

DATABASE SQL



Pesantren PeTIK II YBM PLN

Jl. KH. Bisri Syansuri RT/01 RW/05, Plosogeneng,
Kec. Jombang, Kabupaten Jombang, Jawa Timur



Pertemuan Ke-11





Materi

1. Pengantar Database
2. Pemodelan Data
3. Model Relasional Database
4. Normalisasi Database
5. Pengantar SQL
6. Perintah SQL SELECT 1
7. Perintah SQL SELECT 2
9. Fungsi Aggregate dan Grouping Data
10. Sub Query & SQL Join Table
- 11. View dan Analisa Query**
12. Store Procedure dan Function
13. Trigger dan Transaction
14. Manajemen User
15. Backup dan Restore



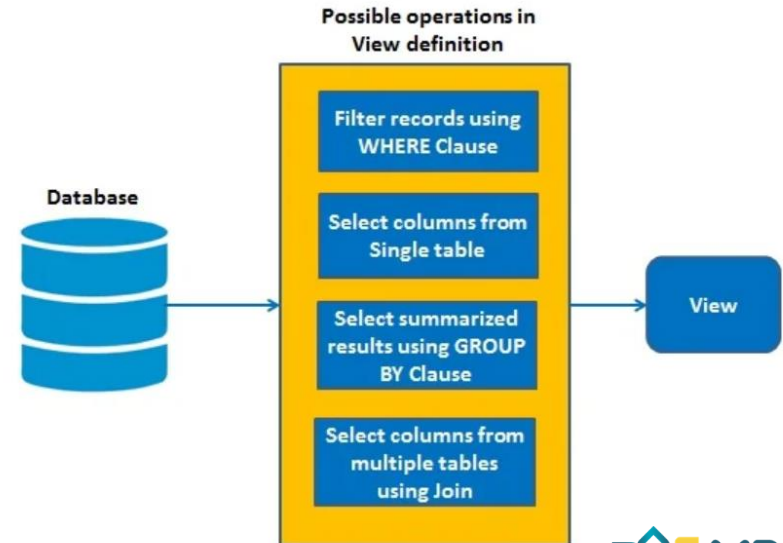
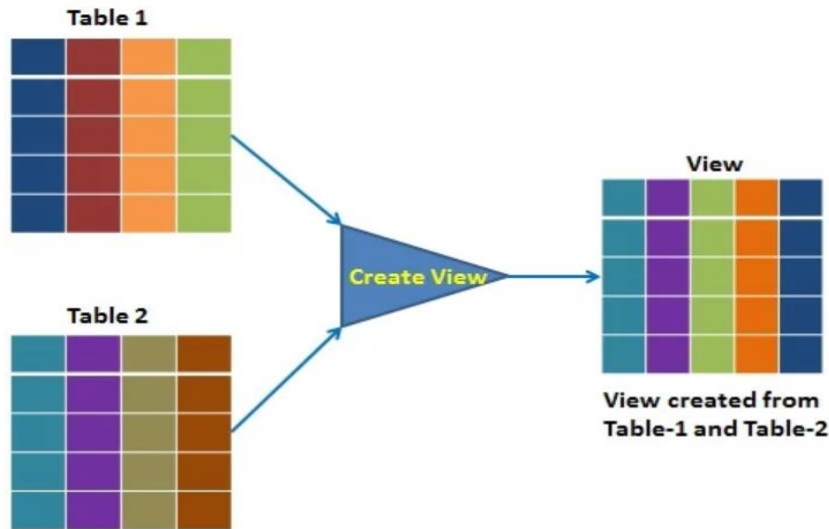
11. View dan Analisa Query





VIEW

- ❑ VIEW: Tabel baru subset dari suatu tabel yang telah ada
- ❑ View membatasi akses pengguna ke tabel-tabel
- ❑ Dibuat dengan perintah CREATE VIEW



<https://www.datacamp.com/community/tutorials/views-in-sql>





Terminologi View

- ❑ **Tabel Asal (*Base Table*)**
 - Tabel yang berisi data aslinya
- ❑ **View Dinamis (*Dynamic View*)**
 - “Tabel virtual” diciptakan secara dinamis atas permintaan pengguna/program.
 - Data tabel baru ini **tidak** disimpan secara fisik
 - Dihasilkan dari perintah SQL: SELECT atas tabel-tabel asal atau view-view lain
- ❑ **Beberapa Operasi Query SELECT untuk dijadikan VIEW:**
 - Filter data menggunakan klausa WHERE
 - Menampilkan data kolom dari satu table
 - Menampilkan summary data menggunakan klausa GROUP BY
 - Menampilkan data kolom dari banyak table menggunakan klausa JOIN





