      1001 BAHUBALI-2                   2017 TELAGU               60
      1002 BAHUBALI-1                   2015 TELAGU               60
      1003 AKASH                        2008 KANNADA              61
      1004 WAR HORSE                    2011 ENGLISH              63


SQL> SELECT * FROM MOVIE_CAST;

    ACT_ID     MOV_ID ROLE
---------- ---------- ----------
       301       1002 HEROINE
       301       1001 HEROINE
       303       1003 HERO
       303       1002 GUEST
       304       1004 HERO



SQL> SELECT * FROM RATING;

    MOV_ID REV_STARS
---------- ------------------------
      1001 4
      1002 2
      1003 5
      1004 4
```

Queries:

1. List the titles of all movies directed by 'Hitchcock'.

```
SELECT MOV_TITLE
FROM MOVIES
WHERE DIR_ID IN (SELECT DIR_ID
FROM DIRECTOR
WHERE DIR_NAME = _HITCHCOCK');
```

```
MOV_TITLE
-------------------------
AKASH
```

2. Find the movie names where one or more actors acted in two or more movies.

```
SELECT MOV_TITLE
FROM MOVIES M, MOVIE_CAST MV
WHERE M.MOV_ID=MV.MOV_ID AND ACT_ID IN (SELECT ACT_ID
FROM MOVIE_CAST GROUP BY ACT_ID
HAVING COUNT (ACT_ID)>1)
GROUP BY MOV_TITLE
HAVING COUNT (*)>1;
```

```
MOV_TITLE
-------------------------
BAHUBALI-1
```

3. List all actors who acted in a movie before 2000 and also in a movie after 2015 (use JOIN operation).

```
SELECT ACT_NAME, MOV_TITLE, MOV_YEAR
FROM ACTOR A
JOIN MOVIE_CAST C
ON A.ACT_ID=C.ACT_ID
JOIN MOVIES M
ON C.MOV_ID=M.MOV_ID
WHERE M.MOV_YEAR NOT BETWEEN 2000 AND 2015;
```

(OR)

```
SELECT A.ACT_NAME, A.ACT_NAME, C.MOV_TITLE, C.MOV_YEAR
FROM ACTOR A, MOVIE_CAST B, MOVIES C
WHERE A.ACT_ID=B.ACT_ID
AND B.MOV_ID=C.MOV_ID
AND C.MOV_YEAR NOT BETWEEN 2000 AND 2015;
```

```
ACT_NAME             MOV_TITLE                     MOV_YEAR
-------------------- ------------------------- ----------
ANUSHKA              BAHUBALI-2                        2017
```

4. Find the title of movies and number of stars for each movie that has at least one rating and find the highest number of stars that movie received. Sort the result by movie title.

```
SELECT MOV_TITLE, MAX (REV_STARS)
FROM MOVIES
INNER JOIN RATING USING (MOV_ID)
GROUP BY MOV_TITLE
HAVING MAX (REV_STARS)>0
ORDER BY MOV_TITLE;
```

```
MOV_TITLE                 MAX(REV_STARS)
------------------------- -------------------------
AKASH                     5
BAHUBALI-1                2
BAHUBALI-2                4
WAR HORSE                 4
```

6. **Update rating of all movies directed by 'Steven Spielberg' to 5** KL

```
UPDATE RATING
SET REV_STARS=5
WHERE MOV_ID IN (SELECT MOV_ID FROM MOVIES
WHERE DIR_ID IN (SELECT DIR_ID
FROM DIRECTOR
WHERE DIR_NAME = _STEVEN SPIELBERG'));
```

```
SQL> SELECT * FROM RATING;

    MOV_ID REV_STARS
---------- -------------------------
      1001 4
      1002 2
      1003 5
      1004 5
```

Program 10

Consider the schema for College Database:

STUDENT (*USN, SName, Address, Phone, Gender*)
SEMSEC (*SSID, Sem, Sec*)
CLASS (*USN, SSID*)
SUBJECT (*Subcode, Title, Sem, Credits*)
IAMARKS (*USN, Subcode, SSID, Test1, Test2, Test3, FinalIA*)

Write SQL queries to

1. List all the student details studying in fourth semester 'C' section.

2. Compute the total number of male and female students in each semester and in each section.

3. Create a view of Test1 marks of student USN '1BI15CS101' in all subjects.

4. Categorize students based on the following criterion:

If FinalIA = 17 to 20 then CAT = 'Outstanding'
If FinalIA = 12 to 16 then CAT = 'Average'
If FinalIA< 12 then CAT = 'Weak'
Give these details only for 8th semester A, B, and C section students.

Schema Diagram

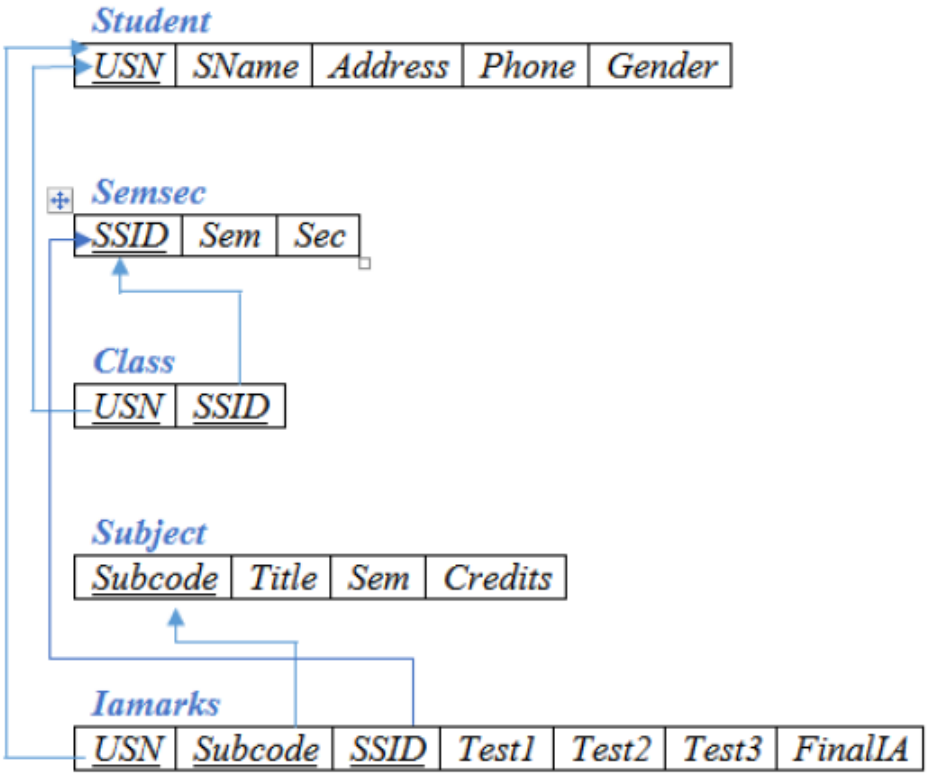


Table Creation

