

1. Create the following tables with properly specifying Primary keys, Foreign keys and solve the following queries.

BRANCH (Branchid, Branchname, HOD)

STUDENT (USN, Name, Address, Branchid, sem)

BOOK (Bookid, Bookname, Authorid, Publisher, Branchid)

AUTHOR (Authorid, Authorname, Country, age)

BORROW (USN, Bookid, Borrowed_Date)

Execute the following Queries:

- i. List the details of Students who are all studying in 2nd sem MCA.
- ii. List the students who are not borrowed any books.
- iii. Display the USN, Student name, Branch_name, Book_name, Author_name, Books_Borrowed_Date of 2nd sem MCA Students who borrowed books.
- iv. Display the number of books written by each Author.
- v. Display the student details who borrowed more than two books.
- vi. Display the student details who borrowed books of more than one Author.
- vii. Display the Book names in descending order of their names.
- viii. List the details of students who borrowed the books which are all published by the same publisher.

```
CREATE TABLE BRANCH(  
    BRANCHID INT PRIMARY KEY,  
    BRANCHNAME VARCHAR(30),  
    HOD VARCHAR(30)  
);
```

```
INSERT INTO BRANCH VALUES (1,'MCA','AMAR');  
INSERT INTO BRANCH VALUES (2,'MBA','AKBAR');  
INSERT INTO BRANCH VALUES (3,'MTECH','ANTONY');
```

```
CREATE TABLE STUDENT(  
    USN VARCHAR(30) PRIMARY KEY,  
    NAME VARCHAR(30),  
    ADDRESS VARCHAR(30),  
    BRANCHID INT REFERENCES BRANCH(BRANCHID),  
    SEM INT  
);
```

```
INSERT INTO STUDENT VALUES ('1001','LOKESH','BANGALORE',1,2);  
INSERT INTO STUDENT VALUES ('1002','AKSHAY','BANGALORE',1,2);  
INSERT INTO STUDENT VALUES ('1003','MANOJ','BANGALORE',2,1);  
INSERT INTO STUDENT VALUES ('1004','AMITH','SHIMOGA',3,1);  
INSERT INTO STUDENT VALUES ('1005','ABHI','BANGALORE',1,2);
```

```
CREATE TABLE AUTHOR(  
    AUTHORID INT PRIMARY KEY,  
    AUTHORNAMENAME VARCHAR(30),  
    COUNTRY VARCHAR(30),  
    AGE INT  
);
```

```

INSERT INTO AUTHOR VALUES (1,'DENNIS','US',56);
INSERT INTO AUTHOR VALUES (2,'RAMAKRISHNA','INDIA',56);
INSERT INTO AUTHOR VALUES (3,'JAMES','US',56);
INSERT INTO AUTHOR VALUES (4,'RITCHE','UK',56);

```

```

CREATE TABLE BOOK(
    BOOKID INT PRIMARY KEY,
    BOOKNAME VARCHAR(30),
    AUTHORID INT REFERENCES AUTHOR(AUTHORID),
    PUBLISHER VARCHAR(30),
    BRANCHID INT REFERENCES BRANCH(BRANCHID)
);

```

```

INSERT INTO BOOK VALUES(1,'JAVA',1,'PEARSON',1);
INSERT INTO BOOK VALUES(2,'C',1,'MC HILL',1);
INSERT INTO BOOK VALUES(3,'C++',2,'PEARSON',1);
INSERT INTO BOOK VALUES(4,'DBMS',3,'PEARSON',2);
INSERT INTO BOOK VALUES(5,'OS',4,'PEARSON',3);

```

```

CREATE TABLE BORROW(
    USN VARCHAR(20) REFERENCES STUDENT(USN),
    BOOKID INT REFERENCES BOOK(BOOKID),
    BORROW_DATE DATE
);

```

```

INSERT INTO BORROW VALUES ('1001',1,'12-JAN-2021');
INSERT INTO BORROW VALUES ('1001',2,'12-JAN-2021');
INSERT INTO BORROW VALUES ('1001',3,'12-JAN-2021');
INSERT INTO BORROW VALUES ('1002',1,'12-JAN-2021');
INSERT INTO BORROW VALUES ('1003',4,'12-JAN-2021');
INSERT INTO BORROW VALUES ('1003',5,'12-JAN-2021');

```

NOTE: IF ABOVE DATE FORMAT NOT WORKS TRY THIS ONE

INSERT INTO BORROW VALUES ('1003',5,'2021-01-12');

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

```

SELECT * FROM STUDENT
WHERE SEM = 2 AND BRANCHID IN (
    SELECT BRANCHID
    FROM BRANCH
    WHERE BRANCHNAME = 'MCA'
);

```

USN	NAME	ADDRESS	BRANCHID	SEM
1001	LOKESH	BANGALORE	1	2
1002	AKSHAY	BANGALORE	1	2
1005	ABHI	BANGALORE	1	2

ii. List the students who are not borrowed any books

```
SELECT * FROM STUDENT
WHERE USN NOT IN(
    SELECT USN
    FROM BORROW
);
```

