

# LAPORAN PROYEK KELOMPOK 5

## SISTEM BASIS DATA

---

*“Sistem Akademik Sederhana”*



No	NIM	Nama	Prodi
1.	11323010	Franklyn Aldo Ignatia Lumbantoruan	D3TI
2.	11323011	Ferry Bastian Siagian	D3TI
3.	11323005	Abeloisa Chelsea Pardosi	D3TI
4.	11323047	Agnes Elyestra Sidabutar	D3TI

**INSTITUT TEKNOLOGI DEL**  
**FAKULTAS VOKASI 2024/2025**

# Daftar Isi

Bab 1 Pendahuluan .....	3
1.1. Latar Belakang .....	3
Bab 2. Rancangan dan Metode .....	4
2.1 Pengumpulan Data .....	4
Bab 3. Implementasi .....	5
3.2. Create Table .....	6
3.3. DUMMY DATA.....	7
3.4. JOIN .....	8
3.5. VIEW .....	9
3.6. TRIGGER .....	12
3.7. Authorization .....	12
3.7. TRANSACTION .....	13
3.8. STORED PROCEDURE .....	14
3.9. CURSOR.....	15
3.10. FITUR UTAMA .....	16
3.11 Backup & Restore .....	19
3.11.1. Backup .....	19
3.11.2. Restore .....	19
3.12. Crawling dan Scraping.....	20
3.12.1. Crawling.....	20
3.12.2. Scraping .....	24

## **Bab 1 Pendahuluan**

### **1.1. Latar Belakang**

Sistem akademik adalah elemen dalam mendukung operasional institusi pendidikan, khususnya dalam mengelola berbagai data yang kompleks seperti jadwal perkuliahan, data mahasiswa, dosen, mata kuliah, serta evaluasi akademik. Pembuatan mini proyek sistem akademik sederhana menggunakan PostgreSQL bertujuan untuk memberikan solusi yang efisien dan terstruktur dalam pengelolaan data akademik. PostgreSQL dipilih sebagai basis utama sistem ini karena keandalannya sebagai sistem manajemen basis data relasional open-source yang mendukung fitur-fitur canggih seperti transaksi ACID (Atomicity, Consistency, Isolation, Durability), keamanan tingkat tinggi, dan kemampuan menangani volume data yang besar. Sistem ini diharapkan dapat membantu institusi pendidikan, khususnya yang memiliki keterbatasan sumber daya, dalam mengotomatiskan proses administrasi, meminimalkan kesalahan manual, dan meningkatkan efisiensi kerja.

Proyek ini dirancang untuk menjadi sederhana namun mencakup fungsi-fungsi dasar yang esensial, seperti pencatatan data mahasiswa, nilai, mata kuliah, serta penyesuaian jadwal mata kuliah dari google kalender menggunakan API. Salah satu fitur dalam proyek ini dapat menampilkan IPK mahasiswa berdasarkan dengan nilai yang di input. Proyek ini juga dirancang untuk memberikan pengalaman langsung kepada mahasiswa dalam mengembangkan sistem basis data relasional, termasuk desain skema database, integrasi data, dan implementasi query SQL untuk kebutuhan akademik. Lebih jauh lagi, mini proyek ini tidak hanya berfungsi untuk memenuhi tugas proyek mata kuliah Sistem Basis Data tetapi juga berfungsi sebagai studi kasus yang dapat dikembangkan lebih lanjut menjadi sistem yang lebih kompleks dan terintegrasi. demikian, proyek ini sebagai media pembelajaran bagi mahasiswa untuk memahami bagaimana membangun dan mengimplementasikan sistem berbasis database yang praktis dan fungsional.

## **Bab 2. Rancangan dan Metode**

### **2.1 Pengumpulan Data**

Langkah awal yang dilakukan dalam pengerjaan proyek ini adalah menganalisis system akademik yang terjadi di dalam CIS. Dengan meninjau serta website CIS dan bertanya kepada Teaching Assiten, dan Dosen kami mendapatkan arahan dan pandangan untuk mengerjakan proyek ini.

### **2.2 Tools Yang Digunakan**

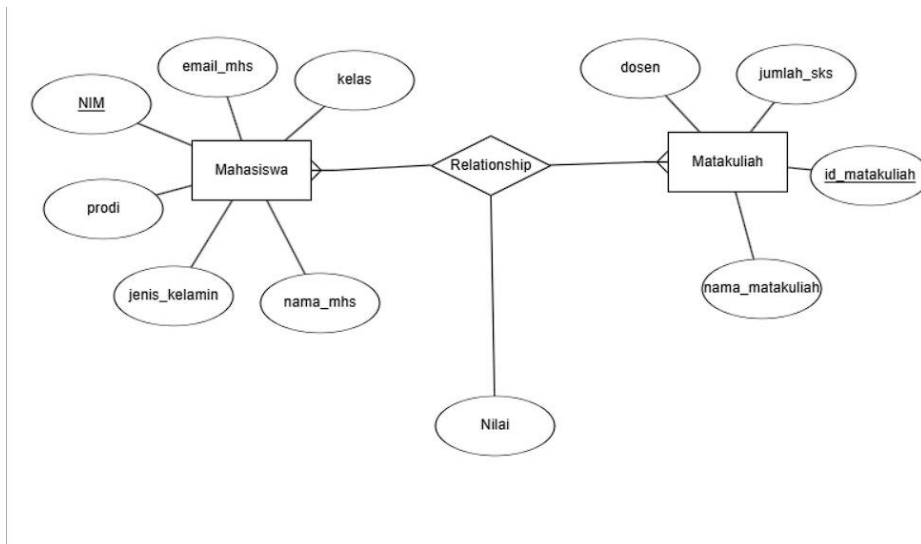
Pada pengerjaan proyek ini ada beberapa tools yang digunakan yaitu :

- ERD plus: Tools ini digunakan untuk merancang Entity-Relationship Diagram (ERD) pada sistem akademik ini. Dengan menggunakan ERD Plus, pengembang dapat membuat representasi visual dari struktur database yang mencakup entitas, atribut, dan relasi antar entitas. Hal ini membantu dalam memahami alur data, mengidentifikasi hubungan penting, dan memastikan desain database yang optimal sebelum implementasi.
- Pg Admin : Merupakan tools manajemen grafis untuk PostgreSQL yang digunakan untuk mengelola database pada sistem akademik ini. Dengan PgAdmin, pengembang dapat membuat, memodifikasi, dan memonitor database
- Google colab: Platform ini digunakan untuk menulis dan menjalankan kode Python yang terintegrasi dengan database PostgreSQL dan layanan API Google. Google Colab mempermudah pengembangan sistem akademik karena memungkinkan kolaborasi langsung, akses dari berbagai perangkat, serta mendukung integrasi pustaka-pustaka Python yang diperlukan. Dalam system ini Google colab digunakan untuk melakukan Scraping dan Crawling jadwal Matakuliah
- Google API : Layanan ini digunakan untuk mengintegrasikan fitur tambahan seperti Google Calendar ke dalam sistem akademik. Google API memungkinkan sistem untuk menambahkan jadwal perkuliahan secara otomatis ke Google Calendar, memberikan pengalaman pengguna yang lebih modern dan terintegrasi. Dengan fitur ini, jadwal yang diinput di sistem akademik dapat disinkronkan langsung ke Google Calenda
- Trello : : Trello digunakan untuk mengelola dan mengakomodasi pengerjaan proyek sistem akademik. Dengan Trello, mahasiswa dapat membuat jadwal dan daftar tugas yang terstruktur, seperti "To Do" (apa yang akan dikerjakan), "In Progress" (tugas yang sedang dikerjakan), dan "Done" (tugas yang telah selesai). Selain itu, Trello memungkinkan pembagian tugas yang jelas sesuai dengan peran dan tanggung jawab masing-masing anggota tim. Trello juga dapat diintegrasikan dengan berbagai layanan seperti GitHub untuk manajemen versi kode atau Google Drive untuk menyimpan dokumen dan hasil kerja.
- GitHub: GitHub digunakan sebagai platform manajemen kolaborasi untuk proyek sistem akademik ini. Dengan GitHub, mahasiswa dapat menyimpan, mengelola, dan melacak perubahan kode yang dibuat selama pengembangan sistem. Fitur version control di GitHub memungkinkan untuk melihat riwayat perubahan, mengidentifikasi siapa yang membuat perubahan tertentu, dan memulihkan versi sebelumnya jika diperlukan

### Bab 3. Implementasi

Pada pengerjaan proyek ini dilakukan beberapa implementasi yang terkait dengan materi Sistem Basis Data yaitu :

#### 3.1. Entity Relationship Diagram (ERD)



*Gambar 1 ERD Sistem akademik Sederhana*

Gambar ini merupakan struktur yang menggambarkan bagaimana struktur data untuk system akademik sederhana yang akan dibuat, dimana pada gambar tersebut terdapat 2 entitas yaitu :

1. **Entitas Mahasiswa** dengan atribut :

- NIM (Nomor Induk Mahasiswa)
- nama\_mhs: Nama mahasiswa.
- jenis\_kelamin: Jenis kelamin mahasiswa.
- prodi: Program studi yang diambil oleh mahasiswa.
- email\_mhs: Email resmi mahasiswa.
- kelas: Kelas atau kelompok belajar mahasiswa

2. **Entitas Matakuliah** dengan atribut :

