PRAKTIKUM SISTEM BASIS DATA

MATERI BACKUP, RECOVERY DAN TUTORIAL REPLICATION

Galih Hermawan



JURUSAN TEKNIK INFORMATIKA

FAKULTAS TEKNIK DAN ILMU KOMPUTER

UNIVERSITAS KOMPUTER INDONESIA

A. BACKUP dan RECOVERY DATABASE

Backup database sangat penting untuk menjaga keamanan data jika terjadi masalah. Sehingga kita dapat memulihkan data dan bisa membuat atau menjalankan lagi. MySQL menawarkan berbagai strategi backup yang dapat dipilih untuk menentukan metode yang terbaik sesuai dengan persyaratan kondisi yang dibutuhkan.

A.1 Secara umum, pembahasan dari metode backup database adalah sebagai berikut :

1. Membuat backup dengan menyalin file

Tabel MyISAM disimpan sebagai file, sehingga mudah untuk melakukan backup dengan menyalin file. Untuk mendapatkan cadangan konsisten, melakukan **LOCK TABLES** relevan pada tabel, diikuti oleh **FLUSH TABLES** untuk tabel. Kita hanya perlu read/baca kunci; ini memungkinkan pelanggan lain untuk melanjutkan ke tabel permintaan yang dimana kita sedang membuat salinan file dalam direktori database. Yang **FLUSH TABLES** pernyataan diperlukan untuk memastikan bahwa semua aktif indeks halaman ditulis ke disk sebelum kita memulai cadangan.

2. Membuat teks delimited-file backup

Untuk membuat sebuah file teks yang berisi tabel data, kita dapat menggunakan **SELECT * INTO OUTFILE ' *file_name* ' FROM *tbl_name*.** File yang dibuat pada server MySQL host, bukan klien host. Untuk pernyataan ini, output file dapat tidak memungkinkan karena sudah ada file yang akan ditimpa akan menimbulkan risiko keamanan. Metode ini berfungsi untuk semua jenis file data, tetapi hanya akan menyimpan tabel data, bukan struktur tabel.

Untuk kembali file keluaran, menggunakan **LOAD DATA INFILE** atau **mysqlimport.**

3. Membuat backup dengan mysqldump atau mysqlhotcopy

Teknik lain untuk membuat cadangan database adalah untuk menggunakan **mysqldump** program atau **mysqlhotcopy** skrip. **Mysqldump** lebih umum karena dapat membuat cadangan semua jenis tabel. **Mysqlhotcopy** bekerja hanya dengan beberapa penyimpanan mesin.

Buat salinan lengkap dari databasenya :

```
shell> mysqldump --tab= /path/to/some/dir  --opt db_name
```

Atau:

```
shell> mysqlhotcopy db_name /path/to/some/dir
```

Kita juga dapat membuat cadangan binari hanya dengan menyalin semua file tabel `*.frm *.MYD` dan `*.MYI` file), sepanjang server tidak memperbarui apapun. Yang **`mysqlhotcopy`** skrip menggunakan metode ini. (Tetapi dicatat bahwa metode ini tidak bekerja jika Anda berisi database `InnoDB` tabel. `InnoDB` tidak selalu menyimpan isi tabel dalam database direktori, dan **`mysqlhotcopy`** bekerja hanya untuk `MyISAM` dan `ISAM` tabel.)

Untuk `InnoDB` meja, ada kemungkinan untuk melakukan online cadangan yang tidak terkunci

4. Menggunakan binary log untuk mengaktifkan incremental backup

MySQL mendukung incremental backup: kita harus mulai dengan server *`--log-bin`* pilihan untuk mengaktifkan logging binari.Binari file-file log anda dengan memberikan informasi yang kita butuhkan untuk menjawab perubahan ke database yang dibuat menyusul ke titik di mana kita melakukan cadangan. Pada saat kita ingin membuat cadangan incremental (berisi semua perubahan yang terjadi sejak terakhir penuh atau incremental backup), kita harus memutar binari log dengan menggunakan **`FLUSH LOGS`**. Hal ini dilakukan, karena kita perlu menyalin ke lokasi cadangan semua binari log yang berkisar dari salah satu momen terakhir penuh atau untuk memperoleh cadangan terakhir tetap satu. Log biner ini adalah memperoleh cadangan kembali di waktu kita menerapkannya. Selanjutnya ketika kita melakukan full backup, kita juga harus memutar binari login menggunakan `FLUSH LOGS` **`mysqldump - siram-log,`** atau **`mysqlhotcopy - flushlog`**.

