**1. Teachers Table**

CREATE TABLE Teachers (

teacher\_id INT PRIMARY KEY AUTO\_INCREMENT,

first\_name VARCHAR(50) NOT NULL,

last\_name VARCHAR(50) NOT NULL,

email VARCHAR(100) UNIQUE NOT NULL,

phone VARCHAR(15),

pass VARCHAR(255) NOT NULL

);

**Purpose**

Stores information about teachers who manage subjects and take attendance.

**Columns**

* **teacher\_id (INT, PRIMARY KEY, AUTO\_INCREMENT)**:
  + A unique identifier for each teacher, automatically generated (e.g., 1, 2, 3).
  + Used as a reference in other tables (Subjects, Attendance).
* **first\_name (VARCHAR(50), NOT NULL)**:
  + Teacher’s first name (e.g., "John"). Limited to 50 characters, required.
* **last\_name (VARCHAR(50), NOT NULL)**:
  + Teacher’s last name (e.g., "Doe"). Required field.
* **email (VARCHAR(100), UNIQUE, NOT NULL)**:
  + Teacher’s email (e.g., "[john.doe@example.com](mailto:john.doe@example.com)"). Unique to identify teachers for login; required.
* **phone (VARCHAR(15))**:
  + Teacher’s phone number (e.g., "+91-1234567890"). Optional (nullable), supports various formats.
* **pass (VARCHAR(255), NOT NULL)**:
  + Teacher’s password (hashed, e.g., using bcrypt). Required for authentication.

**Constraints**

* PRIMARY KEY (teacher\_id): Ensures every teacher has a unique ID.
* UNIQUE (email): Prevents duplicate email addresses.
* NOT NULL on key fields ensures essential data is always provided.

**Role in System**

* Teachers log in using email and pass.
* teacher\_id links to Subjects (who teaches what) and Attendance (who records it).

**2. Students Table**

CREATE TABLE Students (

student\_id INT PRIMARY KEY AUTO\_INCREMENT,

first\_name VARCHAR(50) NOT NULL,

last\_name VARCHAR(50) NOT NULL,

roll\_number VARCHAR(20) UNIQUE NOT NULL,

email VARCHAR(100) UNIQUE NOT NULL,

phone VARCHAR(15)

);

**Purpose**

Stores details of students whose attendance is tracked.

**Columns**

* **student\_id (INT, PRIMARY KEY, AUTO\_INCREMENT)**:
  + Unique ID for each student, auto-generated.
* **first\_name (VARCHAR(50), NOT NULL)**:
  + Student’s first name (e.g., "Alice").
* **last\_name (VARCHAR(50), NOT NULL)**:
  + Student’s last name (e.g., "Smith").
* **roll\_number (VARCHAR(20), UNIQUE, NOT NULL)**:
  + Unique identifier like "A001" or "2023CS001". Used for quick reference.
* **email (VARCHAR(100), UNIQUE, NOT NULL)**:
  + Student’s email (e.g., "[alice.smith@example.com](mailto:alice.smith@example.com)"). Unique and required.
* **phone (VARCHAR(15))**:
  + Optional phone number (e.g., "987-654-3210").

**Constraints**

* PRIMARY KEY (student\_id): Unique student identifier.
* UNIQUE (roll\_number): Ensures no duplicate roll numbers.
* UNIQUE (email): Prevents email reuse.

**Role in System**

* Links to Student\_Subject (enrollment) and Attendance (presence tracking).
* Teachers see students by roll\_number or name when marking attendance.

**3. Subjects Table**

CREATE TABLE Subjects (

subject\_id INT PRIMARY KEY AUTO\_INCREMENT,

subject\_name VARCHAR(100) NOT NULL,

teacher\_id INT,

FOREIGN KEY (teacher\_id) REFERENCES Teachers(teacher\_id) ON DELETE SET NULL

);

**Purpose**

Defines academic subjects and assigns them to teachers.

**Columns**

* **subject\_id (INT, PRIMARY KEY, AUTO\_INCREMENT)**:
  + Unique ID for each subject (e.g., 1 for "Math").
* **subject\_name (VARCHAR(100), NOT NULL)**:
  + Name of the subject (e.g., "Mathematics"). Required.
* **teacher\_id (INT, FOREIGN KEY)**:
  + References Teachers.teacher\_id to indicate who teaches it (e.g., Teacher 1).

