**LAPORAN SISTEM LABOR NCS**

Laporan ini disusun untuk memenuhi Ujian Akhir Semester

Mata Kuliah Basis Data (G)

****

**Disusun Oleh :**

Kelompok 8

Sandrina Novita Sari (4523210101)

M Akbar Ramadhan (4523210132)

**Adi Wahyu Pribadi, S.Si., M.Kom**

**PROGRAM STUDI TEKNIK INFORMATIKA**

**FAKULTAS TEKNIK**

**UNIVERSITAS PANCASILA**

**2024/2025**

1. **Rancang Database**
2. **Identifikasi entitas utama, atribut, dan relasi antar entitas**

**Setelah melakukan wawancara dengan pengurus Laboratorium Network & Cyber Security dan melihat data asli nya, di dalam database LaborNCS terdapat :**

1. **Entitas Utama**

* **Mahasiswa**

**Atribut:**

**- id\_mhs INT: Nomor unik untuk mengidentifiikasi setiap mahasiswa**

**- nama\_mhs VARCHAR(100): Berisi nama lengkap mahasiswa**

**- npm\_mhs BIGINT: Berisi nomor pokok mahasiswa**

**- semester INT: Berisi semester yang ditempuh mahasiswa saat ini**

**- angkatan\_mhs INT: Berisi angkatan semua mahasiswa**

**- kontak\_mhs BIGINT: Berisi kontak pribadi mahasiswa**

* **Asisten Praktikum**

**Atribut:**

**- id\_asprak INT:Nomor unik untuk mengidentifikasi setiap asisten praktikum**

**- nama\_asprak VARCHAR(100): Berisi nama asisten praktikum**

**- npm\_asprak BIGINT: Berisi nomor pokok asisten praktikum**

**- email\_asprak VARCHAR(100): Berisi email aktif asisten praktikum**

**- kontak\_asprak BIGINT: Berisi kontak pribadi asisten praktikum**

* **Matakuliah**

**Atribut:**

**- id\_matkul INT: Nomor unik untuk mengidentifikasi setiap matakuliah**

**- nama\_matkul VARCHAR(100): Berisi nama matakuliah**

**- sks\_matkul int: Berisi total sks dari suatu matakuliah**

**- dosen\_matkul varchar(100): Berisi nama dosen pengajar dari suatu matakuliah**

* **Ruang Labor**

**Atribut:**

**- id\_labor INT: Nomor unik untuk mengidentifikasi setiap meja di LaborNCS**

**- nomer\_monitor INT: Berisi urutan nomor monitor yang terletak di meja LaborNCS**

**- status\_monitor ENUM: Berisi status yang hanya mencantumkan "Tersedia" dan "Tidak Tersedia"**

**- keterangan TEXT: Berisi keterangan spesifikasi dari meja dan monitor di LaborNCS**

1. **Entitas Relasi**

