create database practice\_lab;

use practice\_lab;

-- 1st one ------------------------------------------------------

Create table EMPLOYEE(

Essn int(9),

Ename varchar(50),

Dept\_No int(10),

Salary int not null,

date\_of\_join date not null,

primary key(Essn,Ename)

);

desc EMPLOYEE;

Insert into EMPLOYEE values(101,'chetan',1,75000,'2000-02-10');

Insert into EMPLOYEE values(102,'laxman',1,85000,'2000-02-08');

Insert into EMPLOYEE values(103,'Ajay',1,45000,'2000-08-10');

Insert into EMPLOYEE values(104,'Akshay',2,35000,'2000-05-10');

Insert into EMPLOYEE values(105,'Akil',1,95000,'2000-05-09');

create table DEPENDENT(

Essn int(9) primary key,

Depend\_Name varchar(50) not null,

Relation varchar(50) not null,

Dob date not null,

foreign key (Essn) references EMPLOYEE(Essn)

);

Insert into DEPENDENT values(101,'Veeresh','Brother','2002-04-14');

Insert into DEPENDENT values(102,'Abhi','Father','2000-03-23');

Insert into DEPENDENT values(103,'Munni','Mother','2002-04-25');

Insert into DEPENDENT values(104,'Anil','Father','2002-04-21');

create table DEPARTMENT(

Dept\_NO int(10),

Dept\_Name varchar(50),

Manager int(10),

primary key (Dept\_No,Dept\_Name),

foreign key(Manager) references EMPLOYEE(Essn)

);

insert into DEPARTMENT values(1,'HR',101);

insert into DEPARTMENT values(2,'Sales',102);

insert into DEPARTMENT values(3,'Coding',101);

insert into DEPARTMENT values(4,'Act',101);

-- i. Find details of dependents for employee having name AJAY

select \*

from DEPENDENT D,EMPLOYEE E

where D.Essn=E.Essn and E.Ename='Ajay';

-- ii. Find the name of the manager of the department in which employee with ESSN Code 5078 works.

select Ename

from EMPLOYEE e, DEPARTMENT d

where e.Essn=d.Manager and e.Essn=101;

-- iii. Find the name of all employees whose age is less than 18 years.

-- this is wrong in employee table we dont have dob attribute so it will not work if you take dob then below query will work

SELECT Ename

FROM EMPLOYEE

WHERE DATEDIFF(CURDATE(), date\_of\_birth) < 18 \* 365.25;

-- iv. Find the DOB of the son of the employee having employee code ESSN 5078.

select Dob

from EMPLOYEE e,DEPENDENT d

where e.Essn=d.Essn and Relation='Brother' and e.Essn=101;

-- v. Display the list of employees who joined in the specific year.

select Ename

from EMPLOYEE

where year(date\_of\_join) =2000;

-- vi. Find the details of the departments in which the employee having experience of at least ten years.

select d.Dept\_No,d.Dept\_Name,d.Manager

From DEPARTMENT d,EMPLOYEE e

where d.Dept\_No=e.Dept\_No and datediff(curdate(),date\_of\_join)/365.25 >=10;

-- vii. Find number of employees in each department

select Dept\_No,Count(\*) as No\_Of\_Employees

from EMPLOYEE

group by Dept\_No;

-- viii. Find the employee whose salary is greater than the salary of a manager.

select Ename

from EMPLOYEE

where Salary > (select max(Salary)

from EMPLOYEE e,DEPARTMENT d

where e.Essn=d.Manager);

-- 2 one-------------------------------------------------------------------------

CREATE TABLE Faculty (

F\_id INT PRIMARY KEY, -- Faculty ID, primary key

F\_name VARCHAR(100) NOT NULL, -- Faculty name, cannot be NULL

F\_designation VARCHAR(50) NOT NULL -- Faculty designation

);

CREATE TABLE Patient (

P\_id INT PRIMARY KEY, -- Patient ID, primary key

P\_name VARCHAR(100) NOT NULL, -- Patient name

p\_address VARCHAR(255) NOT NULL, -- Patient address

date\_of\_registration DATE NOT NULL -- Date of registration

);

CREATE TABLE Consultation (

C\_date DATE NOT NULL, -- Consultation date

F\_id INT, -- Foreign key from Faculty

P\_id INT, -- Foreign key from Patient

PRIMARY KEY (C\_date, F\_id, P\_id),

FOREIGN KEY (F\_id) REFERENCES Faculty(F\_id),

FOREIGN KEY (P\_id) REFERENCES Patient(P\_id)

);

-- Insert data into Faculty table

INSERT INTO Faculty (F\_id, F\_name, F\_designation)

VALUES (1, 'Dr. Veeresh', 'Senior Doctor'),

(2, 'Dr. Akshay', 'General Physician'),

(3, 'Dr. Gupta', 'Senior Doctor'),

(4, 'Dr. Patel', 'Junior Doctor'),

(5, 'Dr. Verma', 'Senior Doctor');

-- Insert data into Patient table

INSERT INTO Patient (P\_id, P\_name, p\_address, date\_of\_registration)

VALUES (101, 'Mallu', '123 Main St', '2022-01-15'),

(102, 'Gillu', '456 Oak St', '2022-02-20'),

(103, 'Tokio', '789 Pine St', '2022-03-05'),

(104, 'Ghanta', '321 Maple St', '2022-04-10'),

(105, 'Ramu', '654 Cedar St', '2022-05-25');

-- Insert data into Consultation table

INSERT INTO Consultation (C\_date, F\_id, P\_id)

VALUES ('2023-10-01', 1, 101),

('2023-10-01', 2, 102),

('2023-11-10', 3, 103),

('2023-11-15', 4, 104),

('2023-12-01', 5, 105);

-- i. Generate list of patients and their consultation detail.

SELECT P\_name, C\_date, F\_name

FROM Patient p, Consultation c, Faculty f

WHERE p.P\_id = c.P\_id AND c.F\_id = f.F\_id;

-- ii. Find patients consulted by specific faculty.

SELECT P\_name

FROM Patient p, Consultation c, Faculty f

WHERE p.P\_id = c.P\_id AND c.F\_id = f.F\_id AND f.F\_name = 'Dr. Veeresh';

-- iii. Find the details of the entire faculty whose designation is a senior doctor and have consultation date next month.

SELECT F\_name

FROM Faculty f, Consultation c

WHERE f.F\_id = c.F\_id AND f.F\_designation = 'Senior Doctor'

AND MONTH(c.C\_date) = MONTH(CURDATE()) + 1;

-- iv. Find the patient whose consultation date is today along with the concern faculty detail.

SELECT P\_name, F\_name

FROM Patient p, Consultation c, Faculty f

WHERE p.P\_id = c.P\_id AND c.F\_id = f.F\_id AND c.C\_date = '2023-10-01';

-- v. List the details of all patients who have got consultation dates fixed between 15th August to Sept. 2007.

SELECT P\_name

FROM Patient p, Consultation c

WHERE p.P\_id = c.P\_id AND c.C\_date BETWEEN '2023-01-01' AND '2023-12-31';

-- vi. Find the first patient registered in the center along with consultation details.

SELECT P\_name

FROM Patient p

WHERE p.date\_of\_registration = (SELECT MIN(date\_of\_registration) FROM Patient);

-- vii. Generate up to data list of faculty wise the number of consultations.

SELECT F\_name, COUNT(\*) as No\_of\_consultation

FROM Consultation c, Faculty f

WHERE c.F\_id = f.F\_id

GROUP BY F\_name;

-- viii. Generate the list of faculty wise consultation per month for the specific year.