```
CREATE TABLE STUDENT (
USN VARCHAR (10) PRIMARY KEY,
```

```
SNAME VARCHAR (25),
ADDRESS VARCHAR (25),
PHONE NUMBER (10),
GENDER CHAR (1));

CREATE TABLE SEMSEC (
SSID VARCHAR (5) PRIMARY KEY,
SEM NUMBER (2),
SEC CHAR (1));

CREATE TABLE CLASS (
USN VARCHAR (10),
SSID VARCHAR (5),
PRIMARY KEY (USN, SSID),
FOREIGN KEY (USN) REFERENCES STUDENT (USN),
FOREIGN KEY (SSID) REFERENCES SEMSEC (SSID));

CREATE TABLE SUBJECT (
SUBCODE VARCHAR (8),
TITLE VARCHAR (20),
SEM NUMBER (2),
CREDITS NUMBER (2),
PRIMARY KEY (SUBCODE));

CREATE TABLE IAMARKS (
USN VARCHAR (10),
SUBCODE VARCHAR (8),
SSID VARCHAR (5),
TEST1 NUMBER (2),
TEST2 NUMBER (2),
TEST3 NUMBER (2),
FINALIA NUMBER (2),
PRIMARY KEY (USN, SUBCODE, SSID),
FOREIGN KEY (USN) REFERENCES STUDENT (USN),
FOREIGN KEY (SUBCODE) REFERENCES SUBJECT (SUBCODE),
FOREIGN KEY (SSID) REFERENCES SEMSEC (SSID));
```

Insertion of values to tables

```
INSERT INTO STUDENT VALUES ('1RN13CS020','AKSHAY','BELAGAVI', 8877881122,'M');
INSERT INTO STUDENT VALUES ('1RN13CS062','SANDHYA','BENGALURU',
7722829912,'F');
INSERT INTO STUDENT VALUES ('1RN13CS091','TEESHA','BENGALURU', 7712312312,'F');
INSERT INTO STUDENT VALUES ('1RN13CS066','SUPRIYA','MANGALURU',
8877881122,'F');
INSERT INTO STUDENTVALUES ('1RN14CS010','ABHAY','BENGALURU', 9900211201,'M');
```

```
INSERT INTO STUDENT VALUES ('1RN14CS032','BHASKAR','BENGALURU',
9923211099,'M');
INSERT INTO STUDENTVALUES ('1RN14CS025','ASMI','BENGALURU', 7894737377,'F');
INSERT INTO STUDENT VALUES ('1RN15CS011','AJAY','TUMKUR', 9845091341,'M');


INSERT INTO STUDENT VALUES ('1RN15CS029','CHITRA','DAVANGERE', 7696772121,'F');
INSERT INTO STUDENT VALUES ('1RN15CS045','JEEVA','BELLARY', 9944850121,'M');
INSERT INTO STUDENT VALUES ('1RN15CS091','SANTOSH','MANGALURU',
8812332201,'M');
INSERT INTO STUDENT VALUES ('1RN16CS045','ISMAIL','KALBURGI', 9900232201,'M');
INSERT INTO STUDENT VALUES ('1RN16CS088','SAMEERA','SHIMOGA', 9905542212,'F');
INSERT INTO STUDENT VALUES ('1RN16CS122','VINAYAKA','CHIKAMAGALUR',
8800880011,'M');