5. Snapshot backup

Beberapa filesystem implementasi mengaktifkan "snapshot" yang akan diambil. Ini memberikan salinan logis filesystem di suatu titik waktu, tanpa harus secara fisik menyalin seluruh filesystem. (Misalnya, pelaksanaan dapat menggunakan salinan-on-teknik menulis sehingga hanya bagian filesystem dimodifikasi setelah snapshot memerlukan waktu ditirukan.) MySQL sendiri tidak menyediakan kemampuan untuk mengambil snapshot filesystem. Ini tersedia melalui pihak ketiga solusi seperti Veritas atau LVM.

Jika anda menggunakan filesystem Veritas, Anda dapat membuat cadangan seperti ini:

1. Dari sebuah program klien, jalankan **FLUSH TABLES WITH READ LOCK**
2. Dari shell lain, melakukan **mount vxfs snapshot**
3. Dari pertama klien, jalankan **UNLOCK TABLES**
4. Copy file dari snapshot.
5. Unmount snapshot

6. Online dan offline backup

Online backup dilakukan sementara server MySQL sedang berjalan sehingga database informasi dapat diperoleh dari server. Offline backup dilakukan saat server dihentikan. (Perbedaan ini dapat juga digambarkan sebagai "panas" versus "dingin" backup; yang "hangat" adalah satu cadangan di mana server tetap berjalan tetapi dikunci terhadap memodifikasi data Anda akses sementara file basis data eksternal.)

Metode Online Backup ini memiliki karakteristik:

- Kurang pengorbanan untuk klien lain, yang dapat melakukan koneksi ke server MySQL selama cadangan dan dapat mengakses data tergantung operasi apa yang mereka perlu lakukan.
- Perawatan harus diambil untuk tetap sesuai penguncian sehingga data modifikasi tidak mengambil tempat yang kompromi cadangan integritas.

Mentode Offline Backup metode ini memiliki karakteristik :

- Mempengaruhi klien adversely karena server tidak tersedia selama cadangan.
- Cadangan prosedur lebih sederhana karena tidak ada kemungkinan gangguan dari aktivitas klien.

7. Titik dalam waktu pemulihan

Satu menggunakan binari untuk login adalah untuk mencapai titik-dalam-waktu pemulihan. Hal ini dapat dilakukan dengan pemulihan pertama dari file cadangan untuk dikembalikan ke server dengan negara ketika cadangan dibuat, dan kemudian kembali oleh-perubahan dalam menjalankan binari ditulis kemudian file-file log untuk langkah modifikasi data sampai ke titik yang dikehendaki dalam waktu.

Jika Anda telah mengembalikan tabel **MyISAM** yang telah menjadi rusak, bisa mencoba untuk kembali menggunakan **REPAIR TABLE** atau **myisamchk -r** pertama. Yang harus bekerja di 99,9% dari semua kasus. Jika myisamchk gagal, coba prosedur berikut. Hal ini diduga bahwa Anda telah mengaktifkan binari logging oleh MySQL dimulai dengan `--log-bin` pilihan.

1. Asli mysqldump cadangan, atau binari cadangan.
2. Jalankan perintah berikut untuk kembali menjalankan pembaruan dalam log biner:

```
shell> mysqlbinlog binlog.[0-9]* | mysql
```

Dalam beberapa kasus, Anda mungkin ingin kembali menjalankan binari tertentu hanya log, dari posisi tertentu (biasanya Anda ingin kembali menjalankan semua binari log dari tanggal dipulihkan cadangan, kecuali mungkin beberapa pernyataan salah).

