Mini project report on

PESU CLUBS EVENT MANAGEMENT SYSTEM

Submitted in partial fulfilment of the requirements for the award of degree of

Bachelor of Technology

in

Computer Science & Engineering

UE23CS351A – DBMS Project

Submitted by:

Naveen S	**PES2UG23CS369**
Nandita R Nadig	**PES2UG23CS365**

Under the guidance of

Prof. Shilpa S

Assistant Professor

PES University

AUG - DEC 2025

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

FACULTY OF ENGINEERING

PES UNIVERSITY

(Established under Karnataka Act No. 16 of 2013)

Electronic City, Hosur Road, Bengaluru – 560 100, Karnataka, India

PES UNIVERSITY

(Established under Karnataka Act No. 16 of 2013)
Electronic City, Hosur Road, Bengaluru – 560 100, Karnataka, India

CERTIFICATE

This is to certify that the mini project entitled

PESU CLUBS EVENT MANAGEMENT SYSTEM

is a bonafide work carried out by

Naveen S	**PES2UG23CS369**
Nandita R Nadig	**PES2UG23CS365**

In partial fulfilment for the completion of fifth semester DBMS Project (UE20CSS301) in the Program of Study -Bachelor of Technology in Computer Science and Engineering under rules and regulations of PES University, Bengaluru during the period AUG. 2022 – DEC. 2022. It is certified that all corrections / suggestions indicated for internal assessment have been incorporated in the report. The project has been approved as it satisfies the 5th semester academic requirements in respect of project work.

Signature
Prof. Shilpa S
Assistant Professor

DECLARATION

We hereby declare that the DBMS Project entitled **PESU CLUBS EVENT MANAGEMENT SYSTEM** has been carried out by us under the guidance of **Prof. Shilpa S, Assistant Professor** and submitted in partial fulfilment of the course requirements for the award of degree of **Bachelor of Technology** in **Computer Science and Engineering** of **PES University, Bengaluru** during the academic semester AUG – DEC 2025.

Naveen S	**PES2UG23CS369**	**<signature>**
Nandita R Nadig	**PES2UG23CS365**	**<signature>**

ACKNOWLEDGEMENT

I would like to express my gratitude to Prof. Shilpa S, Department of Computer Science and Engineering, PES University, for her continuous guidance, assistance, and encouragement throughout the development of this UE23CS351 - DBMS Project.

I take this opportunity to thank Dr. Sandesh B J, C, Professor, Chairperson, Department of Computer Science and Engineering, PES University, for all the knowledge and support I have received from the department.

I am deeply grateful to Dr. M. R. Doreswamy, Chancellor, PES University, Prof. Jawahar Doreswamy, Pro Chancellor – PES University, Dr. Suryaprasad J, Vice-Chancellor, PES University for providing to me various opportunities and enlightenment every step of the way. Finally, this DBMS Project could not have been completed without the continual support and encouragement I have received from my family and friends.

ABSTRACT

The project utilizes a relational database to centralize essential data related to students, faculty, venues, clubs, and events. The system is built on a well-structured ER model and normalized relational schema that clearly defines the relationships among participants, organizers, registrations, cancellations, and resource management.

Core database logic is implemented through stored procedures, functions, and triggers that automate event registration, handle cancellations, and ensure data validation. These components collectively maintain transactional integrity, check venue availability, and prevent duplicate or invalid entries. Additional triggers are used to maintain audit logs, recording key user actions such as registration, cancellation, and feedback submission to enhance traceability and accountability.

The database is integrated with a modular Streamlit-based user interface that provides distinct functionalities for administrators and general users. Through this interface, users can register for events, view updates, and access real-time event metrics, while administrators can manage event configurations, budgets, venues, and staffing.

Overall, the Campus Event Management System offers a secure, efficient, and user-friendly platform for managing the complete lifecycle of campus events, ensuring consistency, reliability, and streamlined operations through effective database design and automation.

TABLE OF CONTENTS

1. INTRODUCTION

The main goal of this project is to build a simple, reliable, and centralized Database Management System (DBMS) that can handle every stage of event management in a university. Usually, campus events are managed through scattered spreadsheets or manual coordination, which leads to confusion and errors. This system brings everything together on one platform that helps with:

- **Event Planning:** Managing event details, booking venues, and scheduling resources.
- **Participation:** Allowing students to register for both solo and team events.
- **Post-Event Review:** Collecting feedback, handling issues, and generating useful reports.

The project combines both database and application development to make the system practical and interactive. It uses:

- **Database:** MySQL for data storage and relationship management.
- **Backend Logic:** SQL features like stored procedures, functions, and triggers for automation and consistency.
- **Frontend Interface:** Python with Streamlit to provide an easy-to-use interface for students and administrators.

A major focus here to design a strong and efficient database that ensures accuracy, consistency, and smooth operations. This includes:

- **Schema Design:** Creating a normalized structure based on the E–R diagram to remove redundancy.
- **SQL Operations:** Writing stored procedures and functions to handle event registration, cancellations, and other workflows.
- **Automation:** Using triggers to enforce rules (like preventing double registrations) and to keep an audit log of user actions for better accountability.

2. PROBLEM DEFINITION

2.1 Statement of the Problem

Managing campus events usually involves many different people, teams, and activities — from planning and registration to feedback and reporting. Without a proper system, this often leads to confusion and mistakes. The main problem this project solves is the lack of a single, organized platform to handle everything related to university events.

Some of the common issues in the existing process include:

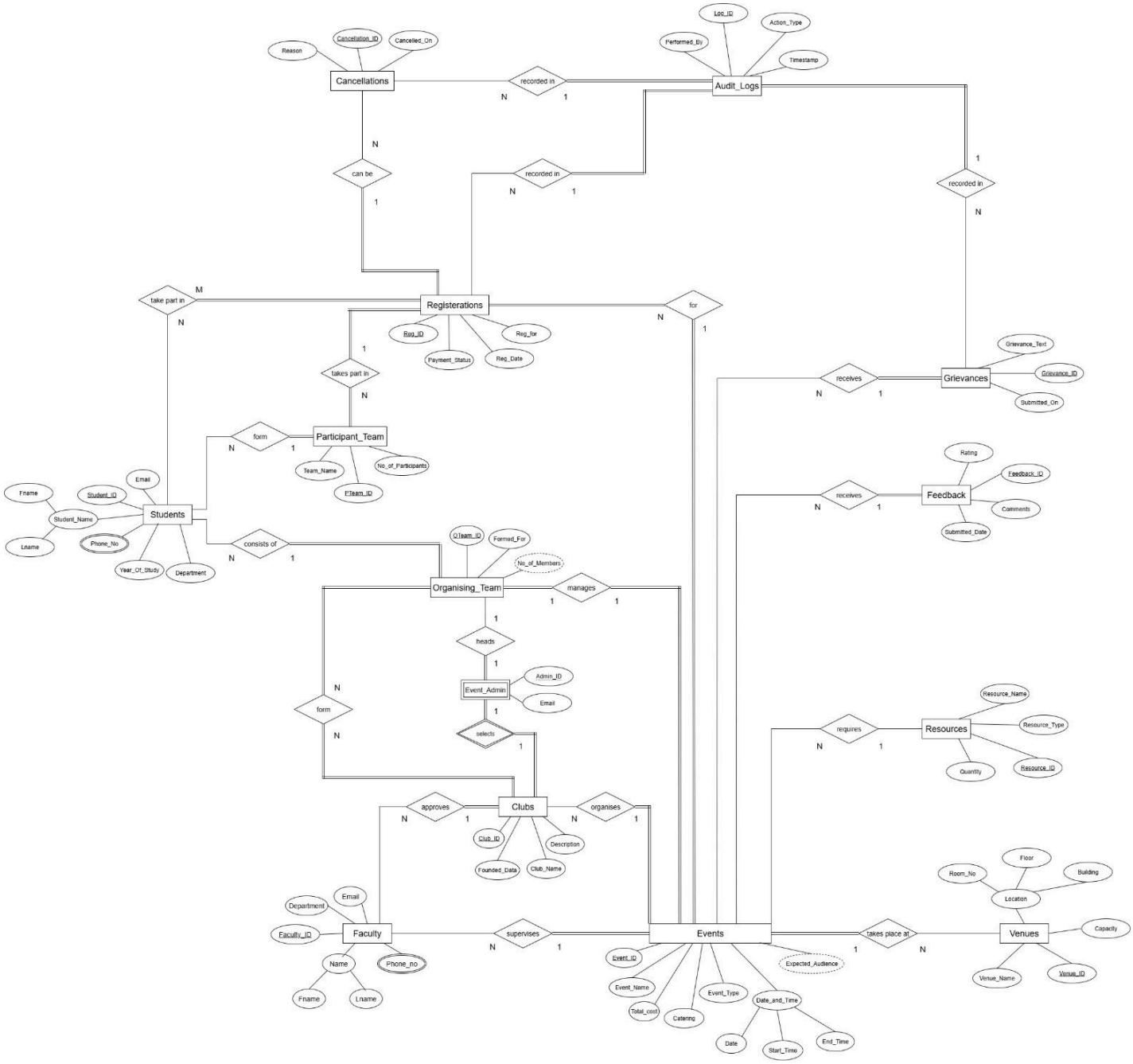
- **Scattered Data:** Information about events, venues, and participants is stored in multiple spreadsheets or files, making it hard to track or update.
- **Scheduling Conflicts:** There's no easy way to check for overlapping venues or timings, which causes clashes and mismanagement.
- **Lack of Accountability:** Since actions like registration, feedback, or cancellation aren't logged properly, it becomes difficult to trace who did what and when.

2.2 Objectives of the Database

The main goal of this database is to create a single and organized system that makes it easier to manage all campus events. It should help both students and administrators handle registrations, scheduling, and reporting without confusion or repeated work. The database focuses on keeping data accurate, preventing mistakes, and making the whole process faster and smoother.

Objective	Description	Outcome / Benefit
Data Integrity	Make sure all data entered into the system is correct and valid — for example, avoiding duplicate registrations or invalid ratings.	Reliable and error-free data.
Operational Efficiency	Reduce manual work by using stored procedures and queries to handle tasks like event registration, cancellation, and venue booking.	Saves time and effort for both users and admins.
Validation & Querying	Use SQL queries and functions to quickly check venue availability and show event details whenever needed.	Easy and quick access to correct information.
Reporting & Analysis	Collect and combine data to generate event summaries and participation reports.	Helps analyze how events are performing and where improvements are needed.
Accountability	Maintain an automatic log of all major actions, like registrations and cancellations.	Ensures transparency and helps track activities if any issue comes up.

3. ER MODEL


Cancellations
Reason
Cancellation_ID
Cancelled_On
recorded in
Audit_Logs
Log_ID
Action_Type
Performed_By
Timestamp
can be
recorded in
recorded in
Registerations
take part in
for
Reg_ID
Reg_for
Payment_Status
Reg_Date
takes part in
Participant_Team
form
Team_Name
No_of_Participants
PTeam_ID
Grievances
receives
Grievance_Text
Grievance_ID
Submitted_On
Feedback
receives
Rating
Feedback_ID
Comments
Submitted_Date
Students
Fname
Student_ID
Email
Student_Name
Lname
Phone_No
Year_Of_Study
Department
consists of
Organising_Team
OTeam_ID
Formed_For
No_of_Members
manages
heads
Event_Admin
Admin_ID
Email
selects
form
Resources
requires
Resource_Name
Resource_Type
Quantity
Resource_ID
Clubs
approves
organises
Club_ID
Description
Founded_Data
Club_Name
Faculty
Department
Email
Faculty_ID
Name
Phone_no
Fname
Lname
supervises
Events
Event_ID
Event_Name
Total_cost
Catering
Event_Type
Date_and_Time
Date
Start_Time
End_Time
Expected_Audience
takes place at
Venues
Floor
Room_No
Building
Location
Capacity
Venue_Name
Venue_ID

4. ER TO RELATIONAL MAPPING

4.1 STEPS OF ALGORITHM FOR CHOSEN PROBLEM

For this project, the E–R diagram contains multiple entities such as *Students, Faculty, Events, Clubs,* and *Venues*, along with several relationships like registrations, teams, and feedback.
The following steps were used to convert the E–R diagram into relational tables:

1. **Identify Regular (Strong) Entities:**

- Each strong entity becomes a separate table.
- The entity's primary key (PK) becomes the table's primary key.

Example: Tables like *Venue*, *Faculty*, *Students*, and *Clubs* were created with their respective IDs as PKs.

2. **Map Weak Entities:**

- A weak entity is converted into a new table.
- Its primary key is a combination of its partial key and the owner entity's PK (as a foreign key).

Example: PTeam_Members uses *PTeam_ID* (FK) and *Student_ID* (FK) as a composite key.

3. **Handle Multi-Valued Attributes:**

- Each multi-valued attribute is stored in a separate table.
- The table includes the attribute and the primary key of the original entity as a foreign key.

Example: Faculty_Phone_No_Table contains *Faculty_ID* (FK) and *Phone_No*.