SELECT F\_name, MONTH(c.C\_date), COUNT(\*) as Consultaion\_per\_month

FROM Consultation c, Faculty f

WHERE c.F\_id = f.F\_id AND YEAR(c.C\_date) = 2023

GROUP BY F\_name, MONTH(c.C\_date);

-- 3rd one---------------------------------------------------------------------

CREATE TABLE BRANCH (

Branchid INT PRIMARY KEY,

Branchname VARCHAR(50),

HOD VARCHAR(100)

);

DESC BRANCH;

CREATE TABLE STUDENT (

USN VARCHAR(20) PRIMARY KEY,

Name VARCHAR(100),

Address VARCHAR(200),

Branchid INT,

sem INT,

FOREIGN KEY (Branchid) REFERENCES BRANCH(Branchid)

);

DESC STUDENT;

CREATE TABLE AUTHOR (

Authorid INT PRIMARY KEY,

Authorname VARCHAR(100),

Country VARCHAR(100),

age INT

);

DESC AUTHOR;

CREATE TABLE BOOK (

Bookid INT PRIMARY KEY,

Bookname VARCHAR(100),

Authorid INT,

Publisher VARCHAR(100),

Branchid INT,

FOREIGN KEY (Authorid) REFERENCES AUTHOR(Authorid),

FOREIGN KEY (Branchid) REFERENCES BRANCH(Branchid)

);

DESC BOOK;

CREATE TABLE BORROW (

USN VARCHAR(20),

Bookid INT,

Borrowed\_Date DATE,

FOREIGN KEY (USN) REFERENCES STUDENT(USN),

FOREIGN KEY (Bookid) REFERENCES BOOK(Bookid)

);

DESC BORROW;

-- Inserting data into the BRANCH table

INSERT INTO BRANCH (Branchid, Branchname, HOD) VALUES

(1, 'MCA', 'Dr. Alice Johnson'),

(2, 'CSE', 'Dr. Bob Williams'),

(3, 'ECE', 'Dr. Charlie Brown');

-- Inserting data into the AUTHOR table

INSERT INTO AUTHOR (Authorid, Authorname, Country, Age) VALUES

(1, 'George Orwell', 'UK', 46),

(2, 'Mark Twain', 'USA', 60),

(3, 'J.K. Rowling', 'UK', 55),

(4, 'Dan Brown', 'USA', 56);

-- Inserting data into the STUDENT table

INSERT INTO STUDENT (USN, Name, Address, Branchid, Sem) VALUES

('S001', 'John Doe', '123 Oak Street', 1, 2),

('S002', 'Jane Smith', '456 Maple Avenue', 1, 2),

('S003', 'Mark White', '789 Elm Street', 2, 4),

('S004', 'Lucy Brown', '321 Pine Road', 1, 2),

('S005', 'Eva Blue', '654 Cedar Street', 3, 6),

('S006', 'Tom Black', '234 Birch Street', 1, 1),

('S007', 'Anna Green', '111 Willow Lane', 2, 4);

-- Inserting data into the BOOK table

INSERT INTO BOOK (Bookid, Bookname, Authorid, Publisher, Branchid) VALUES

(101, '1984', 1, 'Penguin', 1),

(102, 'The Adventures of Tom Sawyer', 2, 'HarperCollins', 2),

(103, 'Harry Potter and the Philosopher\'s Stone', 3, 'Bloomsbury', 1),

(104, 'The Da Vinci Code', 4, 'Random House', 3),

(105, 'Animal Farm', 1, 'Penguin', 1),

(106, 'Harry Potter and the Chamber of Secrets', 3, 'Bloomsbury', 1),

(107, 'Angels & Demons', 4, 'Random House', 3);

-- Inserting data into the BORROW table

INSERT INTO BORROW (USN, Bookid, Borrowed\_Date) VALUES

('S001', 101, '2024-10-01'),

('S002', 103, '2024-10-02'),

('S004', 105, '2024-09-29'),

('S002', 106, '2024-10-04'),

('S001', 102, '2024-10-05'),

('S003', 104, '2024-10-03');

-- i. List the details of Students who are all studying in 2nd sem MCA.

SELECT \*

FROM STUDENT

WHERE sem = 2 AND Branchid = 1;

-- ii. List the students who are not borrowed any books.

SELECT \*

FROM STUDENT

WHERE USN NOT IN (SELECT USN FROM BORROW);

-- iii. Display the USN, Student name, Branch\_name, Book\_name, Author\_name, Books\_Borrowed\_Date of 2nd sem MCA Students who borrowed books.

select s.USN,s.Name,b.Branchname,book.Bookname,a.Authorname,br.Borrowed\_Date

from STUDENT s,BRANCH b,BOOK book,AUTHOR a,BORROW br

where s.USN=br.USN and br.Bookid=book.Bookid and book.Authorid=a.Authorid and s.sem=2 and b.Branchname='MCA';

-- iv. Display the number of books written by each Author.

select a.Authorname,count(\*) as number\_of\_books\_written

from AUTHOR a,BOOK b

where a.Authorid=b.Authorid

group by Authorname;

-- v. Display the student details who borrowed more than two books.

select s.Name

from STUDENT s

where s.USN in (select USN from BORROW group by USN having count(Bookid)>1);

-- vi. Display the student details who borrowed books of more than one Author.

select s.Name

from STUDENT s

where s.USN in (select USN

FROM BORROW br,BOOK b

where br.Bookid=b.bookid group by br.USN having count(distinct b.Authorid)>1);

-- vii. Display the Book names in descending order of their names.

SELECT Bookname

FROM BOOK

ORDER BY Bookname DESC;

-- viii. List the details of students who borrowed the books which are all published by the same publisher.

SELECT S.\*

FROM STUDENT S

WHERE S.USN IN (

SELECT USN

FROM BORROW BR,BOOK B

where BR.Bookid = B.Bookid

GROUP BY BR.USN

HAVING COUNT(DISTINCT B.Publisher) = 1);

-- 4th------------------------------------------------------------

CREATE TABLE STUDENT4 (

USN VARCHAR(20) PRIMARY KEY,

Name VARCHAR(100),

Date\_of\_Birth DATE,

Branch VARCHAR(50),

Mark1 INT,

Mark2 INT,

Mark3 INT,

Total INT,

GPA DECIMAL(3, 2)

);

DESC STUDENT;

INSERT INTO STUDENT4 (USN, Name, Date\_of\_Birth, Branch, Mark1, Mark2, Mark3, Total, GPA) VALUES

('S001', 'Samir Kumar', '2002-05-21', 'CSE', 85, 90, 88, NULL, NULL),

('S002', 'Sara Lee', '2001-08-15', 'ECE', 80, 84, 89, NULL, NULL),

('S003', 'Ravi Sharma', '2000-07-11', 'CSE', 78, 92, 86, NULL, NULL),

('S004', 'Sophia Brown', '1999-12-25', 'IT', 89, 87, 90, NULL, NULL),

('S005', 'Shane Parker', '2002-02-18', 'ECE', 90, 92, 94, NULL, NULL),

('S006', 'John Doe', '2001-10-01', 'CSE', 88, 91, 93, NULL, NULL),

('S007', 'Akash Roy', '2000-11-13', 'ME', 75, 78, 82, NULL, NULL),

('S008', 'Sabrina Turner', '2002-03-29', 'CSE', 84, 88, 90, NULL, NULL),

('S009', 'Alan Walker', '2001-04-10', 'IT', 80, 85, 79, NULL, NULL);

select \* from Student4;

-- i. Update the column total by adding the columns mark1, mark2, mark3.

UPDATE STUDENT4

SET Total = Mark1 + Mark2 + Mark3;

-- ii. Find the GPA score of all the students.

UPDATE STUDENT4

SET GPA = Total / 30; -- Assuming a scale of 10 (example: GPA = Total/30)

-- iii. Find the students who born on a particular year of birth from the date\_of\_birth column.

SELECT \*

FROM STUDENT4

WHERE YEAR(Date\_of\_Birth) = 2001;

-- iv. List the students who are studying in a particular branch of study.

SELECT \*

FROM STUDENT4

WHERE Branch = 'CSE';

-- v. Find the maximum GPA score of the student branch-wise.

SELECT Branch, MAX(GPA) AS Max\_GPA

FROM STUDENT4

GROUP BY Branch;

-- vi. Find the students whose name starts with the alphabet “S”.

SELECT \*

FROM STUDENT4

WHERE Name LIKE 'S%';

-- vii. Find the students whose name ends with the alphabets “AR”.