8. **Cadangan penjadwalan, kompresi, dan enkripsi**

Cadangan penjadwalan adalah berharga untuk Automating cadangan prosedur. Kompresi dari cadangan output mengurangi ruang, dan output dari enkripsi memberikan keamanan yang lebih baik dari akses yang tidak diizinkan didukung-up data. MySQL sendiri tidak menyediakan kemampuan ini. **Ibbackup** dapat memampatkan InnoDB backup, kompresi dan enkripsi atau dari cadangan output dapat dicapai dengan menggunakan utilitas filesystem. Lain pihak ketiga solusi mungkin tersedia.

9. **Tabel pemeliharaan**

Integritas data dapat tercemar jika tabel menjadi rusak. MySQL menyediakan program untuk memeriksa meja dan mereka harus memperbaiki masalah ditemukan. Program-program ini berlaku terutama untuk table MyISAM.

A.2 Implementasi Backup dan Recovery

Langkah 1

Buat terlebih dahulu databasenya dengan menggunakan perintah CREATE DATABASE.

```
mysql> create database latihdb;
Query OK, 1 row affected (0.11 sec)
mysql> use latihdb;
Database changed
```

Langkah 2

Selanjutnya buatlah dua buah tabel contoh (misalnya PEGAWAI dan JABATAN) dengan
menggunakan perintah SQL di bawah ini.

```
mysql> create table jabatan (
-> kode_jabatan char(4) not null,
-> nama_jabatan varchar(25) not null,
-> constraint pk_jabatan primary key(kode_jabatan)
-> );
Query OK, 0 rows affected (0.02 sec)

mysql> create table pegawai (
-> nip char(6) not null,
-> nama varchar(25) not null,
-> kode_jabatan char(4),
-> constraint pk_pegawai primary key(nip),
-> constraint fk_pegawai_jabatan foreign key(kode_jabatan)
-> references jabatan(kode_jabatan)
-> );
Query OK, 0 rows affected (0.03 sec)
```

Langkah 3

Isikan data-data dummy ke dalam kedua buah tabel tersebut, misalnya dengan menggunakan perintah di bawah ini.

```
mysql> insert into jabatan values('J-01', 'Manager');
Query OK, 1 row affected (0.08 sec)
mysql> insert into jabatan values('J-02', 'Sales Marketing');
Query OK, 1 row affected (0.00 sec)
mysql> insert into jabatan values('J-03', 'Pimpinan Proyek');
Query OK, 1 row affected (0.00 sec)
mysql> insert into jabatan values('J-04', 'Analis Sistem');
Query OK, 1 row affected (0.00 sec)
mysql> insert into jabatan values('J-05', 'Programmer');
Query OK, 1 row affected (0.00 sec)
mysql> insert into pegawai values('000001','Imam','J-01');
Query OK, 1 row affected (0.03 sec)
mysql> insert into pegawai values('000002','Dewi','J-02');
Query OK, 1 row affected (0.00 sec)
mysql> insert into pegawai values('000003','Arif','J-03');
Query OK, 1 row affected (0.00 sec)
mysql> insert into pegawai values('000004','Camelia','J-04');
Query OK, 1 row affected (0.00 sec)
mysql> insert into pegawai values('000005','Made','J-05');
Query OK, 1 row affected (0.00 sec)
mysql> insert into pegawai values('000006','Tatang','J-05');
Query OK, 1 row affected (0.00 sec)
mysql> quit
```

Langkah 4
Proses Backup

Setelah database dan data-data siap digunakan, sekarang kita akan mencoba untuk melakukan backup terhadap data-data tersebut.

BACKUP TABLE namatabel TO ‘path_tujuan’;

path_tujuan harus dituliskan secara lengkap, namun tidak perlu disebutkan nama file-nya. Proses ini akan membuat sebuah salinan dari file .frm (definisi) and .MYD (data), tetapi tidak memasukkan file .MYI (index).

```
mysql> backup table pegawai, jabatan to 'c:\db_backup';
+-----------------+--------+----------+-----------------------------------+
| Table | Op | Msg_type | Msg_text |
+-----------------+--------+----------+-----------------------------------+
| latihdb.pegawai | backup | error | Failed copying .frm file (errno: 2) |
| latihdb.pegawai | backup | status | Operation failed |
| latihdb.jabatan | backup | error | Failed copying .frm file (errno: 2) |
| latihdb.jabatan | backup | status | Operation failed |
+-----------------+--------+----------+-----------------------------------+
4 rows in set (0.03 sec)
mysql>
```

