SQL : Data Control Language (DCL)

Ahmadi Irmansyah Lubis

VIEW

- Di dalam MySQL, View dapat didefenisikan sebagai ‘tabel virtual’. Tabel ini bisa berasal dari tabel lain, atau gabungan dari beberapa tabel.
- Tujuan dari pembuatan VIEW adalah untuk kenyamanan (mempermudah penulisan query), untuk keamanan (menyembunyikan beberapa kolom yang bersifat rahasia), atau dalam beberapa kasus bisa digunakan untuk mempercepat proses menampilkan data (terutama jika kita akan menjalankan query tersebut secara berulang).
- View adalah sebuah objek yang mirip seperti tabel yang data-datanya diperoleh dari hasil perintah query SELECT terhadap sebuah atau beberapa tabel.
- Contoh:

  hasil perintah query berikut dapat menjadi sebuah view

```
SELECT transaction_id,account_number
FROM atmwithdrawal
```

VIEW

- Sebuah view didefinisikan menggunakan perintah create view dengan bentuk

  CREATE VIEW v AS <query expression>

  - <query expression> adalah sembarang kalimat SQL.
  - Nama view direpresentasikan dengan ***v***
- Setelah view didefinisikan, nama view dapat digunakan untuk menunjuk ke tabel yang dihasilkan
- Untuk menghapus view digunakan

  DROP VIEW *v*

  - ***v*** adalah nama view yang akan dihapus
- Untuk mengubah view yang telah didefinisikan, digunakan perintah **alter view** dengan bentuk:

  ALTER VIEW *v* AS <query expression>

  - <query expression> adalah sembarang kalimat SQL.
  - Nama view direpresentasikan dengan ***v***

VIEW : CONTOH

ACCOUNT	Number	CustId	Balance	Type

DEPOSIT	Account	TransactionID	Date	Amount

CHECK	Account	Check-number	Date	Amount

ATMWITHDRAWAL	TransactionID	CustID	AcctNo	Amount	WithdrawDate

CUSTOMER	ID	Name	Phone	Address

VIEW : CONTOH

- View ***allcustomer*** berisi nomor account dengan nama dan alamat pemiliknya

```
CREATE VIEW allcustomer AS
SELECT account.number, customer.name, customer.address
FROM account,customer
WHERE account.cust_id=customer.id
```

- Sehingga untuk menampilkan data customer yang tinggal di Stamford digunakan:

```
SELECT * FROM allcustomer WHERE address LIKE ‘Stamford’
```

VIEW : CONTOH

- Menampilkan data customer yang tinggal di Stamford

```
SELECT *
FROM allcustomer
WHERE address LIKE ‘Stamford’
```

Pakai view

- Sama dengan

```
SELECT account.number, customer.name,customer.address
FROM account,customer
WHERE account.cust_id=customer.id
AND address LIKE ‘Stamford’
```

Tidak pakai view

Menambah Data Melalui View

- Menambahkan data dalam view diterjemahkan oleh DMBS menjadi perubahan dalam tabel yang sebenarnya dalam database
- Contoh:
  - View ***cust-data*** berisi data customer
    - **CREATE VIEW cust-data AS**

      SELECT id, name, address FROM customer
  - Menambahkan data ke cust-data
    - **INSERT INTO cust-data VALUES (6,'M. Harris','Oakdale')**
    - Penambahan data ke view diterjemahkan oleh DBMS menjadi kalimat SQL untuk menambahkan data ke tabel customer yang menjadi sumber dari view cust-data

Menambah Data Melalui View

- Penambahan data pada view:
  - **INSERT INTO cust-data VALUES (6,'M. Harris','Oakdale')**
- diterjemahkan menjadi:
  - **INSERT INTO customer VALUES (6,'M. Harris',NULL,'Oakdale')**
- Tidak semua view dapat ditambahkan data

  Contoh:
  - **CREATE VIEW allcustomer AS**

    SELECT account.number, customer.name, customer.address
    FROM account,customer
    WHERE account.cust_id=customer.id
  - **INSERT INTO allcustomer VALUES(107,'M. Harris','Oakdale')**
    - Tidak dapat dilakukan!!
    - Harus memilih tabel account atau customer, dan harus membuat account.cust_id/customer.id!!

Menghapus Data Melalui View

- Data dalam view juga dapat dihapus
- Penghapusan data dalam view juga diterjemahkan menjadi perubahan dalam tabel yang sebenarnya dalam database
- Contoh:
  - View cust-data berisi data customer
    - **CREATE VIEW cust-data AS**

      SELECT id,name,address FROM customer
  - Menghapus data customer id 6 dari cust-data
    - **DELETE FROM cust-data WHERE id=6**
  - Diterjemahkan menjadi:
    - **DELETE FROM customer WHERE id=6**

CATATAN !