Sintaks CREATE VIEW

Perintah membuat VIEW:

CREATE

[OR REPLACE]

[ALGORITHM = {UNDEFINED | MERGE | TEMPTABLE}]

[DEFINER = { user | CURRENT_USER }]

[SQL SECURITY { DEFINER | INVOKER }]

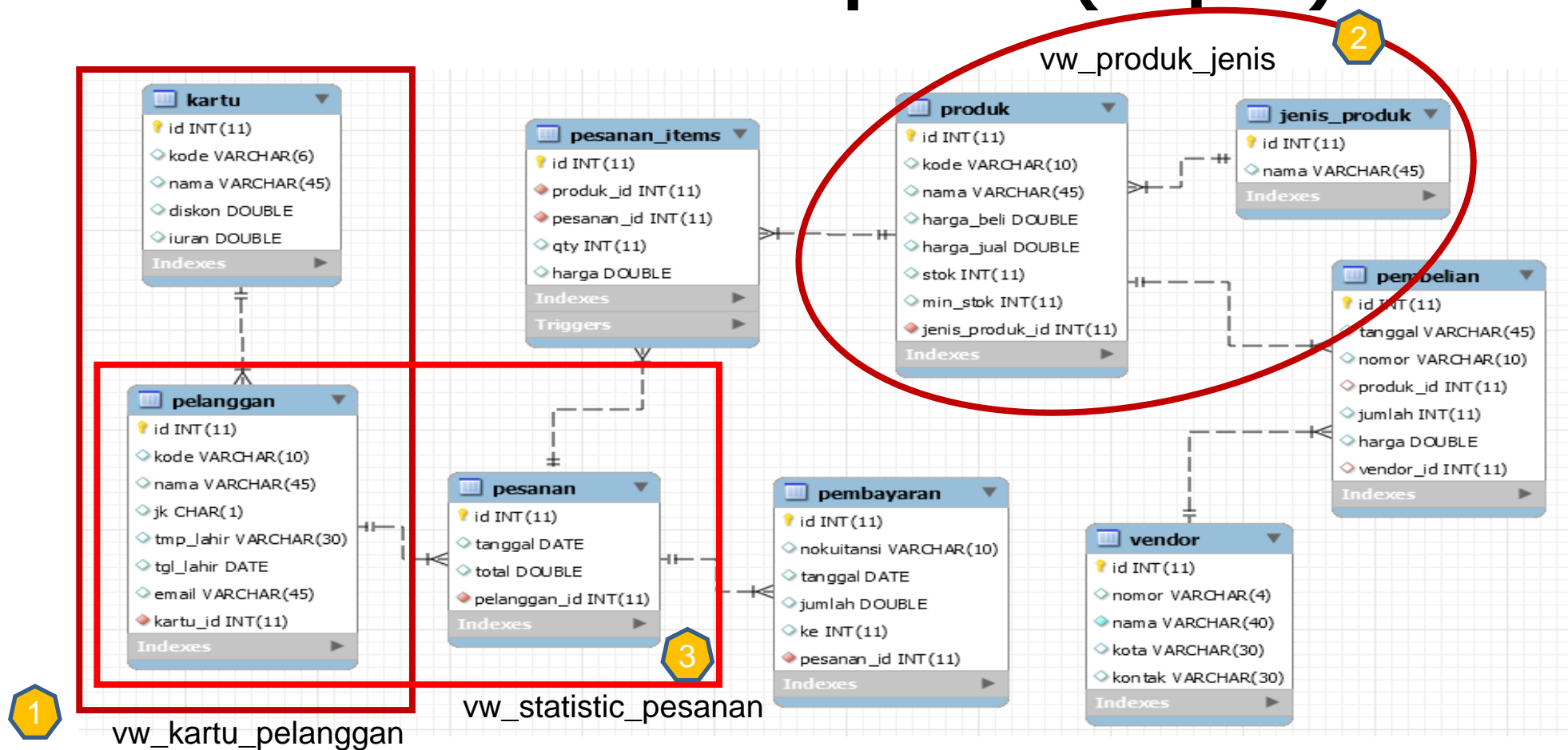
VIEW view_name [(column_list)]