Kenapa proses backup di atas tidak berhasil? Masalahnya adalah kita harus menggunakan *double backslash* (\\) pada saat menuliskan path tujuan. Perhatikan perintah di bawah ini yang merupakan perbaikan dari kode di atas.

```
mysql> backup table pegawai, jabatan to 'c:\\db_backup';
+-----------------+--------+----------+----------+
| Table | Op | Msg_type | Msg_text |
+-----------------+--------+----------+----------+
| latihdb.pegawai | backup | status | OK |
| latihdb.jabatan | backup | status | OK |
+-----------------+--------+----------+----------+
2 rows in set (0.03 sec)
```

A.3 Menggunakan Statemen SELECT INTO

Perintah ini sama mudahnya dengan cara yang pertama (dengan menggunakan perintah BACKUP). Hanya yang perlu untuk diperhatikan di sini adalah bahwa perintah ini hanya berlaku untuk melakukan backup per tabel. Berikut ini sintak umum dari penggunaan statemen SELECT INTO.

SELECT daftar_kolom FROM nama_tabel INTO OUTFILE nama_file;

Berikut ini contoh yang akan menunjukkan bagaimana cara melakukan backup terhadap
tabel PEGAWAI.

```
mysql> select * from pegawai into outfile 'c:\\db_backup\\peg.dmp';
Query OK, 6 rows affected (0.02 sec)
```

Perintah di atas akan membentuk sebuah file baru dengan nama PEG.DMP di dalam direktori C:\DB_BACKUP, yang merupakan file hasil dari proses backup.

B. REPLIKASI MYSQL (mysql replication)

Replikasi adalah duplikasi data dari database sumber dalam hal ini disebut master ke database lainnya atau juga disebut slave dalam suatu jaringan. Tujuan replikasi adalah untuk mengurangi beban dari server dalam pengaksesan data untuk dibagi ke komputer-komputer lain dan juga untuk meningkatkan performance database. Replikasi bersifat asynchronize yang artinya seumpama komputer master dan komputer slave disconnect lalu nyambung lagi maka data di master tidak secara otomatis diupdate ke slave semua. Jadi untuk mensinkron datanya harus melalui perintah manual. Hal itu berbeda dari teknologi MySQL Cluster dimana duplikasi datanya bersifat synchronize, sayangnya teknologi ini tidak ada di Windows.

Beberapa model replikasi antara lain :

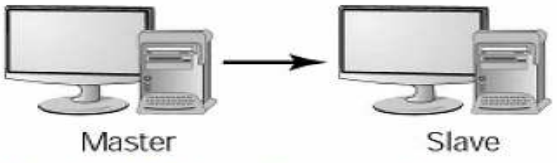


Figure 12.1: One master, one slave



Figure 12.2: One master, many slaves

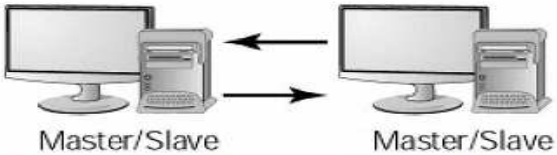


Figure 12.3: Master/slave circular relationship

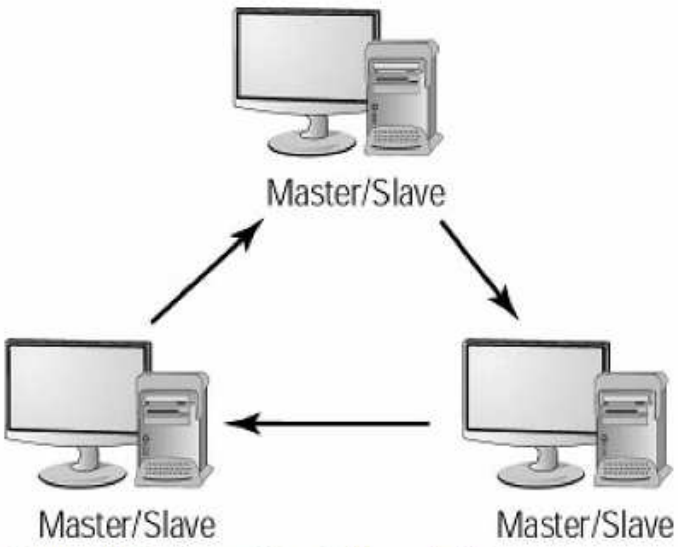


Figure 12.4: Master/slave "daisy chain"

Contoh implementasi Replikasi di MySQL

Figure 16.1. The initial reservation system configuration.

