DBMS Miniproject  
UE23CS351A

PEER TUTORING MATCHING SYSTEM

By:

Mahith K Das PES2UG23CS320

and

Maitreyi V Raghavan PES2UG23CS319

Section: E

Description about the statement

The Peer Tutoring Matching System is a database designed to organize and manage peer learning among students. It helps match mentors and mentees based on their subjects and departments, schedule mentorship sessions, and collect feedback. Without such a system, it becomes difficult to keep track of who is mentoring whom, what subjects are covered, and how effective each session is. This project aims to solve these issues by creating a structured, efficient, and reliable way to manage peer tutoring activities.

User requirement specification

**1. Functional Requirements (Database-Centered)**

1. **Student Data Management**
   * The database must store student details such as name, email, phone number, department, year, and role (mentor or mentee).
   * Each student must have unique identifiers for student\_id, email, and ph\_no.
   * The role field should ensure valid entries using the ENUM type.
2. **Subject Management**
   * The system should maintain a Subject table containing subject names and descriptions.
   * Each subject must have a unique name.
   * Students should be linked to subjects through a junction table (StudentSubject).
3. **Team Management**
   * Each mentor can lead a team stored in the Team table.
   * The relationship between mentors and mentees within a team is managed using the TeamMember junction table.
   * The mentor’s student\_id acts as a foreign key in the Team table.
4. **Mentorship Session Management**
   * Sessions must be stored in the MentorshipSession table with fields such as subject, date/time, duration, and status.
   * The session’s subject is linked to the Subject table through a foreign key.
   * The status field should automatically update from *scheduled* to *completed* using a **trigger** when the scheduled date has passed.
5. **Session Participant Management**
   * The system must record all participants of each session in the SessionParticipant table.
   * Each record includes session\_id, student\_id, and role.
   * Foreign key constraints ensure valid references to existing students and sessions.
6. **Feedback Management**
   * Feedback for each session should be stored in the Feedback table.
   * It must include rating, comment, and anonymous fields.
   * The rating should be validated using a CHECK constraint (1–5).
7. **Procedure Requirement**
   * A stored procedure (AddMentorshipSession) should simplify inserting a new session along with mentor and mentee records in related tables automatically.
8. **Function Requirement**
   * A stored function (MentorSessionCount) should calculate how many completed sessions a particular mentor has conducted.

**2. Non-Functional Requirements (DB-Oriented)**

1. **Data Integrity**
   * Use **primary keys**, **foreign keys**, **unique constraints**, and **CHECK constraints** to maintain data consistency.
   * Use **referential actions** (ON DELETE and ON UPDATE) to ensure related data stays synchronized.
2. **Reliability**
   * The database should prevent invalid or duplicate entries.
   * Data relationships must remain intact even after deletions or updates.
3. **Security and Privacy**
   * Sensitive data such as email and phone number should remain unique and protected from duplication.
   * The feedback system should allow anonymous submissions.
4. **Scalability**
   * The design should support adding more students, subjects, and sessions without altering the schema.
5. **Performance**
   * Queries, joins, and relationships should execute efficiently to retrieve data across multiple tables.