INSERT INTO SEMSEC VALUES ('CSE8A', 8,'A');
INSERT INTO SEMSEC VALUES (_CSE8B', 8,'B');
INSERT INTO SEMSEC VALUES (_CSE8C‘, 8,‘C‘);
INSERT INTO SEMSEC VALUES ('CSE7A', 7,‘A‘);
INSERT INTO SEMSEC VALUES (_CSE7B‘, 7,'B‘);
INSERT INTO SEMSEC VALUES ('CSE7C', 7,'C');
INSERT INTO SEMSEC VALUES (_CSE6A', 6,'A');
INSERT INTO SEMSEC VALUES (_CSE6B‘, 6,‘B‘);
INSERT INTO SEMSEC VALUES ('CSE6C‘, 6,‘C‘);
INSERT INTO SEMSEC VALUES (_CSE5A‘, 5,'A‘);
INSERT INTO SEMSEC VALUES ('CSE5B', 5,'B');
INSERT INTO SEMSEC VALUES (_CSE5C', 5,'C');
INSERT INTO SEMSEC VALUES (_CSE4A‘, 4,‘A‘);
INSERT INTO SEMSEC VALUES ('CSE4B', 4,‘B‘);
INSERT INTO SEMSEC VALUES (_CSE4C‘, 4,'C‘);
INSERT INTO SEMSEC VALUES ('CSE3A', 3,'A');
INSERT INTO SEMSEC VALUES (_CSE3B', 3,'B');
INSERT INTO SEMSEC VALUES (_CSE3C‘, 3,‘C‘);
INSERT INTO SEMSEC VALUES ('CSE2A', 2,‘A‘);
INSERT INTO SEMSEC VALUES (_CSE2B‘, 2,'B‘);
INSERT INTO SEMSEC VALUES ('CSE2C', 2,'C');
INSERT INTO SEMSEC VALUES (_CSE1A', 1,'A');
INSERT INTO SEMSEC VALUES (_CSE1B‘, 1,‘B‘);
INSERT INTO SEMSEC VALUES ('CSE1C', 1,‘C‘);

INSERT INTO CLASS VALUES (_1RN13CS020‘,‘CSE8A‘);
INSERT INTO CLASS VALUES (_1RN13CS062‘,‘CSE8A‘);
INSERT INTO CLASS VALUES (_1RN13CS066‘,‘CSE8B‘);
INSERT INTO CLASS VALUES (_1RN13CS091‘,‘CSE8C‘);
INSERT INTO CLASS VALUES (_1RN14CS010‘,‘CSE7A‘);
INSERT INTO CLASS VALUES (_1RN14CS025‘,‘CSE7A‘);
INSERT INTO CLASS VALUES (_1RN14CS032‘,‘CSE7A‘);
```

```
INSERT INTO CLASS VALUES (_1RN15CS011‘,‘CSE4A‘);
INSERT INTO CLASS VALUES (_1RN15CS029‘,‘CSE4A‘);
INSERT INTO CLASS VALUES (_1RN15CS045‘,‘CSE4B‘);
INSERT INTO CLASS VALUES (_1RN15CS091‘,‘CSE4C‘);
INSERT INTO CLASS VALUES (_1RN16CS045‘,‘CSE3A‘);
INSERT INTO CLASS VALUES (_1RN16CS088‘,‘CSE3B‘);
INSERT INTO CLASS VALUES (_1RN16CS122‘,‘CSE3C‘);


INSERT INTO SUBJECT VALUES ('10CS81','ACA', 8, 4);
INSERT INTO SUBJECT VALUES ('10CS82','SSM', 8, 4);
INSERT INTO SUBJECT VALUES ('10CS83','NM', 8, 4);
INSERT INTO SUBJECT VALUES ('10CS84','CC', 8, 4);
INSERT INTO SUBJECT VALUES ('10CS85','PW', 8, 4);
INSERT INTO SUBJECT VALUES ('10CS71','OOAD', 7, 4);
INSERT INTO SUBJECT VALUES ('10CS72','ECS', 7, 4);
INSERT INTO SUBJECT VALUES ('10CS73','PTW', 7, 4);
INSERT INTO SUBJECT VALUES ('10CS74','DWDM', 7, 4);
INSERT INTO SUBJECT VALUES (_10CS75','JAVA', 7, 4);
INSERT INTO SUBJECT VALUES ('10CS76','SAN', 7, 4);
INSERT INTO SUBJECT VALUES ('15CS51', 'ME', 5, 4);
INSERT INTO SUBJECT VALUES ('15CS52','CN', 5, 4);
INSERT INTO SUBJECT VALUES ('15CS53','DBMS', 5, 4);
INSERT INTO SUBJECT VALUES ('15CS54','ATC', 5, 4);
INSERT INTO SUBJECT VALUES ('15CS55','JAVA', 5, 3);
INSERT INTO SUBJECT VALUES ('15CS56','AI', 5, 3);
INSERT INTO SUBJECT VALUES ('15CS41','M4', 4, 4);
INSERT INTO SUBJECT VALUES ('15CS42','SE', 4, 4);
INSERT INTO SUBJECT VALUES ('15CS43','DAA', 4, 4);
INSERT INTO SUBJECT VALUES ('15CS44','MPMC', 4, 4);
INSERT INTO SUBJECT VALUES ('15CS45','OOC', 4, 3);
INSERT INTO SUBJECT VALUES ('15CS46','DC', 4, 3);
INSERT INTO SUBJECT VALUES ('15CS31','M3', 3, 4);
INSERT INTO SUBJECT VALUES ('15CS32','ADE', 3, 4);
INSERT INTO SUBJECT VALUES ('15CS33','DSA', 3, 4);
INSERT INTO SUBJECT VALUES ('15CS34','CO', 3, 4);
INSERT INTO SUBJECT VALUES ('15CS35','USP', 3, 3);
INSERT INTO SUBJECT VALUES ('15CS36','DMS', 3, 3);

