



# PRESENTATION TITLE

---

SUBTITLE COMES HERE

# CONTENT 01

Ikhtisar (*Overview*) SQL

Elemen-elemen Bahasa SQL

Komponen Bahasa SQL

DDL

Statement INSERT

Statement UPDATE

Statement DELETE



# SQL



- Adalah singkatan dari ***Structured Query Language***
  - Bahasa yang digunakan untuk **berkomunikasi** dengan database.
  - Dikirim oleh **manusia**. diolah oleh **DBMS** diterapkan ke **basisdata**.
- SQL ada yang bersifat:
  - ***Generic*** (umum)    ? Dapat diterima oleh semua DBMS
  - ***Specific*** (khusus)    ? Hanya DBMS tertentu saja
- SQL memiliki:
  - ***Elements***    ? Bagian-bagian kecil penyusun (*structure*)
  - ***Components***    ? Pembagian berdasarkan kegunaannya



# ELEMEN-ELEMEN BAHASA SQL (1/2)

- **Identifiers**

- Adalah nama-nama objek yang ada pada database.
  - Contoh: nama tabel, nama kolom.

- **Keywords**

- Kata-kata tercadang, yang merupakan elemen dasar bahasa SQL yang tidak boleh kita gunakan sebagai identifier.
  - Contoh: SELECT, FROM, CREATE, ALTER, dlsb.

- **Operator**

- Karakter maupun kata yang meng-operasikan 2 buah elemen.
  - Contoh: +, -, \*, /, AND, OR, dlsb.

- **Expressions**

- Dua atau lebih elemen bahasa yang digabungkan dengan operator sehingga memiliki nilai.
  - Contoh: 1 + 1, ipk > 3, terdaftar IS TRUE

## ELEMEN-ELEMEN BAHASA SQL (2/2)

- **Literals**

- Adalah cara penulisan suatu nilai yang menyebabkan nilai tersebut secara otomatis dianggap sebagai tipe data tertentu.
  - Contoh:
    - 'Adi' dan "Adi" ? Dianggap sebagai **string**,
    - '2015-07-21', '20150721', dan 20150721 ? Dianggap sebagai **date**,
    - 2500 ? dianggap sebagai **integer**

- **Comments**

- Atau komentar, adalah sederetan kata yang tidak dieksekusi/diolah oleh DBMS.
  - Gunakan – atau /\* \*/

## ELEMEN-ELEMEN BAHASA SQL (2/2)

- **Literals**

- Adalah cara penulisan suatu nilai yang menyebabkan nilai tersebut secara otomatis dianggap sebagai tipe data tertentu.
  - Contoh:
    - 'Adi' dan "Adi" ? Dianggap sebagai **string**,
    - '2015-07-21', '20150721', dan 20150721 ? Dianggap sebagai **date**,
    - 2500 ? dianggap sebagai **integer**

