Sheet 1 –

Create database using following schema. Apply given Integrity Constraints and answer the following queries using SQL.(Fill up database with at least 10 records in each table).

SUPPLIER (Sno, Sname, address,City)

PARTS (Pno, Pname, Color, Weight, price) PROJECT (Jno, Mame, City)

SPJ (in_q,En, IUO, Qty)

Integrity Constraints:

- The values of any attributes should not be null.
- Legal cities are London, Paris, Rome, New York and Amsterdam.
- Supplier Number must start with 'S' followed by a decimal integer in the range of 0 to 9999.

Queries:

a) Find all the projects which are provided 3 or more parts .
b) Write a trigger on PROJECT table for update / insert such that the .jname value Should not be repeated.
c) Find full details of all projects in London.
d) Write a procedure for calculating the total sales of all the parts which are provided to projects in paris city.

Design an Input form for entering Parts data. Apply possible validations.

```
create table suppplier (
        Sno varchar(5) primary key check (Sno Like 'S%'),
        Sname varchar(20) not null,
        Address varchar(30) not null,
        City varchar(20) check( City in ('New York', 'London', 'Paris', 'Rome', 'Amsterdam'))
);

create table parts (
        Pno varchar(5) primary key,
        Pname varchar(10) not null,
        Color varchar(10) not null,
        weight int not null,
        price int not null
);

create table project (
        Jno varchar(5) primary key,
        Jname varchar(10) not null,
        city varchar(20) check (city in ('New York', 'London', 'Paris', 'Rome', 'Amsterdam'))
);

create table SPJ(
        Sno varchar(5),
        Pno varchar(5),
        Jno varchar(5),
        Qty int,
        primary key (Sno,Pno,Jno),
        Foreign key (Sno) references suppplier(Sno),
        Foreign key (Pno) referencesparts(Pno),
        Foreign key (Jno) references project(Jno),
);
```

```
INSERT INTO suppplier VALUES
('S101', 'A1 Supply','123  St', 'London'),
('S102', 'B2 Supply', '123 Main ','Paris'),
('S103', 'C3 Supply','123 Main St', 'Rome'),
('S104', 'D4 Supply','123 Main St', 'New York'),
('S105', 'E5 Supply','123  St', 'Amsterdam'),
('S106', 'F6 Supply','123 Main St', 'London'),
('S107', 'G7 Supply','123  St', 'Paris'),
('S108', 'H8 Supply','123 Main ', 'Rome'),
('S109', 'I9 Supply','123 Main St', 'London'),
('S110', 'J10 Supply','123 Main ', 'Paris');
```

```
INSERT INTO PARTS (Pno, Pname, Color, Weight, price) values
('P1', 'Bolt', 'Red', 12,123),
('P2', 'Nut', 'Blue', 05,12),
('P3', 'Screw', 'Green', 03,12),
('P4', 'Washer', 'Yellow', 1,4),
('P5', 'Nail', 'Black', 02,54),
('P6', 'Pin', 'White', 01,12),
('P7', 'Clip', 'Silver', 5,34),
('P8', 'Hook', 'Gray', 2,12),
('P9', 'Plate', 'Orange',134, 222),
('P10', 'Rod', 'Purple', 3,234);

INSERT INTO PROJECT (Jno, Jname, City) VALUES
('J1', 'Bridge', 'London'),
('J2', 'Building', 'Paris'),
('J3', 'Road', 'Rome'),
('J4', 'Tunnel', 'Paris'),
('J5', 'Airport', 'New York'),
('J6', 'Port', 'Amsterdam'),
('J7', 'Mall', 'London'),
('J8', 'Hospital', 'Paris'),
('J9', 'School', 'Rome'),
('J10', 'Factory', 'Paris');

INSERT INTO SPJ (Sno, Pno, Jno, Qty) VALUES
('S101','P1','J2',100),
('S101','P2','J2',200),
('S101','P3','J2',300),
('S102','P4','J4',150),
('S102','P5','J4',250),
('S103','P6','J2',180),
('S104','P7','J2',130),
('S105','P8','J2',170),
('S106','P9','J4',190),
('S107','P10','J4',160);

--a)----------------------------->
SELECT Jno
FROM SPJ
GROUP BY Jno
HAVING COUNT(DISTINCT Pno) >= 3;

--b)----------------------------->
CREATE TRIGGER trg_unique_jname
ON PROJECT
INSTEAD OF INSERT, UPDATE
AS
BEGIN
  IF EXISTS (
    SELECT 1 FROM PROJECT p
    JOIN INSERTED i ON p.Jname = i.Jname AND p.Jno != i.Jno
  )
  begin
         print('Duplicate project name not allowed!');
         ROLLBACK TRANSACTION;
  end;
  ELSE
         print('Record Inserted / updated Succesfully...');
END;


INSERT INTO PROJECT (Jno, Jname, City)
VALUES ('J11', 'Metro Rail', 'Rome');

INSERT INTO PROJECT (Jno, Jname, City)
VALUES ('J12', 'Bridge', 'London');


--c)----------------------------->
SELECT * FROM PROJECT WHERE City = 'London';

--d)----------------------------->

CREATE PROCEDURE TotalSalesInParis
AS
BEGIN
    SELECT SUM(P.Price * S.Qty) AS TotalSales -- Changed P.Price to P.Weight
    FROM SPJ S
    JOIN PROJECT J ON S.Jno = J.Jno
    JOIN PARTS P ON S.Pno = P.Pno
    WHERE J.City = 'Paris';
END;

EXEC TotalSalesInParis;
```

Sheet 2 –

Create database using following schema. Apply given Integrity Constraints and answer the following queries using SQL.(Fill up database with at least 10 records in each table).

PRODUCT (Maker, Modelno, Type)
PC (Modelno,,Speed, RAM, HD, CD, Price)
LAPTOP (Modelno, Speed, RAM, HD, Price)
PRINTER (Mg_delaQ, Color, Type, Price)

Details regarding Schemas

- PC relation contains model no. of PC, its speed in MHz, RAM in MB, HD size in GB, Speed of CD reader, and price.
- The value for Maker in Product table can be IBM, Compaq,etc.
- PRINTER relation contains model no., value of Color should be T(if printer is color) or F (if printer is black 8r. white), type(laser, ink-jet, dot- matrix or dry), and price.

Integrity Constraints:

- The values of any attributes should not be null.
- Product Type should one of these (PC, Laptop or Printer)

Queries:

a) Find PC models having a speed of at least 150 MHz.
b) Find those manufacturers that sell Laptops, but not PC's.
c) Write a trigger on LAPTOP table such that the price should not less than 30000
a) Write a procedure to find the manufacturer who has produced the most expensive laptop.

Design an input form for entering LAPTOP data. Apply possible validations.

```sql
CREATE DATABASE Product_Catalog_DB;
USE Product_Catalog_DB;

CREATE TABLE PRODUCT (
  Maker VARCHAR(20) NOT NULL,
  Modelno VARCHAR(10) PRIMARY KEY,
  Type VARCHAR(10) NOT NULL CHECK (Type IN ('PC', 'Laptop', 'Printer'))
);

CREATE TABLE PC (
  Modelno VARCHAR(10) PRIMARY KEY,
  Speed INT NOT NULL,
  RAM INT NOT NULL,
  HD INT NOT NULL,
  CD VARCHAR(10) NOT NULL,
  Price INT NOT NULL
);

CREATE TABLE LAPTOP (
  Modelno VARCHAR(10) PRIMARY KEY,
  Speed INT NOT NULL,
  RAM INT NOT NULL,
  HD INT NOT NULL,
  Price INT NOT NULL
);

CREATE TABLE PRINTER (
  Modelno VARCHAR(10) PRIMARY KEY,
  Color CHAR(1) NOT NULL CHECK (Color IN ('T', 'F')),
  Type VARCHAR(20) NOT NULL CHECK (Type IN ('laser', 'ink-jet', 'dot-matrix', 'dry')),
  Price INT NOT NULL
);

INSERT INTO PRODUCT VALUES
('HP', 'M1', 'PC'),
('Dell', 'M2', 'Laptop'),
('Lenovo', 'M3', 'Laptop'),
('HP', 'M4', 'Printer'),
('Dell', 'M5', 'PC'),
('Acer', 'M6', 'Laptop'),
('HP', 'M7', 'Laptop'),
('Canon', 'M8', 'Printer'),
('Asus', 'M9', 'Laptop'),
('Epson', 'M10', 'Printer');
```

```
INSERT INTO PC VALUES
('M1', 200, 8, 500, '52X', 45000),
('M5', 180, 16, 1000, '48X', 50000),
('M11', 150, 4, 320, '24X', 32000),
('M12', 170, 8, 512, '40X', 36000),
('M13', 190, 4, 250, '52X', 34000),
('M14', 210, 8, 750, '48X', 47000),
('M15', 160, 16, 1000, '40X', 52000),
('M16', 200, 8, 500, '52X', 40000),
('M17', 155, 8, 256, '36X', 35000),
('M18', 220, 16, 2000, '56X', 60000);

INSERT INTO LAPTOP VALUES
('M2', 250, 8, 512, 55000),
('M3', 200, 16, 1000, 60000),
('M6', 180, 8, 500, 35000),
('M7', 300, 16, 512, 75000),
('M9', 220, 8, 256, 30000),
('M19', 210, 4, 256, 32000),
('M20', 160, 4, 320, 36000),
('M21', 280, 16, 512, 65000),
('M22', 200, 8, 750, 48000),
('M23', 190, 16, 1000, 50000);

INSERT INTO PRINTER VALUES
('M4', 'F', 'ink-jet', 10000),
('M8', 'T', 'laser', 15000),
('M10', 'F', 'dot-matrix', 7000),
('M24', 'T', 'laser', 18000),
('M25', 'F', 'ink-jet', 9500),
('M26', 'T', 'laser', 17000),
('M27', 'F', 'dot-matrix', 7500),
('M28', 'F', 'dry', 10500),
('M29', 'T', 'laser', 20000),
('M30', 'F', 'ink-jet', 11000);

--a)------------------------------>
SELECT * FROM PC WHERE Speed <= 150;

--b)------------------------------>

select maker from product where type = 'laptop'
except
select maker from product where type = 'pc';

--c)------------------------------>

create trigger chk_laptop_price
on laptop
after insert, update
as
begin
        if exists ( select * from inserted where price < 300000)
                print('Laptop price not be less, at least 30000');
end;

insert into laptop values('M243', 250, 8, 512, 5000);


--d)------------------------------>
create procedure MostExpensiveLaptop
as begin
        select top 1 P.maker
        from product P
        join laptop L on P.modelNo = L.modelno
        order by L.price desc;
end;

exec MostExpensiveLaptop ;
```

Create database using following schema. Apply given Integrity Constraints and answer the following queries using SQL.(Fill up database with at least 10 records in each table).

PRODUCT (Maker, Modelno, Type)
PC (Modelno, Speed, RAM, HD, CD, Price)
LAPTOP (Modelno, Speed, RAM, HD, Price)
PRINTER (Modelno, Color, Type, Price)

Details regarding Schemas

- PC relation contains model no. of PC, its speed in MHz, RAM in MB, HD size in GB, Speed of CD reader, and price.
- The value for Maker in Product table can be IBM, Compaq,etc.
- PRINTER relation contains model no., value of Color should be T(if printer is color) or F (if printer is black & white), type(laser, ink-jet, dot- matrix or dry), and price.

Integrity Constraints:

- The values of any attributes should not be null.
- Product Type should one of these (PC, Laptop or Printer)

Queries:

a) Find the different types of printers produced by Epson.
b) Find those hard disk sizes which occur in two or more PC's.
c) Write a trigger on LAPTOP table such that the minimum speed should be 1201\4Hz.
d) Demonstrate the use of cursor using PRODUCT table.

