

# Database Design and Query

# Section 3 : Creating Database

- Creating Database
- Creating Table
- Modifying Database, Table and Column
- MySQL Constraint

# Creating Database

- Before doing anything else with the data, you need to create a database. A database is a container of data. It stores contacts, vendors, customers or any kind of data that you can think of.
- To create a database in MySQL, you use the “**CREATE DATABASE**” statement.

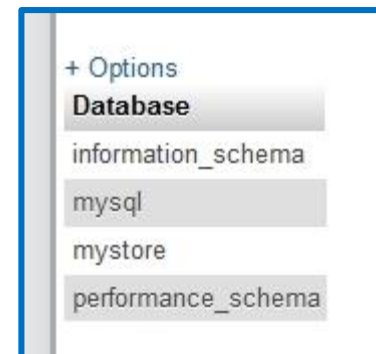
```
1 CREATE DATABASE IF NOT EXISTS database_name;
```

- Let's examine the “**CREATE DATABASE**” statement in greater detail:
  - Followed by the “**CREATE DATABASE**” statement is database name that you want to create. It is recommended that the database name should be as meaningful and descriptive as possible.
  - The “**IF NOT EXISTS**” is an optional clause of the statement. The “**IF NOT EXISTS**” clause prevents you from an error of creating a new database that already exists in the database server. You cannot have 2 databases with the same name in a MySQL database server.

# Displaying Databases

- The “**SHOW DATABASES**” statement displays all databases in the MySQL database server. You can use the “**SHOW DATABASES**” statement to check the database that you’ve created or to see all the databases on the database server before you create a new database, for example:

```
1 SHOW DATABASES;
```



The screenshot shows the output of the `SHOW DATABASES;` command in a MySQL terminal. At the top, there is a header section with a blue plus icon and the text "+ Options" in blue, followed by "Database" in bold. Below this, four database names are listed in separate grey rectangular boxes: "information\_schema", "mysql", "mystore", and "performance\_schema".

Database
information_schema
mysql
mystore
performance_schema

# Removing Database

- Removing database means you delete the database physically. All the data and associated objects inside the database are permanently deleted and this cannot be undone. Therefore, it is very important to execute this query with extra cautions.
- To delete a database, you use the “**DROP DATABASE**” statement as follows:

```
1 DROP DATABASE database_name;
```

# Creating Table

- In order to create a new table within a database, you use the MySQL “CREATE TABLE” statement. The “CREATE TABLE” statement is one of the most complex statement in MySQL.
- The following illustrates the syntax of the CREATE TABLE statement in the simple form:

```
1 CREATE TABLE [IF NOT EXISTS] table_name (  
2     column_list  
3 ) engine=table_type
```

- First, you specify the name of the table that you want to create after the “**CREATE TABLE**” clause. The table name must be unique within a database. The “**IF NOT EXISTS**” is an optional part of the statement that allows you to check if the table that you are creating already exists in the database. If this is the case, MySQL will ignore the whole statement and will not create any new table. It is highly recommended that you use “**IF NOT EXISTS**” in every “**CREATE TABLE**” statement for preventing from an error of creating a new table that already exists.
- Second, you specify a list of columns for the table in the **column\_list** section. Columns are separated by a comma (,).
- Third, you need to specify the storage engine for the table in the engine clause. You can use any storage engine such as **InnoDB, MyISAM, HEAP, EXAMPLE, CSV, ARCHIVE, MERGE FEDERATED** or **NDBCLUSTER**. If you don't declare the storage engine explicitly, MySQL will use **InnoDB** by default.



# Creating Column

- To define a column for the table in the “**CREATE TABLE**” statement, you use the following syntax:

```
1 column_name data_type[size] [NOT NULL|NULL] [DEFAULT value]
2 [AUTO_INCREMENT]
```

- The most important components of the syntax above are:
  - The **column\_name** specifies the name of the column. Each column has a specific data type and the size e.g., VARCHAR(255)
  - The **NOT NULL** or **NULL** indicates that the column accepts **NULL** value or not.
  - The **DEFAULT** value is used to specify the default value of the column.
  - The **AUTO\_INCREMENT** indicates that the value of the column is increased automatically whenever a new row is inserted into the table. Each table has one and only one **AUTO\_INCREMENT** column.

# Primary Key

- If you want to set particular columns of the table as the primary key, you use the following syntax:

```
PRIMARY KEY (col1, col2)
```

# Complete Create Table Query

```
1 CREATE TABLE IF NOT EXISTS product (  
2     id INT(11) NOT NULL AUTO_INCREMENT,  
3     product_name VARCHAR(200) DEFAULT NULL,  
4     price DECIMAL DEFAULT NULL,  
5     PRIMARY KEY (id)  
6 ) ENGINE=InnoDB
```

# Altering Table

- You use the “**ALTER TABLE**” statement to change the structure of existing tables. The “**ALTER TABLE**” statement allows you to add a column, drop a column, change the data type of column, add primary key, rename table, and many more. The following illustrates the “**ALTER TABLE**” statement syntax:

```
1 ALTER TABLE table_name actions
```

- To change the structure an existing table :
  - First, you specify the table name, which you want to change, after the “**ALTER TABLE**” clause.
  - Second, you list a set of actions that you want to apply to the table. An action can be anything such as adding a new column, adding primary key, renaming table, etc. The “**ALTER TABLE**” statement allows you to apply multiple actions in a single “**ALTER TABLE**” statement, each action is separated by a comma (,).

# Changing Columns

```
1 ALTER TABLE product
2 CHANGE COLUMN productName Name VARCHAR(200) DEFAULT NULL;
3
```

# Adding New Column into a Table

```
1 ALTER TABLE product
2 ADD COLUMN description VARCHAR(200) NULL
3 AFTER price;
```

# Removing Column from Table

```
1 ALTER TABLE product  
2 DROP COLUMN description;
```



# Renaming Table

```
1 ALTER TABLE product  
2 RENAME TO product2;
```

# Remove Table

- To remove existing tables, you use the MySQL “**DROP TABLE**” statement.

```
1 DROP TABLE IF EXISTS table_name #[, table_name] ...
```

- The “**DROP TABLE**” statement removes a table and its data permanently from the database. In MySQL, you can also remove multiple tables using a single “**DROP TABLE**” statement, each table is separated by a comma (,).
- The “**IF EXISTS**” addition helps you prevent from removing non-existent tables. When you use “**IF EXISTS**” addition, MySQL generates a **NOTE**, which can be retrieved by using the **SHOW WARNING** statement. It is important to note that the “**DROP TABLE**” statement removes all existing tables and issues an error message or a **NOTE** when you have a non-existent table in the list.
- As mentioned above, the “**DROP TABLE**” statement only removes table and its data. However, it does not remove specific user privileges associated with the table. Therefore if a table with the same name is re-created after that, the existing privileges will apply to the new table, which may pose a security risk.