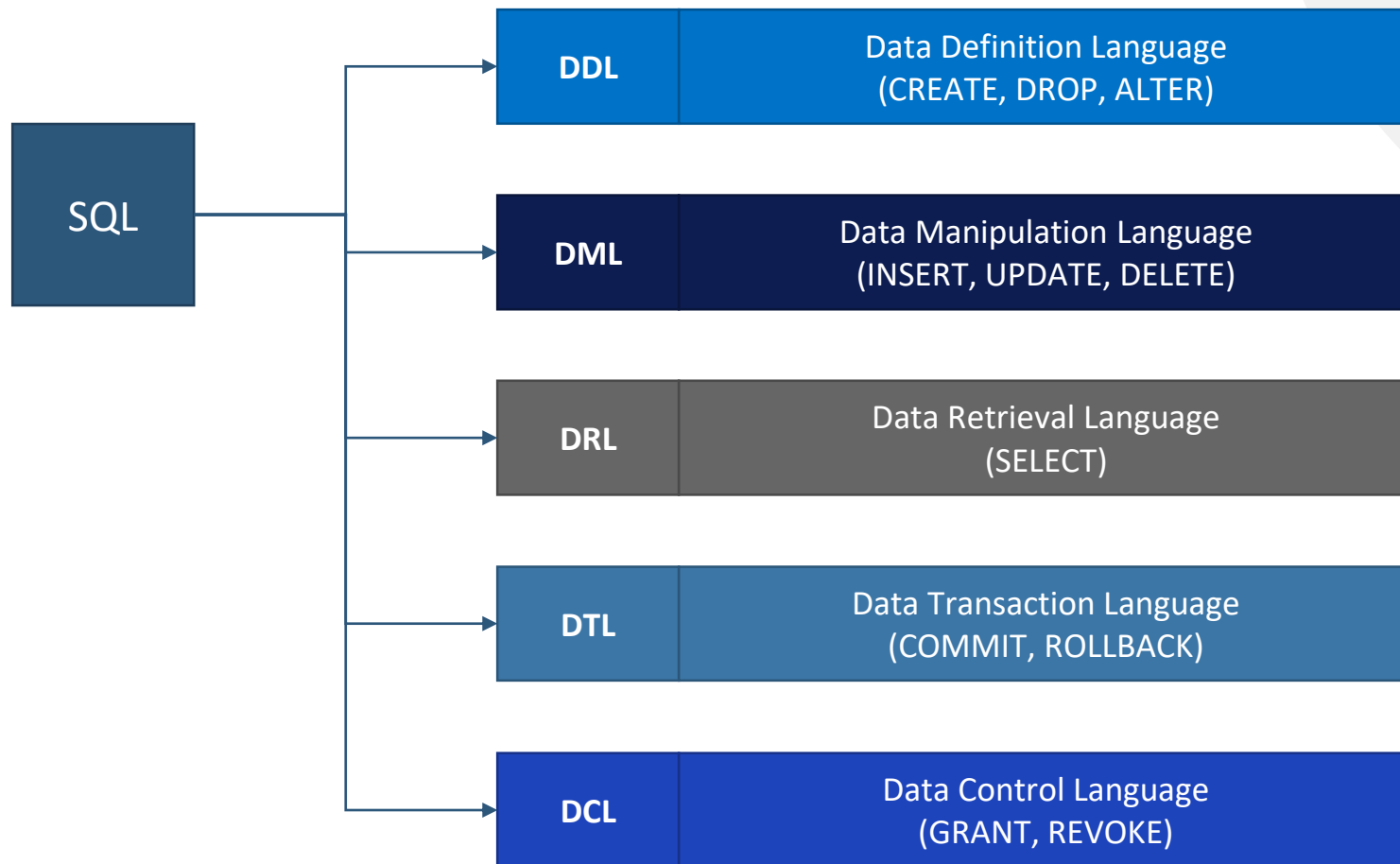
- **Comments**

- Atau komentar, adalah sederetan kata yang tidak dieksekusi/diolah oleh DBMS.
  - Gunakan – atau /\* \*/

## PENYSUNAN ELEMEN-ELEMEN

- Elemen-elemen bahasa yang telah dijelaskan sebelumnya, dapat digabung dan disusun menjadi:
  - Clauses (Klausa)
- Dan klausa-klausa, dapat digabungkan dengan elemen-elemen lain menjadi satu perintah lengkap yang disebut:
  - Statements (statement)
  - Diakhiri dengan semikolon (;)
- **SELECT** nim, nama **FROM** mahasiswa **WHERE** ipk > 3

