**Database Backup Methods**

MySQL Backups

Disk Level [ Mount Points /backups , D Drive ] , Tape Level , Cloud level ( Iron mountain, BlueCube , EMC2, IBM, HCL Storage )

Logical Backups

mysqldump - To export the data logical with using mysqldump utliity

mysql - To import the data

all database backup, single database , object level means tables,SP,views etc..

Physical Backups

complete datafiles will take the backup.

Shutdown Server - Cold Backup / Offline Backup

Without Shutdown - Hot Backup / Online Backup

MYSQL Server - Oracle Product

Physical Backups - Enterprise Edition (MEB), This is supported for Only MySQL Server

- Percona Xtra Backup is utility Supported for MySQL, Mariadb & Percona Server

What backups it will supported ?

- Full Backups (Physical files (.ibd) with My.Cnf )

- Incremental Backups ( An incremental backup is a type of backup that only copies data that has been changed or created since the previous backup activity was conducted )

- Compressed Backups

- Partial Backups ( Database level & Table level )

- Encrypted Backups ( Note:- You have Encrypt the Mysql server )

■ Logical backup—A logical backup is created by saving information that represents the logical database structures using SQL statements like CREATE DATABASE, CREATE TABLE, and INSERT.

It is not correct to say that a logical backup is a text representation of the database server because there can be non-text binary data in logical backup. Other names for a logical backup are a logical export and an export.

**Part**

The advantages of a logical backup are that it allows the database administrator to manipulate the backup data using tools such as dd and grep and programming languages such as awk and Perl.

In addition, logical backups are more compatible between different versions of mysqld when you are upgrading your database server. Sun recommends a logical backup and restore of your database as a standard practice when upgrading between different release series (that is, when upgrading from version 5.1 to 6.0) of mysqld.

The disadvantages compared to a physical backup are that a logical backup is slower to backup and restore and may take up more space (but compresses well).

■ Physical backup—A physical backup is a backup of the actual database files or disk partitions. This can be much faster to back up and restore than logical backups. Though physical backups do not compress much (the data is usually in a binary format and thus somewhat compressed already), physical backups are often smaller than uncompressed logical backups. Physical backups may also be called raw backups.

■ Full backup—A full backup is a standalone backup containing everything in the

database. If necessary you could use the results of the full backup to re-create a server. A full backup can be either a logical or physical backup.

■ Incremental backup—An incremental backup is a backup that only contains the data changed since the previous backup. The previous backup may have been full or incremental.

The advantage of an incremental backup compared to a full backup is quicker

backup times. Incremental backups are used to be able to back up data more frequently than a full backup may allow.

For example, a full backup of a large data set may take 3 hours, and a daily incremental backup may take 30 minutes. An organization may only be able to run a 3-hour full backup once a week on Sundays, but during the other six days of the week an organization can run an incremental backup.

In this way, the data set can be restored to how it was on Sunday, Tuesday, or any other day of the week. To restore to Tuesday, first the Sunday full backup is restored. Then the Monday incremental backup is applied to the Sunday restore, restoring the data to how it was on Monday. Then the Tuesday incremental backup can be applied.

The biggest disadvantage of an incremental backup is that it is not a full data set, and cannot be used by itself to restore a full data set. This means that incremental backups have longer recovery times than full backups, because the latest full backup and all intermediate incremental backups need to be restored.

■ Consistent backup—A consistent backup is a backup at an exact moment in time.

A backup process takes time to complete. Consider the following scenario:

■ The customer table has a field that references a record in the address table.

■ The address table is backed up at 1:00 am.

■ The customer table is backed up at 1:05 am.

■ Between 1:00 am and 1:05 am, a customer named Tracy Inge signs up for a

new account using an address not previously in the system.

**Backups and Recovery 13**

The address table was backed up at 1:00 am, before Tracy signed up for a new account.

Therefore, the backup of address will not contain Tracy’s address. However, customer is backed up at 1:05 am, and by then Tracy’s address is in address, and there is a reference to that record in customer. Thus, a problem occurs: in the backup of customer,

Tracy’s record references an address that does not exist in the backup of address!

