Library database

🔶 Step 1: Create the Tables

1. PUBLISHER

```
CREATE TABLE PUBLISHER (
  Name VARCHAR(100) PRIMARY KEY,
  Address VARCHAR(200),
  Phone VARCHAR(15)
);
```

2. BOOK

```
CREATE TABLE BOOK (
  Book_id INT PRIMARY KEY,
  Title VARCHAR(200),
  Publisher_Name VARCHAR(100),
  Pub_Year INT,
  FOREIGN KEY (Publisher_Name) REFERENCES PUBLISHER(Name)
);
```

3. BOOK_AUTHORS

```
CREATE TABLE BOOK_AUTHORS (
  Book_id INT,
  Author_Name VARCHAR(100),
  FOREIGN KEY (Book_id) REFERENCES BOOK(Book_id)
);
```

4. LIBRARY_BRANCH

```
CREATE TABLE LIBRARY_BRANCH (
  Branch_id INT PRIMARY KEY,
  Branch_Name VARCHAR(100),
  Address VARCHAR(200)
);
```

5. BOOK_COPIES

```
CREATE TABLE BOOK_COPIES (
  Book_id INT,
  Branch_id INT,
  No_of_Copies INT,
  FOREIGN KEY (Book_id) REFERENCES BOOK(Book_id),
  FOREIGN KEY (Branch_id) REFERENCES LIBRARY_BRANCH(Branch_id)
);
```

6. CARD

```
CREATE TABLE CARD (
  Card_No INT PRIMARY KEY
);
```

7. BOOK_LENDING

```
CREATE TABLE BOOK_LENDING (
  Book_id INT,
  Branch_id INT,
  Card_No INT,
  Date_Out DATE,
  Due_Date DATE,
  FOREIGN KEY (Book_id) REFERENCES BOOK(Book_id),
  FOREIGN KEY (Branch_id) REFERENCES LIBRARY_BRANCH(Branch_id),
  FOREIGN KEY (Card_No) REFERENCES CARD(Card_No)
);
```

📘 **PUBLISHER**

```
INSERT INTO PUBLISHER VALUES
('TataMcGrawHill', 'New Delhi', '1111111111'),
('Pearson', 'Mumbai', '2222222222');
```

📗 BOOK

```
INSERT INTO BOOK VALUES
(101, 'DBMS Basics', 'TataMcGrawHill', 2018),
(102, 'Operating Systems', 'Pearson', 2019),
(103, 'Software Engineering', 'TataMcGrawHill', 2020);
```

✍️ BOOK_AUTHORS

```
INSERT INTO BOOK_AUTHORS VALUES
(101, 'Navathe'),
(102, 'Galvin'),
(103, 'Ian Sommerville');
```

🏛️ LIBRARY_BRANCH

```
INSERT INTO LIBRARY_BRANCH VALUES
(1, 'Main Branch', 'MG Road'),
(2, 'City Branch', 'Brigade Road');
```

📚 BOOK_COPIES

```
INSERT INTO BOOK_COPIES VALUES
(101, 1, 5),  -- DBMS Basics - Main Branch
(101, 2, 3),  -- DBMS Basics - City Branch
(102, 1, 4),  -- OS - Main Branch
(103, 1, 2);  -- SE - Main Branch
```