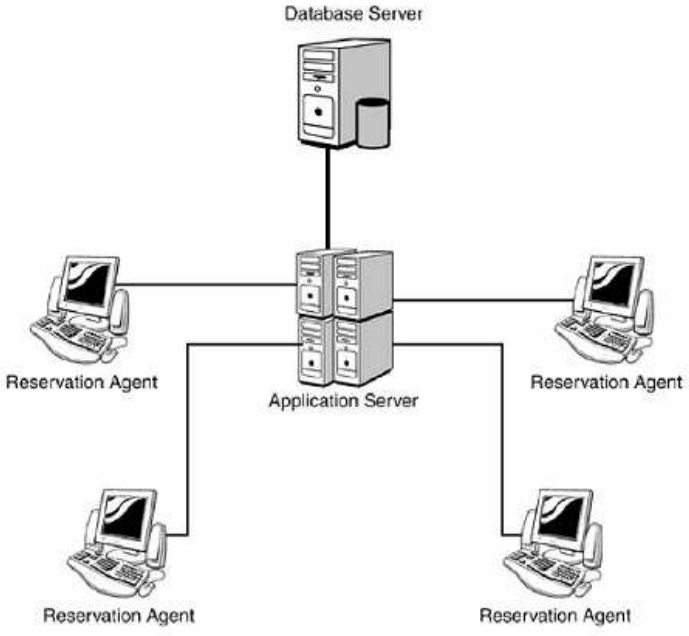
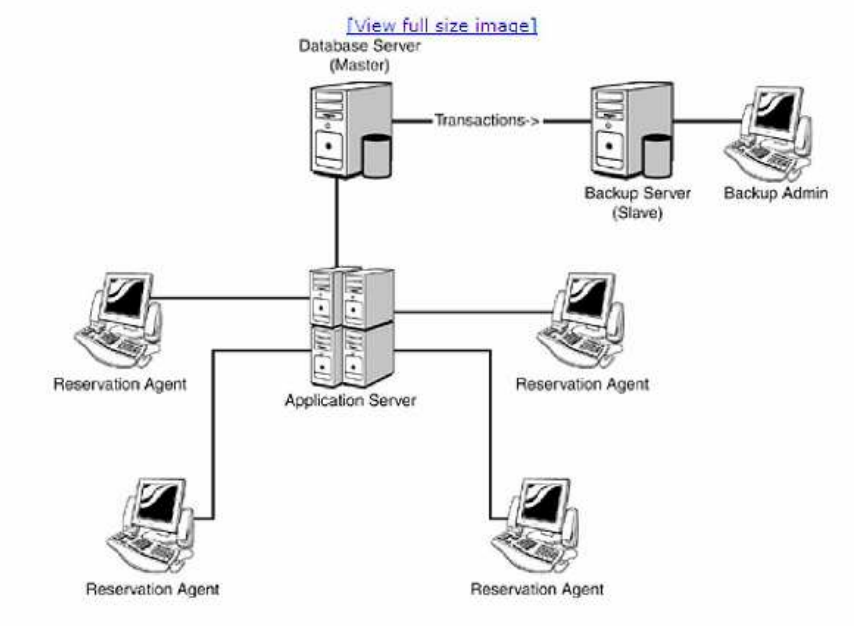
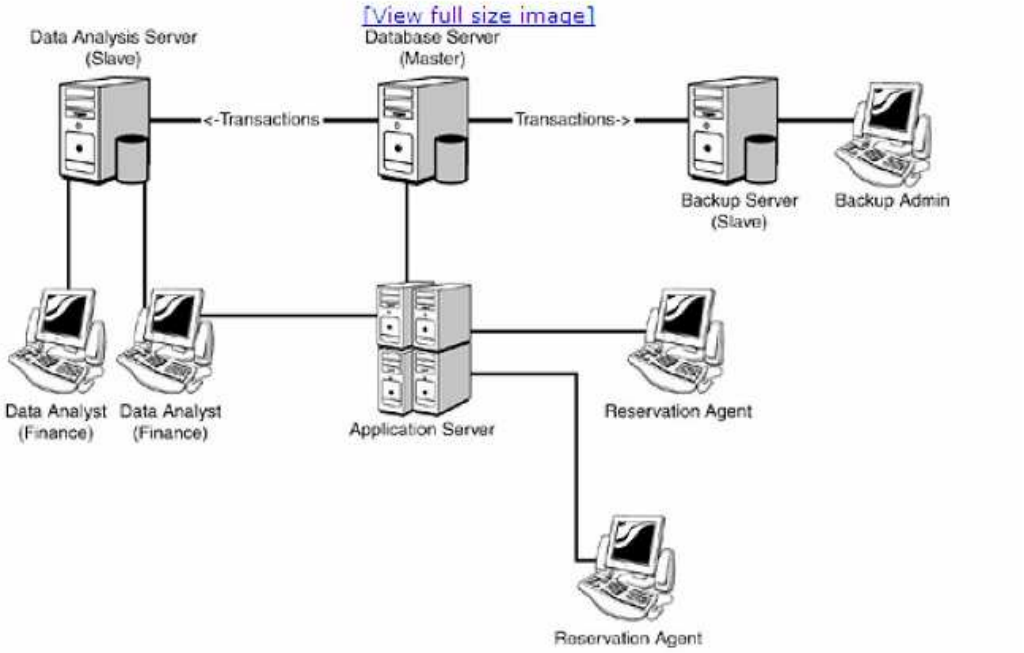