USN	NAME	ADDRESS	BRANCHID	SEM
1004	AMITH	SHIMOGA	3	1
1005	ABHI	BANGALORE	1	2

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,BRANCHNAME,BOOKNAME,AUTHORNAME,BORROW_DATE
FROM STUDENT S,BOOK,BORROW B,AUTHOR,BRANCH
WHERE S.USN = B.USN AND S.BRANCHID = BRANCH.BRANCHID AND BOOK.AUTHORID =
AUTHOR.AUTHORID AND BOOK.BOOKID = B.BOOKID AND SEM = 2 AND BRANCH = 'MCA';
```

USN	NAME	BRANCHNAME	BOOKNAME	AUTHORNAME	BORROW_DATE
1001	LOKESH	MCA	JAVA	DENNIS	12-JAN-21
1001	LOKESH	MCA	C	DENNIS	12-JAN-21
1001	LOKESH	MCA	C++	RAMAKRISHNA	12-JAN-21
1002	AKSHAY	MCA	JAVA	DENNIS	12-JAN-21

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

```
SELECT AUTHORNAME,COUNT(*)
FROM BOOK B,AUTHOR A
WHERE B.AUTHORID = A.AUTHORID
GROUP BY AUTHORNAME;
```

AUTHORNAME	COUNT(*)
DENNIS	2
RAMAKRISHNA	1
JAMES	1
RITCHE	1

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

```
SELECT *
FROM STUDENT
WHERE USN IN(SELECT USN FROM BORROW GROUP BY USN HAVING COUNT(USN) >2 );
```

USN	NAME	ADDRESS	BRANCHID	SEM
1001	LOKESH	BANGALORE	1	2

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

```
SELECT *
FROM STUDENT WHERE USN IN(
    SELECT USN
    FROM BORROW, BOOK
    WHERE BORROW.BOOKID = BOOK.BOOKID
    GROUP BY USN
    HAVING COUNT(DISTINCT AUTHORID) >1
);
```

USN	NAME	ADDRESS	BRANCHID	SEM
1001	LOKESH	BANGALORE	1	2
1003	MANOJ	BANGALORE	2	1

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

```
SELECT BOOKNAME
FROM BOOK
ORDER BY BOOKNAME DESC;
```

BOOKNAME
OS
JAVA
DBMS
C++
C

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

```
SELECT *
FROM STUDENT WHERE USN IN(
    SELECT USN FROM BORROW B, BOOK BK
    WHERE B.BOOKID = BK.BOOKID
    GROUP BY USN
    HAVING COUNT(DISTINCT PUBLISHER) =1
);
```

USN	NAME	ADDRESS	BRANCHID	SEM
1002	AKSHAY	BANGALORE	1	2
1003	MANOJ	BANGALORE	2	1

2. Consider the following schema: STUDENT (USN, name, date_of_birth, branch, mark1, mark2, mark3, total,GPA) Execute the following queries:

- i. Update the column total by adding the columns mark1, mark2, mark3.
- ii. Find the GPA score of all the students.
- iii. Find the students who born on a particular year of birth from the date_of_birth column.
- iv. List the students who are studying in a particular branch of study.
- v. Find the maximum GPA score of the student branch-wise.
- vi. Find the students whose name starts with the alphabet "S".
- vii. Find the students whose name ends with the alphabets "AR".
- viii. Delete the student details whose USN is given as 1001

```
CREATE TABLE STUDENT(  
  USN VARCHAR(30) PRIMARY KEY,  
  NAME VARCHAR(30),  
  DATE_OF_BIRTH DATE,  
  BRANCH VARCHAR(30),  
  MARKS1 INT,  
  MARKS2 INT,  
  MARKS3 INT,  
  TOTAL INT,  
  GPA FLOAT  
);
```

```
INSERT INTO STUDENT VALUES('1001','AMAR','12-JUN-2001','MCA',89,82,92,0,0);  
INSERT INTO STUDENT VALUES('1002','AKBAR','12-JUN-1999','MCA',70,65,91,0,0);  
INSERT INTO STUDENT VALUES('1003','SANJAY','12-JUN-2001','MBA',69,90,90,0,0);  
INSERT INTO STUDENT VALUES('1004','SAGAR','12-JUN-2000','MBA',78,80,91,0,0);  
INSERT INTO STUDENT VALUES('1005','ADHI','12-JUN-1999','MBA',59,70,92,0,0);  
INSERT INTO STUDENT VALUES('1007','LOKESH','12-JUN-01','MCA',52,52,93,0,0);
```

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

```
UPDATE STUDENT  
SET TOTAL = MARKS1 + MARKS2 + MARKS3;
```

6 rows updated.

ii. Find the GPA score of all the students.

```
UPDATE STUDENT  
SET GPA = (TOTAL / 30);
```

6 rows updated.

iii. Find the students who born on a particular year of birth from the date_of_birth column.

```
SELECT *
FROM STUDENT
WHERE DATE_OF_BIRTH LIKE '%99';
```

USN	NAME	DATE_OF_B	BRANCH	MARKS1	MARKS2	MARKS3	TOTAL	GPA
1002	AKBAR	12-JUN-99	MCA	70	65	91	226	7.53333333
1005	ADHI	12-JUN-99	MBA	59	70	92	221	7.36666667

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

```
SELECT *
FROM STUDENT
WHERE BRANCH = 'MCA';
```

USN	NAME	DATE_OF_B	BRANCH	MARKS1	MARKS2	MARKS3	TOTAL	GPA
1001	AMAR	12-JUN-01	MCA	89	82	92	263	8.76666667
1002	AKBAR	12-JUN-99	MCA	70	65	91	226	7.53333333
1007	LOKESH	12-JUN-01	MCA	52	52	93	197	6.56666667

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

```
SELECT BRANCH,MAX(GPA)
FROM STUDENT
GROUP BY BRANCH;
```

BRANCH	MAX(GPA)
MBA	8.3
MCA	8.76666667

vi. Find the students whose name starts with the alphabet "S".

