Introduction to MySQL

Introduction

- MySQL is a relational database management system based on the Structured Query Language, which is the popular language for accessing and managing the records in the database.
- MySQL is open-source and free software under the GNU license. It is supported by Oracle Company.
- It is developed, marketed, and supported by MySQL AB, a Swedish company, and written in C programming language and C++ programming language

- MySQL is a Relational Database Management System (RDBMS) software that provides many things, which are as follows:
  - It allows us to implement database operations on tables, rows, columns, and indexes.
  - It defines the database relationship in the form of tables (collection of rows and columns), also known as relations.
  - It provides the Referential Integrity between rows or columns of various tables.
  - It allows us to updates the table indexes automatically.
  - It uses many SQL queries and combines useful information from multiple tables for the end-users.

MySQL Features

- MySQL is a relational database management system.
- MySQL is easy to use.
- It is secure.
- Free to download
- MySQL is considered one of the very fast database languages.
- MySQL is compatible to run on many operating systems.
- MySQL is faster, more reliable, and cheaper

Application

- MySQL is a relational database management system based on SQL – Structured Query Language.
- The application is used for a wide range of purposes, including
  - data warehousing
  - e-commerce and
  - logging applications
- The most common use for mySQL however, is for the purpose of a web database. It can be used to store anything from a single record of information to an entire inventory of available products for an online store.

Types of SQL Statements

- Data Definition Language (DDL)
- Data Manipulation Language (DML)
- Data Control Language (DCL)

Data Definition Language (DDL)

- Create : to create tables in the database
- Alter : alters the structure of the database
- Drop : delete tables from the database
- Rename : rename an tables

CREATE

- There are two CREATE statements available in SQL:
  - CREATE DATABASE
  - CREATE TABLE

CREATE DATABASE

A Database is defined as a structured set of data. So, in SQL the very first step to store the data in a well structured manner is to create a database. The CREATE DATABASE statement is used to create a new database in SQL.

Syntax:

```
CREATE DATABASE database_name;
```

database_name: name of the database.

Example Query:

This query will create a new database in SQL and name the database as my_database.

```
CREATE DATABASE my_database;
```

- **CREATE TABLE**
- The CREATE TABLE statement is used to create a table in SQL.
- So while creating tables we have to provide all the information to SQL about the names of the columns, type of data to be stored in columns, size of the data etc.
- Syntax:

```
CREATE TABLE table_name
  (
    column1 data_type(size),
    column2 data_type(size),
    column3 data_type(size),
      ....
);
```

Example

```
CREATE TABLE Students
    (ROLL_NO int(3),  NAME varchar(20), SUBJECT varchar(20));
```

ALTER

- ALTER TABLE is used to add, delete/drop or modify columns in the existing table. It is also used to add and drop various constraints on the existing table.
- **ALTER TABLE – ADD**

  ADD is used to add columns into the existing table.

  Syntax:

  ALTER TABLE table_name

  ADD (Columnname_1 datatype,

  Columnname_2 datatype, …);

Example :

To ADD 2 columns AGE and COURSE to table Student.

ALTER TABLE Student ADD (AGE number(3),COURSE varchar(40));

- **ALTER TABLE – DROP**

  DROP COLUMN is used to drop column in a table. Deleting the unwanted columns from the table.

- Syntax:

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

- Example :

  DROP column COURSE in table Student.

  ALTER TABLE Student DROP COLUMN COURSE;

- **ALTER TABLE-MODIFY**

  It is used to modify the existing columns in a table. Multiple columns can also be modified at once.

- Syntax:

  ALTER TABLE table_name

  MODIFY (fieldname datatype()...);

- Example:

  MODIFY column COURSE in table Student

  ALTER TABLE Student MODIFY COURSE varchar(20);

DROP

DROP is used to delete a whole database or just a table.

Syntax:
DROP object object_name

Examples:

DROP TABLE table_name;
table_name: Name of the table to be deleted.

DROP DATABASE database_name;
database_name: Name of the database to be deleted.

