Course: Database Systems

Instructor: Noreen Ashraf

Project Report:

Event Management System

(using MYSQL)

Presented by:

1. Muizz Khan (MySQL) F2023376059
2. Samra Tahir (ERD Diagram) F2023376295
3. Aliza Zainab (MySQL) F2023376320
4. Rahim Khan (Relational Schema) F2023376046

1. Summary:

This project involves the design and implementation of a **relational database system** for managing events. The system ensures efficient tracking of clients, venues, services, staff, and bookings using **SQL (MySQL)**. It supports complex queries, improves data accuracy, and solves common issues in traditional event management.

Table of Contents:

2. Introduction:

Managing events involves handling large amounts of data, including clients, venues, staff, and services. Traditional methods like paper records or spreadsheets can lead to errors and delays.

This project introduces an **Event Management System** using **MySQL**, designed to streamline event planning by organizing data in a relational database. It ensures accuracy through proper relationships and allows quick access to information using SQL queries, making event management more efficient and reliable

3. Problem Statement:

This system handles a lot of unique and repeated data from clients, events, venues, services, staff, and cost tables. If not normalized it may cause:

- **3.1** **Difficulties** in managing and updating information.
- **3.2** **Risk** of losing important data
- **3.3** **Wasting time** to retrieve or verify information
- **3.4** **Confusion about** staff assignments or service status
- **3.5** **Difficulty** in calculating total event costs or tracking payments

Estd. 1990

4. Objectives:

4.1 Design a 3NF normalized database.

4.2 Store and **manage** all event related data efficiently.

4.3 Proper linkage of entities using **foreign keys.**

4.4 Performing useful SQL Queries for accurate reporting.

4.5 Improve planning efficiency.

5. System Design:

5.1: Entities and Attributes:

1) **CLIENT Attributes:**

- Client_ID (PK)
- Name
- Phone
- Email

2) **EVENT Attributes:**

- Event_ID (PK)
- Event_Type
- Event_Date
- Client_ID (FK) → CLIENT(Client_ID)

3) **VENUE Attributes:**

- Venue_ID (PK)
- Name
- Location
- Capacity
- Price

4) **BOOKING Attributes:**

- Booking_ID (PK)
- Event_ID (FK) → EVENT(Event_ID)
- Venue_ID (FK) → VENUE(Venue_ID)
- Total_Cost

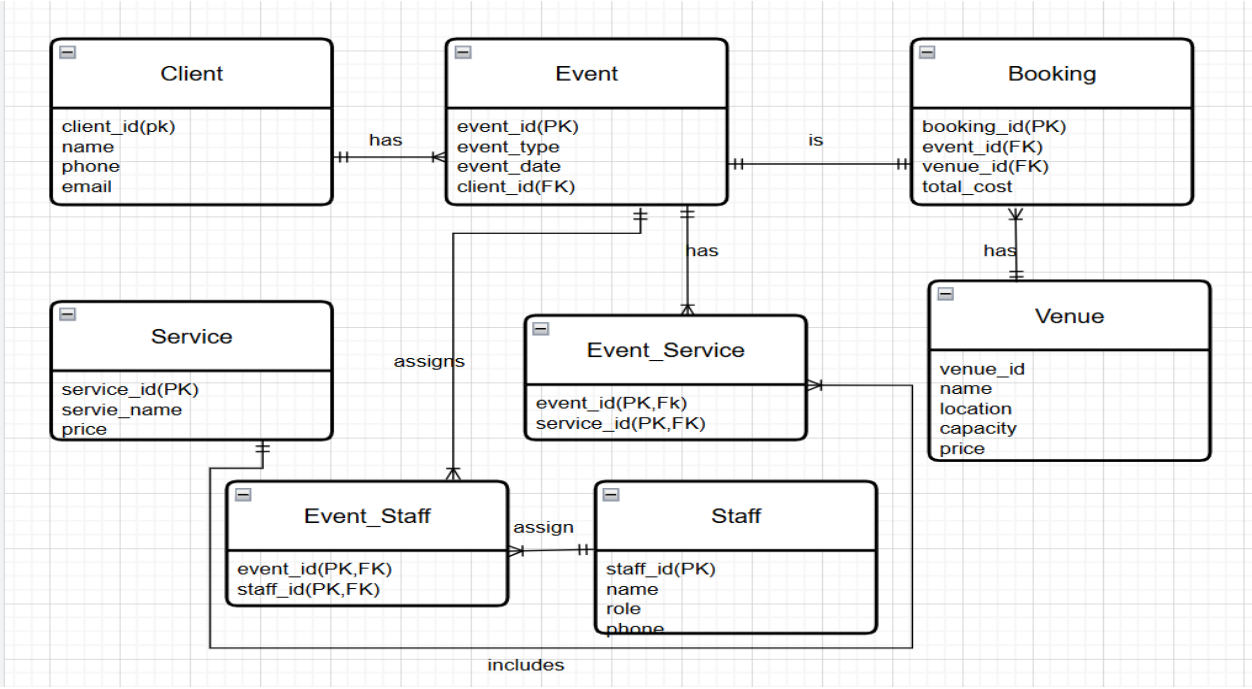
5) **SERVICE Attributes:**
   - Service_ID (PK)
   - Service_Name
   - Price