SELECT \*

FROM STUDENT4

WHERE Name LIKE '%AR';

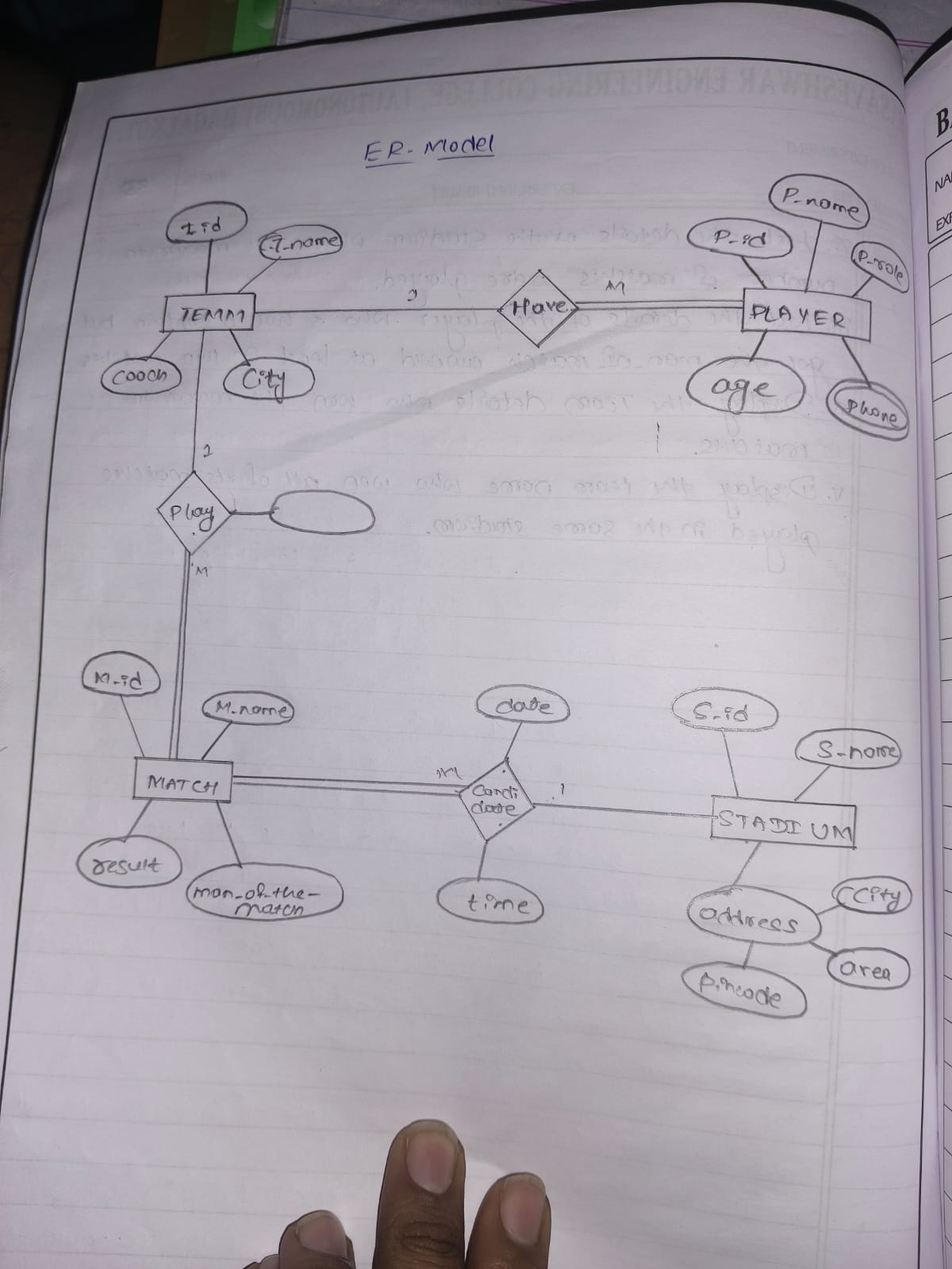
-- viii. Delete the student details whose USN is given as 1001

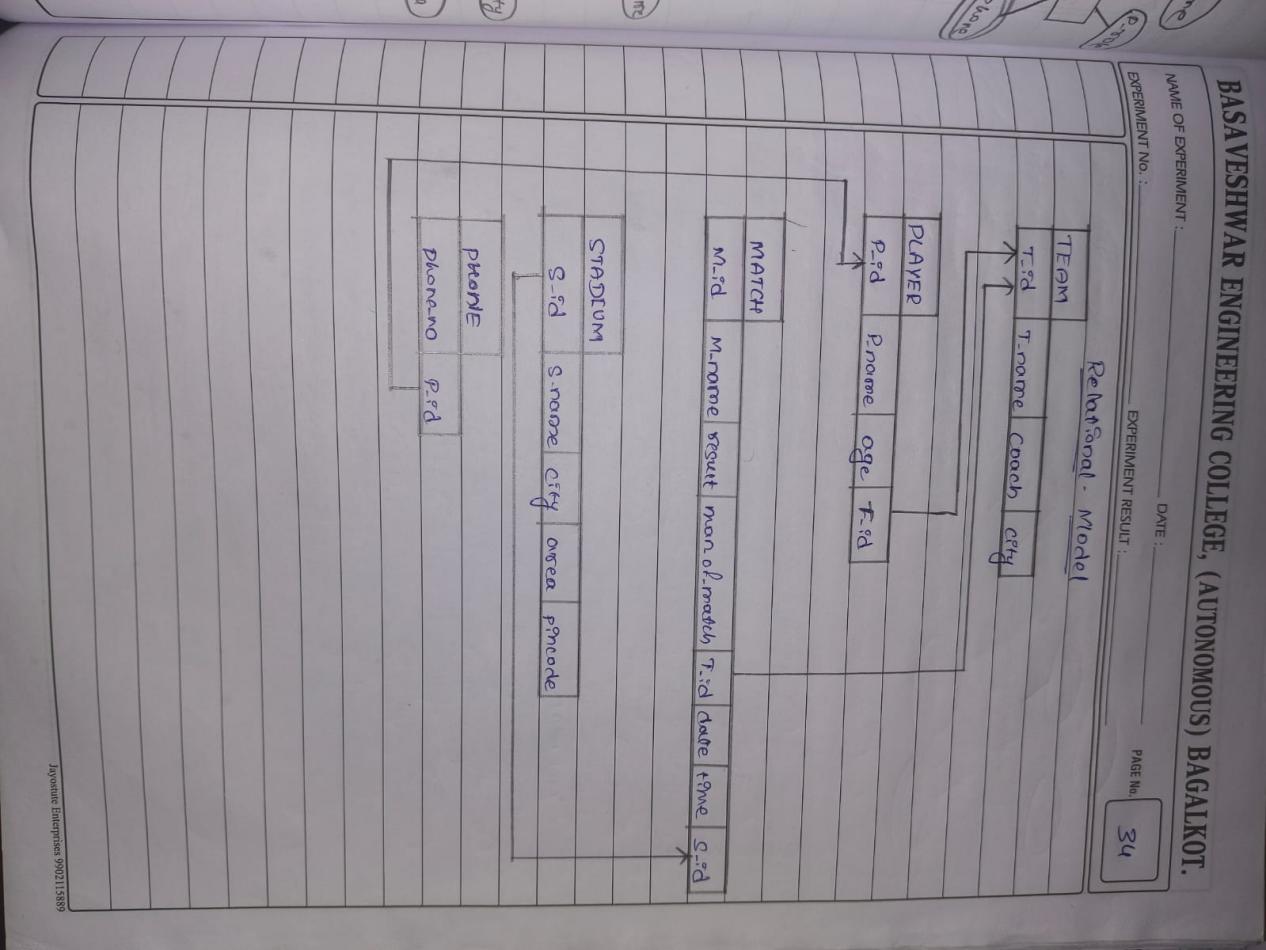
DELETE FROM STUDENT4

WHERE USN = 'S001';

select \* from student4 where USN ='S001';