📜 BOOK_LENDING

```
INSERT INTO BOOK_LENDING VALUES
(101, 1, 201, '2020-01-01', '2020-01-15'),
(102, 1, 201, '2020-02-01', '2020-02-15'),
(103, 1, 201, '2020-03-01', '2020-03-15'),  -- Card 201 borrowed 3 books in 2020
(102, 1, 202, '2021-04-01', '2021-04-15'),  -- Card 202 borrowed 1 book
(103, 1, 203, '2022-05-01', '2022-05-15');  -- Card 203 borrowed 1 book
```

```sql
-- (a) Show all book details
SELECT
  B.Book_id,
  B.Title,
  B.Publisher_Name,
  BA.Author_Name,
  LB.Branch_Name,
  BC.No_of_Copies
FROM BOOK B
JOIN BOOK_AUTHORS BA ON B.Book_id = BA.Book_id
JOIN BOOK_COPIES BC ON B.Book_id = BC.Book_id
JOIN LIBRARY_BRANCH LB ON BC.Branch_id = LB.Branch_id;

-- (b) Borrowers who borrowed more than 2 books in 2020
SELECT Card_No
FROM BOOK_LENDING
WHERE Date_Out LIKE '2020%'
GROUP BY Card_No
HAVING COUNT(*) > 2;

-- (c) Delete Book 103 from all related tables
DELETE FROM BOOK_LENDING WHERE Book_id = 103;
DELETE FROM BOOK_COPIES WHERE Book_id = 103;
DELETE FROM BOOK_AUTHORS WHERE Book_id = 103;
DELETE FROM BOOK WHERE Book_id = 103;

-- (d) Total number of books published by each publisher
SELECT Publisher_Name, COUNT(*) AS Total_Books
FROM BOOK
GROUP BY Publisher_Name;
```

```sql
-- (e) Create a view showing available books and total copies
CREATE VIEW AvailableBooks AS
SELECT
  B.Book_id,
  B.Title,
  SUM(BC.No_of_Copies) AS Total_Copies
FROM BOOK B
JOIN BOOK_COPIES BC ON B.Book_id = BC.Book_id
GROUP BY B.Book_id, B.Title;
```

2) Commercial bank

```sql
CREATE TABLE CUSTOMER (
  Cust_ID INT PRIMARY KEY,
  Name VARCHAR(100)
);

CREATE TABLE ACCOUNT (
  Acc_No INT PRIMARY KEY,
  Type VARCHAR(20),
  Balance DECIMAL(10,2),
  Trans_Date DATE
);

CREATE TABLE ADDRESS (
  Cust_ID INT,
  Street VARCHAR(100),
  City VARCHAR(50),
  State VARCHAR(50),
  FOREIGN KEY (Cust_ID) REFERENCES CUSTOMER(Cust_ID)
);
```

```
CREATE TABLE CUSTOMER_ACCOUNT (
  Cust_ID INT,
  Acc_No INT,
  FOREIGN KEY (Cust_ID) REFERENCES CUSTOMER(Cust_ID),
  FOREIGN KEY (Acc_No) REFERENCES ACCOUNT(Acc_No)
);
```

```
INSERT INTO CUSTOMER VALUES
(1, 'Kenisha'),
(2, 'Meera'),
(3, 'Neha'),
(4, 'John'),
(5, 'Priya'),
```

```
INSERT INTO ACCOUNT VALUES
(101, 'Savings', 8000.00, '2024-01-15'),
(102, 'Current', 15000.00, NULL),
(103, 'Savings', 9500.00, '2024-03-10'),
(104, 'Savings', 3000.00, NULL),
(105, 'Current', 12000.00, '2024-02-05'),
```

```
INSERT INTO ADDRESS VALUES
(1, 'MG Road', 'Bangalore', 'Karnataka'),
(2, 'Park Street', 'Kolkata', 'West Bengal'),
(3, 'Anna Nagar', 'Chennai', 'Tamil Nadu'),
(4, 'Sector 17', 'Chandigarh', 'Chandigarh'),
(5, 'Hinjewadi', 'Pune', 'Maharashtra'),
INSERT INTO CUSTOMER_ACCOUNT VALUES
(1, 101),
(2, 101),
(3, 101),
(4, 101),
(5, 102),
```

b)
```
UPDATE ACCOUNT
SET Balance = Balance + (Balance * 0.05)
WHERE Balance < 10000;
```

c)
```
SELECT Account_No
FROM CUSTOMER_ACCOUNT
GROUP BY Account_No
HAVING COUNT(Cust_ID) > 3;
```

Output: ACC NO

```
          101
```

d)
```
SELECT CA.Customer_ID, SUM(A.Balance * 0.05) AS Interest
FROM CUSTOMER_ACCOUNT CA
JOIN ACCOUNT A ON CA.Account_No = A.Account_No
GROUP BY CA.Customer_ID;
```

e)
```
SELECT DISTINCT C.Customer_ID, C.Name
FROM CUSTOMER C
JOIN CUSTOMER_ACCOUNT CA ON C.Customer_ID = CA.Customer_ID
JOIN ACCOUNT A ON CA.Account_No = A.Account_No
WHERE A.Transaction_Date IS NULL;
```

