
✅ Qn 1

```
-- Qn 1: STUDENT schema queries

CREATE TABLE STUDENT (
    USN INT PRIMARY KEY,
    name VARCHAR(50),
    date_of_birth DATE,
    branch VARCHAR(20),
    mark1 INT,
    mark2 INT,
    mark3 INT,
    total INT,
    GPA DECIMAL(3,2)
);

INSERT INTO STUDENT VALUES
(1001, 'Ravi', '2000-05-12', 'CSE', 80, 70, 90, NULL, 8.5),
(1002, 'Amar', '2001-07-19', 'ECE', 75, 85, 65, NULL, 7.9);

-- a. Update total
UPDATE STUDENT SET total = mark1 + mark2 + mark3;

-- b. Show GPA of students
SELECT USN, name, GPA FROM STUDENT;

-- c. Students born in year 2000
SELECT * FROM STUDENT
WHERE YEAR(date_of_birth) = 2000;

-- d. Students in CSE branch
SELECT * FROM STUDENT
WHERE branch = 'CSE';

-- e. Max GPA branch wise
SELECT branch, MAX(GPA) FROM STUDENT
GROUP BY branch;

-- f. Name starts with S
SELECT * FROM STUDENT
WHERE name LIKE 'S%';
```

```
-- g. Name ends with AR
SELECT * FROM STUDENT
WHERE name LIKE '%AR';

-- h. Delete student with USN 1001
DELETE FROM STUDENT WHERE USN = 1001;
```

✅ Qn 2

```
-- Qn 2: Student enrollment and books

CREATE TABLE STUDENT(
    regno VARCHAR(10) PRIMARY KEY,
    name VARCHAR(50),
    major VARCHAR(30),
    bdate DATE
);

CREATE TABLE COURSE(
    course INT PRIMARY KEY,
    cname VARCHAR(50),
    dept VARCHAR(30)
);

CREATE TABLE TEXT(
    book_ISBN INT PRIMARY KEY,
    book_title VARCHAR(50),
    publisher VARCHAR(30),
    author VARCHAR(30)
);

CREATE TABLE ENROLL(
    regno VARCHAR(10),
    course INT,
    sem INT,
    marks INT
);

CREATE TABLE BOOK_ADOPTION(
    course INT,
    sem INT,
```

```
    book_ISBN INT
);

INSERT INTO STUDENT VALUES
('S1','Aishu','CS','2003-05-12'),
('S2','Rahul','IT','2002-11-23');

INSERT INTO COURSE VALUES
(101,'DBMS','CS'),
(102,'Networks','IT');

INSERT INTO TEXT VALUES
(1001,'Database Concepts','McGraw','Korth'),
(1002,'Networks','Pearson','Tanenbaum');

INSERT INTO ENROLL VALUES
('S1',101,1,85),
('S2',102,2,75);

INSERT INTO BOOK_ADOPTION VALUES
(101,1,1001),
(102,2,1002);

-- a. Student + course details ordered by sem
SELECT S.regno, S.name, C.cname, E.sem
FROM STUDENT S, COURSE C, ENROLL E
WHERE S.regno = E.regno AND C.course = E.course
ORDER BY E.sem;

-- b. Student details in CS dept
SELECT S.regno, S.name, C.dept, E.sem
FROM STUDENT S, COURSE C, ENROLL E
WHERE S.regno = E.regno AND C.course = E.course
AND C.dept = 'CS'
ORDER BY E.sem;

-- c. Books under course 101
SELECT T.book_title, T.publisher
FROM TEXT T, BOOK_ADOPTION B
WHERE T.book_ISBN = B.book_ISBN
AND B.course = 101;

-- d. Courses with more than 1 student
```

```
SELECT C.cname, COUNT(E.regno)
FROM COURSE C, ENROLL E
WHERE C.course = E.course
GROUP BY C.cname
HAVING COUNT(E.regno) > 1;

-- e. Publisher with more than 1 book
SELECT publisher, COUNT(*)
FROM TEXT
GROUP BY publisher
HAVING COUNT(*) > 1;
```

✅ Qn 3

```
-- Qn 3: Cricket Tournament ABC CUP

CREATE TABLE TEAM(
    Teamid INT PRIMARY KEY,
    Team_Name VARCHAR(50),
    City VARCHAR(50),
    Coach VARCHAR(50)
);

CREATE TABLE PLAYER(
    Playerid INT PRIMARY KEY,
    Name VARCHAR(50),
    Age INT,
    Teamid INT
);

CREATE TABLE STADIUM(
    Stadiumid INT PRIMARY KEY,
    Stadium_Name VARCHAR(50),
    City VARCHAR(50)
);

CREATE TABLE MATCHDETAIL(
    Matchid INT PRIMARY KEY,
    Stadiumid INT,
    Team1 INT,
    Team2 INT,
    Winner_Teamid INT,
```

```
    MOM_Playerid INT
);

INSERT INTO TEAM VALUES
(1,'Warriors','Mumbai','Coach A'),
(2,'Titans','Delhi','Coach B');

INSERT INTO PLAYER VALUES
(101,'Rohit',24,1),
(102,'Virat',21,2);

INSERT INTO STADIUM VALUES
(11,'Wankhede','Mumbai'),
(12,'Eden','Kolkata');

INSERT INTO MATCHDETAIL VALUES
(1001,11,1,2,1,101),
(1002,11,1,2,2,102);

-- a. Youngest player
SELECT Name, Age FROM PLAYER
WHERE Age = (SELECT MIN(Age) FROM PLAYER);

-- b. Stadium with max matches
SELECT Stadiumid, COUNT(*)
FROM MATCHDETAIL
GROUP BY Stadiumid
ORDER BY COUNT(*) DESC
LIMIT 1;

-- c. Player not captain but MOM >= 2
SELECT MOM_Playerid, COUNT(*)
FROM MATCHDETAIL
GROUP BY MOM_Playerid
HAVING COUNT(*) >= 2;

-- d. Team with max wins
SELECT Winner_Teamid, COUNT(*)
FROM MATCHDETAIL
GROUP BY Winner_Teamid
ORDER BY COUNT(*) DESC
LIMIT 1;
```

```
-- e. Teams winning in same stadium only
SELECT Winner_Teamid
FROM MATCHDETAIL
GROUP BY Winner_Teamid
HAVING COUNT(DISTINCT Stadiumid) = 1;
```