- Kebanyakan implementasi SQL memungkinkan penambahan atau penghapusan data hanya pada view yang sederhana (tanpa aggregate) yang didefinisikan dari satu table

- View yang dapat ditambah data:
  - Atribut dalam SELECT harus memasukkan atribut yang cukup untuk setiap tuple yang ditambahkan, sehingga kita dapat mengisi atribut lain dengan nilai NULL atau DEFAULT

DATA CONTROL LANGUAGE

- Merupakan perintah-perintah yang dapat digunakan untuk menjaga keamanan database, perintah tersebut dapat dipakai untuk menentukan akses database hanya dapat dilakukan oleh orang-orang tertentu dan dengan macam akses yang dibatasi pula.

- Perintah utama DCL termasuk:
  - Membuat dan menghapus user
  - Memberikan dan menghapus hak kepada user

CREATE USER

- Untuk membuat user digunakan perintah **create user** :
  CREATE USER username IDENTIFIED BY 'password'
- Contoh:
  CREATE USER ade **IDENTIFIED BY** 'ade123';
- Selain itu dapat didefinisikan nama komputer yang digunakan

  CREATE USER user @namakomputer

  [IDENTIFIED BY [PASSWORD] 'password']
- Contoh:

  CREATE USER 'ade'@ 'pusat' IDENTIFIED BY 'ade123';

  CREATE USER 'ade'@ '%' IDENTIFIED BY 'ade123';

  CREATE USER ade@localhost IDENTIFIED BY 'ade123';

CREATE USER

- Di MySql, data user disimpan di database mysql, table user
- Contoh :

  CREATE USER ade@localhost IDENTIFIED BY ‘passwordku’;

  - Ade disimpan di field user
  - Localhost disimpan di field host
  - Passwordku disimpan di field password

DROP USER

- Untuk membuat seorang user digunakan perintah **DROP USER** :

  DROP USER user@computer

- Contoh:

  DROP USER ade;

  DROP USER ‘ade’@ ‘pusat’;

GRANT

- Perintah **GRANT** digunakan untuk memberikan hak atau autorisasi kepada seorang user

```
GRANT privileges  ON
database_name
TO user_name IDENTIFIED BY ‘password’;
```

- user_name dapat berupa:
  - sebuah user-id
  - **public**, sehingga semua users mendapatkan privilege tersebut
- Jika user_name belum ada, maka perintah ini sekaligus membuat user baru

PRIVILEGES

- ALL : hak akses penuh kecuali GRANT
- ALTER : perintah ALTER TABLE
- CREATE : perintah CREATE TABLE
- CREATE USER : perintah CREATE USER, DROP USER, RENAME USER, REVOKE ALL PRIVILEGES
- CREATE VIEW : perintah CREATE VIEW
- DELETE : perintah DELETE
- DROP : perintah DROP TABLE
- INSERT : perintah INSERT
- SELECT : perintah SELECT
- SHOW DATABASES : perintah SHOW DATABASES
- SHOW VIEW : perintah SHOW VIEW
- UPDATE : perintah UPDATE

LEVEL PRIVILEGES

- **Level Global**
  - Dapat mengakses seluruh database
  - Contoh :

```
GRANT all  ON *.*
TO ade IDENTIFIED BY ‘passwordku’;
```

- **Level Database**
  - Hanya mengakses sebuah database
  - Contoh :

```
GRANT all
ON banking.*
TO ade IDENTIFIED BY ‘passwordku’;
```

- **Level Tabel**
  - Hanya mengakses sebuah tabel
  - Contoh :

```
GRANT insert, delete
ON banking.account
TO ade IDENTIFIED BY ‘passwordku’;
```

- **Level Kolom**
  - Hanya mengakses sebuah kolom
  - Contoh :

```
GRANT insert(number)
ON banking.account
TO ade IDENTIFIED BY ‘passwordku’;
```

SHOW GRANTS

- Untuk memperlihatkan informasi hak akses yang dimiliki seorang user, digunakan perintah SHOW GRANTS

- Contoh :

  SHOW GRANTS for ade;

REVOKE

- Perintah **REVOKE** digunakan untuk menghapus hak seorang user :

```
REVOKE privileges  ON
database_name FROM
user_name;
```

- Contoh :

```
REVOKE insert, delete
ON banking.account
FROM ade;
```

PASSWORD

- Password untuk seorang user dapat diganti dengan menggunakan perintah SET PASSWORD :

  SET PASSWORD FOR ‘user_name’@‘komputer’ = PASSWORD(‘password’);

- Contoh :

  SET PASSWORD FOR ‘ade’@‘%’ = PASSWORD (‘passwordku’);

- Atau jika ingin mengubah password sendiri :

  SET PASSWORD = PASSWORD(‘password’);

Thanks!

Do you have any questions?