EVENT MANAGEMENT SYSTEM

Event Management System

User table

```sql
CREATE TABLE Users (
    user_id INT PRIMARY KEY AUTO_INCREMENT,
    name Varchar(20),
    city Varchar(20),
    age int
    );
```

```sql
INSERT INTO Users (name,city,age) VALUES
("Pasindu","Galle",23),
("Navindu","Kandy",34),
("Nithya","Ratnapura",18),
("Tanasha","Nugegoda",20),
("Anjalee","Matara",33),
("Pramod","Kurunegala",25);
```

Events Table

```sql
CREATE TABLE Events(
    event_id int PRIMARY KEY AUTO_INCREMENT,
    event_name Varchar(100),
    category Varchar(50),
    ticket_price INT
    );
```

```sql
INSERT INTO Events(event_name,category,ticket_price) VALUES
("Tech Summit 2025","Technology",1500),
("Jazz Night","Music",800),
("Startup Pitch","Business",1200),
("Art & Craft Fair","Arts",300),
("Yoga Retreat","Wellness",2500),
("Food Carnival","Food",1000);
```

Registrations Table

```
CREATE TABLE Registrations (
    registration_id int PRIMARY KEY AUTO_INCREMENT,
    user_id int,
    event_id int,
    feedback_rating int,
    attended boolean,
    FOREIGN KEY (user_id) REFERENCES Users(user_id),
    FOREIGN KEY (event_id) REFERENCES Events(event_id)
    );
```

```
INSERT INTO Registrations
(user_id,event_id,feedback_rating,attended)VALUES
(1,1,9,1),
(2,1,8,1),
(3,2,7,1),
(4,3,9,1),
(5,3,7,0),
(1,4,6,1),
(2,5,10,1),
(6,5,9,1),
(3,3,8,1),
(6,2,6,0);
```

CRUD operation

Select

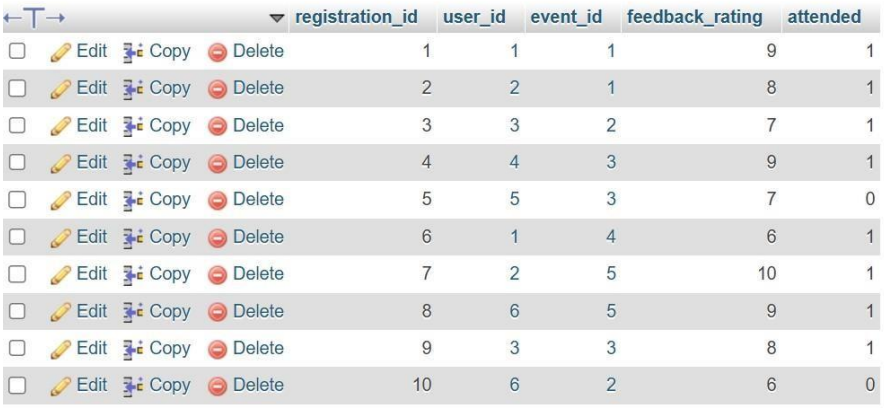
SELECT *FROM Users;

	user_id	name	city	age
Edit Copy Delete	1	Pasindu	Galle	23
Edit Copy Delete	2	Navindu	Kandy	34
Edit Copy Delete	3	Nithya	Ratnapura	18
Edit Copy Delete	4	Tanasha	Nugegoda	20
Edit Copy Delete	5	Anjalee	Matara	33
Edit Copy Delete	6	Pramod	Kurunegala	25

SELECT *FROM Events;

	event_id	event_name	category	ticket_price
Edit Copy Delete	1	Tech Summit 2025	Technology	1500
Edit Copy Delete	2	Jazz Night	Music	800
Edit Copy Delete	3	Startup Pitch	Business	1200
Edit Copy Delete	4	Art & Craft Fair	Arts	300
Edit Copy Delete	5	Yoga Retreat	Wellness	2500
Edit Copy Delete	6	Food Carnival	Food	1000

SELECT *FROM Registrations;



	registration_id	user_id	event_id	feedback_rating	attended
Edit Copy Delete	1	1	1	9	1
Edit Copy Delete	2	2	1	8	1
Edit Copy Delete	3	3	2	7	1
Edit Copy Delete	4	4	3	9	1
Edit Copy Delete	5	5	3	7	0
Edit Copy Delete	6	1	4	6	1
Edit Copy Delete	7	2	5	10	1
Edit Copy Delete	8	6	5	9	1
Edit Copy Delete	9	3	3	8	1
Edit Copy Delete	10	6	2	6	0

Filter

Select event_id, event_name

From Events

Where ticket_price >1000 AND category = “Business”;

	event_id	event_name
Edit Copy Delete	3	Startup Pitch

Sort

Select user_id, name, city

From Users

Order By city ASC LIMIT 3;

		user_id	name	city ▲ 1
☐	Edit Copy Delete	1	Pasindu	Galle
☐	Edit Copy Delete	2	Navindu	Kandy
☐	Edit Copy Delete	6	Pramod	Kurunegala

