NETAJI SUBHAS UNIVERSITY OF TECHNOLOGY

DBMS PROJECT

VIRTUAL LEARNING ENVIRONMENT

GROUP MEMBERS	ROLL NUMBER
RANJEET SINGH	2023UCS1588
SUBHAM	2023UCS1603
AMAN KUMAR	2023UCS1611

Requirement Analysis

- The **Virtual Learning Environment System (VLE)** is an online platform that facilitates learning by providing a centralized space for students and instructors to interact, share resources, manage learning materials, conduct assessments, and track academic progress.

- By incorporating user management, course creation, multimedia resources, assessment tools, and reporting features, **VLE** enhances the educational experience with flexibility and accessibility. It allows users to engage in interactive learning at their own pace, fosters effective communication through discussion forums and messaging, and supports personalized mentorship opportunities.

- While **VLE** offers numerous benefits such as improved engagement and efficient administration, it also faces challenges like user adoption, technical issues, and ensuring data security. Overall, VLES serves as a crucial tool in modern education, accommodating diverse learning needs and preferences in various educational settings.

- The **Course Materials Table** is designed to store and manage the learning resources uploaded by instructors for each course. It links the materials to specific courses and provides essential metadata like the file name, upload date, and description.

- The **Login Table** structure outlines the key fields required for basic user authentication and authorization in a Virtual Learning Environment (VLE) system. The table includes attributes such as ID, name, email, password, and role.

- The **Student Profile Table** stores detailed information about students enrolled in the Virtual Learning Environment (VLE). It includes personal data, academic information, and any other relevant details that can help in managing and tracking student progress.

- The **Class Schedules Table** is essential for organizing and managing class timings, sessions, and schedules in the Virtual Learning Environment (VLE). It helps in tracking when and where each course session is held, and which instructor is responsible.

- The **Assignment Table** is designed to manage the assignments given to students as part of their courses in the Virtual Learning Environment (VLE). It helps in tracking the details of assignments, including deadlines, instructions, and their relationship to courses and instructors.

- The **Teacher Profile Table** stores detailed information about teachers or instructors who are part of the Virtual Learning Environment (VLE). It includes both personal and professional information needed to manage their accounts, track their courses, and provide insight into their expertise.

- The **Enrollments Table** is critical for managing the relationship between students and courses within the Virtual Learning Environment (VLE). It tracks which students are enrolled in which courses, along with information such as enrollment date and course status.

- The **Courses Table** is designed to store and manage information about the courses offered within the Virtual Learning Environment (VLE). It captures essential course details, including the course title, description, assigned instructor, and course duration.

- The **Mentor Profile Table** stores detailed information about mentors who guide students through various courses, provide academic advice, and help with professional development within the Virtual Learning Environment (VLE). Mentors may have different roles from instructors, focusing more on guidance and support rather than teaching specific courses.
- The **Mentors Table** is designed to store information about mentors who provide guidance, advice, and support to students in the Virtual Learning Environment (VLE). The table tracks the mentor’s identity, expertise, and status, ensuring that the system can match students with appropriate mentors based on their needs.

- The **Mentor Assignment Table** is designed to record and manage the assignment of mentors to students. It tracks which mentor is guiding which student and includes information about the duration of the mentorship, ensuring proper matching and accountability.

- The **Mentor Schedules Table** is designed to store the availability and scheduled sessions of mentors within the Virtual Learning Environment (VLE). It tracks when mentors are available for appointments or meetings with students, helping students book time with their assigned mentors.

- The **Mentor Meeting Schedules Table** is specifically designed to manage and track scheduled meetings between mentors and students within the Virtual Learning Environment (VLE). This table helps in organizing appointments, ensuring that both mentors and students can effectively manage their time and commitments.

RELATIONAL SCHEMA

1. CourseMaterials Table

- **Relational Schema:** ◦ CourseMaterials(Material_ID, Course_ID, File_Name, File_Path,

  Upload_Date, Description)

  ◦ **Primary Key:** Material_ID

- **Decomposition:**

  ◦ CourseMaterials_1(Material_ID, Course_ID, File_Name)

  ▪ **Primary Key:** Material_ID

  ◦ CourseMaterials_2(File_Name, File_Path, Upload_Date, Description)

  ▪ **Primary Key:** File_Name

2. Login Table

- **Relational Schema:**

  ◦ Login(ID, Name, Email, Password, Role)

  ◦ **Primary Key:** ID
  ◦ **Alternate Key:** Email

3. StudentProfile Table

- **Relational Schema:**

  ◦ StudentProfile(ID, First_Name, Last_Name, Email, Phone_Number, DOB, Education_Level)

  ◦ **Primary Key:** ID

- **Alternate Keys:** Email, Phone_Number

4. ClassSchedules Table

- **Relational Schema:**
  - ClassSchedules(Schedule_ID, Course_ID, Class_Date, Duration_Minutes, Description, Meeting_Link)
  - **Primary Key:** Schedule_ID
  - **Candidate Key:** {Course_ID, Class_Date}

5. Assignments Table

- **Relational Schema:** ◦ Assignments(Assignment_ID, Course_ID, Title, Description, Due_Date, Max_Score)
  - **Primary Key:** Assignment_ID
- **Decomposition:** ◦ Assignment_1(Assignment_ID, Course_ID, Title)
    - **Primary Key:** Assignment_ID
  - Assignment_2(Course_ID, Description, Due_Date, Max_Score)
    - **Primary Key:** Course_ID

6. TeacherProfile Table

- **Relational Schema:**
  - TeacherProfile(ID, Name, Email, Password, PhoneNumber, Gender, DateOfBirth, Address, SubjectsTaught, Qualification, YearsOfExperience, Bio, ProfilePicture)
  - **Primary Key:** ID
  - **Alternate Key:** Email, PhoneNumber

7. Enrollments Table

- **Relational Schema:**
  - Enrollments(Enrollment_ID, Student_ID, Course_ID, Enrollment_Date, Status, Last_Accessed)
  - **Primary Key:** Enrollment_ID
  - **Candidate Key:** {Student_ID, Course_ID}