- id\_matakuliah: Identitas unik untuk setiap mata kuliah.
- nama\_matakuliah: Nama mata kuliah.
- jumlah\_sks: Jumlah Satuan Kredit Semester (SKS) untuk mata kuliah.
- dosen: Nama dosen pengajar mata kuliah

pada gambar tersebut terdapat relasi menghubungkan **Mahasiswa** dengan **Matakuliah**. Ini menunjukkan bahwa setiap mahasiswa dapat mengambil beberapa mata kuliah, dan setiap mata kuliah dapat diikuti oleh banyak mahasiswa (hubungan **many-to-many**). Pada relasi terdapat atribut yaitu nilai karena merupakan informasi tambahan yang terkait dengan hubungan antara **Mahasiswa** dan **Matakuliah**

## 3.2. Create Table

### a. table mahasiswa

```
24 CREATE TABLE mahasiswa (  
25     NIM VARCHAR(10) PRIMARY KEY,  
26     nama_mhs VARCHAR(100),  
27     prodi VARCHAR(50),  
28     jenis_kelamin CHAR(1) CHECK (jenis_kelamin IN ('L', 'P')),  
29     email_mhs VARCHAR(100),  
30     kelas VARCHAR(10)  
31 );
```

*Gambar 2. Create Tabel Mahasiswa*

Query ini digunakan untuk membuat table mahasiswa dimana pada table ini sudah di set nim sebagai primary key nya, pada table ini juga memiliki atribut lain nya seperti nama\_mhs, prodi, jenis kelamin, email dan kelas sesuai dengan tipe data nya masing masing. Pada table ini juga terdapat method check untuk mengecek jenis kelamin dari data mahasiswa yang akan di insert nanti

### b. Tabel matakuliah

```
CREATE TABLE matakuliah (  
    id_matakuliah VARCHAR(10) PRIMARY KEY,  
    nama_matakuliah VARCHAR(100),  
    jumlah_sks INT CHECK (jumlah_sks > 0),  
    dosen VARCHAR(100)  
);
```

*Gambar 3. Create Tabel Matakuliah*

Query ini menambahkan data ke tabel matakuliah yang mencakup ID mata kuliah (primary key), nama mata kuliah, jumlah SKS, dan dosen pengampu beserta tipe data nya masing masing. Pada table ini juga terjadi pengecekan dimana pada saat memasukkan data nya nanti jumlah sks harus lebih dari 0

### c. table nilai

```
CREATE TABLE nilai (  
    nilai NUMERIC(10, 2) CHECK (nilai >= 0 AND nilai <= 100),  
    NIM VARCHAR(10),  
    id_matakuliah VARCHAR(10),  
    Foreign Key (NIM) references mahasiswa(NIM) ON DELETE CASCADE,  
    foreign key (id_matakuliah) references matakuliah(id_matakuliah) ON DELETE CASCADE  
);
```

Query ini memasukkan data ke tabel nilai yang menghubungkan mahasiswa (melalui NIM) dan mata kuliah (melalui id\_matakuliah) dengan nilai akhir. Nilai yang dimasukkan dibatasi antara 0 hingga 100.

### 3.3. DUMMY DATA

#### a. Tabel mahasiswa

```
50 INSERT INTO mahasiswa (NIM, nama_mhs, prodi, jenis_kelamin, email_mhs, kelas)
51 VALUES
52 ('11323010', 'Franklyn Lumbantoruan', 'Teknologi Informasi', 'L', 'franklyn@itdel.ac.id', '32TI1'),
53 ('11323011', 'Ferry Siagian', 'Teknologi Informasi', 'L', 'ferry@itdel.ac.id', '32TI1'),
54 ('11323047', 'Agnes Sidabutar', 'Teknologi Informasi', 'P', 'agnes@itdel.ac.id', '32TI2'),
55 ('11323005', 'Abeloisa Pardosi', 'Teknologi Informasi', 'P', 'abeloisa@itdel.ac.id', '32TI1'),
56 ('12513005', 'Mabeloisa Manurung', 'Sistem Informasi', 'L', 'mabeloisa@itdel.ac.id', '32SI1'),
57 ('14523001', 'Aan Kiki Siahaan', 'Teknik Elektro', 'L', 'aankiki@itdel.ac.id', '32TE1'),
58 ('11423040', 'Noel Terompet Sitingjak', 'Teknologi Rekayasa Perangkat Lunak', 'L', 'terompet@itdel.ac.id', '42TRPL1'),
59 ('11523019', 'Hanni Barus', 'Informatika', 'P', 'hanni@itdel.ac.id', '32IF1'),
60 ('21523030', 'Ryka Simbolon', 'Manajemen Rekayasa', 'P', 'ryka@itdel.ac.id', '32MR2'),
61 ('13323045', 'Ripi Gramaldy Hugo', 'Teknologi Komputer', 'L', 'disinihugo@itdel.ac.id', '32TK2'),
62 ('12513006', 'Monica Ginting', 'Sistem Informasi', 'P', 'monica@itdel.ac.id', '32SI1'),
63 ('12513060', 'Erick Sitohang', 'Sistem Informasi', 'L', 'erick@itdel.ac.id', '32SI2'),
64 ('14523002', 'Robert Sinaga', 'Teknik Elektro', 'L', 'robert@itdel.ac.id', '32TE1'),
65 ('14523050', 'Yohana Samosir', 'Teknik Elektro', 'P', 'yohana@itdel.ac.id', '32TE2'),
66 ('11423041', 'Julius Hutabarat', 'Teknologi Rekayasa Perangkat Lunak', 'L', 'julius@itdel.ac.id', '42TRPL1'),
67 ('11423042', 'Merry Marpaung', 'Teknologi Rekayasa Perangkat Lunak', 'P', 'merry@itdel.ac.id', '42TRPL2'),
68 ('11523020', 'Joshua Sitanggang', 'Informatika', 'L', 'joshua@itdel.ac.id', '32IF1'),
69 ('11523021', 'Clara Nababan', 'Informatika', 'P', 'clara@itdel.ac.id', '32IF2'),
70 ('21523031', 'Samuel Siahaan', 'Manajemen Rekayasa', 'L', 'samuel@itdel.ac.id', '32MR1'),
71 ('21523032', 'Gabriela Simarmata', 'Manajemen Rekayasa', 'P', 'gabriela@itdel.ac.id', '32MR2'),
72 ('13323046', 'Budi Pardede', 'Teknologi Komputer', 'L', 'budi@itdel.ac.id', '32TK2'),
73 ('13323041', 'Siska Manurung', 'Teknologi Komputer', 'P', 'siska@itdel.ac.id', '32TK1');
```

Gambar 4. Dummy data Mahasiswa

Query ini digunakan untuk memasukkan Dummy data pada tabel mahasiswa sesuai dengan atribut dari table mahasiswa yang terbuat tadi dimana terdiri dari 22 entri mahasiswa dengan kolom NIM, nama\_mhs, prodi, jenis\_kelamin, email\_mhs, dan kelas.

#### b. Tabel matakuliah

```
75 INSERT INTO matakuliah (id_matakuliah, nama_matakuliah, jumlah_sks, dosen)
76 VALUES
77 ('MK001', 'Basis Data', 3, 'Dr. Chintya'),
78 ('MK002', 'Dasar Pemrograman', 3, 'Dr. Rudy Manurung'),
79 ('MK003', 'Matematika Diskrit', 3, 'Dr. Pusi Simarmata'),
80 ('MK004', 'Jaringan Komputer', 3, 'Dr. Siringo-ringo'),
81 ('MK005', 'Sistem Operasi', 2, 'Dr. Janri Simbolon'),
82 ('MK006', 'Algoritma dan Pemrograman', 3, 'Dr. Indra Lumbantoruan'),
83 ('MK007', 'Kecerdasan Buatan', 2, 'Dr. Manalu');
```

Gambar 5. Dummy Data Matakuliah

Query ini digunakan untuk memasukkan Dummy data pada tabel matakuliah sesuai dengan atribut dari table matakuliah yang dibuat tadi dimana mencakup 7 entri mata kuliah dengan kolom id\_matakuliah, nama\_matakuliah, jumlah\_sks, dan dosen. Setiap mata kuliah memiliki ID unik seperti 'MK001' untuk Basis Data atau 'MK007' untuk Kecerdasan Buatan. Data ini mencakup jumlah SKS (2 atau 3 SKS) yang menunjukkan bobot kredit akademik mata kuliah tersebut serta nama dosen pengampu, seperti Dr. Chintya untuk Basis Data dan Dr. Manalu untuk Kecerdasan Buatan.