Design an input form for entering LAPTOP data. Apply possible validations.


```
--a)----------------------------->
select PR.type
from product P
join printer PR on P.Modelno = PR.Modelno
where P.maker = 'epson';

--b)----------------------------->
SELECT HD
FROM PC
GROUP BY HD
HAVING COUNT(*) >= 2;

--c)----------------------------->

CREATE TRIGGER trg_LaptopSpeed
ON LAPTOP
INSTEAD OF INSERT
AS
BEGIN
    IF EXISTS (SELECT * FROM inserted WHERE Speed < 1201)
    BEGIN
        print('Laptop speed must be at least 1201 MHz.');
        ROLLBACK;
    END
    ELSE
         print(' Record inserted Succesfully...');
END;

INSERT INTO LAPTOP VALUES('M61', 1205, 5012, 8, 55000);

--d)----------------------------->
DECLARE @Model VARCHAR(20), @Maker VARCHAR(30), @Type VARCHAR(10);

DECLARE Product_Cursor CURSOR FOR
SELECT Modelno, Maker, Type FROM PRODUCT;

OPEN Product_Cursor;

FETCH NEXT FROM Product_Cursor INTO @Model, @Maker, @Type;
```

```
WHILE @@FETCH_STATUS = 0
BEGIN
    PRINT 'Model: ' + @Model + ', Maker: ' + @Maker + ', Type: ' + @Type;
    FETCH NEXT FROM Product_Cursor INTO @Model, @Maker, @Type;
END

CLOSE Product_Cursor;
DEALLOCATE Product_Cursor;
```

Sheet 4

Create database using following schema. Apply given Integrity Constraints and answer the following queries using SQL. (Fill up database with at least 10 records in each table).

DOCTOR (Did, Dname, Daddress, qualification)

1AT1ENTMASTER (Pcode, Pname, Padd, age, gender, bloodgroup, Pid) ADMITTEDPATIENT (P_code, Entry date, Discharge date, wardno, disease)

Integrity Constraints:

- The values of any attributes should not be null.
- Gender value should be M male) or F(female).
- Wardno should be less than 6.

Queries:

a) Find the details of doctors who are treating the patient of ward no 3.

b) Write a trigger on **PATIENTMASTER** table such that the blood group should be A,B,AB or 0.

c) Find the details of patient who are discharge within the period 03/03/12 to 25/ 03/12

d) Write a procedure on ADMIFTEDPATIENT table such as to calculate bill of all discharged patients. The charges per day for a ward is WardNo. * 100. e.g. For ward no 3 charges/day are 300Rs.

Create a data report for the details of Doctors. Report should also include the details of patients treated by that doctor.

```
create table doctor(
         Did int primary key,
         Dname varchar(20),
         Daddress varchar(50),
         qualification varchar(20)
);

create table patientMaster(
         Pcode int primary key,
         Pname varchar(20),
         Padd varchar(30),
         age int,
         gender char(1) check (gender in ('M', 'F')),
         bloodgroup varchar(5),
         Did int  foreign key (Did) references doctor(Did)
);

create table admittedPatient(
         p_code int foreign key (p_code) references patientMaster(Pcode),
         entryDate date,
         dischargeDate date,
         wardNo int check (wardNo < 6),
         disease varchar(20)
);

CREATE TABLE ADMITTEDPATIENT (
    P_code INT,
    Entry_date DATE NOT NULL,
    Discharge_date DATE NOT NULL,
    wardno INT CHECK (wardno < 6) NOT NULL,
    disease VARCHAR(100) NOT NULL,
    FOREIGN KEY (P_code) REFERENCES PATIENTMASTER(Pcode)
);
```

```
INSERT INTO DOCTOR VALUES
(1, 'Dr. Shah', 'Mumbai', 'MBBS'),
(2, 'Dr. Mehta', 'Pune', 'MD'),
(3, 'Dr. Khan', 'Delhi', 'MBBS'),
(4, 'Dr. Iyer', 'Chennai', 'MS'),
(5, 'Dr. Roy', 'Kolkata', 'MD'),
(6, 'Dr. Singh', 'Bhopal', 'MBBS'),
(7, 'Dr. Das', 'Patna', 'MS'),
(8, 'Dr. Rao', 'Hyderabad', 'MD'),
(9, 'Dr. Verma', 'Nagpur', 'MBBS'),
(10, 'Dr. Joshi', 'Surat', 'MD');

INSERT INTO PATIENTMASTER VALUES
(101, 'Amit', 'Mumbai', 25, 'M', 'A', 1),
(102, 'Neha', 'Delhi', 30, 'F', 'B', 2),
(103, 'Rahul', 'Pune', 40, 'M', 'O', 3),
(104, 'Sneha', 'Chennai', 22, 'F', 'AB', 4),
(105, 'Ajay', 'Kolkata', 33, 'M', 'A', 5),
(106, 'Pooja', 'Bhopal', 28, 'F', 'O', 6),
(107, 'Ravi', 'Patna', 45, 'M', 'B', 7),
(108, 'Kiran', 'Hyderabad', 29, 'F', 'A', 8),
(109, 'Meena', 'Nagpur', 32, 'F', 'AB', 9),
(110, 'Kunal', 'Surat', 26, 'M', 'O', 10);

INSERT INTO ADMITTEDPATIENT VALUES
(101, '2012-03-01', '2012-03-10', 3, 'Malaria'),
(102, '2012-03-05', '2012-03-07', 2, 'Dengue'),
(103, '2012-03-03', '2012-03-15', 1, 'Typhoid'),
(104, '2012-02-25', '2012-03-10', 3, 'Pneumonia'),
(105, '2012-03-20', '2012-03-22', 4, 'COVID'),
(106, '2012-03-10', '2012-03-25', 3, 'Injury'),
(107, '2012-03-12', '2012-03-18', 2, 'Fever'),
(108, '2012-03-15', '2012-03-23', 1, 'Cancer'),
(109, '2012-03-17', '2012-03-20', 5, 'Tumor'),
(110, '2012-03-21', '2012-03-29', 3, 'Allergy');

--a)----------------------------->
select D.*
from doctor D
join patientMaster P on D.Did =P.Did
join admittedPatient A on P.Pcode = A.P_code
where A.wardNo =3;

--b)----------------------------->
CREATE TRIGGER trg_bloodgroup_check
ON PATIENTMASTER
INSTEAD OF INSERT
AS
BEGIN
  IF EXISTS (
    SELECT 1 FROM inserted
    WHERE bloodgroup NOT IN ('A', 'B', 'AB', 'O')
  )
    BEGIN
         print('Invalid blood group!');
      ROLLBACK;
    END
  ELSE
         print('Record Inserted / updated Succesfully...');
END;

INSERT INTO PATIENTMASTER VALUES (111, 'Tina', 'Goa', 29, 'F', 'A', 1);

INSERT INTO PATIENTMASTER VALUES (112, 'Rohan', 'Delhi', 35, 'M', 'X', 2);
--c)----------------------------->
SELECT PM.*
FROM PATIENTMASTER PM
JOIN ADMITTEDPATIENT AP ON PM.Pcode = AP.P_code
WHERE AP.Discharge_date BETWEEN '2012-03-03' AND '2012-03-25';

--d)----------------------------->

CREATE PROCEDURE calc_bill
AS
BEGIN
  SELECT
    P_code,wardNo,
    DATEDIFF(DAY, entryDate, dischargeDate) * wardno * 100 AS Bill
  FROM ADMITTEDPATIENT
  WHERE dischargeDate IS NOT NULL;
END;

EXEC calc_bill;
```

Create database using following schema. Apply given Integrity Constraints and answer the following queries using SQL. (Fill up database with at least 10 records in each table).

DOCTOR (Did, Dname, Daddress, qualification)
PATIENTMASTER (Pcode, Pname, Padd, age, gender, bloodgroup, Did) ADMITTEDPATIENT (Pcode, Entry date, Discharge_date, wardno, disease)

Integrity Constraints:

- The values of any attributes should not be null.
- Gender value should be M (male) or F(female).
- Wardno should be less than 6.

Queries:

a) Find the details of the doctors who are treating the patients of ward no 3 & display the result along with patient name & disease.
b) Find the name of the disease by which maximum patients are suffering.
c) Write a trigger on ADMITTEDPATIENT table such that the wardno value should be between 1-5.
d) Write a procedure to give the details of patients who are admitted in the hospital for more than 5 days.

Create a input form for doctors. Apply all possible validations.

```
--a)----------------------------->
select D.*
from doctor D
join patientMaster P on D.Did =P.Did
join admittedPatient A on P.Pcode = A.P_code
where A.wardNo =3;

--b)----------------------------->

SELECT TOP 1 disease, COUNT(*) AS patient_count
FROM ADMITTEDPATIENT
GROUP BY disease
ORDER BY patient_count DESC;


--c)----------------------------->

CREATE TRIGGER trg_ValidateWardNoRange_InsteadOf
ON ADMITTEDPATIENT
INSTEAD OF INSERT, UPDATE
AS
BEGIN
    IF EXISTS (SELECT 1 FROM inserted WHERE wardno < 1 OR wardno > 5)
    BEGIN
        PRINT ('Error: Ward number must be between 1 and 5.');
        ROLLBACK;
    END
    ELSE
         print('Record Inserted / updated Succesfully...');
END;


INSERT INTO ADMITTEDPATIENT VALUES (200, '2025-01-01', '2025-01-05', 0, 'Test Trigger Disease');


INSERT INTO ADMITTEDPATIENT VALUES (200, '2025-01-01', '2025-01-05', 4, 'Test Trigger Disease');

--d)----------------------------->


drop PROCEDURE Get_Long_Stay_Patients
CREATE PROCEDURE Get_Long_Stay_Patients
AS
BEGIN
    SELECT P.Pcode, P.Pname, DATEDIFF(DAY, A.entryDate, A.dischargeDate) AS Days_Admitted
```

```
    FROM PATIENTMASTER P
    JOIN ADMITTEDPATIENT A ON P.Pcode = A.P_code
    WHERE DATEDIFF(DAY, A.entryDate, A.dischargeDate) > 5;
END;

-- Call the procedure:
EXEC Get_Long_Stay_Patients;
```

Sheet - 7

Create database using following schema. Apply given Integrity Constraints and answer the following queries using SQL. (Fill up database with at least 10 records in each table).

DOCTOR (Did, Dname, Daddress, qualification)
PATIENTMASTER (Pcocle, Pname, Padd, age, gender, bloodgroup, Did) ADMITTEDPATIENT (Pcode, Entry date, Discharge date. wardno, disease)

Integrity Constraints:

- The values of any attributes should not be null.
- Gender value should be M (male) or F(female).
- Wardno should be less than 6.

Queries:

a) Find details of the patients who are treated by M.S. doctors.
b) Find the name of doctor who is treating maximum number of patients.
c) Write a procedure to give the details of patients who are admitted in the hospital for more than 15 days.
d) Create a view on DOCTOR & PATIENTMASTER tables. Update details of the patients who are treated by B.A.-M.S. doctors **to M.B.B.S** doctor.