-- 5TH one-------------------------------------------------------------------------------------------------------





create table TEAM(

Teamid int(10) primary key,

Team\_Name varchar(50) not null,

City varchar(50) not null,

Coach varchar(50) not null

);

create table PLAYER(

Player\_id int(10) primary key,

Player\_name varchar(50) not null,

Player\_age int(10) not null,

Player\_role varchar(50) not null,

Teamid int(10) not null,

foreign key (Teamid) references TEAM(Teamid)

);

create table STADIUM (

Std\_id int(4) primary key,

Std\_name varchar(100) not null,

city varchar(100) not null,

Std\_area varchar(100) not null,

pincode varchar(100) not null

);

create table MATCH1(

Match\_id int(10) primary key,

Match\_name varchar(50) not null,

Result varchar(50) not null,

Man\_of\_match varchar(50) not null,

Date date not null,

Time time not null,

Teamid int(10) not null,

Std\_id int(10) not null,

foreign key(Teamid) references TEAM (Teamid),

foreign key(Std\_id)references STADIUM(Std\_id)

);

create table PLAYER\_PH(

Phone\_NO int(10) not null,

Player\_id int(4),

foreign key(Player\_id) references PLAYER(Player\_id)

);

-- Inserting data into TEAM table

INSERT INTO TEAM (Teamid, Team\_Name, City, Coach) VALUES

(1, 'Warriors', 'Mumbai', 'John Smith'),

(2, 'Titans', 'Delhi', 'David Jones'),

(3, 'Eagles', 'Bangalore', 'Michael Clark'),

(4, 'Knights', 'Chennai', 'Steve Brown'),

(5, 'Dragons', 'Kolkata', 'James Adams');

-- Inserting data into PLAYER table

INSERT INTO PLAYER (Player\_id, Player\_name, Player\_age, Player\_role, Teamid) VALUES

(1, 'Rahul Sharma', 22, 'Batsman', 1),

(2, 'Amit Kumar', 23, 'Bowler', 1),

(3, 'Vijay Singh', 21, 'All-Rounder', 2),

(4, 'Sunil Patil', 25, 'Batsman', 2),

(5, 'Anil Mehta', 24, 'Bowler', 3),

(6, 'Rakesh Rao', 26, 'All-Rounder', 3),

(7, 'Suresh Nair', 20, 'Batsman', 4),

(8, 'Pravin Desai', 23, 'Wicketkeeper', 5);

-- Inserting data into STADIUM table

INSERT INTO STADIUM (Std\_id, Std\_name, city, Std\_area, pincode) VALUES

(1, 'Wankhede Stadium', 'Mumbai', 'Churchgate', '400020'),

(2, 'Feroz Shah Kotla', 'Delhi', 'Bahadur Shah Zafar Marg', '110002'),

(3, 'M. Chinnaswamy Stadium', 'Bangalore', 'MG Road', '560001'),

(4, 'MA Chidambaram Stadium', 'Chennai', 'Triplicane', '600005'),

(5, 'Eden Gardens', 'Kolkata', 'BBD Bagh', '700001');

-- Inserting data into MATCH1 table

INSERT INTO MATCH1 (Match\_id, Match\_name, Result, Man\_of\_match, Date, Time, Teamid, Std\_id) VALUES

(1, 'Warriors vs Titans', 'Warriors', 'Rahul Sharma', '2024-10-10', '14:30:00', 1, 1),

(2, 'Eagles vs Dragons', 'Eagles', 'Anil Mehta', '2024-10-12', '15:00:00', 3, 3),

(3, 'Knights vs Warriors', 'Warriors', 'Rahul Sharma', '2024-10-14', '16:00:00', 1, 4),

(4, 'Dragons vs Titans', 'Dragons', 'Pravin Desai', '2024-10-15', '17:30:00', 5, 5),

(5, 'Eagles vs Knights', 'Knights', 'Suresh Nair', '2024-10-18', '18:00:00', 4, 3);

-- Inserting data into PLAYER\_PH table

INSERT INTO PLAYER\_PH (Phone\_NO, Player\_id) VALUES

(98765432, 1),

(98765432, 2),

(98765432, 3),

(98765432, 4),

(98765432, 5),

(98765432, 6),

(98765432, 7),

(98765432, 8);

-- i. Display the youngest player (in terms of age) Name, Team name, age in which he belongs of the tournament

SELECT p.Player\_name, t.Team\_Name, p.Player\_age

FROM PLAYER p, TEAM t

WHERE p.Teamid = t.Teamid

ORDER BY p.Player\_age ASC

LIMIT 1;

-- ii. List the details of the stadium where the maximum number of matches were played.

SELECT s.Std\_name, s.city, s.Std\_area, s.pincode

FROM STADIUM s, MATCH1 m

WHERE s.Std\_id = m.Std\_id

GROUP BY s.Std\_id

ORDER BY COUNT(m.Match\_id) DESC

LIMIT 1;

-- iii. List the details of the player who is not a captain but got the man\_of\_the\_match award at least in two matches:

SELECT p.Player\_name, p.Player\_id, p.Player\_age

FROM PLAYER p, MATCH1 m

WHERE p.Player\_name = m.Man\_of\_match

GROUP BY p.Player\_name

HAVING COUNT(m.Match\_id) >= 2;

-- iv. Display the Team details who won the maximum matches.

SELECT t.Team\_Name, t.City, t.Coach

FROM TEAM t, MATCH1 m

WHERE t.Team\_Name = m.Result

GROUP BY t.Team\_Name

ORDER BY COUNT(m.Match\_id) DESC

LIMIT 1;

-- v. Display the team name who won all of its matches played in the same stadium.

SELECT t.Team\_Name

FROM TEAM t, MATCH1 m

WHERE t.Team\_Name = m.Result

GROUP BY t.Team\_Name, m.Std\_id

HAVING COUNT(m.Match\_id) = (

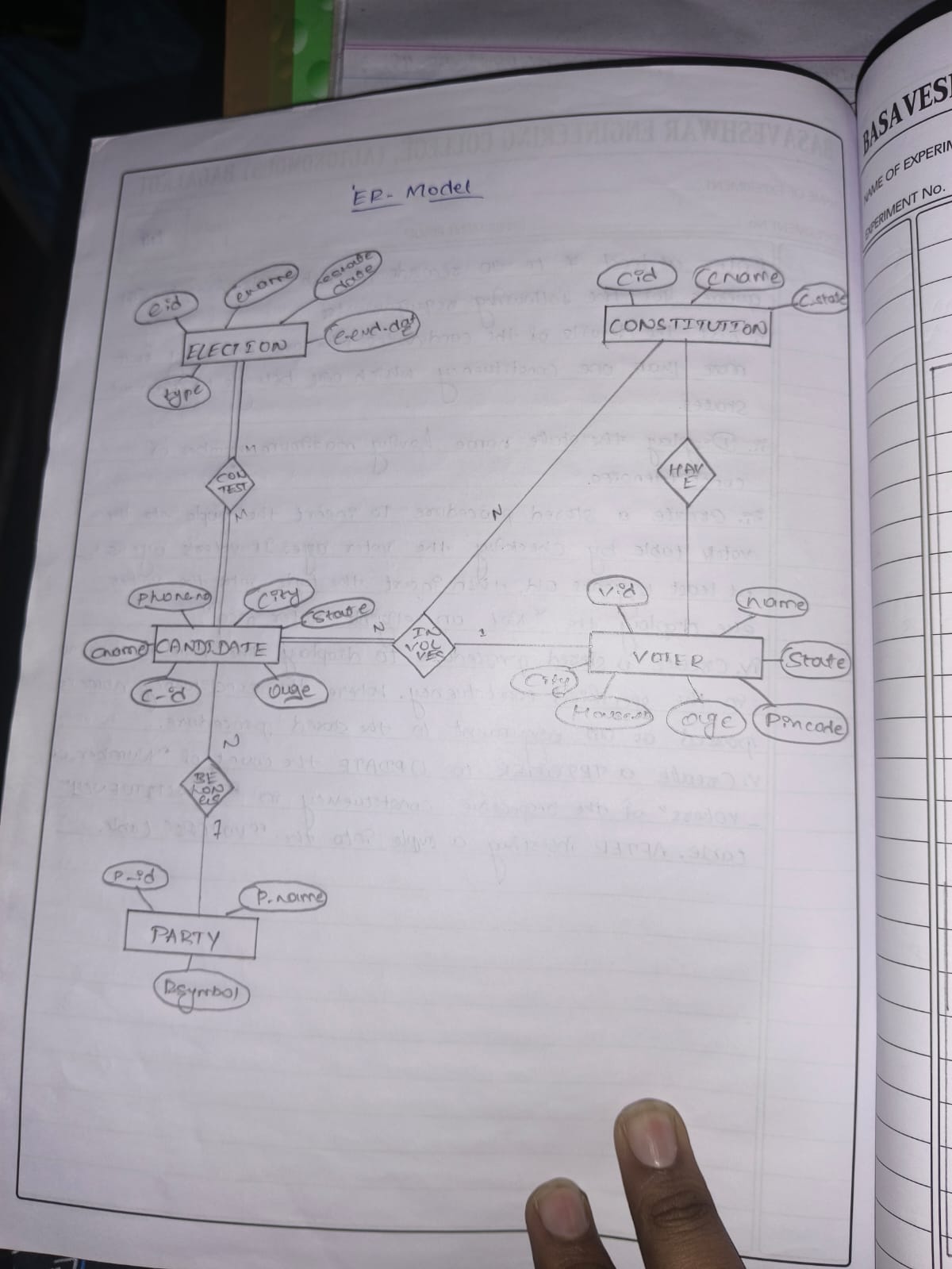
SELECT COUNT(m2.Match\_id)

