Part 1: Selected Scenario

Chosen Scenario:

3. Higher Education System: Students, Subjects, Instructors, Timetable, etc.

This system is designed to manage academic data in a higher education institution, including students, instructors, subjects (courses), class schedules, and student enrollments. It supports core operations such as course registration, class scheduling, instructor assignment, and performance tracking.

Part 2: Database Design and Documentation

Entity and Attribute Identification

1. Students – Information about enrolled students.

2. Subjects – Academic courses offered by departments.

3. Teachers (Instructors) – Faculty members who teach subjects.

4. Timetable – Class schedules including time, day, and room.

5. Enrollments – Manages the many-to-many relationship between students and subjects.

Table Designs

1. Table Name: Students

- Description: Stores information about enrolled students.

- Attributes:

- StudentID: INTEGER, PK, NOT NULL, UNIQUE

→ Unique identifier for each student.

- FirstName: VARCHAR(100), NOT NULL

→ First name of the student.

- LastName: VARCHAR(100), NOT NULL

→ Last name of the student.

- Email: VARCHAR(255), NOT NULL, UNIQUE

→ Student's email address (must be unique).

- DateOfBirth: DATE, NOT NULL

→ Date of birth of the student.

- EnrollmentDate: DATE, NOT NULL, DEFAULT CURRENT\_DATE

→ Date when the student was enrolled.

- Constraints:

- PK\_Students: PRIMARY KEY (StudentID)

→ Ensures each student has a unique ID.

- UQ\_StudentFullName: UNIQUE (FirstName, LastName)

→ Prevents duplicate full names (optional constraint for data quality).

- UQ\_Email: UNIQUE (Email)

→ Guarantees no two students share the same email.

- CHK\_Age: CHECK (DateOfBirth <= CURRENT\_DATE)

→ Ensures date of birth is not in the future.

2. Table Name: Subjects

- Description: Contains information about academic subjects (courses) offered.

- Attributes:

- SubjectID: INTEGER, PK, NOT NULL, UNIQUE

→ Unique identifier for each subject.

- SubjectName: VARCHAR(100), NOT NULL

→ Name of the subject (e.g., "Database Systems", "Calculus").

- Credits: INTEGER, NOT NULL

→ Number of credit hours assigned to the subject.

- Department: VARCHAR(100), NOT NULL

→ Academic department offering the subject (e.g., Computer Science, Mathematics).

- Constraints:

- PK\_Subject: PRIMARY KEY (SubjectID)

→ Ensures each subject has a unique ID.

- UQ\_SubjectName: UNIQUE (SubjectName)

→ Prevents duplicate subject names.

- CHK\_Credits: CHECK (Credits >= 1 AND Credits <= 6)

→ Ensures credit value is within a realistic range.

- CHK\_Department: NOT NULL

→ Every subject must belong to a department.

3. Table Name: Teachers

- Description: Stores data about teaching staff (instructors).

- Attribute:

- TeacherID: INTEGER, PK, NOT NULL, UNIQUE

→ Unique identifier for each instructor.

- FirstName: VARCHAR(100), NOT NULL

→ First name of the instructor.

- LastName: VARCHAR(100), NOT NULL

→ Last name of the instructor.

- Email: VARCHAR(255), NOT NULL, UNIQUE

→ Instructor's email address.

- Department: VARCHAR(100), NOT NULL

→ Department the instructor belongs to.

- Constraints:

- PK\_Teachers: PRIMARY KEY (TeacherID)

→ Ensures unique identification.

- UQ\_TeacherFullName: UNIQUE (FirstName, LastName)

→ Prevents duplicate full names (for data consistency).

- UQ\_Email: UNIQUE (Email)

→ No two instructors can have the same email.

- CHK\_Department: NOT NULL

→ Every instructor must be assigned to a department.

4. Table Name: Timetable

- Description: Stores class schedules including day, start/end time, room, and associated subject and instructor.

- Attributes:

- TimetableID: INTEGER, PK, NOT NULL, UNIQUE

→ Unique identifier for each scheduled class.

- SubjectID: INTEGER, FK (REFERENCES Subjects), NOT NULL

→ Subject being taught in this session.

- TeacherID: INTEGER, FK (REFERENCES Teachers), NOT NULL

→ Instructor teaching this session.

- DayOfWeek: VARCHAR(10), NOT NULL

→ Day of the week (e.g., Monday, Tuesday).

→ Allowed values: Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday.

- StartTime: TIME, NOT NULL

→ Start time of the class.

- EndTime: TIME, NOT NULL

→ End time of the class.

- RoomNumber: VARCHAR(100), NOT NULL

→ Classroom or lab where the session is held.

- Constraints:

- PK\_Timetable: PRIMARY KEY (TimetableID)

→ Ensures each schedule entry is unique.

- FK\_Timetable\_Subject: FOREIGN KEY (SubjectID) REFERENCES Subjects(SubjectID)

→ Links to the subject being taught.