Create a data entry form for discharging a patient. Also give information regarding his bill. (bill no_of_days * 500)

```
CREATE DATABASE Hospital_Management_DB;
USE Hospital_Management_DB;

CREATE TABLE DOCTOR (
    Did INT PRIMARY KEY,
    Dname VARCHAR(50) NOT NULL,
    Daddress VARCHAR(100) NOT NULL,
    qualification VARCHAR(50) NOT NULL
);

CREATE TABLE PATIENTMASTER (
    Pcode INT PRIMARY KEY,
    Pname VARCHAR(50) NOT NULL,
    Padd VARCHAR(100) NOT NULL,
    age INT NOT NULL,
    gender CHAR(1) NOT NULL CHECK (gender IN ('M', 'F')),
    bloodgroup VARCHAR(3) NOT NULL,
    Did INT NOT NULL,
    FOREIGN KEY (Did) REFERENCES DOCTOR(Did)
);

CREATE TABLE ADMITTEDPATIENT (
    Pcode INT PRIMARY KEY,
    Entry_date DATE NOT NULL,
    Discharge_date DATE NOT NULL,
    wardno INT NOT NULL CHECK (wardno >= 1 AND wardno <= 5),
    disease VARCHAR(100) NOT NULL,
    FOREIGN KEY (Pcode) REFERENCES PATIENTMASTER(Pcode)
);

INSERT INTO DOCTOR (Did, Dname, Daddress, qualification) VALUES
(1, 'Dr. Sharma', 'Mumbai', 'MBBS'),
```

```
(2, 'Dr. Khan', 'Pune', 'MS'),
(3, 'Dr. Reddy', 'Delhi', 'MD'),
(4, 'Dr. Gupta', 'Chennai', 'MBBS'),
(5, 'Dr. Patel', 'Kolkata', 'MS'),
(6, 'Dr. Das', 'Bhopal', 'MD'),
(7, 'Dr. Singh', 'Patna', 'MBBS'),
(8, 'Dr. Rao', 'Hyderabad', 'MD'),
(9, 'Dr. Kumar', 'Nagpur', 'B.A.-M.S.'),
(10, 'Dr. Mishra', 'Surat', 'MS'),
(11, 'Dr. Bose', 'Lucknow', 'MBBS'),
(12, 'Dr. Jain', 'Jaipur', 'MD');

INSERT INTO PATIENTMASTER (Pcode, Pname, Padd, age, gender, bloodgroup, Did) VALUES
(101, 'Amit', 'Mumbai', 25, 'M', 'A+', 1),
(102, 'Neha', 'Delhi', 30, 'F', 'B-', 2),
(103, 'Rahul', 'Pune', 40, 'M', 'O+', 3),
(104, 'Sneha', 'Chennai', 22, 'F', 'AB+', 2),
(105, 'Ajay', 'Kolkata', 33, 'M', 'A-', 5),
(106, 'Pooja', 'Bhopal', 28, 'F', 'O-', 1),
(107, 'Ravi', 'Patna', 45, 'M', 'B+', 7),
(108, 'Kiran', 'Hyderabad', 29, 'F', 'A+', 9),
(109, 'Meena', 'Nagpur', 32, 'F', 'AB-', 1),
(110, 'Kunal', 'Surat', 26, 'M', 'O+', 10),
(111, 'Divya', 'Bangalore', 35, 'F', 'A+', 11),
(112, 'Arjun', 'Goa', 55, 'M', 'B-', 12),
(113, 'Sonia', 'Agra', 28, 'F', 'O+', 9),
(114, 'Varun', 'Nashik', 31, 'M', 'AB+', 2),
(115, 'Ritu', 'Mysore', 42, 'F', 'A-', 7);

INSERT INTO ADMITTEDPATIENT (Pcode, Entry_date, Discharge_date, wardno, disease) VALUES
(101, '2024-05-01', '2024-05-05', 3, 'Malaria'),
(102, '2024-04-10', '2024-04-28', 2, 'Dengue'),
(103, '2024-05-15', '2024-05-17', 1, 'Typhoid'),
(104, '2024-04-20', '2024-05-08', 3, 'Pneumonia'),
(105, '2024-05-01', '2024-05-03', 4, 'COVID'),
(106, '2024-04-01', '2024-04-20', 3, 'Fracture'),
(107, '2024-05-12', '2024-05-16', 2, 'Fever'),
(108, '2024-04-05', '2024-04-25', 1, 'Appendicitis'),
(109, '2024-05-01', '2024-05-04', 5, 'Common Cold'),
(110, '2024-04-18', '2024-05-05', 3, 'Migraine');

--a)------------------------------>
SELECT * FROM
    PATIENTMASTER PM
JOIN DOCTOR D ON PM.Did = D.Did
WHERE D.qualification = 'MS';

--b)------------------------------>
INSERT INTO PATIENTMASTER VALUES
(111, 'Rahul', 'Nagpur', 39, 'M', 'B', 2),
(112, 'Komal', 'Pune', 28, 'F', 'O', 2);

INSERT INTO ADMITTEDPATIENT VALUES
(111, '2025-06-06', '2025-06-10', 2, 'Infection'),
(112, '2025-06-07', '2025-06-14', 3, 'Typhoid');


SELECT TOP 1 D.Dname, COUNT(P.Pcode) AS TotalPatients
FROM DOCTOR D
JOIN PATIENTMASTER P ON D.Did = P.Did
GROUP BY D.Dname
ORDER BY TotalPatients DESC;

--c)------------------------------>
INSERT INTO PATIENTMASTER VALUES
(113, 'Saurabh', 'Goa', 32, 'M', 'A+', 1),
(114, 'Neelam', 'Kolkata', 29, 'F', 'B+', 2),
(115, 'Dev', 'Hyderabad', 40, 'M', 'O+', 3);

INSERT INTO admittedPatient VALUES
(113, '2025-05-01', '2025-05-21', 2, 'TB'),
(114, '2025-04-10', '2025-04-28', 3, 'Malaria'),
(115, '2025-03-01', '2025-03-26', 4, 'Typhoid');

CREATE PROCEDURE Get_LongAdmitPatients
AS
BEGIN
    SELECT P_code, entryDate, dischargeDate, DATEDIFF(DAY, entryDate, dischargeDate) AS NoOfDays
    FROM ADMITTEDPATIENT
    WHERE DATEDIFF(DAY, entryDate, dischargeDate) > 15;
END;

exec Get_LongAdmitPatients;
--d)------------------------------>
```

```sql
-- DOCTOR table
INSERT INTO DOCTOR VALUES
(11, 'Dr.Ahuja', 'Pune', 'M.B.B.S'),
(12, 'Dr.Sharma', 'Mumbai', 'M.S'),
(13, 'Dr.Verma', 'Delhi', 'B.A.-M.S'),
(14, 'Dr.Iqbal', 'Nagpur', 'M.D');

-- PATIENTMASTER table
INSERT INTO PATIENTMASTER VALUES
(116, 'Ravi', 'Pune', 32, 'M', 'B+', 1),
(117, 'Neha', 'Mumbai', 28, 'F', 'A+', 2),
(118, 'Aman', 'Delhi', 40, 'M', 'O+', 3),
(119, 'Pooja', 'Nagpur', 25, 'F', 'AB+', 4),
(120, 'Mehul', 'Pune', 31, 'M', 'A-', 2),
(121, 'Seema', 'Delhi', 36, 'F', 'B+', 2);

CREATE VIEW DoctorPatientView AS
SELECT D.Did, D.Dname, D.qualification, P.Pcode, P.Pname
FROM DOCTOR D
JOIN PATIENTMASTER P ON D.Did = P.Did;

SELECT * FROM DoctorPatientView;

UPDATE DOCTOR
SET qualification = 'M.B.B.S'
WHERE qualification = 'B.A.-M.S';
```

Sheet – 8

Create database using following schema. Apply given Integrity Constraints and answer the following queries using SQL. (Fill up database with at least 10 records in each table).

ACCOUNT (accno, open date. acctype, balance)
TRANSACTION (trans id, trans date, accno, trans type, amount)
CUSTOMER (cust id, name, address, Accno)

Integrity Constraints:

- The values of any attributes should not be null.
- acctype value should be P(Personal) or J(Joint).
- Accno should be less than 3 digits.
- Trans type should be C(Credit) or D(Debit)

Queries:

a) Find the details of customers whose minimum balance is 1 lakhs.
a) Find the details of amount credited within the period 25-3-2012 to 28-3- 2012
a) Write a trigger on TRANSACTION table to calculate current balance of account on which transaction is made.
b) Write a cursor on ACCOUNT table of balance attribute such that if the balance is less than 10000 then print the 'loan is not provided'else 'loan is provided'.

Create a data entry for New account entry. Apply all possible validations

```sql
CREATE DATABASE BankDB;
USE BankDB;

CREATE TABLE ACCOUNT (
    accno INT PRIMARY KEY CHECK (accno < 1000),
    open_date DATE,
    acctype CHAR(1) CHECK (acctype IN ('P', 'J')),
    balance DECIMAL(10, 2)
```

```
);

CREATE TABLE CUSTOMER (
    cust_id INT PRIMARY KEY,
    name VARCHAR(50),
    address VARCHAR(100),
    accno INT NOT NULL FOREIGN KEY REFERENCES ACCOUNT(accno)
);

CREATE TABLE TRANSACT (
    trans_id INT PRIMARY KEY,
    trans_date DATE,
    accno INT FOREIGN KEY REFERENCES ACCOUNT(accno),
    trans_type CHAR(1) CHECK (trans_type IN ('C', 'D')),
    amount DECIMAL(10, 2)
);

-- ACCOUNT RECORDS
INSERT INTO ACCOUNT VALUES
(1, '2024-01-01', 'P', 100000),
(2, '2023-06-12', 'J', 9000),
(3, '2022-09-20', 'P', 50000),
(4, '2021-03-15', 'J', 250000),
(5, '2020-11-11', 'P', 30000),
(6, '2022-07-07', 'P', 7000),
(7, '2021-12-01', 'J', 1000),
(8, '2023-04-10', 'P', 110000),
(9, '2024-05-01', 'J', 85000),
(10, '2022-01-25', 'P', 99000);

-- CUSTOMER RECORDS
INSERT INTO CUSTOMER VALUES
(101, 'Ravi', 'Pune', 1),
(102, 'Neha', 'Mumbai', 2),
(103, 'Aman', 'Delhi', 3),
(104, 'Pooja', 'Nagpur', 4),
(105, 'Mehul', 'Goa', 5),
(106, 'Seema', 'Chennai', 6),
(107, 'Arjun', 'Surat', 7),
(108, 'Rina', 'Kolkata', 8),
(109, 'Karan', 'Indore', 9),
(110, 'Simran', 'Ahmedabad', 10);

INSERT INTO TRANSACT VALUES
(1, '2012-03-25', 1, 'C', 50000),
(2, '2012-03-26', 2, 'C', 10000),
(3, '2012-03-27', 3, 'C', 3000),
(4, '2012-03-28', 4, 'C', 25000),
(5, '2012-03-29', 5, 'D', 2000),
(6, '2023-01-01', 6, 'D', 500),
(7, '2024-04-10', 7, 'C', 6000),
(8, '2023-05-20', 8, 'C', 50000),
(9, '2022-11-11', 9, 'D', 4000),
(10, '2024-06-01', 10, 'C', 100000);



--a)----------------------------->
SELECT *
FROM CUSTOMER
WHERE accno IN (
    SELECT accno FROM ACCOUNT WHERE balance >= 100000
);


--b)----------------------------->
SELECT *
FROM TRANSACT
WHERE trans_type = 'C'
  AND trans_date BETWEEN '2012-03-25' AND '2012-03-28';


--c)----------------------------->
CREATE TRIGGER trg_UpdateBalance
ON TRANSACT
AFTER INSERT
AS
BEGIN
    SET NOCOUNT ON;

    UPDATE A
    SET A.balance = A.balance +
        (CASE WHEN I.trans_type = 'C' THEN I.amount ELSE -I.amount END)
    FROM ACCOUNT A
    JOIN INSERTED I ON A.accno = I.accno;
```

```
END;

INSERT INTO TRANSACT VALUES (11, '2025-06-06', 1, 'C', 5000);
INSERT INTO TRANSACT VALUES (12, '2025-06-06', 2, 'D', 3000);

SELECT * FROM ACCOUNT WHERE accno = 1;  -- 1 or accno = 2

--d)----------------------------->
DECLARE @accno INT, @balance INT;

DECLARE cur_balance CURSOR FOR
SELECT accno, balance FROM ACCOUNT;

OPEN cur_balance;

FETCH NEXT FROM cur_balance INTO @accno, @balance;

WHILE @@FETCH_STATUS = 0
BEGIN
    IF @balance < 10000
        PRINT 'AccNo: ' + CAST(@accno AS VARCHAR) + ' → Loan is NOT provided';
    ELSE
        PRINT 'AccNo: ' + CAST(@accno AS VARCHAR) + ' → Loan is provided';

    FETCH NEXT FROM cur_balance INTO @accno, @balance;
END

CLOSE cur_balance;
DEALLOCATE cur_balance;
```

Sheet 9

Create database using following schema. Apply given Integrity Constraints and answer the following queries using SQL. (Fill up database with at least 10 records in each table).

ACCOUNT (accno, open date, acctype, balance)
TRANSACTION (trans id, trans date, cno, trans type, amount) CUSTOMER (cust id, name, address, accno)

Integrity Constraints:

- The values of any attributes should not be null.
- acctype value should be P(Personal) or J(Joint).
- Accno should be less than 3 digits.
- Trans_type should be **C(Credit)** or D(Debit)

Queries:

a) Find the details of customers who have personal accounts & balance is less than 2 lakhs.
b) Find the details of customers who have joint accounts.
c) Write a trigger on ACCOUNT table such that if balance is less than 300 then customer should not withdraw the money.
d) Write a procedure on ACCOUNT & TRANSACTION table such that as user enters new transaction the balance is, updated in ACCOUNT table.