8. Courses Table

- **Relational Schema:**
  - Courses(Course_ID, Course_Title, Description, Category, Duration_Hours, Start_Date, End_Date, Teacher_ID, Teacher_Email, Creation_Date)
  - **Primary Key:** Course_ID
  - **Alternate Key:** Teacher_Email

9. MentorProfile Table

- **Relational Schema:**
  - MentorProfile(MentorID, FullName, Email, Expertise, Bio, ExperienceYears, Education_Level, Rate, CreatedAt, UpdatedAt)
  - **Primary Key:** MentorID
  - **Candidate Key:** {FullName, Email}

10. Mentors Table

- **Relational Schema:** ◦ Mentors(id, email, full_name, expertise, experience_years, rate, bio, linkedin, github, created_at, updated_at)
  - **Primary Key:** id ◦ **Alternate Key:** email
- **Decomposition:**

- Mentors_1(id, email)
    - **Primary Key:** id
- Mentors_2(email, full_name, expertise, experience_years, rate, bio, linkedin, github, created_at, updated_at)
    - **Primary Key:** email

11. MentorsAssignment Table

- **Relational Schema:**
    - MentorsAssignment(id, student_email, mentor_email, assignment_date)
    - **Primary Key:** id

12. MentorSchedule Table

- **Relational Schema:**
    - MentorSchedule(Schedule_ID, Mentor_Email, Student_Email, Class_Date, Duration_Minutes, Description, Meeting_Link, Created_At)
    - **Primary Key:** Schedule_ID

13. ScheduledMeetings Table

- **Relational Schema:**
    - ScheduledMeetings(id, mentor_email, student_email, meeting_date, meeting_link, created_at)
    - **Primary Key:** id
    - **Candidate Key:** {mentor_email, student_email, meeting_date}

RELATIONSHIPS

1. Login(ID) ↔ StudentProfile(ID): One-to-One (HasProfile)

2. Login(ID) ↔ TeacherProfile(ID): One-to-One (HasProfile)

3. TeacherProfile(ID) ↔ Courses(Teacher_ID): One-to-Many (Teaches)

4. StudentProfile(ID) ↔ Enrollments(Student_ID): One-to-Many (EnrolledIn)

5. Courses(Course_ID) ↔ ClassSchedules(Course_ID): One-to-Many (HasSchedules)

6. Courses(Course_ID) ↔ Assignments(Course_ID): One-to-Many (HasAssignments)

7. MentorProfile(MentorID) ↔ MentorsAssignment(mentor_email): One-to-Many (AssignsMentors)

8. StudentProfile(Email) ↔ MentorsAssignment(student_email): One-to-Many (AssignedMentor)

9. MentorProfile(MentorID) ↔ MentorSchedule(Mentor_Email): One-to-Many (HasSchedule)

10. StudentProfile(Email) ↔ MentorSchedule(Student_Email): One-to-Many (HasMentorMeetings)

11. MentorProfile(MentorID) ↔ ScheduledMeetings(mentor_email): One-to-Many (HasMeetings)

12. StudentProfile(Email) ↔ ScheduledMeetings(student_email): One-to-Many (HasScheduledMeetings)

13. Employees(empId) ↔ Contact_person(empId): One-to-One (HasContact)

RELATIONSHIPS

(HasLogin) StudentProfile ↔ Login

- **Type**: One-to-One relationship
- **Details**:
  - Email in StudentProfile is a foreign key that references Email in the Login table.
  - Each student profile is linked to exactly one login account.

(HasLogin) TeacherProfile ↔ Login

- **Type**: One-to-One relationship
- **Details**:
  - Email in TeacherProfile is a foreign key that references Email in the Login table.
  - Each teacher profile is linked to exactly one login account.

(HasTeacher) Courses ↔ TeacherProfile

- **Type**: Many-to-One relationship
- **Details**:
  - Teacher_Email in Courses is a foreign key that references Email in the TeacherProfile table.
  - One teacher can teach multiple courses, but each course is assigned to only one teacher.

(HasStudentEnrollment) Enrollments ↔ StudentProfile

- **Type**: Many-to-One relationship
- **Details**:
  - Student_ID in Enrollments is a foreign key that references ID in the StudentProfile table. ◦ A student can enroll in many courses, but each enrollment record corresponds to a single student.

(HasCourseEnrollment) Enrollments ↔ Courses

- **Type**: Many-to-One relationship

- **Details**:
  - Course_ID in Enrollments is a foreign key that references Course_ID in the Courses table. ◦ A course can have many students enrolled, but each enrollment corresponds to a single course.

(HasSchedule) ClassSchedules ↔ Courses

- **Type**: Many-to-One relationship
- **Details**:
  - Course_ID in ClassSchedules is a foreign key that references Course_ID in the Courses table.
  - Each course can have multiple scheduled classes, but each class schedule is linked to a specific course.

(HasMaterials) CourseMaterials ↔ Courses

- **Type**: Many-to-One relationship
- **Details**:
  - Course_ID in CourseMaterials is a foreign key that references Course_ID in the Courses table.
  - A course can have multiple materials, but each material is associated with a single course.

(HasAssignments) Assignments ↔ Courses

- **Type**: Many-to-One relationship
- **Details**:
  - Course_ID in Assignments is a foreign key that references Course_ID in the Courses table. ◦ Each course can have multiple assignments, but each assignment belongs to a single course.

(HasMentor) MentorProfile ↔ Login

- **Type**: One-to-One relationship
- **Details**:
  - Email in MentorProfile is a foreign key that references Email in the Login table.
  - Each mentor profile is linked to exactly one login account.

(MentorStudentAssignments) mentor_assignments ↔ MentorProfile & StudentProfile

- **Type**: Many-to-Many relationship
- **Details**:
  - mentor_email in mentor_assignments is a foreign key that references email in the MentorProfile table.
  - student_email in mentor_assignments is a foreign key that references Email in the StudentProfile table. ◦ One mentor can be assigned to multiple students, and one student can be assigned multiple mentors.