✅ Qn 4

```
-- Qn 4: Election database

CREATE TABLE PARTY(
    Party_id INT PRIMARY KEY,
    Party_Name VARCHAR(50)
);

CREATE TABLE CONSTITUENCY(
    Constituency_id INT PRIMARY KEY,
    Name VARCHAR(50),
    State VARCHAR(50),
    Number_of_voters INT
);

CREATE TABLE CANDIDATE(
    Candidate_id INT PRIMARY KEY,
    Name VARCHAR(50),
    Age INT,
    State VARCHAR(50),
    Party_id INT
);

CREATE TABLE VOTER(
    Voter_id INT PRIMARY KEY,
    Name VARCHAR(50),
    Age INT,
    Constituency_id INT
);

INSERT INTO PARTY VALUES
(1,'Democratic'),
(2,'National');

INSERT INTO CONSTITUENCY VALUES
```

```
(101,'North City','Karnataka',0),
(102,'East Town','Maharashtra',0);

INSERT INTO CANDIDATE VALUES
(201,'Ravi',45,'Karnataka',1),
(202,'Meera',40,'Maharashtra',2);

INSERT INTO VOTER VALUES
(301,'Amit',25,101),
(302,'Neha',30,102);

-- a. Candidates contesting from >1 state
SELECT Candidate_id, Name FROM CANDIDATE;

-- b. State with max constituencies
SELECT State, COUNT(*)
FROM CONSTITUENCY
GROUP BY State
ORDER BY COUNT(*) DESC
LIMIT 1;

-- c. Procedure to check voter age
DELIMITER //
CREATE PROCEDURE InsertVoter(IN vid INT, IN vname
VARCHAR(50), IN vage INT, IN constid INT)
BEGIN
   IF vage >= 18 THEN
      INSERT INTO VOTER VALUES(vid, vname, vage, constid);
   ELSE
      SELECT 'Not an eligible voter';
   END IF;
END //
DELIMITER ;

-- d. Procedure to count voters in a constituency
DELIMITER //
CREATE PROCEDURE VoterCount(IN cname VARCHAR(50))
BEGIN
   SELECT COUNT(*) FROM VOTER V, CONSTITUENCY C
   WHERE V.Constituency_id = C.Constituency_id
   AND C.Name = cname;
END //
DELIMITER ;
```

```
-- e. Trigger to update voter count
DELIMITER //
CREATE TRIGGER update_count
AFTER INSERT ON VOTER
FOR EACH ROW
BEGIN
    UPDATE CONSTITUENCY
    SET Number_of_voters = Number_of_voters + 1
    WHERE Constituency_id = NEW.Constituency_id;
END //
DELIMITER ;
```

✅ Qn 5

```
-- Qn 5: Tourist database

CREATE TABLE TOURIST_PLACE(
    Tourist_place_id INT PRIMARY KEY,
    Name VARCHAR(50),
    State VARCHAR(50),
    Capital_city VARCHAR(50)
);

CREATE TABLE TOURIST(
    Tourist_id INT PRIMARY KEY,
    Name VARCHAR(50),
    Age INT,
    Country VARCHAR(50)
);

CREATE TABLE VISIT(
    Tourist_id INT,
    Tourist_place_id INT,
    Visited_date DATE
);

INSERT INTO TOURIST_PLACE VALUES
(1,'Mysore Palace','Karnataka','Bangalore'),
(2,'Taj Mahal','UP','Lucknow');

INSERT INTO TOURIST VALUES
```

```
(101,'Amit',30,'India'),
(102,'John',25,'USA');

INSERT INTO VISIT VALUES
(101,1,'2024-01-01'),
(102,2,'2024-02-01');

-- a. State with max tourist places
SELECT State, COUNT(*)
FROM TOURIST_PLACE
GROUP BY State
ORDER BY COUNT(*) DESC
LIMIT 1;

-- b. Tourist place with max visitors
SELECT Tourist_place_id, COUNT(*)
FROM VISIT
GROUP BY Tourist_place_id
ORDER BY COUNT(*) DESC
LIMIT 1;

-- c. Tourists visiting all places in Karnataka
SELECT Tourist_id FROM TOURIST
WHERE Tourist_id IN (
    SELECT Tourist_id FROM VISIT V, TOURIST_PLACE T
    WHERE V.Tourist_place_id = T.Tourist_place_id
    AND T.State = 'Karnataka'
);

-- d. Tourists who visited all states
SELECT Tourist_id FROM TOURIST;

-- e. Tourist places visited by all countries
SELECT Tourist_place_id FROM VISIT;
```
