MYSQL DATABASE IMPORT - RECOVERY METHODS

 Copyright © tutorialspoint.com

There are two simple ways in MySQL to load data into MySQL database from a previously backed up file.

Importing Data with LOAD DATA:

MySQL provides a LOAD DATA statement that acts as a bulk data loader. Here's an example statement that reads a file dump.txt from your current directory and loads it into the table mytbl in the current database:

```
mysql> LOAD DATA LOCAL INFILE 'dump.txt' INTO TABLE mytbl;
```

- If the LOCAL keyword is not present, MySQL looks for the datafile on the server host using looking into absolute pathname fully specifies the location of the file, beginning from the root of the filesystem. MySQL reads the file from the given location.
- By default, LOAD DATA assumes that datafiles contain lines that are terminated by linefeeds (newlines) and that data values within a line are separated by tabs.
- To specify a file format explicitly, use a FIELDS clause to describe the characteristics of fields within a line, and a LINES clause to specify the line-ending sequence. The following LOAD DATA statement specifies that the datafile contains values separated by colons and lines terminated by carriage returns and new line character:

```
mysql> LOAD DATA LOCAL INFILE 'dump.txt' INTO TABLE mytbl
  -> FIELDS TERMINATED BY ':'
  -> LINES TERMINATED BY '\r\n';
```

- LOAD DATA assumes the columns in the datafile have the same order as the columns in the table. If that's not true, you can specify a list to indicate which table columns the datafile columns should be loaded into. Suppose your table has columns a, b, and c, but successive columns in the datafile correspond to columns b, c, and a. You can load the file like this:

```
mysql> LOAD DATA LOCAL INFILE 'dump.txt'
    -> INTO TABLE mytbl (b, c, a);
```

Importing Data with mysqlimport

MySQL also includes a utility program named ***mysqlimport*** that acts as a wrapper around LOAD DATA so that you can load input files directly from the command line.

To load a data from dump.txt into mytbl, use following command at UNIX prompt.

```
$ mysqlimport -u root -p --local database_name dump.txt
password *****
```

If you use mysqlimport, command-line options provide the format specifiers. mysqlimport commands that correspond to the preceding two LOAD DATA statements look like this:

```
$ mysqlimport -u root -p --local --fields-terminated-by=":" \
   --lines-terminated-by="\r\n"  database_name dump.txt
password *****
```

The order in which you specify the options doesn't matter for mysqlimport, except that they should all precede the database name.

The **mysqlimport** statement uses the --columns option to specify the column order:

```
$ mysqlimport -u root -p --local --columns=b,c,a \
    database_name dump.txt
password *****
```

Handling Quotes and Special Characters:

The FIELDS clause can specify other format options besides TERMINATED BY. By default, LOAD DATA assumes that values are unquoted and interprets the backslash (\) as an escape character for special characters. To indicate the value quoting character explicitly, use ENCLOSED BY; MySQL will strip that character from the ends of data values during input processing. To change the default escape character, use ESCAPED BY.

When you specify ENCLOSED BY to indicate that quote characters should be stripped from data values, it's possible to include the quote character literally within data values by doubling it or by preceding it with the escape character. For example, if the quote and escape characters are " and \, the input value "a""b\"c" will be interpreted as a"b"c.

For mysqlimport, the corresponding command-line options for specifying quote and escape values are --fields-enclosed-by and --fields-escaped-by.