```
SELECT NAME
FROM STUDENT
WHERE NAME LIKE 'S%';
```

NAME
SANJAY
SAGAR

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

```
SELECT NAME  
FROM STUDENT  
WHERE NAME LIKE '%AR';
```

```
NAME  
-----  
AMAR  
AKBAR  
SAGAR
```

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

```
DELETE FROM STUDENT WHERE USN = '1001';
```

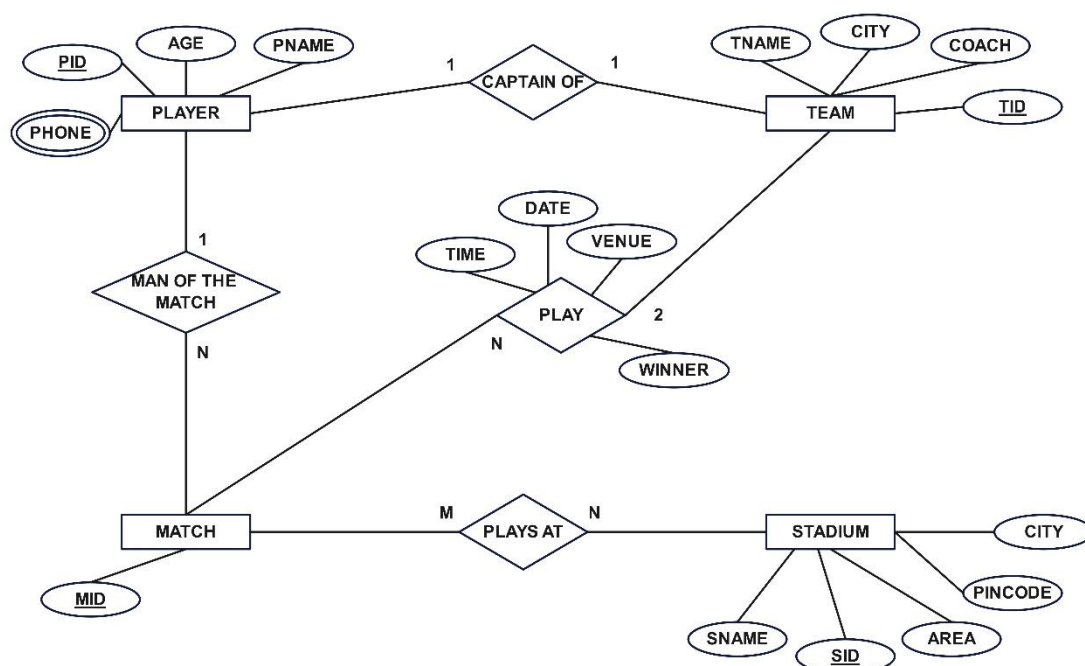
1 row deleted.

3. Design an ER-diagram for the following scenario, Convert the same into a relational model and then solve the following queries. Consider a Cricket Tournament “ABC CUP” organized by an organization. In the tournament there are many teams are contesting each having a Teamid, Team_Name, City, a coach. Each team is uniquely identified by using Teamid. A team can have many Players and a captain. Each player is uniquely identified by Playerid, having a Name, and multiple phone numbers, age. A player represents only one team. There are many Stadiums to conduct matches. Each stadium is identified using Stadiumid, having a stadium_name, Address (involves city, area_name, pincode). A team can play many matches. Each match played between the two teams in the scheduled date and time in the predefined Stadium. Each match is identified uniquely by using Matchid. Each match won by any of the one team that also wants to record in the database. For each match man_of_the match award given to a player.

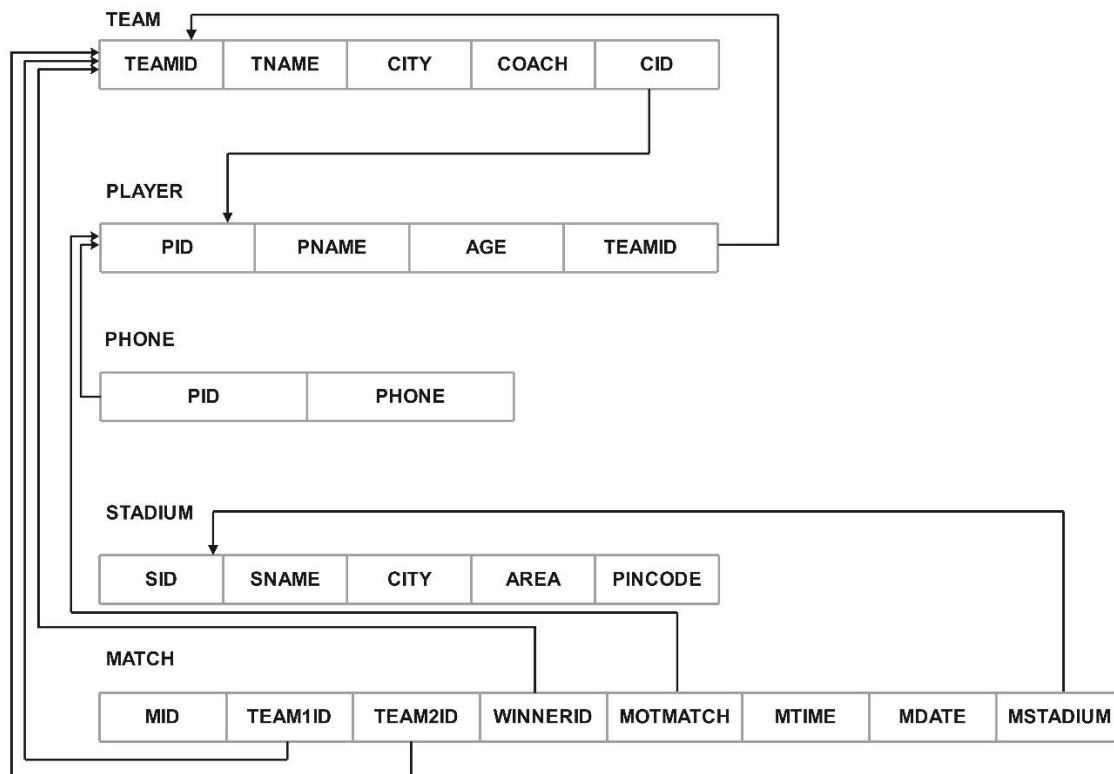
Execute the following Queries:

- 1 Display the youngest player (in terms of age) Name, Team name , age in which he belongs of the tournament.
- 2 List the details of the stadium where the maximum number of matches were played.
- 3 List the details of the player who is not a captain but got the man_of _match award at least in two matches.
- 4 Display the Team details who won the maximum matches.
- 5 Display the team name where all its won matches played in the same stadium.

ER DIGRAM



SCHEMA