### c. Tabel nilai

```
85 INSERT INTO nilai (nilai, NIM, id_matakuliah)
86 VALUES
87 (85, '11323010', 'MK001'),
88 (90, '11323010', 'MK002'),
89 (78, '11323011', 'MK001'),
90 (82, '11323011', 'MK003'),
91 (88, '11323047', 'MK002'),
92 (76, '11323005', 'MK003'),
93 (89, '12S13005', 'MK003'),
94 (75, '12S13006', 'MK004'),
95 (88, '12S13060', 'MK006'),
96 (80, '14S23001', 'MK004'),
97 (78, '14S23002', 'MK006'),
98 (85, '14S23050', 'MK007'),
99 (93, '11423040', 'MK006'),
100 (90, '11423041', 'MK005'),
101 (88, '11423042', 'MK004'),
102 (95, '11S23019', 'MK007'),
103 (89, '11S23020', 'MK001'),
104 (84, '11S23021', 'MK002'),
105 (77, '21S23030', 'MK003'),
106 (88, '21S23031', 'MK004'),
107 (92, '21S23032', 'MK005'),
108 (81, '13323045', 'MK001'),
109 (87, '13323046', 'MK002'),
110 (79, '13323041', 'MK003');
```

Gambar 6. Dummy data tabel nilai

Query ini digunakan untuk memasukkan Dummy data pada tabel nilai sesuai dengan atribut dari tabel nilai yang terbuat tadi dimana terdiri dari 24 entri yang mencatat nilai akademik mahasiswa dalam mata kuliah tertentu. Kolom nilai berisi angka 0–100, menunjukkan performa mahasiswa dalam mata kuliah terkait. Setiap baris dihubungkan dengan NIM dari tabel mahasiswa dan id\_matakuliah dari tabel matakuliah sebagai foreign key. Contohnya, mahasiswa dengan NIM '11323010' memperoleh nilai 85 pada mata kuliah Basis Data (MK001) dan 90 pada Dasar Pemrograman (MK002). Data ini menghubungkan performa akademik dengan mahasiswa dan mata kuliah tertentu

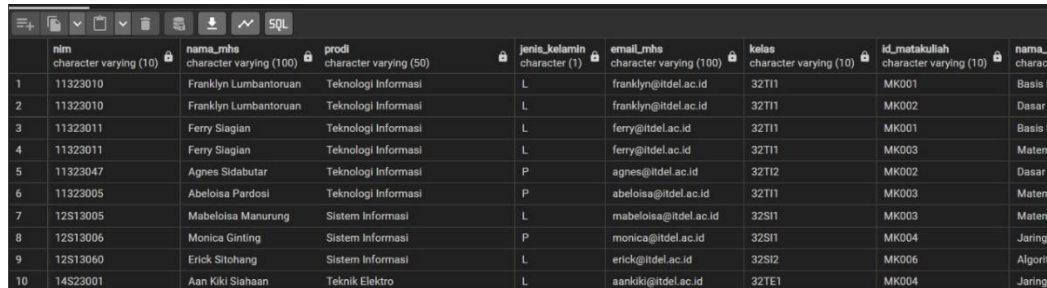
### 3.4. JOIN

```
117 -- join menampilkan semua data dari ketiga tabel --
118 SELECT
119     mhs.NIM,
120     mhs.nama_mhs,
121     mhs.prodi,
122     mhs.jenis_kelamin,
123     mhs.email_mhs,
124     mhs.kelas,
125     mk.id_matakuliah,
126     mk.nama_matakuliah,
127     mk.jumlah_sks,
128     mk.dosen,
129     nl.nilai
130 FROM
131     mahasiswa mhs
132 LEFT JOIN
133     nilai nl
134 ON
135     mhs.NIM = nl.NIM
136 LEFT JOIN
137     matakuliah mk
138 ON
139     nl.id_matakuliah = mk.id_matakuliah;
```

Gambar 7. Join antara tabel



Query di atas merupakan operasi **LEFT JOIN** untuk menggabungkan tiga tabel, yaitu mahasiswa, nilai, dan matakuliah. Query ini bertujuan untuk menampilkan semua data dari tabel mahasiswa serta data yang relevan dari tabel nilai dan matakuliah. Pada query ini juga digunakan alias untuk melakukan join agar tidak terjadi kekeliruan. Ketika menggabungkan tabel.



	nim character varying (10)	nama_mhs character varying (100)	prodi character varying (50)	jenis_kelamin character (1)	email_mhs character varying (100)	kelas character varying (10)	id_matakuliah character varying (10)	nama_m character varying (100)
1	11323010	Franklyn Lumbantoruan	Teknologi Informasi	L	franklyn@itdel.ac.id	32T11	MK001	Basis I
2	11323010	Franklyn Lumbantoruan	Teknologi Informasi	L	franklyn@itdel.ac.id	32T11	MK002	Dasar I
3	11323011	Ferry Siagian	Teknologi Informasi	L	ferry@itdel.ac.id	32T11	MK001	Basis I
4	11323011	Ferry Siagian	Teknologi Informasi	L	ferry@itdel.ac.id	32T11	MK003	Matem
5	11323047	Agnes Sidabutar	Teknologi Informasi	P	agnes@itdel.ac.id	32T12	MK002	Dasar I
6	11323005	Abeloisa Pardosi	Teknologi Informasi	P	abeloisa@itdel.ac.id	32T11	MK003	Matem
7	12S13005	Mabeloisa Manurung	Sistem Informasi	L	mabeloisa@itdel.ac.id	32S11	MK003	Matem
8	12S13006	Monica Ginting	Sistem Informasi	P	monica@itdel.ac.id	32S11	MK004	Jaring
9	12S13060	Erick Sitohang	Sistem Informasi	L	erick@itdel.ac.id	32S12	MK006	Algorit
10	14S23001	Aan Kiki Siahaan	Teknik Elektro	L	aankiki@itdel.ac.id	32TE1	MK004	Jaring

Gambar 8. Output Join

### 3.5. VIEW

#### a. View mahasiswa

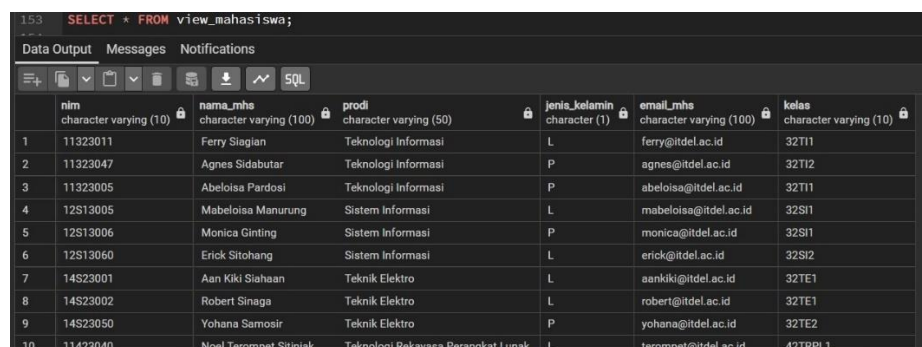
```

142 -- 1. membuat view untuk menampilkan data mahasiswa --
143 CREATE VIEW view_mahasiswa AS
144 SELECT
145     NIM,
146     nama_mhs,
147     prodi,
148     jenis_kelamin,
149     email_mhs,
150     kelas
151 FROM
152     mahasiswa;
153 SELECT * FROM view_mahasiswa;

```

Gambar 9. View Mahasiswa

View ini digunakan untuk menampilkan data mahasiswa secara sederhana. Data yang ditampilkan adalah atribut penting seperti NIM, nama\_mhs, prodi, jenis\_kelamin, email\_mhs, dan kelas dari tabel mahasiswa. View ini memudahkan akses data mahasiswa tanpa harus mengakses seluruh tabel mahasiswa, sehingga lebih efisien untuk operasi tertentu, seperti laporan. Select View mahasiswa yang dibuat maka akan tampil



	nim character varying (10)	nama_mhs character varying (100)	prodi character varying (50)	jenis_kelamin character (1)	email_mhs character varying (100)	kelas character varying (10)
1	11323011	Ferry Siagian	Teknologi Informasi	L	ferry@itdel.ac.id	32T11
2	11323047	Agnes Sidabutar	Teknologi Informasi	P	agnes@itdel.ac.id	32T12
3	11323005	Abeloisa Pardosi	Teknologi Informasi	P	abeloisa@itdel.ac.id	32T11
4	12S13005	Mabeloisa Manurung	Sistem Informasi	L	mabeloisa@itdel.ac.id	32S11
5	12S13006	Monica Ginting	Sistem Informasi	P	monica@itdel.ac.id	32S11
6	12S13060	Erick Sitohang	Sistem Informasi	L	erick@itdel.ac.id	32S12
7	14S23001	Aan Kiki Siahaan	Teknik Elektro	L	aankiki@itdel.ac.id	32TE1
8	14S23002	Robert Sinaga	Teknik Elektro	L	robert@itdel.ac.id	32TE1
9	14S23050	Yohana Samsir	Teknik Elektro	P	yohana@itdel.ac.id	32TE2
10	11423040	Noel Terompet Sitinjak	Teknologi Rekayasa Peranakat Lunak	L	terompet@itdel.ac.id	42TRPL1

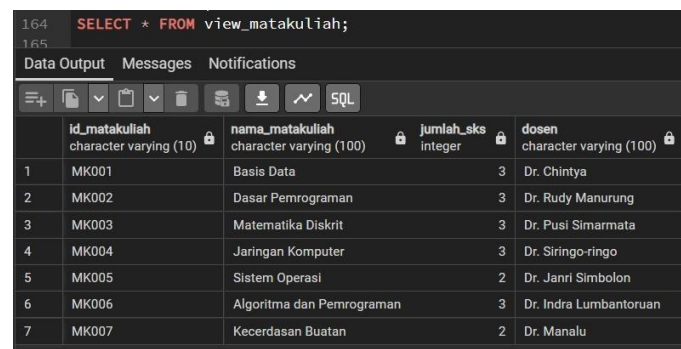
Gambar 10. Output View Mahasiswa

## b. View matakuliah

```
155 -- 2. Menampilkan data mata kuliah
156 CREATE VIEW view_matakuliah AS
157 SELECT
158     id_matakuliah,
159     nama_matakuliah,
160     jumlah_sks,
161     dosen
162 FROM
163     matakuliah;
164 SELECT * FROM view_matakuliah;
165
```