INSERT INTO IAMARKS (USN, SUBCODE, SSID, TEST1, TEST2, TEST3) VALUES
('1RN13CS091','10CS81','CSE8C', 15, 16, 18);
INSERT INTO IAMARKS (USN, SUBCODE, SSID, TEST1, TEST2, TEST3) VALUES
('1RN13CS091','10CS82','CSE8C', 12, 19, 14);
INSERT INTO IAMARKS (USN, SUBCODE, SSID, TEST1, TEST2, TEST3) VALUES
('1RN13CS091','10CS83','CSE8C', 19, 15, 20);
INSERT INTO IAMARKS (USN, SUBCODE, SSID, TEST1, TEST2, TEST3) VALUES
('1RN13CS091','10CS84','CSE8C', 20, 16, 19);
INSERT INTO IAMARKS (USN, SUBCODE, SSID, TEST1, TEST2, TEST3) VALUES
('1RN13CS091','10CS85','CSE8C', 15, 15, 12);
```

```
SQL> SELECT * FROM STUDENT1;

USN        SNAME                     ADDRESS                       PHONE G
---------- ------------------------- ------------------------- ---------- -
1RN13CS020 AKSHAY                    BELAGAVI                  8877881122 M
1RN13CS062 SANDHYA                   BENGALURU                 7722829912 F
1RN13CS091 TEESHA                    BENGALURU                 7712312312 F
1RN13CS066 SUPRIYA                   MANGALURU                 8877881122 F
1RN14CS010 ABHAY                     BENGALURU                 9900211201 M
1RN14CS032 BHASKAR                   BENGALURU                 9923211099 M
1RN15CS011 AJAY                      TUMKUR                    9845091341 M
1RN15CS029 CHITRA                    DAVANGERE                 7696772121 F
1RN15CS045 JEEVA                     BELLARY                   9944850121 M
1RN15CS091 SANTOSH                   MANGALURU                 8812332201 M
1RN16CS045 ISMAIL                    KALBURGI                  9900232201 M
1RN16CS088 SAMEERA                   SHIMOGA                   9905542212 F
1RN16CS122 VINAYAKA                  CHIKAMAGALUR              8800880011 M
1RN14CS025 ASMI                      BENGALURU                 7894737377 F


SQL> SELECT * FROM SEMSEC;

SSID          SEM S
----- ---------- -
CSE8A           8 A
CSE8B           8 B
CSE8C           8 C
CSE7A           7 A
CSE7B           7 B
CSE7C           7 C
CSE6A           6 A
CSE6B           6 B
CSE6C           6 C
CSE5A           5 A
CSE5B           5 B
CSE5C           5 C
CSE4A           4 A
CSE4B           4 B
CSE4C           4 C
CSE3A           3 A
CSE3B           3 B
CSE3C           3 C
CSE2A           2 A
CSE2C           2 C
CSE2B           2 B
CSE1A           1 A
CSE1B           1 B
CSE1C           1 C
```

```
SQL> SELECT * FROM CLASS;

USN        SSID
---------- -----
1RN13CS020 CSE8A
1RN13CS062 CSE8A
1RN13CS066 CSE8B
1RN13CS091 CSE8C
1RN14CS010 CSE7A
1RN14CS025 CSE7A
1RN14CS032 CSE7A
1RN15CS011 CSE4A
1RN15CS029 CSE4A
1RN15CS045 CSE4B
1RN15CS091 CSE4C
1RN16CS045 CSE3A
1RN16CS088 CSE3B
1RN16CS122 CSE3C

14 rows selected.
```

SUBCODE	TITLE	SEM	CREDITS
10CS81	ACA	8	4
10CS82	SSM	8	4
10CS83	NM	8	4
10CS84	CC	8	4
10CS85	PW	8	4
10CS71	OOAD	7	4
10CS72	ECS	7	4
10CS73	PTW	7	4
10CS74	DWDM	7	4
10CS75	JAVA	7	4
10CS76	SAN	7	4
15CS51	ME	5	4
15CS52	CN	5	4
15CS53	DBMS	5	4
15CS54	ATC	5	4
15CS55	JAVA	5	3
15CS56	AI	5	3
15CS41	M4	4	4
15CS42	SE	4	4
15CS43	DAA	4	4
15CS44	MPMC	4	4
15CS45	OOC	4	3
15CS46	DC	4	3
15CS31	M3	3	4
15CS32	ADE	3	4
15CS33	DSA	3	4
15CS34	CO	3	4
15CS35	USP	3	3
15CS36	DMS	3	3

```
SQL> SELECT * FROM IAMARKS;

USN        SUBCODE  SSID       TEST1      TEST2      TEST3    FINALIA
---------- -------- ----- ---------- ---------- ---------- ----------
1RN13CS091 10CS81   CSE8C          15         16         18
1RN13CS091 10CS82   CSE8C          12         19         14
1RN13CS091 10CS83   CSE8C          19         15         20
1RN13CS091 10CS84   CSE8C          20         16         19
1RN13CS091 10CS85   CSE8C          15         15         12
```

Queries:

1. List all the student details studying in fourth semester 'C' section.