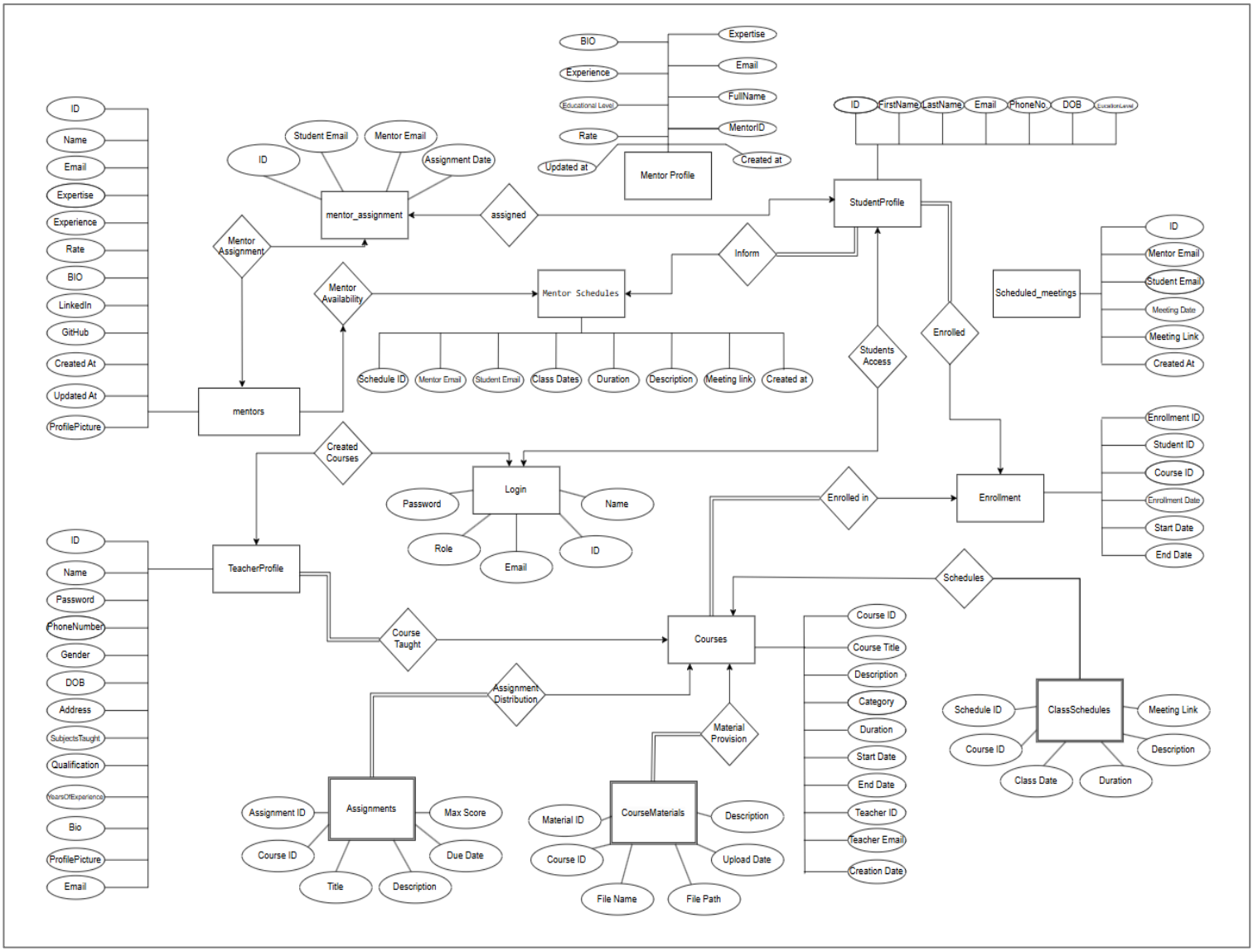
(HasMentorSchedule) mentor_schedules ↔ MentorProfile & StudentProfile

- **Type**: Many-to-Many relationship
- **Details**:
  - mentor_email in mentor_schedules is a foreign key that references email in the MentorProfile table.
  - student_email in mentor_schedules is a foreign key that references Email in the StudentProfile table.
  - Each mentor-student pair can have multiple scheduled classes.

(HasMeeting) scheduled_meetings ↔ MentorProfile & StudentProfile

- **Type**: Many-to-Many relationship
- **Details**:
  - mentor_email in scheduled_meetings is a foreign key that references email in the MentorProfile table.
  - student_email in scheduled_meetings is a foreign key that references Email in the StudentProfile table.
  - Each mentor and student pair can have multiple scheduled meetings.

ENTITY-RELATIONSHIP DIAGRAM



FUNCTIONAL DEPENDENCIES

1. CourseMaterials Table

Functional Dependencies:

- Material_ID → Course_ID, File_Name, File_Path, Upload_Date, Description
- File_Name → File_Path, Upload_Date, Description

BCNF Analysis:

- The dependency Material_ID → Course_ID, File_Name, File_Path, Upload_Date, Description is already in BCNF because Material_ID is a superkey.
- The dependency File_Name → File_Path, Upload_Date, Description is **not** in BCNF because File_Name is not a superkey.

Decomposition into BCNF:

- **First Relation:**
  Table1(Material_ID, Course_ID, File_Name)
  Key: Material_ID
- **Second Relation:**
  Table2(File_Name, File_Path, Upload_Date, Description) Key: File_Name

2. Login Table

Functional Dependencies:

- ID → Name, Email, Password, Role
- Email → ID, Name, Password, Role

BCNF Analysis:

- The table is in BCNF. Both ID → Name, Email, Password, Role and Email → ID, Name, Password, Role hold because both ID and Email are superkeys.