6) **EVENT_SERVICE Attributes:**
   - Event_ID (PK) (FK) → EVENT(Event_ID)
   - Service_ID (PK) (FK) → SERVICE(Service_ID)

7) **STAFF Attributes:**
   - Staff_ID (PK)
   - Name
   - Role
   - Phone

8) **EVENT_STAFF Attributes:**
   - Event_ID (PK) (FK)→ EVENT(Event_ID)
   - Staff_ID (PK) (FK) → STAFF(Staff_ID)

5.2: ERD Diagram (Crow Foot Notation):



5.3: Relational Schema:

- **CLIENT**(Client_ID [PK], Name, Phone, Email)
- **EVENT**(Event_ID [PK], Event_Type, Event_Date, Client_ID [FK] → CLIENT(Client_ID)
- **VENUE**(Venue_ID [PK], Name, Location, Capacity, Price)
- **BOOKING**(Booking_ID [PK], Event_ID [FK] →EVENT(Event_ID),Venue_ID [FK] → VENUE(Venue_ID), Total_Cost)
- **SERVICE**(Service_ID [PK], Service_Name, Price)
- **EVENT_SERVICE**(Event_ID [PK] [FK] → EVENT(Event_ID),Service_ID [PK] [FK] → SERVICE(Service_ID))
- **STAFF**(Staff_ID [PK],Name, Role, Phone))
- **EVENT_STAFF**(Event_ID [PK] [FK] → EVENT(Event_ID), Staff_ID [PK] [FK] → STAFF(Staff_ID))