AS select_statement

[WITH [CASCADED | LOCAL] CHECK OPTION]



ERD Database Koperasi (dbpos)





Contoh 1:

- View untuk menampilkan data pelanggan dan kartu anggotanya :

```
mysql> CREATE VIEW vw_kartu_pelanggan AS  
        SELECT a.kode,a.nama as pelanggan,  
        a.email,b.kode as kode_kartu,b.nama AS  
        kartu,b.iuran, b.diskon  
        FROM pelanggan a INNER JOIN kartu b  
        ON a.kartu_id=b.id;
```

- Eksekusi VIEW:

```
mysql> select * from vw_kartu_pelanggan;
```





Contoh 2:

- View untuk menampilkan data produk dan jenis produknya:

```
mysql> CREATE VIEW vw_produk_jenis AS  
        SELECT a.id, a.nama, a.harga_jual, a.stok, a.min_stok,  
               b.nama as jenis  
        FROM produk a INNER JOIN jenis_produk b  
        ON a.jenis_produk_id=b.id;
```

- Eksekusi VIEW:

```
mysql> select * from vw_produk_jenis;
```





Contoh 3:

- Tampilkan data statistik pelanggan dan berapa kali lakukan pesanan, pelanggan yang tidak melakukan pesanan juga dimunculkan

Pelanggan		jumlah_pesanan	total_nilai_transaksi
id	nama	count(pesanan.id)	sum(pesanan.total)

- View untuk menampilkan data statistic pesanan pelanggan:

```

mysql> CREATE VIEW vw_statistic_pesanan AS
      SELECT  a.id,a.nama,count(b.id) as jumlah,
              COALESCE(sum(b.total),0) as total_nilai_transaksi
      FROM    pelanggan a LEFT JOIN pesanan b
              ON a.id=b.pelanggan_id
              GROUP BY a.id,a.nama ORDER BY jumlah DESC;
    
```





EXPLAIN

- ❑ Perintah EXPLAIN adalah sinonim perintah dari DESCRIBE untuk melihat informasi dari table
- ❑ EXPLAIN dapat digunakan untuk melihat informasi dari eksekusi query SELECT
- ❑ Query dapat dianalisa untuk tujuan evaluasi query agar lebih cepat prosesnya

- ❑ Contoh :

```
mysql> explain
```

```
-> select * from table_name WHERE field_1 = 'TEST' \G;
```



EXPLAIN

```
mysql> explain select * from produk;
```

id	select_type	table	type	possible_keys	key	key_len	ref	rows	Extra
1	SIMPLE	produk	ALL	NULL	NULL	NULL	NULL	7	

- ❑ **table** : nama table
- ❑ **type** : tipe join yang digunakan (dari yang paling baik hingga yang paling buruk adalah system, const, eq_ref, ref, range, index, all)
- ❑ **possible_key** : index yang mungkin digunakan oleh table
- ❑ **key** : kunci yang digunakan secara aktual
- ❑ **key_len** : panjang yang digunakan. Semakin pendek sebuah kunci, semakin baik hasilnya
- ❑ **ref** : menunjukan sebuah kolom atau konstan yang digunakan
- ❑ **rows** : jumlah baris yang ditelusuri oleh mysql untuk memperoleh output data
- ❑ **extra** : merupakan extra info-info yang buruk disini adalah "using temporary " dan "using filesort"