Figure 16.2. The original environment with replication enabled for backup.



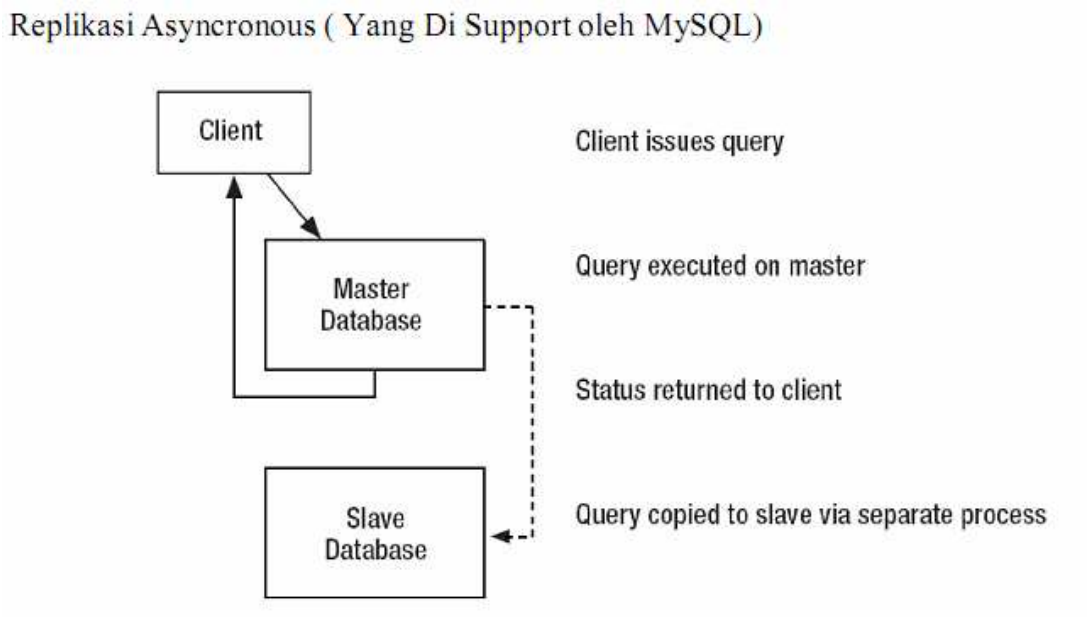
Replikasi digunakan sebagai back up database server

Figure 16.3. Replication extended to support users in another department.