4. **Map 1:N (One-to-Many) Relationships:**
   Add the primary key of the "one" side as a foreign key to the "many" side.
   Example: Club_ID and *Venue_ID* are added as foreign keys in the Event table.

5. **Map M:N (Many-to-Many) Relationships:**

- Create a new table (intersection table) for the relationship.
- The new table's primary key is the combination of both participating entities' primary keys.

Example: PTeam_Members connects *Students* and *Participating_Team* using both IDs.

6. **Map 1:1 (One-to-One) Relationships:**

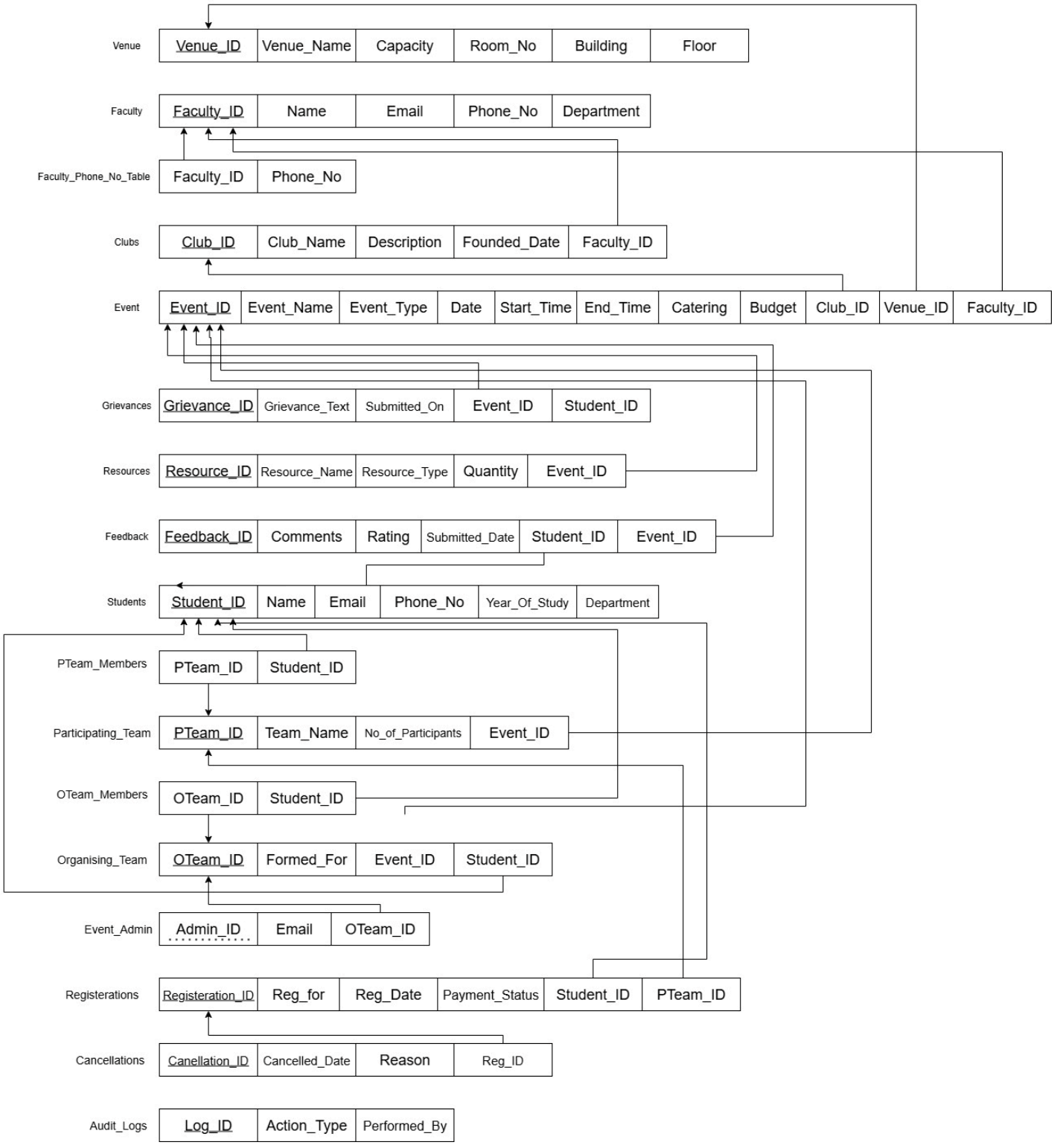
- Place the primary key of one entity as a foreign key in the other entity's table.
- If one side has total participation, the foreign key is added there.

  Example: Admin_ID in *Event_Admin* links to *OTeam_ID*.

7. **Verify Referential Integrity:**

- Ensure all foreign keys correctly reference the primary keys of their parent tables.
- This step helps maintain data consistency across all relationships.