Gambar 11. View Matakuliah

View ini digunakan untuk menampilkan data mata kuliah. Informasi yang disediakan meliputi id\_matakuliah, nama\_matakuliah, jumlah\_sks, dan nama dosen. Dengan view ini, pengguna dapat melihat daftar mata kuliah beserta detailnya secara langsung tanpa harus melakukan query dari tabel matakuliah setiap saat.



	id_matakuliah character varying (10)	nama_matakuliah character varying (100)	jumlah_sks integer	dosen character varying (100)
1	MK001	Basis Data	3	Dr. Chintya
2	MK002	Dasar Pemrograman	3	Dr. Rudy Manurung
3	MK003	Matematika Diskrit	3	Dr. Pusi Simarmata
4	MK004	Jaringan Komputer	3	Dr. Siringo-ringo
5	MK005	Sistem Operasi	2	Dr. Janri Simbolon
6	MK006	Algoritma dan Pemrograman	3	Dr. Indra Lumbantoruan
7	MK007	Kecerdasan Buatan	2	Dr. Manalu

Gambar 12. Output View Matakuliah

## c. View nilai\_mahasiswa

```
166 -- 3. Menampilkan Nilai mahasiswa --
167 CREATE VIEW view_nilai_mhs AS
168 SELECT
169     n.NIM, m.nama_mhs, mk.nama_matakuliah, n.nilai
170 FROM nilai n JOIN mahasiswa m ON n.NIM = m.NIM
171 JOIN matakuliah mk ON n.id_matakuliah = mk.id_matakuliah;
172 SELECT * FROM view_nilai_mhs;
173
```

Gambar 13. View Nilai Mahasiswa

View ini digunakan untuk menampilkan informasi nilai mahasiswa yang digabungkan dari tiga tabel, yaitu nilai, mahasiswa, dan matakuliah. View ini menunjukkan NIM, nama\_mhs, nama\_matakuliah, dan nilai. Dengan adanya view ini, pengguna dapat dengan mudah melihat nilai setiap mahasiswa untuk setiap mata kuliah tanpa harus menulis query join yang kompleks.

172 **SELECT \* FROM view\_nilai\_mhs;**

	nim character varying (10)	nama_mhs character varying (100)	nama_matakuliah character varying (100)	nilai numeric (10,2)
1	11323010	Franklyn Lumbantoruan	Basis Data	85.00
2	11323010	Franklyn Lumbantoruan	Dasar Pemrograman	90.00
3	11323011	Ferry Siagian	Basis Data	78.00
4	11323011	Ferry Siagian	Matematika Diskrit	82.00
5	11323047	Agnes Sidabutar	Dasar Pemrograman	88.00
6	11323005	Abeloisa Pardosi	Matematika Diskrit	76.00
7	12S13005	Mabeloisa Manurung	Matematika Diskrit	89.00
8	12S13006	Monica Ginting	Jaringan Komputer	75.00
9	12S13060	Erick Sitohang	Algoritma dan Pemrograman	88.00
10	14S23001	Aan Kiki Siahaan	Jaringan Komputer	80.00

#### d. View rata rata nilai mahasiwa

```

174 -- 4. Menampilkan Rata-rata nilai mahasiswa-
175 CREATE VIEW rata_nilai AS
176 SELECT
177     NIM, AVG(nilai) AS rata_rata_nilai
178 FROM nilai GROUP BY NIM;
179 SELECT * FROM rata_nilai;

```

Gambar 14. View Rata- Rata Nilai mahasiswa

View ini dibuat untuk menghitung rata-rata nilai (AVG(nilai)) setiap mahasiswa berdasarkan NIM. Data yang dihasilkan meliputi NIM mahasiswa dan rata-rata nilai mereka. Dengan view ini, pengguna dapat dengan mudah memperoleh informasi tentang performa akademik mahasiswa secara ringkas,

179 **SELECT \* FROM rata\_nilai;**

	nim character varying (10)	rata_rata_nilai numeric
1	13323041	79.0000000000000000
2	11S23020	89.0000000000000000
3	11423040	93.0000000000000000
4	11323070	70.0000000000000000
5	12S13060	88.0000000000000000
6	14S23001	80.0000000000000000
7	21S23031	88.0000000000000000
8	11S23021	84.0000000000000000
9	13323045	81.0000000000000000
10	11323005	76.0000000000000000
11	14S23002	78.0000000000000000

Gambar 15. Output View rata rata nilai

*Gambar 19. Membuat User*

Query ini digunakan untuk membuat user bernama user\_mhs beserta dengan password nya

b. Memberikan akses

```
GRANT SELECT ON mahasiswa, matakuliah, nilai to user_mhs;
```

*Gambar 20. Memberi akses*

Query ini digunakan untuk memberikan akses tertentu kepada user

### Uji authorization

Untuk melakukan ini kita coba menginsert nilai mahasiswa

```
223 INSERT INTO nilai VALUES(20, '11323010', 'MK001');
```

Data Output Messages Notifications

```
ERROR: permission denied for table nilai

SQL state: 42501
```

*Gambar 21. Uji Authorization*

Terjadi eror karena user mahasiswa hanya boleh melakukan select saja tidak dengan insert

## 3.7. TRANSACTION

a. Transaction untuk melakukan insert data

```
227 -- TRANSACTION --
228 -- Mulai transaksi
229 BEGIN;
230
231 -- Tambahkan data ke tabel mahasiswa
232 INSERT INTO mahasiswa (NIM, nama_mhs, prodi, jenis_kelamin, email_mhs, kelas)
233 VALUES ('11523001', 'Dodi', 'Informatika', 'L', 'dodi@itdel.ac.id', '32IF2');
234
235 -- Tambahkan data ke tabel nilai
236 INSERT INTO nilai (nilai, NIM, id_matakuliah)
237 VALUES (60, '11523001', 'MK005');
238
239 -- Jika semua berhasil, commit transaksi
240 COMMIT;
241 -- Jika tidak berhasil, rollback transaksi
242 ROLLBACK;
243 -- Cek hasil transaksi
244 SELECT * FROM mahasiswa where nama_mhs = 'Dodi';
245
246 -- ERROR HANDLING TRANSACTION--
247 DO $$
248 BEGIN
249 -- Mulai transaksi
250 BEGIN
251
252 -- Tambahkan data ke tabel mahasiswa
253 INSERT INTO mahasiswa (NIM, nama_mhs, prodi, jenis_kelamin, email_mhs, kelas)
254 VALUES ('11323060', 'Joniana', 'Teknologi Informasi', 'P', 'nisanadizini@itdel.ac.id', '32TI2');
255
256 -- Tambahkan data ke tabel nilai
257 INSERT INTO nilai (nilai, NIM, id_matakuliah)
258 VALUES (100, '11323060', 'MK008');
259
260 -- Commit transaksi jika semua berhasil
261 COMMIT;
262
263 RAISE NOTICE 'Data berhasil disimpan.';
264 EXCEPTION WHEN OTHERS THEN
265 -- Rollback jika ada kesalahan
266 ROLLBACK;
267 RAISE NOTICE 'Terjadi kesalahan: %', SQLERRM;
268 END;
269 END $$;
```

*Gambar 22. Transaction untuk melakukan insert data*

Pada transaksi pertama, BEGIN digunakan untuk memulai transaksi, kemudian data baru ditambahkan ke tabel mahasiswa dan nilai. Jika semua perintah berhasil, transaksi diselesaikan dengan COMMIT; jika terjadi kesalahan, transaksi dibatalkan dengan ROLLBACK sehingga tidak ada perubahan yang disimpan. Pada transaksi ini juga terdapat penggunaan raise notice untuk memunculkan notifikasi apakah data tersebut berhasil dimasukkan atau tidak

```

246 -- ERROR HANDLING TRANSACTION--
247 DO $$
248 BEGIN
249 -- Mulai transaksi
250 BEGIN
251
252 -- Tambahkan data ke tabel mahasiswa
253 INSERT INTO mahasiswa (NIM, nama_mhs, prodi, jenis_kelamin, email_mhs, kelas)
254 VALUES ('11223060', 'Jontana', 'Teknologi Informasi', 'P', 'jtanadistek@tel.ac.id', '22112');
255
256 -- Tambahkan data ke tabel nilai
257 INSERT INTO nilai (nilai, NIM, id_matakuliah)
258 VALUES (100, '11223060', 'MK008');
259
260 -- Commit transaksi jika semua berhasil
261 COMMIT;
262
263 RAISE NOTICE 'Data berhasil disimpan.';
264 EXCEPTION WHEN OTHERS THEN
265 -- Rollback jika ada kesalahan
266 ROLLBACK;
267 RAISE NOTICE 'terjadi kesalahan: %', SQLERRM;
268 END;
269 END $$;

```

Data Output Messages Notifications

NOTICE: Terjadi kesalahan: insert on table "nilai" violates foreign key constraint: "nilai\_id\_matakuliah\_fkey"

Query returned successfully in 112 msec.