**Jadwal Kuliah**

**- ide\_jadwal INT: Nomor unik untuk mengidentifikasi setiap jadwal kuliah**

**- hari\_pemakaian VARCHAR(100): Berisi jadwal harian pemakaian pada LaborNCS**

**- tanggal\_pemakaian DATE: Berisi tanggal setiap pemakaian pada LaborNCS**

**- jam\_mulai TIME: Berisi waktu dimulainya praktikum**

**- jam\_selesai TIME: Berisi waktu selesainya praktikum**

1. **Relasi Antar Entitas**

**·  Mahasiswa - Jadwal Kuliah**

**- Relasi: Setiap mahasiswa dapat menerima pembelajaran pada satu jadwal kuliah**

**- Tipe Relasi: *One-to-Many* (Satu mahasiswa dapat memiliki banyak jadwal kuliah)**

**·     Asisten Praktikum - Jadwal Kuliah**

**- Relasi: Setiap asisten praktikum dapat mengajar pada satu jadwal kuliah**

**- Tipe Relasi: *One-to-Many* (Satu asisten praktikum dapat memandu banyak jadwal kuliah)**

**·   Matakuliah - Jadwal Kuliah**

**- Relasi: Setiap matakuliah dapat diajarkan pada satu jadwal kuliah**

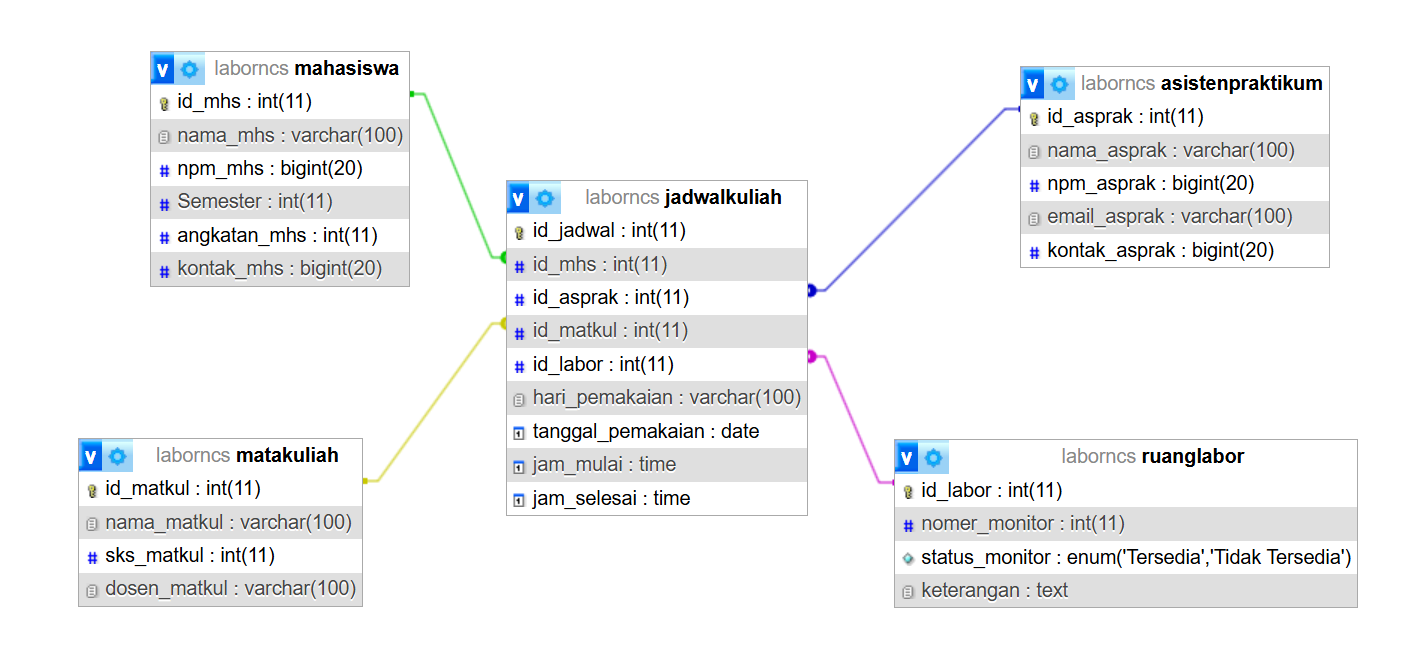
**- Tipe Relasi: *One-to-Many* (Satu matakuliah bisa diajarkan di banyak jadwal kuliah)**

**·   Ruang labor - Jadwal Kuliah**

**- Relasi: Setiap meja laboratorium tersedia dan siap digunakan apabila ada jadwal kuliah**

**- Tipe Relasi: *One-to-Many* (Satu ruang labor bisa digunakan untuk banyak jadwal kuliah)**

1. **ERD untuk menggambarkan struktur Database Labor NCS**

****

ERD diatas merupakan hasil analisis kami yang didapat dari wawancara kami dengan salah satu petugas labor ncs. Jadi sistemnya yaitu di suatu jadwal kuliah terdapat mahasiswa yang sedang belajar sesuai matakuliah yang terjadwal dan diajarkan langsung dengan asisten praktikum di ruang laboratorium yang sudah tersedia fasilitasnya. Dan di jadwal kuliah juga terdapat jam mulai dan jam selesai serta tanggal pemakaian.

1. **Implementasi *Database***
2. **Proses membuat Database, tabel, relasi, hingga *input* data *Update,* dan Delete menggunakan MySQL, InnoDB, dan UTF8.**