4.2 COMPLETE DIAGRAM OF RELATIONAL MAPPING



5. DDL STATEMENTS

```
mysql> CREATE TABLE Venue (
    ->     Venue_ID INT AUTO_INCREMENT PRIMARY KEY,
    ->     Venue_Name VARCHAR(100),
    ->     Capacity INT,
    ->     Room_No VARCHAR(20),
    ->     Building VARCHAR(50),
    ->     Floor INT
    -> );
Query OK, 0 rows affected (0.14 sec)

mysql> desc Venue;
+------------+--------------+------+-----+---------+----------------+
| Field      | Type         | Null | Key | Default | Extra          |
+------------+--------------+------+-----+---------+----------------+
| Venue_ID   | int          | NO   | PRI | NULL    | auto_increment |
| Venue_Name | varchar(100) | YES  |     | NULL    |                |
| Capacity   | int          | YES  |     | NULL    |                |
| Room_No    | varchar(20)  | YES  |     | NULL    |                |
| Building   | varchar(50)  | YES  |     | NULL    |                |
| Floor      | int          | YES  |     | NULL    |                |
+------------+--------------+------+-----+---------+----------------+
6 rows in set (0.10 sec)
```

```
mysql> CREATE TABLE Faculty (
    ->     Faculty_ID INT AUTO_INCREMENT PRIMARY KEY,
    ->     Name VARCHAR(100),
    ->     Email VARCHAR(100) UNIQUE,
    ->     Department VARCHAR(100)
    -> );
Query OK, 0 rows affected (0.06 sec)

mysql> desc Faculty;
+------------+--------------+------+-----+---------+----------------+
| Field      | Type         | Null | Key | Default | Extra          |
+------------+--------------+------+-----+---------+----------------+
| Faculty_ID | int          | NO   | PRI | NULL    | auto_increment |
| Name       | varchar(100) | YES  |     | NULL    |                |
| Email      | varchar(100) | YES  | UNI | NULL    |                |
| Department | varchar(100) | YES  |     | NULL    |                |
+------------+--------------+------+-----+---------+----------------+
4 rows in set (0.01 sec)
```

```
mysql> CREATE TABLE Faculty_Phone_No_Table (
    ->     Faculty_ID INT,
    ->     Phone_No VARCHAR(20),
    ->     PRIMARY KEY (Faculty_ID, Phone_No)
    -> );
Query OK, 0 rows affected (0.08 sec)

mysql> desc Faculty_Phone_No_Table;
+------------+-------------+------+-----+---------+-------+
| Field      | Type        | Null | Key | Default | Extra |
+------------+-------------+------+-----+---------+-------+
| Faculty_ID | int         | NO   | PRI | NULL    |       |
| Phone_No   | varchar(20) | NO   | PRI | NULL    |       |
+------------+-------------+------+-----+---------+-------+
2 rows in set (0.02 sec)
```

```
mysql> CREATE TABLE Clubs (
    ->     Club_ID INT AUTO_INCREMENT PRIMARY KEY,
    ->     Club_Name VARCHAR(100),
    ->     Description TEXT,
    ->     Founded_Date DATE,
    ->     Faculty_ID INT
    -> );
Query OK, 0 rows affected (0.07 sec)

mysql> desc Clubs;
+-------------+-------------+------+-----+---------+----------------+
| Field       | Type        | Null | Key | Default | Extra          |
+-------------+-------------+------+-----+---------+----------------+
| Club_ID     | int         | NO   | PRI | NULL    | auto_increment |
| Club_Name   | varchar(100) | YES  |     | NULL    |                |
| Description | text        | YES  |     | NULL    |                |
| Founded_Date | date       | YES  |     | NULL    |                |
| Faculty_ID  | int         | YES  |     | NULL    |                |
+-------------+-------------+------+-----+---------+----------------+
5 rows in set (0.04 sec)
```

```
mysql> CREATE TABLE Event (
    ->     Event_ID INT AUTO_INCREMENT PRIMARY KEY,
    ->     Event_Name VARCHAR(100),
    ->     Event_Type VARCHAR(50),
    ->     Date DATE,
    ->     Start_Time TIME,
    ->     End_Time TIME,
    ->     Catering ENUM('Yes', 'No'),
    ->     Budget DECIMAL(12,2),
    ->     Club_ID INT,
    ->     Venue_ID INT,
    ->     Faculty_ID INT
    -> );
Query OK, 0 rows affected (0.04 sec)

mysql> desc Event;
+------------+------------------+------+-----+---------+----------------+
| Field      | Type             | Null | Key | Default | Extra          |
+------------+------------------+------+-----+---------+----------------+
| Event_ID   | int              | NO   | PRI | NULL    | auto_increment |
| Event_Name | varchar(100)     | YES  |     | NULL    |                |
| Event_Type | varchar(50)      | YES  |     | NULL    |                |
| Date       | date             | YES  |     | NULL    |                |
| Start_Time | time             | YES  |     | NULL    |                |
| End_Time   | time             | YES  |     | NULL    |                |
| Catering   | enum('Yes','No') | YES  |     | NULL    |                |
| Budget     | decimal(12,2)    | YES  |     | NULL    |                |
| Club_ID    | int              | YES  |     | NULL    |                |
| Venue_ID   | int              | YES  |     | NULL    |                |
| Faculty_ID | int              | YES  |     | NULL    |                |
+------------+------------------+------+-----+---------+----------------+
11 rows in set (0.00 sec)
```

```
mysql> CREATE TABLE Students (
    ->     Student_ID INT AUTO_INCREMENT PRIMARY KEY,
    ->     Name VARCHAR(100),
    ->     Email VARCHAR(100) UNIQUE,
    ->     Phone_No VARCHAR(20),
    ->     Year_Of_Study INT,
    ->     Department VARCHAR(100)
    -> );
Query OK, 0 rows affected (0.04 sec)

mysql> desc Students;
+--------------+-------------+------+-----+---------+----------------+
| Field        | Type        | Null | Key | Default | Extra          |
+--------------+-------------+------+-----+---------+----------------+
| Student_ID   | int         | NO   | PRI | NULL    | auto_increment |
| Name         | varchar(100) | YES  |     | NULL    |                |
| Email        | varchar(100) | YES  | UNI | NULL    |                |
| Phone_No     | varchar(20)  | YES  |     | NULL    |                |
| Year_Of_Study | int         | YES  |     | NULL    |                |
| Department   | varchar(100) | YES  |     | NULL    |                |
+--------------+-------------+------+-----+---------+----------------+
6 rows in set (0.01 sec)
```

```
mysql> CREATE TABLE Grievances (
    ->     Grievance_ID INT AUTO_INCREMENT PRIMARY KEY,
    ->     Grievance_Text TEXT,
    ->     Submitted_On DATE,
    ->     Event_ID INT,
    ->     Student_ID INT
    -> );
Query OK, 0 rows affected (0.07 sec)

mysql> desc Grievances;
+---------------+------+------+-----+---------+----------------+
| Field         | Type | Null | Key | Default | Extra          |
+---------------+------+------+-----+---------+----------------+
| Grievance_ID  | int  | NO   | PRI | NULL    | auto_increment |
| Grievance_Text | text | YES  |     | NULL    |                |
| Submitted_On  | date | YES  |     | NULL    |                |
| Event_ID      | int  | YES  |     | NULL    |                |
| Student_ID    | int  | YES  |     | NULL    |                |
+---------------+------+------+-----+---------+----------------+
5 rows in set (0.00 sec)
```

```
mysql> CREATE TABLE Resources (
    ->     Resource_ID INT AUTO_INCREMENT PRIMARY KEY,
    ->     Resource_Name VARCHAR(100),
    ->     Resource_Type VARCHAR(50),
    ->     Quantity INT,
    ->     Event_ID INT
    -> );
Query OK, 0 rows affected (0.03 sec)

mysql> desc Resources;
+---------------+-------------+------+-----+---------+----------------+
| Field         | Type        | Null | Key | Default | Extra          |
+---------------+-------------+------+-----+---------+----------------+
| Resource_ID   | int         | NO   | PRI | NULL    | auto_increment |
| Resource_Name | varchar(100) | YES  |     | NULL    |                |
| Resource_Type | varchar(50) | YES  |     | NULL    |                |
| Quantity      | int         | YES  |     | NULL    |                |
| Event_ID      | int         | YES  |     | NULL    |                |
+---------------+-------------+------+-----+---------+----------------+
5 rows in set (0.00 sec)
```

```
mysql> CREATE TABLE Feedback (
    ->     Feedback_ID INT AUTO_INCREMENT PRIMARY KEY,
    ->     Comments TEXT,
    ->     Rating INT,
    ->     Submitted_Date DATE,
    ->     Student_ID INT,
    ->     Event_ID INT
    -> );
Query OK, 0 rows affected (0.07 sec)

mysql> desc Feedback;
+----------------+------+------+-----+---------+----------------+
| Field          | Type | Null | Key | Default | Extra          |
+----------------+------+------+-----+---------+----------------+
| Feedback_ID    | int  | NO   | PRI | NULL    | auto_increment |
| Comments       | text | YES  |     | NULL    |                |
| Rating         | int  | YES  |     | NULL    |                |
| Submitted_Date | date | YES  |     | NULL    |                |
| Student_ID     | int  | YES  |     | NULL    |                |
| Event_ID       | int  | YES  |     | NULL    |                |
+----------------+------+------+-----+---------+----------------+
6 rows in set (0.00 sec)
```

```
mysql> CREATE TABLE Participating_Team (
    ->     PTeam_ID INT AUTO_INCREMENT PRIMARY KEY,
    ->     Team_Name VARCHAR(100),
    ->     No_of_Participants INT,
    ->     Event_ID INT
    -> );
Query OK, 0 rows affected (0.03 sec)

mysql> desc Participating_Team;
+-------------------+-------------+------+-----+---------+----------------+
| Field             | Type        | Null | Key | Default | Extra          |
+-------------------+-------------+------+-----+---------+----------------+
| PTeam_ID          | int         | NO   | PRI | NULL    | auto_increment |
| Team_Name         | varchar(100) | YES  |     | NULL    |                |
| No_of_Participants | int         | YES  |     | NULL    |                |
| Event_ID          | int         | YES  |     | NULL    |                |
+-------------------+-------------+------+-----+---------+----------------+
4 rows in set (0.00 sec)
```

```
mysql> CREATE TABLE PTeam_Members (
    ->     PTeam_ID INT,
    ->     Student_ID INT,
    ->     PRIMARY KEY (PTeam_ID, Student_ID)
    -> );
Query OK, 0 rows affected (0.03 sec)

mysql> desc PTeam_Members;
+------------+------+------+-----+---------+-------+
| Field      | Type | Null | Key | Default | Extra |
+------------+------+------+-----+---------+-------+
| PTeam_ID   | int  | NO   | PRI | NULL    |       |
| Student_ID | int  | NO   | PRI | NULL    |       |
+------------+------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```