```
SELECT S.*, SS.SEM, SS.SEC
FROM STUDENT S, SEMSEC SS, CLASS C
WHERE S.USN = C.USN AND
SS.SSID = C.SSID AND
SS.SEM = 4 ANDSS.SEc='C';
```

```
USN        SNAME                ADDRESS                   PHONE G        SEM S
---------- -------------------- ------------------------- ---------- - ---------- -
1RN15CS091 SANTOSH              MANGALURU                 8812332201 M          4 C
```

2. Compute the total number of male and female students in each semester and in each section.

```
SELECT SS.SEM, SS.SEC, S.GENDER, COUNT (S.GENDER) AS COUNT
FROM STUDENT S, SEMSEC SS, CLASS C
WHERES.USN = C.USN AND
SS.SSID = C.SSID
GROUP BY SS.SEM, SS.SEC, S.GENDER
ORDER BY SEM
```

```
       SEM S G      COUNT
---------- - - ----------
         3 A M          1
         3 B F          1
         3 C M          1
         4 A F          1
         4 A M          1
         4 B M          1
         4 C M          1
         7 A F          1
         7 A M          2
         8 A F          1
         8 A M          1
         8 B F          1
         8 C F          1
```

3. Create a view of Test1 marks of student USN '1BI15CS101' in all subjects.

```
CREATE VIEW STU_TEST1_MARKS_VIEW
AS
SELECT TEST1, SUBCODE
FROM IAMARKS
WHERE USN = '1RN13CS091';
```

```
     TEST1 SUBCODE
---------- --------
        15 10CS81
        12 10CS82
        19 10CS83
        20 10CS84
        15 10CS85
```

5. Categorize students based on the following criterion:
If FinalIA = 17 to 20 then CAT = 'Outstanding'
If FinalIA = 12 to 16 then CAT = 'Average'
If FinalIA< 12 then CAT = 'Weak'
Give these details only for 8th semester A, B, and C section students.

```
SELECT S.USN,S.SNAME,S.ADDRESS,S.PHONE,S.GENDER,
(CASE
WHEN IA.FINALIA BETWEEN 17 AND 20 THEN 'OUTSTANDING'
WHEN IA.FINALIA BETWEEN 12 AND 16 THEN 'AVERAGE'
ELSE 'WEAK'
END) AS CAT
FROM STUDENT S, SEMSEC SS, IAMARKS IA, SUBJECT SUB
WHERE S.USN = IA.USN AND
SS.SSID = IA.SSID AND
SUB.SUBCODE = IA.SUBCODE AND
SUB.SEM = 8;
```

