**Applied Database Technology**

**Final Project Part - 2**

**Streamlining School Operations: A Cutting-Edge School Management Platform - Database Design**

**GROUP : 23**

**Tumul Rajvedi**

MS in Data Science

[trajvedi@iu.edu](mailto:trajvedi@iu.edu)

**Sumit Dighe**

MS in Data Science

[sdighe@iu.edu](mailto:sdighe@iu.edu)

**Vedavyas Chakkirala**

MS in Data Science

[vchakkir@iu.edu](mailto:vchakkir@iu.edu)

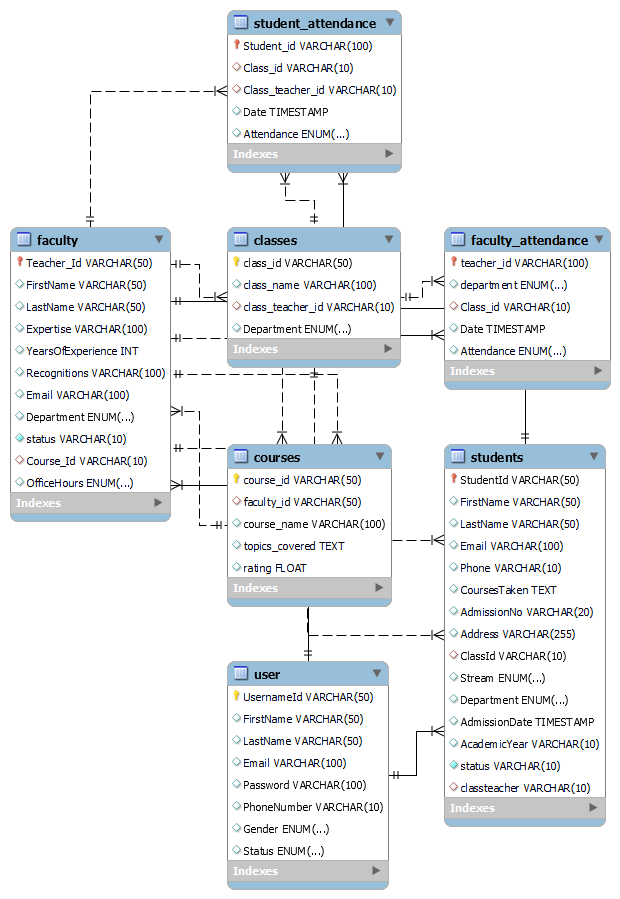
**Database Schema**:

The architectural design of our database emphasizes a meticulous approach to managing school operations, enclosing course scheduling, faculty assignments, student enrollments, and attendance tracking. It displays a well-organized and systematically structured data environment that is essential for the efficient administration of educational activities and resources.

Here, we used the EER diagram feature of MySQL workbench to create the database schema.

**Diagram Structure and Database Constraints:**

To generate an Enhanced Entity-Relationship (EER) diagram for the given data, we must illustrate the interconnections among the entities (or tables) and their respective attributes. The EER diagram serves as a visual representation of the database schema, showcasing how different tables relate to one another and the characteristics they possess. In the provided EER diagram, each entity is represented by a box, and its attributes are listed within the box. Relationships between entities are depicted by lines connecting them, indicating how they are related to each other. For example, a line connecting two entities indicates a relationship between them, and the type of relationship is specified (e.g., one-to-many, many-to-many).

**Below is an EER diagram representing the schema:**

The Database diagram displays 7 entities. Each entity and its attributes are described in detail below.

1. **User:**

This entity maintains a record of individuals who have access to the system, whether they are students, faculty, administrative staff, or other users. This entity has 8 attributes.

Number of Foreign keys: 0 (No foreign key constraint)

* **UsernameId**: This attribute is the *primary key*, as it uniquely identifies each user within the system. This is crucial for authentication and user management processes.
* **FirstName**, **LastName**, **Email**, **Password**, **PhoneNumber**: These attributes constitute the fundamental personal information necessary for user identification, communication, and account security.
* **Gender**, **Status**: These attributes capture the demographic detail and the current status(Teacher, student.) of the user. Login privileges will be exclusively granted to faculty members and those with administrative roles.

| **Attribute** | **Data Type** | **Constraint** | **Constraint Description** |
| --- | --- | --- | --- |
| UsernameId | VARCHAR(50) | PRIMARY KEY,  Not Null | Ensures that each UsernameId is unique  across the table, meaning no two users can have the same identifier. |
| FirstName ,  LastName | VARCHAR(50) |  |  |
| Email | VARCHAR(100) | CHECK (Email  LIKE '%@%'),  Not Null | Validates that the Email field contains an '@' character. This helps in ensuring that the email addresses stored in the database are in a proper format, which is important for communication and authentication purposes. It reduces the chances of data entry errors and ensures that the institution can reliably contact its faculty members. |
| Password | VARCHAR(100) | Not Null |  |
| PhoneNumber | VARCHAR(10) | CHECK  (CHAR\_LENGTH  (PhoneNumber) =  10), | Ensures that the phone number contains  exactly 10 characters, assuming a specific  format for phone numbers.  This ensures that all phone numbers are  stored in a standardized manner. |
| Gender | ENUM | ENUM | Limits the values of the Gender column to  the specified list ('male', 'female').  This standardizes gender data within the  system, reducing complexity and avoiding  data inconsistencies. |
| Status | ENUM | ENUM  Not Null | Limits the values of the Status column to the  specified list ('Teacher', 'Student').  Clarifies the role of the user within the system,  which is essential for managing access,  permissions, and functionalities based on user  roles. |

**2. Faculty:**

This entity catalogs the faculty details of the school faculty members. It has 11 attributes.