```
mysql> CREATE TABLE Organising_Team (
    ->     OTeam_ID INT AUTO_INCREMENT PRIMARY KEY,
    ->     Formed_For VARCHAR(100),
    ->     Event_ID INT
    -> );
Query OK, 0 rows affected (0.04 sec)

mysql> desc Organising_Team;
+------------+--------------+------+-----+---------+----------------+
| Field      | Type         | Null | Key | Default | Extra          |
+------------+--------------+------+-----+---------+----------------+
| OTeam_ID   | int          | NO   | PRI | NULL    | auto_increment |
| Formed_For | varchar(100) | YES  |     | NULL    |                |
| Event_ID   | int          | YES  |     | NULL    |                |
+------------+--------------+------+-----+---------+----------------+
3 rows in set (0.01 sec)
```

```
mysql> CREATE TABLE OTeam_Members (
    ->     OTeam_ID INT,
    ->     Student_ID INT,
    ->     PRIMARY KEY (OTeam_ID, Student_ID)
    -> );
Query OK, 0 rows affected (0.03 sec)

mysql> desc OTeam_Members;
+------------+------+------+-----+---------+-------+
| Field      | Type | Null | Key | Default | Extra |
+------------+------+------+-----+---------+-------+
| OTeam_ID   | int  | NO   | PRI | NULL    |       |
| Student_ID | int  | NO   | PRI | NULL    |       |
+------------+------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```

```
mysql> CREATE TABLE Event_Admin (
    ->     Admin_ID INT AUTO_INCREMENT PRIMARY KEY,
    ->     Student_ID INT,
    ->     OTeam_ID INT,
    ->     Email VARCHAR(100)
    -> );
Query OK, 0 rows affected (0.03 sec)

mysql> desc Event_Admin;
+------------+--------------+------+-----+---------+----------------+
| Field      | Type         | Null | Key | Default | Extra          |
+------------+--------------+------+-----+---------+----------------+
| Admin_ID   | int          | NO   | PRI | NULL    | auto_increment |
| Student_ID | int          | YES  |     | NULL    |                |
| OTeam_ID   | int          | YES  |     | NULL    |                |
| Email      | varchar(100) | YES  |     | NULL    |                |
+------------+--------------+------+-----+---------+----------------+
4 rows in set (0.01 sec)
```

```
mysql> CREATE TABLE Registrations (
    ->     Registration_ID INT AUTO_INCREMENT PRIMARY KEY,
    ->     Reg_For VARCHAR(100),
    ->     Reg_Date DATE,
    ->     Payment_Status VARCHAR(50),
    ->     Student_ID INT,
    ->     PTeam_ID INT
    -> );
Query OK, 0 rows affected (0.04 sec)

mysql> desc Registrations;
+-----------------+--------------+------+-----+---------+----------------+
| Field           | Type         | Null | Key | Default | Extra          |
+-----------------+--------------+------+-----+---------+----------------+
| Registration_ID | int          | NO   | PRI | NULL    | auto_increment |
| Reg_For         | varchar(100) | YES  |     | NULL    |                |
| Reg_Date        | date         | YES  |     | NULL    |                |
| Payment_Status  | varchar(50)  | YES  |     | NULL    |                |
| Student_ID      | int          | YES  |     | NULL    |                |
| PTeam_ID        | int          | YES  |     | NULL    |                |
+-----------------+--------------+------+-----+---------+----------------+
6 rows in set (0.00 sec)
```

```
mysql> CREATE TABLE Cancellations (
    ->     Cancellation_ID INT AUTO_INCREMENT PRIMARY KEY,
    ->     Cancelled_Date DATE,
    ->     Reason TEXT,
    ->     Reg_ID INT
    -> );
Query OK, 0 rows affected (0.02 sec)

mysql> desc Cancellations;
+-----------------+------+------+-----+---------+----------------+
| Field           | Type | Null | Key | Default | Extra          |
+-----------------+------+------+-----+---------+----------------+
| Cancellation_ID | int  | NO   | PRI | NULL    | auto_increment |
| Cancelled_Date  | date | YES  |     | NULL    |                |
| Reason          | text | YES  |     | NULL    |                |
| Reg_ID          | int  | YES  |     | NULL    |                |
+-----------------+------+------+-----+---------+----------------+
4 rows in set (0.00 sec)
```

```
mysql> CREATE TABLE Audit_Logs (
    ->     Log_ID INT AUTO_INCREMENT PRIMARY KEY,
    ->     Action_Type VARCHAR(100),
    ->     Performed_On TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    ->     Student_ID INT,
    ->     Faculty_ID INT,
    ->     Admin_ID INT,
    ->     CHECK (
    ->                 (Student_ID IS NOT NULL AND Faculty_ID IS NULL AND Admin_ID IS NULL)
    ->                 OR (Student_ID IS NULL AND Faculty_ID IS NOT NULL AND Admin_ID IS NULL)
    ->                 OR (Student_ID IS NULL AND Faculty_ID IS NULL AND Admin_ID IS NOT NULL)
    ->     )
    -> );
Query OK, 0 rows affected (0.03 sec)

mysql> desc Audit_Logs;
+--------------+-------------+------+-----+-------------------+-------------------+
| Field        | Type        | Null | Key | Default           | Extra             |
+--------------+-------------+------+-----+-------------------+-------------------+
| Log_ID       | int         | NO   | PRI | NULL              | auto_increment    |
| Action_Type  | varchar(100)| YES  |     | NULL              |                   |
| Performed_On | timestamp   | YES  |     | CURRENT_TIMESTAMP | DEFAULT_GENERATED |
| Student_ID   | int         | YES  |     | NULL              |                   |
| Faculty_ID   | int         | YES  |     | NULL              |                   |
| Admin_ID     | int         | YES  |     | NULL              |                   |
+--------------+-------------+------+-----+-------------------+-------------------+
6 rows in set (0.00 sec)
```

```
mysql> ALTER TABLE Faculty_Phone_No_Table
    -> ADD FOREIGN KEY (Faculty_ID) REFERENCES Faculty(Faculty_ID);
Query OK, 0 rows affected (0.17 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> desc Faculty_Phone_No_Table;
+------------+-------------+------+-----+---------+-------+
| Field      | Type        | Null | Key | Default | Extra |
+------------+-------------+------+-----+---------+-------+
| Faculty_ID | int         | NO   | PRI | NULL    |       |
| Phone_No   | varchar(20) | NO   | PRI | NULL    |       |
+------------+-------------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```