3. Student Profile Table

Functional Dependencies:

- ID → First_Name, Last_Name, Email, Phone_Number, DOB, Education_Level
- Email → ID, First_Name, Last_Name, Phone_Number, DOB, Education_Level
- Phone_Number → ID, First_Name, Last_Name, Email, DOB, Education_Level

BCNF Analysis:

- The table is in BCNF because every functional dependency has a superkey (ID, Email, or Phone_Number) on the left-hand side. **4. Class Schedules Table**

Functional Dependencies:

- Schedule_ID → Course_ID, Class_Date, Duration_Minutes, Description, Meeting_Link
- Course_ID, Class_Date → Schedule_ID, Duration_Minutes, Description, Meeting_Link

BCNF Analysis:

- The dependency Schedule_ID → Course_ID, Class_Date, Duration_Minutes, Description, Meeting_Link holds because Schedule_ID is a superkey.
- The second dependency Course_ID, Class_Date → Schedule_ID, Duration_Minutes, Description, Meeting_Link may violate BCNF if a course can have multiple classes at the same time. However, if that's not the case, it is in BCNF.

5. Assignments Table

Functional Dependencies:

- Assignment_ID → Title, Description, Due_Date, Max_Score
- Course_ID → Title

BCNF Analysis:

- This table is **not** in BCNF due to the Course_ID → Title dependency. To normalize, decompose into:

- **Table 1:** Assignment (Assignment_ID, Course_ID, Title)
  - **Table 2:** Course (Course_ID, Description, Due_Date, Max_Score)

6. Teacher Profile Table

Functional Dependencies:

- ID → Name, Email, Password, PhoneNumber, Gender, DateOfBirth, Address, SubjectsTaught, Qualification, YearsOfExperience, Bio, ProfilePicture
- Email → ID, Name, Password, PhoneNumber, Gender, DateOfBirth, Address, SubjectsTaught, Qualification, YearsOfExperience, Bio, ProfilePicture
- PhoneNumber → ID, Name, Email, Password, Gender, DateOfBirth, Address, SubjectsTaught, Qualification, YearsOfExperience, Bio, ProfilePicture

BCNF Analysis:

- The table is already in BCNF because each functional dependency has a superkey on the left-hand side (ID, Email, PhoneNumber).

7. Enrollments Table

Functional Dependencies:

- Enrollment_ID → Student_ID, Course_ID, Enrollment_Date, Status, Last_Accessed
- Student_ID, Course_ID → Enrollment_Date, Status, Last_Accessed

BCNF Analysis:

- The table is in BCNF because both dependencies hold: Enrollment_ID is a superkey, and the combination of Student_ID, Course_ID acts as a candidate key.

8. Courses Table

Functional Dependencies:

- Course_ID → Course_Title, Description, Category, Duration_Hours, Start_Date, End_Date, Teacher_ID, Teacher_Email, Creation_Date
- Teacher_Email → Teacher_ID

BCNF Analysis:

- The table is already in BCNF. Both Course_ID and Teacher_Email are superkeys, ensuring no violations. **9. Mentor Profile Table**

Functional Dependencies:

- MentorID → FullName, Email, Expertise, Bio, ExperienceYears, Education_Level, Rate, CreatedAt, UpdatedAt
- FullName, Email → MentorID, Expertise, Bio, ExperienceYears, Education_Level, Rate, CreatedAt, UpdatedAt

BCNF Analysis:

- The table is in BCNF as both MentorID and the combination of FullName, Email are superkeys, ensuring compliance.

10. Mentors Table

Functional Dependencies:

- id → email, full_name, expertise, experience_years, rate, bio, linkedin, github, created_at, updated_at
- email → id, full_name, expertise, experience_years, rate, bio, linkedin, github, created_at, updated_at

BCNF Analysis:

- The table violates BCNF due to the dependency email → id. To achieve BCNF: ◦ **Table 1:** Mentors(id, email)
  ◦ **Table 2:** MentorDetails(email, full_name, expertise, experience_years, rate, bio, linkedin, github, created_at, updated_at) **11. Mentors Assignment Table**

Functional Dependencies:

- id → student_email, mentor_email, assignment_date

BCNF Analysis:

- The table is in BCNF because id is a superkey and determines all other attributes.

12. Mentor Schedule Table

Functional Dependencies:

- Schedule_ID → Mentor_Email, Student_Email, Class_Date, Duration_Minutes, Description, Meeting_Link, Created_At **BCNF Analysis:**

- The table is in BCNF because Schedule_ID is a superkey that determines all other attributes.

13. Scheduled Meetings Table

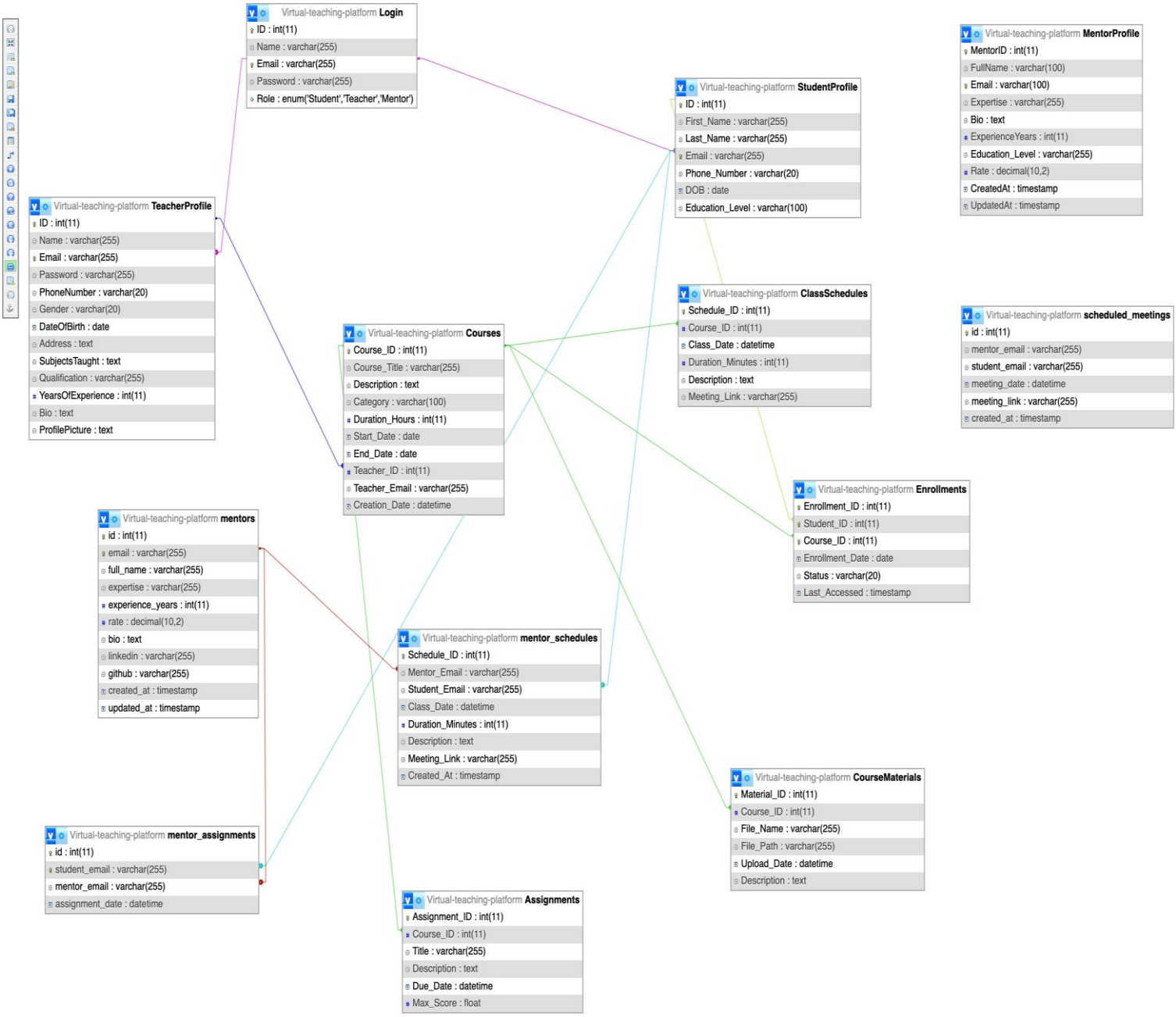
Functional Dependencies:

- id → mentor_email, student_email, meeting_date, meeting_link, created_at
- mentor_email, student_email, meeting_date → meeting_link, created_at

BCNF Analysis:

- The table is in BCNF because id is a superkey, and the combination of mentor_email, student_email, meeting_date acts as a candidate key.

CONNECTING DATA BASE USING XAMPP



Code for connecting to Database :-

```
const express = require("express");
const mysql = require("mysql"); const
cors = require("cors"); const multer =
require('multer'); const path =
require('path'); const fs = require('fs');

const app = express();

app.use(
  cors({      origin: "http://localhost:3000",      methods: ["GET", "POST", "PUT",
"DELETE"],        allowedHeaders:  ["Content-Type",  "Authorization",  "student-
email"],     credentials: true
  })

);
app.use(express.json()); app.use('/uploads',
express.static('uploads'));

const db = mysql.createConnection({
  host: "localhost",
  user: "root",   password: "",   database:
"Virtual-teaching-platform",   port: "3306",
});
```

```
db.connect((err) => {
  if (err) {
    console.error("Error connecting to database:", err);     return;
  }
  console.log("Connected to database successfully");
```

Creating Tables and Inserting Data

1. CourseMaterials Table

- (`CREATE TABLE IF NOT EXISTS CourseMaterials (

  Material_ID INT PRIMARY KEY AUTO_INCREMENT,

  Course_ID INT NOT NULL,

  File_Name VARCHAR(255) NOT NULL,

  File_Path VARCHAR(255) NOT NULL,

  Upload_Date DATETIME DEFAULT CURRENT_TIMESTAMP,

  Description TEXT,

  FOREIGN KEY (Course_ID) REFERENCES Courses(Course_ID))`);

- CourseMaterials (Course_ID, File_Name, File_Path, Upload_Date, Description)

- INSERT INTO materials (Material_ID, Course_ID, File_Name, File_Path, Upload_Date, Description)

  VALUES

  (5, 25, 'OS lab2.pdf', 'uploads/assignments/assignment-1729627549890-56619...', '2024-10-23 01:35:49', 'this is a test'),

  (6, 27, 'Subham_2023UCS1603_DBMS.pdf', 'uploads/assignments/ assignment-1729700655028-90418...', '2024-10-23 21:54:15', 'this is a electro magnetic wave assignment');

←T→	Material_ID	Course_ID	File_Name	File_Path	Upload_Date	Description
Edit Copy Delete	5	25	OS lab2.pdf	uploads/assignments/assignment-1729627549890-56619...	2024-10-23 01:35:49	this is a test
Edit Copy Delete	6	27	Subham_2023UCS1603_DBMS.pdf	uploads/assignments/assignment-1729700655028-90418...	2024-10-23 21:54:15	this is a electro magnetic wave assignment

2. Login Table

- CREATE TABLE login (

  ID INT PRIMARY KEY,

  Name VARCHAR(50),

Email VARCHAR(100),

Password VARCHAR(50),

Role VARCHAR(20));

- INSERT INTO Login(`Name`,`Email`,`Password`,`Role`)

  VALUES

←T→			ID	Name	Email	Password	Role
Edit	Copy	Delete	21	Shubham	subhamchauhan522@gmail.com	nnn	Teacher
Edit	Copy	Delete	22	Jishu	Jishu123@gmail.com	123	Student
Edit	Copy	Delete	23	Manit Singh	manit123@gmail.com	vvv	Mentor
Edit	Copy	Delete	24	alpha	alpha@gmail.com	nnn	Student
Edit	Copy	Delete	25	Ranjeet	ranjeetsingh17@gmail.com	Ranjeet	Student
Edit	Copy	Delete	27	Ranjeet	ranjeetsingh171@gmail.com	RanjeetSingh	Mentor
Edit	Copy	Delete	28	Ranjeet	ranjeetsingh107@gmail.com	mmm	Student

3. Student Profile Table

- CREATE TABLE student_profile (

  ID INT PRIMARY KEY,

  First_Name VARCHAR(50),

  Last_Name VARCHAR(50),

  Email VARCHAR(100),

  Phone_Number VARCHAR(15),

  DOB DATE,

Education_Level VARCHAR(50));

- INSERT INTO student_profile (ID, First_Name, Last_Name, Email, Phone_Number,
  DOB, Education_Level)

  VALUES (?, ?, ?, ?, ?, ?)