RENAME

- Sometimes we may want to rename our table to give it a more relevant name. For this purpose we use rename command.
- **Syntax**:

  rename table <old TableName> to <New TableName>;

- **QUERY:**

  Change the name of column NAME to FIRST_NAME in table Student.

  RENAME STU TO STUDENT;

Data Manipulation Language (DML)

- SELECT: It is used to retrieve data from the database.
- INSERT : It is used to insert data into a table.
- UPDATE: It is used to update existing data within a table.
- DELETE : It is used to delete records from a database table.

SELECT

- The SELECT Statement in SQL is used to retrieve or fetch data from a database.
- We can fetch either the entire table or according to some specified rules.
- The data returned is stored in a result table. This result table is also called result-set.
- Basic Syntax:

```
SELECT column1,column2 FROM table_name
```

column1 , column2: names of the fields of the table

table_name: from where we want to fetch

This query will return all the rows in the table with fields column1 , column2.

- **To fetch the entire table or all the fields in the table:**

  SELECT * FROM table_name;

- **Example :**

  Query to fetch the fields ROLL_NO, NAME, AGE from the table Student:

  SELECT ROLL_NO, NAME, AGE FROM Student;

INSERT

- The INSERT INTO statement of SQL is used to insert a new row in a table.
- There are two ways of using INSERT INTO statement for inserting rows:

  1. Only values: First method is to specify only the value of data to be inserted without the column names.

  Syntax :

  INSERT INTO table_name VALUES (value1, value2, value3,…);

  table_name: name of the table.

  value1, value2,.. : *value of first column, second column,. for the new record*

  Example :

  INSERT INTO Student VALUES (‘5','HARSH’,’WEST BENGAL’,’XXXXXXXXXX’,’19’);

2. Column names and values both: In the second method we will specify both the columns which we want to fill and their corresponding values as shown below:

```
INSERT INTO table_name (column1, column2, column3,..)
    VALUES ( value1, value2, value3,..);
```

table_name: name of the table.

column1: name of first column, second column …

value1, value2, value3 : value of first column, second column,… for the new record

Example

```
INSERT INTO Student (ROLL_NO, NAME, Age) VALUES
(‘5′,’PRATIK’,’19’);
```

UPDATE

- The UPDATE statement in SQL is used to update the data of an existing table in database.
- We can update single columns as well as multiple columns using UPDATE statement as per our requirement.
- **Basic Syntax**

```
UPDATE table_name SET column1 = value1, column2 = value2,...
    WHERE condition;
```

  - **table_name:** name of the table
  - **column1:** name of first , second, third column....
  - **value1:** new value for first, second, third column....
  - **condition:** condition to select the rows for which the values of columns needs to be updated.

- Examples according to scenario:

1. Updating single column: Update the column NAME and set the value to ‘PRATIK’ in all the rows where Age is 20.

UPDATE Student SET NAME = 'PRATIK' WHERE Age = 20;

2. Updating multiple columns: Update the columns NAME to ‘PRATIK’ and ADDRESS to ‘SIKKIM’ where ROLL_NO is 1.

UPDATE Student SET NAME = 'PRATIK', ADDRESS = 'SIKKIM' WHERE ROLL_NO = 1;

3. Omitting WHERE clause: If we omit the WHERE clause from the update query then all of the rows will get updated.

UPDATE Student SET NAME = 'PRATIK';

DELETE

- The DELETE Statement in SQL is used to delete existing records from a table.
- We can delete a single record or multiple records depending on the condition we specify in the WHERE clause.

Basic Syntax:

```
DELETE FROM table_name WHERE some_condition;
```

- **table_name:** name of the table
- **some_condition:** condition to choose particular record.

- Example according to scenario:

  1 . Deleting single record: Delete the rows where NAME = ‘Ram’. This will delete only the first row.

  DELETE FROM Student WHERE NAME = 'Ram';

  2. Deleting multiple records: Delete the rows from the table Student where Age is 20. This will delete 2 rows(third row and fifth row).

