PROGRAM STUDI
TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS DIAN NUSWANTORO

MATA KULIAH:
Sistem Basis Data

DISUSUN OLEH:
Tim Pengampu SBD TI

View and Index

Capaian Pembelajaran

- Mahasiswa mampu:
  - Membangun View
  - Membangkitkan Index

Pokok Bahasan

- View
- Index

Referensi

UTAMA

1. Silberschatz, A., Korth, H. F. & Sudarshan, S., 2022. Database System Concepts. 7th ed. New York: McGraw-Hill Education
2. Connolly, T. & Begg, C., 2015. Database Systems Apracticial Approach to Design, Implementation, and Management. Sixth Edition ed. s.l.:Pearson.
3. Elmasri, R. & Navathe, S. B., 2016. Fundamentalsmof Database Systems. 7th ed. s.l.:Pearson

PENDUKUNG

Aripin., 2005. *Praktikum Basis Data Dengan Database Server MySQL.* Semarang: Fakultas Ilmu Komputer

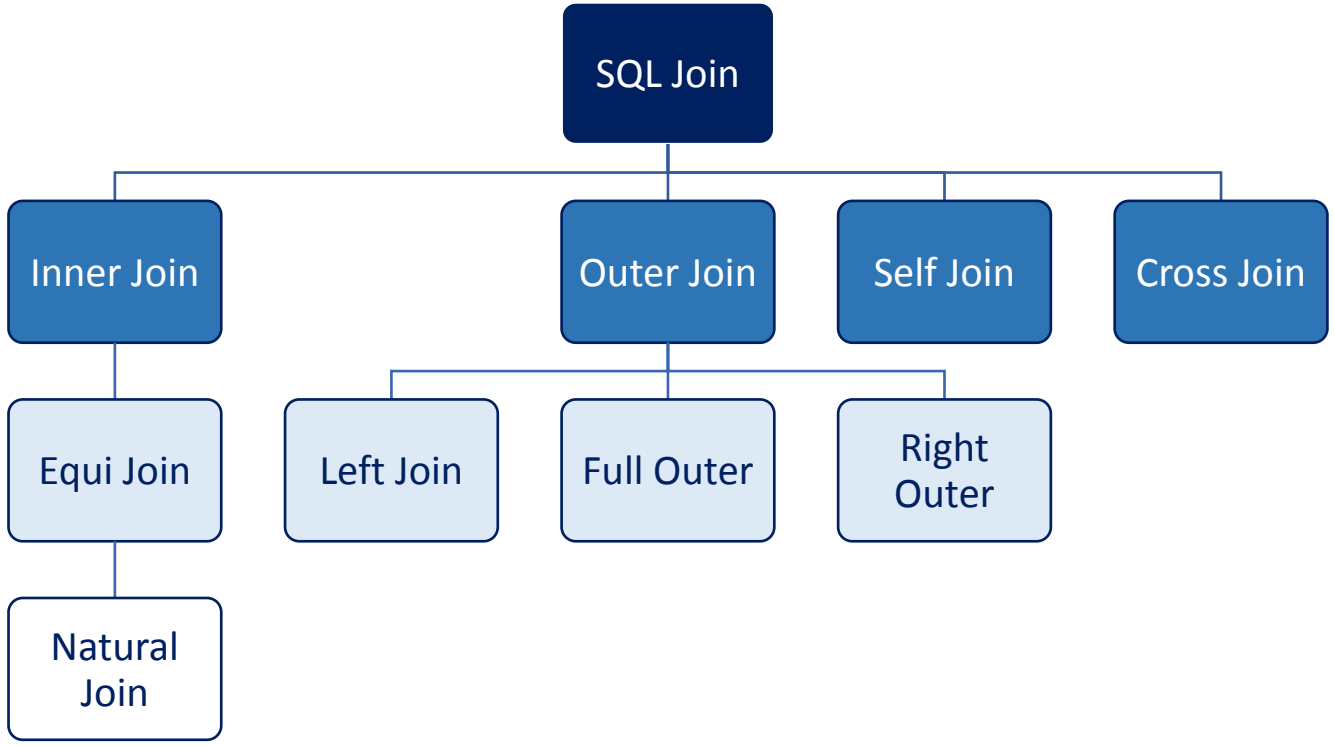
FAKULTAS ILMU KOMPUTER
UNIVERSITAS DIAN NUSWANTORO

Review SQL Join

SQL JOIN

Digunakan ketika kita ingin melakukan operasi SELECT pada 2 atau lebih tabel dari suatu database, dengan ketentuan:

- **Inner Join:** table A harus memiliki kolom-kolom yang memiliki nilai yang sama pada table B.
- **Outer Join:** table A tidak harus memiliki nilai yang sama pada table B.
- **Self Join:** menggabungkan table ke dirinya sendiri menggunakan bantuan operator alias.
- **Cross Join:** menggabungkan tabel-tabel dengan hasil berupa CARTESIAN PRODUCT antar tabel



SQL JOIN - *Persiapan*

Buat database baru (Latihan_2) dengan 2 buah tabel sederhana, yaitu:

- Tabel Kategori: berisi id kategori dan jenis produk
- Tabel Produk: berisi id produk, merk dan id kategori

```
MariaDB [(none)]> create database Latihan_2;
Query OK, 1 row affected (0.002 sec)

MariaDB [(none)]> use Latihan_2;
Database changed
```

```
MariaDB [Latihan_2]> CREATE TABLE Kategori
    -> (
    -> ID_Kategori CHAR(3) PRIMARY KEY,
    -> Jenis VARCHAR(20)
    -> );
Query OK, 0 rows affected (0.012 sec)

MariaDB [Latihan_2]> CREATE TABLE Produk
    -> (
    -> ID_Produk CHAR(3),
    -> Merk VARCHAR(20),
    -> ID_Kategori CHAR(3)
    -> );
Query OK, 0 rows affected (0.011 sec)
```

```
MariaDB [Latihan_2]> DESC Kategori;
+-------------+-------------+------+-----+---------+-------+
| Field       | Type        | Null | Key | Default | Extra |
+-------------+-------------+------+-----+---------+-------+
| ID_Kategori | char(3)     | NO   | PRI | NULL    |       |
| Jenis       | varchar(20) | YES  |     | NULL    |       |
+-------------+-------------+------+-----+---------+-------+
2 rows in set (0.005 sec)

MariaDB [Latihan_2]> DESC Produk;
+-------------+-------------+------+-----+---------+-------+
| Field       | Type        | Null | Key | Default | Extra |
+-------------+-------------+------+-----+---------+-------+
| ID_Produk   | char(3)     | NO   | PRI | NULL    |       |
| Merk        | varchar(20) | YES  |     | NULL    |       |
| ID_Kategori | char(3)     | YES  |     | NULL    |       |
+-------------+-------------+------+-----+---------+-------+
3 rows in set (0.005 sec)
```

Tabel 1: **Kategori**

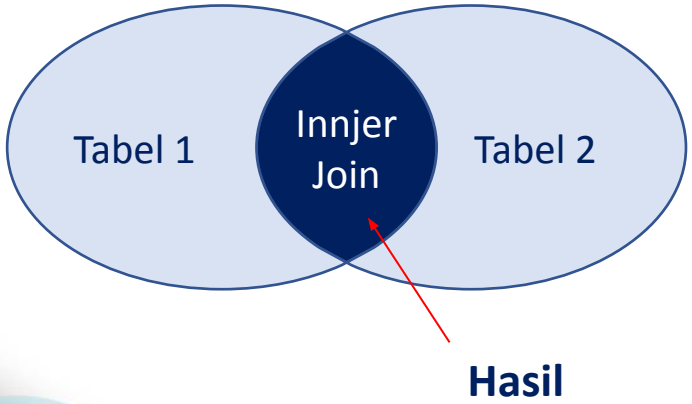
ID_Kategori*	Jenis
001	Handphone
002	Laptop
003	Tablet
004	Kamera
005	Gamming

