MySQL command-line client Commands

Connect to MySQL server

using mysql command-line client with a username and password (MySQL will prompt for a password):

```
mysql -u [username] -p;
```

Connect to MySQL Server with a specified database using a username and password:

```
mysql -u [username] -p [database];
```

Exit mysql command-line client:

```
exit;
```

Export data using mysqldump tool

```
mysqldump -u [username] -p [database] > data_backup.sql;
```

To clear MySQL screen console window on Linux, you use the following command:

```
mysql> system clear;
```

Currently, there is no command available on Windows OS for clearing MySQL screen console window.

Working with databases

Create a database with a specified name if it does not exist in the database server

```
CREATE DATABASE [IF NOT EXISTS] database_name;
```

Use a database or change the current database to another database that you are working with:

```
USE database_name;
```

Drop a database with a specified name permanently. All physical files associated with the database will be deleted.

```
DROP DATABASE [IF EXISTS] database_name;
```

Show all available databases in the current MySQL database server

```
SHOW DATABASE;
```

Working with tables

Show all tables in a current database.

```
SHOW TABLES;
```

Create a new table

```
CREATE TABLE [IF NOT EXISTS] table_name(
  column_list
);
```

Add a new column into a table:

```
ALTER TABLE table
ADD [COLUMN] column_name;
```

Drop a column from a table:

```
ALTER TABLE table_name
DROP [COLUMN] column_name;
```

Add index with a specific name to a table on a column:

```
ALTER TABLE table
ADD INDEX [name](column, ...);
```

Add primary key into a table:

```
ALTER TABLE table_name
ADD PRIMARY KEY (column_name,...);
```

Remove the primary key of a table:

```
ALTER TABLE table_name
DROP PRIMARY KEY;
```

Drop a table

```
DROP TABLE [IF EXISTS] table_name;
```

Show the columns of a table:

```
DESCRIBE table_name;
```

Show the information of a column in a table:

```
DESCRIBE table_name column_name;
```

Working with indexes

Creating an index

with the specified name on a table:

```
CREATE INDEX index_name
ON table_name (column,...);
```

Drop an index

```
DROP INDEX index_name;
```

Create a unique index

```
CREATE UNIQUE INDEX index_name
ON table_name (column,...);
```

Working with views

Create a new view:

```
CREATE VIEW [IF NOT EXISTS] view_name
AS
  select_statement;
```

Create a new view with the `WITH CHECK OPTION` :

```
CREATE VIEW [IF NOT EXISTS] view_name
AS select_statement
WITH CHECK OPTION;
```

Create or replace a view:

```
CREATE OR REPLACE view_name
AS
```

```
select_statement;
```

Drop a view:

```
DROP VIEW [IF EXISTS] view_name;
```

Drop multiple views:

```
DROP VIEW [IF EXISTS] view1, view2, ...;
```

Rename a view:

```
RENAME TABLE view_name
TO new_view_name;
```

Show views from a database:

```
SHOW FULL TABLES
[{FROM | IN } database_name]
WHERE table_type = 'VIEW';
```

Working with triggers

Create a new trigger:

```
CREATE TRIGGER trigger_name
{BEFORE | AFTER} {INSERT | UPDATE| DELETE }
ON table_name FOR EACH ROW
trigger_body;
```

Drop a trigger:

```
DROP TRIGGER [IF EXISTS] trigger_name;
```

Show triggers in a database:

```
SHOW TRIGGERS
[{FROM | IN} database_name]
[LIKE 'pattern' | WHERE search_condition];
```

Working with stored procedures

Create a stored procedure:

```
DELIMITER $$

CREATE PROCEDURE procedure_name(parameter_list)
BEGIN
   body;
END $$

DELIMITER ;
```

Drop a stored procedure:

```
DROP PROCEDURE [IF EXISTS] procedure_name;
```

Show stored procedures:

```
SHOW PROCEDURE STATUS
[LIKE 'pattern' | WHERE search_condition];
```

Working with stored functions

Create a new stored function:

```
DELIMITER $$

CREATE FUNCTION function_name(parameter_list)
RETURNS datatype
[NOT] DETERMINISTIC
BEGIN
```

```
 -- statements
END $$

DELIMITER ;
```

Drop a stored function:

```
DROP FUNCTION [IF EXISTS] function_name;
```

Show stored functions:

```
SHOW FUNCTION STATUS
[LIKE 'pattern' | WHERE search_condition];
```

Querying data from tables

Query all data from a table:

```
SELECT * FROM table_name;
```

Query data from one or more column of a table:

```
SELECT
    column1, column2, ...
FROM
    table_name;
```

Remove duplicate rows from the result of a query:

```
SELECT
    DISTINCT (column)
FROM
   table_name;
```

Query data with a filter using a `WHERE` clause:

```
SELECT select_list
FROM table_name
WHERE condition;
```

Change the output of the column name using column alias

```
SELECT
    column1 AS alias_name,
    expression AS alias,
    ...
FROM
    table_name;
```

Query data from multiple tables using inner join

```
SELECT select_list
FROM table1
INNER JOIN table2 ON condition;
```

Query data from multiple tables using left join

```
SELECT select_list
FROM table1
LEFT JOIN table2 ON condition;
```

Query data from multiple tables using right join

```
SELECT select_list
FROM table1
RIGHT JOIN table2 ON condition;
```

Make a Cartesian product of rows:

```
SELECT select_list
FROM table1
CROSS JOIN table2;
```

Counting rows in a table.

```
SELECT COUNT(*)
FROM table_name;
```

Sorting a result set:

```
SELECT
    select_list
FROM
    table_name
ORDER BY
    column1 ASC [DESC],
    column2 ASC [DESC];
```

Group rows using the `GROUP BY` clause.

```
SELECT select_list
FROM table_name
GROUP BY column_1, column_2, ...;
```

Filter group using the `HAVING` clause:

```
SELECT select_list
FROM table_name
GROUP BY column1
HAVING condition;
```

Modifying data in tables

Insert a new row into a table:

```
INSERT INTO table_name(column_list)
VALUES(value_list);
```

Insert multiple rows into a table:

```
INSERT INTO table_name(column_list)
VALUES(value_list1),
      (value_list2),
      (value_list3),
      ...;
```

Update all rows in a table:

```
UPDATE table_name
SET column1 = value1,
    ...;
```

Update data for a set of rows specified by a condition in `WHERE` clause.

```
UPDATE table_name
SET column_1 = value_1,
    ...
WHERE condition
```

Update with join

```
UPDATE
    table1,
    table2
INNER JOIN table1 ON table1.column1 = table2.column2
SET column1 = value1,
WHERE condition;
```

Delete all rows in a table

```
DELETE FROM table_name;
```

Delete rows specified by a condition:

```
DELETE FROM table_name
WHERE condition;
```

Delete with join

```
DELETE table1, table2
FROM table1
INNER JOIN table2
    ON table1.column1 = table2.column2
WHERE condition;
```

Searching

Search for data using the `LIKE` operator:

```
SELECT select_list
FROM table_name
WHERE column LIKE '%pattern%';
```

Text search using a regular expression with `RLIKE` operator.

```
SELECT select_list
FROM table_name
WHERE column RLIKE 'regular_expression';
```