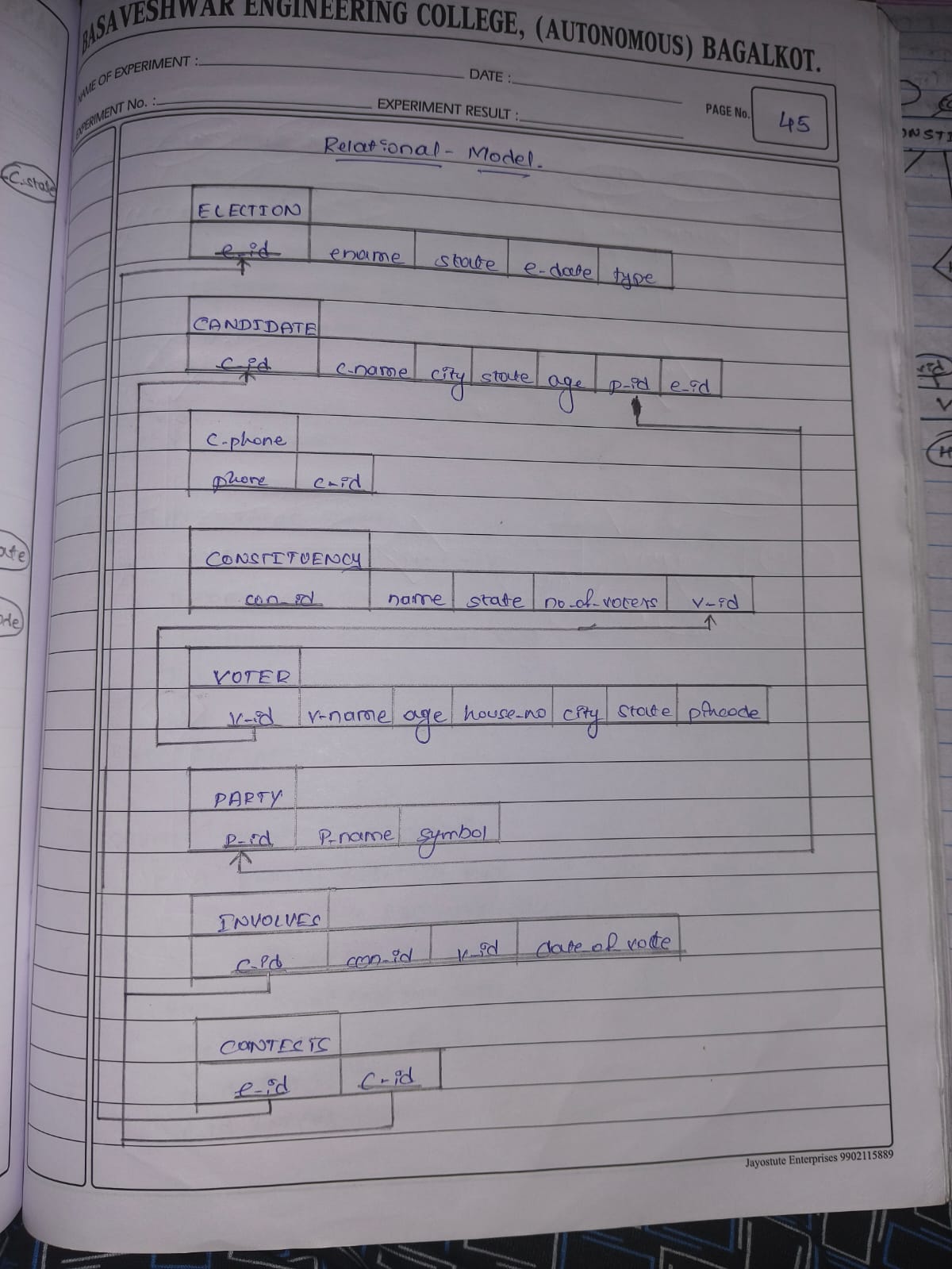
FROM MATCH1 m2

WHERE m2.Result = t.Team\_Name

);

-- 6th one-----------------------------------------------------------------------------------------------------





create table ELECTION(

Ele\_id int(5) primary key,

Ele\_name varchar(100) not null,

Ele\_Start\_date date not null,

Ele\_End\_date date not null,

Ele\_Type varchar(100) not null

);

-- Inserting data into ELECTION table

INSERT INTO ELECTION (Ele\_id, Ele\_name, Ele\_Start\_date, Ele\_End\_date, Ele\_Type) VALUES

(1, 'Karnataka Assembly Elections 2024', '2024-05-01', '2024-05-15', 'Assembly');

create table CONSTITUTION(

Con\_id int(5) primary key,

Con\_name varchar(100) not null,

Con\_state varchar(100) not null

);

-- Inserting data into CONSTITUTION table

INSERT INTO CONSTITUTION (Con\_id, Con\_name, Con\_state) VALUES

(1, 'Bangalore South', 'Karnataka'),

(2, 'Mysore', 'Karnataka'),

(3, 'Hubli', 'Karnataka'),

(4, 'Belgaum', 'Karnataka'),

(5, 'Mangalore', 'Karnataka');

create table PARTY(

Party\_id int(5) primary key,

Party\_name varchar(100) not null,

Party\_symbol varchar(100) not null

);

-- Inserting data into PARTY table

INSERT INTO PARTY (Party\_id, Party\_name, Party\_symbol) VALUES

(1, 'BJP', 'Lotus'),

(2, 'Congress', 'Hand'),

(3, 'JD(S)', 'Farmer'),

(4, 'AAP', 'Broom');

create table CANDIDATE(

Cand\_id int(5) primary key,

Cand\_name varchar(100) not null,

Cand\_ph\_no int(100) not null,

Cand\_age int(3) not null,

Cand\_state varchar(100) not null,

Party\_id int(4),

foreign key(Party\_id) references PARTY(Party\_id)

);

-- Inserting data into CANDIDATE table

INSERT INTO CANDIDATE (Cand\_id, Cand\_name, Cand\_ph\_no, Cand\_age, Cand\_state, Party\_id) VALUES

(1, 'Ravi Kumar', 98765432, 45, 'Karnataka', 1),

(2, 'Suresh Reddy', 98765431, 50, 'Karnataka', 2),

(3, 'Vinod Gowda', 98765432, 38, 'Karnataka', 3),

(4, 'Anil Patil', 98765433, 42, 'Karnataka', 4);

create table VOTER(

Voter\_id int (5) primary key,

Voter\_name varchar(100) not null,

Voter\_age int (3) not null,

House\_no int(5) not null,

City varchar (100) not null,

State varchar(1000) not null,

Pincode int(10) not null,

Cand\_id int (5),

foreign key(Cand\_id) references CANDIDATE(Cand\_id)