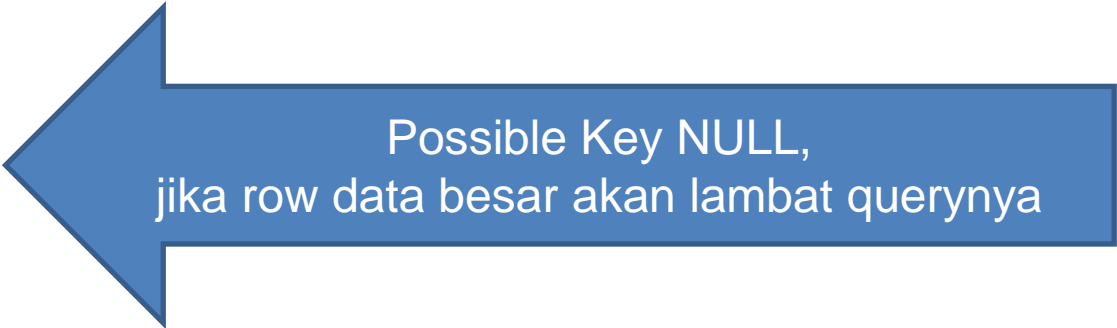




Explain

```
explain select * from produk where nama LIKE 'Note%' \G;  
***** 1. row *****
```

```
id: 1  
select_type: SIMPLE  
table: produk  
type: ALL  
possible_keys: NULL  
key: NULL  
key_len: NULL  
ref: NULL  
rows: 10  
Extra: Using where  
1 row in set (0.010 sec)
```



Possible Key NULL,
jika row data besar akan lambat querynya





Index

- ❑ Digunakan untuk mempercepat proses query
- ❑ Dikenakan pada suatu field / column
- ❑ MySQL akan membangun index dimana query pencarian dapat berjalan dengan cepat
- ❑ Index sebaiknya digunakan untuk:
 - ❑ kolom yang sering digunakan untuk kriteria pencarian bagian dari klausa WHERE
 - ❑ kolom yang biasanya juga diterapkan pada bagian ORDER BY
 - ❑ kolom dengan banyak nilai yang berbeda sebaiknya tidak di index





Jenis Index

- ❑ Primary Key : akan otomatis di index
- ❑ Unique Key : akan otomatis di index
- ❑ Non Unique
- ❑ Perintah melihat index pada table

mysql> show index from nama_table;





Create Index

Saat buat Table: Primary Key & Unique Key

```
CREATE TABLE books (  
    id integer auto_increment,  
    title VARCHAR(100) NOT NULL,  
    author VARCHAR(100) NOT NULL,  
    published_date TIMESTAMP NOT NULL,  
    isbn INT,  
    PRIMARY KEY (id),  
    UNIQUE (isbn)  
);
```





Index

- ❑ Sintaks CREATE INDEX :

```
CREATE [ONLINE|OFFLINE] [UNIQUE|FULLTEXT|SPATIAL] INDEX  
    index_name  
    [index_type]  
    ON tbl_name (index_col_name,...)  
    [index_option]
```

- ❑ Contoh Buat INDEX:

```
CREATE INDEX index_idkartu ON pelanggan(kartu_id);
```

- ❑ Contoh Hapus INDEX:

```
DROP INDEX index_idkartu ON pelanggan;
```





Sebelum di Index

```
explain select * from pelanggan where tmp_lahir='Jakarta' \G;
***** 1. row *****
```

```
      id: 1
  select_type: SIMPLE
        table: pelanggan
         type: ALL
possible_keys: NULL
          key: NULL
       key_len: NULL
         ref: NULL
        rows: 10
      Extra: Using where
1 row in set (0.000 sec)
```





Create Index kolom tmp_lahir

```
CREATE INDEX idx_tmplahir ON pelanggan(tmp_lahir);
desc pelanggan;
```

Field	Type	Null	Key	Default	Extra
id	int(11)	NO	PRI	NULL	auto_increment
kode	varchar(10)	YES		NULL	
nama	varchar(45)	YES	MUL	NULL	
jk	char(1)	YES		NULL	
tmp_lahir	varchar(30)	YES	MUL	NULL	
tgl_lahir	date	YES	MUL	NULL	
email	varchar(45)	YES		NULL	
kartu_id	int(11)	NO	MUL	NULL	

```
show index from pelanggan;
```





Setelah di Index

```
explain select * from pelanggan where tmp_lahir='Jakarta' \G;  
***** 1. row *****
```

```
      id: 1  
select_type: SIMPLE  
      table: pelanggan  
      type: ALL  
possible_keys: NULL  
      key: NULL  
    key_len: NULL  
      ref: NULL  
     rows: 10  
  Extra: Using where  
1 row in set (0.000 sec)
```



```
      id: 1  
select_type: SIMPLE  
      table: pelanggan  
      type: ref  
possible_keys: idx_tmplahir  
      key: idx_tmplahir  
    key_len: 93  
      ref: const  
     rows: 2  
  Extra: Using index condition  
1 row in set (0.001 sec)
```



**TERIMA KASIH
ATAS SEGALA PERHATIAN
SEMOGA BERMANFAAT...**