6. Implementation:

6.1 SQL Structure Code (Using DDL)

```
-- Creating Database
CREATE DATABASE EVENT_MANAGEMENT_SYSTEM;
USE EVENT_MANAGEMENT_SYSTEM;

-- Creating Table "Client"
CREATE TABLE Client (
    client_id INT PRIMARY KEY AUTO_INCREMENT,
    name VARCHAR(100) NOT NULL,
    phone VARCHAR(20),
    email VARCHAR(100)
);

-- Viewing Table "Client"
DESCRIBE Client;

-- Creating Table "Venue"
CREATE TABLE Venue (
    venue_id INT PRIMARY KEY AUTO_INCREMENT,
    name VARCHAR(100) NOT NULL,
    location VARCHAR(100),
    capacity INT,
    price DECIMAL(10,2)
);
```


```
-- Creating Table "Event"
CREATE TABLE Event (
    event_id INT PRIMARY KEY AUTO_INCREMENT,
    event_type VARCHAR(50),
    event_date DATE,
    client_id INT,
    FOREIGN KEY (client_id) REFERENCES Client(client_id)
);

-- Creating Table "Service"
CREATE TABLE Service (
    service_id INT PRIMARY KEY AUTO_INCREMENT,
    service_name VARCHAR(100),
    price DECIMAL(10,2)
);

-- Creating Table "Staff"
CREATE TABLE Staff (
    staff_id INT PRIMARY KEY AUTO_INCREMENT,
    name VARCHAR(100),
    role VARCHAR(50),
    phone VARCHAR(20)
);
```

```
-- Creating Table "Booking"
CREATE TABLE Booking (
    booking_id INT PRIMARY KEY AUTO_INCREMENT,
    event_id INT,
    venue_id INT,
    total_cost DECIMAL(10,2),
    FOREIGN KEY (event_id) REFERENCES Event(event_id),
    FOREIGN KEY (venue_id) REFERENCES Venue(venue_id)
);

-- Creating Table "Event Services"
CREATE TABLE Event_Service (
    event_id INT,
    service_id INT,
    PRIMARY KEY (event_id, service_id),
    FOREIGN KEY (event_id) REFERENCES Event(event_id),
    FOREIGN KEY (service_id) REFERENCES
Service(service_id)
);

-- Creating Table "Event Staff"
CREATE TABLE Event_Staff (
    event_id INT,
    staff_id INT,
    PRIMARY KEY (event_id, staff_id),
    FOREIGN KEY (event_id) REFERENCES Event(event_id),
    FOREIGN KEY (staff_id) REFERENCES Staff(staff_id) );
```

6.2 Inserting data (Using DML)

-- Inserting 20 Values into Table "Client"

```
INSERT INTO Client (name, phone, email)
VALUES
('Ali Khan', '03001234567', 'ali.khan@example.com'),
('Sara Ahmed', '03111234567', 'sara.ahmed@example.com'),
('Zain Raza', '03221234567', 'zain.raza@example.com'),
('Amna Tariq', '03331234567', 'amna.tariq@example.com'),
('Fahad Shah', '03441234567', 'fahad.shah@example.com'),
('Mehwish Noor', '03551234567', 'mehwish.noor@example.com'),
('Hamza Malik', '03661234567', 'hamza.malik@example.com'),
('Nida Qureshi', '03771234567', 'nida.q@example.com'),
('Omar Farooq', '03881234567', 'omar.f@example.com'),
('Rabia Ali', '03991234567', 'rabia.ali@example.com'),
('Yasir Khan', '03012345678', 'yasir.k@example.com'),
('Fatima Siddiqui', '03112345678', 'fatima.s@example.com'),
('Talha Mirza', '03212345678', 'talha.m@example.com'),
('Sana Javed', '03312345678', 'sana.j@example.com'),
('Imran Hashmi', '03412345678', 'imran.h@example.com'),
('Maha Rehman', '03512345678', 'maha.r@example.com'),
('Bilal Qazi', '03612345678', 'bilal.q@example.com'),
('Areeba Tariq', '03712345678', 'areeba.t@example.com'),
('Asad Saeed', '03812345678', 'asad.s@example.com'),
('Hira Shah', '03912345678', 'hira.shah@example.com');
```

-- Inserting 10 Values into Table "Venue"

```
INSERT INTO Venue (name, location, capacity, price)
VALUES
('Grand Marquee', 'Lahore', 300, 200000.00),
('Event Planet', 'Islamabad', 150, 120000.00),
('Dream Hall', 'Karachi', 250, 180000.00),
('Luxury Lawn', 'Lahore', 400, 250000.00),
('Elegant Events', 'Rawalpindi', 200, 140000.00),
('Crystal Palace', 'Faisalabad', 350, 220000.00),
('The Garden', 'Multan', 180, 130000.00),
('Empire Hall', 'Peshawar', 300, 210000.00),
('Sky Banquet', 'Quetta', 170, 110000.00),
('Royal Pavilion', 'Hyderabad', 230, 160000.00);
```