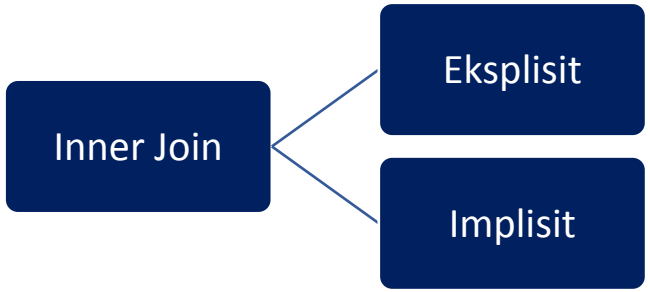
Tabel 2: **Produk**

ID_Produk*	Merk	ID_Kategori
P01	Xiaomi	001
P02	Samsung	001
P03	Asus	002
P04	Dell	002
P05	Apple	003
P06	Nikon	004
P07	Playstation	Null

Inner Join

- Digunakan untuk menampilkan baris-baris data yang sama pada 2 tabel atau lebih.
- Syarat suatu baris data akan dimunculkan adalah jika memiliki key yang sama (punya kolom penghubung).
- Pada table di samping, kolom penghubungnya adalah 'ID_Kategori'.
- EQUI JOIN merupakan subset dari INNER JOIN.
- Dalam menghubungkan 2 tabel, INNER JOIN mendukung penggunaan operator '=', '<', dan '>', sementara EQUI JOIN hanya mendukung operator '='.





Tabel 1: **Kategori**

ID_Kategori*	Jenis
001	Handphone
002	Laptop
003	Tablet
004	Kamera
005	Gamming

Tabel 2: **Produk**

ID_Produk*	Merk	ID_Kategori
P01	Xiaomi	001
P02	Samsung	001
P03	Asus	002
P04	Dell	002
P05	Apple	003
P06	Nikon	004
P07	Playstation	Null

Inner Join

```
MariaDB [Latihan_2]> SELECT * FROM Produk INNER JOIN Kategori
ON Produk.ID_Kategori = Kategori.ID_Kategori;
+-----------+---------+-------------+-------------+-----------+
| ID_Produk | Merk    | ID_Kategori | ID_Kategori | Jenis     |
+-----------+---------+-------------+-------------+-----------+
| P01       | Xiaomi  | 001         | 001         | Handphone |
| P02       | Samsung | 001         | 001         | Handphone |
| P03       | Asus    | 002         | 002         | Laptop    |
| P04       | Dell    | 002         | 002         | Laptop    |
| P05       | Apple   | 003         | 003         | Tablet    |
| P06       | Nikon   | 004         | 004         | Kamera    |
+-----------+---------+-------------+-------------+-----------+
6 rows in set (0.001 sec)


MariaDB [Latihan_2]> SELECT * FROM Produk, Kategori WHERE
Produk.ID_Kategori = Kategori.ID_Kategori;
+-----------+---------+-------------+-------------+-----------+
| ID_Produk | Merk    | ID_Kategori | ID_Kategori | Jenis     |
+-----------+---------+-------------+-------------+-----------+
| P01       | Xiaomi  | 001         | 001         | Handphone |
| P02       | Samsung | 001         | 001         | Handphone |
| P03       | Asus    | 002         | 002         | Laptop    |
| P04       | Dell    | 002         | 002         | Laptop    |
| P05       | Apple   | 003         | 003         | Tablet    |
| P06       | Nikon   | 004         | 004         | Kamera    |
+-----------+---------+-------------+-------------+-----------+
6 rows in set (0.001 sec)
```