```
mysql> ALTER TABLE Clubs
    -> ADD FOREIGN KEY (Faculty_ID) REFERENCES Faculty(Faculty_ID);
Query OK, 0 rows affected (0.09 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> desc Clubs;
+--------------+--------------+------+-----+---------+----------------+
| Field        | Type         | Null | Key | Default | Extra          |
+--------------+--------------+------+-----+---------+----------------+
| Club_ID      | int          | NO   | PRI | NULL    | auto_increment |
| Club_Name    | varchar(100) | YES  |     | NULL    |                |
| Description  | text         | YES  |     | NULL    |                |
| Founded_Date | date         | YES  |     | NULL    |                |
| Faculty_ID   | int          | YES  | MUL | NULL    |                |
+--------------+--------------+------+-----+---------+----------------+
5 rows in set (0.01 sec)
```

```
mysql> ALTER TABLE Event
    -> ADD FOREIGN KEY (Club_ID) REFERENCES Clubs(Club_ID),
    -> ADD FOREIGN KEY (Venue_ID) REFERENCES Venue(Venue_ID),
    -> ADD FOREIGN KEY (Faculty_ID) REFERENCES Faculty(Faculty_ID);
Query OK, 0 rows affected (0.15 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> desc Event;
+------------+-----------------+------+-----+---------+----------------+
| Field      | Type            | Null | Key | Default | Extra          |
+------------+-----------------+------+-----+---------+----------------+
| Event_ID   | int             | NO   | PRI | NULL    | auto_increment |
| Event_Name | varchar(100)    | YES  |     | NULL    |                |
| Event_Type | varchar(50)     | YES  |     | NULL    |                |
| Date       | date            | YES  |     | NULL    |                |
| Start_Time | time            | YES  |     | NULL    |                |
| End_Time   | time            | YES  |     | NULL    |                |
| Catering   | enum('Yes','No') | YES  |     | NULL    |                |
| Budget     | decimal(12,2)   | YES  |     | NULL    |                |
| Club_ID    | int             | YES  | MUL | NULL    |                |
| Venue_ID   | int             | YES  | MUL | NULL    |                |
| Faculty_ID | int             | YES  | MUL | NULL    |                |
+------------+-----------------+------+-----+---------+----------------+
11 rows in set (0.01 sec)
```

```
mysql> ALTER TABLE Grievances
    -> ADD FOREIGN KEY (Event_ID) REFERENCES Event(Event_ID),
    -> ADD FOREIGN KEY (Student_ID) REFERENCES Students(Student_ID);
Query OK, 0 rows affected (0.08 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> desc Grievances;
+----------------+------+------+-----+---------+----------------+
| Field          | Type | Null | Key | Default | Extra          |
+----------------+------+------+-----+---------+----------------+
| Grievance_ID   | int  | NO   | PRI | NULL    | auto_increment |
| Grievance_Text | text | YES  |     | NULL    |                |
| Submitted_On   | date | YES  |     | NULL    |                |
| Event_ID       | int  | YES  | MUL | NULL    |                |
| Student_ID     | int  | YES  | MUL | NULL    |                |
+----------------+------+------+-----+---------+----------------+
5 rows in set (0.00 sec)
```

```
mysql> ALTER TABLE Resources
    -> ADD FOREIGN KEY (Event_ID) REFERENCES Event(Event_ID);
Query OK, 0 rows affected (0.07 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> desc Resources;
+---------------+-------------+------+-----+---------+----------------+
| Field         | Type        | Null | Key | Default | Extra          |
+---------------+-------------+------+-----+---------+----------------+
| Resource_ID   | int         | NO   | PRI | NULL    | auto_increment |
| Resource_Name | varchar(100) | YES  |     | NULL    |                |
| Resource_Type | varchar(50)  | YES  |     | NULL    |                |
| Quantity      | int         | YES  |     | NULL    |                |
| Event_ID      | int         | YES  | MUL | NULL    |                |
+---------------+-------------+------+-----+---------+----------------+
5 rows in set (0.00 sec)
```