Create a data entry for New customer entry. Apply all possible validations.

```
CREATE TABLE ACCOUNT (
  accno INT PRIMARY KEY CHECK (accno < 100),
  open_date DATE NOT NULL,
  acctype CHAR(1) CHECK (acctype IN ('P', 'J')),
  balance DECIMAL(10, 2) CHECK (balance >= 0)
);

-- CUSTOMER TABLE
CREATE TABLE CUSTOMER (
```

```
  cust_id INT PRIMARY KEY,
  name VARCHAR(50) NOT NULL,
  address VARCHAR(100) NOT NULL,
  accno INT NOT NULL FOREIGN KEY REFERENCES ACCOUNT(accno)
);


CREATE TABLE TRANSACT (
  trans_id INT PRIMARY KEY,
  trans_date DATE NOT NULL,
  accno INT NOT NULL FOREIGN KEY REFERENCES ACCOUNT(accno),
  trans_type CHAR(1) CHECK (trans_type IN ('C', 'D')),
  amount DECIMAL(10, 2) NOT NULL
);

INSERT INTO ACCOUNT VALUES
(11, '2024-01-01', 'P', 50000),
(12, '2024-01-10', 'J', 100000),
(13, '2024-02-15', 'P', 150000),
(14, '2024-03-20', 'J', 20000),
(15, '2024-04-01', 'P', 299),
(16, '2024-04-02', 'J', 220000),
(17, '2024-04-03', 'P', 180000),
(18, '2024-04-04', 'J', 60000),
(19, '2024-04-05', 'P', 210000),
(20, '2024-04-06', 'P', 199999);

-- CUSTOMER
INSERT INTO CUSTOMER VALUES
(1, 'Raj', 'Delhi', 11),
(2, 'Neha', 'Mumbai', 12),
(3, 'Amit', 'Bhopal', 13),
(4, 'Pooja', 'Indore', 14),
(5, 'John', 'Goa', 15),
(6, 'Sara', 'Delhi', 16),
(7, 'Zara', 'Bangalore', 17),
(8, 'Ravi', 'Chennai', 18),
(9, 'Simran', 'Delhi', 19),
(10, 'Ali', 'Lucknow', 20);

INSERT INTO TRANSACT VALUES
(1, '2012-03-25', 11, 'C', 50000),
(2, '2012-03-26', 14, 'D', 10000),
(3, '2012-03-27', 12, 'C', 3000);

--a)------------------------------>
SELECT c.* FROM CUSTOMER c
JOIN ACCOUNT a ON c.accno = a.accno
WHERE a.acctype = 'P' AND a.balance < 200000;


--b)------------------------------>
SELECT c.* FROM CUSTOMER c
JOIN ACCOUNT a ON c.accno = a.accno
WHERE a.acctype = 'J';


--c)------------------------------>
CREATE TRIGGER trg_BlockLowBalance
ON TRANSACT
INSTEAD OF INSERT
AS
BEGIN
  DECLARE @accno INT, @type CHAR(1), @amt DECIMAL(10,2), @current DECIMAL(10,2);
  SELECT @accno = accno, @type = trans_type, @amt = amount FROM INSERTED;

  SELECT @current = balance FROM ACCOUNT WHERE accno = @accno;

  IF @type = 'D' AND @current - @amt < 300
  BEGIN
    print('Cannot withdraw. Balance would fall below 300......');
    ROLLBACK;
  END
  ELSE
   print('Record inserted / withdraw succesfully.....');
END;

select * from TRANSACT;

INSERT INTO TRANSACT VALUES
(11, '2012-03-25', 11, 'D', 5000000);


--d)------------------------------>
CREATE PROCEDURE AddTransaction
```

```
  @trans_id INT, @trans_date DATE, @accno INT,
  @trans_type CHAR(1), @amount DECIMAL(10,2)
AS
BEGIN
  IF @trans_type = 'C'
    UPDATE ACCOUNT SET balance = balance + @amount WHERE accno = @accno;
  ELSE IF @trans_type = 'D'
    UPDATE ACCOUNT SET balance = balance - @amount WHERE accno = @accno;

        INSERT INTO TRANSACT VALUES (@trans_id, @trans_date, @accno, @trans_type, @amount);
        print('Record inserted succesfully.....');
END;

-- Credit Test
EXEC AddTransaction 101, '2025-06-10', 11, 'C', 1000;
select * from TRANSACT;
-- Debit Test (safe)
EXEC AddTransaction 102, '2025-06-10', 11, 'D', 100;
```

Sheet 10

Create database using following schema. Apply given integrity Constraints and answer the following queries using SQL. (Fill up database with at least 10 records in each table).

ACCOUNT (accno, open date. acctype, balance)
TRANSACTION (trans id, trans date, accno, trans type, amount) CUSTOMER (cust id, name, address. accno)

Integrity Constraints:

- The values of any attributes should not be null.
- acctype value should be P(Personal) or Moira):
- Accno should be less than 3 digits.
- Transtype should be C(Credit) or D(Debit)

Queries:

a) Find the details of all transactions performed on account number 101. Also specify the name/names of cutomers who owns that account.
b) Find the details ol amount credited within the period 15 -3-2012 to 18 -3 -2012.
c) Write a trigger on insert on ACCOUNT table such that the account which is having balance less than or equal to 500 should not be debited.
d) Write a procedure on ACCOUNT table to calculate interest on current balance from open_date to today's date. (Take interest rate from user).

Create a data entry for New account entry. Apply all possible validations.

```
CREATE DATABASE BankDB;
USE BankDB;

CREATE TABLE ACCOUNT (
    accno INT primary key,
    open_date DATE NOT NULL,
    acctype CHAR(1) NOT NULL CHECK (acctype IN ('P', 'M')),
    balance DECIMAL(10,2) NOT NULL,
);

CREATE TABLE TRANSACT (
    trans_id INT NOT NULL PRIMARY KEY,
    trans_date DATE NOT NULL,
    accno INT ,
    trans_type CHAR(1) NOT NULL CHECK (trans_type IN ('C', 'D')),
    amount DECIMAL(10,2) NOT NULL,
    FOREIGN KEY (accno) REFERENCES ACCOUNT(accno)
);

CREATE TABLE CUSTOMER (
```

```
    cust_id INT NOT NULL PRIMARY KEY,
    name VARCHAR(255) NOT NULL,
    address VARCHAR(255) NOT NULL,
    accno INT,
    FOREIGN KEY (accno) REFERENCES ACCOUNT(accno)
);
INSERT INTO ACCOUNT VALUES
    (101, '2023-01-01', 'P', 1500),
    (102, '2022-05-10', 'M', 2500),
    (103, '2021-12-15', 'P', 800),
    (104, '2020-07-22', 'M', 3000),
    (105, '2023-02-18', 'P', 950),
    (106, '2019-09-03', 'M', 5000),
    (107, '2021-05-11', 'P', 1200),
    (108, '2020-11-29', 'M', 450),
    (109, '2018-06-07', 'P', 600),
    (110, '2017-03-14', 'M', 3500);

select * from ACCOUNT

INSERT INTO TRANSACT VALUES
    (1, '2023-06-10', 101, 'C', 500),
    (2, '2023-06-11', 101, 'D', 200),
    (3, '2012-03-15', 102, 'C', 1500),
    (4, '2012-03-17', 103, 'C', 200),
    (5, '2024-02-05', 104, 'D', 900),
    (6, '2022-08-14', 105, 'C', 1200),
    (7, '2023-01-30', 106, 'D', 1000),
    (8, '2012-03-16', 107, 'C', 750),
    (9, '2020-09-10', 108, 'D', 400),
    (10, '2012-03-18', 109, 'C', 300);

INSERT INTO CUSTOMER VALUES
    (1, 'John Doe', '123 Street, City', 101),
    (2, 'Alice Smith', '456 Avenue, City', 102),
    (3, 'Michael Johnson', '789 Boulevard, City', 103),
    (4, 'Emily Davis', '101 Road, Town', 104),
    (5, 'Daniel Brown', '202 Lane, Village', 105),
    (6, 'Sophia Wilson', '303 Street, City', 106),
    (7, 'Christopher Lee', '404 Avenue, City', 107),
    (8, 'Olivia Harris', '505 Boulevard, Town', 108),
    (9, 'Matthew Martin', '606 Road, Village', 109),
    (10, 'Emma Anderson', '707 Lane, City', 110);

--a)------------------------------>

SELECT T.*, C.name
FROM TRANSACT T
JOIN CUSTOMER C ON T.accno = C.accno
WHERE T.accno = 101;

--b)------------------------------>
SELECT * FROM TRANSACT
WHERE trans_type = 'C'
AND trans_date BETWEEN '2012-03-15' AND '2012-03-18';

--c)------------------------------>
CREATE TRIGGER PreventLowBalanceDebit
ON TRANSACT
FOR INSERT
AS
BEGIN
    DECLARE @accno INT, @amount DECIMAL(10,2), @current_balance DECIMAL(10,2);

    SELECT @accno = inserted.accno, @amount = inserted.amount
    FROM inserted;

    SELECT @current_balance = balance FROM ACCOUNT WHERE accno = @accno;

    IF (SELECT trans_type FROM inserted) = 'D' AND @current_balance - @amount <= 500
    BEGIN
        print('Insufficient balance for debit transaction');
        ROLLBACK;
    END
END;

INSERT INTO TRANSACT VALUES (11, '2023-06-10', 101, 'D', 50000000);


--d)------------------------------>

CREATE PROCEDURE CalculateInterest
    @acc_number INT,
    @rate DECIMAL(5,2)
AS
```

```sql
BEGIN
    DECLARE @days_diff INT, @interest_amount DECIMAL(10,2), @current_balance DECIMAL(10,2);

    SELECT @days_diff = DATEDIFF(DAY, open_date, GETDATE()),
           @current_balance = balance
    FROM ACCOUNT
    WHERE accno = @acc_number;

    SET @interest_amount = (@current_balance * @rate * @days_diff) / 36500;

    SELECT @acc_number AS Account, @interest_amount AS InterestAmount;
END;

EXEC CalculateInterest 101, 5;
```

Sheet 11

.Creat4 database using following schema. Apply given Integrity Constraints and answer the following queries using SQL. (Fill up database with at least 10 records in each table).

ACCOUNT (accno, open date, acctype, balance)
TRANSACTION (trans id, trans date, accno, trans type, amount)
CUSTOMER (cust id name, address, accno)

Integrity Constraints:

- The values of any attributes should not be null.
- acctype value should be P(Personal) or Moira).
- Accno should be less than 3 digits.
- Trans_type should be C(Credit) or D(Debit)

Queries:

a) Find the details of customers who have opened the accounts within the period 25-3-2012 to 28-3-2012.
b) Find the details of customers who have joint accounts & balance is less than 2 lakhs.
c) Write a trigger TRANSACTION on table to calculate the current balance of the account on which transaction is made.
(if trans_type = c then bal = bal amt else if trans_type = d then bal = bal — amt)
a) write a cursor on CUSTOMER table to fetch the last row.