Eksplisit Inner Join

Operator yang digunakan:
… INNER JOIN … ON …

Implisit Inner Join

Operator yang digunakan:
… WHERE …

Query yang dihasilkan dari eksplisit dan implisit Inner Join adalah sama

Tabel 1: **Kategori**

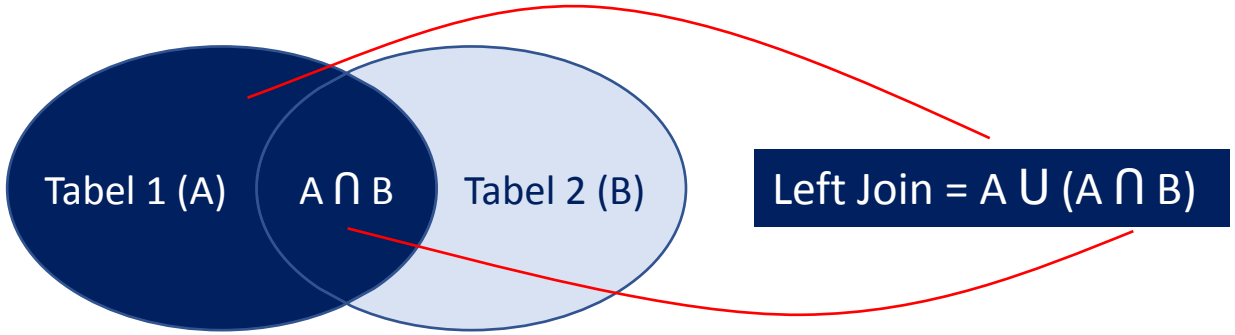
ID_Kategori*	Jenis
001	Handphone
002	Laptop
003	Tablet
004	Kamera
005	Gamming

Tabel 2: **Produk**

ID_Produk*	Merk	ID_Kategori
P01	Xiaomi	001
P02	Samsung	001
P03	Asus	002
P04	Dell	002
P05	Apple	003
P06	Nikon	004
P07	Playstation	Null

Left Join

- Left Join akan menghasilkan data dari table sebelah kiri (tabel yang disebut pertama pada perintah SELECT), tidak peduli data di table kiri memiliki kesamaan dengan table sebelah kanan (table yang disebut kedua) atau tidak.
- Pada Left Join ini juga akan menampilkan data-data dari table sebelah kanan yang memiliki kesamaan dengan table sebelah kiri.



Tabel 1: **Kategori**

ID_Kategori*	Jenis
001	Handphone
002	Laptop
003	Tablet
004	Kamera
005	Gamming

Tabel 2: **Produk**

ID_Produk*	Merk	ID_Kategori
P01	Xiaomi	001
P02	Samsung	001
P03	Asus	002
P04	Dell	002
P05	Apple	003
P06	Nikon	004
P07	Playstation	Null

Left Join

```
MariaDB [Latihan_2]> SELECT * FROM Produk LEFT OUTER JOIN Kategori
    -> ON Produk.ID_Kategori = Kategori.ID_Kategori;

+-----------+-------------+-------------+-------------+-----------+
| ID_Produk | Merk        | ID_Kategori | ID_Kategori | Jenis     |
+-----------+-------------+-------------+-------------+-----------+
| P01       | Xiaomi      | 001         | 001         | Handphone |
| P02       | Samsung     | 001         | 001         | Handphone |
| P03       | Asus        | 002         | 002         | Laptop    |
| P04       | Dell        | 002         | 002         | Laptop    |
| P05       | Apple       | 003         | 003         | Tablet    |
| P06       | Nikon       | 004         | 004         | Kamera    |
| P07       | Playstation | NULL        | NULL        | NULL      |
+-----------+-------------+-------------+-------------+-----------+
7 rows in set (0.000 sec)
```

```
MariaDB [Latihan_2]> SELECT
    -> Produk.ID_Produk, Produk.Merk, Produk.ID_Kategori, Kategori.Jenis
    -> FROM Produk LEFT OUTER JOIN Kategori
    -> ON Produk.ID_Kategori = Kategori.ID_Kategori;

+-----------+-------------+-------------+-----------+
| ID_Produk | Merk        | ID_Kategori | Jenis     |
+-----------+-------------+-------------+-----------+
| P01       | Xiaomi      | 001         | Handphone |
| P02       | Samsung     | 001         | Handphone |
| P03       | Asus        | 002         | Laptop    |
| P04       | Dell        | 002         | Laptop    |
| P05       | Apple       | 003         | Tablet    |
| P06       | Nikon       | 004         | Kamera    |
| P07       | Playstation | NULL        | NULL      |
+-----------+-------------+-------------+-----------+
7 rows in set (0.001 sec)
```

Dengan nama kolom disebutkan satu per satu, maka tabel query yang terbentuk menjadi lebih sederhana

