Laporan Implementasi Skema SQL
Proyek Database "Eventify"

Disusun oleh Kelompok 9:

1. Rafael Mahardika Arya Dewamurti (24/536279/PA/22755)
2. Bobby Rahman Hartanto (24/539383/PA/22903)

DEPARTEMEN ILMU KOMPUTER DAN ELEKTRONIKA INSTRUMENTASI
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS GADJAH MADA
YOGYAKARTA
2025

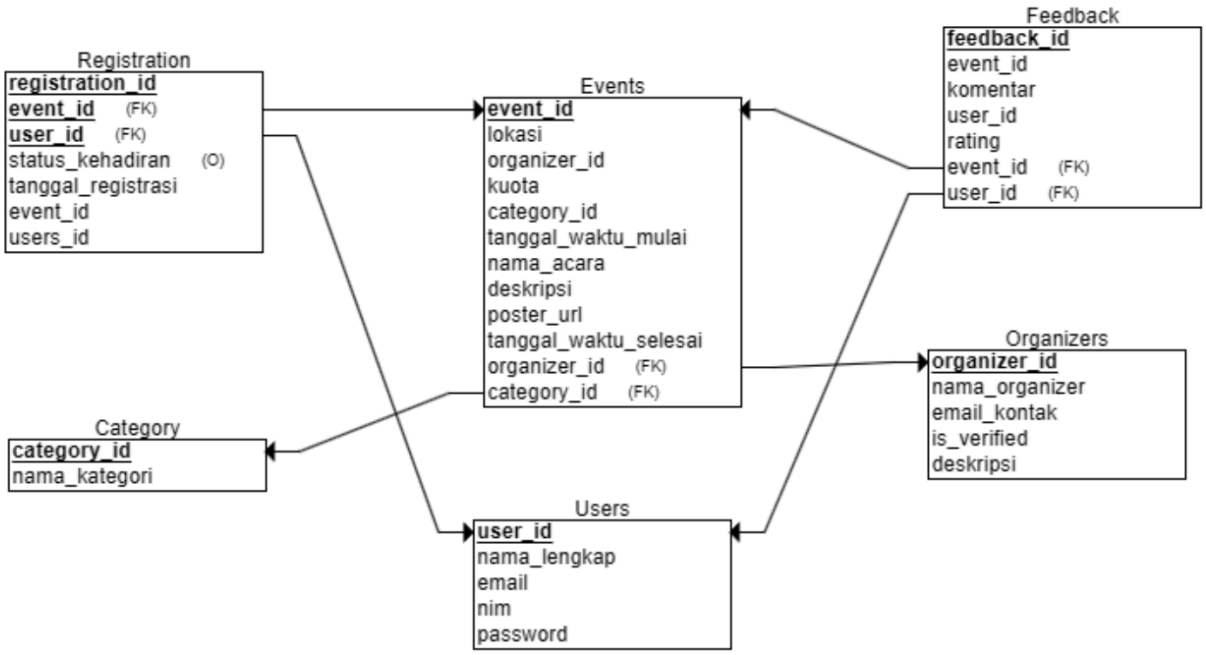
I. Pendahuluan

Dokumen ini adalah Laporan *Progress* untuk tugas *Week* 2. Tujuannya adalah untuk menjelaskan proses penerjemahan *Entity-Relationship Diagram* (ERD) konseptual dari proposal Week 1 menjadi skema relasional yang logis dan implementasi skrip SQL (Data Definition Language - DDL).

Skrip SQL (schema_eventify.sql) yang menyertai laporan ini dirancang untuk menciptakan struktur, relasi, dan batasan (constraints) yang diperlukan untuk database "Eventify" agar dapat berfungsi dengan integritas data yang terjamin.

II. Visualisasi Skema Relasional Logis (LRS)

Gambar di bawah ini adalah *Logical Relational Schema* (LRS) yang menjadi jembatan visual antara desain ERD konseptual (Week 1) dan implementasi kode SQL (Week 2).



LRS ini mengkonfirmasi struktur tabel yang berasal dari ERD Week 1. Penjelasan utama dari skema ini adalah:

- **Tabel Induk (Master):** Tabel Users, Organizers, dan Categories ditampilkan sebagai tabel independen yang menyimpan data master. Kolom PRIMARY KEY (PK) mereka (seperti user_id, organizer_id, category_id) akan menjadi referensi bagi tabel lain.
- **Tabel Transaksi (Events):** Tabel Events adalah pusat dari skema, yang terhubung ke Organizers dan Categories. Garis-garis relasi di LRS menunjukkan bahwa kolom

organizer_id dan category_id di dalam Events adalah *Foreign Key* (FK).