Create a data entry for New customer entry. Apply all possible validations.

```sql
CREATE TABLE ACCOUNT (
    accno INT CHECK (accno < 100) PRIMARY KEY,
    open_date DATE NOT NULL,
    acctype VARCHAR(10) CHECK (acctype IN ('P', 'Moira')) NOT NULL,
    balance DECIMAL(10, 2) NOT NULL
);

CREATE TABLE CUSTOMER (
    cust_id INT PRIMARY KEY,
    name VARCHAR(50) NOT NULL,
    address VARCHAR(100) NOT NULL,
    accno INT NOT NULL,
    FOREIGN KEY (accno) REFERENCES ACCOUNT(accno)
);

CREATE TABLE TRANSACT (
    trans_id INT PRIMARY KEY,
    trans_date DATE NOT NULL,
    accno INT NOT NULL,
```

```
    trans_type CHAR(1) CHECK (trans_type IN ('C', 'D')) NOT NULL,
    amount DECIMAL(10,2) NOT NULL,
    FOREIGN KEY (accno) REFERENCES ACCOUNT(accno)
);

INSERT INTO ACCOUNT VALUES
(1, '2012-03-25', 'P', 150000),
(2, '2012-03-26', 'Moira', 180000),
(3, '2012-03-27', 'Moira', 90000),
(4, '2012-03-28', 'P', 50000),
(5, '2012-03-29', 'P', 300000),
(6, '2012-03-23', 'Moira', 250000),
(7, '2012-03-25', 'Moira', 40000),
(8, '2012-03-26', 'P', 120000),
(9, '2012-03-28', 'Moira', 195000),
(10, '2012-03-25', 'P', 100000);

INSERT INTO CUSTOMER VALUES
(1, 'Yogesh', 'Mumbai', 1),
(2, 'Amit', 'Pune', 2),
(3, 'Sana', 'Delhi', 3),
(4, 'Rohit', 'Mumbai', 4),
(5, 'Pooja', 'Nashik', 5),
(6, 'Ravi', 'Delhi', 6),
(7, 'Neha', 'Nagpur', 7),
(8, 'Vikas', 'Solapur', 8),
(9, 'Sanjana', 'Thane', 9),
(10, 'Kiran', 'Pune', 10);

INSERT INTO TRANSACT VALUES
(101, '2024-06-01', 1, 'C', 5000),
(102, '2024-06-02', 2, 'D', 20000),
(103, '2024-06-03', 3, 'C', 10000),
(104, '2024-06-04', 4, 'D', 5000),
(105, '2024-06-05', 5, 'C', 3000),
(106, '2024-06-06', 6, 'D', 10000),
(107, '2024-06-07', 7, 'C', 15000),
(108, '2024-06-08', 8, 'D', 8000),
(109, '2024-06-09', 9, 'C', 12000),
(110, '2024-06-10', 10, 'D', 4000);


--a)------------------------------>

SELECT C.*
FROM CUSTOMER C
JOIN ACCOUNT A ON C.accno = A.accno
WHERE A.open_date BETWEEN '2012-03-25' AND '2012-03-28';


--b)------------------------------>
SELECT C.*
FROM CUSTOMER C
JOIN ACCOUNT A ON C.accno = A.accno
WHERE A.acctype = 'Moira' AND A.balance < 200000;


--c)------------------------------>
CREATE TRIGGER trg_update_balance
ON TRANSACT
AFTER INSERT
AS
BEGIN
    UPDATE A
    SET A.balance =
        CASE
            WHEN I.trans_type = 'C' THEN A.balance + I.amount
            WHEN I.trans_type = 'D' THEN A.balance - I.amount
        END
    FROM ACCOUNT A
    JOIN INSERTED I ON A.accno = I.accno;
END;

SELECT * FROM ACCOUNT WHERE accno = 1;

INSERT INTO TRANSACT VALUES (201, GETDATE(), 1, 'C', 10000);

SELECT * FROM ACCOUNT WHERE accno = 1;


--d)------------------------------>
DECLARE @id INT, @name VARCHAR(50), @addr VARCHAR(100), @acc INT;

DECLARE c CURSOR SCROLL FOR
SELECT cust_id, name, address, accno FROM CUSTOMER ORDER BY cust_id;
```

```
OPEN c;
FETCH LAST FROM c INTO @id, @name, @addr, @acc;

PRINT 'Last Row -> ID: ' + CAST(@id AS VARCHAR) + ', Name: ' + @name + ', Address: ' + @addr + ', AccNo: ' +
CAST(@acc AS VARCHAR);

CLOSE c;
DEALLOCATE c;
```

Sheet 12

Create database using following schema. Apply given Integrity Constraints and answer the following queries using SQL. (Fill up database with at least 10 records in eae,h table).

EMPLOYEE (fname, Mame, ssn, sex, salary, joindate,superssn, dno,) DEPT
(dname, dnum, mgrssn, dlocation)
PROJECT (pname, pno, plocation, dnumber)
WORK ON (ssn, pno, hours)

Integrity Constraints:

- The values of any attributes should not be null.
- The deptno should less than 4 digit
- ssn -(social security no of-employee)
- mgrssn(manager_social_security_no)
- superssn(supervisor_social_securit)'_no)

Queries:

a) For every project located in 'jalgaon". List the pno,the controlling detptno and dept manager last name.

b) For each project on which more than two employee work, Find the pno, pname & no. of employees who work on the project.

a) create a view that has the deptname,manager name & manager salary for every dept.

a) Express the following constraint as SQL assertions -
"salary of employee must not be greater than the salary of the manager of the dept".

Create a data entry for New employee entry. Apply all possible validations.


```
CREATE DATABASE CompanyDB;
USE CompanyDB;

CREATE TABLE DEPT (
    dname VARCHAR(50) NOT NULL,
    dnum INT CHECK (dnum < 10000) PRIMARY KEY,
    mgrssn CHAR(9) NOT NULL,
    dlocation VARCHAR(50) NOT NULL
);

CREATE TABLE EMPLOYEE (
    fname VARCHAR(30) NOT NULL,
    mname VARCHAR(30) NOT NULL,
    ssn CHAR(9) PRIMARY KEY,
    sex CHAR(1) NOT NULL,
    salary DECIMAL(10,2) NOT NULL,
    joindate DATE NOT NULL,
    superssn CHAR(9),
    dno INT NOT NULL,
    FOREIGN KEY (dno) REFERENCES DEPT(dnum)
);

CREATE TABLE PROJECT (
    pname VARCHAR(50) NOT NULL,
    pno INT PRIMARY KEY,
    plocation VARCHAR(50) NOT NULL,
```

```
    dnumber INT NOT NULL,
    FOREIGN KEY (dnumber) REFERENCES DEPT(dnum)
);

CREATE TABLE WORK_ON (
    ssn CHAR(9) NOT NULL,
    pno INT NOT NULL,
    hours DECIMAL(5,2) NOT NULL,
    FOREIGN KEY (ssn) REFERENCES EMPLOYEE(ssn),
    FOREIGN KEY (pno) REFERENCES PROJECT(pno)
);

INSERT INTO DEPT VALUES
('HR', 101, '111111111', 'Mumbai'),
('Finance', 102, '222222222', 'Pune'),
('IT', 103, '333333333', 'Jalgaon'),
('Marketing', 104, '444444444', 'Delhi');

INSERT INTO EMPLOYEE VALUES
('Yogesh', 'R', '111111111', 'M', 80000, '2015-01-01', NULL, 101),
('Pooja', 'S', '222222222', 'F', 90000, '2016-03-12', '111111111', 102),
('Rahul', 'K', '333333333', 'M', 95000, '2017-05-25', '111111111', 103),
('Sneha', 'M', '444444444', 'F', 70000, '2018-07-15', '222222222', 104),
('Amit', 'A', '555555555', 'M', 85000, '2019-02-10', '333333333', 101),
('Neha', 'B', '666666666', 'F', 78000, '2020-06-18', '444444444', 102),
('Ravi', 'J', '777777777', 'M', 62000, '2021-01-05', '333333333', 103),
('Anita', 'C', '888888888', 'F', 65000, '2021-04-22', '444444444', 104),
('Karan', 'D', '999999999', 'M', 72000, '2022-03-10', '555555555', 101),
('Swati', 'L', '123123123', 'F', 60000, '2022-05-08', '222222222', 102);

INSERT INTO PROJECT VALUES
('HRMS', 1, 'Jalgaon', 101),
('Payroll', 2, 'Pune', 102),
('ERP', 3, 'Jalgaon', 103),
('Website', 4, 'Delhi', 104),
('CRM', 5, 'Jalgaon', 103),
('App Dev', 6, 'Mumbai', 101),
('Analytics', 7, 'Pune', 102),
('Cloud', 8, 'Jalgaon', 103),
('Security', 9, 'Delhi', 104),
('SEO', 10, 'Delhi', 104);

INSERT INTO WORK_ON VALUES
('111111111', 1, 10),
('222222222', 2, 8),
('333333333', 3, 12),
('444444444', 4, 7),
('555555555', 1, 9),
('666666666', 3, 6),
('777777777', 3, 5),
('888888888', 5, 7),
('999999999', 5, 8),
('123123123', 5, 6);

--a)----------------------------->
SELECT P.pno, D.dnum AS deptno, E.fname AS mgr_lastname
FROM PROJECT P
JOIN DEPT D ON P.dnumber = D.dnum
JOIN EMPLOYEE E ON D.mgrssn = E.ssn
WHERE P.plocation = 'Jalgaon';

--b)----------------------------->
SELECT P.pno, P.pname, COUNT(W.ssn) AS emp_count
FROM PROJECT P
JOIN WORK_ON W ON P.pno = W.pno
GROUP BY P.pno, P.pname
HAVING COUNT(W.ssn) > 2;

--c)----------------------------->
CREATE VIEW DeptManagerInfo AS
SELECT D.dname, E.fname + ' ' + E.mname AS manager_name, E.salary AS manager_salary
FROM DEPT D
JOIN EMPLOYEE E ON D.mgrssn = E.ssn;

select * from  DeptManagerInfo;

--d)----------------------------->
SELECT E.fname, E.salary, M.salary AS manager_salary
FROM EMPLOYEE E
JOIN EMPLOYEE M ON E.dno = M.dno AND M.ssn = (
    SELECT mgrssn FROM DEPT WHERE dnum = E.dno
)
WHERE E.salary > M.salary;
```

Sheet 13

Create database using following schema. Apply given Integrity Constraints and answer the following queries using SQL. (Fill up database with at least 10 records in each table).

EMPLOYEE (fname, lname, ssn, sex, salary, joindate,superssn, dno,)
DEPT (dname, dnum, En_grssn, (llocation)
PROJECT (pname, pno, plocation, dnumber)
WORK ON (5sn, pno, hours)

Integrity Constraints:

- The values of any attributes should not be null.
- The deptno should less than 4 digit
- Ssn (social security_no of employee)
- Mgrssn(managersocial_security_no)
- Superssn(supervisor_social_security_no)

Queries:

a) For each employee, Find the employee first & last name & the first & last name of his or her immediate supervisor.
b) For each dept, Find the deptno, the no. of employees in the dept & their average salary.
c) Create a view that has pname,controlling dept name, no of employees, & total hours worked per week on the project for each project with more than one employee working on it.
d) Create a procedure on EMPLOYEE table to determine the employees who will get promotion. (An employee will get promotion after working on 5 projects.)

Create a data report showing the information of all female employees working in "Research" department.

```sql
CREATE DATABASE CompanyDB;
USE CompanyDB;

CREATE TABLE DEPT (
    dname VARCHAR(50) NOT NULL,
    dnum INT NOT NULL CHECK (dnum < 10000),
    En_grssn CHAR(9) NOT NULL,
    llocation VARCHAR(50) NOT NULL,
    PRIMARY KEY (dnum)
);

CREATE TABLE EMPLOYEE (
    fname VARCHAR(30) NOT NULL,
    lname VARCHAR(30) NOT NULL,
    ssn CHAR(9) NOT NULL,
    sex CHAR(1) NOT NULL,
    salary DECIMAL(10,2) NOT NULL,
    joindate DATE NOT NULL,
    superssn CHAR(9) NOT NULL,
    dno INT NOT NULL,
    PRIMARY KEY (ssn),
    FOREIGN KEY (superssn) REFERENCES EMPLOYEE(ssn),
    FOREIGN KEY (dno) REFERENCES DEPT(dnum)
);

CREATE TABLE PROJECT (
    pname VARCHAR(50) NOT NULL,
    pno INT NOT NULL,
    plocation VARCHAR(50) NOT NULL,
```

```
    dnumber INT NOT NULL,
    PRIMARY KEY (pno),
    FOREIGN KEY (dnumber) REFERENCES DEPT(dnum)
);

CREATE TABLE WORK_ON (
    ssn CHAR(9) NOT NULL,
    pno INT NOT NULL,
    hours DECIMAL(5,2) NOT NULL,
    PRIMARY KEY (ssn, pno),
    FOREIGN KEY (ssn) REFERENCES EMPLOYEE(ssn),
    FOREIGN KEY (pno) REFERENCES PROJECT(pno)
);

INSERT INTO DEPT VALUES
('HR', 101, '111111111', 'New York'),
('Finance', 102, '222222222', 'Chicago'),
('IT', 103, '333333333', 'San Francisco');

INSERT INTO EMPLOYEE VALUES
('John', 'Smith', '123456789', 'M', 50000, '2018-06-01', '123456789', 101),
('Alice', 'Johnson', '987654321', 'F', 60000, '2019-01-15', '123456789', 101),
('Bob', 'Williams', '555443333', 'M', 55000, '2020-03-10', '987654321', 101),
('Carol', 'Jones', '112233445', 'F', 70000, '2017-11-30', '987654321', 102),
('David', 'Brown', '223344556', 'M', 65000, '2021-04-20', '112233445', 102),
('Eva', 'Davis', '334455667', 'F', 58000, '2022-09-12', '112233445', 102),
('Frank', 'Miller', '445566778', 'M', 62000, '2018-07-01', '334455667', 103),
('Grace', 'Wilson', '556677889', 'F', 64000, '2020-05-05', '445566778', 103),
('Henry', 'Moore', '667788990', 'M', 63000, '2016-10-15', '445566778', 103),
('Ivy', 'Taylor', '778899001', 'F', 71000, '2019-12-01', '556677889', 103);

INSERT INTO PROJECT VALUES
('HR Revamp', 1, 'New York', 101),
('Audit Tool', 2, 'Chicago', 102),
('Payroll System', 3, 'Chicago', 102),
('AI Platform', 4, 'San Francisco', 103),
('Cloud Migration', 5, 'San Francisco', 103);

INSERT INTO WORK_ON VALUES
('123456789', 1, 10),
('987654321', 1, 12),
('555443333', 1, 8),
('112233445', 2, 15),
('223344556', 2, 10),
('334455667', 3, 14),
('445566778', 4, 12),
('556677889', 4, 10),
('667788990', 4, 8),
('778899001', 5, 16),
('123456789', 2, 5),
('123456789', 3, 6),
('123456789', 4, 7),
('123456789', 5, 4);


--a)------------------------------>
SELECT E.fname AS EmpFname, E.lname AS EmpLname, S.fname AS SuperFname, S.lname AS SuperLname
FROM EMPLOYEE E
JOIN EMPLOYEE S ON E.superssn = S.ssn;


--b)------------------------------>
SELECT D.dnum, COUNT(E.ssn) AS NumEmployees, AVG(E.salary) AS AvgSalary
FROM DEPT D
JOIN EMPLOYEE E ON D.dnum = E.dno
GROUP BY D.dnum;


--c)------------------------------>
CREATE VIEW ProjectSummary AS
SELECT pname, dname, COUNT(W.ssn) AS NumEmployees, SUM(hours) AS TotalHours
FROM PROJECT P
JOIN DEPT D ON P.dnumber = D.dnum
JOIN WORK_ON W ON P.pno = W.pno
GROUP BY pname, dname
HAVING COUNT(W.ssn) > 1;


select * from ProjectSummary;

--d)------------------------------>

CREATE PROCEDURE GetPromotionCandidates
as
BEGIN
```

```
    SELECT fname, lname, ssn
    FROM EMPLOYEE
    WHERE ssn IN (
        SELECT ssn
        FROM WORK_ON
        GROUP BY ssn
        HAVING COUNT(DISTINCT pno) >= 5
    );
END;


exec GetPromotionCandidates;
```