Tabel 1: **Kategori**

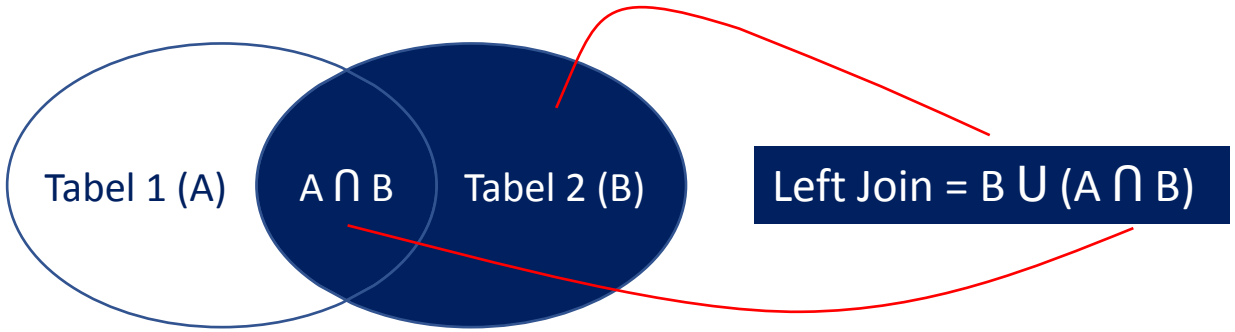
ID_Kategori*	Jenis
001	Handphone
002	Laptop
003	Tablet
004	Kamera
005	Gamming

Tabel 2: **Produk**

ID_Produk*	Merk	ID_Kategori
P01	Xiaomi	001
P02	Samsung	001
P03	Asus	002
P04	Dell	002
P05	Apple	003
P06	Nikon	004
P07	Playstation	Null

Right Join

- Right Join akan menghasilkan seluruh data dari table sebelah kanan (tabel yang disebut kedua pada perintah SELECT), tidak peduli data di table kanan memiliki kesamaan dengan table sebelah kiri (table yang disebut pertama) atau tidak.
- Pada Right Join ini juga akan menampilkan data-data dari table sebelah kiri yang memiliki kesamaan dengan table sebelah kanan.



Tabel 1: **Kategori**

ID_Kategori*	Jenis
001	Handphone
002	Laptop
003	Tablet
004	Kamera
005	Gamming

Tabel 2: **Produk**

ID_Produk*	Merk	ID_Kategori
P01	Xiaomi	001
P02	Samsung	001
P03	Asus	002
P04	Dell	002
P05	Apple	003
P06	Nikon	004
P07	Playstation	Null

Right Join

```
MariaDB [latihan_2]> SELECT *
    -> FROM Produk RIGHT OUTER JOIN Kategori
    -> ON Produk.ID_Kategori = Kategori.ID_Kategori;

+-----------+---------+-------------+-------------+-----------+
| ID_Produk | Merk    | ID_Kategori | ID_Kategori | Jenis     |
+-----------+---------+-------------+-------------+-----------+
| P01       | Xiaomi  | 001         | 001         | Handphone |
| P02       | Samsung | 001         | 001         | Handphone |
| P03       | Asus    | 002         | 002         | Laptop    |
| P04       | Dell    | 002         | 002         | Laptop    |
| P05       | Apple   | 003         | 003         | Tablet    |
| P06       | Nikon   | 004         | 004         | Kamera    |
| NULL      | NULL    | NULL        | 005         | Gamming   |
+-----------+---------+-------------+-------------+-----------+
7 rows in set (0.015 sec)
```

```
MariaDB [latihan_2]> SELECT
    -> Produk.ID_Produk, Produk.Merk, Produk.ID_Kategori, Kategori.Jenis
    -> FROM Produk RIGHT OUTER JOIN Kategori
    -> ON Produk.ID_Kategori = Kategori.ID_Kategori;
+-----------+---------+-------------+-----------+
| ID_Produk | Merk    | ID_Kategori | Jenis     |
+-----------+---------+-------------+-----------+
| P01       | Xiaomi  | 001         | Handphone |
| P02       | Samsung | 001         | Handphone |
| P03       | Asus    | 002         | Laptop    |
| P04       | Dell    | 002         | Laptop    |
| P05       | Apple   | 003         | Tablet    |
| P06       | Nikon   | 004         | Kamera    |
| NULL      | NULL    | NULL        | Gamming   |
+-----------+---------+-------------+-----------+
7 rows in set (0.001 sec)
```

Dengan nama kolom disebutkan satu per satu, maka tabel query yang terbentuk menjadi lebih sederhana