);

-- Inserting data into VOTER table

INSERT INTO VOTER (Voter\_id, Voter\_name, Voter\_age, House\_no, City, State, Pincode, Cand\_id) VALUES

(1, 'Rajesh Naik', 35, 101, 'Bangalore', 'Karnataka', 560001, 1),

(2, 'Manjunath Gowda', 40, 102, 'Mysore', 'Karnataka', 570001, 2),

(3, 'Sita Rao', 30, 103, 'Hubli', 'Karnataka', 580001, 3),

(4, 'Priya Shetty', 28, 104, 'Mangalore', 'Karnataka', 575001, 4),

(5, 'Kiran Rao', 19, 105, 'Bangalore', 'Karnataka', 560001, 1);

create table CONTEST(

Con\_id int(5),

Cand\_id int(5),

date date,

primary key(Con\_id ,Cand\_id ,date),

foreign key(Con\_id) references CONSTITUTION(Con\_id),

foreign key(Cand\_id ) references CANDIDATE(Cand\_id)

);

-- Inserting data into CONTEST table

INSERT INTO CONTEST (Con\_id, Cand\_id, date) VALUES

(1, 1, '2024-05-10'),

(2, 2, '2024-05-12'),

(3, 3, '2024-05-13'),

(4, 4, '2024-05-14');

create table INVOLVES(

Con\_id int(5),

Cand\_id int(5),

primary key(Con\_id ,Cand\_id),

foreign key(Con\_id) references CONSTITUTION(Con\_id),

foreign key(Cand\_id ) references CANDIDATE(Cand\_id)

);

-- Inserting data into INVOLVES table

INSERT INTO INVOLVES (Con\_id, Cand\_id) VALUES

(1, 1),

(2, 2),

(3, 3),

(4, 4);

-- i. List the details of the candidates who are contesting from more than one constituency which are belongs to different states.

SELECT c.Cand\_name, c.Cand\_state, COUNT(DISTINCT con.Con\_state) AS StateCount

FROM CANDIDATE c, INVOLVES i, CONSTITUTION con

WHERE c.Cand\_id = i.Cand\_id AND i.Con\_id = con.Con\_id

GROUP BY c.Cand\_name

HAVING StateCount > 1;

-- ii. Display the state name having maximum number of constituencies.

SELECT Con\_state, COUNT(Con\_id) AS TotalConstituencies

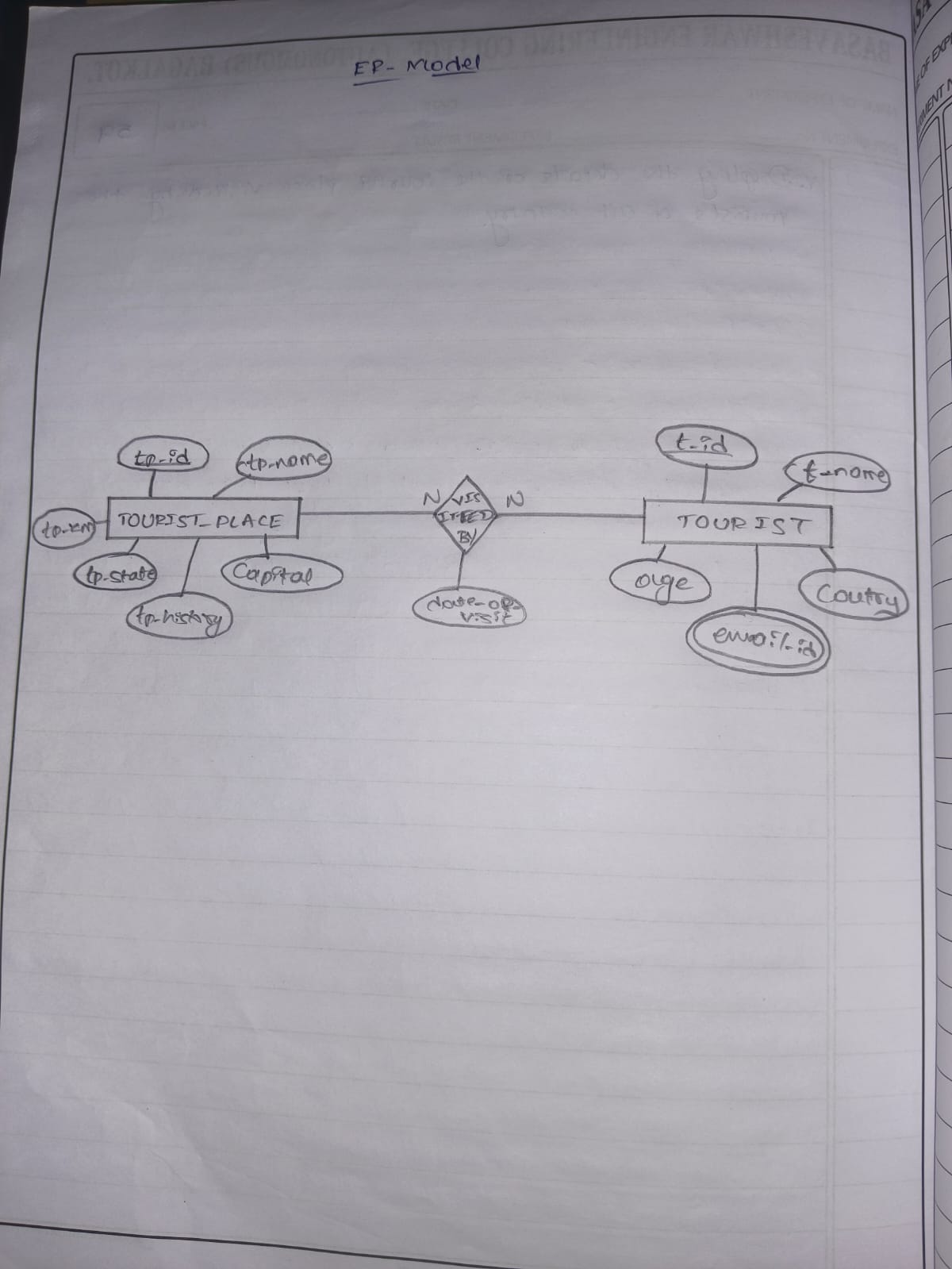
FROM CONSTITUTION

GROUP BY Con\_state

ORDER BY TotalConstituencies DESC

LIMIT 1;

-- 7th--------------------------------------------------------------------------------------------------------