sheet 14

assertion, cursor,views along with their output.

Create database using following schema. Apply given Integrity Constraints and answer the following queries using SQL. (Fill up database with at least 10 records in eaCh table).

EMPLOYEE (fna.me, Mame, ssn, sex, salary, joindate.superssn, dno4) DEPT (dname, dnum, mgrssn, dlocation)

PROJECT (pname, pno, plocation, dnumber)
WORK_ ON (ssn, pno, hours)

Integrity Constraints:

- The values of any attributes should not be null.
- The deptno should less than 4 digit

- Ssn (social_security_no of employee)
- Mgrssn(manager_social_security_no)
- Superssn(supervisor_social_security_no)

Queries:

a) Find the ssn of all employees who work on pno 101, 102 or 103.

b) Make a list of all pno for project that involve an employee whose last name is 'sonar' either as a worker or as a manager of the dept that control the project.

a) Write a trigger on insert on WORK_ON table such that if total work hours of employee in company is less than 20 hours then his salary is deducted.

a) Write a cursor on PROJECT table to fetch the first row from the table & display the total number of rows present in the table.

Create a data report showing the informatio.n of all the projects and names of employees working on individual projects.

```
CREATE DATABASE CompanyDB;
USE CompanyDB;

CREATE TABLE DEPT (
    dname VARCHAR(50) NOT NULL,
    dnum INT NOT NULL CHECK (dnum < 10000),
    mgrssn CHAR(9) NOT NULL,
    dlocation VARCHAR(50) NOT NULL,
    PRIMARY KEY (dnum)
);

CREATE TABLE EMPLOYEE (
    fname VARCHAR(30) NOT NULL,
    lname VARCHAR(30) NOT NULL,
    ssn CHAR(9) NOT NULL PRIMARY KEY,
    sex CHAR(1) NOT NULL,
    salary DECIMAL(10, 2) NOT NULL,
    joindate DATE NOT NULL,
    superssn CHAR(9) NOT NULL,
    dno INT NOT NULL,
    FOREIGN KEY (superssn) REFERENCES EMPLOYEE(ssn),
```

```
    FOREIGN KEY (dno) REFERENCES DEPT(dnum)
);

CREATE TABLE PROJECT (
    pname VARCHAR(50) NOT NULL,
    pno INT NOT NULL PRIMARY KEY,
    plocation VARCHAR(50) NOT NULL,
    dnumber INT NOT NULL,
    FOREIGN KEY (dnumber) REFERENCES DEPT(dnum)
);

CREATE TABLE WORK_ON (
    ssn CHAR(9) NOT NULL,
    pno INT NOT NULL,
    hours DECIMAL(5,2) NOT NULL,
    PRIMARY KEY (ssn, pno),
    FOREIGN KEY (ssn) REFERENCES EMPLOYEE(ssn),
    FOREIGN KEY (pno) REFERENCES PROJECT(pno)
);


INSERT INTO DEPT VALUES
('HR', 101, '111111111', 'New York'),
('Finance', 102, '222222222', 'Chicago'),
('IT', 103, '333333333', 'Dallas');

-- EMPLOYEE
INSERT INTO EMPLOYEE VALUES
('John', 'Smith', '123456789', 'M', 50000, '2020-01-01', '123456789', 101),
('Alice', 'Jones', '223456789', 'F', 52000, '2020-02-01', '123456789', 101),
('Rahul', 'Sonar', '323456789', 'M', 53000, '2021-03-01', '123456789', 102),
('Emma', 'Stone', '423456789', 'F', 54000, '2021-04-01', '223456789', 102),
('Liam', 'Brown', '523456789', 'M', 55000, '2021-05-01', '323456789', 102),
('Mia', 'Taylor', '623456789', 'F', 56000, '2021-06-01', '423456789', 103),
('Noah', 'Lee', '723456789', 'M', 57000, '2021-07-01', '523456789', 103),
('Ava', 'Clark', '823456789', 'F', 58000, '2021-08-01', '623456789', 103),
('Oliver', 'Hill', '923456789', 'M', 59000, '2021-09-01', '723456789', 103),
('Sophia', 'Scott', '133456789', 'F', 60000, '2021-10-01', '823456789', 101);

-- PROJECT
INSERT INTO PROJECT VALUES
('Recruitment', 101, 'New York', 101),
('Budgeting', 102, 'Chicago', 102),
('AI Development', 103, 'Dallas', 103),
('Payroll System', 104, 'Chicago', 102),
('Cloud Upgrade', 105, 'Dallas', 103);

-- WORK_ON
INSERT INTO WORK_ON VALUES
('123456789', 101, 10),
('223456789', 101, 8),
('323456789', 102, 6),
('423456789', 103, 5),
('523456789', 103, 4),
('623456789', 104, 12),
('723456789', 105, 14),
('823456789', 104, 5),
('923456789', 105, 6),
('133456789', 101, 7);


--a)----------------------------->
SELECT ssn
FROM WORK_ON
WHERE pno IN (101, 102, 103);



--b)----------------------------->
SELECT DISTINCT P.pno
FROM PROJECT P
JOIN DEPT D ON P.dnumber = D.dnum
LEFT JOIN EMPLOYEE E1 ON D.mgrssn = E1.ssn
LEFT JOIN WORK_ON W ON P.pno = W.pno
LEFT JOIN EMPLOYEE E2 ON W.ssn = E2.ssn
WHERE E1.lname = 'Sonar' OR E2.lname = 'Sonar';



--c)----------------------------->
CREATE TRIGGER trg_SetDueDate
ON ISSUETABLE
INSTEAD OF INSERT
AS
BEGIN
```

```
  INSERT INTO ISSUETABLE(issueid, accession_no, stud_enrollno, issuedate, duedate, ret_date, bid)
  SELECT issueid, accession_no, stud_enrollno, issuedate,
         DATEADD(DAY, 7, issuedate), ret_date, bid
  FROM INSERTED;
END;


INSERT INTO STUDENTMASTER VALUES (111, 'Vikas', '12th', 'Computer');
INSERT INTO BOOKMASTER VALUES (11, 'Deep Learning', 'Author K', 1200);
INSERT INTO ACCESSIONTABLE VALUES (11, 1011, 'T');

-- Issue book (trigger sets duedate)
INSERT INTO ISSUETABLE(issueid, accession_no, stud_enrollno, issuedate, ret_date, bid)
VALUES (211, 1011, 111, '2025-06-01', NULL, 11);



--d)----------------------------->

CREATE PROCEDURE ProjectCursorProc
AS
BEGIN
    DECLARE @pname VARCHAR(50);
    DECLARE project_cursor CURSOR FOR SELECT pname FROM PROJECT;

    OPEN project_cursor;
    FETCH NEXT FROM project_cursor INTO @pname;

    PRINT 'First project: ' + @pname;

    CLOSE project_cursor;
    DEALLOCATE project_cursor;

    SELECT COUNT(*) AS TotalProjects FROM PROJECT;
END;


EXEC ProjectCursorProc;
```

Sheet 15

Create database using following schema. Apply given Integrity Constraints and answer the following queries using SQL. (Fill up database with at least 10 records in each table).

BOOKMASTER (bid, title, author, price)
STUDENTMASTER (stud enrollno, sname,class,dept)
ACCESSIONTABLE (bid, accession_no,avail)
ISSUETABLE(issueid,accession no,stud enrollno,issuedate,cluedate, ret_date,bid)

Integrity Constraints:

- The values of any attributes should not be null.
- Avail should be T (if book is not issue) or F (if book is issue)

Queries:

a) Find the name of books which is issued maximum times.
b) Find the detail information of books that are issued by computer department students.
c) Write a procedure to calculate the fines for the books which are not return on or before due date. no.of days = (ret_date - due_date) fine = no.of days (ret_date - due_date) * 10
d) Write a trigger on insert of ISSUETABLE such that
duedate = issuedate + 7

```sql
CREATE DATABASE LibraryDB;
USE LibraryDB;

CREATE TABLE BOOKMASTER (
  bid INT PRIMARY KEY,
  title VARCHAR(100) NOT NULL,
  author VARCHAR(100) NOT NULL,
  price DECIMAL(8,2) NOT NULL
);

CREATE TABLE STUDENTMASTER (
  enrollno INT PRIMARY KEY,
  sname VARCHAR(50) NOT NULL,
  class VARCHAR(20) NOT NULL,
  dept VARCHAR(50) NOT NULL
);



CREATE TABLE ACCESSIONTABLE (
  bid INT NOT NULL FOREIGN KEY REFERENCES BOOKMASTER(bid),
  accession_no INT PRIMARY KEY,
  avail CHAR(1) NOT NULL CHECK (avail IN ('T', 'F'))
);

CREATE TABLE ISSUETABLE (
  issueid INT PRIMARY KEY,
  accession_no INT NOT NULL FOREIGN KEY REFERENCES ACCESSIONTABLE(accession_no),
  stud_enrollno INT NOT NULL FOREIGN KEY REFERENCES STUDENTMASTER(enrollno),
  issuedate DATE NOT NULL,
  duedate DATE NOT NULL,
  ret_date DATE NULL,
  bid INT NOT NULL FOREIGN KEY REFERENCES BOOKMASTER(bid)
);


INSERT INTO BOOKMASTER VALUES
(1,'Learn SQL','Author A',500),(2,'Master C#','Author B',600),
(3,'Data Structures','Author C',700),(4,'Web Dev','Author D',650),
(5,'Algorithms','Author E',800),(6,'Networking','Author F',750),
(7,'Databases','Author G',550),(8,'AI Basics','Author H',900),
(9,'Machine Learning','Author I',950),(10,'Cloud Computing','Author J',1000);

INSERT INTO STUDENTMASTER VALUES
(101,'Amit','10th','Computer'),(102,'Pooja','11th','Electronics'),
(103,'Ravi','12th','Computer'),(104,'Sneha','10th','EEE'),
(105,'Rohan','11th','Mechanical'),(106,'Deepa','12th','Computer'),
(107,'Meena','10th','Civil'),(108,'Arjun','11th','Computer'),
(109,'Kiran','12th','Chemical'),(110,'Anjali','11th','Computer');

INSERT INTO ACCESSIONTABLE VALUES
(1,1001,'T'),(2,1002,'T'),(3,1003,'F'),
(4,1004,'T'),(5,1005,'T'),(6,1006,'F'),
(7,1007,'T'),(8,1008,'T'),(9,1009,'T'),
(10,1010,'F');

INSERT INTO ISSUETABLE VALUES
(201,1003,101,'2025-05-01','2025-05-08','2025-05-07',3),
(202,1006,102,'2025-05-02','2025-05-09',NULL,6),
(203,1009,103,'2025-05-03','2025-05-10','2025-05-12',9),
(204,1004,104,'2025-05-04','2025-05-11','2025-05-09',4),
(205,1005,105,'2025-05-05','2025-05-12',NULL,5),
(206,1001,106,'2025-05-06','2025-05-13','2025-05-14',1),
(207,1002,107,'2025-05-07','2025-05-14','2025-05-13',2),
(208,1008,108,'2025-05-08','2025-05-15',NULL,8),
(209,1007,109,'2025-05-09','2025-05-16','2025-05-16',7),
(210,1010,110,'2025-05-10','2025-05-17',NULL,10);



--a)----------------------------->
SELECT TOP 1 B.title, COUNT(*) AS issued_count
FROM ISSUETABLE I
JOIN BOOKMASTER B ON I.bid = B.bid
GROUP BY B.title
ORDER BY COUNT(*) DESC;


--b)----------------------------->
SELECT DISTINCT B.*
FROM ISSUETABLE I
JOIN BOOKMASTER B ON I.bid = B.bid
JOIN STUDENTMASTER S ON I.stud_enrollno = S.enrollno
WHERE S.dept = 'Computer';
```

```sql
--c)----------------------------->
CREATE PROCEDURE CalcFines
AS
BEGIN
  SELECT I.issueid, I.stud_enrollno,
         DATEDIFF(DAY, I.duedate, I.ret_date) AS days_late,
         CASE
           WHEN I.ret_date > I.duedate THEN DATEDIFF(DAY, I.duedate, I.ret_date)*10
           ELSE 0 END AS fine_amount
  FROM ISSUETABLE I
  WHERE I.ret_date IS NOT NULL;
END;

exec CalcFines;



--d)----------------------------->
CREATE TRIGGER trg_SetDueDate
ON ISSUETABLE
INSTEAD OF INSERT
AS
BEGIN
  INSERT INTO ISSUETABLE(issueid, accession_no, stud_enrollno, issuedate, duedate, ret_date, bid)
  SELECT issueid, accession_no, stud_enrollno, issuedate,
         DATEADD(DAY, 7, issuedate), ret_date, bid
  FROM INSERTED;
END;


-- Insert a new student (validations ensured by schema)
INSERT INTO STUDENTMASTER VALUES
(111, 'Vikas', 'Indore', '12th', 'Computer');

-- Insert a new book and accession
INSERT INTO BOOKMASTER VALUES (11,'Deep Learning','Author K',1200);
INSERT INTO ACCESSIONTABLE VALUES (11,1011,'T');

-- Issue the book (duedate auto-set by trigger)
INSERT INTO ISSUETABLE(issueid, accession_no, stud_enrollno, issuedate, ret_date, bid)
VALUES (211, 1011, 111, '2025-06-01', NULL, 11);
```

Sheet 16

Create database using following schema. Apply given Integrity Constraints and answer the following queries using SQL. (Fill up database with at least 10 records in each table).

BOOKMASTER (bid, title, author, price)
STUDENTMASTER (stud enrollno, snaine,class,dept)
ACCESSIONTABLE (accession no,avail)
fiMUETABLE(issueid,as_ussion no,stud earalno,issuedate,duedate, ret_date,bi,d)

Integrity Constraints:

- The values of any attributes should not be null.
- Avail should be T (if book is not issue) or 1' (if book is issue)

Queries:

a) Find the detail information of the students who have issued books Between two given dates.
b) Create a view that display all the accession infortnatiOn for a book having bid = 100
c) Write a cursor to fetch last record from view **in (b).**
d) Find the information of books issued by MCA students.

Create a input form for new book issue. Apply all possible validations.

```sql
CREATE DATABASE LibraryDB;
USE LibraryDB;

-- Book Master Table
CREATE TABLE BOOKMASTER (
    bid INT PRIMARY KEY,
    title VARCHAR(100) NOT NULL,
    author VARCHAR(100) NOT NULL,
    price DECIMAL(10,2) NOT NULL
);

-- Student Master Table
CREATE TABLE STUDENTMASTER (
    stud_enrollno INT PRIMARY KEY,
    sname VARCHAR(100) NOT NULL,
    class VARCHAR(20) NOT NULL,
    dept VARCHAR(50) NOT NULL
);

-- Accession Table
CREATE TABLE ACCESSIONTABLE (
    accession_no INT PRIMARY KEY,
    bid INT NOT NULL,
    avail CHAR(1) CHECK (avail IN ('T', 'F')) NOT NULL,
    FOREIGN KEY (bid) REFERENCES BOOKMASTER(bid)
);

-- Issue Table
CREATE TABLE ISSUETABLE (
    issueid INT PRIMARY KEY,
    accession_no INT NOT NULL,
    stud_enrollno INT NOT NULL,
    issuedate DATE NOT NULL,
    duedate DATE NOT NULL,
    ret_date DATE,
    bid INT NOT NULL,
    FOREIGN KEY (accession_no) REFERENCES ACCESSIONTABLE(accession_no),
    FOREIGN KEY (stud_enrollno) REFERENCES STUDENTMASTER(stud_enrollno),
    FOREIGN KEY (bid) REFERENCES BOOKMASTER(bid)
);

INSERT INTO BOOKMASTER VALUES
(100, 'Database Systems', 'Elmasri', 550),
(101, 'Operating Systems', 'Galvin', 620),
(102, 'Data Structures', 'Sahni', 450);


INSERT INTO STUDENTMASTER VALUES
(201, 'Ravi', 'MCA-I', 'MCA'),
(202, 'Anjali', 'MCA-II', 'MCA'),
(203, 'Karan', 'BSc-I', 'CS');

INSERT INTO ACCESSIONTABLE VALUES
(1001, 100, 'F'),
(1002, 101, 'T'),
(1003, 102, 'F');

INSERT INTO ISSUETABLE VALUES
(301, 1001, 201, '2025-06-01', '2025-06-08', NULL, 100),
(302, 1003, 202, '2025-06-02', '2025-06-09', NULL, 102),
(303, 1001, 203, '2025-06-03', '2025-06-10', NULL, 100);

--a)----------------------------->
SELECT S.*, I.issuedate
FROM STUDENTMASTER S
JOIN ISSUETABLE I ON S.stud_enrollno = I.stud_enrollno
WHERE I.issuedate BETWEEN '2025-06-01' AND '2025-06-05';

--b)----------------------------->
CREATE VIEW View_Accession_Book100 AS
SELECT A.*
FROM ACCESSIONTABLE A
WHERE bid = 100;

SELECT * FROM View_Accession_Book100;

--c)----------------------------->
CREATE PROCEDURE FetchLastAccession
AS
BEGIN
    DECLARE @accno INT, @avail CHAR(1);

    DECLARE acc_cursor CURSOR FOR
    SELECT accession_no, avail FROM View_Accession_Book100;
    OPEN acc_cursor;
```

```
    FETCH NEXT FROM acc_cursor INTO @accno, @avail;
    WHILE @@FETCH_STATUS = 0
    BEGIN
        FETCH NEXT FROM acc_cursor INTO @accno, @avail;
    END
    PRINT 'Last Accession No: ' + CAST(@accno AS VARCHAR) + ', Avail: ' + @avail;

    CLOSE acc_cursor;
    DEALLOCATE acc_cursor;
END;

EXEC FetchLastAccession;


--d)----------------------------->
SELECT B.*, S.sname, S.dept
FROM BOOKMASTER B
JOIN ISSUETABLE I ON B.bid = I.bid
JOIN STUDENTMASTER S ON S.stud_enrollno = I.stud_enrollno
WHERE S.dept = 'MCA';
```

Sheet 17

Create database using following schema. Apply given Integrity Constraints and answer the followingqueries using SQL. (Fill up database with at least 10 records ir+ eaoh,tab4e).

BOOKMASTER (bid, title, author, price)
STUDENTMASTER (stud enroll no, sname,class,dept)
ACCESSIONTABLE (bid, accession no,avail)
SUETABLE(issueid,accessipn no.stl.KLmrsillag,issuedate,cluedate, ret_date,hisp

Integrity Constraints:

- The values of any attributes should not be null.
- Avail should be T (if book is not issue) or F (if book i8 issue)

Queries:

a) Write a procedure for giving the detail information of books available in the library.
a) Find the number of books issued by each student.
a) Write a trigger such that the return date should not exceed today's date.
b) Find the number of books available in the library & written by "Henry Korth".

```
CREATE DATABASE LibraryDB;
USE LibraryDB;

CREATE TABLE BOOKMASTER (
  bid INT PRIMARY KEY,
  title VARCHAR(100) NOT NULL,
  author VARCHAR(100) NOT NULL,
  price DECIMAL(8,2) NOT NULL
);

CREATE TABLE STUDENTMASTER (
  stud_enrollno INT PRIMARY KEY,
  sname VARCHAR(50) NOT NULL,
  class VARCHAR(20) NOT NULL,
  dept VARCHAR(50) NOT NULL
);

CREATE TABLE ACCESSIONTABLE (
  accession_no INT PRIMARY KEY,
  bid INT NOT NULL,
  avail CHAR(1) CHECK (avail IN ('T', 'F')) NOT NULL,
  FOREIGN KEY (bid) REFERENCES BOOKMASTER(bid)
);
```

```sql
CREATE TABLE ISSUETABLE (
  issueid INT PRIMARY KEY,
  accession_no INT NOT NULL,
  stud_enrollno INT NOT NULL,
  issuedate DATE NOT NULL,
  duedate DATE NOT NULL,
  ret_date DATE,
  bid INT NOT NULL,
  FOREIGN KEY (accession_no) REFERENCES ACCESSIONTABLE(accession_no),
  FOREIGN KEY (stud_enrollno) REFERENCES STUDENTMASTER(stud_enrollno),
  FOREIGN KEY (bid) REFERENCES BOOKMASTER(bid)
);

INSERT INTO BOOKMASTER VALUES
(100, 'Database Systems', 'Elmasri', 550),
(101, 'Operating Systems', 'Galvin', 620),
(102, 'Data Structures', 'Sahni', 450);


INSERT INTO STUDENTMASTER VALUES
(201, 'Ravi', 'MCA-I', 'MCA'),
(202, 'Anjali', 'MCA-II', 'MCA'),
(203, 'Karan', 'BSc-I', 'CS');

INSERT INTO ACCESSIONTABLE VALUES
(1001, 100, 'F'),
(1002, 101, 'T'),
(1003, 102, 'F');

INSERT INTO ISSUETABLE VALUES
(301, 1001, 201, '2025-06-01', '2025-06-08', NULL, 100),
(302, 1003, 202, '2025-06-02', '2025-06-09', NULL, 102),
(303, 1001, 203, '2025-06-03', '2025-06-10', NULL, 100);

--a)----------------------------->
CREATE PROCEDURE ShowAvailableBooks
AS
BEGIN
  SELECT B.*
  FROM BOOKMASTER B
  JOIN ACCESSIONTABLE A ON B.bid = A.bid
  WHERE A.avail = 'T';
END;
 EXEC ShowAvailableBooks;

--b)----------------------------->
SELECT stud_enrollno, COUNT(*) AS books_issued
FROM ISSUETABLE
GROUP BY stud_enrollno;

--c)----------------------------->
CREATE TRIGGER trg_CheckReturnDate
ON ISSUETABLE
AFTER INSERT
AS
BEGIN
  IF EXISTS (SELECT 1 FROM INSERTED WHERE ret_date > GETDATE())
    print('Return date cannot exceed today.');
END;


--d)----------------------------->
SELECT COUNT(*) AS total_books
FROM BOOKMASTER B
JOIN ACCESSIONTABLE A ON B.bid = A.bid
WHERE B.author = 'Henry Korth' AND A.avail = 'T';
```

Create database using following schema. Apply given Integrity Constraints and answer the following queries using SQL.(Fill up database with at least 10 records in each table).