-- Inserting 10 Values into Table "Service"

```
INSERT INTO Service (service_name, price) VALUES
('Catering', 50000.00),
('Photography', 30000.00),
('Decoration', 40000.00),
('Music/DJ', 20000.00),
('Live Streaming', 15000.00),
('Security', 10000.00),
('Lighting Setup', 25000.00),
('Valet Parking', 8000.00),
('Event Planning', 35000.00),
('Invitation Design', 5000.00);
```

-- Inserting 20 Values into Table "Staff"

```
INSERT INTO Staff (name, role, phone)
VALUES
('Ahmed Raza', 'Photographer', '03031234567'),
('Fatima Noor', 'Event Manager', '03041234567'),
('Usman Malik', 'Catering Head', '03051234567'),
('Ayesha Khan', 'Decorator', '03061234567'),
('Hassan Jamil', 'Lighting Expert', '03071234567'),
('Sana Mir', 'Security Lead', '03081234567'),
('Zeeshan Ali', 'Valet Supervisor', '03091234567'),
('Lubna Rehman', 'Planner', '03101234567'),
('Ibrahim Farid', 'Sound Engineer', '03111234567'),
('Maria Iqbal', 'Live Streamer', '03121234567'),
('Kashif Rao', 'Invitation Designer', '03131234567'),
('Amina Sheikh', 'Catering Assistant', '03141234567'),
('Rizwan Siddiqui', 'DJ', '03151234567'),
('Yumna Tariq', 'Photographer', '03161234567'),
('Adnan Shah', 'Event Assistant', '03171234567'),
('Tariq Mehmood', 'Security', '03181234567'),
('Fariha Malik', 'Receptionist', '03191234567'),
('Waqar Azim', 'Manager', '03201234567'),
('Bushra Nasir', 'Decorator', '03211234567'),
('Shahzad Rafiq', 'Cleaning Head', '03221234567');
```

-- Inserting 20 Values into Table "Event"

```
INSERT INTO Event
(event_type, event_date,
client_id)
VALUES
('Wedding', '2025-07-01', 1),
('Birthday', '2025-07-02', 2),
('Corporate', '2025-07-03', 3),
('Wedding', '2025-07-04', 4),
('Birthday', '2025-07-05', 5),
('Wedding', '2025-07-06', 6),
('Corporate', '2025-07-07', 7),
('Birthday', '2025-07-08', 8),
('Wedding', '2025-07-09', 9),
('Corporate', '2025-07-10', 10),
('Birthday', '2025-07-11', 11),
('Wedding', '2025-07-12', 12),
('Corporate', '2025-07-13', 13),
('Birthday', '2025-07-14', 14),
('Wedding', '2025-07-15', 15),
('Corporate', '2025-07-16', 16),
('Birthday', '2025-07-17', 17),
('Wedding', '2025-07-18', 18),
('Corporate', '2025-07-19', 19),
('Birthday', '2025-07-20', 20));
```

-- Inserting 20 Values into Table "Booking"

```
INSERT INTO Booking
(event_id, venue_id,
total_cost)
VALUES
(1, 1, 280000.00),
(2, 2, 160000.00),
(3, 3, 220000.00),
(4, 4, 270000.00),
(5, 5, 150000.00),
(6, 6, 290000.00),
(7, 7, 240000.00),
(8, 8, 170000.00),
(9, 9, 230000.00),
(10, 10, 180000.00),
(11, 1, 250000.00),
(12, 2, 200000.00),
(13, 3, 260000.00),
(14, 4, 140000.00),
(15, 5, 300000.00),
(16, 6, 210000.00),
(17, 7, 190000.00),
(18, 8, 225000.00),
(19, 9, 235000.00),
(20, 10, 155000.00);
```