	ID	First_Name	Last_Name	Email	Phone_Number	DOB	Education_Level
Edit Copy Delete	10	JIshu	Chauhan	Jishu123@gmail.com	7253996990	2000-01-09	Undergraduate
Edit Copy Delete	11	alpha	khan	alpha@gmail.com	9213347890	2024-10-23	Undergraduate
Edit Copy Delete	12	Ranjeet	Singh	ranjeetsingh17@gmail.com	7253996990	2024-10-03	Undergraduate

4. ClassSchedules Table

- CREATE TABLE Class_Schedules (

  Schedule_ID INT PRIMARY KEY,

  Course_ID INT,

  Class_Date DATETIME,

  Duration_Minutes INT,

  Description VARCHAR(255),

  Meeting_Link VARCHAR(255));

- INSERT INTO ClassSchedules

  (Course_ID, Class_Date, Duration_Minutes, Description, Meeting_Link)

  VALUES (?, STR_TO_DATE(?, '%Y-%m-%d %H:%i:%s'), ?, ?, ?);

Extra options

	Schedule_ID	Course_ID	Class_Date	Duration_Minutes	Description	Meeting_Link
Edit Copy Delete	5	25	2024-10-23 21:35:00	60	this is my first class	https://meet.google.com/xji-kmtu-mju
Edit Copy Delete	6	26	2024-11-04 12:30:00	60	General Information about light	https://meet.google.com/fua-dwfy-xfw
Edit Copy Delete	7	25	2024-10-27 11:30:00	120	this is a checking class	https://meet.google.com/mdw-evva-xjt

5. Assignments Table

CREATE TABLE Assignments (

Assignment_ID INT PRIMARY KEY,

Course_ID INT,

Title VARCHAR(255),

-

Description TEXT,

Due_Date DATETIME,

Max_Score INT);

- INSERT INTO Assignments

  (Course_ID, Title, Description, Due_Date, Max_Score)

  VALUES (?, ?, ?, ?, ?)`;

Assignment_ID	Course_ID	Title	Description	Due_Date	Max_Score

6. Teacher Profile Table

- CREATE TABLE Teacher_Profiles (

  ID INT PRIMARY KEY,

  Name VARCHAR(100),

  Email VARCHAR(255),

  Password VARCHAR(100),

  PhoneNumber VARCHAR(15),

  Gender VARCHAR(10),

  DateOfBirth DATE,

  Address VARCHAR(255),

  SubjectsTaught VARCHAR(255),

  Qualification VARCHAR(100),

  YearsOfExperience INT,

  Bio TEXT,

  ProfilePicture VARCHAR(255));

- INSERT INTO TeacherProfile

  (Name, Email, Password, PhoneNumber, Gender, DateOfBirth, Address,

SubjectsTaught, Qualification, YearsOfExperience, Bio, ProfilePicture)
VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?) `;

←T→	ID	Name	Email	Password	PhoneNumber	Gender	DateOfBirth	Address	SubjectsTaught	Qualification	YearsOfExperience	Bio	ProfilePicture
Edit Copy Delete	12	Shubham	subhamchauhan522@gmail.com	nnn	7253994928	Male	2006-07-04	New Delhi	Maths , Science and CS	BTech	2	hello everyone	www.photo.com
Edit Copy Delete	13	Ranjeet	ranjeetsingh17@gmail.com	Ranjeet	93298374238	Male	2005-05-29	New Delhi	Science	BTech	10	I am a expert in teaching science to student by gi...	www.photo2.com

7. Enrollments Table

- CREATE TABLE Enrollments (

  Enrollment_ID INT PRIMARY KEY,

  Student_ID INT,

  Course_ID INT,

  Enrollment_Date DATE,

  Status VARCHAR(20),

  Last_Accessed DATETIME);

- INSERT INTO Enrollments (Student_ID, Course_ID, Enrollment_Date)

  VALUES (?, ?, ?);

←T→	Enrollment_ID	Student_ID	Course_ID	Enrollment_Date	Status	Last_Accessed
Edit Copy Delete	15	10	25	2024-10-21	Enrolled	2024-10-22 03:05:20
Edit Copy Delete	16	11	25	2024-10-22	Enrolled	2024-10-22 12:18:43
Edit Copy Delete	17	10	26	2024-10-22	Enrolled	2024-10-22 16:29:54
Edit Copy Delete	18	12	25	2024-10-22	Enrolled	2024-10-23 00:10:54
Edit Copy Delete	19	10	27	2024-10-23	Enrolled	2024-10-23 21:54:59

8. Courses Table

CREATE TABLE Courses (

Course_ID INT PRIMARY KEY,

Course_Title VARCHAR(100),

Description TEXT,

-

Category VARCHAR(50),

Duration_Hours INT,

Start_Date DATE,

End_Date DATE,

Teacher_ID INT,

Teacher_Email VARCHAR(100),

Creation_Date DATETIME);

- INSERT INTO Courses (

  Course_Title,

  Description,

  Category,

  Duration_Hours,

  Start_Date,

  End_Date,

  Teacher_Email)

  VALUES (?, ?, ?, ?, ?, ?, ?) ;

	Course_ID	Course_Title	Description	Category	Duration_Hours	Start_Date	End_Date	Teacher_ID	Teacher_Email	Creation_Date
Edit Copy Delete	25	web development	this is a web development course	Web Development	30	2024-10-25	2024-11-10	NULL	subhamchauhan522@gmail.com	2024-10-22 03:03:56
Edit Copy Delete	26	Ray Optics	This course will tell about the behavior of light ...	Science	20	2024-11-04	2024-11-30	NULL	ranjeetsingh17@gmail.com	2024-10-22 16:25:58
Edit Copy Delete	27	Machine Learning	this a machine laerning course	Other	50	2024-11-01	2024-11-30	NULL	subhamchauhan522@gmail.com	2024-10-23 21:42:56