```
mysql> ALTER TABLE Feedback
    -> ADD FOREIGN KEY (Student_ID) REFERENCES Students(Student_ID),
    -> ADD FOREIGN KEY (Event_ID) REFERENCES Event(Event_ID);
Query OK, 0 rows affected (0.07 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> desc Feedback;
+----------------+------+------+-----+---------+----------------+
| Field          | Type | Null | Key | Default | Extra          |
+----------------+------+------+-----+---------+----------------+
| Feedback_ID    | int  | NO   | PRI | NULL    | auto_increment |
| Comments       | text | YES  |     | NULL    |                |
| Rating         | int  | YES  |     | NULL    |                |
| Submitted_Date | date | YES  |     | NULL    |                |
| Student_ID     | int  | YES  | MUL | NULL    |                |
| Event_ID       | int  | YES  | MUL | NULL    |                |
+----------------+------+------+-----+---------+----------------+
6 rows in set (0.00 sec)
```

```
mysql> ALTER TABLE Participating_Team
    -> ADD FOREIGN KEY (Event_ID) REFERENCES Event(Event_ID);
Query OK, 0 rows affected (0.07 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> desc Participating_Team;
+--------------------+--------------+------+-----+---------+----------------+
| Field              | Type         | Null | Key | Default | Extra          |
+--------------------+--------------+------+-----+---------+----------------+
| PTeam_ID           | int          | NO   | PRI | NULL    | auto_increment |
| Team_Name          | varchar(100) | YES  |     | NULL    |                |
| No_of_Participants | int          | YES  |     | NULL    |                |
| Event_ID           | int          | YES  | MUL | NULL    |                |
+--------------------+--------------+------+-----+---------+----------------+
4 rows in set (0.00 sec)
```

```
mysql> ALTER TABLE PTeam_Members
    -> ADD FOREIGN KEY (Student_ID) REFERENCES Students(Student_ID),
    -> ADD FOREIGN KEY (PTeam_ID) REFERENCES Participating_Team(PTeam_ID);
Query OK, 0 rows affected (0.05 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

```
mysql> ALTER TABLE PTeam_Members
    -> ADD CONSTRAINT fk_pteam FOREIGN KEY (PTeam_ID) REFERENCES Participating_Team(PTeam_ID) ON DELETE CASCADE;
Query OK, 0 rows affected (0.11 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> desc PTeam_Members;
+------------+------+------+-----+---------+-------+
| Field      | Type | Null | Key | Default | Extra |
+------------+------+------+-----+---------+-------+
| PTeam_ID   | int  | NO   | PRI | NULL    |       |
| Student_ID | int  | NO   | PRI | NULL    |       |
+------------+------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```

```
mysql> ALTER TABLE Organising_Team
    -> ADD FOREIGN KEY (Event_ID) REFERENCES Event(Event_ID);
Query OK, 0 rows affected (0.05 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> desc Organising_Team;
+------------+--------------+------+-----+---------+----------------+
| Field      | Type         | Null | Key | Default | Extra          |
+------------+--------------+------+-----+---------+----------------+
| OTeam_ID   | int          | NO   | PRI | NULL    | auto_increment |
| Formed_For | varchar(100) | YES  |     | NULL    |                |
| Event_ID   | int          | YES  | MUL | NULL    |                |
+------------+--------------+------+-----+---------+----------------+
3 rows in set (0.00 sec)
```

```
mysql> ALTER TABLE OTeam_Members
    -> ADD FOREIGN KEY (Student_ID) REFERENCES Students(Student_ID),
    -> ADD FOREIGN KEY (OTeam_ID) REFERENCES Organising_Team(OTeam_ID);
Query OK, 0 rows affected (0.04 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> desc OTeam_Members;
+------------+------+------+-----+---------+-------+
| Field      | Type | Null | Key | Default | Extra |
+------------+------+------+-----+---------+-------+
| OTeam_ID   | int  | NO   | PRI | NULL    |       |
| Student_ID | int  | NO   | PRI | NULL    |       |
+------------+------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```

```
mysql> ALTER TABLE Event_Admin
    -> ADD FOREIGN KEY (Student_ID) REFERENCES Students(Student_ID),
    -> ADD FOREIGN KEY (OTeam_ID) REFERENCES Organising_Team(OTeam_ID);
Query OK, 0 rows affected (0.04 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> desc Event_Admin;
+------------+-------------+------+-----+---------+----------------+
| Field      | Type        | Null | Key | Default | Extra          |
+------------+-------------+------+-----+---------+----------------+
| Admin_ID   | int         | NO   | PRI | NULL    | auto_increment |
| Student_ID | int         | YES  | MUL | NULL    |                |
| OTeam_ID   | int         | YES  | MUL | NULL    |                |
| Email      | varchar(100) | YES  |     | NULL    |                |
+------------+-------------+------+-----+---------+----------------+
4 rows in set (0.00 sec)
```

```
mysql> ALTER TABLE Registrations
    -> ADD FOREIGN KEY (Student_ID) REFERENCES Students(Student_ID),
    -> ADD FOREIGN KEY (PTeam_ID) REFERENCES Participating_Team(PTeam_ID);
Query OK, 0 rows affected (0.05 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

```
mysql> ALTER TABLE Registrations
    -> ADD CONSTRAINT fk_reg_pteam FOREIGN KEY (PTeam_ID) REFERENCES Participating_Team(PTeam_ID) ON DELETE SET NULL;
Query OK, 0 rows affected (0.12 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> desc Registrations;
+-----------------+--------------+------+-----+---------+----------------+
| Field           | Type         | Null | Key | Default | Extra          |
+-----------------+--------------+------+-----+---------+----------------+
| Registration_ID | int          | NO   | PRI | NULL    | auto_increment |
| Reg_For         | varchar(100) | YES  |     | NULL    |                |
| Reg_Date        | date         | YES  |     | NULL    |                |
| Payment_Status  | varchar(50)  | YES  |     | NULL    |                |
| Student_ID      | int          | YES  | MUL | NULL    |                |
| PTeam_ID        | int          | YES  | MUL | NULL    |                |
+-----------------+--------------+------+-----+---------+----------------+
6 rows in set (0.00 sec)
```

```
mysql> ALTER TABLE Cancellations
    -> ADD FOREIGN KEY (Reg_ID) REFERENCES Registrations(Registration_ID);
Query OK, 0 rows affected (0.05 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> desc Cancellations;
+------------------+------+------+-----+---------+----------------+
| Field            | Type | Null | Key | Default | Extra          |
+------------------+------+------+-----+---------+----------------+
| Cancellation_ID | int  | NO   | PRI | NULL    | auto_increment |
| Cancelled_Date  | date | YES  |     | NULL    |                |
| Reason          | text | YES  |     | NULL    |                |
| Reg_ID          | int  | YES  | MUL | NULL    |                |
+------------------+------+------+-----+---------+----------------+
4 rows in set (0.00 sec)
```