|  |  |
| --- | --- |
| 1. **Membuat**   ***Database*** | **Query:**    **Penjelasan:**  Sesuai perintah, kami memakai MySQL buat pembuatan *Database*, serta penyimpanan dengan InnoDB serta UTF8. |
| 1. **Membuat**   **Tabel**  **Mahasiswa** | **Query:**    **Penjelasan:**  Membuat tabel nama Mahasiswa. Tabel ini berisi data tentang mahasiswa, dengan kolom-kolom selaku berikut: id\_mhs selaku primary key ataupun kunci utama yang tidak boleh kosong), nama\_mhs (nama mahasiswa, optimal 100 kepribadian npm\_mhs no Pokok Mahasiswa berbentuk angka besar), semester (semester mahasiswa), angkatan\_mhs (tahun angkatan mahasiswa), serta kontak\_mhs no kontak mahasiswa berbentuk angka besar). Query ini membenarkan kalau tiap mahasiswa mempunyai ID yang unik serta tidak terdapat 2 informasi mahasiswa yang mempunyai ID yang sama. |
| 1. **Membuat**   **Tabel**  **Asisten Praktikum** | **Query:**    **Penjelasan:**  Pada tabel ‘Asistenpraktikum’ id\_asprak selaku *primary key* yang unik serta hendak meningkat otomatis tiap kali informasi baru ditambahkan), nama\_asprak (nama asisten, optimal 100 kepribadian npm\_asprak no Pokok Mahasiswa asisten berbentuk angka besar), email\_asprak (alamat email asisten, optimal 100 kepribadian serta kontak\_asprak no kontak asisten berbentuk angka besar). Guna AUTO\_INCREMENT pada id\_asprak membenarkan nilai ID hendak otomatis meningkat tanpa butuh diisi secara manual. |
| 1. **Membuat Tabel Matakuliah** | **Query:**    **Penjelasan:**  Pada tabel ‘Matakuliah’ pula *primary key*-nya memakai auto\_icrement supaya tidak butuh mengisi ID secara manual, serta atribut nama\_matkul, dosen\_matkul dengan jenis informasi varchar yang wajib diisi/not null. |
| 1. **Membuat Tabel Ruang Labor** | **Query:**    **Penjelasan:**  Membuat tabel Ruang labor dengan atribut id\_labor selaku *primary key*, no monitor, status monitor serta penjelasan dari monitor tersebut |
| 1. **Membuat**   **Tabel Jadwal Kuliah** | **Query:**    **Penjelasan:**  Membuat tabel kedekatan bernama agenda kuliah yang terdiri dari id mahasiswa (tabel mahasiswa), id asisten praktikum (tabel asisten praktikum, id matakuliah (tabel matakuliah), id labor (tabel ruang labor), hari konsumsi bertepatan pada konsumsi jam mulai, jam berakhir. |
| 1. **Menunjuk**   **Kan tabel pada *Database***  **laborncs** | **Query:**    **Penjelasan:**  erintah diatas buat menampilkan apa saja tabel yang terdapat pada *Database* laborncs. |
| 1. ***Insert* values ke tabel Mahasiswa** | **Query:**  INSERT INTO `mahasiswa`(`id\_mhs`, `nama\_mhs`, `npm\_mhs`, `Semester`, `angkatan\_mhs`, `kontak\_mhs`) VALUES  (01,'M Akbar Ramadhan',4523210132,3,2023,083196390884),  (02,'Sandrina Novitasari',4523210101,3,2023,085174271126),  (03,'Fahran Maulana',4523210044,3,2023,081282047377),  (04,'Firdaus Fatan',4523210049,3,2023,081380040356),  (05,'Handra Putra Alma',4523210053,3,2023,085155032911),  (06,'Ramzy Syah Khalifah',4523210089,3,2023,081806694772),  (07,'Hilya Najwa',4523210112,3,2023,082283209672),  (08,'Muhammad Abiyu Muflih',4523210124,3,2023,085767241273),  (09,'Alifia Wenny',4523210126,3,2023,0881024144328),  (10,'Hany Nadya',4523210128,3,2023,081386382210),  (11,'Evelyn Ade Oktalia',4524210031,1,2024,085244910026),  (12,'Kornelius Timoty',4524210050,1,2024,083100196722),  (13,'Ririn Verdawati',4524210090,1,2024,081266610281),  (14,'Syalva Kirania',4524210101,1,2024,085200129910),  (15,'Reno Budiman',4524210124,1,2024,08821567721),  (16,'Moreno Renaldy',4519210115,11,2019,081164555801),  (17,'Hyza Proteus',4520210101,9,2020,08520661910),  (18,'Ken Farabi',4521210002,7,2021,082210899251),  (19,'Vina Aisyah',4522210066,5,2022,081106743821),  (20,'Aulia Zahra Sabila',4522210132,5,2022,081314585984);  **Penjelasan:**  Perintah *INSERT* dicoba buat akumulasi informasi pada tiap fields yang terdapat pada tabel Mahasiswa |
| 1. **Melihat field tabel Maha-siswa** | **Query:**    **Penjelasan:**  Perintah ‘*desc*’ di mari kami pakai buat memandang *fields* apa saja yang terdapat pada tabel Mahasiswa. |
| 1. **Select untuk melihat Insert yang sudah ditambah-kan** | **Query:**    **Penjelasan:**  Perintah ‘*SELECT*’ ini kami gunakan untuk melihat seluruh *Insert* yang sudah ditambahkan sebelumnya. |
| 1. **Insert values ke tabel Asisten**   **praktikum** | **Query:**    **Penjelasan:**  Di sini kami melaksanakan perintah ‘*INSERT*’ buat menginput informasi ke tabel Asistenpraktikum. Kami menginput sebanyak 20 informasi buat masing-masing fields (nama\_asprak, npm\_asprak, email\_asprak, kontak\_asprak). |
| 1. **Melihat *fields* tabel Asisten praktikum** | **Query:**    **Penjelasan:**  Perintah untuk melihat apa saja *field* yang ada pada tabel Asistenpraktikum. |
| 1. **Select untuk melihat Insert yang sudah ditambah-**   **kan** | **Query:**    **Penjelasan:**  Perintah yang digunakan untuk melihat data apa saja yang sudah berhasil diinput setelah melakukan perintah ‘*INSERT*’ sebelumnya ke dalam tabel Asistenpraktikum. |
| 1. **Insert values ke tabel**   **Matakuli-ah** | **Query:**    **Penjelasan:**  *Query* untuk menambahkan/menginput data ke dalam tabel Transport sesuai *fields*nya. |
| 1. **Melihat *fields* tabel**   **Matakuli-ah** | **Query:**    **Penjelasan:**  *Desc* digunakan untuk melihat deskripsi tabel Matakuliah atau melihat isi *fields* dari tabel tersebut. |
| 1. **Select untuk melihat *Insert* yang sudah ditambah-kan** | **Query:**    **Penjelasan:**  Untuk melihat data yang sudah diinput ke dalam tabel Matakuliah. |
| 1. **Insert values ke tabel**   **Jadwal**  **kuliah** | **Penjelasan :**  Menambahkan data pada tabel jadwal kuliah sebanyak 20 row. |
| 1. **Melihat *fields* tabel Jadwal**   **kuliah** | **Penjelasan :**  Menampilkan isi tabel dari jadwal kuliah yang dimana tabel ini adalah tabel relasi dari para tabel entitas utama. |
| 1. ***Select* untuk melihat *Insert* yang sudah ditambah-kan** | **Penjelasan :**  Perintah select bertujuan untuk menampilkan isi row pada tabel jadwal kuliah. |
| 1. ***Update* pada tabel Ruang Labor** | **Tabel :**    **Penjelasan :**  Perintah Update pada tabel Ruang Labor untuk memperbarui keterangan pada id labor 103 yang sebelumnya belum terinstal semua aplikasi menjadi hanya tersedia Visual Studio Code. |
| 1. ***Delete* untuk melihat *Insert* yang sudah ditambah-kan** | **Query :**    **Penjelasan :**  Perintah untuk menghapus row dengan memakai 2 kondisi yang dimana id\_matkul > 19 AND nama\_matkul = “Algoritma Pemrograman”. |