```
USN        SNAME                ADDRESS                   PHONE G CAT
---------- -------------------- ------------------------- ---------- - -----------
1RN13CS091 TEESHA               BENGALURU                 7712312312 F OutStanding
1RN13CS091 TEESHA               BENGALURU                 7712312312 F OutStanding
1RN13CS091 TEESHA               BENGALURU                 7712312312 F OutStanding
1RN13CS091 TEESHA               BENGALURU                 7712312312 F OutStanding
1RN13CS091 TEESHA               BENGALURU                 7712312312 F Average
```

rt C: Practise Questions

SQL-QUERIES

Emp table data

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH	CLERK	7902	17-Dec-80	800		20
7499	ALLEN	SALESMAN	7698	20-Feb-81	1600	300	30
7521	WARD	SALESMAN	7698	22-Feb-81	1250	500	30
7566	JONES	MANAGER	7839	02-Apr-81	2975		20
7654	MARTIN	SALESMAN	7698	28-Sep-81	1250	1400	30
7698	BLAKE	MANAGER	7839	01-May-81	2850		30
7782	CLARK	MANAGER	7839	09-Jun-81	2450		10
7788	SCOTT	ANALYST	7566	09-Dec-82	3000		20
7839	KING	PRESIDENT		17-Nov-81	5000		10
7844	TURNER	SALESMAN	7698	08-Sep-81	1500	0	30
7876	ADAMS	CLERK	7788	12-Jan-83	1100		20
7900	JAMES	CLERK	7698	03-Dec-81	950		30
7902	FORD	ANALYST	7566	03-Dec-81	3000		20
7934	MILLER	CLERK	7782	23-Jan-82	1300		10

Dept table data

DEPTNO	DNAME	LOC
10	**ACCOUNTING**	**NEW YORK**
20	**RESEARCH**	**DALLAS**
30	**SALES**	**CHICAGO**
40	**OPERATIONS**	**BOSTON**

1. Display all the information of the EMP table?
2. Display unique Jobs from EMP table?
3. List the emps in the asc order of their Salaries?
4. List the details of the emps in asc order of the Dptnos and desc of Jobs?
5. Display all the unique job groups in the descending order?
6. Display all the details of all ‘Mgrs’
7. List the emps who joined before 1981.
8. List the Empno, Ename, Sal, Daily sal of all emps in the asc order of Annsal.

9. Display the Empno, Ename, job, Hiredate, Exp of all Mgrs
10. List the Empno, Ename, Sal, Exp of all emps working for Mgr 7369.
11. Display all the details of the emps whose Comm. Is more than their Sal.
12. List the emps in the asc order of Designations of those joined after the second half of 1981.

13. List the emps along with their Exp and Daily Sal is more than Rs.100.
14. List the emps who are either 'CLERK' or 'ANALYST' in the Desc order.
15. List the emps who joined on 1-MAY-81,3-DEC-81,17-DEC-81,19-JAN-80 in asc order of seniority.

16. List the emp who are working for the Deptno 10 or20.
17. List the emps who are joined in the year 81.
18. List the emps who are joined in the month of Aug 1980.
19. List the emps Who Annual sal ranging from 22000 and 45000.
20. List the Enames those are having five characters in their Names.
21. List the Enames those are starting with 'S' and with five characters.
22. List the emps those are having four chars and third character must be 'r'.
23. List the Five character names starting with 'S' and ending with 'H'.
24. List the emps who joined in January.
25. List the emps who joined in the month of which second character is 'a'.
26. List the emps whose Sal is four digit number ending with Zero.
27. List the emps whose names having a character set 'll' together.

28. List the emps those who joined in 80's.

29. List the emps who does not belong to Deptno 20.
30. List all the emps except 'PRESIDENT' & 'MGR" in asc order of Salaries.
31. List all the emps who joined before or after 1981.
32. List the emps whose Empno not starting with digit78.
33. List the emps who are working under 'MGR'.

34. List the emps who joined in any year but not belongs to the month of March.
35. List all the Clerks of Deptno 20.
36. List the emps of Deptno 30 or 10 joined in the year 1981.
37. Display the details of SMITH.
38. Display the location of SMITH.
39. List the total information of EMP table along with DNAME and Loc of all the emps Working Under 'ACCOUNTING' & 'RESEARCH' in the asc Deptno.

40. List the Empno, Ename, Sal, Dname of all the 'MGRS' and 'ANALYST' working in New York, Dallas with an exp more than 7 years without receiving the Comm asc order of Loc.

41. Display the Empno, Ename, Sal, Dname, Loc, Deptno, Job of all emps working at CJICAGO or working for ACCOUNTING dept with Ann Sal>28000, but the Sal should not be=3000 or 2800 who doesn't belongs to the Mgr and whose no is having a digit '7' or '8' in 3rd position in the asc order of Deptno and desc order of job.

42. Display the total information of the emps along with Grades in the asc order.

43. List all the Grade2 and Grade 3 emps.
44. Display all Grade 4,5 Analyst and Mgr.

45. List the Empno, Ename, Sal, Dname, Grade, Exp, and Ann Sal of emps working for Dept10 or20.

46. List all the information of emp with Loc and the Grade of all the emps belong to the Grade range from 2 to 4 working at the Dept those are not starting with char set 'OP' and not ending with 'S' with the designation having a char 'a' any where joined in the year 1981 but not in the month of Mar or Sep and Sal not end with '00' in the asc order of Grades

47. List the details of the Depts along with Empno, Ename or without the emps
48. List the details of the emps whose Salaries more than the employee BLAKE.
49. List the emps whose Jobs are same as ALLEN.
50. List the emps who are senior to King.

51. List the Emps who are senior to their own MGRS.
52. List the Emps of Deptno 20 whose Jobs are same as Deptno10.

53. List the Emps whose Sal is same as FORD or SMITH in desc order of Sal.
54. List the emps Whose Jobs are same as MILLER or Sal is more than ALLEN.
55. List the Emps whose Sal is > the total remuneration of the SALESMAN.
56. List the emps who are senior to BLAKE working at CHICAGO & BOSTON.
57. List the Emps of Grade 3,4 belongs to the dept ACCOUNTING and RESEARCH whose Sal is more than ALLEN and exp more than SMITH in the asc order of EXP.

58. List the emps whose jobs same as SMITH or ALLEN.

59. Write a Query to display the details of emps whose Sal is same as of

60. Any jobs of deptno 10 those that are not found in deptno 20.

61. List of emps of emp1 who are not found in emp2.
62. Find the highest sal of EMP table.
63. Find details of highest paid employee.

64. Find the highest paid employee of sales department.
65. List the most recently hired emp of grade3 belongs to location CHICAGO.

66. List the employees who are senior to most recently hired employee working under king.

67. List the details of the employee belongs to newyork with grade 3 to 5 except 'PRESIDENT' whose sal> the highest paid employee of Chicago in a group where there is manager and salesman not working under king

68. List the details of the senior employee belongs to 1981.

69. List the employees who joined in 1981 with the job same as the most senior person of the year 1981.