Number of Foreign keys : 1 (Course\_Id)

* **Teacher\_Id:** This is the *primary key* as it uniquely distinguishes each faculty member within the database.
* **FirstName**, **LastName**: These 2 attributes constitute the full name of the faculty member.
* **Expertise**: This attribute denotes the specific field or domain of knowledge where the faculty member specializes.
* **YearsOfExperience**: This attribute specifies the level of experience of the faculty member.
* **Recognitions**: This entity captures awards and recognitions received by the faculty highlighting notable achievements in their career..
* **Email**: This entity stores the Email address, for official communication with the faculty.
* **Department**: This attribute indicates the department to which the faculty belongs.
* **Status**: This attribute reflects the current employment condition, such as active, retired, or on leave.
* **Course\_Id**: This attribute establishes a relational link to the specific courses that the faculty member instructs. This is a *foreign key* referencing the “course\_id” attribute of the “courses” table.
* **OfficeHours**: This attribute specifies the designated time periods when the faculty member is available for student consultation.

| **Attribute** | **Data Type** | **Constraint** | **Constraint Description** |
| --- | --- | --- | --- |
| Teacher\_Id | VARCHAR(50) | Primary Key  Not Null | Ensures that each Teacher\_Id is unique across the faculty table.  This guarantees that each faculty member can be uniquely identified, which is crucial for referencing and managing faculty data throughout the database. It establishes a reliable link with other related data entities, like courses or classes. |
| FirstName,  LastName | VARCHAR(50) |  |  |
| Email | VARCHAR(100) | CHECK (Email LIKE '%@%')  Not Null | Validates that the Email field contains an '@' character.  This helps in ensuring that the email addresses stored in the database are in a proper format, which is important for communication and authentication purposes. It reduces the chances of data entry errors and ensures that the institution can reliably contact its faculty members. |
| Department | ENUM | ENUM | Limits the values of the Department column to the specified list ('Computer Science', 'Data Science', 'Statistics', 'Chemistry'). |
| status | VARCHAR(10) | Default Teacher |  |
| Course\_Id | VARCHAR(10) | Foreign Key,  ON DELETE CASCADE | Establishes a relationship between the Teacher\_Id in the faculty table and the UsernameId in the User table, with the ON DELETE CASCADE action.  This ensures referential integrity, linking each faculty member to a corresponding user account. The ON DELETE CASCADE action further means that if a user record is deleted, the corresponding faculty record will also be automatically deleted, which helps maintain data consistency and prevents orphan records in the faculty table. |
| Expertise | VARCHAR(100) |  |  |
| YearsOfExperie-nce | INT |  |  |
| Recognitions | VARCHAR(100) |  |  |
| OfficeHours | ENUM | ENUM | ENUM('Mon-Fri: 9am-5pm', 'Thu-Sat: 9am-5pm', 'Mon-Wed: 8am-4pm', 'Tue-Sat: 10am-6pm')  . Restricts the OfficeHours to one of the specified string values. |
| teacher\_id,  Class\_id,  Date | VARCHAR(100)  VARCHAR(10),  TIMESTAMP | Primary Key | teacher\_id: This column identifies the faculty member whose attendance is being recorded. It links the attendance record to a specific faculty member in the faculty table.  Class\_id: This identifies the class for which the attendance is being recorded. It connects the attendance record to a specific class in the classes table, indicating which class the faculty member was supposed to teach on the recorded date.  Date: This timestamp indicates the specific day for which the attendance record is applicable. It helps to determine when the attendance was taken.  Combined (teacher\_id, Class\_id, Date) ensure that each record in the Faculty\_attendance table is unique. This means that for any given teacher, class, and date combination, there can only be one attendance entry in the table. |
| Attendance | ENUM |  | Restricts the attendance status to one of the specified values, ('Present', 'Day-Off', 'On-Duty'). This standardizes the attendance records, making it clear whether faculty were present, on a day off, or on duty elsewhere. |
| department | ENUM | ENUM | Limits the values of the Department column to the specified list ('Computer Science', 'Data Science', 'Statistics', 'Chemistry'). |

**3. Students:**

This entity comprehensively records and allows to manage the personal and academic information of each student enrolled in The school. It is fundamental for tracking student progression. This entity has 14 attributes.