create table TOURIST\_PLACE(

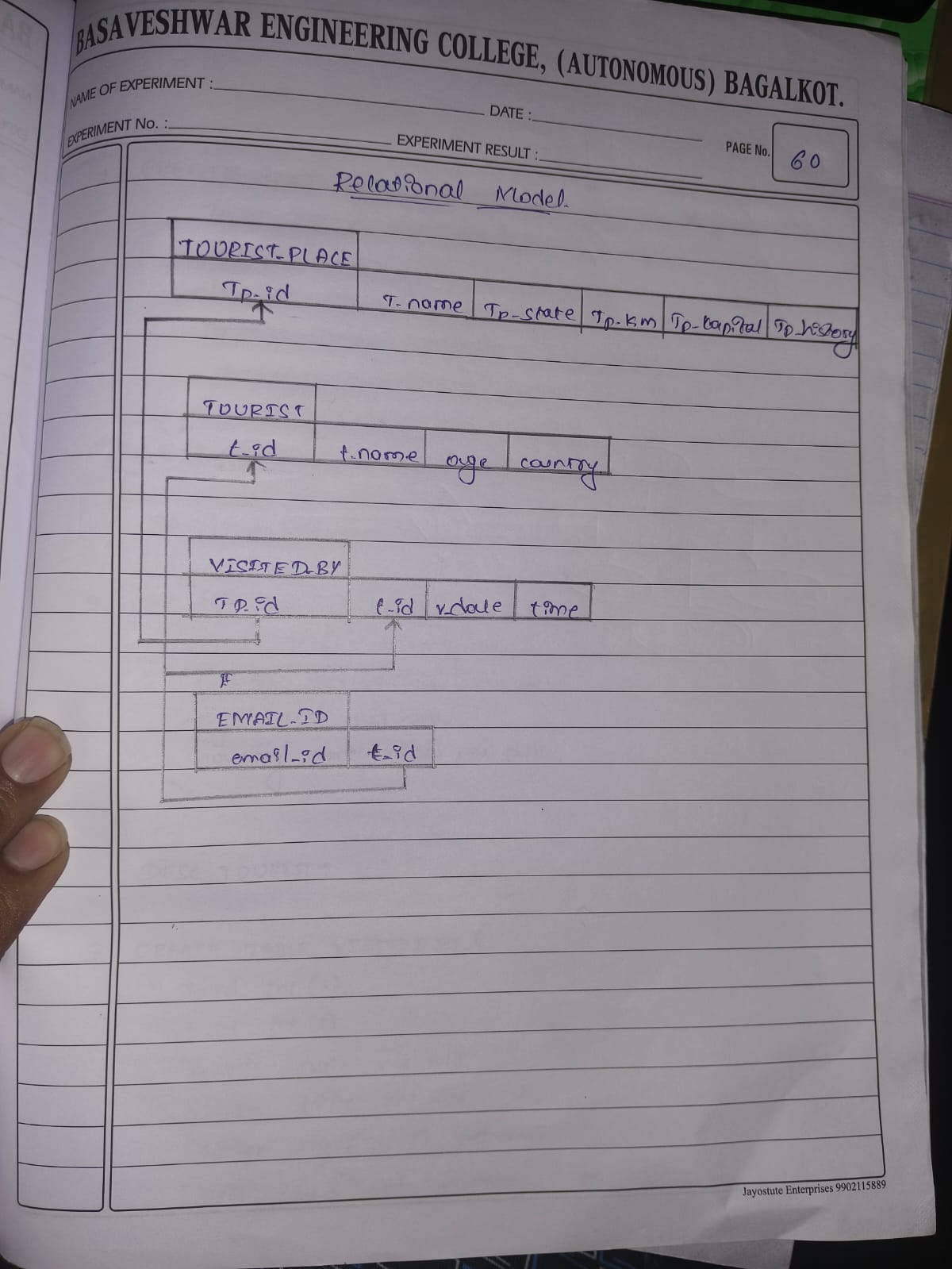
Tp\_id int(4) primary key,

Tp\_name varchar(100) not null,

Tp\_State varchar(100) not null,

Tp\_Captial varchar(100) not null,

Tp\_Km int(5) not null,

Tp\_History varchar(50) not null

);

-- Inserting data into TOURIST\_PLACE

INSERT INTO TOURIST\_PLACE (Tp\_id, Tp\_name, Tp\_State, Tp\_Captial, Tp\_Km, Tp\_History) VALUES

(1, 'Mysore Palace', 'Karnataka', 'Bangalore', 150, 'Royal Palace'),

(2, 'Hampi', 'Karnataka', 'Bangalore', 350, 'Ancient Ruins'),

(3, 'Taj Mahal', 'Uttar Pradesh', 'Lucknow', 200, 'Historical Monument'),

(4, 'Red Fort', 'Delhi', 'New Delhi', 50, 'Historic Fort'),

(5, 'Golden Temple', 'Punjab', 'Chandigarh', 300, 'Sikh Shrine');

-- Adding more tourist places in different states

INSERT INTO TOURIST\_PLACE (Tp\_id, Tp\_name, Tp\_State, Tp\_Captial, Tp\_Km, Tp\_History) VALUES

(6, 'Gateway of India', 'Maharashtra', 'Mumbai', 0, 'Iconic Landmark'),

(7, 'Charminar', 'Telangana', 'Hyderabad', 0, 'Historical Monument'),

(8, 'Jaisalmer Fort', 'Rajasthan', 'Jaisalmer', 0, 'Fortification'),

(9, 'Victoria Memorial', 'West Bengal', 'Kolkata', 0, 'Cultural Heritage');

create table TOURIST(

T\_id int(4) primary key,

T\_name varchar(100) not null,

T\_age int(3) not null,

T\_country varchar(100) not null

);

-- Inserting data into TOURIST

INSERT INTO TOURIST (T\_id, T\_name, T\_age, T\_country) VALUES

(1, 'John Doe', 35, 'USA'),

(2, 'Amit Kumar', 40, 'India'),

(3, 'Sarah Lee', 30, 'UK'),

(4, 'Li Wei', 25, 'China'),

(5, 'Carlos Lopez', 45, 'Spain');

create table VISITEDBY(

Tp\_id int(4),

T\_id int(4),

Visit\_date date,

primary key(Tp\_id,T\_id,Visit\_date),

foreign key(Tp\_id) references TOURIST\_PLACE(Tp\_id),

foreign key(T\_id) references TOURIST(T\_id)

);

-- Inserting data into VISITEDBY

INSERT INTO VISITEDBY (Tp\_id, T\_id, Visit\_date) VALUES

(1, 1, '2024-01-10'),

(2, 1, '2024-01-12'),

(3, 2, '2024-01-15'),

(4, 3, '2024-01-18'),

(5, 4, '2024-01-20'),

(1, 2, '2024-02-10'),

(2, 3, '2024-02-15');

-- Adding more tourist visits to those places

INSERT INTO VISITEDBY (Tp\_id, T\_id, Visit\_date) VALUES

(6, 1, '2024-01-11'), -- John Doe visited Gateway of India

(7, 2, '2024-01-16'), -- Amit Kumar visited Charminar

(8, 3, '2024-01-19'), -- Sarah Lee visited Jaisalmer Fort

(9, 4, '2024-01-21'); -- Li Wei visited Victoria Memorial

create table TOURIST\_EMAIL(

T\_id int(4),

Email\_id varchar(100) primary key ,

foreign key(T\_id) references TOURIST(T\_id)

);

-- Inserting data into TOURIST\_EMAIL

INSERT INTO TOURIST\_EMAIL (T\_id, Email\_id) VALUES

(1, 'john@example.com'),

(2, 'amit@example.com'),

(3, 'sarah@example.com'),

(4, 'liwei@example.com'),

(5, 'carlos@example.com');

-- i. List the state name which is having maximum number of tourist places.

SELECT Tp\_State, COUNT(Tp\_id) AS TotalPlaces

FROM TOURIST\_PLACE

GROUP BY Tp\_State

ORDER BY TotalPlaces DESC

LIMIT 1;

-- ii. List details of Tourist place where maximum number of tourists visited.

SELECT Tp.Tp\_name, Tp.Tp\_State, Tp.Tp\_Captial, Tp.Tp\_Km, Tp.Tp\_History

FROM TOURIST\_PLACE Tp, VISITEDBY V

WHERE Tp.Tp\_id = V.Tp\_id

GROUP BY Tp.Tp\_id

ORDER BY COUNT(V.T\_id) DESC

LIMIT 1;

-- iii. List the details of tourists visited all tourist places of the state “KARNATAKA”.

SELECT T.T\_name, T.T\_age, T.T\_country

FROM TOURIST T

WHERE NOT EXISTS (

SELECT Tp.Tp\_id

FROM TOURIST\_PLACE Tp

WHERE Tp.Tp\_State = 'Karnataka'

AND Tp.Tp\_id NOT IN (

SELECT V.Tp\_id

FROM VISITEDBY V

WHERE V.T\_id = T.T\_id

)

);

-- iv. Display the details of the tourists visited at least one tourist place of the state, but visited all states tourist places.