70. List the most senior empl working under the king and grade is more than 3.

71. Find the total sal given to the MGR.

72. Find the total annual sal to distribute job wise in the year 81.

73. Display total sal employee belonging to grade 3.

74. Display the average salaries of all the clerks.

75. List the employeein dept 20 whose sal is >the average sal 0f dept 10 emps.

76. Display the number of employee for each job group deptno wise.

77. List the manage rno and the number of employees working for those mgrs in the ascending Mgrno.

78. List the department,details where at least two emps are working

79. Display the Grade, Number of emps, and max sal of each grade.

80. Display dname, grade, No. of emps where at least two emps are clerks.

81. List the details of the department where maximum number of emps are working.

82. Display the emps whose manager name is jones.

83. List the employees whose salary is more than 3000 after giving 20% increment.

84. List the emps with dept names.

85. List the emps who are not working in sales dept.

86. List the emps name ,dept, sal and comm. For those whose salary is between 2000 and 5000 while loc is Chicago.

87. List the emps whose sal is greater than his managers salary

88. List the grade, EMP name for the deptno 10 or deptno 30 but sal grade is not 4 while they joined the company before '31-dec-82'.

89. List the name ,job, dname, location for those who are working as MGRS.

90. List the emps whose mgr name is jones and also list their manager name.

91. List the name and salary of ford if his salary is equal to hisal of his grade.

92. Lit the name, job, dname ,sal, grade dept wise

93. List the emp name, job, sal, grade and dname except clerks and sort on the basis of highest sal.

94. List the emps name, job who are with out manager.

95. List the names of the emps who are getting the highest sal dept wise.

96. List the emps whose sal is equal to the average of max and minimum

97. List the no. of emps in each department where the no. is more than 3.

98. List the names of depts. Where atleast 3 are working in that department.

99. List the managers whose sal is more than his employess avg salary.

100. List the name,salary,comm. For those employees whose net pay is greater than or equal to any other employee salary of the company.

101. List the emp whose sal<his manager but more than any other manager.

102. List the employee names and his average salary department wise.

103. Find out least 5 earners of the company.

104. Find out emps whose salaries greater than salaries of their managers.

105. List the managers who are not working under the president.

106. List the records from emp whose deptno isnot in dept.

107. List the Name , Salary, Comm and Net Pay is more than any other employee.

108. List the Enames who are retiring after 31-Dec-89 the max Job period is 20Y.

109. List those Emps whose Salary is odd value.
110. List the emp's whose Salary contain 3 digits.

111. List the emps who joined in the month of DEC.
112. List the emps whose names contains 'A'.
113. List the emps whose Deptno is available in his Salary.

114. List the emps whose first 2 chars from Hiredate=last 2 characters of Salary.
115. List the emps Whose 10% of Salary is equal to year of joining.

116. List first 50% of chars of Ename in Lower Case and remaining are upper Case.

117. List the Dname whose No. of Emps is =to number of chars in the Dname.

118. List the emps those who joined in company before 15th of the month.

119. List the Dname, no of chars of which is = no. of emp's in any other Dept.

120. List the emps who are working as Managers.

121. List THE Name of dept where highest no.of emps are working.

122. Count the No.of emps who are working as 'Managers'(using set option).

123. List the emps who joined in the company on the same date.

124. List the details of the emps whose Grade is equal to one tenth of Sales Dept.

125. List the name of the dept where more than average no. of emps are working.

126. List the Managers name who is having max no.of emps working under him.

127. List the Ename and Sal is increased by 15% and expressed as no.of Dollars.

128. Produce the output of EMP table 'EMP_AND_JOB' for Ename and Job.

129. Produce the following output from EMP.

EMPLOYEE

SMITH (clerk)
ALLEN (Salesman)

130) List the emps with Hire date in format June 4, 1988.

131) Print a list of emp's Listing 'just salary' if Salary is more than 1500, on target if Salary is 1500 and 'Below 1500' if Salary is less than 1500.

132) Write a query which return the day of the week for any date entered in format 'DD-MM-YY'.

133) Write a query to calculate the length of service of any employee with the company, use DEFINE to avoid repetitive typing of functions.

134) Give a string of format 'NN/NN', verify that the first and last two characters are numbers and that the middle character is'/'. Print the expression 'YES' if valid, 'NO' if not valid. Use the following values to test your solution. '12/34','01/1a', '99/98'.

135) Emps hired on or before 15th of any month are paid on the last Friday of that month those hired after 15th are paid on the first Friday of the following month. Print a list of emps their hire date and the first pay date. Sort on hire date.

136) Count the no. of characters with out considering spaces for each name.

137) Find out the emps who are getting decimal value in their Sal without using like operator.

138) List those emps whose Salary contains first four digit of their Deptno.

139) List those Managers who are getting less than his emps Salary.

140) Print the details of all the emps who are sub-ordinates to Blake.

141) List the emps who are working as Managers using co-related sub-query.

142) List the emps whose Mgr name is 'Jones' and also with his Manager name.

143) Define a variable representing the expression used to calculate on emps total annual remuneration use the variable in a statement, which finds all emps who can earn 30000 a year or more.

144) Find out how may Managers are their in the company.

145) Find Average salary and Average total remuneration for each Job type. Remember Salesman earn commission.secommm

146) Check whether all the emps numbers are indeed unique.

147) List the emps who are drawing less than 1000 Sort the output by Salary.

148) List the employee Name, Job, Annual Salary, deptno, Dept name and grade who earn 36000 a year or who are not CLERKS.

149) Find out the Job that was filled in the first half of 1983 and same job that was filled during the same period of 1984.

150) Find out the emps who joined in the company before their Managers.

151) List all the emps by name and number along with their Manager's name and number. Also List KING who has no 'Manager'.