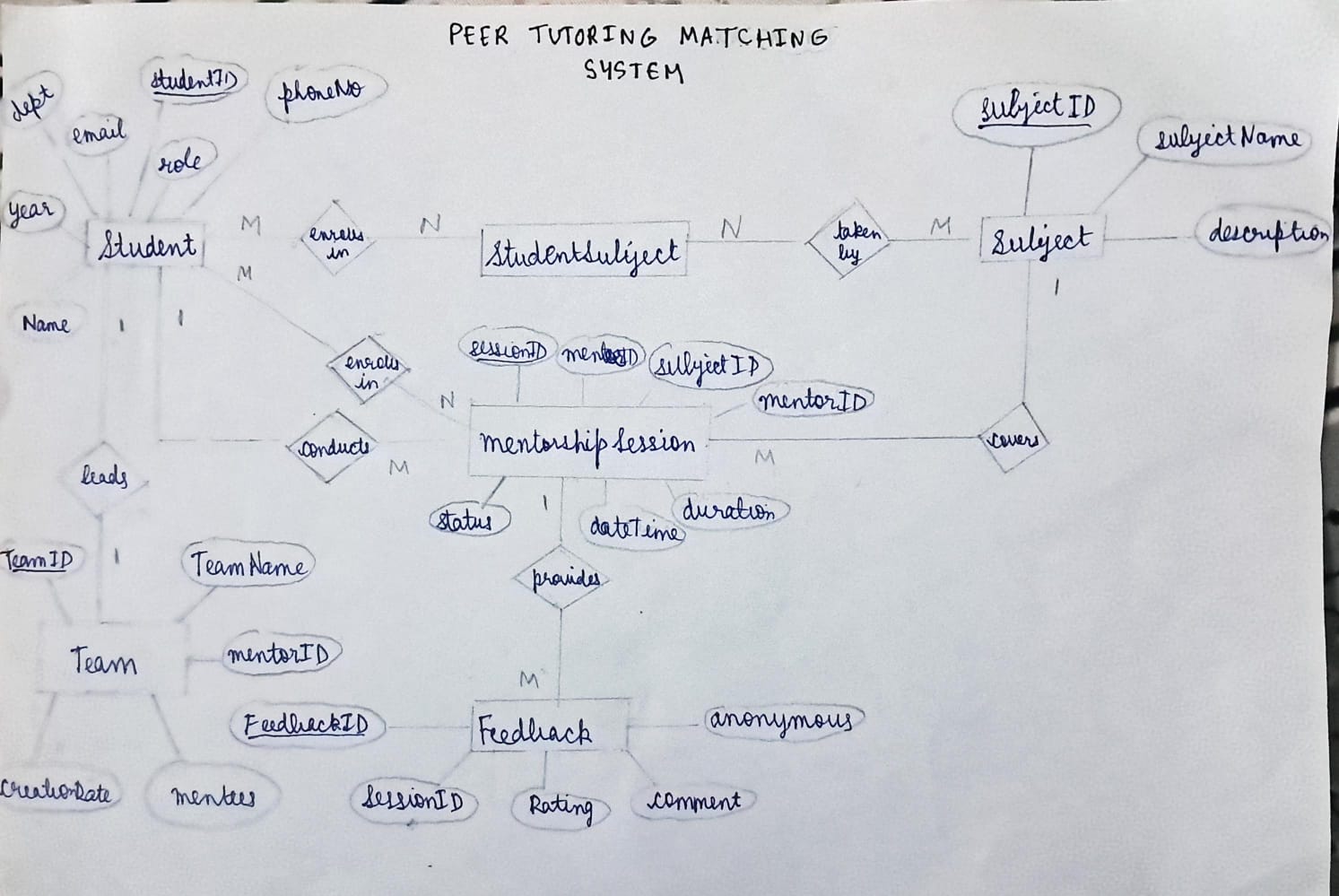
**3. Intended Users**

* **Database Administrator (DBA):** Manages tables, constraints, and data integrity.
* **Faculty Coordinator:** Adds or reviews mentoring sessions and team details.
* **Mentors and Mentees:** Indirectly represented through data entries and relationships.

List of Softwares/Tools/Programming languages used

* 1. **MySQL Workbench** – Used for designing, creating, and managing the database, writing SQL queries, and visualizing the schema.
  2. **MySQL Server** – Backend database server where all the data, tables, triggers, procedures, and functions are stored and executed.
  3. **Python** – Used for building the **frontend interface** to interact with the database (for example, inserting records, viewing sessions, etc.).
  4. **Tkinter / PyQt (optional)** – Used in Python to create the **GUI (Graphical User Interface)** for the Peer Tutoring Management System.
  5. **GitHub** – Used for **version control** and storing the project source code online for collaboration and backup.
  6. **VS Code / PyCharm** – Used as the **code editor** for writing and testing Python and SQL scripts.
  7. **Operating System:** Windows / Linux (any OS supporting MySQL and Python).