1. **Implementasi konsep *Transaction***
2. **Atomicity manual transaksi sukses**

* Transaksi Sukses

Query:

**START TRANSACTION;**

**INSERT INTO JadwalKuliah (id\_jadwal, id\_mhs, id\_asprak, id\_matkul, id\_labor, hari\_pemakaian, tanggal\_pemakaian, jam\_mulai, jam\_selesai) VALUES**

**(21, 3, 1, 105, 109, 'Rabu', '2024-03-06', '08:00', '10:30');**

**INSERT INTO JadwalKuliah (id\_jadwal, id\_mhs, id\_asprak, id\_matkul, id\_labor, hari\_pemakaian, tanggal\_pemakaian, jam\_mulai, jam\_selesai) VALUES**

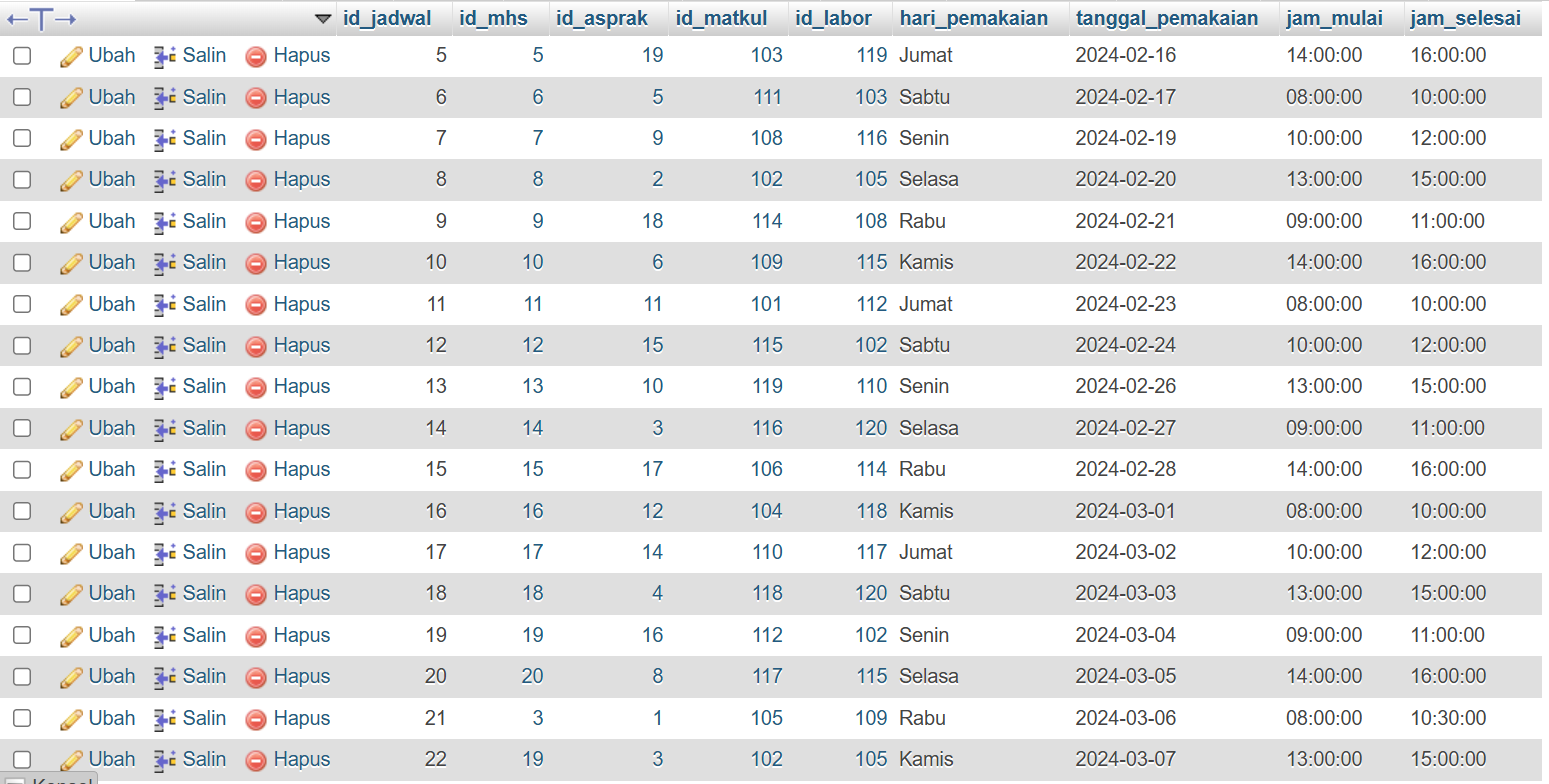
**(22, 19, 3, 102, 105, 'Kamis', '2024-03-07', '13:00', '15:00');**

**COMMIT;**

Penjelasan:

Berawal dengan memulai transaksi database dengan perintah *Start Transaction*. Selanjutnya kita menambahkan 2 data baru yang akan dimasukkan ke dalam tabel Jadwal Kuliah dan di akhiri dengan *commit* untuk menyimpan permanen ke dalam tabel Jadwal Kuliah. Cara kerja dari *atomaticity* ialah kedua operasi insert dianggap sebagai satu unit. Jadi, jika kedua pernyataan insert berhasil dijalankan tanpa kesalahan, perubahan akan di *commit* dan data akan ditambahkan secara permanen ke tabel Jadwal Kuliah. Akan tetapi, jika terjadi kesalahan pada salah satu pernyataan insert, seluruh transaksi akan dibatalkan, dan tidak ada perubahan yang diterapkan pada tabel.