152) Find all the emps who earn the minimum Salary for each job wise in ascending order.

153) Find out all the emps who earn highest salary in each job type. Sort in descending salary order.
154) Find out the most recently hired emps in each Dept order by Hiredate.
155) List the employee name,Salary and Deptno for each employee who earns a salary greater than the average for their department order by Deptno.

156) List the Deptno where there are no emps.
157) List the No.of emp's and Avg salary within each department for each job.
158) Find the maximum average salary drawn for each job except for 'President'.
159) Find the name and Job of the emps who earn Max salary and Commission.
160) List the Name, Job and Salary of the emps who are not belonging to the department 10 but who have the same job and Salary as the emps of dept 10.

161) List the Deptno, Name, Job, Salary and Sal+Comm of the SALESMAN who are earning maximum salary and commission in descending order.
162) List the Deptno, Name, Job, Salary and Sal+Comm of the emps who earn the second highest earnings (sal + comm.).

163) List the Deptno and their average salaries for dept with the average salary less than the averages for all department

164) List out the Names and Salaries of the emps along with their manager names and salaries for those emps who earn more salary than their Manager.
165) List out the Name, Job, Salary of the emps in the department with the highest average salary.

166) List the empno,sal,comm. Of emps.

167) List the details of the emps in the ascending order of the sal.

168) List the dept in the ascending order of the job and the desc order of the emps print empno, ename.

169) Display the unique dept of the emps.

170) Display the unique dept with jobs.
171) Display the details of the blake.

172) List all the clerks.

173) list all the employees joined on 1st may 81.

174) List the empno,ename,sal,deptno of the dept 10 emps in the ascending order of salary.

175) List the emps whose salaries are less than 3500.

176) List the empno,ename,sal of all the emp joined before 1 apr 81.

177) List the emp whose annual sal is <25000 in the asc order of the salaries.

178) List the empno,ename,annsal,dailysal of all the salesmen in the asc ann sal

179) List the empno,ename,hiredate,current date & exp in the ascending order of the exp.

180) List the emps whose exp is more than 10 years.

181) List the empno,ename,sal,TA30%,DA 40%,HRA 50%,GROSS,LIC,PF,net,deduction,net allow and net sal in the ascending order of the net salary.

182) List the emps who are working as managers.

183) List the emps who are either clerks or managers.

184) List the emps who have joined on the following dates 1 may 81,17 nov 81,30 dec 81

185) List the emps who have joined in the year 1981.

186) List the emps whose annual sal ranging from 23000 to 40000.

187) List the emps working under the mgrs 7369,7890,7654,7900.

188) List the emps who joined in the second half of 82.

189) List all the 4char emps.

190) List the emp names starting with ‘M’ with 5 chars.

191) List the emps end with ‘H’ all together 5 chars.

192) List names start with ‘M’.

193) List the emps who joined in the year 81.

194) List the emps whose sal is ending with 00.

195) List the emp who joined in the month of JAN.

196) Who joined in the month having char ‘a’.

197) Who joined in the month having second char ‘a’

198) List the emps whose salary is 4 digit number.

199) List the emp who joined in 80’s.

200) List the emp who are clerks who have exp more than 8ys.

201) List the mgrs of dept 10 or 20.

202) List the emps joined in jan with salary ranging from 1500 to 4000.

203) List the unique jobs of dept 20 and 30 in desc order.

204) List the emps along with exp of those working under the mgr whose number is starting with 7 but should not have a 9 joined before 1983.

205) List the emps who are working as either mgr or analyst with the salary ranging from 2000 to 5000 and with out comm.

206) List the empno,ename,sal,job of the emps with /ann sal <34000 but receiving some comm. Which should not be>sal and desg should be sales man working for dept 30.

207) List the emps who are working for dept 10 or 20 with desgs as clerk or analyst with a sal is either 3 or 4 digits with an exp>8ys but does not belong to mons of mar,apr,sep and working for mgrs &no is not ending with 88 and 56.

208) List the empno,ename,sal,job,deptno&exp of all the emps belongs to dept 10 or 20 with an exp 6 to 10 y working under the same mgr with out comm. With a job not ending irrespective of the position with comm.>200 with exp>=7y and sal<2500 but not belongs to the month sep or nov working under the mgr whose no is not having digits either 9 or 0 in the asc dept& desc dept

209) List the details of the emps working at Chicago.

210) List the empno,ename,deptno,loc of all the emps.

211) List the empno,ename,loc,dname of all the depts.,10 and 20.

212) List the empno, ename, sal, loc of the emps working at Chicago dallas with an exp>6ys.

213) List the emps along with loc of those who belongs to dallas ,newyork with sal ranging from 2000 to 5000 joined in 81.

214) List the empno,ename,sal,grade of all emps.

215) List the grade 2 and 3 emp of Chicago.

216) List the emps with loc and grade of accounting dept or the locs dallas or Chicago with the grades 3 to 5 &exp >6y

217) List the grades 3 emps of research and operations depts..joined after 1987 and whose names should not be either miller or allen.

218) List the emps whose job is same as smith.

219) List the emps who are senior to miller.

220) List the emps whose job is same as either allen or sal>allen.

221) List the emps who are senior to their own manager.

222) List the emps whose sal greater than blakes sal.

223) List the dept 10 emps whose sal>allen sal.

224) List the mgrs who are senior to king and who are junior to smith.

225) List the empno,ename,loc,sal,dname,loc of the all the emps belonging to king dept.

226) List the emps whose salgrade are greater than the grade of miller.

227) List the emps who are belonging dallas or Chicago with the grade same as adamsor exp more than smith.

228) List the emps whose sal is same as ford or blake.

229) List the emps whose sal is same as any one of the following.

230) Sal of any clerk of emp1 table.

231) Any emp of emp2 joined before 82.

232) The total remuneration (sal+comm.) of all sales person of Sales dept belonging to emp3 table.

233) Any Grade 4 emps Sal of emp 4 table.

234) Any emp Sal of emp5 table.
235) List the highest paid emp.

236) List the details of most recently hired emp of dept 30.

237) List the highest paid emp of Chicago joined before the most recently hired emp of grade 2.

238) List the highest paid emp working under k**ing.**