9. Mentor Profile Table

- CREATE TABLE Mentors (

  MentorID INT PRIMARY KEY,

  FullName VARCHAR(100),

  Email VARCHAR(100),

  Expertise VARCHAR(100),

  Bio TEXT,

  ExperienceYears INT, Education_Level VARCHAR(50),

Rate DECIMAL(10, 2),

CreatedAt DATETIME,

UpdatedAt DATETIME);

- INSERT INTO MentorProfile (FullName, Email, Expertise, Bio, ExperienceYears, Education_Level, Rate)

  VALUES (?, ?, ?, ?, ?, ?, ?);

	MentorID	FullName	Email	Expertise	Bio	ExperienceYears	Education_Level	Rate	CreatedAt	UpdatedAt
Edit Copy Delete	3	Manit Singh	manit123@gmail.com	Data Science	Hello everyone	2	Undergraduate	2000.00	2024-10-22 03:09:14	2024-10-22 03:09:14
Edit Copy Delete	4	Ranjeet SIngh	ranjeetsingh17@gmail.com	Bakchodi	Hello everyone	5	Undergraduate	20.00	2024-10-22 16:34:56	2024-10-22 16:34:56
Edit Copy Delete	6	Ranjeet Singh	ranjeetsingh171@gmail.com	biology	Hello everyone	1	Undergraduate	20.00	2024-10-22 16:37:49	2024-10-22 20:17:23

10.Mentors Table

- CREATE TABLE mentors (id

  INT PRIMARY KEY, email

  VARCHAR(255),

  full_name VARCHAR(255),

  expertise VARCHAR(255),

  experience_years INT, rate

  DECIMAL(10, 2), bio

  TEXT, linkedin

  VARCHAR(255), github

  VARCHAR(255),

  created_at DATETIME,

  updated_at DATETIME);

  INSERT INTO Mentors (email, full_name, expertise, experience_years, rate, bio, linkedin, github)

  VALUES (?, ?, ?, ?, ?, ?, ?, ?)

  ON DUPLICATE KEY UPDATE

  full_name = ?,

-

expertise = ?,

experience_years = ?,

rate = ?, bio = ?,

linkedin = ?, github =

? `;

	id	email	full_name	expertise	experience_years	rate	bio	linkedin	github	created_at	updated_at
Edit Copy Delete	2	Manit123@gmail.com	Manit Singh	Data Science	3	50.00	hello everyone	https://www.linkedin.com/	https://github.com/	2024-10-22 03:10:32	2024-10-23 10:02:28
Edit Copy Delete	3	ranjeetsingh171@gmail.com	Ranjeet Singh	Web Development	1	500.00	hi everyone	https://www.linkedin.com/	https://github.com/	2024-10-22 20:21:41	2024-10-22 20:21:41

11.Mentor_assignments Table

- CREATE TABLE mentor_assignment (id INT PRIMARY

  KEY, student_email VARCHAR(255), mentor_email

  VARCHAR(255), assignment_date DATETIME);

- INSERT INTO mentor_assignments (student_email, mentor_email, assignment_date)

  VALUES (?, ?, NOW())

  ON DUPLICATE KEY UPDATE

  mentor_email = ?,

assignment_date = NOW()`;

	id	student_email	mentor_email	assignment_date
Edit Copy Delete	2	Jishu123@gmail.com	Manit123@gmail.com	2024-10-23 21:41:29

12.Mentor Schedule Table

- CREATE TABLE mentor_schedule (Schedule_ID INT

  PRIMARY KEY,

  Mentor_Email VARCHAR(255),

Student_Email VARCHAR(255),

Class_Date DATETIME,

Duration_Minutes INT,

Description TEXT,

Meeting_Link VARCHAR(255),

Created_At DATETIME);

- INSERT INTO mentor_schedules

  (mentor_email, student_email, class_date, duration_minutes, description, meeting_link)

  VALUES (?, ?, ?, ?, ?, ?)

Schedule_ID	Mentor_Email	Student_Email	Class_Date	Duration_Minutes	Description	Meeting_Link	Created_At

13.Scheduled_Meetings Table

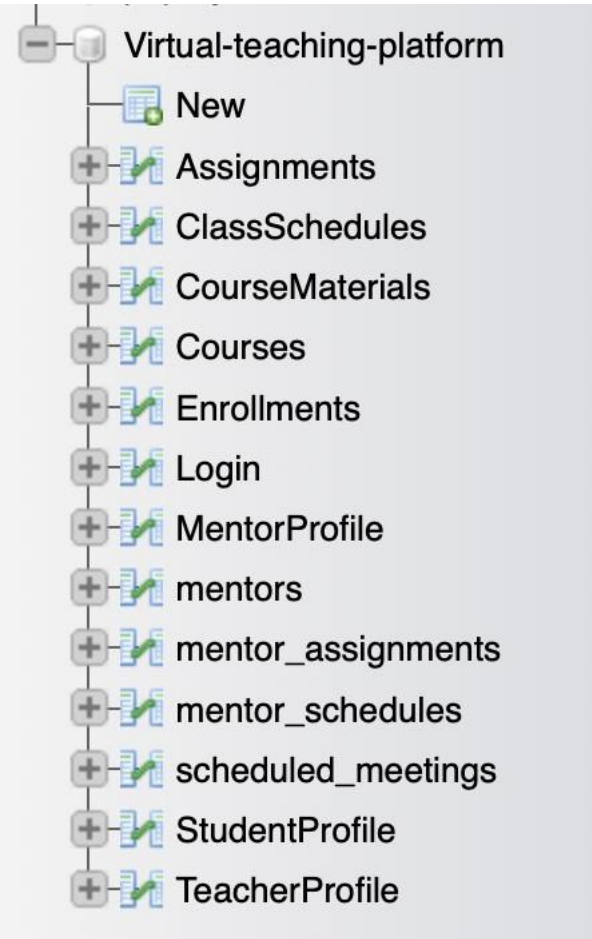
- CREATE TABLE scheduled_meetings(id INT PRIMARY KEY, mentor_email Varchar(100),student_email Varchar(100),meeting _date DATE, meeting_link VARCHAR(250),created_at DATE & TIME):
- INSERT INTO scheduled_meetings (mentor_email, student_email,meeting_date,meeting_link)