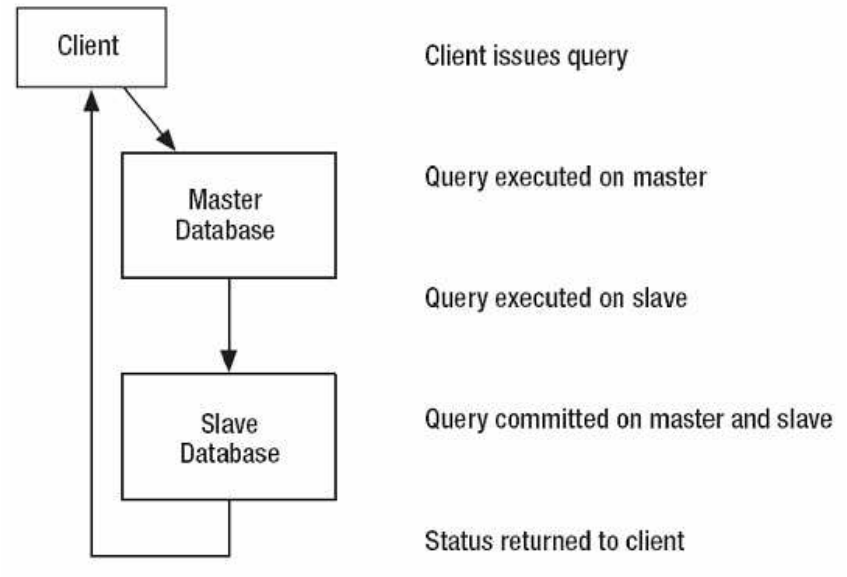
Replikasi digunakan sebagai optimasi performance database server, aplikasi modular.

2.1 Jenis Replikasi di MySQL



Dalam replikasi asyncronous yang di manage sebagai satu transaksi adalah proses yang terjadi dalam urutan Client → Master Database → Client. Proses Update di slave akan terjadi setelah transaksi di master telah selesai.

Replikasi Syncronous (Tidak di Support oleh MySQL).



2.3 Tutorial Replikasi

Dalam replikasi sysncronous statu traksaksi merupakan proses yang terdiri dari Client ->Master Database -> Slave Database -> Client.

1. Konfigurasi Master

Pertama kita edit file */etc/my.cnf* pada file ini kita definisikan file log yang berfungsi oleh slave untuk melihat apa yang telah berubah pada master dan kita juga harus mendefinisikan bahwa MySQL pada sever utama ini adalah master. Dan kita juga ingin mereplikasi database dengan nama misalnya *ehrdatabase* maka kita harus menambahkannya ke file */etc/my.cnf*

Langkah 1

log-bin = /var/log/mysql/mysql-bin.log

binlog-do-db=exampledb

server-id=1

Langkah 2

lalu kita restart Mysql

/etc/init.d/mysql restart

Langkah 3

Lalu kita log in ke MySQL sebagai root dan membuat user dengan kewenangan melakukan replikasi.

Mysql -u root -p

Enter password:

Sekarang kita dalam MySQL shell.

Langkah 4

GRANT REPLICATION SLAVE ON *.* TO 'slave_user'@'%' IDENTIFIED BY '<some_password>'; (ganti<some_password> dengan password kita!)

FLUSH PRIVILEGES;

Langkah 5

Berikutnya (masih pada shell MySQL) beri perintah:

USE exampledb;

FLUSH TABLES WITH READ LOCK;

SHOW MASTER STATUS;

Perintah terakhir akan menampilkan seperti berikut :

+——————+———-+——————+——————+

| File | Position | Binlog_do_db | Binlog_ignore_db |

+——————+———-+——————+——————+

| mysql-bin.006 | 183 | ehrdatabase | |

+——————+———-+——————+——————+

1 row in set (0.00 sec)

Langkah 6

Catat konfigurasi diatas karena akan digunakan saat mengkonfigurasi server slave. S kemudian keluar dari shell MySQL :

quit;

Langkah 7

Karena kita ingin melakukan replikasi maka pada slave juga harus memiliki database yang sama seperti pada master, maka dari itu kita dump dulu database yang ada pada master, dengan perintah :

mysqldump -u root -p<password> –opt ehrdatabase > ehrdatabase.sql

Sekarang konfigurasi master telah selesai. Sekarang set up slave ..