Gambar 23. Pesan error

Pada transaksi ini muncul pesan eror karena susai dengan query nya tadi jika ingin menginsert data harus sesuai dengan ketentuan yang ada pada saat ini di insert data yang tidak sesuai maka mncul lah eror

### 3.8. STORED PROCEDURE

a. Stored menambahkan mahasiswa baru dengan validasi

```

-- STORED PROCEDURE --
-- Menambahkan mahasiswa baru dengan validasi
CREATE OR REPLACE PROCEDURE tambah_mahasiswa(
    p_nim VARCHAR(10),
    p_nama VARCHAR(100),
    p_prodi VARCHAR(50),
    p_jenis_kelamin CHAR(1),
    p_email VARCHAR(100),
    p_kelas VARCHAR(10)
)
LANGUAGE plpgsql
AS $$
BEGIN
    -- Validasi input
    IF p_jenis_kelamin NOT IN ('L', 'P') THEN
        RAISE EXCEPTION 'Jenis kelamin harus "L" (Laki-laki) atau "P" (Perempuan)';
    END IF;

    -- Cek apakah NIM sudah ada
    IF EXISTS (SELECT 1 FROM mahasiswa WHERE NIM = p_nim) THEN
        RAISE EXCEPTION 'NIM % sudah terdaftar', p_nim;
    END IF;

    -- Tambahkan mahasiswa baru
    INSERT INTO mahasiswa (
        NIM,
        nama_mhs,
        prodi,
        jenis_kelamin,
        email_mhs,
        kelas
    ) VALUES (
        p_nim,
        p_nama,
        p_prodi,
        p_jenis_kelamin,
        p_email,
        p_kelas
    );

    -- Tampilkan pesan sukses
    RAISE NOTICE 'Mahasiswa % berhasil ditambahkan', p_nama;
END;
$$;

```

Gambar 24. Stored menambahkan mahasiswa baru dengan validasi

Query ini adalah sebuah prosedur penyimpanan bernama tambah\_mahasiswa, yang bertujuan untuk menambahkan data mahasiswa baru dengan melakukan validasi terlebih dahulu. Prosedur ini menerima beberapa parameter, yaitu NIM, nama, program studi, jenis kelamin, email, dan kelas mahasiswa. Pada procedure ini juga terdapat beberapa kondisi, Jika semua kondisi berhasil, data mahasiswa baru akan dimasukkan ke dalam tabel, dan sistem akan menampilkan pesan sukses yang menyatakan bahwa mahasiswa berhasil ditambahkan

### Pemanggilan Stored Procedure

Untuk mengecek apakah procedure berhasil atau tidak maka perlu dilakukan pemanggilan

```
-- Contoh penggunaan stored procedure
CALL tambah_mahasiswa(
    '11323099',
    'Baru Sekali',
    'Teknologi Informasi',
    'L',
    'baru@itdel.ac.id',
    '32TI2'
);
```

*Gambar 25. Pemanggilan Stored Procedure*

Jika berhasil maka akan tampil seperti ini

```
317 CALL tambah_mahasiswa(
318     '11323100',
319     'Baru Duakali',
320     'Teknologi Informasi',
321     'L',
322     'baru2@itdel.ac.id',
323     '32TI2'
324 );
325
```

Data Output	Messages	Notifications
NOTICE: Mahasiswa Baru Duakali berhasil ditambahkan		
CALL		
Query returned successfully in 108 msec.		

*Gambar 26. Output pemanggilan Stored Procedure jika berhasil*

### 3.9. CURSOR

a. Menampilkan data mahasiswa beserta nilai pada setiap mata kuliah

```
326 -- CURSOR --
327 -- menampilkan data mahasiswa beserta nilai mereka pada setiap mata kuliah. --
328 DO $$
329 DECLARE
330     studentRecord RECORD;
331 BEGIN
332     -- Cursor untuk membaca data mahasiswa dan nilai mereka
333     FOR studentRecord IN
334         SELECT m.nim, m.nama_mhs, m.prodi, mk.nama_matakuliah, n.nilai
335         FROM mahasiswa m
336         JOIN nilai n ON m.nim = n.nim
337         JOIN matakuliah mk ON n.id_matakuliah = mk.id_matakuliah
338     LOOP
339         -- Cetak data setiap baris yang diambil
340         RAISE NOTICE 'NIM: %, Nama: %, Prodi: %, Mata Kuliah: %, Nilai: %',
341             studentRecord.nim, studentRecord.nama_mhs, studentRecord.prodi,
342             studentRecord.nama_matakuliah, studentRecord.nilai;
343     END LOOP;
344 END $$;
```

*Gambar 27. Cursor menampilkan data mahasiswa beserta nilai pada setiap mata kuliah*

Query ini digunakan untuk menampilkan data mahasiswa beserta nilai mereka pada setiap mata kuliah. Prosedur ini mendeklarasikan sebuah cursor bernama studentRecord, yang digunakan untuk membaca data mahasiswa dan nilai mereka melalui query SQL yang melakukan join antara tabel mahasiswa dan tabel nilai. Dalam bagian loop, setiap baris hasil dari cursor diambil dan ditampilkan



menggunakan perintah RAISE NOTICE, yang mencetak informasi seperti NIM, nama, program studi, dan nilai mata kuliah. Maka akan ditampilkan

```

329 DO $$
330 DECLARE
331     studentRecord RECORD;
332 BEGIN
333     -- Cursor untuk membaca data mahasiswa dan nilai mereka
334     FOR studentRecord IN
335         SELECT m.nim, m.nama_mhs, m.prodi, mk.nama_matakuliah, n.nilai
336         FROM mahasiswa m
337         JOIN nilai n ON m.nim = n.nim
338         JOIN matakuliah mk ON n.id_matakuliah = mk.id_matakuliah
339     LOOP
340         -- Cetak data setiap baris yang diambil
341         RAISE NOTICE 'NIM: %, Nama: %, Prodi: %, Mata Kuliah: %, Nilai: %',
342             studentRecord.nim, studentRecord.nama_mhs, studentRecord.prodi,
343             studentRecord.nama_matakuliah, studentRecord.nilai;
344     END LOOP;
345 END $$;
346

```

Data Output Messages Notifications

```

NOTICE: NIM: 11323019, Nama: Franklyn Lumbantoruan, Prodi: Teknologi Informasi, Mata Kuliah: Dasar Pemrograman, Nilai: 90.00
NOTICE: NIM: 11323011, Nama: Ferry Siagian, Prodi: Teknologi Informasi, Mata Kuliah: Basis Data, Nilai: 78.00
NOTICE: NIM: 11323011, Nama: Ferry Siagian, Prodi: Teknologi Informasi, Mata Kuliah: Matematika Diskrit, Nilai: 82.00
NOTICE: NIM: 11323047, Nama: Agnes Sidabutar, Prodi: Teknologi Informasi, Mata Kuliah: Dasar Pemrograman, Nilai: 88.00
NOTICE: NIM: 11323005, Nama: Abeloisa Pardosi, Prodi: Teknologi Informasi, Mata Kuliah: Matematika Diskrit, Nilai: 76.00
NOTICE: NIM: 12513005, Nama: Mabeloisa Manurung, Prodi: Sistem Informasi, Mata Kuliah: Matematika Diskrit, Nilai: 89.00
NOTICE: NIM: 12513096, Nama: Monica Ginting, Prodi: Sistem Informasi, Mata Kuliah: Jaringan Komputer, Nilai: 75.00

```

Gambar 28. Output data mahasiswa beserta nilai pada setiap mata kuliah

### 3.10. FITUR UTAMA

#### a. Fungsi menghitung IPK otomatis

untuk melakukan fungsi ini, terlebih dahulu kita menambahkan atribut ipk kedalam table mahasiswa dengan alter table

```

-- Tambahkan kolom ipk ke tabel mahasiswa
ALTER TABLE mahasiswa ADD COLUMN ipk NUMERIC(10,2);

```

Gambar 29. Fungsi menghitung IPK otomatis

Selanjutnya tambahkan la fungsi nya

```

CREATE OR REPLACE FUNCTION hitung_ipk()
RETURNS TRIGGER AS $$
DECLARE
    total_bobot DECIMAL := 0;
    total_sks INT := 0;
BEGIN
    -- Menghitung total bobot dan total SKS
    SELECT
        SUM(CASE
            WHEN n.nilai >= 79.5 THEN 4.0 * mk.jumlah_sks
            WHEN n.nilai >= 72 AND n.nilai < 79.5 THEN 3.5 * mk.jumlah_sks
            WHEN n.nilai >= 64.5 AND n.nilai < 72 THEN 3.0 * mk.jumlah_sks
            WHEN n.nilai >= 57 AND n.nilai < 64.5 THEN 2.5 * mk.jumlah_sks
            WHEN n.nilai >= 49.5 AND n.nilai < 57 THEN 2.0 * mk.jumlah_sks
            WHEN n.nilai >= 34 AND n.nilai < 49.5 THEN 1.0 * mk.jumlah_sks
            ELSE 0.0
        END),
        SUM(mk.jumlah_sks)
    INTO
        total_bobot, total_sks
    FROM
        nilai n
    JOIN
        matakuliah mk ON n.id_matakuliah = mk.id_matakuliah
    WHERE
        n.NIM = NEW.NIM;

    -- Menghindari pembagian dengan nol dan memperbarui IPK
    IF total_sks = 0 THEN
        UPDATE mahasiswa SET ipk = 0 WHERE NIM = NEW.NIM;
    ELSE
        UPDATE mahasiswa SET ipk = total_bobot / total_sks WHERE NIM = NEW.NIM;
    END IF;

    RETURN NEW;
END;
$$
LANGUAGE plpgsql;