- FK\_Timetable\_Teacher: FOREIGN KEY (TeacherID) REFERENCES Teachers(TeacherID)

→ Links to the assigned instructor.

- CHK\_Times: CHECK (EndTime > StartTime)

→ Ensures end time is after start time.

- CHK\_Day: CHECK (DayOfWeek IN ('Monday','Tuesday','Wednesday','Thursday','Friday','Saturday','Sunday'))

→ Restricts to valid days of the week.

- UQ\_RoomTime: UNIQUE (RoomNumber, DayOfWeek, StartTime, EndTime)

→ Prevents double-booking of rooms.

5. Table Name: Enrollments

- Description: Resolves the many-to-many relationship between Students and Subjects. Tracks which students are enrolled in which subjects and their grades.

- Attributes:

- EnrollmentID: INTEGER, PK, NOT NULL, UNIQUE

→ Unique identifier for each enrollment record.

- StudentID: INTEGER, FK (REFERENCES Students), NOT NULL

→ Student who is enrolled.

- SubjectID: INTEGER, FK (REFERENCES Subjects), NOT NULL

→ Subject the student is enrolled in.

- EnrollmentDate: DATE, NOT NULL, DEFAULT CURRENT\_DATE

→ Date when enrollment was made.

- Grade: CHAR(2)

→ Final grade received (e.g., A, B+, F). Can be NULL before grading.

- **Constraints**:

- PK\_Enrollments: PRIMARY KEY (EnrollmentID)

→ Ensures each enrollment is uniquely identified.

- FK\_Enrollments\_Student: FOREIGN KEY (StudentID) REFERENCES Students(StudentID)

→ Enforces valid student.

- FK\_Enrollments\_Subject: FOREIGN KEY (SubjectID) REFERENCES Subjects(SubjectID)

→ Enforces valid subject.

- UQ\_Enrollment: UNIQUE (StudentID, SubjectID)

→ Prevents a student from enrolling in the same subject twice.

- CHK\_Grade: CHECK (Grade IN ('A', 'A-', 'B+', 'B', 'B-', 'C+', 'C', 'C-', 'D', 'F', 'I'))

→ Ensures only valid grades are entered.

Part 2: Relationships (Cardinality)

* Students → Enrollments | One-to-Many | One student can enroll in multiple subjects.
* Subjects → Enrollments | One-to-Many | One subject can have many students enrolled.
* Students ↔ Subjects | Many-to-Many | Implemented via the Enrollments table.
* Teachers → Subjects | One-to-Many | One instructor can teach multiple subjects.
* Subjects → Timetable | One-to-Many | One subject can have multiple scheduled sessions (e.g., lecture + lab)
* Teachers → Timetable | One-to-Many | One instructor can have multiple scheduled classes.

> Many-to-Many Relationship Implemented: Between **Students** and **Subjects** via the **Enrollments** table.

Normalization (3NF)

All tables are in Third Normal Form (3NF):

1. 1N: All attributes are atomic and contain single values. No repeating groups.

2. 2NF: All non-key attributes are fully functionally dependent on the entire primary key (no partial dependencies).

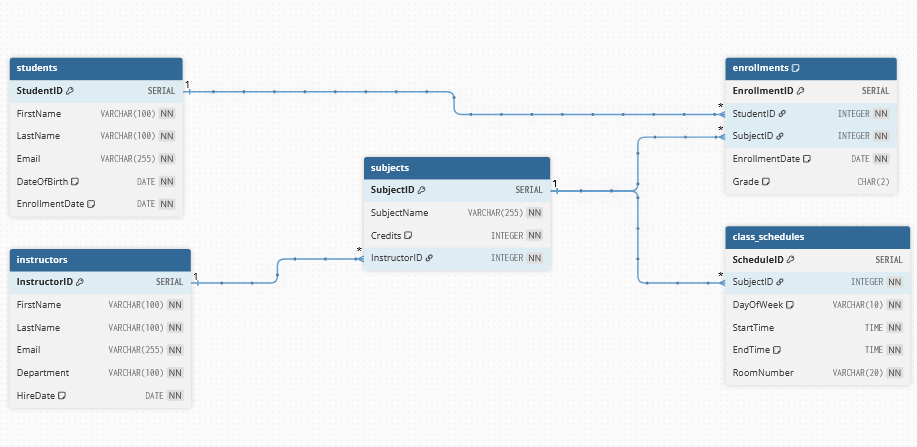
3. 3NF: No transitive dependencies. For example:

- In `Subjects`, `Department` depends on the subject, not on `TeacherID`.

- Instructor details are stored in the `Teachers` table, not duplicated in `Subjects` or `Timetable`

All tables meet 3NF requirements.

Part 3: ER Diagram (Physical Model Description)



- Students — linked to Enrollments via ‘StudentID’

- Subjects — linked to Enrollments, Timetable, and Teachers

- Teachers — linked to Subjects and Timetable

- Timetable — linked to Subjects and Teachers

- Enrollments — linked to Students and Subject