Number of Foreign keys : 3 (StudentId, ClassId, classteacher)

* **StudentId**: This attribute Uniquely identifies a student in the database. This is the *primary key*. It is also a *foreign key* referencing the “username\_ID” attribute of the “user” table.
* **FirstName**, **LastName**, **Email**, **Phone, Address**: These attributes record a student's essential personal details that help with identification and communication.
* **CoursesTaken**:This attribute lists the courses the student has enrolled in, helping in the monitoring of academic progress.
* **AdmissionNo**: This attribute is a unique number assigned to the student at the time of admission. It also serves as an additional identifier
* **ClassId**: This attribute is a *foreign key*, referencing the “class\_id” column in the “classes” table, linking the student to their current class.
* **Stream**, **Department**: These attributes indicate the academic stream and departmental affiliation of the student.
* **AdmissionDate**:This attribute stores the date on which the student was admitted.
* **AcademicYear**: This attribute specifies the academic year in which the student is enrolled,helps in tracking their academic status.
* **status**: This attribute represents the current enrollment status of the student, such as active, graduated, or withdrawn.
* **classteacher**: This attribute identifies the class teacher responsible for the particular student. It establishes a *foreign key* relationship with the “Teacher\_Id” column of the “faculty” table.

| **Attribute** | **Data Type** | **Constraint** | **Constraint Description** |
| --- | --- | --- | --- |
| StudentId | VARCHAR(50) | Primary Key  Not Null | It allows for the distinct identification of each student, which is essential for tracking and managing student records, linking with other academic data, and ensuring that each student's data remains unique and easily retrievable. |
| FirstName,  LastName,  Phone,  Address | VARCHAR(50),  VARCHAR(50),  VARCHAR(10),  VARCHAR(255) |  |  |
| Email | VARCHAR(100) | Not Null  CHECK (Email LIKE '%@%') | Validates that the Email field contains an '@' character.  This helps in ensuring that the email addresses stored in the database are in a proper format, which is important for communication and authentication purposes. It reduces the chances of data entry errors and ensures that the institution can reliably contact its faculty members. |
| CoursesTaken | TEXT |  |  |
| AdmissionNo | VARCHAR(20) |  | Limits the admission number to 20 characters. |
| ClassId | VARCHAR(10) | Foreign Key | This links to class\_id in the classes table. .It links each student to a specific class. |
| Stream | ENUM | ENUM | Limits the values of the Steam column to the 2  values specified in the list ('Graduate', 'Undergraduate'), |
| Department | ENUM | ENUM | Limits the values of the Department column to the specified list ('Computer Science', 'Data Science', 'Statistics', 'Chemistry') |
| AdmissionDate | TIMESTAMP |  | Provides a temporal reference for each student's  admission, which is crucial for historical  records and tracks the duration of a student's academic journey. |
| AcademicYear | VARCHAR(10) |  | Determines the current stage of the student's  education and for academic planning and  reporting purposes. |
| classteacher | VARCHAR(10) | Foreign Key | This links faculty\_id in the courses table to Teacher\_Id in the faculty table. |

**4. Courses:**

This entity defines the courses available and is structured with 5 attributes.

Number of Foreign keys : 1 (faculty\_id)

* **course\_id**: This attribute serves as the *primary key*, uniquely identifying each course within the database.
* **faculty\_id**: This attribute links each course to its corresponding instructor within the faculty. It establishes a *foreign key* relationship ,referencing the “Teacher\_Id” attribute in the “faculty” table.
* **course\_name**: This attribute lists the name of the course.
* **topics\_covered**: This attribute contains a description of the subject matter addressed in the course.
* **rating**: This attribute is a floating-point value representing the course rating.

| **Attribute** | **Data Type** | **Constraint** | **Constraint Description** |
| --- | --- | --- | --- |
| course\_id | VARCHAR(50) | Primary Key  Not Null | This ensures that each course\_id is unique across the database.  This is crucial for reliably identifying and referencing specific courses within the database and across related entities. |
| faculty\_id | VARCHAR(50) | Foreign Key  Not Null | This links faculty\_id in the courses table to Teacher\_Id in the faculty table.  This relationship is essential for maintaining referential integrity, ensuring that each course is associated with an existing faculty member. It prevents the assignment of courses to non-existent faculty members |
| course\_name | VARCHAR(100) |  |  |
| rating | FLOAT CHECK | Float Check | Ensures that the rating value for a course falls within the specified range of 0 to 5.  This is important for data validation, ensuring that all course ratings are within a realistic and standardized scale |

**5. Classes:**

This entity focuses on classes in the school and comprises 4 attributes.