-- Inserting 60+ Values into Table "Event Services"

```
INSERT INTO Event_Service
(event_id, service_id)
VALUES
(1, 1), (1, 2), (1, 3),
(2, 1), (2, 4),
(3, 2), (3, 5), (3, 6),
(4, 1), (4, 3), (4, 7),
(5, 4), (5, 5),
(6, 1), (6, 2), (6, 8),
(7, 3), (7, 6), (7, 9),
(8, 4), (8, 5),
(9, 2), (9, 3), (9, 7),
(10, 1), (10, 4),
(11, 5), (11, 6),
(12, 1), (12, 2), (12, 3),
(13, 4), (13, 7),
(14, 8), (14, 9),
(15, 1), (15, 5), (15, 10),
(16, 2), (16, 3), (16, 6),
(17, 1), (17, 9),
(18, 4), (18, 5), (18, 7),
(19, 6), (19, 10),
(20, 3), (20, 4), (20, 8);
```

-- Inserting 50+ Values into Table "Event Staff"

```
INSERT INTO Event_Staff
(event_id, staff_id) VALUES
(1, 1), (1, 2), (1, 3),
(2, 4), (2, 5),
(3, 6), (3, 7), (3, 8),
(4, 9), (4, 10), (4, 11),
(5, 12), (5, 13),
(6, 14), (6, 15), (6, 16),
(7, 17), (7, 18),
(8, 19), (8, 20),
(9, 1), (9, 4),
(10, 5), (10, 6),
(11, 7), (11, 8),
(12, 9), (12, 10),
(13, 11), (13, 12), (13, 13),
(14, 14), (14, 15),
(15, 16), (15, 17), (15, 18),
(16, 19), (16, 20),
(17, 1), (17, 2), (17, 3),
(18, 4), (18, 5), (18, 6),
(19, 7), (19, 8),
(20, 9), (20, 10), (20, 11);
```

6.3 SQL Queries:

```
-- 1. List all events with client names
SELECT e.event_id, e.event_type, e.event_date, c.name AS
client_name
FROM Event e
JOIN Client c ON e.client_id = c.client_id;

-- 2. Show all bookings with total cost > 200,000
SELECT b.booking_id, b.total_cost, v.name AS venue
FROM Booking b
JOIN Venue v ON b.venue_id = v.venue_id
WHERE b.total_cost > 200000;

-- 3. Get all services used in event ID 1
SELECT s.service_name
FROM Event_Service es
JOIN Service s ON es.service_id = s.service_id
WHERE es.event_id = 1;

-- 4. List total number of events per client
SELECT c.name AS client_name, COUNT(e.event_id) AS
total_events
FROM Client c
JOIN Event e ON c.client_id = e.client_id
GROUP BY c.client_id;
```


```
-- 5. Events scheduled in July 2025
SELECT event_id, event_type, event_date
FROM Event
WHERE MONTH(event_date) = 7 AND YEAR(event_date) =
2025;

-- 6. Average total cost of bookings by venue
SELECT v.name AS venue_name, AVG(b.total_cost) AS avg_cost
FROM Booking b
JOIN Venue v ON b.venue_id = v.venue_id
GROUP BY v.venue_id;

-- 7. List staff assigned to Event 5
SELECT s.name, s.role
FROM Event_Staff es
JOIN Staff s ON es.staff_id = s.staff_id
WHERE es.event_id = 5;

-- 8. Top 3 most expensive bookings
SELECT booking_id, total_cost
FROM Booking
ORDER BY total_cost DESC
LIMIT 3;
```

```
-- 9. Clients who booked events costing over 250,000
SELECT DISTINCT c.name
FROM Booking b
JOIN Event e ON b.event_id = e.event_id
JOIN Client c ON e.client_id = c.client_id
WHERE b.total_cost > 250000;

-- 10. Count of each service usage
SELECT s.service_name, COUNT(es.event_id) AS times_used
FROM Service s
JOIN Event_Service es ON s.service_id = es.service_id
GROUP BY s.service_id
ORDER BY times_used DESC
```

7. Conclusion:

The Event Management System successfully demonstrates how a relational database can *simplify and organize* complex event planning tasks. By using MySQL, the system efficiently handles client details, event bookings, venue management, service assignments, and staff allocation.

The use of primary and foreign keys ensures *data integrity*, while SQL queries provide valuable insights such as event summaries, service usage, and cost analysis. Overall, this *system improves accuracy*, reduces *manual workload*, and offers a scalable business.