Tabel 1: **Kategori**

ID_Kategori*	Jenis
001	Handphone
002	Laptop
003	Tablet
004	Kamera
005	Gamming

Tabel 2: **Produk**

ID_Produk*	Merk	ID_Kategori
P01	Xiaomi	001
P02	Samsung	001
P03	Asus	002
P04	Dell	002
P05	Apple	003
P06	Nikon	004
P07	Playstation	Null

FAKULTAS ILMU KOMPUTER
UNIVERSITAS DIAN NUSWANTORO

VIEW

Definisi View

- Sebuah view dalam terminologi SQL adalah sebuah tabel tunggal yang diturunkan dari tabel-tabel yang lain.
- Sebuah view tidak harus ada dalam bentuk fisik, view dianggap sebagai tabel virtual berbeda dengan tabel dasar yang tupelnya selalu disimpan secara fisik di dalam database.
- Hal ini membatasi operasi pembaruan yang dapat diterapkan pada view, tetapi tidak memberikan batasan pada kueri view.

(Elmasri & Navathe, 2016)

Definisi View – lanjut

- View didefinisikan pada SQL dengan menggunakan perintah `create view`, untuk mendefinisikan *view* kita harus memberi nama *view* dan harus menyatakan query yang menyatakan *view*.
- Bentuk perintah create view adalah:

```
create view V as <query expression>;
```

- di mana <ekspresi kueri> adalah ekspresi kueri yang sah. Nama view diwakili oleh `V`.

- Dengan menggunakan database **Latihan_3, tabel Pelanggan.** Buatlah view untuk menampilkan daftar pelanggan yang berasal dari Jawa Tengah. Berdasarkan data, kota yang termasuk Jawa Tengah dari pelanggan meliputi: Solo, Semarang, Pemalang
- Lakukanlah dahulu **query** untuk menampilkan informasi pelanggan yang berasal dari Jawa Tengah

```
MariaDB [latihan_3]> select * from pelanggan;
+-----+--------+------+-----------+---------+
| id  | nama   | usia | kota      | gaji    |
+-----+--------+------+-----------+---------+
| 001 | Nathan |   35 | Surabaya  | 6250000 |
| 002 | Andini |   25 | Solo      | 9000000 |
| 003 | Sita   |   23 | Bandung   | 8000000 |
| 004 | Soni   |   25 | Bekasi    | 5000000 |
| 005 | Dedi   |   27 | Palembang | 6000000 |
| 006 | Ega    |   22 | Semarang  | 5000000 |
| 007 | Vika   |   24 | Pemalang  | 7000000 |
+-----+--------+------+-----------+---------+
7 rows in set (0.000 sec)
```

```
MariaDB [latihan_3]> select * from pelanggan where kota IN('Solo','Semarang','Pemalang');
+-----+--------+------+----------+---------+
| id  | nama   | usia | kota     | gaji    |
+-----+--------+------+----------+---------+
| 002 | Andini |   25 | Solo     | 9000000 |
| 006 | Ega    |   22 | Semarang | 5000000 |
| 007 | Vika   |   24 | Pemalang | 7000000 |
+-----+--------+------+----------+---------+
3 rows in set (0.001 sec)
```

Contoh Pembuatan View

- Selanjutnya buatlah View sesuai query yang telah ditulis sebelumnya sbg *query expression*
- Syntax: create view **ViewName** as <query expression>;
- Create view **pelanggan_jateng** as select * from pelanggan where kota IN('Solo','Semarang','Pemalang');

```
MariaDB [latihan_3]> create view pelanggan_jateng as
    -> select * from pelanggan where kota IN('Solo','Semarang','Pemalang');
Query OK, 0 rows affected (0.003 sec)
```

Pemanggilan View

View dapat dipanggil menggunakan perintah Select seperti pada Tabel

Syntax:

```
SELECT [ * | column1 [, column2] ]
FROM VIEW
[WHERE VALUE OPERATOR]
```