Number of Foreign keys : 1 (class\_teacher\_id)

* **class\_id**: This is the unique identifier for a class. This is our *primary Key*.
* **class\_name**: This attribute denotes the designated name of the class.
* **class\_teacher\_id**: Identifier for the teacher of the class. This column references the “Teacher\_Id” attribute in the “faculty” table. This is the *foreign key* for “classes” entity
* **Department**: This attribute specifies the department affiliation of the class.

| **Attribute** | **Data Type** | **Constraint** | **Constraint Description** |
| --- | --- | --- | --- |
| class\_id | VARCHAR(50) | Primary Key  Not Null | This constraint guarantees that the class\_id  column cannot have any NULL values. Every  class record must have a valid identifier,  ensuring that each class can be uniquely and  reliably referenced within the database. |
| class\_teacher\_id | VARCHAR(10) | Foreign Key  Not Null | The class\_teacher\_id acts as a foreign key, creating a reference to the Teacher\_Id field in the faculty table. This foreign key constraint is essential for maintaining referential integrity between the two tables. It ensures that each class is associated with a valid faculty member who exists in the faculty table. |
| class\_name | VARCHAR(100) |  |  |
| Department | ENUM | ENUM | Limits the values of the Department column to the specified list ('Computer Science', 'Data Science', 'Statistics', 'Chemistry'). |

**6. Faculty\_Attendance:**

This entity helps to track and manage the attendance records of faculty members. This entity consists of 5 attributes.

Number of Foreign keys : 2 (teacher\_id, Class\_id )

* **teacher\_id**: This attribute serves as a *foreign key* as well as the *primary key*. It uniquely identifies the faculty member, referencing the “faculty” table on attribute “Teacher\_Id”.
* **Class\_id**: This attribute connects the attendance record to a specific class. It is also a *foreign key*, referencing attribute “class\_id” from “classes” table
* **department**: This attribute specifies the department to which the faculty member is affiliated, allowing departmental attendance tracking and analysis.
* **Date**: This attribute records the date of the attendance record.
* **Attendance**: This attribute denotes the attendance status, such as present, absent, or on official leave.

| **Attribute** | **Data Type** | **Constraint** | **Constraint Description** |
| --- | --- | --- | --- |
| teacher\_id,  Class\_id,  Date | VARCHAR(100)  VARCHAR(10),  TIMESTAMP | Primary Key  Not Null | teacher\_id: This column identifies the faculty member whose attendance is being recorded. It links the attendance record to a specific faculty member in the faculty table.  Class\_id: This identifies the class for which the attendance is being recorded. It connects the attendance record to a specific class in the classes table, indicating which class the faculty member was supposed to teach on the recorded date.  Date: This timestamp indicates the specific day for which the attendance record is applicable. It helps to determine when the attendance was taken.  Combined (teacher\_id, Class\_id, Date) ensure that each record in the Faculty\_attendance table is unique. This means that for any given teacher, class, and date combination, there can only be one attendance entry in the table. |
| Attendance | ENUM |  | Restricts the attendance status to one of the specified values, ('Present', 'Day-Off', 'On-Duty'). This standardizes the attendance records, making it clear whether faculty were present, on a day off, or on duty elsewhere. |
| department | ENUM | ENUM | Limits the values of the Department column to the specified list ('Computer Science', 'Data Science', 'Statistics', 'Chemistry'). |

**7. Student\_Attendance:**

This entity is crucial for tracking daily attendance,methodically documenting and analyzing the attendance records of each student. This entity comprises 5 attributes.

Number of Foreign keys : 3 (Student\_id, Class\_id, Class\_teacher\_id)

* **Student\_id**: This attribute uniquely identifies the student, serving as a *primary key.* It is also a *foreign key* referencing the “StudentId“ attribute of the “students” table. It links their attendance record to their personal and academic profile.
* **Class\_id**: This attribute represents a *foreign key* that connects the attendance entry to a specific class. It references “class\_id” column of the table “classes”
* **Class\_teacher\_id**: This attribute is also a *foreign key*, which identifies the teacher responsible for the class, referencing the “Teacher\_Id” column from the “faculty” table.
* **Date**: This attribute captures the specific date on which the attendance was recorded.
* **Attendance**: This attribute records the attendance status for the student on the specified date, such as present, absent, or excused.

| **Attribute** | **Data Type** | **Constraint** | **Constraint Description** |
| --- | --- | --- | --- |
| Student\_id, | VARCHAR(100) | Foreign Key,  Primary Key | Identifies the student whose attendance is  being recorded. This is a reference to the  student’s unique identifier in the students table. |
| Class\_id,    Date | VARCHAR(10),  TIMESTAMP | Foreign Key  Not Null | Class\_id: Specifies the class for which the  attendance is being recorded. This links to a  unique identifier in the classes table, showing  the specified class session the student attended  or missed.  Date: Records the specific day on which the  attendance status applies, providing a  temporal context to the attendance entry.  The composite key of (Student\_id, Class\_id,  Date) ensures that each record in the Student\_attendance table is unique. This  uniqueness means that for a given student, on  a specific date, there can only be one attendance  record for each class. It prevents the possibility  of duplicate attendance entries for the same  student in the same class on the same day,  thereby maintaining the integrity and accuracy  of the attendance data. |
| Class\_teacher\_id | VARCHAR(10) | Foreign Key | Links to the Teacher\_Id in the faculty table,  indicating the teacher of the class for which attendance is recorded.  This helps in correlating the attendance data  with the specific teacher, allowing analysis of  attendance patterns in relation to the class  teacher, and ensuring that attendance records  are consistent with the teaching assignments. |
| Attendance | ENUM | ENUM | Restricts the attendance status to one of the specified values, ('Present', 'Day-Off', 'On-Duty'). This standardizes the attendance records, making it clear whether a student was present, on a day off, or on duty elsewhere. |