# KOMPONEN BAHASA SQL

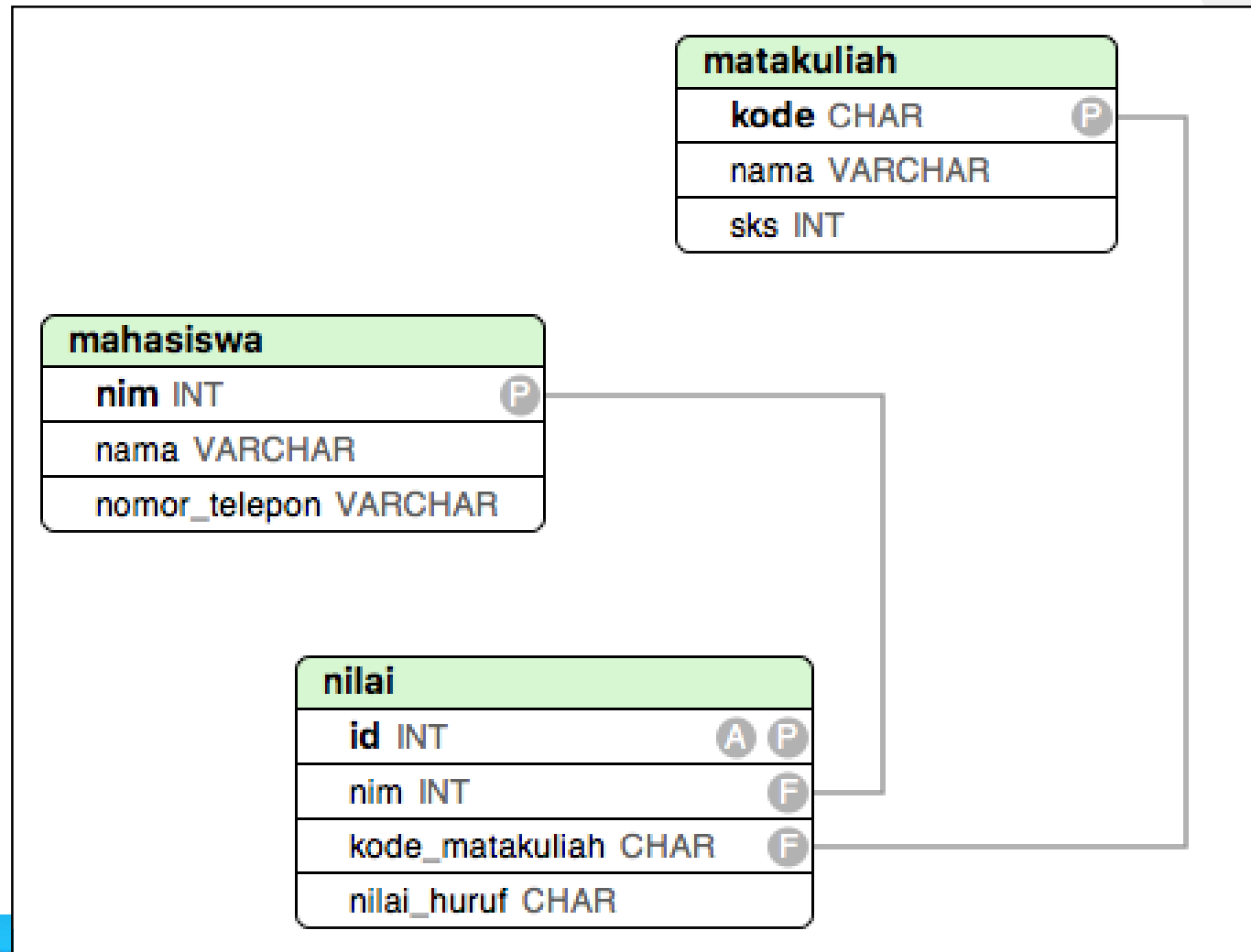




# DML (DATA MANIPULATION LANGUAGE)

- Adalah BAHASA yang digunakan untuk memerintahkan DBMS agar melakukan operasi-operasi yang sifatnya **MENGUBAH** nilai-nilai data pada (**ISI**) **tabel**.
- Yang diubah oleh:
  - DDL **?** **Struktur** tabel.
  - DML **?** **Isi** tabel.
- Perspektif:
  - DDL **?** **TABEL**.
  - DML **?** **BARIS**/Row/Record/Tuple.
- Ada 3 klausa utama:
  - INSERT : **Menambahkan** suatu BARIS baru.
  - UPDATE : **Mengganti** nilai pada suatu BARIS.
  - DELETE : **Menghapus** suatu BARIS.
- Dan 1 klausa syarat (filtering):
  - **WHERE**