```

Gambar 30. Fungsi untuk menghitung Indeks Prestasi Kumulatif (IPK) secara otomatis

Query merupakan fungsi untuk menghitung Indeks Prestasi Kumulatif (IPK) secara otomatis dalam sistem basis data mahasiswa. Pertama, kolom ipk ditambahkan ke tabel mahasiswa untuk menyimpan nilai IPK. Fungsi hitung\_ipk didefinisikan untuk menghitung total bobot dan total SKS berdasarkan nilai mahasiswa yang tersimpan di tabel nilai. Fungsi ini menggunakan query SQL dengan kondisi untuk mengonversi nilai ke bobot sesuai dengan rentang nilai yang ditentukan



```
-- Trigger untuk tabel nilai
CREATE TRIGGER trigger_hitung_ipk
AFTER INSERT OR UPDATE OR DELETE
ON nilai
FOR EACH ROW
EXECUTE FUNCTION hitung_ipk();
```

*Gambar 31.Trigger\_hitung\_IPK*

Selanjutnya, trigger bernama trigger\_hitung\_ipk dibuat untuk memastikan bahwa fungsi hitung\_ipk dieksekusi setiap kali terjadi perubahan (insert, update, atau delete) pada tabel nilai

Kita lakukan pengujian untuk menguji apakah fungsi tersebut berhasil dilakukan atau tidak

```
402 -- Cek tabel mahasiswa
403 select * from mahasiswa where nim = '11323010';
```

Data Output Messages Notifications

	nim [PK] character varying (10)	nama_mhs character varying (100)	prodi character varying (50)	jenis_kelamin character (1)	email_mhs character varying (100)	kelas character varying (10)	ipk numeric (10,2)
1	11323010	Franklyn Lumbantoruan	Teknologi Informasi	L	franklyn@itdel.ac.id	32T11	4.00

*Gambar 32.Output Fungsi menghitung IPK otomatis*

## b. Fitur Menampilkan data mahasiswa berdasarkan kriteria tertentu

- *Menampilkan Data Mahasiswa Berdasarkan Jenis Kelamin Laki-laki*

Fitur ini membuat dua *view* untuk mempermudah menampilkan data mahasiswa berdasarkan jenis kelamin

```
404 -- 2.Menampilkan data mahasiswa berdasarkan kriteria tertentu.
405 --Menampilkan mahasiswa yang hanya berjenis kelamin laki-laki
406 CREATE VIEW view_mahasiswa_laki_laki AS
407 SELECT NIM, nama_mhs, prodi, kelas, email_mhs
408 FROM mahasiswa WHERE jenis_kelamin = 'L';
409 SELECT * FROM view_mahasiswa_laki_laki;
410
```

*Gambar 33.Menampilkan Data Mahasiswa Berdasarkan Jenis Kelamin Laki-laki*

Fungsi ini membuat *view* view\_mahasiswa\_laki\_laki yang menyajikan data mahasiswa laki-laki dari tabel mahasiswa. Hanya data mahasiswa dengan kolom jenis\_kelamin bernilai 'L' yang ditampilkan.

- **Menampilkan Mahasiswa yang Berjenis Kelamin Perempuan**

```
411 --Menampilkan mahasiswa yang hanya berjenis kelamin perempuan
412 CREATE VIEW view_mahasiswa_perempuan AS
413 SELECT NIM, nama_mhs, prodi, kelas, email_mhs
414 FROM mahasiswa WHERE jenis_kelamin = 'P';
415 SELECT * FROM view_mahasiswa_perempuan;
416
```

*Gambar 34. Menampilkan Mahasiswa yang Berjenis Kelamin Perempuan*

Fungsi ini membuat *view* *view\_mahasiswa\_perempuan* untuk menampilkan data mahasiswa perempuan dari tabel *mahasiswa*. Hanya mahasiswa dengan kolom *jenis\_kelamin* bernilai 'P' yang akan muncul dalam hasil *view*. *View* ini dirancang untuk mempermudah akses data mahasiswa berdasarkan gender perempuan.

- **Menampilkan Nilai Mahasiswa Berdasarkan Kelas**

```
417 --Menampilkan nilai mahasiswa berdasarkan kelas --
418 CREATE VIEW view_nilai_mahasiswa_per_kelas AS
419 SELECT a.kelas, a.NIM, a.nama_mhs, mk.nama_matakuliah, n.nilai
420 FROM mahasiswa a JOIN nilai n ON a.NIM = n.NIM
421 JOIN matakuliah mk ON n.id_matakuliah = mk.id_matakuliah
422 ORDER BY a.kelas, a.NIM;
423 SELECT * FROM view_nilai_mahasiswa_per_kelas;
424
```

*Gambar 35. Menampilkan Nilai Mahasiswa Berdasarkan Kelas*

Fungsi ini membuat *view* *view\_nilai\_mahasiswa\_per\_kelas* yang menggabungkan data dari tabel *mahasiswa*, *nilai*, dan *matakuliah*. *View* ini menampilkan nilai mahasiswa berdasarkan kelas mereka, Data dalam *view* ini diurutkan berdasarkan kelas dan NIM untuk mempermudah analisis nilai berdasarkan kelompok kelas tertentu.

- **Menampilkan Data Mahasiswa Berdasarkan IPK Tertinggi**

```
425 --Menampilkan data mahasiswa berdasarkan IPK Tertinggi --
426 CREATE VIEW view_mahasiswa_ipk_tertinggi AS
427 SELECT NIM, nama_mhs, prodi, jenis_kelamin, kelas, email_mhs, ipk
428 FROM mahasiswa WHERE ipk = '4';
429 SELECT * FROM view_mahasiswa_ipk_tertinggi;
```

*Gambar 36. Menampilkan Data Mahasiswa Berdasarkan IPK Tertinggi*

Fungsi ini membuat *view* *view\_mahasiswa\_ipk\_tertinggi* untuk menyaring data mahasiswa dengan IPK tertinggi (bernilai 4.0). Data yang ditampilkan meliputi NIM, nama mahasiswa, program studi, jenis kelamin, kelas, email, dan IPK. *View* ini mempermudah identifikasi mahasiswa berprestasi tanpa perlu menulis ulang query untuk mencari mahasiswa dengan

Maka setelah di lakukan restore table yang ada akan Kembali lagi.

```
C:\Program Files\PostgreSQL\16\bin>psql -U postgres -d kelompok_lima
Password for user postgres:

psql (16.4)
WARNING: Console code page (850) differs from Windows code page (1252)
        8-bit characters might not work correctly. See psql reference
        page "Notes for Windows users" for details.
Type "help" for help.

kelompok_lima=# \dt
               List of relations
 Schema |      Name      | Type  | Owner
-----+-----+-----+-----
 public | mahasiswa      | table | postgres
 public | matakuliah     | table | postgres
 public | nilai          | table | postgres
(3 rows)

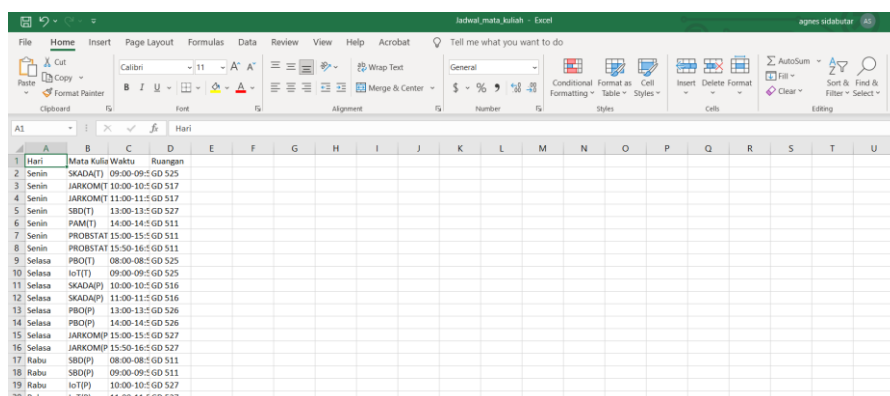
kelompok_lima=# |
```

Gambar 40. Output dari Restore

## 3.12. Crawling dan Scraping

### 3.12.1. Crawling

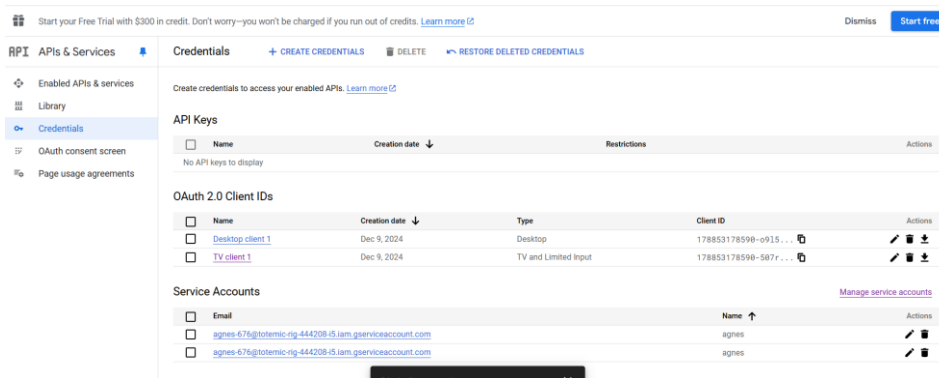
Pada proyek ini terdapat Crawling jadwal kelas dengan mengambil data jadwal dari Google Calendar. Crawling ini dilakukan dengan sumber data (seperti Google Calendar) dengan menggunakan API yang disediakan. Adapun tahap yang dilakukan yaitu kita terlebih dahulu membuat file csv berisi jadwal matakuliah



