



ER Diagram on the project.

Normalization on the project

Normalization is the process of organizing data in a database to reduce redundancy and improve data integrity. For the Tiny College database, normalization up to **Third Normal Form (3NF)** was applied.

First Normal Form (1NF)

- **Rule:** All attributes must be atomic, and each record must be unique.
- **Fix:**
 - Instead of having a single attribute like StudentName, we split it into FirstName and LastName.
 - Courses are stored once per row on the Course table, instead of listing multiple courses in a single column.

Second Normal Form (2NF)

- **Rule:** Must be in 1NF and have **no partial dependency** — i.e., non-key attributes must depend on the whole primary key.
- **Fix:**
 - In the Enrollment table, the composite key is (StudentID, CourseID, Semester).
 - Grade depends on the full composite key, not on just StudentID or CourseID.

Third Normal Form (3NF)

- **Rule:** Must be in 2NF and have **no transitive dependencies** — non-key attributes must depend only on the primary key, not on other non-key attributes.
- **Fix:**
 - Suppose we stored Department Location inside the Student or Instructor table.
 - That would create a transitive dependency, since Department Location depends on DeptID, which is not the primary key of Student/Instructor.
- **Eliminates redundancy:** Department location is stored only once in the `Department` table.
- **Prevents updating anomalies:** If a department changes its location, we update it in one place, not across multiple tables.
- **Ensures data consistency:** All relationships (Student–Department, Instructor–Department, Course–Department) remain accurate.
- **Supports scalability:** Adding new students, instructors, or courses doesn't require restructuring existing tables

SQL Code

```
-- =====
-- CREATE TABLES
-- =====

CREATE TABLE Department (
    DeptID INT PRIMARY KEY,
    DeptName VARCHAR(100) NOT NULL,
    Location VARCHAR(100)
);

CREATE TABLE Student (
    StudentID INT PRIMARY KEY,
    FirstName VARCHAR(50),
    LastName VARCHAR(50),
    DOB DATE,
    Major VARCHAR(50),
    DeptID INT,
    FOREIGN KEY (DeptID) REFERENCES Department(DeptID)
);

CREATE TABLE Instructor (
    InstructorID INT PRIMARY KEY,
    FirstName VARCHAR(50),
    LastName VARCHAR(50),
    Title VARCHAR(50),
    DeptID INT,
    FOREIGN KEY (DeptID) REFERENCES Department(DeptID)
);

CREATE TABLE Course (
    CourseID INT PRIMARY KEY,
    CourseName VARCHAR(100),
    Credits INT,
    DeptID INT,
    FOREIGN KEY (DeptID) REFERENCES Department(DeptID)
);

CREATE TABLE Enrollment (
    StudentID INT,
    CourseID INT,
    Semester VARCHAR(20),
    Grade CHAR(2),
    PRIMARY KEY (StudentID, CourseID, Semester),
    FOREIGN KEY (StudentID) REFERENCES Student(StudentID),
    FOREIGN KEY (CourseID) REFERENCES Course(CourseID)
);

-- =====
-- INSERT SAMPLE DATA
-- =====

INSERT INTO Department VALUES (1, 'Computer Science', 'Building A');
INSERT INTO Department VALUES (2, 'Mathematics', 'Building B');

INSERT INTO Student VALUES (101, 'Alice', 'Johnson', '2001-04-15', 'CS', 1);
INSERT INTO Student VALUES (102, 'Bob', 'Smith', '2000-08-21', 'Math', 2);

INSERT INTO Instructor VALUES (201, 'Dr. John', 'Doe', 'Professor', 1);
INSERT INTO Instructor VALUES (202, 'Dr. Mary', 'Lee', 'Associate Prof', 2);

INSERT INTO Course VALUES (301, 'Database Systems', 3, 1);
INSERT INTO Course VALUES (302, 'Calculus I', 4, 2);

INSERT INTO Enrollment VALUES (101, 301, 'Fall 2025', 'A');
INSERT INTO Enrollment VALUES (102, 302, 'Fall 2025', 'B');
```

Output or Screenshot

The screenshot displays a database management application with a dark theme. The interface is divided into several sections:

- Top Bar:** Contains menu items like 'Pricing', 'Help', 'Import', and 'Export'.
- Left Sidebar:** Features a 'Private.DB' section with a prompt to 'Create a database linked to your account.' Below this is a 'Demo.Memory' section and a list of databases including 'SQLite', 'DuckDB', and 'PGLite'. Under 'SQLite', there's a 'Table' section listing 'Course', 'demo', 'Department', 'Enrollment', 'Instructor', and 'Student'.
- Central Panel:** Shows a 'Run' button and a 'SQLite' tab. The query editor contains the following SQL code:

```
1 SELECT * FROM Student;
2
```

Below the query, the results are displayed in a table format:

StudentID	FirstName	LastName	DOB	Major	DeptID
101	Alice	Johnson	2001-04-15	CS	1
102	Bob	Smith	2000-08-21	Math	2
- Right Panel:** Titled 'History', it shows a list of executed queries with their timestamps. The first query is 'SELECT * FROM Student;' at 23:45:30. The second query is a multi-line SQL statement at 23:45:13. The third query is a 'CREATE TABLE' statement at 23:44:24.