# DATABASE AKADEMIK



# DATABASE AKADEMIK: DDL

```
CREATE DATABASE akademik;

USE akademik;

CREATE TABLE mahasiswa (
    nim INT(2),
    nama VARCHAR(255),
    nomor_telepon VARCHAR(20),
    PRIMARY KEY (nim)
);

CREATE TABLE matakuliah (
    kode CHAR(3),
    nama VARCHAR(255),
    sks INT(1),
    PRIMARY KEY (kode)
);

CREATE TABLE nilai (
    id INT AUTO_INCREMENT,
    nim INT(2),
    kode_matakuliah CHAR(3),
    nilai_huruf CHAR(2),
    PRIMARY KEY (id)
);

ALTER TABLE nilai
    ADD FOREIGN KEY nim_idxfk (nim) REFERENCES mahasiswa (nim);

ALTER TABLE nilai
    ADD FOREIGN KEY kode_matakuliah_idxfk (kode_matakuliah) REFERENCES matakuliah (kode);
```

# INSERT

- Digunakan untuk **menambahkan RECORD/Baris** baru pada suatu tabel.
- Klausa pembentuk:
  - **INSERT**
  - **INTO**
  - **VALUES**
- Format:
  1. **INSERT INTO** nama\_tabel (kolom1, kolom2, ...dst.) **VALUES** (nilai\_kolom1, nilai\_kolom2, ...dst.);
  2. **INSERT INTO** nama\_tabel **VALUES** (nilai\_kolom1, nilai\_kolom2, ...dst.);
  3. [Salah satu dari kedua format sebelumnya], (nilai\_kolom\_kolom\_baris1), (nilai\_kolom\_kolom\_baris2), ...dst.

- Contoh Format #1:
  - **INSERT INTO** nama\_tabel (kolom1, kolom2, ...dst.) **VALUES** (nilai\_kolom1, nilai\_kolom2, ...dst.);
- Digunakan jika kita ingin menambahkan data pada **sebagian** kolom saja.

SQL:

```
INSERT INTO mahasiswa (nim, nama) VALUES (1, 'Ani Rahmawati');
```

Akan menghasilkan:

nim	nama	nomor_telepon
1	Ani Rahmawati	NULL



# INSERT

- Contoh Format #2:
  - **INSERT INTO** nama\_tabel **VALUES** (nilai\_kolom1, nilai\_kolom2, ...dst.);
- Digunakan jika kita ingin menambahkan baris baru dengan data pada **semua** kolom.

SQL:

```
INSERT INTO mahasiswa VALUES (2, 'Budi Raharjo', '0858776453');
```

Akan menghasilkan:

+-----+	+-----+	+-----+	+-----+
nim	nama		nomor_telepon
+-----+	+-----+	+-----+	+-----+
1	Ani Rahmawati		NULL
2	Budi Raharjo		0858776453
+-----+	+-----+	+-----+	+-----+

# INSERT

- Contoh Format **#3**:
  - [Salah satu dari kedua format sebelumnya], (nilai\_kolom\_kolom\_baris1), (nilai\_kolom\_kolom\_baris2), ...dst.
- Digunakan jika kita ingin menambahkan **beberapa baris** baru **sekaligus** dalam 1 SQL.

SQL:

```
INSERT INTO mahasiswa VALUES  
  (3, 'Charlie Setiabudi', '0859767553'),  
  (4, 'Diandra Paramita', '0858998745');
```

Akan menghasilkan:

nim	nama	nomor_telepon
1	Ani Rahmawati	NULL
2	Budi Raharjo	0858776453
3	Charlie Setiabudi	0859767553
4	Diandra Paramita	0858998745

# INSERT

- [Salah satu dari kedua format sebelumnya], (nilai\_semua\_kolom\_baris1), (nilai\_semua\_kolom\_baris2), ...dst.

