# Event Management Database Report

## Event Management System - Full Project Report

Event Management System - Project Summary and Database Documentation

1. **Project Overview**

The Event Management System is designed to simplify the process of organizing and managing events. It allows users to register, create events, manage attendees, coordinate with vendors and sponsors, and maintain all related data efficiently. The system ensures data consistency and prevents scheduling conflicts through database triggers and validation functions.

2. **Project Objectives**

- Streamline the process of event planning and execution.
- Enable user-friendly management of attendees, vendors, and sponsors.
- Automate conflict detection for event scheduling.
- Provide summarized event reports and analytics.
- Maintain data integrity across related tables.

3. **Key Features**

- **User Management:** Secure user registration and authentication.
- **Event Management:** Create, update, and delete event details with conflict prevention.
- **Attendee Tracking:** Record and manage attendees for each event.
- **Vendor and Sponsor Management:** Track service providers, payments, and sponsor contributions.
- **Event Items:** Maintain a list of items needed for successful event execution.

# Event Management Database Report

- **Automated Summaries:** Generate event summaries through stored procedures.

4. **Database Design Summary**

- **Database Name:** event5
- **Core Tables:**
  - `users`: Stores user details and credentials.
  - `events`: Main table for event metadata including time, location, and user association.
  - `attendees`: Maintains event participant data.
  - `vendors`: Manages event vendor details and payments.
  - `sponsors`: Records sponsor details and their contribution amounts.
  - `event_items`: Tracks inventory and supplies per event.
- **Constraints:**
  - Primary keys ensure unique identification.
  - Foreign keys establish relationships across tables.
  - Cascading deletions maintain data integrity.

5. **Business Logic Implementations**

- **Function `check_login_credentials`:** Verifies user credentials and ensures secure access.
- **Triggers `prevent_venue_conflicts` & `prevent_venue_conflicts_update`:** Automatically detect and prevent event scheduling overlaps at the same location.
- **Stored Procedure `get_event_summary`:** Provides a concise report with attendee count and vendor cost aggregation.

6. **Sample SQL Queries and Analysis**

# Event Management Database Report

- **Nested Query:** Retrieves usernames of event organizers based on event location.
- **Correlated Subquery:** Calculates attendee count dynamically for each event.
- **Aggregations:** Provides total event counts per location and user with `GROUP BY` and `HAVING` clauses.
- **Joins:** Combines event and vendor information for detailed financial analysis.

7. **Implementation Details**

- The backend database is built on MySQL.
- The SQL file (`event_management.sql`) defines all tables, triggers, and stored procedures required for the application.
- The system can be integrated with a web-based front-end developed in frameworks such as Django, Flask, or Node.js, which interacts with this SQL schema.

8. **Future Enhancements**

- Adding user roles and permissions for different system actors (Admin, Organizer, Vendor).
- Integrating payment gateways for online transactions.
- Implementing automated email notifications for attendees and vendors.
- Providing analytics dashboards for event performance insights.

9. **Conclusion**

The Event Management System offers a complete, relational, and efficient solution for handling events from creation to completion. It supports automation, data consistency, and scalability, making it suitable for both small and large-scale event operations.