ER Diagram



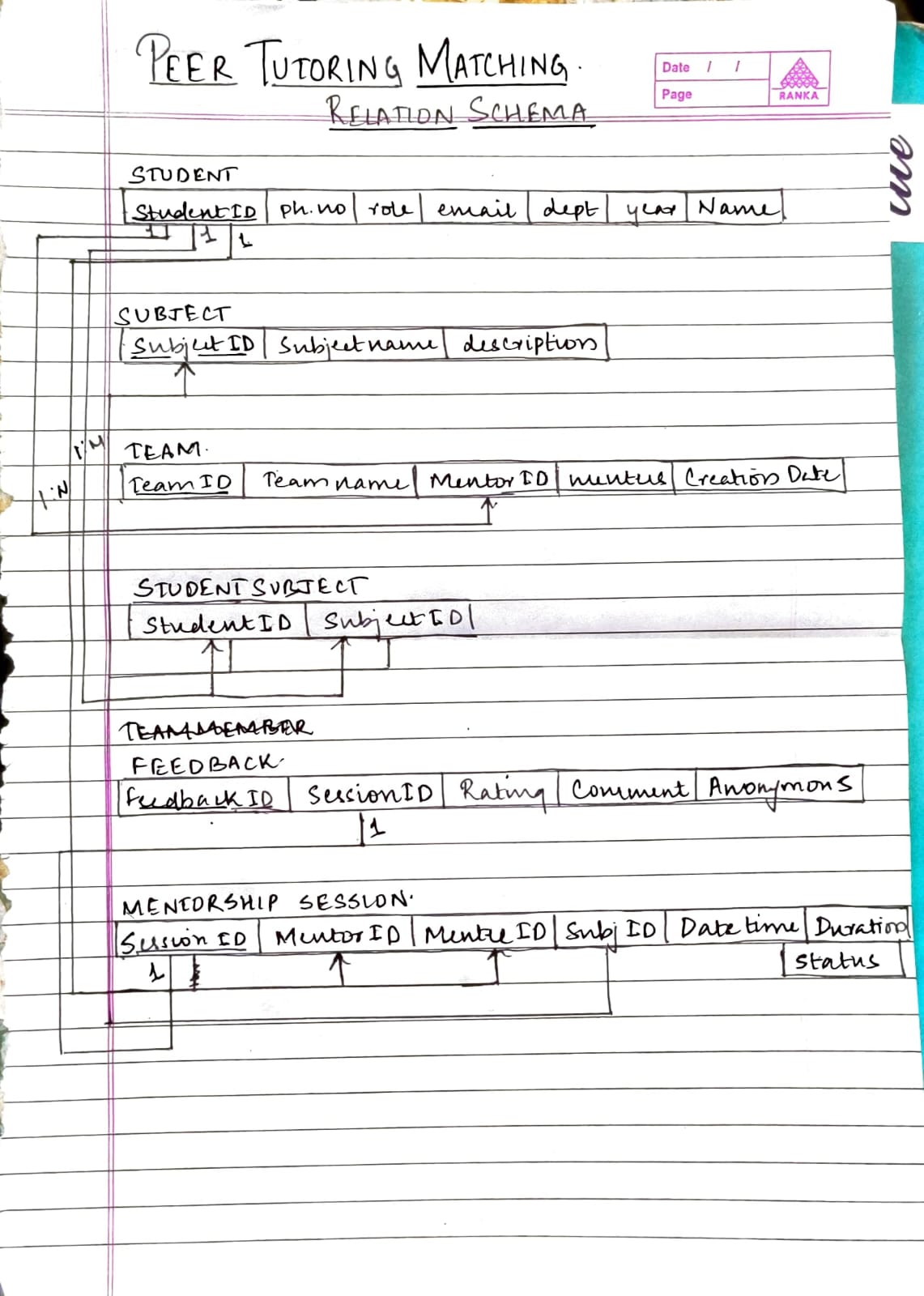
**Entities and Their Attributes**

1. **Student**
   * **Attributes:**
     + *studentID* (Primary Key)
     + *name*
     + *email*
     + *phoneNo*
     + *role* (mentor / mentee)
     + *dept*
     + *year*
   * **Description:**  
     Represents each student in the system. A student can act as either a mentor or a mentee.
2. **Subject**
   * **Attributes:**
     + *subjectID* (Primary Key)
     + *subjectName*
     + *description*
   * **Description:**  
     Contains information about subjects offered for tutoring.
3. **Team**
   * **Attributes:**
     + *teamID* (Primary Key)
     + *teamName*
     + *mentorID* (Foreign Key referencing Student)
   * **Description:**  
     Each team is led by a mentor and includes multiple mentees.
4. **MentorshipSession**
   * **Attributes:**
     + *sessionID* (Primary Key)
     + *subjectID* (Foreign Key referencing Subject)
     + *mentorID* (Foreign Key referencing Student)
     + *dateTime*
     + *duration*
     + *status* (scheduled / completed / cancelled)
   * **Description:**  
     Represents a scheduled or completed mentoring session conducted by a mentor for specific mentees.
5. **Feedback**
   * **Attributes:**
     + *feedbackID* (Primary Key)
     + *sessionID* (Foreign Key referencing MentorshipSession)
     + *rating*
     + *comment*
     + *anonymous* (boolean)
   * **Description:**  
     Stores feedback and ratings provided by mentees after attending a session.
6. **StudentSubject (Associative Entity)**
   * **Attributes:**
     + *studentID* (Foreign Key referencing Student)
     + *subjectID* (Foreign Key referencing Subject)
   * **Description:**  
     Connects students with subjects they learn or teach.

**Relationships**

* **Student – Team:**  
  One student (mentor) *leads* one team; a team can have multiple members (1–M).
* **Student – Subject:**  
  A student *enrolls in* or *teaches* many subjects; each subject can have many students (M–N).
* **Student – MentorshipSession:**  
  A student (as a mentor) *conducts* multiple sessions; mentees *attend* multiple sessions (M–N).