An inconsistent backup is usually simpler and less resource-intensive to produce than a consistent backup. However, an inconsistent backup cannot be used as a standalone backup. An inconsistent backup may be used for partial data restores (such as restoring only Tracy’s e-mail address).

■ Hot backup—A hot backup is a backup of a database that is still running. During a hot backup, neither reads nor writes are blocked.

■ Warm backup—A warm backup is a backup of a database that is still running. During a warm backup, read queries are not blocked but writes are prohibited from making any modifications to the database for the duration of the backup.

■ Cold backup—A cold backup is a backup performed while the database is shut down.

This makes it very easy to make a consistent copy of your data. The disadvantage is that the server is not accessible during the time the backup is performed.

■ Point-in-time restore—A point-in-time restore is a restoration of a database to a specified date and time. If this does not correspond to the time a full backup was performed, incremental backups and/or server logs must be used to finish the restoration process.

**mysqldump** is an effective tool to backup MySQL database. It creates a \*.sql file with **DROP table**, **CREATE table** and **INSERT** intosql-statements of the source database. To restore the database, execute the \*.sql file on destination database.

For MyISAM, use mysqlhotcopy method that we explained earlier, as it is faster for MyISAM tables.  
  
Using mysqldump, you can backup a local database and restore it on a remote database at the same time, using a single command.

let us review several practical examples on how to use mysqldump to backup and restore.  
  
For the impatient, here is the quick snippet of how backup and restore MySQL database using mysqldump:

Syntax:-

backup: # mysqldump -u root -p[root\_password] [database\_name] > dumpfilename.sql

restore:# mysql -u root -p[root\_password] [database\_name] < dumpfilename.sql

**How To Backup MySQL database**

1. Backup a single database:

This example takes a backup of dbacentre database and dumps the output to dbacentre.sql

# mysqldump -u root -ppassword database\_name > database\_bkp.sql

2. Backup multiple databases:

If you want to backup multiple databases, first identify the databases that you want to backup using the show databases as shown below:

#mysql -u root -ppassword

mysql> show databases;

+--------------------+

| Database |

+--------------------+

| information\_schema |

| prod01 |

| dbacentre |

| dev01 |

| mysql |

| performance\_schema |

| sys |

+--------------------+

7 rows in set (0.00 sec)

For example, if you want to take backup of both dbacentre and prod01 database, execute the mysqldump as shown below:

**# mysqldump -u root -ppassword --databases dbacentre prod01 > prod\_dbacentre.sql**

Verify the prod\_dbacentre.sql dumpfile contains both the database backup.

# grep -i "Current database:" /tmp/prod\_dbacentre.sql

-- Current Database: `mysql`

-- Current Database: `dbacentre`

### 3. Backup all the databases:

The following example takes a backup of  all the database of the MySQL instance.

# **mysqldump -u root –ppassword --all-databases > /backups/all-dbs.dmp**

### 4. Backup a specific table:

In this example, we backup only the emp\_contacts table from dbacentre database.

# **mysqldump -u root -ppassword dbacentre.emp\_contacts \**

**> /tmp/dbacentre\_emp\_contacts\_bkp.sql**

### 4. Different mysqldump group options:

* **–opt is a group option**, which is same as –add-drop-table, –add-locks, –create-options, –quick, –extended-insert, –lock-tables, –set-charset, and –disable-keys. opt is enabled by default, disable with –skip-opt.
* **–compact is a group option**, which gives less verbose output (useful for debugging). Disables structure comments and header/footer constructs. Enables options –skip-add-drop-table –no-set-names –skip-disable-keys –skip-add-locks

## How To Restore MySQL database

### 1. Restore a database

In this example, to restore the dbacentre database, execute mysql with < as shown below. When you are restoring the dumpfilename.sql on a remote database, make sure to create the dbacentre database before you can perform the restore.