```
mysql> ALTER TABLE Audit_Logs
    -> ADD FOREIGN KEY (Student_ID) REFERENCES Students(Student_ID),
    -> ADD FOREIGN KEY (Faculty_ID) REFERENCES Faculty(Faculty_ID),
    -> ADD FOREIGN KEY (Admin_ID) REFERENCES Event_Admin(Admin_ID);
Query OK, 0 rows affected (0.09 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> desc Audit_Logs;
+--------------+--------------+------+-----+-------------------+-------------------+
| Field        | Type         | Null | Key | Default           | Extra             |
+--------------+--------------+------+-----+-------------------+-------------------+
| Log_ID       | int          | NO   | PRI | NULL              | auto_increment    |
| Action_Type  | varchar(100) | YES  |     | NULL              |                   |
| Performed_On | timestamp    | YES  |     | CURRENT_TIMESTAMP | DEFAULT_GENERATED |
| Student_ID   | int          | YES  | MUL | NULL              |                   |
| Faculty_ID   | int          | YES  | MUL | NULL              |                   |
| Admin_ID     | int          | YES  | MUL | NULL              |                   |
+--------------+--------------+------+-----+-------------------+-------------------+
6 rows in set (0.00 sec)
```

6. DML STATEMENTS

```
mysql> INSERT INTO faculty (Faculty_ID , Name , Email ) VALUES (6969 , 'Dr.WHat ra Sudeep ' , 'cometomalleshwaram@blore.
maga');
Query OK, 1 row affected (0.06 sec)
```

7. QUERIES

7.1 SIMPLE QUERY WITH GROUP BY, AGRREGATE

```
mysql> SELECT
    ->     COUNT(DISTINCT E.Event_ID) AS Events_Reviewed,
    ->     ROUND(AVG(F.Rating), 2) AS Average_Rating,
    ->     ROUND(AVG(R_Count.Total_Registrations), 0) AS Avg_Registrations_Per_Event
    -> FROM
    ->     Event E
    -> JOIN
    ->     Feedback F ON E.Event_ID = F.Event_ID
    -> LEFT JOIN
    ->     (SELECT Event_ID, COUNT(*) AS Total_Registrations
    ->      FROM Participating_Team PT
    ->      JOIN Registrations R ON PT.PTeam_ID = R.PTeam_ID
    ->      GROUP BY Event_ID) R_Count ON E.Event_ID = R_Count.Event_ID;
+-----------------+----------------+----------------------------+
| Events_Reviewed | Average_Rating | Avg_Registrations_Per_Event |
+-----------------+----------------+----------------------------+
|               3 |           4.25 |                          2 |
+-----------------+----------------+----------------------------+
1 row in set (0.03 sec)
```

7.2 UPDATE OPERATION

```
mysql> UPDATE faculty SET Department = 'Pop Off' WHERE Faculty_ID = 6969;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

7.3 DELETE OPERATION

```
mysql> DELETE FROM faculty WHERE Faculty_ID = 6969;
Query OK, 1 row affected (0.01 sec)
```

7.4 CORRELATED QUERY

```
mysql> SELECT
    ->     S.Name AS Student_Name,
    ->     F.Rating,
    ->     E.Event_Name
    -> FROM
    ->     Feedback F
    -> JOIN
    ->     Students S ON F.Student_ID = S.Student_ID
    -> JOIN
    ->     Event E ON F.Event_ID = E.Event_ID
    -> WHERE
    ->     F.Rating < (
    ->         SELECT AVG(F2.Rating)
    ->         FROM Feedback F2
    ->         WHERE F2.Event_ID = F.Event_ID
    ->     );
+--------------+--------+----------------+
| Student_Name | Rating | Event_Name     |
+--------------+--------+----------------+
| Diya Thomas  |      4 | Hackathon 2025 |
+--------------+--------+----------------+
1 row in set (0.13 sec)
```

7.5 NESTED QUERY

```
mysql> SELECT
    ->     Event_Name,
    ->     Budget
    -> FROM
    ->     Event
    -> WHERE
    ->     Budget > (SELECT AVG(Budget) FROM Event);
+--------------------+----------+
| Event_Name         | Budget   |
+--------------------+----------+
| Hackathon 2025     | 50000.00 |
| AutoCAD Challenge  | 30000.00 |
| Green Campus Design | 50000.00 |
| AI & Robotics Expo | 40000.00 |
+--------------------+----------+
4 rows in set (0.00 sec)
```

8. STORED PROCEDURES, FUCNTIONS AND TRIGGERS

8.1 STORED PROCEDURES OR FUNCTIONS

```
CREATE PROCEDURE RegisterStudentForEvent(
    IN student_id_in INT,
    IN event_id_in INT,
    IN team_name_in VARCHAR(100),
    IN reg_for_in VARCHAR(100),
    IN payment_status_in VARCHAR(50)
)
BEGIN
    DECLARE pteam_id_var INT;

    SELECT pt.PTeam_ID INTO pteam_id_var
    FROM Participating_Team pt
    JOIN PTeam_Members ptm ON pt.PTeam_ID = ptm.PTeam_ID
    WHERE pt.Event_ID = event_id_in AND ptm.Student_ID = student_id_in
    LIMIT 1;

    IF pteam_id_var IS NULL THEN
        INSERT INTO Participating_Team (Team_Name, Event_ID)
        VALUES (team_name_in, event_id_in);
        SET pteam_id_var = LAST_INSERT_ID();

        INSERT INTO PTeam_Members (PTeam_ID, Student_ID)
        VALUES (pteam_id_var, student_id_in);
    END IF;
```

8.2 TRIGGERS

```
DELIMITER $$

CREATE TRIGGER before_feedback_insert
BEFORE INSERT ON feedback
FOR EACH ROW
BEGIN
    DECLARE reg_count INT;
    SELECT COUNT(*) INTO reg_count
    FROM registrations r
    WHERE r.Student_ID = NEW.Student_ID
      AND r.PTeam_ID IN (
          SELECT PTeam_ID FROM participating_team WHERE Event_ID = NEW.Event_ID
      );

    IF reg_count = 0 THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Student cannot submit feedback without registration for this event.';
    END IF;
END$$

DELIMITER ;

-- Trigger 2: Grievances – ensure student participated before grievance
DELIMITER $$
```

9. FRONT END DEVELOPEMNT