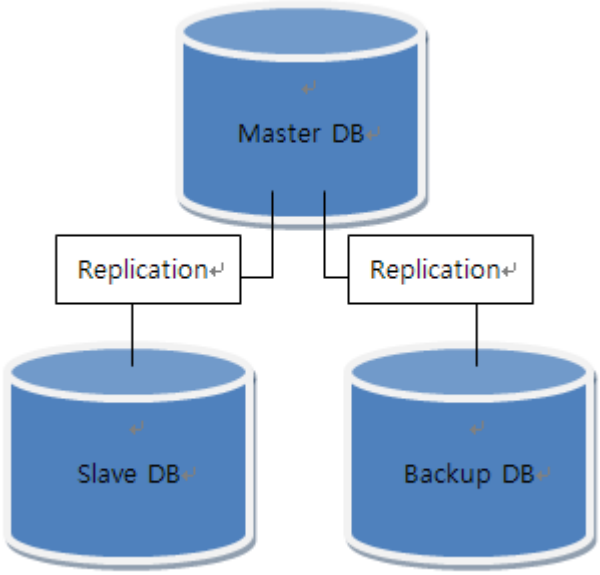
목차

MySQL Replication 복구 3

MySQL Replication 복구

요즘은 MySQL Replication 기능을 이용해서 DB System을 구축하는 경우가 많다. 또한 MySQL DB를 사용하면서 백업은 보통 Replication Slave 단에 있는 백업 Server에서 별도로 받게 된다. 백업 서버에서 Dump로 받게 된 Data는 DML 쿼리로 된 text data이며, 이 데이터를 이용해서 신규 장비를 설치하던가, 아니면 기존 장비에 문제가 생겼을 시 복구를 하게 된다.



그런데 복구를 하더라도 Replication이란 문제로 인해 Master DB와 싱크를 맞추는 문제가 발생하게 된다. 따라서 백업 서버에서 백업을 받게 되는 경우, Master DB의 binary log의 포지션을 알아야지만 된다. 이럴 경우에는 다음과 같은 절차를 따라서 Dump를 받으면 된다.

```
/usr/local/mysql/bin/mysqldump -uroot -p --master-data=2  >
/data/…/backup.sql
```

이 명령어 중에서 `–master-data=2` 옵션을 주게 되면 덤프 시점의 binary log의 포지션 정보가 같이 Dump 되게 된다. Dump된 파일을 에디터로 열어보면 다음과 같은 내용이 나온다.

```
CHANGE MASTER TO MASTER_LOG_FILE='binary파일', MASTER_LOG_POS=포지션번호;
```

위 말은 binary파일의 포지션 번호 시점에서 백입이 이루어 졌다는 의미이다. 물론 binary 파일과 포지션은 복구하기 위한 중요한 내용이 되겠지만, 위와 같은 구성에서 복구를 하기 위해서는 조금 애매 보호한 감이 있다. 가령, Backup DB에서 Dump받은 파일을 Slave DB를 복구하기 위해 사용한다던가, 아니면 신규 Slave 장비가 와서 새로운 시스템을 구축할 시에는 Backup DB의 binary 파일명과 포지션은 별로 소용이 없기 때문이다. 따라서 이럴 때에는 다음과 같은 절차로 Master DB의 binary 파일과 포지션을 알아낼 수 있다.

1. Backup DB에서 포지션이 속한 바이너리 로그를 sql 파일로 export 한다. 사용 툴은 mysqlbinlog이다.

```
/usr/local/mysql/bin/mysqlbinlog --start-datetime="2008-12-15 02:00:00" --
stop-datetime='2008-12-15 02:30:00' /data/mysql_data/binary파일 > binlog.sql
```

2. Master DB에서도 위 방법대로 바이너리 로그를 sql 파일로 export 한다.

```
/usr/local/mysql/bin/mysqlbinlog --start-datetime="2008-12-15 02:00:00" --
stop-datetime='2008-12-15 02:30:00' /data/mysql_data/binary파일 > binlog.sql
```

3. Backup DB에서 export한 sql 파일을 에디터로 열어서 확인해 본다.

```
# at 576856551
#060824  2:18:30 server id 1  log_pos 576856551     Intvar
SET INSERT_ID=136;
# at 576856579
#060824  2:18:30 server id 1  log_pos 576856579     Query
thread_id=16587969  exec_time=16157 error_code=0
SET TIMESTAMP=1156353510;

...

# at 576856919
```

최초에 Backup DB에서 Dump받은 파일에 기록된 포지션 정보를 찾는다.

4. Master DB에서 export 한 sql 파일을 에디터로 열어서 동일한 포지션을 찾는다.

```
# at 576375776
#060824  2:18:30 server id 1  log_pos 576375776         Intvar
SET INSERT_ID=136;
# at 576375804
#060824  2:18:30 server id 1  log_pos 576375804         Query
thread_id=16587969      exec_time=0     error_code=0
SET TIMESTAMP=1156353510;
```

Backup DB에서 해당 포지션에 기록된 TIMESTAMP값과 동일한 정보를 Master DB에서 export한 sql 파일에서 찾는다. 위의 경우에는 576375776이 포지션 번호가 된다.

5. 최초에 Dump 받은 Dump Data의 바이너리 파일과 포지션을 수정해 준다.

```
CHANGE MASTER TO MASTER_LOG_FILE='마스터DB의 binary파일',
MASTER_LOG_POS=576375776;
```

위와 같이 되었다면, Dump 받은 파일은 언제라도 Master DB에 붙여서 replication을 동기화 시킬 수 있는 준비가 된 상태이다. 만일 새로운 장비를 들여와서 Dump Data로 복구를 했다면, 위 명령어를 내린 후

slave를 start 시키면, Master DB의 해당 포지션 부터 sync를 맞춰가기 시작한다. Master DB의 포지션을 모두 다 따라갔다고 판단되면 Slave DB를 서비스에 투입시키면 끝이다.

From:
https://atl.kr/dokuwiki/ - **AllThatLinux!**

Permanent link:
https://atl.kr/dokuwiki/doku.php/mysql_replication_%EB%B3%B5%EA%B5%AC?rev=1475040398

Last update: **2016/09/28 05:26**