- **Tabel Junction (M:N):** Relasi *Many-to-Many* (M:N) diimplementasikan melalui dua tabel penghubung:
  1. Registrations: Menghubungkan Users dan Events.
  2. Feedback: Juga menghubungkan Users dan Events.

Setiap garis di LRS ini diterjemahkan langsung menjadi sintaks FOREIGN KEY (...) REFERENCES (...) di dalam file schema.sql.

III. Analisis Sintaks SQL per Tabel

Skema database ini terdiri dari enam tabel. Urutan pembuatan tabel sangat penting untuk memenuhi ketergantungan relasional (*referential integrity*). Tabel tanpa *Foreign Key* (induk) harus dibuat terlebih dahulu.

Urutan Pembuatan:

1. Users, Organizers, Categories (Tabel Induk)
2. Events (Tabel Anak, bergantung pada Organizers dan Categories)
3. Registrations, Feedback (Tabel Anak, bergantung pada Users dan Events)

A. Tabel Users

Tabel ini menyimpan data fundamental mahasiswa atau peserta. Sintaks CREATE TABLE Users (...) mendefinisikan kolom-kolom berikut:

- user_id INT AUTO_INCREMENT PRIMARY KEY: Kolom ini berfungsi sebagai pengidentifikasi unik (Primary Key) untuk setiap pengguna. Penggunaan AUTO_INCREMENT sangat membantu karena membebaskan aplikasi dari tugas mengelola ID; database secara otomatis akan memberikan ID unik (1, 2, 3, ...) untuk setiap pengguna baru.
- nama_lengkap VARCHAR(255) NOT NULL: Kolom teks dengan batas 255 karakter. Diberi batasan NOT NULL untuk memastikan nama pengguna wajib diisi.
- email VARCHAR(255) NOT NULL UNIQUE: Selain NOT NULL, batasan UNIQUE ditambahkan di sini sebagai kunci integritas bisnis yang krusial. Ini mencegah dua pengguna mendaftar dengan alamat email yang sama.
- password VARCHAR(255) NOT NULL: Kolom ini akan menyimpan *hash* dari kata sandi, bukan teks aslinya, untuk menjaga keamanan data.
- nim VARCHAR(50) UNIQUE: Mirip dengan email, UNIQUE diterapkan untuk memastikan tidak ada duplikasi data mahasiswa berdasarkan NIM.

B. Tabel Organizers dan Categories

Dua tabel ini adalah tabel *master* atau *lookup* yang berfungsi untuk menyediakan data bagi tabel Events. Desain ini penting untuk normalisasi (3NF), karena detail penyelenggara atau kategori tidak bergantung pada acara, melainkan sebaliknya.

- organizer_id dan category_id keduanya diatur sebagai INT AUTO_INCREMENT PRIMARY KEY.
- Pada Organizers, kolom is_verified BOOLEAN diberi nilai DEFAULT false. Ini adalah pengaturan yang logis, di mana setiap penyelenggara baru yang mendaftar akan otomatis dianggap "belum terverifikasi" sampai disetujui oleh Administrator.
- Pada Categories, kolom nama_kategori juga disetel UNIQUE untuk mencegah duplikasi data master (misalnya, mencegah seseorang memasukkan "Seminar" dua kali).

C. Tabel Events

Ini adalah tabel inti dari aplikasi yang menyimpan semua data acara. Tabel ini memiliki ketergantungan (Foreign Key) pada Organizers dan Categories.

- tanggal_waktu_mulai DATETIME NOT NULL: Penggunaan tipe data DATETIME sangat penting untuk penjadwalan. Kolom ini diwajibkan (NOT NULL).
- tanggal_waktu_selesai DATETIME: Kolom ini **tidak** memiliki NOT NULL. Ini adalah keputusan desain yang disengaja. Ini mengizinkan penyelenggara untuk membuat "draft" acara di sistem meskipun waktu selesainya belum final atau belum ditentukan.
- kuota INT DEFAULT 0: Jika penyelenggara tidak mengisi kuota, sistem akan otomatis mengaturnya ke 0 (nol), bukan NULL, untuk menghindari ambiguitas dalam perhitungan.

