Binary Log Backups

Binary Log Backups

How Do MySQL Binary Log Backups Work?

MySQL Binary Logs contain a record of each SQL query executed against the database that changes data since the logs where last flushed (new log file started). MySQL Binary Logs must be enabled by passing the --log-binoption to mysqld. The log contains queries like UPDATE and DELETE. The log does not record SELECT or SHOW queries for example because they do not change the database.MySQL Binary Logs are some times confused with the InnoDB binary transaction log. InnoDB uses a binary log to journal changes to the InnoDB table space file(s) as a protection from crashes to protect table space data integrity. When --log-bin is enabled even InnoDB transactions are written to the binary transaction log used for replication or restores.

How Do MySQL Binary Log Backups Work?

If you have binary logging enabled then your MySQL binary log files might look like:

```
localhost-bin.000001
localhost-bin.000002
localhost-bin.000003
localhost-bin.index
```

For each database and table the backup program executes:

1. LOCK TABLES
2. FLUSH LOGS
3. SHOW CREATE TABLE table name
4. SELECT * INTO OUTFILE temporary file
5. Write the contents of the temporary file to the end of the dump file
6. UNLOCK TABLES

Note: see http://dev.mysql.com/doc/refman/5.0/en/backup-policy.html

How the FLUSH LOGS Query Works

When the FLUSH LOGS query is executed MySQL starts a new binary log file to record queries in. So now the binary log files look like:

```
localhost-bin.000001
localhost-bin.000002
localhost-bin.000003
localhost-bin.000004
localhost-bin.index
```

Incremental queries made AFTER the backup operation appear in the localhost-bin.000004 binary log file.

Restoring with Binary Logs

To restore from a point in time using binary logs you would:

1. Restore the database from the last SQL Dump completed before the desired recovery point.
2. Use mysqlbinlog to restore to the desired point in timewhere N is the log entry number you want to restore up to.

```
mysqlbinlog --stop-position=Nlocalhost-bin.000004 | mysql
```

Advantages

1. Allows point-in-time restore right down to individual queries.

Disadvantages

1. Requires a Full backup be done periodically There is no way to do an incremental backup with a SQL Dump. This means a backup can be very time consuming especially on larger databases.

2. Holds a global read lock on all tables blocking writes from other connections for the duration of the full backup. Locking can be optional. If locking is not performed there is no consistency in the backup.

3. Restoring only desired databases or tables requiring editing the SQL Dump file before restoring form it.

4. Restoring to point in time with the mysqlbinlog utility can be complicated.