* **MentorshipSession – Feedback:**  
  Each mentorship session *receives* feedback from mentees (1–M).

Relational Schema



The relation schema represents how the database tables are structured and how they are connected with each other. It defines the attributes (columns) of each table and the relationships between them.

1. **Student**
   * **Attributes:** StudentID, Name, Email, Ph\_no, Role, Dept, Year
   * **Description:** Stores details of all students. Each student can be either a **mentor** or a **mentee**.
2. **Subject**
   * **Attributes:** SubjectID, SubjectName, Description
   * **Description:** Contains the list of subjects available for peer tutoring.
3. **Team**
   * **Attributes:** TeamID, TeamName, MentorID, NumMentees, CreationDate
   * **Description:** Each team is managed by one mentor and consists of multiple mentees.
4. **StudentSubject**
   * **Attributes:** StudentID, SubjectID
   * **Description:** This is a **linking table** that connects students with the subjects they are interested in or teaching (many-to-many relationship).
5. **TeamMember**
   * **Attributes:** TeamID, StudentID
   * **Description:** Represents which students belong to which team (many-to-many relationship between students and teams).
6. **Feedback**
   * **Attributes:** FeedbackID, SessionID, Rating, Comment, Anonymous
   * **Description:** Stores the feedback and ratings that mentees give after each mentorship session.
7. **MentorshipSession**
   * **Attributes:** SessionID, MentorID, MenteeID, SubjectID, DateTime, Duration, Status
   * **Description:** Records details of each mentorship session, such as which mentor and mentee participated, the subject discussed, and the session status (scheduled, completed, etc.).

**Relationships**

* One **mentor** can guide **many mentees**.
* A **student** can be linked with multiple **subjects** through the StudentSubject table.
* Each **session** belongs to one **subject**, one **mentor**, and one **mentee**.
* Each **session** can receive multiple **feedback entries** from mentees.