The categories of the SQL statements:

- DDL: for creating tables, relationships, etc.
- DML: for select, insert, update, delete, etc. statements.
- PSM: procedural programming ex. Triggers, functions, stored procedures.
- TCL: used to control and mark transaction boundaries.
- DCL: used to grant or revoke DB permissions to or from users.

In this chapter they will teach you how to create a table using MySQL.

When creating tables, you can add constraints which are:

- Constraints can be defined within the CREATE TABLE statement, or they can be added to the table after it is created using the ALTER table statement.
- **Column and table constraints** include:
  - **PRIMARY KEY** — may not have NULL values
  - **FOREIGN KEY** — may not have NULL values
  - **NULL / NOT NULL**
  - **UNIQUE**
  - **CHECK**
- The **DEFAULT** keyword (not a constraint)

Implementing the constraints.

Relationship Type	CREATE TABLE Constraints
1:N relationship, parent optional	Specify FOREIGN KEY constraint. Set foreign key NULL.
1:N relationship, parent required	Specify FOREIGN KEY constraint. Set foreign key NOT NULL.
1:1 relationship, parent optional	Specify FOREIGN KEY constraint. Specify foreign key UNIQUE constraint. Set foreign key NULL.
1:1 relationship, parent required	Specify FOREIGN KEY constraint. Specify foreign key UNIQUE constraint. Set foreign key NOT NULL.
Casual relationship	Create a foreign key column, but do not specify FOREIGN KEY constraint. If relationship is 1:1, specify foreign key UNIQUE.

How to create, insert, delete, drop and update tables:

1- To create a table, write this statement:

```
Create table table_name (
        Value1    type    other types,
        Value2    type    other types,
);
```

2- To alter a table, write this statement:

```
Alter table table_name (
        Add    value1    type,
        Drop    column    value1,
        Modify column    value1    type,
);
```

3- To drop a table, write this statement:

```
Drop  table  table_name;
```

4- To delete a table, write this statement:

```
Delete  from  table_name
Where  value1 = 0; ex.
```

5- To insert in a table, write this statement:

```
Insert  into  (column1, column2, column3)
Values  (value1, value2, value3);
```

OR

```
Insert  into  (column1, column2, column3)
Select  column1, column2, column3
From  table_name;
```

6- To update a table, write this statement:

```
Update  table_name
Set  value1 = “example”, value2 = “example”
Where value3 = 0; ex.
```

How to create a view table:

```
Create  view  view_table_name  as
        Select  value1 as view_value1, value2 as view_value2
        From  table_name;
```

Updatable views:

Updatable Views
View based on a single table with no computed columns and all non-null columns present in the view.
View based on any number of tables, with or without computed columns, and INSTEAD OF trigger defined for the view.
Possibly Updatable Views
Based on a single table, primary key in view, some required columns missing from view, update and delete may be allowed. Insert is not allowed.
Based on multiple tables, updates may be allowed on the most subordinate table in the view if rows of that table can be uniquely identified.

How to create functions:

```
CREATE FUNCTION dbo.NameConcatenation

-- These are the input parameters
       (
       @FirstName    CHAR(25),
       @LastName     CHAR(25)
       )
RETURNS VARCHAR(60)
AS
BEGIN
       -- This is the variable that will hold the value to be returned
       DECLARE @FullName VARCHAR(60);

       -- SQL statements to concatenate the names in the proper order
       SELECT @FullName = RTRIM(@LastName) + ', ' + RTRIM(@FirstName);

       -- Return the concatentate name
       RETURN @FullName;
END;
```

Triggers:

There are three types of triggers:

1- Before
2- Instead of
3- After

Each type can be declared when you insert, delete or update.

When the trigger is fired the database management systems supplies:

1- Old and new values for the update.
2- Old for the delete.
3- New for the insert.

Stored procedures:

Stored procedures can receive parameters as the functions and the can return results.

The advantages of stored procedures:

1- Greater security
2- Decrease network traffic
3- SQL can be optimized
4- Code sharing
5- Less work
6- Standardized processing
7- Specialization between developers

Triggers vs stored procedures:

Triggers Versus Stored Procedures
Trigger
Module of code that is called by the DBMS when INSERT, UPDATE, or DELETE commands are issued.
Assigned to a table or view.
Depending on the DBMS, may have more than one trigger per table or view.
Triggers may issue INSERT, UPDATE, and DELETE commands and thereby may cause the invocation of other triggers.
Stored Procedure
Module of code that is called by a user or database administrator.
Assigned to a database, but not to a table or a view.
Can issue INSERT, UPDATE, DELETE, and MERGE commands.
Used for repetitive administration tasks or as part of an application.

The difference between triggers, functions and stored procedures:

	User-Defined Functions	Triggers	Stored Procedures
Can accept parameters	Yes	No	Yes
Can return a result value or values	Yes	No	Yes
Can be used in SELECT statements	Yes	No	No
Can use SELECT statements	Yes	Yes	Yes
Can use INSERT statements	No	Yes	Yes
Can use UPDATE statements	No	Yes	Yes
Can use DELETE statements	No	Yes	Yes
Can call a User-Defined Function	Yes	Yes	Yes
Can invoke a Trigger	No	Yes (Indirectly via INSERT, UPDATE, or DELETE)	Yes (Indirectly via INSERT, UPDATE, or DELETE)
Can invoke a Stored Procedure	No	Yes	Yes
Is stored as a database-wide object	Yes	No	Yes
Is stored as a table-specific object	No	Yes	No