**Relationships from Schema:**

* The **class\_teacher\_id** attribute in the *classes* table, which translates to the **Teacher\_Id** in the *faculty* database, is a crucial link between the Classes and Faculty entities. This defines the teaching responsibilities of faculty members in the academic setting by defining their accountability to particular classes.
* The **faculty\_id** in the *courses* entity corresponds to the **Teacher\_Id** in the *faculty* entity, establishing a systematic association between the Courses and Faculty entities. The relationship indicates which faculty members are responsible for teaching specific courses. This ensures that the curriculum is delivered in an orderly and systematic manner.
* The *faculty\_attendance* and *student\_attendance* entities are both connected to the *classes* entity through the **Class\_id** attribute, fostering a systematic record-keeping mechanism for attendance. The **teacher\_id** in *faculty\_attendance* links to **Teacher\_Id** in the *faculty* entity, while **Student\_id** in *student\_attendance* correlates with **StudentId** in the *students* entity. These relations make it possible for the school to monitor attendance trends and adherence to academic requirements through a thorough tracking system.
* The **ClassId** attribute, which denotes a student's enrollment in a particular class, links the *students* entity to the *classes* entity. This relationship supports each student's individual educational trajectory planning inside the institution in addition to helping with the administrative administration of class assignments.

**Group Contribution:**

| **Name** | **Tasks** | **Contribution** | **AVR Time Spent (hrs)** |
| --- | --- | --- | --- |
| **Tumul Rajvedi** | 1 - Database Design and Implementation.  2 - Frontend Development.  3 - Collaboration and Documentation. | 1.1 - Developed the Students table, outlining crucial columns to capture student-specific information effectively. Also, created the Student\_Attendance table, focusing on essential columns to accurately record and track student attendance.  1.2 - Populated the Students, Faculty\_Attendance , and Student\_Attendance tables with initial data entries, ensuring the foundational setup of the database.  1.3 - The Students table is linked to the Student\_Attendance table through the Student\_id field, which is a primary key in the Students entity and a foreign key in the Student\_Attendance, hence ensuring consistent attendance records for each student and maintaining referential integrity.  -----------------------------------------------  2.1 - Worked together to design and construct the homepage, establishing the website's primary layout and interface.  2.2 - Created the user login page, which includes safe authentication techniques to ensure system security.  2.3 - Developed the frontend framework, focusing on user experience and responsive design for managing a range of devices.  ----------------------------------------------  3.1 - Collaborated with the team to ensure seamless collaboration and timely project completion.  3.2 - Assembled thorough records of accomplished tasks, including database structure, API creation, and implementation details, all while maintaining alignment with established industry protocols.  3.3 - Improved the project documentation by communicating the deployed features and functions in a clear and concise manner. | 1.1 → 4 hrs  1.2 → 3 hrs  1.3 → 3 hrs  2 → 12 hrs  3 → 3 hrs  -----------------  Total → 25 hrs |
| **Sumit Dighe** | 1 - Database Design and Implementation.  2 - Frontend Development.  3 - Documentation. | 1.1 - Created and structured the User,Classes, and Faculty\_attendance tables in the database. Defined and implemented the schema for these tables, including column definitions and data types.  1.2 - Populated the User and Classes tables with initial data by inserting relevant records.  1.3 - Established a parent-child relationship between the User and faculty\_Attendance tables through appropriate primary and foreign key constraints, ensuring data integrity and enforcing referential relationships.  ---------------------------------------  2.1 -Designed and developed the user dashboard page, providing a centralized view of relevant information and features.  2.2 - Began the frontend implementation for CRUD operations, establishing interactive and user-friendly interfaces for creating, reading, updating, and deleting data related to students.  ---------------------------------------  3- Contributed to the project documentation by providing detailed descriptions of the tasks performed, including database design, frontend implementation, and design decisions. | 1.1 → 4 hrs  1.2 → 3 hrs  1.3 → 3 hrs  2 → 12 hrs  3 → 3 hrs  -----------------  Total → 25 hrs |
| **Vedavyas Chakkirala** | 1 - Database Design and Implementation.  2 - Backend API Development with Flask.  3 - Collaboration and Documentation. | 1.1 - Created the `Students` and `Student\_Attendance ` tables in the database to store relevant information about students and record attendance for each student.  1.2 - Implemented data entries for both `Faculty` and `Courses` tables, ensuring data integrity and consistency.  1.3 - Established a parent-child relationship between the `Faculty`, `Courses`, and `Classes` tables by assigning primary and foreign keys, facilitating proper data normalization and referential integrity.  ---------------------------------------  2.1 - Implemented APIs for user authentication, enabling secure access and authorization mechanisms.  2.2 - Designed and developed APIs for the dashboard page, providing a centralized view of relevant information and functionalities.  2.3 - Initiated the implementation of CRUD (Create, Read, Update, Delete) operations for managing data related to students, faculty members, classes, and courses.  -------------------------------------  3.1 - Collaborated effectively with team members, ensuring smooth coordination and efficient project progress.  3.2 - Documented the tasks performed, including database design, API development, and implementation details, adhering to industry-standard practices.  3.3 - Contributed to the project documentation, providing clear and concise descriptions of the implemented features and their functionalities. | 1.1 → 4 hrs  1.2 → 3 hrs  1.3 → 3 hrs  2 → 12 hrs  3 → 3 hrs  -----------------  Total → 25 hrs |