```

CREATE TABLE TEAM(
    TEAMID INT PRIMARY KEY,
    TNAME VARCHAR(30),
    CITY VARCHAR(30),
    COACH VARCHAR(30),
    CID INT
);
  
```

```

INSERT INTO TEAM VALUES (1,'RCB','BANGALORE','RAHUL',1);
INSERT INTO TEAM VALUES (2,'CSK','CHENNAI','AKBAR',2);
INSERT INTO TEAM VALUES (3,'MI','MUMBAI','ANTONY',3);
INSERT INTO TEAM VALUES (4,'GT','GUJARAT','RAHUL DRAVID',4);
  
```

```

CREATE TABLE PLAYER(
    PID INT PRIMARY KEY,
    PNAME VARCHAR(30),
    AGE INT,
    TEAMID INT REFERENCES TEAM(TEAMID)
);
  
```

```

INSERT INTO PLAYER VALUES (1,'VIRAT',35,1);
  
```

```
INSERT INTO PLAYER VALUES (2,'DHONI',40,2);
INSERT INTO PLAYER VALUES (3,'ROHITH',39,3);
INSERT INTO PLAYER VALUES (4,'HARDIK',30,4);
INSERT INTO PLAYER VALUES (5,'K L RAHUL',32,1);
```

```
ALTER TABLE TEAM
ADD FOREIGN KEY (CID) REFERENCES PLAYER(PID);
```

```
CREATE TABLE PHONE(
  PID INT REFERENCES PLAYER(PID),
  PHONE VARCHAR(20)
);
```

```
INSERT INTO PHONE VALUES (1,'9999999999');
INSERT INTO PHONE VALUES (2,'9999999998');
INSERT INTO PHONE VALUES (3,'9999999997');
INSERT INTO PHONE VALUES (4,'9999999996');
INSERT INTO PHONE VALUES (5,'9999999995');
INSERT INTO PHONE VALUES (1,'9999999994');
```

```
CREATE TABLE STADIUM(
  SID INT PRIMARY KEY,
  SNAME VARCHAR(30),
  CITY VARCHAR(30),
  AREA VARCHAR(30),
  PINCODE INT
);
```

```
INSERT INTO STADIUM VALUES(1,'CHINNASWAMY','BANGALORE','MG ROAD','560092');
INSERT INTO STADIUM VALUES(2,'KANTEERAVA','DELHI','MAIN ROAD','560093');
INSERT INTO STADIUM VALUES(3,'MODI','GUJARAT','CITY','560094');
```

```
CREATE TABLE MATCH(
  MID INT PRIMARY KEY,
  TEAM1ID INT REFERENCES TEAM(TEAMID),
  TEAM2ID INT REFERENCES TEAM(TEAMID),
  WINNERID INT REFERENCES TEAM(TEAMID),
  MOTMATCH INT REFERENCES PLAYER(PID),
  MTIME VARCHAR(10),
  MDATE DATE,
  MSTADIUM INT REFERENCES STADIUM(SID)
);
```

```

INSERT INTO MATCH VALUES(1,1,2,1,1,'7PM','12-JAN-23',1);
INSERT INTO MATCH VALUES(2,1,3,1,1,'7PM','13-JAN-23',3);
INSERT INTO MATCH VALUES(3,1,4,1,5,'7PM','14-JAN-23',1);
INSERT INTO MATCH VALUES(4,4,2,2,2,'7PM','15-JAN-23',2);
INSERT INTO MATCH VALUES(5,1,2,1,5,'7PM','16-JAN-23',1);

```

NOTE: IF ABOVE DATE FORMAT NOT WORKS TRY THIS ONE

INSERT INTO MATCH VALUES(5,1,2,1,5,'7PM','2023-01-16',1);

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