Memanggil view

```
Select * from
pelanggan_jateng
```

```
MariaDB [latihan_3]> select * from pelanggan_jateng;
+-----+--------+------+----------+---------+
| id  | nama   | usia | kota     | gaji    |
+-----+--------+------+----------+---------+
| 002 | Andini |   25 | Solo     | 9000000 |
| 006 | Ega    |   22 | Semarang | 5000000 |
| 007 | Vika   |   24 | Pemalang | 7000000 |
+-----+--------+------+----------+---------+
3 rows in set (0.001 sec)
```

Pemanggilan View Dengan Kolom Tertentu

Misal: Menampilkan nama dari view pelanggan_jateng

Syntax:

```
SELECT [ * | column1 [, column2] ]
FROM VIEW
[WHERE VALUE OPERATOR]
```

Contoh menanggil view dengan kolom tertentu

```
Select nama from
pelanggan_jateng
```

```
MariaDB [latihan_3]> select nama from pelanggan_jateng;
+--------+
| nama   |
+--------+
| Andini |
| Ega    |
| Vika   |
+--------+
3 rows in set (0.001 sec)
```

Pemanggilan View Dengan Kondisi Tertentu

Misal: Menampilkan nama, dan kota dari view pelanggan_jateng dengan gaji di atas 5.000.000

Syntax:

```
SELECT [ * | column1 [, column2] ]
FROM VIEW
[WHERE VALUE OPERATOR]
```

Menanggil view dengan kondisi tertentu

```
Select nama, gaji
from
pelanggan_jateng
where gaji>5000000;
```

```
MariaDB [latihan_3]> select nama, kota from pelanggan_jateng
    -> where gaji > 5000000;
+--------+----------+
| nama   | kota     |
+--------+----------+
| Andini | Solo     |
| Vika   | Pemalang |
+--------+----------+
2 rows in set (0.001 sec)
```

Menampilkan Daftar View

- Daftar view yang telah kita buat dapat ditampilkan dengan menggunakan syntax
- `SHOW FULL TABLES IN database_name WHERE TABLE_TYPE LIKE 'VIEW';`

Mengubah View

- View memiliki keterbatasan pada pendefinisian operasi yang diterapkan.
- Misal terdapat data pelanggan baru dari kota Pekalongan (termasuk Jawa Tengah), maka apabila view Pelanggan_Jateng dipanggil tidak akan menampilkan data baru tersebut.
- Maka view perlu diubah menyesuaikan perubahan query tersebut

```
MariaDB [latihan_3]> insert into pelanggan
    -> values('008','Putri', 28, 'Pekalongan',5000000);
Query OK, 1 row affected (0.003 sec)

MariaDB [latihan_3]> select * from pelanggan_jateng;
+-----+--------+-------+----------+---------+
| id  | nama   | usia  | kota     | gaji    |
+-----+--------+-------+----------+---------+
| 002 | Andini |    25 | Solo     | 9000000 |
| 006 | Ega    |    22 | Semarang | 5000000 |
| 007 | Vika   |    24 | Pemalang | 7000000 |
+-----+--------+-------+----------+---------+
3 rows in set (0.001 sec)
```

Mengubah View

Mengubah view dapat dilakukan menggunakan perintah ALTER

Syntax:

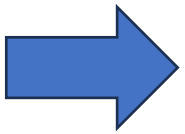
```
ALTER VIEW <viewName> AS <query expression>
```

Penerapan:

```
ALTER VIEW pelanggan_jateng as select * from
pelanggan where kota
IN('Solo','Semarang','Pemalang','Pekalongan')
```

```
MariaDB [latihan_3]> alter view pelanggan_jateng as
    -> select * from pelanggan where
    -> kota IN('Solo','Semarang','Pemalang','Pekalongan');
Query OK, 0 rows affected (0.007 sec)
```

Setelah view diubah maka data baru dapat tampil



```
MariaDB [latihan_3]> select * from pelanggan_jateng;
+-----+--------+-------+-----------+---------+
| id  | nama   | usia  | kota      | gaji    |
+-----+--------+-------+-----------+---------+
| 002 | Andini |    25 | Solo      | 9000000 |
| 006 | Ega    |    22 | Semarang  | 5000000 |
| 007 | Vika   |    24 | Pemalang  | 7000000 |
| 008 | Putri  |    28 | Pekalongan| 5000000 |
+-----+--------+-------+-----------+---------+
4 rows in set (0.001 sec)
```