  DELETE FROM Student WHERE Age = 20;

  3. Delete all of the records: There are two queries to do this as shown below,

  DELETE FROM Student;

Data Control Language(DCL)

- DCL is the abstract of Data Control Language. Data Control Language includes commands such as GRANT, and is concerned with rights, permissions, and other controls of the database system. DCL is used to grant/revoke permissions on databases and their contents.
- DCL is used to control the database transaction. DCL statements allow you to control who has access to a specific object in your database.
- 1. GRANT: allow users access privileges to the database
- 2. REVOKE: withdraw users access privileges given by using the GRANT command

GRANT

- It provides the user's access privileges to the database. The MySQL database offers both the administrator and user a great extent of the control options. It creates an entry in the security system that allows a user in the current database to work with data in the current database or execute specific statements.
- Syntax:

  GRANT privileges_names ON object TO user;

- Parameters Used:
  - privileges_name: These are the access rights or privileges granted to the user.
  - object:It is the name of the database object to which permissions are being granted. In the case of granting privileges on a table, this would be the table name.
  - user:It is the name of the user to whom the privileges would be granted.

Example :

- **Granting SELECT Privilege to a User in a Table:**

  GRANT SELECT ON Users TO'Amit'@'localhost;

- **Granting more than one Privilege to a User in a Table:**

  GRANT SELECT, INSERT, DELETE, UPDATE ON Users TO 'Amit'@'localhost;

- **Granting All the Privilege to a User in a Table:**

  GRANT ALL ON Users TO 'Amit'@'localhost;

- **Granting a Privilege to all Users in a Table:**

  GRANT SELECT ON Users TO '*'@'localhost;

REVOKE

•The REVOKE statement enables system administrators and to revoke (back permission) the privileges from MySQL accounts.
•Syntax:

REVOKE privileges ON object FROM user;

•Parameters Used:

- •object:It is the name of the database object from which permissions are being revoked. In the case of revoking privileges from a table, this would be the table name.
- •user:It is the name of the user from whom the privileges are being revoked.

Example :

- **Revoking SELECT Privilege to a User in a Table:**

  REVOKE SELECT ON users TO 'Amit'@localhost';

- **Revoking more than Privilege to a User in a Table:**

  REVOKE SELECT, INSERT, DELETE, UPDATE ON Users TO 'Amit'@'localhost;

- **Revoking All the Privilege to a User in a Table:**

  REVOKE ALL ON Users TO 'Amit'@'localhost;