```

SELECT PNAME, TNAME, AGE
FROM PLAYER P, TEAM T
WHERE P.TEAMID = T.TEAMID AND AGE = (SELECT MIN(AGE) FROM PLAYER);

```

PNAME	TNAME	AGE

HARDIK	GT	30

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

```

SELECT *
FROM STADIUM
WHERE SID IN (
    SELECT MSTADIUM
    FROM MATCH
    GROUP BY MSTADIUM
    HAVING COUNT(MSTADIUM) = (
        SELECT MAX(COUNTS) FROM (
            SELECT COUNT(*) COUNTS
            FROM MATCH
            GROUP BY MSTADIUM) COUNTER
        )
);

```

SID	SNAME	CITY	AREA	PINCODE

1	CHINNASWAMY	BANGALORE	MG ROAD	560092

3 List the details of the player who is not a captain but got the man_of _match award at least in two matches.

```
SELECT *
FROM PLAYER
WHERE PID IN (
    SELECT MOTMATCH
    FROM MATCH
    GROUP BY MOTMATCH
    HAVING COUNT(MOTMATCH)>=2
) AND PID NOT IN (SELECT CID FROM TEAM);
```

PID	PNAME	AGE	TEAMID
5	K L RAHUL	32	1

4 Display the Team details who won the maximum matches.

```
SELECT *
FROM TEAM
WHERE TEAMID IN (
    SELECT WINNERID
    FROM MATCH
    GROUP BY WINNERID
    HAVING COUNT(WINNERID) = (
        SELECT MAX(COUNTS) FROM (
            SELECT COUNT(*) COUNTS
            FROM MATCH
            GROUP BY WINNERID) COUNTER
        )
);
```

TEAMID	TNAME	CITY	COACH	CID
1	RCB	BANGALORE	RAHUL	1

5 Display the team name where all its won matches played in the same stadium.

```
SELECT TNAME
FROM TEAM, MATCH
WHERE TEAMID = WINNERID
GROUP BY TNAME
HAVING COUNT(DISTINCT MSTADIUM) = 1;
```

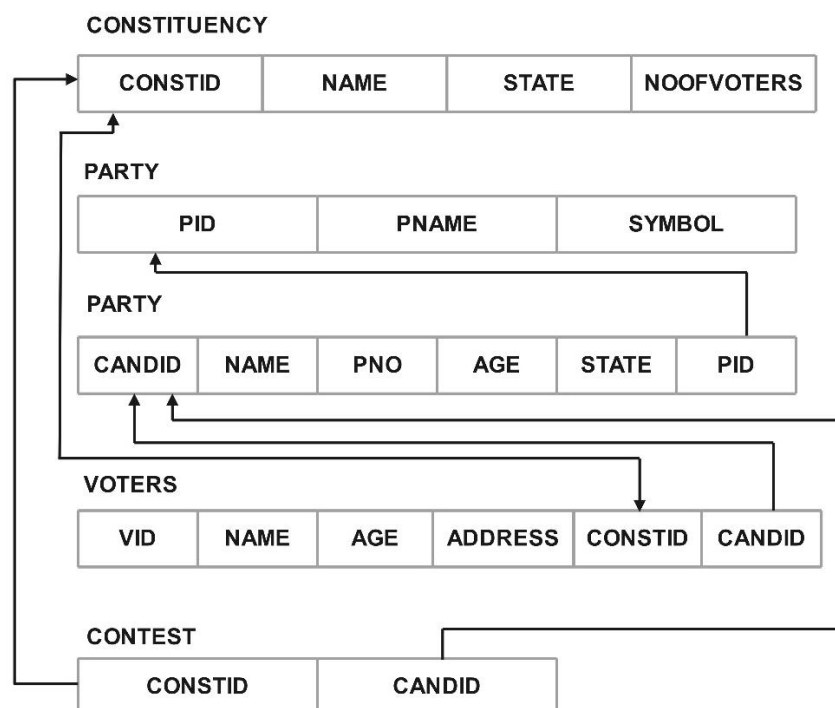
TNAME
CSK

4. A country wants to conduct an election for the parliament. A country having many constituencies. Each constituency is identified uniquely by Constituency_id, having the Name, belongs to a state, Number_of_voters. A constituency can have many voters. Each voter is uniquely identified by using Voter_id, having the Name, age, address (involves Houseno,city,state,pincode). Each voter belongs to only one constituency. There are many candidates contesting in the election. Each candidates are uniquely identified by using candidate_id, having Name, phone_no, age, state. A candidate belongs to only one party. There are many parties. Each party is uniquely identified by using Party_id, having Party_Name, Party_symbol. A candidate can contest from many constituencies under a same party. A party can have many candidates contesting from different constituencies. No constituency having the candidates from the same party. A constituency can have many contesting candidates belongs to different parties. Each voter votes only one candidate of his/her constituency.

Queries:

- List the details of the candidates who are contesting from more than one constituencies which are belongs to different states.
- Display the state name having maximum number of constituencies.
- Create a stored procedure to insert the tuple into the voter table by checking the voter age. If voter's age is at least 18 years old, then insert the tuple into the voter else display the "Not an eligible voter msg".
- Create a stored procedure to display the number_of_voters in the specified constituency. Where the constituency name is passed as an argument to the stored procedure.
- Create a TRIGGER to UPDATE the count of "Number_of_voters" of the respective constituency in "CONSTITUENCY" table, AFTER inserting a tuple into the "VOTERS" table.

SCHEMA