**SQL File:- Write code to create a database and build queries. Your task is to create a reproducible code.**

**-- 1)Tables creation:**

-- User - Sumit

-- Faculty - Veda

-- Students - Tumul

-- Courses - Veda

-- Classes - Sumit

-- Faculty\_Attendance - Sumit

-- Student\_Attendance - Tumul

**-- 2)Entries creation -**

-- (Sumit -[ User,Classes ],

-- Veda - [ Faculty, Courses ],

-- Tumul - [ Students, Faculty\_Attendance & Student\_Attendance ])

**-- 3) building relationship by defining primary and foreign keys -**

-- (Sumit -[ User,Faculty\_Attendance ],

-- Veda - [ Faculty, Courses, Classes ],

-- Tumul - [ Students, Student\_Attendance ])

create database dummy;

use dummy;

-- Created by Vedavyas, Sumit

CREATE TABLE IF NOT EXISTS User (

UsernameId VARCHAR(50) not null PRIMARY KEY,

FirstName VARCHAR(50),

LastName VARCHAR(50),

Email VARCHAR(100) CHECK (Email LIKE '%@%') not null,

Password VARCHAR(100) not null,

PhoneNumber VARCHAR(10) CHECK (CHAR\_LENGTH(PhoneNumber) = 10),

Gender ENUM('male', 'female'),

Status ENUM('Teacher', 'Student') not null

);

-- Created by Vedavyas

CREATE TABLE faculty (

Teacher\_Id VARCHAR(50) not null PRIMARY KEY,

FirstName VARCHAR(50),

LastName VARCHAR(50),

Course\_Id VARCHAR(10),

Expertise VARCHAR(100),

YearsOfExperience INT,

Recognitions VARCHAR(100),

Email VARCHAR(100) CHECK (Email LIKE '%@%') not null,

OfficeHours ENUM('Mon-Fri: 9am-5pm', 'Thu-Sat: 9am-5pm', 'Mon-Wed: 8am-4pm', 'Tue-Sat: 10am-6pm'),

FOREIGN KEY (Teacher\_Id) REFERENCES User(UsernameId) ON DELETE CASCADE

);

-- Add the department column to the faculty table

ALTER TABLE faculty

ADD COLUMN Department ENUM('Computer Science', 'Data Science', 'Statistics', 'Chemistry');

ALTER TABLE faculty

ADD COLUMN status VARCHAR(10) NOT NULL DEFAULT 'teacher';

-- Created by Tumul

CREATE TABLE students (

StudentId VARCHAR(50) not null PRIMARY KEY,

FirstName VARCHAR(50),

LastName VARCHAR(50),

Email VARCHAR(100) CHECK (Email LIKE '%@%') not null,

Phone VARCHAR(10),

CoursesTaken TEXT,

AdmissionNo VARCHAR(20),

Address VARCHAR(255),

ClassId VARCHAR(10),

Stream ENUM('Graduate', 'Undergraduate'),

Department ENUM('Computer Science', 'Data Science', 'Statistics', 'Chemistry'),

AdmissionDate TIMESTAMP,

AcademicYear VARCHAR(10)

);

ALTER TABLE students

ADD COLUMN status VARCHAR(10) NOT NULL DEFAULT 'student';

ALTER TABLE faculty

DROP COLUMN OfficeHours;

ALTER TABLE faculty

ADD COLUMN OfficeHours ENUM('Mon-Fri: 9am-5pm', 'Thu-Sat: 9am-5pm', 'Mon-Wed: 8am-4pm', 'Tue-Sat: 10am-6pm');

-- Adding classteacher column to students table

ALTER TABLE students

ADD COLUMN classteacher VARCHAR(10); -- Assuming Teacher\_Id is of type INT in the faculty table

-- Adding foreign key constraint for classteacher referencing Teacher\_Id in faculty table

ALTER TABLE students

ADD CONSTRAINT fk\_students\_faculty FOREIGN KEY (classteacher) REFERENCES faculty(Teacher\_Id);