**Constraints**

* PRIMARY KEY (subject\_id): Unique subject identifier.
* FOREIGN KEY (teacher\_id) REFERENCES Teachers(teacher\_id) ON DELETE SET NULL:
  + Links to a teacher. If the teacher is deleted, teacher\_id becomes NULL (subject remains unassigned).

**Role in System**

* Ties teachers to their subjects, ensuring they only manage attendance for their own classes.
* Links to Timetable for scheduling.

**4. Timetable Table**

CREATE TABLE Timetable (

timetable\_id INT PRIMARY KEY AUTO\_INCREMENT,

subject\_id INT,

day\_of\_week ENUM('Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday', 'Sunday') NOT NULL,

start\_time TIME NOT NULL,

end\_time TIME NOT NULL,

room VARCHAR(50),

semester\_start\_date DATE NOT NULL,

semester\_end\_date DATE NOT NULL,

UNIQUE (subject\_id, day\_of\_week, start\_time),

CHECK (start\_time < end\_time),

FOREIGN KEY (subject\_id) REFERENCES Subjects(subject\_id) ON DELETE CASCADE

);

**Purpose**

Defines the recurring weekly schedule for subjects.

**Columns**

* **timetable\_id (INT, PRIMARY KEY, AUTO\_INCREMENT)**:
  + Unique ID for each timetable entry.
* **subject\_id (INT, FOREIGN KEY)**:
  + Links to Subjects.subject\_id (e.g., Math).
* **day\_of\_week (ENUM, NOT NULL)**:
  + Day of the week (e.g., "Monday"). Restricted to valid days.
* **start\_time (TIME, NOT NULL)**:
  + Class start time (e.g., "09:00:00").
* **end\_time (TIME, NOT NULL)**:
  + Class end time (e.g., "10:00:00").
* **room (VARCHAR(50))**:
  + Optional classroom (e.g., "Room 101").
* **semester\_start\_date (DATE, NOT NULL)**:
  + Start of the active period (e.g., "2025-03-01").
* **semester\_end\_date (DATE, NOT NULL)**:
  + End of the active period (e.g., "2025-06-30").

**Constraints**

* UNIQUE (subject\_id, day\_of\_week, start\_time):
  + Prevents a subject from being scheduled twice at the same time on the same day.
* CHECK (start\_time < end\_time):
  + Ensures logical time ranges.
* FOREIGN KEY (subject\_id) REFERENCES Subjects(subject\_id) ON DELETE CASCADE:
  + If a subject is deleted, its timetable entries are removed.

**Role in System**

* Provides the recurring pattern for the calendar view (e.g., "Math every Monday, 9–10 AM").
* semester\_start\_date and semester\_end\_date define the period for session generation.

**5. Sessions Table**

CREATE TABLE Sessions (

session\_id INT PRIMARY KEY AUTO\_INCREMENT,

timetable\_id INT,

date DATE NOT NULL,

status ENUM('Scheduled', 'Completed', 'Cancelled') DEFAULT 'Scheduled',

UNIQUE (timetable\_id, date),

FOREIGN KEY (timetable\_id) REFERENCES Timetable(timetable\_id) ON DELETE CASCADE

);

**Purpose**

Tracks specific instances of classes based on the timetable.

**Columns**

* **session\_id (INT, PRIMARY KEY, AUTO\_INCREMENT)**:
  + Unique ID for each session.
* **timetable\_id (INT, FOREIGN KEY)**:
  + Links to Timetable.timetable\_id.
* **date (DATE, NOT NULL)**:
  + Specific date of the session (e.g., "2025-03-24").
* **status (ENUM, DEFAULT 'Scheduled')**:
  + 'Scheduled': Planned but not yet held.
  + 'Completed': Held and attendance taken.
  + 'Cancelled': Didn’t happen (e.g., holiday).

**Constraints**

* UNIQUE (timetable\_id, date):
  + Prevents duplicate sessions for the same timetable entry on the same date.
* FOREIGN KEY (timetable\_id) REFERENCES Timetable(timetable\_id) ON DELETE CASCADE:
  + Deletes sessions if the timetable entry is removed.

**Role in System**

* Populates the calendar with specific dates.
* status ensures only 'Completed' sessions count for attendance calculations.