SQL:

```
INSERT INTO matakuliah (kode, nama) VALUES
  ('BDD', 'Basis Data Dasar'),
  ('PBO', 'Pemrograman Berorientasi Objek'),
  ('MMT', 'Multimedia Terapan'),
  ('SPK', 'Sistem Pendukung Keputusan'),
  ('KCB', 'Kecerdasan Buatan'),
  ('ASD', 'Algoritma dan Struktur Data');
```

Akan menghasilkan:

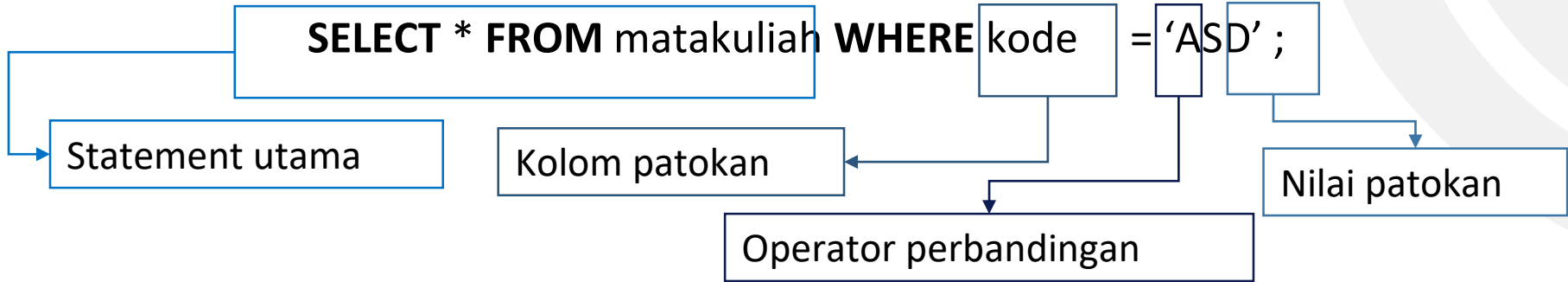
kode	nama	sks
ASD	Algoritma dan Struktur Data	NULL
BDD	Basis Data Dasar	NULL
KCB	Kecerdasan Buatan	NULL
MMT	Multimedia Terapan	NULL
PBO	Pemrograman Berorientasi Objek	NULL
SPK	Sistem Pendukung Keputusan	NULL

# KLAUSA 'WHERE'

- **WHERE** digunakan pada statement-statement UPDATE, DELETE, dan SELECT sebagai **filter/pembatas** terhadap **hasil** yang dikembalikan.
- Format:
  - [Statement Utama] **WHERE** kolom\_patokan [operator\_perbandingan] nilai\_patokan;
  - [Statement Utama] **WHERE** kolom\_patokan1 [operator\_perbandingan1] nilai\_patokan1 [operator\_logika1] kolom\_patokan1 [operator\_perbandingan2] nilai\_patokan2 [operator\_logika2] ...dst.;
- **Operator perbandingan**/comparison operator dapat berupa:
  - =, <, >, <=, >=, <>
- **Operator logika** dapat berupa:
  - **AND, OR**
- Contoh:
  - **SELECT \* FROM** matakuliah **WHERE** kode = 'ASD' ;
  - **UPDATE** matakuliah **SET** sks = 2 **WHERE** nama = 'Kecerdasan Buatan';
  - **DELETE FROM** matakuliah **WHERE** kode = 'SPK';
  - **DELETE FROM** matakuliah **WHERE** kode = 'SPK' **OR** kode = 'ASD';

# KLAUSA 'WHERE'

**SELECT \* FROM matakuliah WHERE kode = 'ASD' ;**



```
[mysql> SELECT * FROM mahasiswa;
```

nim	nama	nomor_telepon
1	Ani Rahmawati	NULL
2	Budi Raharjo	NULL
3	Charlie Setiabudi	0859767553
4	Diandra Paramita	0858998745

```
[mysql> SELECT * FROM mahasiswa WHERE nim < 4;
```

nim	nama	nomor_telepon
1	Ani Rahmawati	NULL
2	Budi Raharjo	NULL
3	Charlie Setiabudi	0859767553

\*Klausu WHERE **membatasi** hasil query SELECT.