```
CREATE TABLE CONSTITUENCY(  
    CONSTID INT PRIMARY KEY,  
    NAME VARCHAR(30),  
    STATE VARCHAR(30),  
    NOOFVOTERS INT  
);
```

```
INSERT INTO CONSTITUENCY VALUES(1,'BANGALORE','KARNATAKA',2);  
INSERT INTO CONSTITUENCY VALUES(2,'SHIMOGA','KARNATAKA',2);  
INSERT INTO CONSTITUENCY VALUES(3,'HYDARABAD','TELANGANA',3);
```

```
CREATE TABLE PARTY(  
    PID INT PRIMARY KEY,  
    PNAME VARCHAR(30),  
    SYMBOL VARCHAR(30)  
);
```

```
INSERT INTO PARTY VALUES(1,'CONGRESS','PALM');  
INSERT INTO PARTY VALUES(2,'BJP','LOTUS');
```

```
CREATE TABLE CANDIDATE(  
    CANDID INT PRIMARY KEY,  
    NAME VARCHAR(30),  
    PNO VARCHAR(30),  
    AGE INT,  
    STATE VARCHAR(30),  
    PID INT REFERENCES PARTY(PID)  
);
```

```
INSERT INTO CANDIDATE VALUES(1,'AMAR','8660639260',35,'KARNATAKA',1);  
INSERT INTO CANDIDATE VALUES(2,'ANTONY','8660639262',37,'TELANGANA',2);
```

```
CREATE TABLE VOTERS(  
    VID INT,  
    NAME VARCHAR(30),  
    AGE VARCHAR(30),  
    ADDRESS VARCHAR(150),  
    CONSTID INT REFERENCES CONSTITUENCY(CONSTID),  
    CANDID INT REFERENCES CANDIDATE(CANDID)  
);
```

```
INSERT INTO VOTERS VALUES(1,'ABHI',20,'1 BALKI BIDAR KARNATAKA 577001',1,1);  
INSERT INTO VOTERS VALUES(2,'ARIF',20,'1 BHADRAVATHI BHADRAVATHI KARNATAKA  
577201',2,2);  
INSERT INTO VOTERS VALUES(3,'MANOJ',18,'1 ANAVERI SHIMOGA KARNATAKA 577243',1,2);
```

```
CREATE TABLE CONTEST(  
    CONSTID INT REFERENCES CONSTITUENCY(CONSTID),  
    CANDID INT REFERENCES CANDIDATE(CANDID)  
);
```

```
INSERT INTO CONTEST VALUES(1,1);  
INSERT INTO CONTEST VALUES(2,2);  
INSERT INTO CONTEST VALUES(3,1);
```

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

```
SELECT *  
FROM CANDIDATE  
WHERE CANDID IN (  
    SELECT CANDID  
    FROM CONTEST CT, CONSTITUENCY CS  
    WHERE CT.CONSTID = CS.CONSTID  
    GROUP BY CANDID  
    HAVING COUNT(DISTINCT(STATE))>1  
);
```

CANDID	NAME	PNO	AGE	STATE
1	AMAR	8660639260	35	KARNATAKA

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

```
SELECT STATE FROM CONSTITUENCY  
GROUP BY STATE  
HAVING COUNT(STATE) = (  
    SELECT MAX(COUNTS) FROM (  
        SELECT COUNT(*) COUNTS  
        FROM CONSTITUENCY  
        GROUP BY STATE) COUNTER  
);
```

STATE
KARNATAKA

iii. Create a stored procedure to insert the tuple into the voter table by checking the voter age. If voter's age is at least 18 years old, then insert the tuple into the voter else display the "Not an eligible voter msg".

```
CREATE OR REPLACE PROCEDURE CHECKAGE(ID IN NUMBER,AGE IN NUMBER)
AS
BEGIN
    IF AGE>=18 THEN
        INSERT INTO VOTERS(VID,AGE)VALUES(ID,AGE);
    ELSE
        DBMS_OUTPUT.PUT_LINE('NOT AN ELIGIBLE VOTER');
    END IF;
END;
```

Procedure created.

SQL> **SET SERVEROUTPUT ON;**

SQL> **EXEC CHECKAGE(4,17);**

NOT AN ELIGIBLE VOTER

PL/SQL procedure successfully completed.

SQL> **EXEC CHECKAGE(3,22);**

PL/SQL procedure successfully completed.

FOR MYSQL

```
DELIMITER //

CREATE OR REPLACE PROCEDURE CHECKAGE(IN ID INT, IN AGE INT)
BEGIN
    IF AGE >= 18 THEN
        INSERT INTO VOTERS (VID, AGE) VALUES (ID, AGE);
    ELSE
        SELECT 'NOT AN ELIGIBLE VOTER' AS Message;
    END IF;
END//

DELIMITER ;
```

CALL CHECKAGE(5,12)

Message

NOT AN ELIGIBLE VOTER

iv. Create a stored procedure to display the number_of_voters in the specified constituency.

Where the constituency name is passed as an argument to the stored procedure.

FOR ORACLE

```
CREATE OR REPLACE PROCEDURE COUNTVOTERS(CNAME IN VARCHAR)
AS
    CNT INTEGER;
BEGIN
    SELECT NOOFVOTERS INTO CNT
    FROM CONSTITUENCY
    WHERE NAME = CNAME;
    DBMS_OUTPUT.PUT_LINE ( 'TOTAL VOTERS ARE: ' || CNT);
END;
/
```

Procedure created.