Update

Update Events

Set event_name = “Food festival”

Where event_id =6;

		event_id	event_name	category	ticket_price
☐	Edit Copy Delete	1	Tech Summit 2025	Technology	1500
☐	Edit Copy Delete	2	Jazz Night	Music	800
☐	Edit Copy Delete	3	Startup Pitch	Business	1200
☐	Edit Copy Delete	4	Art & Craft Fair	Arts	300
☐	Edit Copy Delete	5	Yoga Retreat	Wellness	2500
☐	Edit Copy Delete	6	Food festival	Food	1000

Delete

Delete From Events

Where event_id =6;

	event_id	event_name	category	ticket_price
Edit Copy Delete	1	Tech Summit 2025	Technology	1500
Edit Copy Delete	2	Jazz Night	Music	800
Edit Copy Delete	3	Startup Pitch	Business	1200
Edit Copy Delete	4	Art & Craft Fair	Arts	300
Edit Copy Delete	5	Yoga Retreat	Wellness	2500

Advanced SQL Analysis

1.Analyze event popularity and user participation

```
SELECT event_id,
       COUNT(r.registration_id) AS Event_popularity,
       SUM(r.attended) AS User_participation
 FROM Registrations r
 GROUP BY event_id;
```

event_id	Event_popularity	User_participation
1	2	2
2	2	1
3	3	2
4	1	1
5	2	2

2.Calculate total registrations & average feedback for each event.

```
SELECT event_id,
       COUNT(r.registration_id)AS Total_Registration,
       AVG(r.feedback_rating)AS Average_feedback
FROM Registrations r
GROUP BY event_id;
```

event_id	Total_Registration	Average_feedback
1	2	8.5000
2	2	6.5000
3	3	8.0000
4	1	6.0000
5	2	9.5000

3.Find events with an average rating greater than 8.

```
SELECT e.event_id, e.event_name,
       AVG(r.feedback_rating) AS Avg_rating
FROM Events e
JOIN Registrations r
ON r.event_id = e.event_id
GROUP BY event_id,event_name
HAVING AVG(r.feedback_rating)>8;
```

event_id	event_name	Avg_rating
1	Tech Summit 2025	8.5000
5	Yoga Retreat	9.5000

4.Determine total revenue generated by each event.

```
SELECT e.event_id,e.event_name,
       COUNT(r.registration_id)*ticket_price AS Total_revenue
FROM Events e
JOIN Registrations r
ON r.event_id = e.event_id
GROUP BY event_id,event_name,ticket_price;
```

event_id	event_name	Total_revenue
1	Tech Summit 2025	3000
2	Jazz Night	1600
3	Startup Pitch	3600
4	Art & Craft Fair	300
5	Yoga Retreat	5000

5.Identify top-performing events in each category.

```
SELECT e.event_id, e.category,
         AVG(r.feedback_rating) AS Top_performing
FROM Events e
JOIN Registrations r
ON e.event_id = r.event_id
GROUP BY event_id,category
ORDER BY Top_performing DESC;
```

event_id	category	Top_performing ▾ 1
5	Wellness	9.5000
1	Technology	8.5000
3	Business	8.0000
2	Music	6.5000
4	Arts	6.0000

6.Display users who attended more than one event

```
SELECT u.user_id,u.name,
       COUNT(r.registration_id) AS Total_attended
FROM Users u
JOIN Registrations r
ON r.user_id= u.user_id
WHERE attended = 1
GROUP BY user_id,name
HAVING Total_attended >1;
```

user_id	name	Total_attended
1	Pasindu	2
2	Navindu	2
3	Nithya	2

7.Show users who have not attended any event.

```
SELECT u.user_id, u.name
FROM Users u
LEFT JOIN Registrations r
ON r.user_id = u.user_id
AND attended = 1
WHERE r.user_id IS NULL;
```

user_id	name
5	Anjalee

8.Retrieve users who attended the most expensive event.

```
SELECT u.user_id,u.name
FROM Users u
JOIN Registrations r
ON r.user_id = u.user_id
JOIN Events e
ON e.event_id = r.event_id
WHERE ticket_price = (
    SELECT MAX(e.ticket_price) FROM Events e)AND r.attended =1;
```

user_id	name
2	Navindu
6	Pramod

9. Generate a leader board showing each user's total attended events and average rating.

```
SELECT u.user_id,u.name,
       COUNT(r.registration_id)AS total_attended_events,
       AVG(r.feedback_rating) AS Average_rating
FROM  Users u
JOIN Registrations r
ON r.user_id = u.user_id
WHERE attended = 1
GROUP BY user_id,name
ORDER BY total_attended_events DESC, Average_rating DESC;
```

user_id	name	total_attended_events ▾ 1	Average_rating ▾ 2
2	Navindu	2	9.0000
3	Nithya	2	7.5000
1	Pasindu	2	7.5000
6	Pramod	1	9.0000
4	Tanasha	1	9.0000