# UPDATE

- Digunakan untuk **mengubah/mengganti** nilai **RECORD/Baris** yang sudah ada pada suatu tabel.
- Klausa pembentuk:
  - **UPDATE**
  - **SET**
  - **WHERE**
- Format:
  1. **UPDATE** nama\_tabel **SET** nama\_kolom = nilai\_baru **WHERE** nama\_kolom\_patokan **[operator\_perbandingan]** nilai\_patokan;
  2. **UPDATE** nama\_tabel **SET** nama\_kolom1 = nilai\_baru1, nama\_kolom2 = nilai\_baru2, ...dst. **WHERE** nama\_kolom\_patokan **[operator\_perbandingan]** nilai\_patokan;
- **Operator perbandingan/comparison operator** dapat berupa:
  - =, <, >, <=, >=, <>

# UPDATE

- Contoh Format #1:
  - **UPDATE** nama\_tabel **SET** nama\_kolom = nilai\_baru **WHERE** nama\_kolom\_patokan [operator\_perbandingan] nilai\_patokan;
- Digunakan ketika kita ingin mengganti nilai suatu baris untuk **1 kolom** tertentu saja.

SQL:

```
UPDATE mahasiswa SET nomor_telepon = '0857550234'  
WHERE nim = 1;
```

Akan menghasilkan:

nim	nama	nomor_telepon
1	Ani Rahmawati	0857550234
2	Budi Raharjo	0858776453
3	Charlie Setiabudi	0859767553
4	Diandra Paramita	0858998745

# UPDATE

- Contoh Format #2:
  - **UPDATE** nama\_tabel **SET** nama\_kolom1 = nilai\_baru1, nama\_kolom2 = nilai\_baru2, ...dst. **WHERE** nama\_kolom\_patokan [operator\_perbandingan] nilai\_patokan;
- Digunakan ketika kita ingin mengganti nilai suatu baris untuk **beberapa** kolom sekaligus.

```
UPDATE matakuliah SET
  nama = 'Multimedia Terapan Tingkat Lanjut',
  sks = 3
WHERE kode = 'MMT';
```

Akan menghasilkan:

kode	nama	sks
ASD	Algoritma dan Struktur Data	NULL
BDD	Basis Data Dasar	NULL
KCB	Kecerdasan Buatan	NULL
MMT	Multimedia Terapan Tingkat Lanjut	3
PBO	Pemrograman Berorientasi Objek	NULL
SPK	Sistem Pendukung Keputusan	NULL

# DELETE

- Digunakan untuk **menghapus** suatu **RECORD/Baris** yang sebelumnya ada pada suatu tabel.
- Klausa pembentuk:
  - **DELETE**
  - **FROM**
  - **WHERE**
- Format:
  1. **DELETE FROM** nama\_tabel **WHERE** nama\_kolom\_patokan **[operator\_perbandingan]** nilai\_patokan;
  2. **DELETE \* FROM** nama\_tabel; atau **DELETE FROM** nama\_tabel;
- **Operator perbandingan/comparison operator** dapat berupa:
  - **=, <, >, <=, >=, <>**

# DELETE

- Contoh Format #1:
  - **DELETE FROM** nama\_tabel **WHERE** nama\_kolom\_patokan [operator\_perbandingan] nilai\_patokan;
- Digunakan ketika kita ingin menghapus suatu baris dengan **syarat** tertentu.