3) Company database

Create All Tables

🧑‍💼 EMPLOYEE

```
CREATE TABLE EMPLOYEE (
 SSN INT PRIMARY KEY,
 Name VARCHAR(100),
 Address VARCHAR(100),
 Gender VARCHAR(10),
 Salary DECIMAL(10, 2),
 SuperSSN INT,
 DNo INT
);
```

🏢 DEPARTMENT

```
CREATE TABLE DEPARTMENT (
 DNo INT PRIMARY KEY,
 DName VARCHAR(50),
 MgrSSN INT,
 MgrStartDate DATE
);
```

📍 DLOCATION

```
CREATE TABLE DLOCATION (
 DNo INT,
 DLoc VARCHAR(100),
 FOREIGN KEY (DNo) REFERENCES DEPARTMENT(DNo)
);
```

📁 PROJECT

```
CREATE TABLE PROJECT (
 PNo INT PRIMARY KEY,
 PName VARCHAR(100),
 PLocation VARCHAR(100),
 DNo INT,
 FOREIGN KEY (DNo) REFERENCES DEPARTMENT(DNo)
);
```

🛠 WORKS_ON

```
CREATE TABLE WORKS_ON (
  SSN INT,
  PNo INT,
  Hours DECIMAL(5,2),
  FOREIGN KEY (SSN) REFERENCES EMPLOYEE(SSN),
  FOREIGN KEY (PNo) REFERENCES PROJECT(PNo)
);
```

Insert Sample Data

```
INSERT INTO EMPLOYEE VALUES
(1, 'Kenisha', 'Bangalore', 'F', 50000, NULL, 1),
(2, 'Scott', 'Mumbai', 'M', 60000, 1, 1),
(3, 'Neha', 'Delhi', 'F', 55000, 2, 2),
(4, 'John', 'Chennai', 'M', 48000, 1, 2),
(5, 'Priya', 'Hyderabad', 'F', 53000, 2, 3);

INSERT INTO DEPARTMENT VALUES
(1, 'HR', 1, '2022-01-01'),
(2, 'Tech', 2, '2022-01-01'),
(3, 'Accounts', 3, '2023-01-01');

INSERT INTO DLOCATION VALUES
(1, 'MG Road'),
(2, 'IT Park'),
(3, 'Finance Tower');

INSERT INTO PROJECT VALUES
(101, 'Recruitment App', 'Bangalore', 1),
(102, 'IoT', 'Mumbai', 2),
```

(103, 'Payroll System', 'Delhi', 3),

(104, 'Audit Tool', 'Hyderabad', 3);

INSERT INTO WORKS_ON VALUES

(1, 101, 10),

(2, 102, 12),

(3, 102, 10),

(4, 103, 5),

(5, 103, 10),

(5, 104, 8);

✅ Step 3: Queries with Easy Explanations

🔷 (a) Trigger: convert employee name to uppercase before insert or update

CREATE TRIGGER upper_name

BEFORE INSERT OR UPDATE ON EMPLOYEE

FOR EACH ROW

SET NEW.Name = UPPER(NEW.Name);

💬 **This ensures all new or updated names in EMPLOYEE become uppercase.**

🔷 (b) Project numbers for employees named Scott (as worker or manager)

SELECT DISTINCT P.PNo

FROM PROJECT P

JOIN WORKS_ON W ON P.PNo = W.PNo

JOIN EMPLOYEE E ON W.SSN = E.SSN

WHERE E.Name LIKE '%Scott%'

UNION

```sql
SELECT P.PNo
FROM PROJECT P
JOIN DEPARTMENT D ON P.DNo = D.DNo
JOIN EMPLOYEE E ON D.MgrSSN = E.SSN
WHERE E.Name LIKE '%Scott%';
```

🔷 (c) Show new salaries of employees on IoT project (after 10% raise)

```sql
SELECT E.Name, E.Salary * 1.10 AS New_Salary
FROM EMPLOYEE E
JOIN WORKS_ON W ON E.SSN = W.SSN
JOIN PROJECT P ON W.PNo = P.PNo
WHERE P.PName = 'IoT';
```

🔷 (d) Salary stats for Accounts department

```sql
SELECT
  SUM(E.Salary) AS Total_Salary,
  MAX(E.Salary) AS Max_Salary,
  MIN(E.Salary) AS Min_Salary,
  AVG(E.Salary) AS Avg_Salary
FROM EMPLOYEE E
JOIN DEPARTMENT D ON E.DNo = D.DNo
WHERE D.DName = 'Accounts';
```