Langkah 8

2. Konfigurasi Slave

Pada slave pertama kali kita buat database dulu dengan nama yang sama *ehrdatabase*:

mysql -u root -p

Enter password:

CREATE DATABASE ehrdatabase;

quit;

Langkah 9

Kemudian kita import/restore kembali data yang telah kita backup dari master pada slave:

mysql -u root -p<password> ehrdatabase < ehrdatabase.sql

Langkah 10

Sekarang kita harus Memberitahu MySQL yang ada pada slave, bahwa alamat IP server master adalah 192.168.1.100 , dan database yang akan di replikasi adalah *ehrdatabase,* dan database yang akan direplikasi adalah ehrdatabase. Maka dari itu kita tambahkan beberapa baris pada */etc/mysql/my.cnf*:

server-id=2

master-host=192.168.1.100

master-user=slave_user

master-password=secret

master-connect-retry=60

replicate-do-db=ehrdatabase

Langkah 11

Lalu kita restart MySQL:

/etc/init.d/mysql restart

Langkah 12

Terakhir , kita harus melakukan ini :

mysql -u root -p

Enter password:

SLAVE STOP;

Langkah 13

Pada perintah berikutnya (masih on the MySQL shell) kita harus menyesuaikannya sesuai data yang ada:

CHANGE MASTER TO MASTER_HOST='192.168.1.100',

MASTER_USER='slave_user', MASTER_PASSWORD='<some_password>',

MASTER_LOG_FILE='mysql-bin.006', MASTER_LOG_POS=183;

MASTER_HOST **is the IP address or hostname of the master (in this example it is *192.168.0.100*).**

MASTER_USER **is the user we granted replication privileges on the master.**

MASTER_PASSWORD **is the password of *MASTER_USER* on the master.**

MASTER_LOG_FILE **is the file MySQL gave back when you ran *SHOW MASTER STATUS;* on the master.**

***MASTER_LOG_POS* is the position MySQL gave back when you ran *SHOW MASTER STATUS;* on the master.**

Langkah 14

Lalu beri perintah (masih pada shell MySQL) :

START SLAVE;

Kapanpun database terupdate di master, semua perubahan akan di replikasi ke database *ehrdatabase* pada slave.

Langkah 15

Saatnya di tes.

(pada shell MySQL) :

SHOW SLAVE STATUS\G;

********** *1. row* **********

Slave_IO_State: Waiting for master to send event

Master_Host: 192.168.1.100

Master_User: slave_user

Master_Port: 3306

Connect_Retry: 60

Master_Log_File: MASTERMYSQL01-bin.000009

Read_Master_Log_Pos: 4

Relay_Log_File: MASTERMYSQL02-relay-bin.000015

Relay_Log_Pos: 3630

Relay_Master_Log_File: MASTERMYSQL01-bin.000009

Slave_IO_Running: Yes

Slave_SQL_Running: Yes

Replicate_Do_DB: ehrdatabase

Replicate_Ignore_DB:

Replicate_Do_Table:

Replicate_Ignore_Table:

Replicate_Wild_Do_Table:

Replicate_Wild_Ignore_Table:

Last_Errno: 0

Last_Error:

Skip_Counter: 0

Exec_Master_Log_Pos: 4

Relay_Log_Space: 3630

Until_Condition: None

Until_Log_File:

Until_Log_Pos: 0

Master_SSL_Allowed: No

Master_SSL_CA_File:

Master_SSL_CA_Path:

Master_SSL_Cert:

Master_SSL_Cipher:

Master_SSL_Key:

Seconds_Behind_Master: 1519187

Untuk melihat jalan tidaknya proses replikasi ***Parameter Slave_IO_Running*** **and** ***Slave_SQL_Running*****: harus YES.**