```
SQL> EXEC COUNTVOTERS('SHIMOGA');
```

TOTAL VOTERS ARE: 2

PL/SQL procedure successfully completed.

FOR MYSQL

```
DELIMITER //

CREATE PROCEDURE COUNTVOTERS(IN CNAME VARCHAR(255))
BEGIN
    DECLARE CNT INT;

    SELECT NOOFVOTERS INTO CNT
    FROM CONSTITUENCY
    WHERE NAME = CNAME;

    IF CNT IS NOT NULL THEN
        SELECT CONCAT('TOTAL VOTERS ARE: ', CNT) AS Message;
    END IF;
END//
```

```
CALL COUNTVOTERS('SHIMOGA');
```

Message

TOTAL VOTERS ARE: 2

v. Create a TRIGGER to UPDATE the count of "Number_of_voters" of the respective constituency in "CONSTITUENCY" table , AFTER inserting a tuple into the "VOTERS" table.

FOR ORACLE

```
CREATE OR REPLACE TRIGGER UPDATECOUNT
AFTER INSERT ON VOTERS
FOR EACH ROW
BEGIN
    UPDATE CONSTITUENCY
    SET NOOFVOTERS = NOOFVOTERS + 1
    WHERE CONSTID=:NEW.CONSTID;
END;
/
```

Trigger created.

FOR MYSQL

```
FOR MYSQL
DELIMITER //

CREATE TRIGGER UPDATECOUNT AFTER INSERT ON VOTERS
FOR EACH ROW
BEGIN
    UPDATE CONSTITUENCY
    SET NOOFVOTERS = NOOFVOTERS + 1
    WHERE CONSTID = NEW.CONSTID;
END//

DELIMITER ;
```

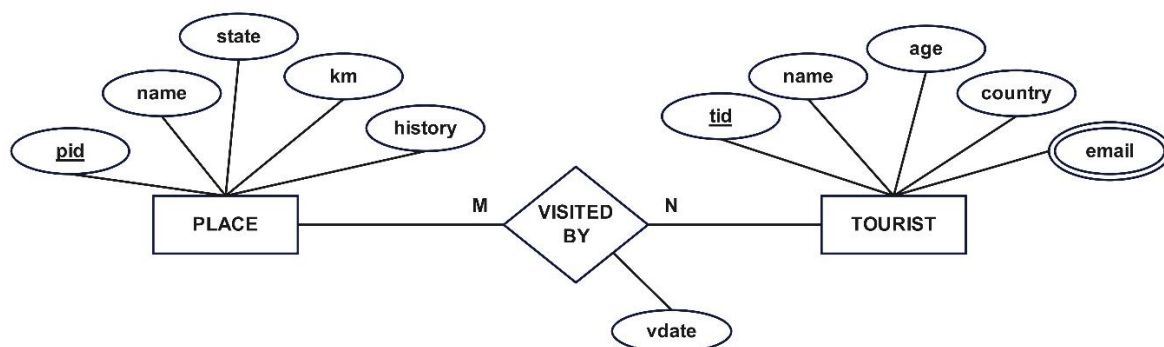
```
INSERT INTO VOTERS VALUES(10,'AKSHAY',22,'1 BALKI BIDAR KARNATAKA 577002',1,1);
```

5. Design an ER-diagram for the following scenario, Convert the same into a relational model, normalize Relations into a suitable Normal form and then solve the following queries. A country can have many Tourist places . Each Tourist place is identified by using tourist_place_id, having a name, belongs to a state, Number of kilometers away from the capital city of that state, history. There are many Tourists visits tourist places every year. Each tourist is identified uniquely by using Tourist_id, having a Name, age, Country and multiple emailids. A tourist visits many Tourist places, it is also required to record the visted_date in the database. A tourist can visit a Tourist place many times at different dates. A Tourist place can be visited by many tourists either in the same date or at different dates.

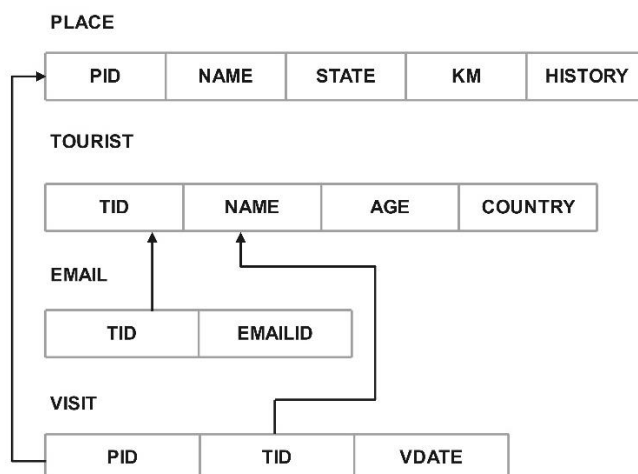
Queries:

- List the state name which is having maximum number of tourist places.
- List details of Tourist place where maximum number of tourists visited.
- List the details of tourists visited all tourist places of the state "KARNATAKA".
- Display the details of the tourists visited at least one tourist place of the state, but visited all states tourist places.
- Display the details of the tourist place visited by the tourists of all country.

ER DIAGRAM



SCHEMA