  VALUES (?, ?, ?, ?)`;

←T→	id	mentor_email	student_email	meeting_date	meeting_link	created_at
Edit Copy Delete	2	manit123@gmail.com	Jishu123@gmail.com	2024-10-23 03:11:00	https://meet.google.com/vvt-vkaq-ytw	2024-10-22 03:11:41

Software used- XAMPP

DATABASE : –


Virtual-teaching-platform
New
Assignments
ClassSchedules
CourseMaterials
Courses
Enrollments
Login
MentorProfile
mentors
mentor_assignments
mentor_schedules
scheduled_meetings
StudentProfile
TeacherProfile

QUERIES-

1) **Query to get all enrolled students for a specific course with their profiles (for example Course = 25)**

Code :-

```
SELECT
 s.First_Name,
 s.Last_Name,
s.Email,
s.Phone_Number,
 s.Education_Level,
e.Enrollment_Date,
 e.Status
 FROM StudentProfile s
JOIN Enrollments e ON s.ID = e.Student_ID
WHERE e.Course_ID = 25;
```

Output :-

First_Name	Last_Name	Email	Phone_Number	Education_Level	Enrollment_Date	Status
JIshu	Chauhan	Jishu123@gmail.com	7253996990	Undergraduate	2024-10-21	Enrolled
alpha	khan	alpha@gmail.com	9213347890	Undergraduate	2024-10-22	Enrolled
Ranjeet	Singh	ranjeetsingh17@gmail.com	7253996990	Undergraduate	2024-10-22	Enrolled

2) Query to get upcoming class schedules with course details

Code :-

```
SELECT
 c.Course_Title,
cs.Class_Date,
cs.Duration_Minutes,
cs.Description,
cs.Meeting_Link,
t.Name AS Teacher_Name
FROM ClassSchedules cs
JOIN Courses c ON cs.Course_ID = c.Course_ID
JOIN TeacherProfile t ON c.Teacher_Email = t.Email
 WHERE cs.Class_Date > NOW()
 ORDER BY cs.Class_Date ASC;
```

Output :-

Course_Title	Class_Date ▲ 1	Duration_Minutes	Description	Meeting_Link	Teacher_Name
web development	2024-10-27 11:30:00	120	this is a checking class	https://meet.google.com/mdw-evva-xjt	Shubham
Ray Optics	2024-11-04 12:30:00	60	General Information about light	https://meet.google.com/fua-dwfy-xfw	Ranjeet

3) Query to get All Students assigned to a Mentor with Assignment Date

Code :-

```
SELECT m.full_name AS mentor_name,
sp.First_Name,
```

```
 sp.Last_Name, sp.Email
AS student_email,
ma.assignment_date
FROM mentors m
 JOIN mentor_assignments ma ON m.email = ma.mentor_email
JOIN StudentProfile sp ON ma.student_email = sp.Email;
```

Output :-

mentor_name	First_Name	Last_Name	student_email	assignment_date
Manit Singh	JIshu	Chauhan	Jishu123@gmail.com	2024-10-23 21:41:29

4) Query to get Mentor's Full Profile with Contact Details

Code :-

```
SELECT
m.full_name,
 m.email, m.expertise,
m.experience_years,
m.rate, m.linkedin,
 m.github,
m.bio
FROM mentors m;
```

Output :-

←T→			full_name	email	expertise	experience_years	rate	linkedin	github	bio
Edit	Copy	Delete	Manit Singh	Manit123@gmail.com	Data Science	3	50.00	https://www.linkedin.com/	https://github.com/	hello everyone
Edit	Copy	Delete	Ranjeet Singh	ranjeetsingh171@gmail.com	Web Development	1	500.00	https://www.linkedin.com/	https://github.com/	hi everyone

5) Query to retrieve students who have joined course before a certain date

Code :-

```
SELECT
    s.ID,
    s.First_Name,
    s.Last_Name,
    s.Email,
    s.Phone_Number,
    s.Education_Level,
    c.Course_Title,
    e.Enrollment_Date
FROM
    StudentProfile s
JOIN
    Enrollments e ON s.ID = e.Student_ID
JOIN
    Courses c ON e.Course_ID = c.Course_ID
WHERE
    e.Enrollment_Date < '2024-10-24'
ORDER BY
    e.Enrollment_Date DESC;
```

Output :-

ID	First_Name	Last_Name	Email	Phone_Number	Education_Level	Course_Title	Enrollment_Date 1
10	JIshu	Chauhan	Jishu123@gmail.com	7253996990	Undergraduate	Machine Learning	2024-10-23
12	Ranjeet	Singh	ranjeetsingh17@gmail.com	7253996990	Undergraduate	web development	2024-10-22
10	JIshu	Chauhan	Jishu123@gmail.com	7253996990	Undergraduate	Ray Optics	2024-10-22
11	alpha	khan	alpha@gmail.com	9213347890	Undergraduate	web development	2024-10-22
10	JIshu	Chauhan	Jishu123@gmail.com	7253996990	Undergraduate	web development	2024-10-21

RELATIONAL ALGEBRA

1. π(Courses.Course_Title, TeacherProfile.Name) ⋈ (Courses.Teacher_Email = TeacherProfile.Email)

2.. π(mentor_assignments.mentor_email, mentor_assignments.student_email) ⋈ (mentor_assignments.student_email = StudentProfile.Email)

3. π(scheduled_meetings.meeting_date, mentors.full_name, StudentProfile.First_Name, StudentProfile.Last_Name) ⋈ (scheduled_meetings.mentor_email = mentors.email) ⋈ (scheduled_meetings.student_email = StudentProfile.Email)

4. π(mentors.full_name, mentors.expertise) (mentors)

5. π(StudentProfile.First_Name, StudentProfile.Last_Name, Enrollments.Course_ID) ⋈ (StudentProfile.ID = Enrollments.Student_ID)