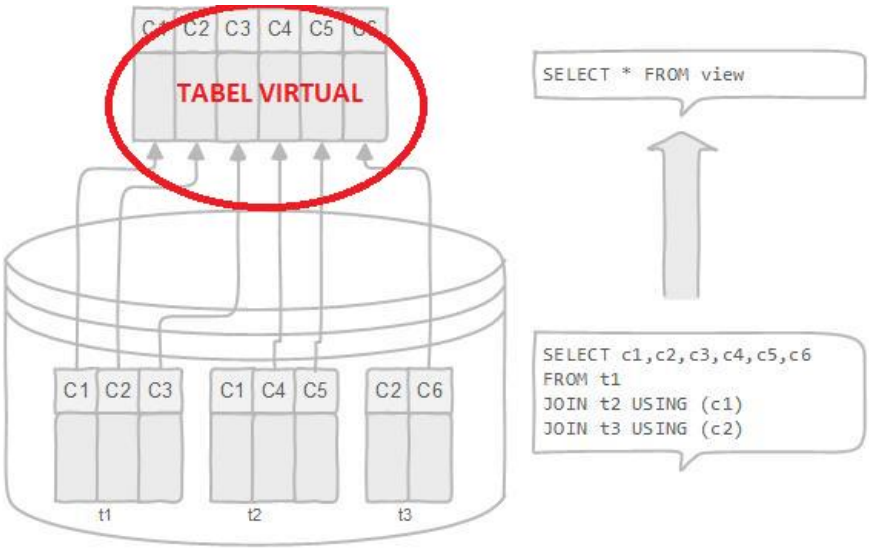
Database Views

18 December 2017
7:47

Database views adalah sebuah **TABEL VIRTUAL** atau TABLE LOGIC yang didefinisikan dengan menggunakan perintah query SELECT. Karena DATABASE VIEW serupa dengan TABLE biasa, yang terdiri atas BARIS dan KOLOM, sehingga kita dapat menerapkan perintah QUERY yang lain terhadap database VIEW.

VIEW bersifat dinamic karena tidak terkait dengan TABEL FISIK, MySQL (sistem database) menyimpan VIEW sebagai statemen SQL SELECT dengan atau tanpa JOIN.



KEUNTUNGAN menggunakan VIEW:

- **Dapat menyederhanakan perintah SQL yang kompleks** (tidak menggantikan). SQL kompleks kita bisa simpan sebagai VIEWS, selanjutnya untuk membaca SQL kompleks tersebut dapat dilakukan dengan menggunakan VIEWS (sebagai TABEL virtual). Selain itu juga dapat mempercepat proses pencaraian data, performan lebih baik.
- Dapat membantu untuk akses data oleh USER tertentu yang tidak dibolehkan melihat data dari sebuah tabel secara langsung.
- Dapat digunakan sebagai alternatif untuk kemana akses data.
- Dapat digunakan untuk membuat sebuah KOLOM yang merupakan hasil perhitungan dari kolom-kolom yang lain, bahkan dari kolom yang berasal dari TABEL yang lain.
- Dapat digunakan untuk menjaga kompatibilitas versi database server.

KERUGIAN menggunakan VIEW:

- Performansi: VIEW dapat menyebabkan performa mejadi menurun ketika dalam sebuah VIEW melibatkan VIEW yang lain. Jika VIEW dibuat dengan melibatkan satu atau beberapa VIEW yang lain
- **Ketergantungan pada TABEL FISIK**: Jika TABEL fisik yang terlibat dalam VIEW mengalamai perubahan STRUKTUR maka VIEW harus didefinisikan ULANG. VIEW harus selalu di CREATE ULANG ketika terjadi perubahan struktur dalam satu atau semua tabel fisik.

Bagaimana MySQL menyimpan VIEW dalam sistem database:

VIEW mulai disediakan sejak versi 5+, dan VIEW ini sesuai dengan STANDARD SQL: 2003. Terdapat DUA cara bagaimana MySQL memproses VIEW.

- MySQL membuat sebuah TABLE TEMPORARY (sementara) berdasarkan pada perintah SQL SELECT (dalam definisi VIEW) kemudian mengeksekusi query yang beroperasi pada tabel tersebut.
- MySQL mengeksekusi secara langsung perintah SQL query. Seperti mengeksekusi perintah cascade select. Select * from (select * from nama_tabel).