D. Tabel Registrations dan Feedback (Tabel Junction)

Tabel-tabel ini adalah implementasi teknis dari relasi *Many-to-Many* (M:N) yang diidentifikasi dalam ERD. Registrations menghubungkan Users dengan Events, dan Feedback juga menghubungkan Users dengan Events.

Beberapa batasan penting ditambahkan di sini untuk menjaga integritas data:

- **Nilai Default Otomatis:** Pada Registrations, kolom tanggal_registrasi menggunakan DEFAULT CURRENT_TIMESTAMP. Ini adalah fitur SQL yang sangat efisien. Aplikasi tidak perlu mengirim data waktu; database akan otomatis mencatat kapan pendaftaran itu terjadi berdasarkan waktu server.
- **Mencegah Duplikasi:** Batasan UNIQUE(user_id, event_id) pada Registrations adalah implementasi aturan bisnis yang vital: "seorang mahasiswa tidak boleh mendaftar di

acara yang sama lebih dari satu kali". Hal yang sama berlaku untuk UNIQUE(event_id, user_id) di tabel Feedback.

- **Validasi Data:** Pada Feedback, batasan CHECK (rating >= 1 AND rating <= 5) diterapkan. Ini adalah batasan di level database yang akan menolak masukan data jika aplikasi mencoba memasukkan nilai di luar rentang 1-5, memastikan data rating selalu valid.

E. Analisis Relasi dan Foreign Key (Aksi ON DELETE)

Bagian terpenting dari implementasi skema adalah mendefinisikan bagaimana tabel-tabel saling terhubung dan apa yang terjadi jika data induk dihapus.

Sintaks: FOREIGN KEY (kolom_lokal) REFERENCES TabelInduk(kolom_induk) [aksi]

Dalam desain ini, kami menggunakan dua strategi aksi ON DELETE yang berbeda:

1. **ON DELETE SET NULL** (pada tabel Events)
   - **Logika:** Aksi ini diterapkan pada organizer_id dan category_id. Jika sebuah Organizer (UKM) dibubarkan dan dihapus dari database, apa yang terjadi pada Events yang mereka buat?
   - **Keuntungan:** Dengan SET NULL, acara tersebut **tidak ikut terhapus**. Kolom organizer_id-nya hanya akan diatur menjadi NULL. Ini adalah keputusan desain untuk menjaga data historis acara. Acara tersebut menjadi "yatim" (orphaned), tapi datanya tetap ada untuk arsip.
2. **ON DELETE CASCADE** (pada Registrations dan Feedback)
   - **Logika:** Aksi ini diterapkan pada event_id dan user_id. Jika sebuah Event dibatalkan permanen dan dihapus dari database, semua data pendaftaran (Registrations) dan ulasan (Feedback) untuk acara itu sudah tidak relevan lagi.
   - **Keuntungan:** CASCADE (Beruntun) berarti "jika induknya dihapus, hapus juga semua data anaknya." Ini secara otomatis membersihkan database dari data "sampah" yang sudah tidak memiliki induk, menjaga database tetap bersih, efisien, dan relevan.

IV. Verifikasi Normalisasi (3NF)

Skema SQL ini divalidasi berdasarkan aturan normalisasi yang disyaratkan:

1. **1NF:** Terpenuhi. Semua kolom bersifat atomik (tidak ada *multi-valued attributes*).
2. **2NF:** Terpenuhi. Semua atribut non-kunci (seperti nama_acara, tanggal_registrasi) bergantung pada keseluruhan Primary Key.
3. **3NF:** Terpenuhi. Tidak ada *transitive dependency*. Atribut yang bergantung pada atribut

non-kunci lain telah dipisah.

- **Contoh:** nama_organizer bergantung pada organizer_id (PK-nya), bukan pada event_id. Oleh karena itu, kita membuat tabel Organizers. nama_kategori bergantung pada category_id (PK-nya), bukan pada event_id. Oleh karena itu, kita membuat tabel Categories.

V. Penutup

Laporan ini telah merinci penerjemahan ERD konseptual (Week 1) menjadi skema logis (LRS) dan skrip SQL fungsional (Week 2). Skrip schema.sql yang disediakan telah mencakup semua entitas, atribut, dan relasi yang diidentifikasi pada proposal.

Dengan penerapan tipe data yang sesuai serta batasan (constraints) seperti PRIMARY KEY, FOREIGN KEY, UNIQUE, CHECK, dan DEFAULT, struktur database ini siap untuk menampung data aplikasi "Eventify" dengan integritas dan aturan bisnis yang terjaga.