- **Revoking a Privilege to all Users in a Table:**

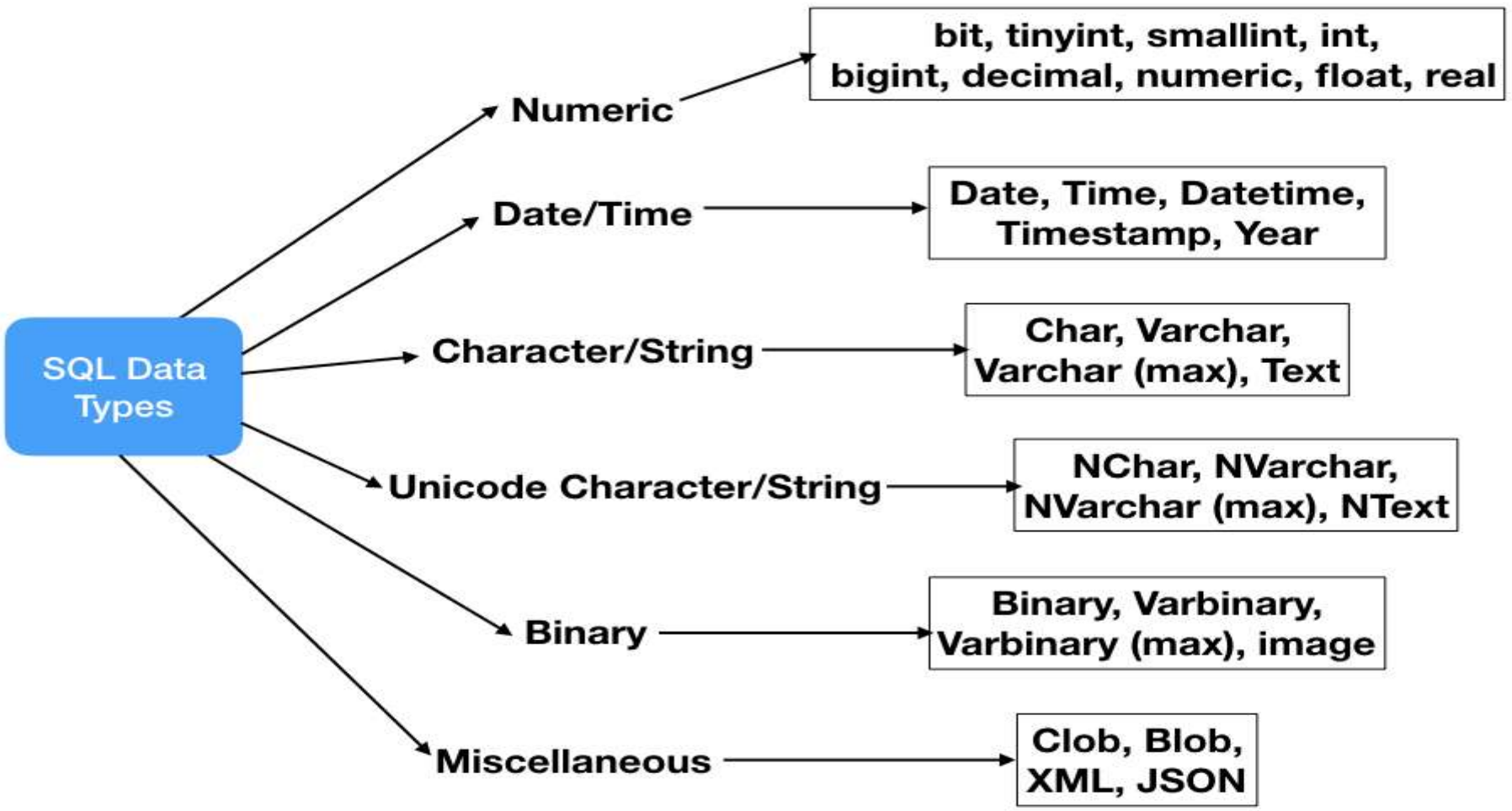
  REVOKE SELECT ON Users TO '*'@'localhost;