MEMBUAT VIEW

```
CREATE
    [ALGORITHM = {MERGE  | TEMPTABLE | UNDEFINED}]
VIEW [database_name].[view_name]
AS
[SELECT  statement]
```

Tersedia TIGA macam **ALGORITMA**:

- MERGE : Mysql akan menggabung preintah select dengan definisi perintah select dalam view, kemudian mengeksekusinya, (Tidak ada TABEL TEMPORARY)
- TEMPTABLE: MySQL akan membuat TABEL TEMPORARY pada saat VIEW dibuat, kemudian perintah select akan dieksekusi pada TABEL TEMPORARY tersebut.
- UNDEFINED: mysql akan memilih secara acak antara MERGE atau TEMPTABLE.

View_name : adalah nama VIEW atau nama TABEL temporary
SELECT statement: perintah SQL SELECT seperti pada query select biasanya.

Contoh:
Membuat VIEW untuk menampilkan data ORDERNUMBER (nomer nota)
dan TOTAL nilai penjualan dari tabel orderdetails.

```
CREATE VIEW SalePerOrder AS
    SELECT
        orderNumber, SUM(quantityOrdered * priceEach) total
    FROM
        orderDetails
    GROUP by orderNumber
    ORDER BY total DESC;
```

Setelah perintah di atas maka tabel "SalePerOrder" (berasal dari view) akan muncul dalam daftar tabel yang ditampilkan dengan perintah "SHOW TABLES"

```
| Tables_in_classicmodels |
+-------------------------+
| customers               |
| employees               |
| employees_audit         |
| messages                |
| offices                 |
| orderdetails            |
| orders                  |
| payments                |
| price_logs              |
| productlines            |
| products                |
| saleperorder            |
| user_change_logs        |
+-------------------------+
13 rows in set (0.00 sec)
```

Contoh VIEW yang melibatkan VIEW yang lain:

View untuk menampilkan TOTAL penjualan untuk odernumber yang melebihi
$ 60000

```sql
CREATE VIEW BigSalesOrder AS
    SELECT
        orderNumber, ROUND(total,2) as total
    FROM
        saleperorder
    WHERE
        total > 60000;
```

Contoh penggunaan view:

Select * from BigSalesOrder;

BigSalesorder adlah sebuah VIEW (tabel virtual)
Akan menampilkan total order yang melebihi 60000;

Contoh VIEW yang melibatkan 3 Tabel:

Unutk menampilkan daftar nama pelanggan dan nilai total pembelian:

```sql
CREATE VIEW customerOrders AS
    SELECT
        d.orderNumber,
        customerName,
        SUM(quantityOrdered * priceEach) total
    FROM
        orderDetails d
            INNER JOIN
        orders o ON o.orderNumber = d.orderNumber
            INNER JOIN
        customers c ON c.customerNumber = c.customerNumber
    GROUP BY d.orderNumber
    ORDER BY total DESC;
```

```
MariaDB [classicmodels]> create view CustomerOrders as
    -> select d.orderNumber, c.customerName,
    -> sum(d.quantityOrdered * d.priceEach) total
    -> from orderDetails d
    -> inner join orders o ON o.orderNumber = d.orderNumber
    -> inner join customers c ON c.customerNumber = o.customerNumber
    -> group by d.orderNumber
    -> order by total desc;
Query OK, 0 rows affected (1.00 sec)

MariaDB [classicmodels]> select * from CustomerOrder limit 3;
ERROR 1146 (42S02): Table 'classicmodels.customerorder' doesn't exist
MariaDB [classicmodels]> select * from CustomerOrders limit 3;
+-------------+-------------------------+----------+
| orderNumber | customerName            | total    |
+-------------+-------------------------+----------+
|       10165 | Dragon Souveniers, Ltd. | 67392.85 |
|       10287 | Vida Sport, Ltd         | 61402.00 |
|       10310 | Toms Spezialitäten, Ltd | 61234.67 |
+-------------+-------------------------+----------+
3 rows in set (0.06 sec)
```