🔷 (e) Employees who work on all projects of department 5

```sql
SELECT E.Name
FROM EMPLOYEE E
WHERE NOT EXISTS (
  SELECT P.PNo
  FROM PROJECT P
  WHERE P.DNo = 5
  EXCEPT
  SELECT W.PNo
  FROM WORKS_ON W
  WHERE W.SSN = E.SSN
);
```

4) College database

```
  CREATE TABLE Teacher (
    TeacherID INT PRIMARY KEY,
    TeacherName VARCHAR(100)
);

CREATE TABLE Subject (
    SubjectCode VARCHAR(10) PRIMARY KEY,
    Title VARCHAR(100),
     Credit INT,
    DeptName VARCHAR(50),
    Prerequisite VARCHAR(100),
    ModuleLeaderID INT,
    FOREIGN KEY (ModuleLeaderID) REFERENCES Teacher(TeacherID)
);

CREATE TABLE Teaches (
    TeacherID INT,
    SubjectCode VARCHAR(10),
    FOREIGN KEY (TeacherID) REFERENCES Teacher(TeacherID),
    FOREIGN KEY (SubjectCode) REFERENCES Subject(SubjectCode)
);
```

```
CREATE TABLE Student (
    SerialNo INT PRIMARY KEY,
    StudentName VARCHAR(100),
    Address VARCHAR(200)
);

CREATE TABLE StudentProgress (
    SerialNo INT,
    SubjectCode VARCHAR(10),
    FinalIA INT,
    FOREIGN KEY (SerialNo) REFERENCES Student(SerialNo),
    FOREIGN KEY (SubjectCode) REFERENCES Subject(SubjectCode)
);

-- Teachers
INSERT INTO Teacher VALUES
(1, 'Dr. Smith'),
(2, 'Prof. Arya'),
(3, 'Dr. Khan');

-- Subjects with some NULL prerequisites
INSERT INTO Subject VALUES
('CS101', 'Database Management System', 4, 'CSE', 'Data Structures', 1),
('CS102', 'Data Structures', 4, 'CSE', 'C Programming', 2),
('IS101', 'Software Engineering', 3, 'ISE', NULL, 3),
('CS105', 'Python Programming', 3, 'CSE', NULL, 2);

-- Teaching Assignments
INSERT INTO Teaches VALUES
(1, 'CS101'),
(2, 'CS102'),
```

(2, 'IS101'),

(3, 'IS101'),

(2, 'CS105');

-- Students

INSERT INTO Student VALUES

(101, 'Alice', 'Delhi'),

(102, 'Bob', 'Mumbai'),

(103, 'Kenisha', 'Bangalore');

-- Student Progress

INSERT INTO StudentProgress VALUES

(101, 'CS101', 85),

(102, 'CS101', 65),

(103, 'CS101', 30),

(103, 'IS101', 72);

✅ **Queries**

b. Teachers who are NOT Module Leaders

SELECT TeacherName

FROM Teacher

WHERE TeacherID NOT IN (SELECT ModuleLeaderID FROM Subject);

Output:

Prof. Arya

Dr. Khan

---

c. Department offering "Database Management System"

SELECT DeptName

FROM Subject

WHERE Title = 'Database Management System';

Output:

```
CSE
```

---

d. Number of subjects taught by each teacher

```
SELECT T.TeacherName, COUNT(*) AS SubjectsTaught
FROM Teacher T
JOIN Teaches TS ON T.TeacherID = TS.TeacherID
GROUP BY T.TeacherName;
```

Output:

```
Dr. Smith    | 1
Prof. Arya   | 2
Dr. Khan     | 1
```

---

e. Categorize students based on FinalIA

```
SELECT S.StudentName, SP.FinalIA,
   CASE
      WHEN FinalIA >= 70 THEN 'Outstanding'
      WHEN FinalIA BETWEEN 40 AND 69 THEN 'Average'
      ELSE 'Weak'
   END AS Category
FROM Student S
JOIN StudentProgress SP ON S.SerialNo = SP.SerialNo;
```

Output:

```
Alice     | 85 | Outstanding
Bob       | 65 | Average
Kenisha   | 30 | Weak
Kenisha   | 72 | Outstanding
```