-- Adding the foreign key constraint for students table referencing user table

ALTER TABLE students

ADD CONSTRAINT fk\_students\_user FOREIGN KEY (StudentId) REFERENCES user(usernameId);

-- -- Created by Vedavyas, Sumit

INSERT INTO User (UsernameId, FirstName, LastName, Email, Password, PhoneNumber, Gender, Status)

VALUES

('101\_Mark', 'Mark', 'Taylor', 'mark.taylor@example.com', 'password123', '1234567890', 'male', 'Teacher'),

('104\_Emily', 'Emily', 'Clark', 'emily.clark@example.com', 'password456', '9876543210', 'female', 'Teacher'),

('105\_David', 'David', 'Wilson', 'david.wilson@example.com', 'password789', '4567890123', 'male', 'Teacher'),

('106\_Sophia', 'Sophia', 'Brown', 'sophia.brown@example.com', 'passwordabc', '7890123456', 'female', 'Teacher'),

('1001\_Michael', 'Michael', 'Johnson', 'michael.johnson@example.com','password123', '1234567790', 'male','Student'),

('1002\_Emma', 'Emma', 'Wilson', 'emma.wilson@example.com','password456', '9876544210', 'female','Student'),

('1003\_Oliver', 'Oliver', 'Taylor', 'oliver.taylor@example.com','password789', '4566890123', 'male','Student'),

('1004\_Sophia', 'Sophia', 'Moore', 'sophia.moore@example.com','passwordabc', '7890123456', 'female','Student');

-- -- Created by Vedavyas

INSERT INTO faculty (Teacher\_Id, FirstName, LastName, Course\_Id, Expertise, YearsOfExperience, Recognitions, Email, OfficeHours, Department, Status)

VALUES

('101\_Mark', 'Mark', 'Taylor', 'CSE101', 'Computer Science', 4, 'Best Teacher Award 2020', 'mark.taylor@example.com',

'Thu-Sat: 9am-5pm', 'Computer Science', 'Teacher'),

('104\_Emily', 'Emily', 'Clark', 'DS201', 'Data Science', 3, 'Research Excellence Award', 'emily.clark@example.com',

'Mon-Wed: 8am-4pm', 'Data Science', 'Teacher'),

('105\_David', 'David', 'Wilson', 'STAT101', 'Statistics', 7, 'Outstanding Educator Award', 'david.wilson@example.com',

'Tue-Sat: 10am-6pm', 'Statistics', 'Teacher'),

('106\_Sophia', 'Sophia', 'Brown', 'CHEM101', 'Chemistry', 2, 'Innovative Teaching Award', 'sophia.brown@example.com',

'Mon-Fri: 9am-5pm', 'Chemistry', 'Teacher');

-- -- Created by Tumul

INSERT INTO students (StudentId, FirstName, LastName, Email, Phone, CoursesTaken, AdmissionNo, Address, ClassId,

Stream, Department, AdmissionDate, AcademicYear, classteacher, status)

VALUES

('1001\_Michael', 'Michael', 'Johnson', 'michael.johnson@example.com', '9876543210',

'CSE101, MATH101', 'ADM001', '123 Main St, City',

'CLASS\_101', 'Graduate', 'Computer Science', '2022-01-15', '2022-2023', '101\_Mark', 'Student'),

('1002\_Emma', 'Emma', 'Wilson', 'emma.wilson@example.com', '1234567890',

'DS201, STAT101', 'ADM002', '456 Elm St, Town',

'CLASS\_102', 'Undergraduate', 'Data Science', '2021-09-10', '2021-2022', '104\_Emily', 'Student'),

('1003\_Oliver', 'Oliver', 'Taylor', 'oliver.taylor@example.com', '7890123456',

'CHEM101, BIO101', 'ADM003', '789 Oak St, Village',

'CLASS\_103', 'Undergraduate', 'Chemistry', '2020-08-20', '2020-2021', '106\_Sophia', 'Student'),

('1004\_Sophia', 'Sophia', 'Moore', 'sophia.moore@example.com', '2345678901',

'STAT101, CSE101', 'ADM004', '345 Pine St, County',

'CLASS\_104', 'Graduate', 'Statistics', '2019-12-05', '2019-2020', '105\_David', 'Student');

-- Created by Vedavyas

CREATE TABLE IF NOT EXISTS courses (

course\_id VARCHAR(50) not null PRIMARY KEY,

faculty\_id VARCHAR(50) not null,

course\_name VARCHAR(100),

topics\_covered TEXT,

rating FLOAT CHECK (rating >= 0 AND rating <= 5),

FOREIGN KEY (faculty\_id) REFERENCES faculty(Teacher\_Id)

);

-- Add foreign key constraint for faculty\_id in courses table

ALTER TABLE courses

ADD CONSTRAINT fk\_courses\_faculty FOREIGN KEY (faculty\_id) REFERENCES faculty(Teacher\_Id);