SQL:

```
DELETE FROM matakuliah WHERE kode = 'BDD';
```

Akan menghasilkan:

kode	nama	sks
ASD	Algoritma dan Struktur Data	NULL
KCB	Kecerdasan Buatan	NULL
MMT	Multimedia Terapan Tingkat Lanjut	3
PBO	Pemrograman Berorientasi Objek	NULL
SPK	Sistem Pendukung Keputusan	NULL



# DELETE

- Contoh Format #2:
  - **DELETE \* FROM** nama\_tabel; atau **DELETE FROM** nama\_tabel;
- Digunakan ketika kita ingin menghapus **semua** baris/records pada suatu tabel.
- **WARNING: Tidak dapat di-undo!**

SQL:

```
SET SQL_SAFE_UPDATES = 0;  
DELETE FROM matakuliah;  
SET SQL_SAFE_UPDATES = 1;
```

Akan menghasilkan:

```
[mysql> SELECT * FROM matakuliah;  
Empty set (0.00 sec)
```

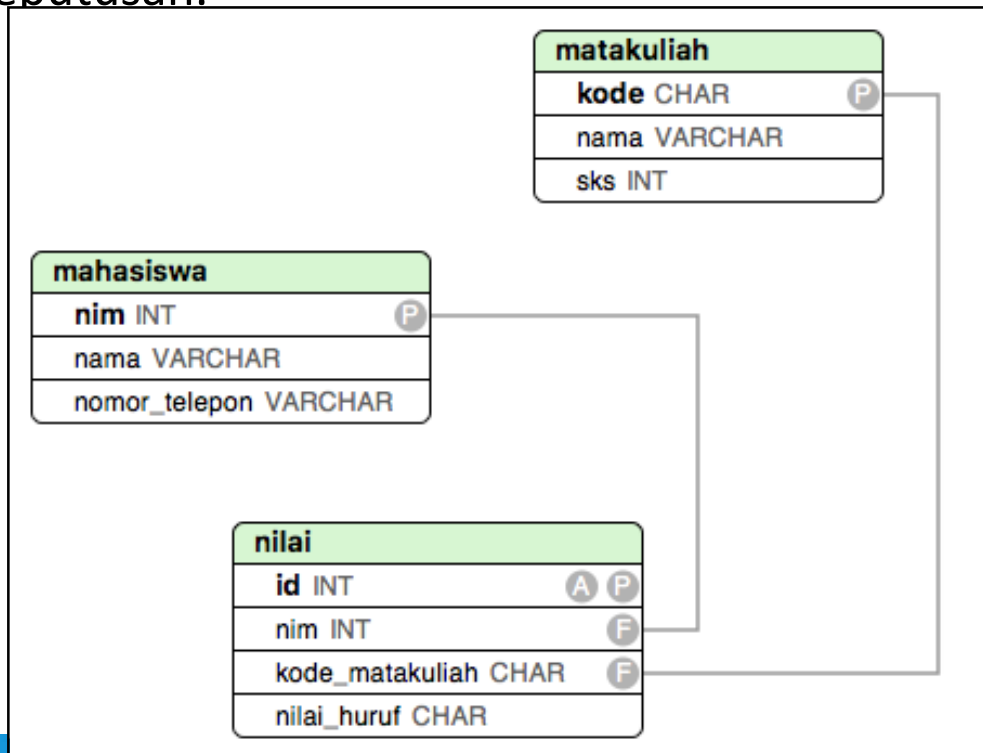
```
mysql> █
```

# PERTANYAAN??



# TUGAS LATIHAN

- Buatlah SQL untuk mengisi tabel nilai dengan 5 data yang valid!
- Buatlah SQL untuk mengisi SKS yang kosong pada tabel matakuliah sehingga semua matakuliah tersebut SKS-nya menjadi = 2.
- Buatlah SQL untuk menghapus data matakuliah dengan nama Sistem Pendukung Keputusan.





JURUSAN  
**TEKNOLOGI INFORMASI**  
POLITEKNIK NEGERI MALANG



# THANK YOU!



+62 (0341) 404424 – 404425



[HTTPS://JTI.POLINEMA.AC.ID/](https://jti.polinema.ac.id/)