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
1	Hari	Mata Kuliah Waktu Ruang																			
2	Senin	SKADA(T)	09:00-09:55	GD 525																	
3	Senin	JARKOM(T)	10:00-10:55	GD 517																	
4	Senin	JARKOM(T)	11:00-11:55	GD 517																	
5	Senin	SBD(T)	13:00-13:55	GD 527																	
6	Senin	PRM(T)	14:00-14:55	GD 511																	
7	Senin	PROBSTAT	15:00-15:55	GD 511																	
8	Senin	PROBSTAT	15:50-16:55	GD 511																	
9	Selasa	PRM(T)	08:00-08:55	GD 525																	
10	Selasa	IoT(T)	09:00-09:55	GD 525																	
11	Selasa	SKADA(P)	10:00-10:55	GD 516																	
12	Selasa	SKADA(P)	11:00-11:55	GD 516																	
13	Selasa	PRM(P)	13:00-13:55	GD 526																	
14	Selasa	PRM(P)	14:00-14:55	GD 526																	
15	Selasa	JARKOM(P)	15:00-15:55	GD 527																	
16	Selasa	JARKOM(P)	15:50-16:55	GD 527																	
17	Rabu	SBD(P)	08:00-08:55	GD 511																	
18	Rabu	SBD(P)	09:00-09:55	GD 511																	
19	Rabu	IoT(P)	10:00-10:55	GD 527																	
20	Rabu	IoT(P)	11:00-11:55	GD 527																	

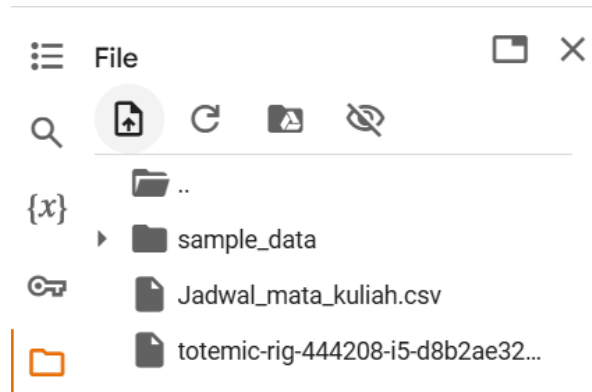
Gambar 41. CSV Jadwal Update

Selanjut nya kita akan membuat proyek di dalam google cloud dengan menerapkan google calender kemudian mendownload file autentikasi yang akan digunakan pada google colab



*Gambar 42. Autentikasi google api*

Selanjutnya pada google colab upload file csv dan file autentikasi nya yang kita gunakan untuk mengupload jadwal mata kuliah kita kedalam google kalender



*Gambar 43. Google colab upload file csv dan file autentikasi*

```
import pandas as pd
from googleapiclient.discovery import build
from google.oauth2 import service_account
import datetime

# Nama file JSON dan file CSV
CREDENTIALS_FILE = 'totemic-rig-444208-i5-d8b2ae32bb7.json'
CSV_FILE = 'Jadwal_mata_kuliah.csv'

# Scope Google Calendar API
SCOPES = ['https://www.googleapis.com/auth/calendar']

# Autentikasi menggunakan service account
def authenticate_google():
    creds = service_account.Credentials.from_service_account_file(CREDENTIALS_FILE, scopes=SCOPES)
    service = build('calendar', 'v3', credentials=creds)
    return service

# Menambahkan acara berulang ke Google Calendar
def add_recurring_event(summary, location, start_time, end_time, byday):
    event = {
        'summary': summary,
        'location': location,
        'start': {
            'dateTime': start_time,
            'timeZone': 'Asia/Jakarta',
        },
        'end': {
            'dateTime': end_time,
            'timeZone': 'Asia/Jakarta',
        },
        'recurrence': [
            f'RRULE:FREQ=WEEKLY;BYDAY={byday}'
        ]
    }
    event_result = service.events().insert(calendarId='primary', body=event).execute()
    print(f"Event created: {event_result.get('htmlLink')}")

# Map hari ke RRULE BYDAY
day_map = {
    'Senin': "MO", "Selasa": "TU", "Rabu": "WE",
    "Kamis": "TH", "Jumat": "FR", "Sabtu": "SA", "Minggu": "SU"
}

# Proses utama
def main():
    service = authenticate_google()
    df = pd.read_csv(CSV_FILE)

    for index, row in df.iterrows():
        hari = row['Hari']
        waktu = row['Waktu']
        mata_kuliah = row['Mata Kuliah']
        ruangan = row['Ruangan']

        # Pisahkan waktu mulai dan selesai
        waktu_mulai, waktu_selesai = waktu.split('-')

        # Format waktu mulai dan selesai (gunakan tanggal acak)
        start_time = f"2024-01-01T{waktu_mulai}:00" # Gunakan tanggal acak (Senin)
        end_time = f"2024-01-01T{waktu_selesai}:00"

        # Hari ke BYDAY (recurrence)
        byday = day_map.get(hari, None)
        if not byday:
            print(f"Error: Hari {hari} tidak valid.")
            continue

        # Tambahkan acara berulang
        add_recurring_event(service, mata_kuliah, ruangan, start_time, end_time, byday)

    print("Semua jadwal berhasil ditambahkan ke Google Calendar.")

# Menjalankan program utama
if __name__ == '__main__':
    main()
```

*Gambar 44. Menambahkan jadwal mata kuliah ke Google Calendar*

Syntax ini digunakan untuk menambahkan jadwal mata kuliah ke Google Calendar secara otomatis menggunakan Google Calendar API. Program ini dimulai dengan mengimpor beberapa library yang diperlukan, seperti pandas untuk memanipulasi data dari file CSV, dan googleapiclient untuk mengakses Google Calendar. Setelah mendeklarasikan nama file JSON untuk autentikasi dan file CSV untuk jadwal, program menetapkan scope akses ke Google Calendar. Setelah berhasil akan muncul seperti ini,

Event created: <https://www.google.com/calendar/event?eid=Ym1wM28yYXA3OXRhaDEybmRucTRoYzZjcmdmJmJyNDAXMDFUMDQwMDI/>  
 Event created: <https://www.google.com/calendar/event?eid=am42M3N0a3N0OTkzM2hlcXBPbA40XNoZm9fMjAyNDAXMDFUMDQwMDI/>  
 Event created: <https://www.google.com/calendar/event?eid=b2NmcmVwYmY1YWhhdjJua2E1aWRtZHBkMjBfMjAyNDAXMDFUMDQwMDI/>  
 Event created: <https://www.google.com/calendar/event?eid=ZG1nNzIwZDY0YXoxYXpwZzRuMmQ5Mm82Z3NfMjAyNDAXMDFUMDQwMDI/>  
 Event created: <https://www.google.com/calendar/event?eid=cWg3dGpva3YzZm9hdjJ1Z2w4YmM3a2Zhdw9fMjAyNDAXMDFUMDQwMDI/>  
 Event created: <https://www.google.com/calendar/event?eid=dWZpZmtjCHZ1bTdmOGpZbnUyOGh2bXVrYmtfMjAyNDAXMDFUMDQwMDI/>  
 Event created: <https://www.google.com/calendar/event?eid=Y21yOXN2cGk1N3A5c3Y1N2ZuYWhmMjVjSmlmVfMjAyNDAXMDFUMDQwMDI/>  
 Event created: <https://www.google.com/calendar/event?eid=MwRUNDYzdDlrMmFjdXFrOWsxdDBzcTJjam9fMjAyNDAXMDFUMDQwMDI/>  
 Semua jadwal berhasil ditambahkan ke Google Calendar.