-- this query is not working

SELECT DISTINCT T.T\_id, T.T\_name, T.T\_age, T.T\_country

FROM TOURIST T

WHERE EXISTS (

SELECT 1

FROM VISITEDBY V

JOIN TOURIST\_PLACE Tp ON V.Tp\_id = Tp.Tp\_id

WHERE V.T\_id = T.T\_id

)

AND NOT EXISTS (

SELECT DISTINCT Tp.Tp\_State

FROM TOURIST\_PLACE Tp

WHERE NOT EXISTS (

SELECT 1

FROM VISITEDBY V2

WHERE V2.T\_id = T.T\_id AND V2.Tp\_id IN (

SELECT Tp2.Tp\_id

FROM TOURIST\_PLACE Tp2

WHERE Tp2.Tp\_State = Tp.Tp\_State

)

)

);

-- atleast one tourist place in karnataka

SELECT T.T\_name, T.T\_age, T.T\_country

FROM TOURIST T

WHERE EXISTS (

SELECT 1

FROM VISITEDBY V

WHERE V.T\_id = T.T\_id AND V.Tp\_id IN (

SELECT Tp\_id FROM TOURIST\_PLACE WHERE Tp\_State = 'Karnataka'

)

);

-- v. Display the details of the tourist place visited by the tourists of all country

-- this is also not working

SELECT Tp.Tp\_id, Tp.Tp\_name, Tp.Tp\_State, Tp.Tp\_Captial, Tp.Tp\_Km, Tp.Tp\_History

FROM TOURIST\_PLACE Tp

WHERE NOT EXISTS (

SELECT T.T\_country

FROM TOURIST T

WHERE T.T\_country NOT IN (

SELECT T2.T\_country

FROM VISITEDBY V

JOIN TOURIST T2 ON V.T\_id = T2.T\_id

WHERE V.Tp\_id = Tp.Tp\_id

)

);

-- 8th one -----------------------------------------------------------------------------------------------------

-- Create the STUDENT table

CREATE TABLE STUDENT8 (

regno VARCHAR(20) PRIMARY KEY,

name VARCHAR(100) NOT NULL,

major VARCHAR(50),

bdate DATE

);

-- Create the COURSE table

CREATE TABLE COURSE (

course\_no INT PRIMARY KEY,

cname VARCHAR(100) NOT NULL,

dept VARCHAR(50) NOT NULL

);

-- Create the TEXT (book) table

CREATE TABLE TEXT1 (

book\_ISBN INT PRIMARY KEY,

book\_title VARCHAR(200) NOT NULL,

publisher VARCHAR(100) NOT NULL,

author VARCHAR(100) NOT NULL

);

-- Create the ENROLL table

CREATE TABLE ENROLL (

regno VARCHAR(20),

course\_no INT,

sem INT,

marks INT,

PRIMARY KEY (regno, course\_no, sem),

FOREIGN KEY (regno) REFERENCES STUDENT8(regno),

FOREIGN KEY (course\_no) REFERENCES COURSE(course\_no)

);

-- Create the BOOK\_ADOPTION table

CREATE TABLE BOOK\_ADOPTION (

course\_no INT,

sem INT,

book\_ISBN INT,

PRIMARY KEY (course\_no, sem, book\_ISBN),

FOREIGN KEY (course\_no) REFERENCES COURSE(course\_no),

FOREIGN KEY (book\_ISBN) REFERENCES TEXT1(book\_ISBN)

);

-- Insert data into STUDENT

INSERT INTO STUDENT8 (regno, name, major, bdate) VALUES

('S001', 'Alice Johnson', 'Computer Science', '2000-05-15'),

('S002', 'Bob Smith', 'Mechanical Engineering', '1999-10-20'),

('S003', 'Catherine Lee', 'Electrical Engineering', '2001-01-10'),

('S004', 'David Williams', 'Computer Science', '1998-03-25'),

('S005', 'Emily Davis', 'Mathematics', '1997-07-14'),

('S006', 'Frank Thompson', 'Computer Science', '2000-12-30'),

('S007', 'Grace Miller', 'Physics', '1999-11-11');

select \* from STUDENT8;

-- Insert data into COURSE

INSERT INTO COURSE (course\_no, cname, dept) VALUES

(101, 'Data Structures', 'Computer Science'),

(102, 'Thermodynamics', 'Mechanical Engineering'),

(103, 'Circuits', 'Electrical Engineering'),

(104, 'Discrete Mathematics', 'Mathematics'),

(105, 'Quantum Mechanics', 'Physics'),

(106, 'Algorithms', 'Computer Science'),

(107, 'Heat Transfer', 'Mechanical Engineering');

select \* from COURSE;

-- Insert data into TEXT1

INSERT INTO TEXT1 (book\_ISBN, book\_title, publisher, author) VALUES

(978013110, 'The C Programming Language', 'Prentice Hall', 'Brian Kernighan'),

(978026203, 'Introduction to Algorithms', 'MIT Press', 'Thomas H. Cormen'),

(978013790, 'Operating System Concepts', 'Wiley', 'Abraham Silberschatz'),

(978020161, 'Computer Networks', 'Pearson', 'Andrew S. Tanenbaum'),

(978032157, 'Artificial Intelligence: A Modern Approach', 'Pearson', 'Stuart Russell'),

(978013235, 'Clean Code', 'Prentice Hall', 'Robert C. Martin'),

(978052177, 'Quantum Computing', 'Cambridge University Press', 'Michael A. Nielsen');

-- Insert data into ENROLL

INSERT INTO ENROLL (regno, course\_no, sem, marks) VALUES

('S001', 101, 1, 85),

('S002', 102, 1, 90),

('S003', 103, 1, 75),

('S004', 104, 1, 88),

('S005', 105, 1, 92),

('S006', 106, 2, 95),

('S007', 107, 2, 78),

('S001', 106, 2, 90);

INSERT INTO ENROLL (regno, course\_no, sem, marks) VALUES

('S002', 101, 1, 80), -- S002 enrolls in Course 101

('S003', 101, 1, 85), -- S003 enrolls in Course 101

('S004', 101, 1, 88); -- S004 enrolls in Course 101

-- Insert data into BOOK\_ADOPTION

INSERT INTO BOOK\_ADOPTION (course\_no, sem, book\_ISBN) VALUES

(101, 1, 978013110),

(106, 2, 978026203),

(103, 1, 978013790),

(105, 1, 978052177),

(101, 1, 978032157);

-- i. List out the student details, and their course details. The records should be ordered in a semester wise manner.

select s.regno,s.name,c.course\_no,c.cname,en.sem

from student8 s, course c, enroll en

where s.regno=en.regno and en.course\_no=c.course\_no

order by en.sem ;

-- ii. List out the student details under a particular department whose name is ordered in a semester wise

SELECT S.regno, S.name, S.major, E.sem

FROM STUDENT8 S, ENROLL E, COURSE C

WHERE S.regno = E.regno AND C.dept = 'Computer Science'

ORDER BY E.sem;

-- iii. List out all the book details under a particular course

select t.\*

from TEXT1 t,BOOK\_ADOPTION b

where t.book\_ISBN=b.book\_ISBN and b.course\_no=101;

-- iv. Find out the Courses in which number of students studying will be more than 2.

SELECT C.course\_no, C.cname, COUNT(E.regno) AS num\_students

FROM COURSE C,ENROLL E

where C.course\_no = E.course\_no

GROUP BY C.course\_no, C.cname

HAVING COUNT(E.regno) > 2;

-- v. Find out the Publisher who has published more than 2 books.

SELECT publisher, COUNT(\*) AS book\_count

FROM TEXT1

GROUP BY publisher

HAVING COUNT(\*) > 2;

-- vi. Find out the authors who have written book for I semester, computer science course.

SELECT DISTINCT T.author

FROM TEXT1 T, BOOK\_ADOPTION B, COURSE C

WHERE T.book\_ISBN = B.book\_ISBN AND B.sem = 1 AND C.dept = 'Computer Science';

-- vii. List out the student details whose total number of months starting from their date of birth is more than -

SELECT regno, name, major

FROM STUDENT8

WHERE DATEDIFF(CURDATE(), bdate) / 365 > 18; -- Check if age is greater than 18 years

-- viii. Find out the course name to which maximum number of students have joined

SELECT C.cname, COUNT(\*) AS student\_count

FROM ENROLL E, COURSE C

WHERE E.course\_no = C.course\_no

GROUP BY C.cname

ORDER BY student\_count DESC

LIMIT 1;