```
CREATE TABLE TOURIST (  
    TID INT PRIMARY KEY,  
    NAME VARCHAR(30),  
    AGE INT,  
    COUNTRY VARCHAR(30)  
);
```

```
INSERT INTO TOURIST VALUES (1,'AMAR',21,'INDIA');  
INSERT INTO TOURIST VALUES (2,'AKBAR',22,'INDIA');  
INSERT INTO TOURIST VALUES (3,'ANTONY',20,'UK');  
INSERT INTO TOURIST VALUES (4,'RAM',25,'US');
```

```
CREATE TABLE EMAIL(  
    TOURISTID INT REFERENCES TOURIST(TID),  
    EMAIL VARCHAR(50)  
);
```

```
INSERT INTO EMAIL VALUES(1,'AMAR@GMAIL.COM');  
INSERT INTO EMAIL VALUES(1,'AMAR.P@GMAIL.COM');  
INSERT INTO EMAIL VALUES(2,'AKBARAMAR@GMAIL.COM');  
INSERT INTO EMAIL VALUES(3,'ANTONY@GMAIL.COM');  
INSERT INTO EMAIL VALUES(4,'RAM@GMAIL.COM');
```

```
CREATE TABLE PLACE(  
    PID INT PRIMARY KEY,  
    NAME VARCHAR(50),  
    STATE VARCHAR(50),  
    KM INT,  
    HISTORY VARCHAR(200)  
);
```

```
INSERT INTO PLACE VALUES(1,'JOG FALSE','KARNATAKA',400,'WATER FALSE');  
INSERT INTO PLACE VALUES(2,'MYSORE','KARNATAKA',150,'PALACE');  
INSERT INTO PLACE VALUES(3,'TAJMAHAL','DELHI',100,'BURIAL');  
INSERT INTO PLACE VALUES(4,'TIRUPATHI','AP',50,'TEMPLE');
```

```
CREATE TABLE VISIT(  
    TID INT REFERENCES TOURIST(TID),  
    PID INT REFERENCES PLACE(PID),  
    VISITDATE DATE  
);
```

```
INSERT INTO VISIT VALUES(1,1,'12-JAN-2022');  
INSERT INTO VISIT VALUES(2,1,'12-JAN-2022');  
INSERT INTO VISIT VALUES(3,1,'12-JAN-2022');
```

```
INSERT INTO VISIT VALUES(4,1,'12-JAN-2022');
INSERT INTO VISIT VALUES(2,2,'12-JAN-2022');
INSERT INTO VISIT VALUES(2,3,'12-JAN-2022');
INSERT INTO VISIT VALUES(2,4,'12-JAN-2022');
```

NOTE: IF ABOVE DATE FORMAT NOT WORKS TRY THIS ONE

INSERT INTO VISIT VALUES(2,4,'2022-01-12');

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

```
SELECT STATE
FROM PLACE
GROUP BY STATE
HAVING COUNT(*) = (
    SELECT MAX(COUNTS) FROM (
        SELECT COUNT(*) COUNTS
        FROM PLACE
        GROUP BY STATE) COUNTER
);
```

STATE

KARNATAKA

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

```
SELECT *
FROM PLACE
WHERE PID IN (
    SELECT PID
    FROM VISIT
    GROUP BY PID
    HAVING COUNT(*) = (
        SELECT MAX(COUNTS) FROM (
            SELECT COUNT(*) COUNTS
            FROM VISIT
            GROUP BY PID) COUNTER
    )
);
```

PID NAME STATE KM HISTORY

1 JOG FALSE KARNATAKA 400 WATER FALSE

iii. List the details of tourists visited all tourist places of the state "KARNATAKA".

```
SELECT T.NAME, AGE, COUNTRY
FROM VISIT V, PLACE P, TOURIST T
WHERE V.PID = P.PID AND V.TID = T.TID AND STATE = 'KARNATAKA'
GROUP BY T.NAME, AGE, COUNTRY
HAVING COUNT(DISTINCT V.PID) = (
    SELECT COUNT(*)
    FROM PLACE
    WHERE STATE = 'KARNATAKA'
);
```

NAME	AGE	COUNTRY
AKBAR	22	INDIA

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

```
SELECT T.NAME, AGE, COUNTRY
FROM VISIT V, PLACE P, TOURIST T
WHERE V.PID = P.PID AND V.TID = T.TID
GROUP BY T.NAME, AGE, COUNTRY
HAVING COUNT(DISTINCT STATE) = (
    SELECT COUNT(DISTINCT STATE)
    FROM PLACE
);
```

NAME	AGE	COUNTRY
AKBAR	22	INDIA

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

```
SELECT P.NAME, KM, HISTORY
FROM VISIT V, PLACE P, TOURIST T
WHERE V.PID = P.PID AND V.TID = T.TID
GROUP BY P.NAME, KM, HISTORY
HAVING COUNT(DISTINCT COUNTRY) = (
    SELECT COUNT(DISTINCT COUNTRY)
    FROM TOURIST
);
```

NAME	KM	HISTORY
JOG FALSE	400	WATER FALSE