Bukti :



* Transaksi Rollback

Query:

**START TRANSACTION;**

**DELETE FROM JadwalKuliah WHERE id\_jadwal = 21;**

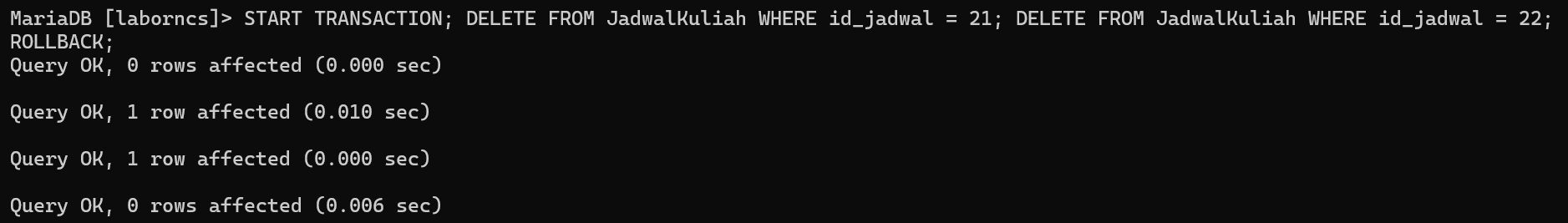
**DELETE FROM JadwalKuliah WHERE id\_jadwal = 22;**

**ROLLBACK;**

Penjelasan:

Berawal dengan memulai transaksi database dengan perintah *Start Transaction*. Selanjutnya kita menghapus 2 data dengan menggunakan perintah Delete. Dan diakhir memakai rollback. Cara kerja dari *atomaticity* ini berbeda dengan sebelumnya, karena data yang terhapus hanya bersifat sementara dan langsung dibatalkan dengan adanya perintah *Rollback* sehingga perintah delete langsung dibatalkan*.*

Bukti :



**2)**    **Consistency *Invalid Upda*te pada Labor NCS**

Query:

**START TRANSACTION;**

**UPDATE jadwalkuliah SET id\_matkul = NULL**

**WHERE id\_mhs = 1;**

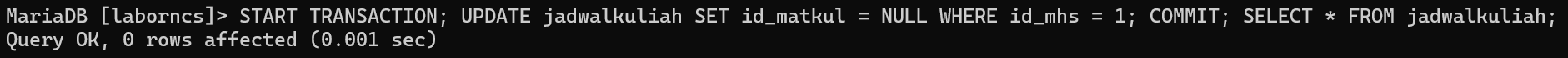
**COMMIT;**

**SELECT \* FROM jadwalkuliah;**

Penjelasan:

Berawal dengan memulai transaksi database dengan perintah *Start Transaction*. Selanjutnya memperbarui tabel jadwalkuliah dengan mengubah jam mulai dan jam selesai. Dikarenakan perintah UPDATE nya inkonsistensi yang dimana id\_matkul tidak boleh NULL, meski di commit tetap tidak akan mengubah id\_matkul menjadi NULL.

Bukti:



**3)**    ***Isolation* (isolasi antar transaksi pada Labor NCS)**

Query:

**START TRANSACTION;**

**UPDATE jadwalkuliah**

**SET jam\_mulai = ADDTIME(jam\_mulai, '01:00:00') WHERE id\_jadwal = 21;**

**SELECT \* FROM jadwalkuliah WHERE id\_jadwal = 21 FOR UPDATE;**

**UPDATE jadwalkuliah**

**SET jam\_selesai = ADDTIME(jam\_selesai, '01:30:00') WHERE id\_jadwal = 21;**

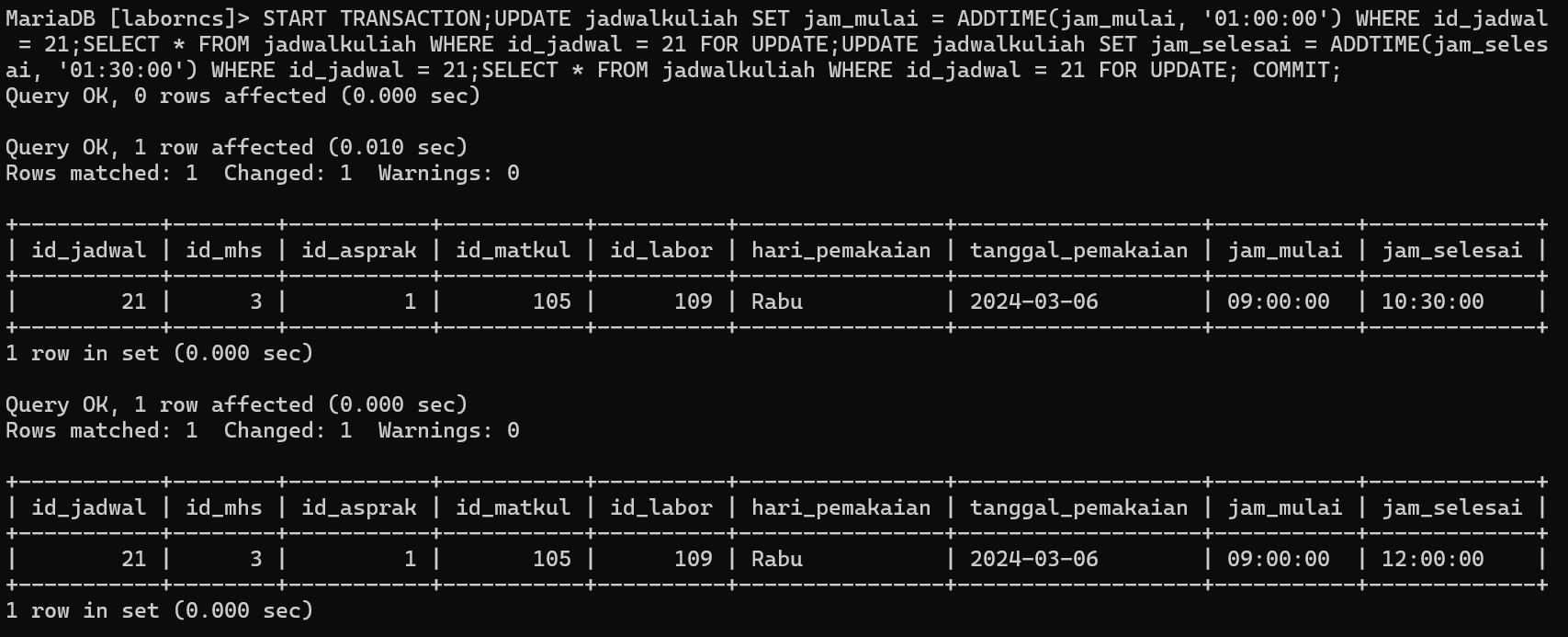
**SELECT \* FROM jadwalkuliah WHERE id\_jadwal = 21 FOR UPDATE;**

**COMMIT;**

Penjelasan:

Pada prinsip isolation yang dimana harus transaksi secara bersamaan namun berurutan, kami memperbarui jam mulai dan jam selesai pada tabel jadwalkuliah. Meskipun memakai perintah UPDATE secara bersamaan, namun proses mengupdate transaksinya secara berurutan.

Bukti:



**C.**    **Buatkan skenario apabila ada kesalahan maka seluruh transaksi batal**

Query:

**START TRANSACTION;**

**DECLARE EXIT HANDLER FOR SQLEXCEPTION**

**BEGIN**

**ROLLBACK;**

**SELECT** **'Tidak dapat diperbarui.' AS Error\_Message;**

**END;**

**UPDATE jadwalkuliah SET jam\_selesai = (jam\_selesai, '01:30:00') WHERE id\_jadwal = 22;**

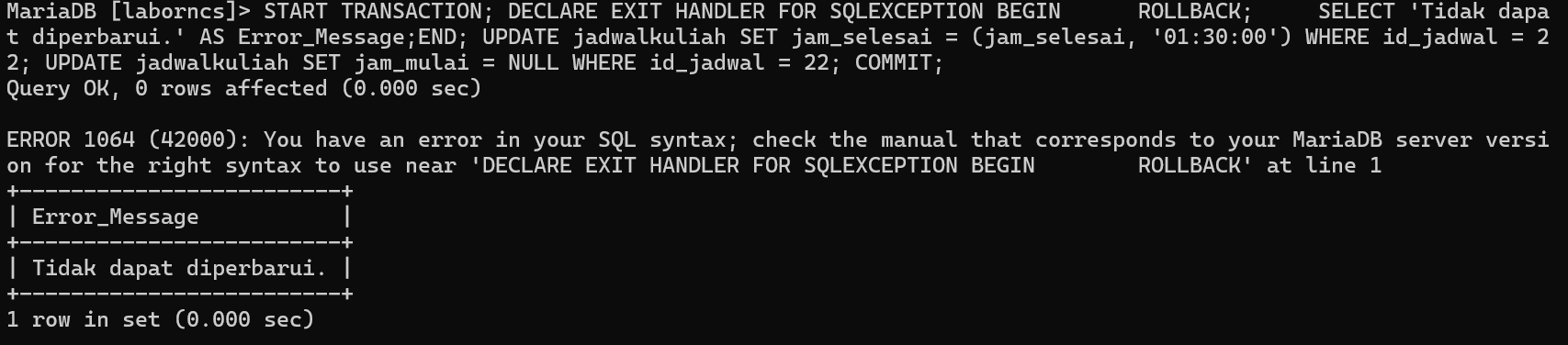
**UPDATE jadwalkuliah SET jam\_mulai = NULL WHERE id\_jadwal = 22;**