# **mysql -u root -ppassword**

mysql> **create database dbacentre;**

Query OK, 1 row affected (0.02 sec)

# **mysql -u root -ppassword dbacentre < /backups/dbacentre.sql**

# **mysql -u root -p[root\_password] [database\_name] < dumpfilename.sql**

### 2. Backup a local database and restore to remote server using single command:

This is a sleek option, if you want to keep a read-only database on the remote-server, which is a copy of the master database on local-server. The example below will backup the dbacentre database on the local-server and restore it as dbacentre1 database on the remote-server. Please note that you should first create the dbacentre1 database on the remote-server before executing the following command.

[local-server]# **mysqldump -u root -ppassword dbacentre | mysql \**

**-u root -ppassword --host=remote-server -C dbacentre1**

[Note: There are two -- (hyphen) in front of host]

Practice Logical Backups

-- All Databases .dmp or .sql formats

> exporting the data

mysqldump -u root -p --all-databases > /backups/alldb\_backups.dmp

mysqldump --host=host\_name --all-databases --events --routines --master-data=2> dump\_file

-- Single Database

mysqldump -u root -p demo > /backups/demodb\_feb06.sql

mysqldump -u root -p --events --routines --master-data=2 test > /backup/testdb.dmp

-- Multipe Databases

mysqldump -u root -p --databases demo classicmodels world > /backups/3dbbackups.dmp

-- Table in a Database

mysqldump -u root -p dbname table1 > /backups/db1tbl.sql

-- Multiple tables in a Database

mysqldump -u root -p dbname table1 table2 > /root/backups/3tablesdb.sql

--Database Zipping (Compressed Method)

mysqldump -u root -p classicmodels | gzip > classicmodels.sql.gz

--Multiple tables with no data

mysqldump -u root -p demo emp hr --tables --no-data > /backups/nodata\_bkp.sql

Restore Backups < importing the data

mysql -u root -p dbname < testdb.dmp

mysql -uroot -p dbname < table1.sql [ table name ]

---cold backup

mkdir /backups/cold

/var/lib/mysql#cp -rf \* /backups/cold

/etc/mysql#cp my.cnf /backups/cold

only a table cannot able to restore from full backup or any database backup, using sed command we can as mentioned below

sed -n -n '/CREATE TABLE.\*EMP/,/UNLOCK TABLES/p' /backups/proddb.sql > /backups/emp\_tbl.sql

Taking Incremental backups from binlog

mysqldump -u root -pmysql123 --all-databases --single-transaction --flush-logs --master-data=2 > full.sql

| mysql-bin.000021 | 177 |

| mysql-bin.000022 | 1206 |

| mysql-bin.000023 | 425 |

+------------------+-----------+

23 rows in set (0.00 sec)

mysql> flush logs;

Query OK, 0 rows affected (0.03 sec)

mysql> insert into student values (5,'lokesh','90');

Query OK, 1 row affected (0.04 sec)

mysql> show binary logs;

+------------------+-----------+

| Log\_name | File\_size |

+------------------+-----------+

| mysql-bin.000001 | 201 |

| mysql-bin.000002 | 0 |

| mysql-bin.000003 | 177 |

| mysql-bin.000004 | 177 |

| mysql-bin.000005 | 177 |

| mysql-bin.000006 | 177 |

| mysql-bin.000007 | 177 |

..

| mysql-bin.000019 | 5038 |

| mysql-bin.000020 | 3900 |

| mysql-bin.000021 | 177 |

| mysql-bin.000022 | 1206 |

| mysql-bin.000023 | 472 |

| mysql-bin.000024 | 427 |

+------------------+-----------+

24 rows in set (0.00 sec)

mysql> create table teacher(id int,name char(20));

Query OK, 0 rows affected (0.19 sec)

mysql> drop database prod;

Query OK, 18 rows affected (0.65 sec)

mysql> create database prod;

Query OK, 1 row affected (0.05 sec)

find / -name full.sql

mysql -u root -pmysql123 prod < /home/dba/full.sql

/var/log/mysql# mysqlbinlog mysql-bin.000023 mysql-bin.000024 | mysql -u root -p prod

shell> mysqlbinlog mysql-bin.000023 | mysql -u root -p prod

shell> mysqlbinlog mysql-bin.000024 | mysql -u root -p prod