PRODUCT (Maker, Modelno, Type)
PC (IvIodeln, Speed, RAM, HD, CD, Price)
LAPTOP (Rh, Speed, RAM, HD, Price)
PRINTER (Model, Color, Type, Price)

Details regarding Schemas

- PC relation contains model no. of PC, its speed in MHz, RAM in MB, HD size in GB, Speed of CD reader, and price.
- The value for Maker in Product table can be IBM, Compaq,etc.
- PRINTER relation contains model no., value of Color should be T(if printer is color) or F (if printer is black & white), type(laser, ink-jet, dot- matrix or dry), and price.

Integrity Constraints:

- The values of any attributes should not be null.
- Product Type should one of these (PC, Laptop or Printer)

Queries:

a) Find the manufacturers of color printers.
b) Find the laptops whose speed is slower than that of any PC.
c) Express the following constraint as SQL assertions -
No black & white printer should have price greater than color printers."
d) write a trigger on PC & LAPTOP table such that the hard disk size should be greater than 20 GB

```sql
CREATE DATABASE ComputerStore;
USE ComputerStore;

CREATE TABLE PRODUCT (
    Maker VARCHAR(50) NOT NULL,
    ModelNo INT PRIMARY KEY,
    Type VARCHAR(20) NOT NULL CHECK (Type IN ('PC', 'Laptop', 'Printer'))
);

CREATE TABLE PC (
    ModelNo INT PRIMARY KEY,
    Speed INT NOT NULL,
    RAM INT NOT NULL,
    HD INT NOT NULL CHECK (HD > 20),
    CD VARCHAR(10) NOT NULL,
    Price DECIMAL(10,2) NOT NULL,
    FOREIGN KEY (ModelNo) REFERENCES PRODUCT(ModelNo)
);

CREATE TABLE LAPTOP (
    ModelNo INT PRIMARY KEY,
    Speed INT NOT NULL,
    RAM INT NOT NULL,
    HD INT NOT NULL CHECK (HD > 20),
    Price DECIMAL(10,2) NOT NULL,
    FOREIGN KEY (ModelNo) REFERENCES PRODUCT(ModelNo)
);

CREATE TABLE PRINTER (
    ModelNo INT PRIMARY KEY,
    Color CHAR(1) NOT NULL CHECK (Color IN ('T','F')),
    Type VARCHAR(20) NOT NULL CHECK (Type IN ('Laser','Ink-Jet','Dot-Matrix','Dry')),
    Price DECIMAL(10,2) NOT NULL,
    FOREIGN KEY (ModelNo) REFERENCES PRODUCT(ModelNo)
);

INSERT INTO PRODUCT VALUES
('IBM', 101, 'PC'),
('HP', 102, 'Laptop'),
('Canon', 103, 'Printer'),
('Lenovo', 104, 'Laptop'),
('Dell', 105, 'PC'),
('Canon', 201, 'Printer'),
```

```
('HP', 202, 'Printer'),
('Epson', 203, 'Printer'),
('Brother', 204, 'Printer'),
('Samsung', 205, 'Printer');

INSERT INTO PC VALUES
(101, 2400, 4, 160, '52x', 25000),
(105, 2800, 8, 500, '48x', 32000),
(104, 2200, 4, 250, '40x', 27000);

INSERT INTO LAPTOP VALUES
(102, 1800, 2, 80, 22000),
(104, 1500, 4, 120, 30000),
(105, 2000, 8, 250, 35000),
(101, 1700, 4, 160, 27000);

-- PRINTER
INSERT INTO PRINTER VALUES
(103, 'T', 'Laser', 15000),
(104, 'F', 'Ink-Jet', 7000),
(102, 'T', 'Dot-Matrix', 13000),
(201, 'F', 'Laser', 12000),
(202, 'F', 'Ink-Jet', 7000),
(203, 'T', 'Laser', 11000),
(204, 'T', 'Dot-Matrix', 15000),
(205, 'F', 'Dry', 13000);


--a)------------------------------>
SELECT DISTINCT p.Maker
FROM PRODUCT p
JOIN PRINTER pr ON p.ModelNo = pr.ModelNo
WHERE pr.Color = 'T';


--b)------------------------------>

SELECT *
FROM Laptop
WHERE Speed < ALL (SELECT Speed FROM PC);


--dc)------------------------------>
SELECT *
FROM PRINTER
WHERE Color = 'F'
AND Price > (SELECT MIN(Price) FROM PRINTER WHERE Color = 'T');


--d)------------------------------>

-- For PC
CREATE TRIGGER trg_pc_hd_check
ON PC
INSTEAD OF INSERT
AS
BEGIN
    IF EXISTS (SELECT * FROM inserted WHERE HD <= 20)
          begin
        print('Hard disk size must be greater than 20GB for PCs.');
                rollback;
          end
    ELSE
        INSERT INTO PC SELECT * FROM inserted;
END
GO

INSERT INTO PC VALUES (302, 2400, 4, 20, '48x', 26000);

-- For Laptop
CREATE TRIGGER trg_laptop_hd_check
ON LAPTOP
INSTEAD OF INSERT
AS
BEGIN
    IF EXISTS (SELECT * FROM inserted WHERE HD <= 20)
        print('Hard disk size must be greater than 20GB for Laptops.');
    ELSE
        INSERT INTO LAPTOP SELECT * FROM inserted;
END
GO

INSERT INTO PRODUCT VALUES ('Asus', 402, 'Laptop');

INSERT INTO LAPTOP VALUES (402, 1800, 4, 20, 30000);
```

Create database using following schema. Apply given Integrity Constraints and answer the following queries using SQL. (Fill up database with at least 10 records in each table).

DOCTOR (Did, Dname, Daddress, qualification)
PATIENTMASTER (Pcode, Pname, Padd, age, gender, bloodgroup, Did) ADMITTEDPATIENT (Pcode, Entry date, Discharge_clate, wardno, disease)

Integrity Constraints:

- The values of any attributes should not be null.
- Gender value should be M (male) or F(feinale).
- Wardno should be less than 6. Queries:

a) Find the details of patient who are admitted within the period 03/03/08 to 25/ 03/08.
b) Find the names of doctors who are treating Jalf_zaon patients.
c) write a procedure on ADMITTEDPATIENT table such as to calculate the bill of all patients currently admitted in the hospital.
(bill = no _ of_ days * 500)
d) Write a trigger on Doctor table such that the specialization should be :- M.B.B.S./B.A.M.S/M.S.

Create a data entry form for new DOCTOR. Apply all possible validations.

```sql
CREATE DATABASE HospitalDB;
USE HospitalDB;

-- DOCTOR Table
CREATE TABLE DOCTOR (
    Did INT PRIMARY KEY,
    Dname VARCHAR(50) NOT NULL,
    Daddress VARCHAR(100) NOT NULL,
    Qualification VARCHAR(20) NOT NULL
);

-- PATIENTMASTER Table
CREATE TABLE PATIENTMASTER (
    Pcode INT PRIMARY KEY,
    Pname VARCHAR(50) NOT NULL,
    Padd VARCHAR(100) NOT NULL,
    Age INT NOT NULL,
    Gender CHAR(1) NOT NULL CHECK (Gender IN ('M', 'F')),
    BloodGroup VARCHAR(5) NOT NULL,
    Did INT NOT NULL FOREIGN KEY REFERENCES DOCTOR(Did)
);

-- ADMITTEDPATIENT Table
CREATE TABLE ADMITTEDPATIENT (
    Pcode INT NOT NULL FOREIGN KEY REFERENCES PATIENTMASTER(Pcode),
    EntryDate DATE NOT NULL,
    Discharge_Date DATE,
    WardNo INT NOT NULL CHECK (WardNo < 6),
    Disease VARCHAR(50) NOT NULL
);


-- DOCTOR
INSERT INTO DOCTOR VALUES
(1, 'Dr. Sharma', 'Delhi', 'M.B.B.S.'),
(2, 'Dr. Mehta', 'Mumbai', 'B.A.M.S.'),
(3, 'Dr. Roy', 'Kolkata', 'M.S.'),
(4, 'Dr. Verma', 'Pune', 'M.B.B.S.'),
(5, 'Dr. Reddy', 'Hyderabad', 'B.A.M.S.'),
(6, 'Dr. Khan', 'Jaipur', 'M.S.'),
(7, 'Dr. Patel', 'Ahmedabad', 'M.B.B.S.'),
(8, 'Dr. Das', 'Bhubaneswar', 'M.B.B.S.'),
```

```
(9, 'Dr. Joshi', 'Nagpur', 'B.A.M.S.'),
(10, 'Dr. Iqbal', 'Lucknow', 'M.S.');

-- PATIENTMASTER
INSERT INTO PATIENTMASTER VALUES
(101, 'Amit Jain', 'Delhi', 25, 'M', 'B+', 1),
(102, 'Riya Sen', 'Kolkata', 30, 'F', 'O+', 2),
(103, 'Sameer Rao', 'Mumbai', 22, 'M', 'A-', 3),
(104, 'Kavita Yadav', 'Lucknow', 28, 'F', 'B-', 4),
(105, 'Rohan Singh', 'Jaipur', 35, 'M', 'AB+', 5),
(106, 'Pooja Shah', 'Ahmedabad', 27, 'F', 'O-', 6),
(107, 'Rahul Das', 'Bhubaneswar', 40, 'M', 'A+', 7),
(108, 'Sneha Reddy', 'Hyderabad', 19, 'F', 'B+', 8),
(109, 'Zoya Khan', 'Jalgaon', 32, 'F', 'A+', 9),
(110, 'Vikas Mehra', 'Delhi', 45, 'M', 'AB-', 10);

-- ADMITTEDPATIENT
INSERT INTO ADMITTEDPATIENT VALUES
(101, '2008-03-01', '2008-03-10', 2, 'Malaria'),
(102, '2008-03-04', '2008-03-15', 1, 'Typhoid'),
(103, '2008-03-06', NULL, 3, 'COVID-19'),
(104, '2008-02-25', '2008-03-05', 2, 'Dengue'),
(105, '2008-03-10', NULL, 1, 'Flu'),
(106, '2008-03-20', '2008-03-25', 4, 'Malaria'),
(107, '2008-03-03', '2008-03-11', 2, 'TB'),
(108, '2008-04-01', NULL, 3, 'Injury'),
(109, '2008-03-15', '2008-03-21', 5, 'Fever'),
(110, '2008-03-24', NULL, 1, 'Cancer');


--a)------------------------------>

SELECT *
FROM ADMITTEDPATIENT
WHERE EntryDate BETWEEN '2008-03-03' AND '2008-03-25';


--b)------------------------------>

SELECT DISTINCT D.Dname
FROM DOCTOR D
JOIN PATIENTMASTER P ON D.Did = P.Did
WHERE P.Padd LIKE '%Jalgaon%';


--c)------------------------------>

CREATE PROCEDURE CalculateBills
AS
BEGIN
    SELECT
        ap.Pcode,
        pm.Pname,
        DATEDIFF(DAY, ap.EntryDate, ISNULL(ap.Discharge_Date, GETDATE())) AS Days,
        DATEDIFF(DAY, ap.EntryDate, ISNULL(ap.Discharge_Date, GETDATE())) * 500 AS Bill
    FROM ADMITTEDPATIENT ap
    JOIN PATIENTMASTER pm ON ap.Pcode = pm.Pcode
    WHERE ap.Discharge_Date IS NULL;
END;
GO

-- Execute the procedure
EXEC CalculateBills;



--d)------------------------------>
CREATE TRIGGER trg_DoctorQualification
ON DOCTOR
INSTEAD OF INSERT
AS
BEGIN
    IF EXISTS (
        SELECT * FROM inserted
        WHERE Qualification NOT IN ('M.B.B.S.', 'B.A.M.S.', 'M.S.')
    )begin
        PRINT 'Invalid Qualification! Only M.B.B.S., B.A.M.S., or M.S. allowed.';
                rollback transaction;
         end
    ELSE
        INSERT INTO DOCTOR SELECT * FROM inserted;
END;
GO

INSERT INTO DOCTOR VALUES (11, 'Dr. Fake', 'Nowhere', 'Ph.D.');
```

```sql
INSERT INTO DOCTOR VALUES (12, 'Dr. Real', 'Chennai', 'M.S.');
```