**6. Attendance Table**

CREATE TABLE Attendance (

attendance\_id INT PRIMARY KEY AUTO\_INCREMENT,

student\_id INT,

session\_id INT,

status ENUM('Present', 'Absent', 'Late') DEFAULT NULL,

timestamp DATETIME DEFAULT CURRENT\_TIMESTAMP,

recorded\_by INT,

UNIQUE (student\_id, session\_id),

FOREIGN KEY (student\_id) REFERENCES Students(student\_id) ON DELETE CASCADE,

FOREIGN KEY (session\_id) REFERENCES Sessions(session\_id) ON DELETE CASCADE,

FOREIGN KEY (recorded\_by) REFERENCES Teachers(teacher\_id) ON DELETE SET NULL

);

**Purpose**

Records student attendance for each session.

**Columns**

* **attendance\_id (INT, PRIMARY KEY, AUTO\_INCREMENT)**:
  + Unique ID for each attendance record.
* **student\_id (INT, FOREIGN KEY)**:
  + Links to Students.student\_id.
* **session\_id (INT, FOREIGN KEY)**:
  + Links to Sessions.session\_id.
* **status (ENUM, DEFAULT NULL)**:
  + 'Present', 'Absent', 'Late', or NULL (not yet marked).
* **timestamp (DATETIME, DEFAULT CURRENT\_TIMESTAMP)**:
  + When the record was created/updated.
* **recorded\_by (INT, FOREIGN KEY)**:
  + Links to Teachers.teacher\_id (who marked it).

**Constraints**

* UNIQUE (student\_id, session\_id):
  + One attendance record per student per session.
* FOREIGN KEY constraints:
  + Deletes records if student or session is removed (ON DELETE CASCADE).
  + Sets recorded\_by to NULL if the teacher is deleted (ON DELETE SET NULL).

**Role in System**

* Teachers mark attendance for their students.
* NULL default simplifies marking; only conducted sessions ('Completed') matter for calculations.

**7. Student\_Subject Table**

CREATE TABLE Student\_Subject (

student\_id INT,

subject\_id INT,

PRIMARY KEY (student\_id, subject\_id),

FOREIGN KEY (student\_id) REFERENCES Students(student\_id) ON DELETE CASCADE,

FOREIGN KEY (subject\_id) REFERENCES Subjects(subject\_id) ON DELETE CASCADE

);

**Purpose**

Manages the many-to-many relationship between students and subjects (enrollment).

**Columns**

* **student\_id (INT, FOREIGN KEY)**:
  + Links to Students.student\_id.
* **subject\_id (INT, FOREIGN KEY)**:
  + Links to Subjects.subject\_id.

**Constraints**

* PRIMARY KEY (student\_id, subject\_id):
  + Ensures no duplicate enrollments.
* FOREIGN KEY with ON DELETE CASCADE:
  + Removes enrollments if a student or subject is deleted.

**Role in System**

* Filters students visible to a teacher (via Subjects.teacher\_id).
* Determines who gets pre-populated in Attendance.

**8. Calendar\_Exceptions Table (Optional)**

CREATE TABLE Calendar\_Exceptions (

exception\_id INT PRIMARY KEY AUTO\_INCREMENT,

date DATE NOT NULL,

description VARCHAR(100),

UNIQUE (date)

);

**Purpose**

Tracks holidays or other exceptions affecting session status.

**Columns**

* **exception\_id (INT, PRIMARY KEY, AUTO\_INCREMENT)**:
  + Unique ID for each exception.
* **date (DATE, NOT NULL)**:
  + Date of the exception (e.g., "2025-03-15").
* **description (VARCHAR(100))**:
  + Reason (e.g., "Spring Break").

**Constraints**

* UNIQUE (date):
  + Prevents duplicate exceptions on the same day.

**Role in System**

* Used to set Sessions.status to 'Cancelled' for holidays, excluding them from attendance totals.

**How It All Ties Together**

1. **Teacher Logs In**: Uses Teachers.email and pass.
2. **Sees Calendar**: Timetable (weekly schedule) and Sessions (specific dates) populate the view.
3. **Marks Attendance**:
   * Queries Student\_Subject and Subjects to list students for a session.
   * Updates Attendance.status for their students in a 'Completed' session.
4. **Holidays**: Calendar\_Exceptions marks sessions as 'Cancelled'.
5. **Attendance Calculation**: Counts 'Present' vs. total 'Completed' sessions.

**Final Notes**

This schema is **robust and complete** for your needs:

* **Scalable**: Handles multiple teachers, students, and subjects.
* **Flexible**: Supports calendar views and holiday adjustments.
* **Secure**: Enforces teacher-specific access via relationships.