*Gambar 45.CSV jadwal matakuliah berhasil di tambahkan kedalam google calender*

Dimana pada gambar ini ditunjukkan bahwa semua data yang ada pada csv jadwal matakuliah berhasil di tambahkan kedalam google calender, selanjutnya kita akan melakukan crawling untuk mengambil jadwal nya tadi

```
import pandas as pd
from googleapiclient.discovery import build
from google.oauth2 import service_account
import datetime
import pytz # Pastikan untuk menginstal pytz

# Nama file JSON dan file CSV
CREDENTIALS_FILE = 'totemic-rig-444208-i5-d8b2ae322bb7.json'
OUTPUT_CSV_FILE = 'Jadwal_Terupload.csv'

# Scope Google Calendar API
SCOPES = ['https://www.googleapis.com/auth/calendar']

# Autentikasi menggunakan service account
def authenticate_google():
    creds = service_account.Credentials.from_service_account_file(CREDENTIALS_FILE, scopes=SCOPES)
    service = build('calendar', 'v3', credentials=creds)
    return service

# Mengambil data dari Google Calendar
def fetch_events(service, calendar_id='primary'):
    # Tentukan rentang tanggal
    time_min = datetime.datetime(2024, 8, 20, 0, 0, 0, tzinfo=pytz.utc).isoformat()
    time_max = datetime.datetime(2024, 12, 10, 23, 59, 59, tzinfo=pytz.utc).isoformat()
```

```
events_result = service.events().list(
    calendarId=calendar_id,
    timeMin=time_min,
    timeMax=time_max,
    maxResults=1000,
    singleEvents=True,
    orderBy='startTime'
).execute()
events = events_result.get('items', [])
return events

# Map hari dari angka ke nama
day_reverse_map = {0: "Senin", 1: "Selasa", 2: "Rabu", 3: "Kamis", 4: "Jumat", 5: "Sabtu", 6: "Minggu"}

# Proses utama
def main():
    service = authenticate_google()

    # Ambil semua event dari Google Calendar
    events = fetch_events(service)

    # Proses acara ke dalam format yang diinginkan
    event_data = []
    for event in events:
        start_time = event['start'].get('dateTime', event['start'].get('date'))
        end_time = event['end'].get('dateTime', event['end'].get('date'))
```

```
print(f"Start Time: {start_time}, End Time: {end_time}") # Tambahkan ini untuk debugging

if 'dateTime' in event['start']: # Acara dengan waktu
    # Handle 'Z' at the end of the string if present
    start_time = start_time.replace("Z", "+00:00") if "Z" in start_time else start_time
    end_time = end_time.replace("Z", "+00:00") if "Z" in end_time else end_time

    # Konversi ke timezone lokal jika diperlukan
    local_timezone = pytz.timezone('Asia/Jakarta') # Ganti dengan timezone yang sesuai
    start_datetime = datetime.datetime.fromisoformat(start_time).astimezone(local_timezone)
    end_datetime = datetime.datetime.fromisoformat(end_time).astimezone(local_timezone)

    print(f"Converted Start: {start_datetime}, Converted End: {end_datetime}") # Tambahkan ini untuk debugging

    hari = day_reverse_map[start_datetime.weekday()]
    waktu = f"{start_datetime.strftime('%H:%M')}-{end_datetime.strftime('%H:%M')}"
else: # Acara sepanjang hari (tidak digunakan di sini)
    continue
```

```

# Tambahkan ke daftar hasil
event_data.append({
    'Tanggal': start_datetime.strftime('%Y-%m-%d'),
    'Hari': hari,
    'Mata Kuliah': event.get('summary', 'Tanpa Judul'),
    'Waktu': waktu,
    'Ruangan': event.get('location', 'Tanpa Lokasi')
})

# Simpan hasil ke file CSV
df = pd.DataFrame(event_data)
df.to_csv(OUTPUT_CSV_FILE, index=False)
print(f"Jadwal berhasil disimpan ke {OUTPUT_CSV_FILE}")

# Menjalankan program utama
if __name__ == '__main__':
    main()

```

*Gambar 46. Mengambil jadwal kelas dari Google Calendar menggunakan akun layanan (service account) dan menyimpannya ke dalam file CSV*

Syntax diatas digunakan untuk mengambil jadwal kelas dari Google Calendar menggunakan akun layanan (service account) dan menyimpannya ke dalam file CSV. Proses ini mencakup autentikasi dengan Google Calendar API, pengambilan data acara dalam rentang waktu tertentu, serta pemrosesan dan penyimpanan data tersebut ke dalam format yang mudah dibaca. Setelah di run maka akan muncul

```

Start Time: 2024-12-10T07:00:00Z, End Time: 2024-12-10T07:50:00Z
Converted Start: 2024-12-10 14:00:00+07:00, Converted End: 2024-12-10 14:50:00+07:00
Start Time: 2024-12-10T08:00:00Z, End Time: 2024-12-10T08:50:00Z
Converted Start: 2024-12-10 15:00:00+07:00, Converted End: 2024-12-10 15:50:00+07:00
Start Time: 2024-12-10T08:50:00Z, End Time: 2024-12-10T09:50:00Z
Converted Start: 2024-12-10 15:50:00+07:00, Converted End: 2024-12-10 16:50:00+07:00
Jadwal berhasil disimpan ke Jadwal_Terupload.csv

```

*Gambar 47. Semua data berhasil di ambil dan disimpan*

Ini berarti bahwa semua data yang ada di dalam google calender berhasil diambil dan disimpan dalam file csv baru yang langsung terdownload yang berisikan hasil dari google calender

	A	B	C	D	E	F	G	H
1	Tanggal	Hari	Mata Kuliah	Waktu	Ruangan			
2	8/20/2024	Selasa	PBO(T)	08:00-08:50	GD 525			
3	8/20/2024	Selasa	IoT(T)	09:00-09:50	GD 525			
4	8/20/2024	Selasa	SKADA(P)	10:00-10:50	GD 516			
5	8/20/2024	Selasa	SKADA(P)	11:00-11:50	GD 516			
6	8/20/2024	Selasa	PBO(P)	13:00-13:50	GD 526			
7	8/20/2024	Selasa	PBO(P)	14:00-14:50	GD 526			
8	8/20/2024	Selasa	JARKOM(P)	15:00-15:50	GD 527			
9	8/20/2024	Selasa	JARKOM(P)	15:50-16:50	GD 527			
10	8/21/2024	Rabu	SBD(P)	08:00-08:50	GD 511			

*Gambar 48. Jadwal yang sudah berhasil*

Dimana tanggal pada jadwal ini sudah mengikuti tanggal yang ada pada google calender

### 3.12.2. Scraping

```
import pandas as pd

def scrape_class_schedule(file_path):
    """
    Scrape class schedule data from a CSV file.

    Parameters:
    file_path (str): Path to the CSV file containing the class schedule.

    Returns:
    pandas.DataFrame: DataFrame containing the class schedule data.
    """
    # Read the CSV file into a DataFrame
    df = pd.read_csv(file_path)

    # Parse the 'Waktu' column to extract start and end times
    df[['Start_Time', 'End_Time']] = df['Waktu'].str.split('-', expand=True)
    df['Start_Time'] = pd.to_datetime(df['Start_Time'], format='%H:%M').dt.time
    df['End_Time'] = pd.to_datetime(df['End_Time'], format='%H:%M').dt.time

    # Create a new DataFrame with the desired columns
    class_schedule = df[['Hari', 'Mata Kuliah', 'Start_Time', 'End_Time', 'Ruangan']]

    return class_schedule

# Example usage
class_schedule = scrape_class_schedule('Jadwal_mata_kuliah.csv')
print(class_schedule)
```

Gambar 49. Fungsi `scrape_class_schedule`

Fungsi `scrape_class_schedule` yang didefinisikan dalam ini bertujuan untuk memproses jadwal mata kuliah dari file CSV. Fungsi ini menerima satu parameter, yaitu `file_path`, yang merupakan jalur menuju file CSV. akhirnya, fungsi mengembalikan DataFrame yang telah diproses ini yaitu

	Hari	Mata Kuliah	Start_Time	End_Time	Ruangan
0	Senin	SKADA(T)	09:00:00	09:50:00	GD 525
1	Senin	JARKOM(T)	10:00:00	10:50:00	GD 517
2	Senin	JARKOM(T)	11:00:00	11:50:00	GD 517
3	Senin	SBD(T)	13:00:00	13:50:00	GD 527
4	Senin	PAM(T)	14:00:00	14:50:00	GD 511
5	Senin	PROBSTAT(T)	15:00:00	15:50:00	GD 511
6	Senin	PROBSTAT(T)	15:50:00	16:50:00	GD 511
7	Selasa	PBO(T)	08:00:00	08:50:00	GD 525
8	Selasa	IoT(T)	09:00:00	09:50:00	GD 525
9	Selasa	SKADA(P)	10:00:00	10:50:00	GD 516
10	Selasa	SKADA(P)	11:00:00	11:50:00	GD 516

Gambar 50. Hasil `scrape_class_schedule`



## HASIL PRESENTASI

No.	Pertanyaan	11323010	11323011	11323005	11323047
1.	Apakah keseluruhan implementasi fitur tercapai dan relevan dengan tujuannya?	Implementasi fitur sudah tercapai dan relevan dengan tujuan.	Implementasi sudah tercapai dengan isi sesuai dengan apa yang telah dikerjakan pada praktikum sebelumnya.	Menurut saya sudah, karena pada proyek ini fitur" utama sudah dapat di jalankan dengan baik.	Iya sudah dilakukan implementasi dan fitur utama berhasil di jalankan.
2.	Apa yang lebih dan apa yang kurang untuk diperbaiki (kalau ada)?	Kami terlalu lebih membuat query sederhana dan kurang eksplorasi dan jumlah tabel ataupun entity dan role nya juga masih terlalu sedikit.	Pada erd ada baiknya menambah sebuah tabel baru untuk jadwal (dari hasil scrapping).	ERD yang dibuat untuk proyek kelompok 5 masih belum memenuhi, ada baiknya di tambahkan dan melakukan eksplorasi lebih banyak lagi di kemudian hari	Semakin melakukan eksplorasi lagi agar dapat mengembangkan sistem yang lebih baik lagi.
3.	Menurutmu apakah kontribusimu sudah maksimal dan kamu paham dengan yang kamu implementasikan?	Kontribusi saya sudah maksimal dan saya sudah paham dengan yang saya buat.	Untuk kontribusi sudah bagus, untuk query yang tersedia pada proyek pun sudah terimplementasi.	Implementasi dan kontribusi saya masih kurang untuk pembelajaran ini, karena saya belum mengerti dan belum terlalu paham.	Kurang, karena masih mengimplementasikan yang sederhana.

## PENILAIAN

No.	Jenis Penilaian	11323010	11323011	11323005	11323047
1.	Hasil Kelompok	90	100	100	100
2.	Diri Sendiri	90	100	100	90