Menghapus View

Agar dapat menghapus view maka dapat diperintahkan Drop seperti menghapus tabel

Syntax:

```
DROP VIEW <viewName>
```

Penerapan:

```
Drop view pelanggan_jateng;
```

```
MariaDB [latihan_3]> DROP VIEW pelanggan_jateng;
Query OK, 0 rows affected (0.001 sec)
```

Latihan View

Berdasarkan database **Latihan_2** yang ditelah dibuat sebelumnya:

1. Buatlah view bernama **produk_handphone** dengan query yang menghasilkan informasi **id_produk, merk, jenis** dari gabungan 2 tabel (produk dan merk)
2. Buatlah view bernama **produk_laptop** dengan query yang menghasilkan informasi **id_produk, merk, jenis** dari gabungan 2 tabel (produk dan merk)

Index

Pengertian Index

- Indeks dapat diilustrasikan sebagai indeks buku, sehingga melalui indeks buku tersebut maka dapat dilakukan pencarian letak item tertentu dalam buku dengan mudah.
- Keberadaan indeks dalam basis data antara lain adalah untuk mempercepat pencarian data berdasarkan kolom tertentu misalnya diberikan perintah :
- Mysql> Select * from staf where nip = ‘001’;

Pengertian Index

- Jika nip tidak dijadikan sebagai indeks, pencarian data akan dilakukan terhadap seluruh tabel, sama seperti jika akan mencari sesuatu dalam buku tetapi buku tersebut tidak dilengkapi dengan indeks.
- Namun jika indeks yang berkaitan dengan nip ditemukan, maka sistem akan menampilkan hasilnya dengan cepat.
- Beberapa hal yang sangat terbantu dengan adanya indeks adalah :
  - Proses penggabungan sejumlah tabel
  - Proses dengan ORDER BY
  - Proses fungsi agregat seperti MIN dan MAX

Menciptakan Index

- Untuk membuat file indeks menggunakan perintah **CREATE INDEX**
- Misalnya terdapat sebuah tabel dengan perintah pembuatannya adalah sebagai berikut :
- Mysql> Create Table Pelamar (nama char(20) not null, no_tes char(3) not null);
- Maka kita dapat membuat indeks dari tabel tersebut, seperti :
- Mysql> **Create Index no_tes_idx on pelamar (no_tes);**

Menciptakan Index

Misalkan:

Mysql> **Create Index no_tes_idx on pelamar (no_tes);**

Keterangan :

- No_tes_idx adalah nama indeks yang dibuat
- Pelamar adalah nama tabel yang diindeks
- No_tes yang berada di dalam tanda kurung menyatakan kolom / field / atribut yang digunakan untuk mengindeks.

Latihan Index

Berdasarkan database Latihan_3 buatlah Index pada kolom nip dengan nama **index nip_idx** pada tabel staf

Nama table: **staf**

nip	nama	posisi	tgl_masuk	Gaji
A01	Wahyu	Asisten	2014-08-10	5000000
M01	Budi	Manager	1996-01-01	10000000
S01	Sari	Supervisor	2000-02-05	7500000

Menciptakan Index yang Unik

- Untuk menciptakan indeks yang unik maka harus ditambahkan kata UNIQUE diantara create dan index.
- Misalkan kita akan membuat index unik dengan nama no_tes_idx pada tabel pelamar berdasarkan kolom no_tes

Contoh :

- Mysql> Create unique index no_tes_idx on pelamar (no_tes);

Menciptakan Index yang Unik

- Menciptakan indeks yang unik dengan nama id_idx pada tabel pelanggan
- `Create unique index id_idx on pelanggan(id);`

```
MariaDB [latihan]> create unique index id_idx on pelanggan(id);
Query OK, 0 rows affected (0.014 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

Menghapus Index

- Menghapus index dapat dilakukan dengan perintah `Drop Index`
- Secara lengkap syntax penulisan penghapus index sebagai berikut:

```
Drop index nama_index on nama_tabel
```

- Apabila akan menghapus index Bernama id_idx pada tabel pelanggan maka dapat dituliskan

```
MariaDB [latihan]> drop index id_idx on pelanggan;
Query OK, 0 rows affected (0.010 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

TERIMA KASIH

ANY QUESTIONS?