-- Add foreign key constraint for course\_id in faculty table

ALTER TABLE faculty

ADD CONSTRAINT fk\_faculty\_courses FOREIGN KEY (Course\_Id) REFERENCES courses(course\_id);

-- Created by Vedavyas

INSERT INTO courses (course\_id, faculty\_id, course\_name, topics\_covered, rating)

VALUES

('STAT101', '105\_David', 'Statistics 101', 'Descriptive Statistics, Probability Distributions, Hypothesis Testing', 4.5),

('CSE101', '101\_Mark', 'Computer Science 101', 'Introduction to Programming, Data Structures, Algorithms', 4.8),

('DS201', '104\_Emily', 'Data Science 201', 'Data Preprocessing, Machine Learning Algorithms, Data Visualization', 4.6),

('CHEM101', '106\_Sophia', 'Chemistry 101', 'Atomic Structure, Chemical Reactions, Periodic Table', 4.3);

-- Created by Sumit

CREATE TABLE classes (

class\_id VARCHAR(50) not null PRIMARY KEY,

class\_name VARCHAR(100),

class\_teacher\_id VARCHAR(10) not null,

Department ENUM('Computer Science', 'Data Science', 'Statistics', 'Chemistry')

);

-- Adding foreign key constraint for class\_teacher\_id referencing Teacher\_Id in faculty table

ALTER TABLE classes

ADD CONSTRAINT fk\_classes\_faculty FOREIGN KEY (class\_teacher\_id) REFERENCES faculty(Teacher\_Id);

-- Adding the foreign key constraint for class\_id referencing class\_id in students table

ALTER TABLE students

ADD CONSTRAINT fk\_students\_classes FOREIGN KEY (ClassId) REFERENCES classes(class\_id);

-- -- Created by Sumit

INSERT INTO classes (class\_id, class\_name, class\_teacher\_id, Department)

VALUES

('CLASS\_101', 'Introduction to Computer Science', '101\_Mark', 'Computer Science'),

('CLASS\_102', 'Data Science Fundamentals', '104\_Emily', 'Data Science'),

('CLASS\_104', 'Statistics for Beginners', '105\_David', 'Statistics'),

('CLASS\_103', 'Chemistry Basics', '106\_Sophia', 'Chemistry');

-- -- Created by Sumit

CREATE TABLE Faculty\_attendance (

teacher\_id VARCHAR(100) not null,

department ENUM('Computer Science', 'Data Science', 'Statistics', 'Chemistry'),

Class\_id VARCHAR(10) not null,

Date TIMESTAMP,

Attendance ENUM('Present', 'Day-Off', 'On-Duty'),

PRIMARY KEY (teacher\_id),

FOREIGN KEY (teacher\_id) REFERENCES faculty(Teacher\_Id),

FOREIGN KEY (Class\_id) REFERENCES classes(class\_id)

);

-- -- Created by Sumit

INSERT INTO Faculty\_attendance (teacher\_id, department, Class\_id, Date, Attendance)

VALUES

('101\_Mark', 'Computer Science', 'CLASS\_101', '2024-03-06 09:00:00', 'Present'),

('104\_Emily', 'Data Science', 'CLASS\_102', '2024-03-06 09:00:00', 'Present'),

('106\_Sophia', 'Chemistry', 'CLASS\_103', '2024-03-06 09:00:00', 'Day-Off'),

('105\_David', 'Statistics', 'CLASS\_104', '2024-03-06 09:00:00', 'On-Duty');

-- Created by Tumul

CREATE TABLE Student\_attendance (

Student\_id VARCHAR(100) not null PRIMARY KEY,

Class\_id VARCHAR(10) not null,

Class\_teacher\_id VARCHAR(10) not null,

Date TIMESTAMP,

Attendance ENUM('Present', 'Absent', 'On-Duty'),

FOREIGN KEY (Student\_id) REFERENCES Students(StudentId),

FOREIGN KEY (Class\_id) REFERENCES Classes(class\_id),

FOREIGN KEY (Class\_teacher\_id) REFERENCES Faculty(Teacher\_Id)

);

--

-- -- Created by Tumul

INSERT INTO Student\_attendance (Student\_id, Class\_id, Class\_teacher\_id, Date, Attendance)

VALUES

('1001\_Michael', 'CLASS\_101', '101\_Mark', '2024-03-15 09:00:00', 'Present'),

('1004\_Sophia', 'CLASS\_104', '105\_David', '2024-03-15 10:00:00', 'Absent'),

('1003\_Oliver', 'CLASS\_103', '106\_Sophia', '2024-03-15 11:00:00', 'On-Duty'),

('1002\_Emma', 'CLASS\_102', '104\_Emily', '2024-03-15 12:00:00', 'Present');

-- ---------------------------------------------------------------

select \* from user;

select \* from faculty;

select \* from students;

select \* from courses;

select \* from classes;

select \* from faculty\_attendance;

select \* from student\_attendance;

**References:**

* [Flask Documentation](https://flask.palletsprojects.com/en/3.0.x/#api-reference)