**COMMIT;**

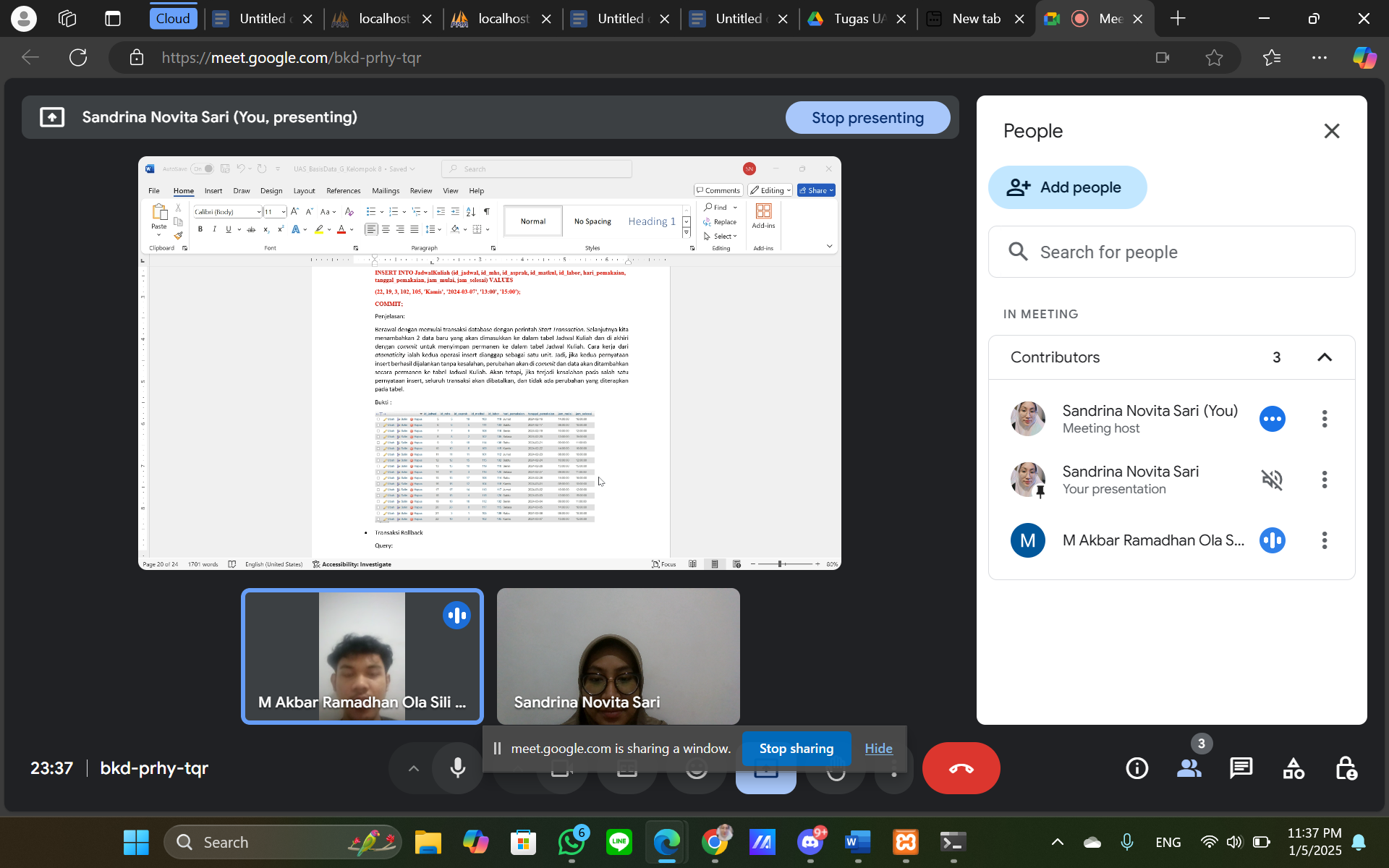
Penjelasan:

Jadi skenario di atas adalah contoh transaksi dengan penggunaan *Rollback* yang berperan sebagai mekanisme penanganan error (error handling) yang otomatis dijalankan apabila salah satu perintah gagal. Jadi di skenario diatas kami membuat perintah yang ada kesalahannya sehingga dengan begitu, perubahan yang setengah jalan atau tidak lengkap tidak akan disimpan ke database meski sudah di *commit*, supaya data tetap aman dan konsisten.

Bukti:



1. **Dokumentasi Pengerjaan**



1. **Link Github**

[**https://github.com/akbarramadhn/LaborNCS**](https://github.com/akbarramadhn/LaborNCS)