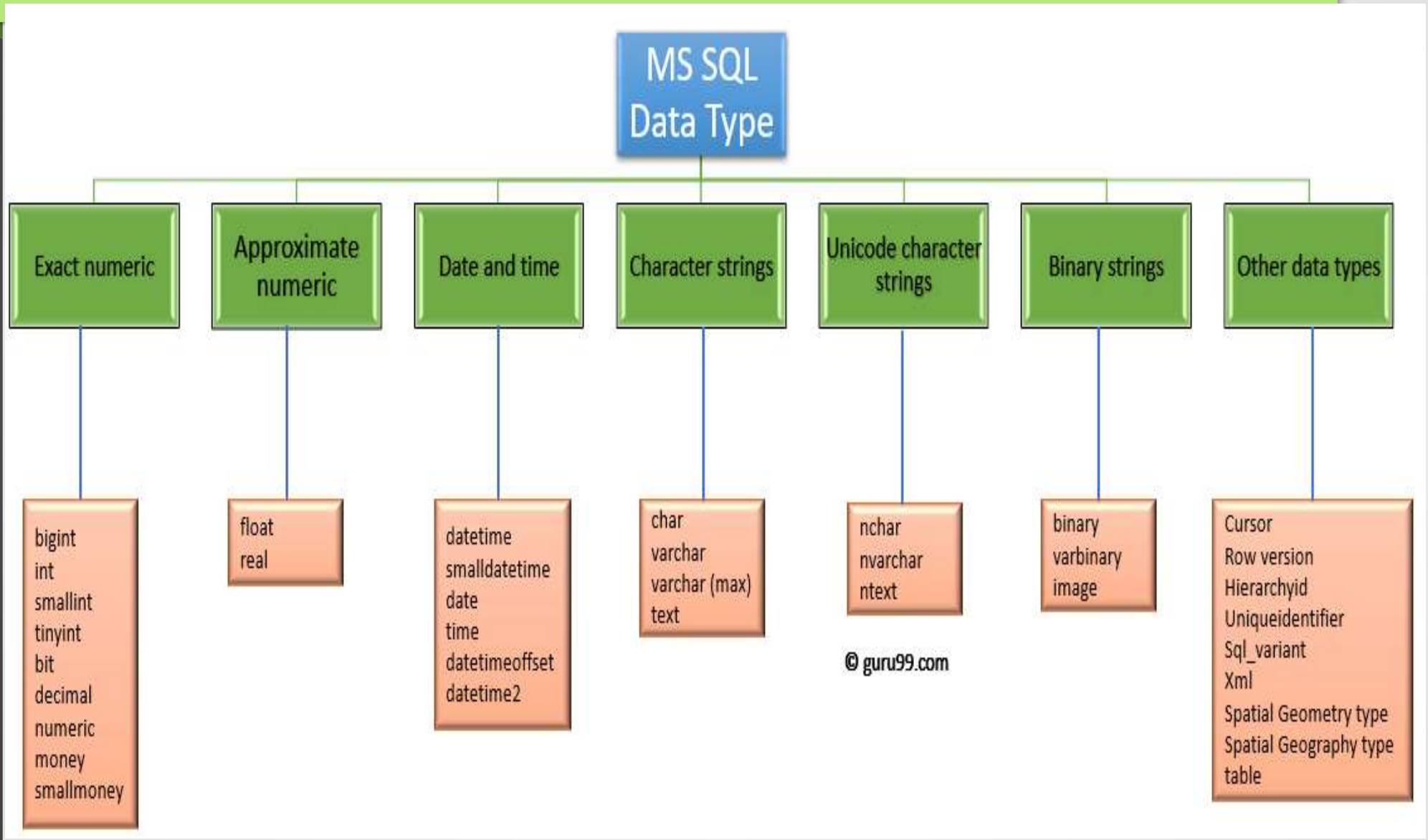
MYSQL DATA TYPE

A database table contains multiple columns with specific data types such as numeric or string. MySQL provides more data types other than just numeric and string. Each data type in MySQL can be determined by the following characteristics:

- The kind of values it represents.
- The space that takes up and whether the values are a fixed-length or variable length.
- The values of the data type can be indexed or not.
- How MySQL compares the values of a specific data type.

MYSQL Data Types




MS SQL Data Type
Exact numeric
Approximate numeric
Date and time
Character strings
Unicode character strings
Binary strings
Other data types
bigint
int
smallint
tinyint
bit
decimal
numeric
money
smallmoney
float
real
datetime
smalldatetime
date
time
datetimeoffset
datetime2
char
varchar
varchar (max)
text
nchar
nvarchar
ntext
binary
varbinary
image
Cursor
Row version
Hierarchyid
Uniqueidentifier
Sql_variant
Xml
Spatial Geometry type
Spatial Geography type
table
© guru99.com

MySQL numeric data types

- In MySQL, you can find all SQL standard numeric types including exact number data type and approximate numeric data types including integer, fixed-point and floating-point.MySQL also has BIT data type for storing bit values.

Numeric Types	Description
TINYINT	A very small integer
SMALLINT	A small integer
MEDIUMINT	A medium-sized integer
INT	A standard integer
BIGINT	A large integer
DECIMAL	A fixed-point number
FLOAT	A single-precision floating point number
DOUBLE	A double-precision floating point number
BIT	A bit field

MySQL Boolean data type

MySQL does not have the built-in BOOLEAN or BOOL data type. To represent boolean values, MySQL uses the smallest integer type which is TINYINT(1). In other words, BOOLEAN and BOOL are synonyms for TINYINT(1).

MySQL String data types

String Types	Description
CHAR	A fixed-length nonbinary (character) string
VARCHAR	A variable-length non-binary string
BINARY	A fixed-length binary string
VARBINARY	A variable-length binary string
TINYBLOB	A very small BLOB (binary large object)
BLOB	A small BLOB
MEDIUMBLOB	A medium-sized BLOB
LONGBLOB	A large BLOB
TINYTEXT	A very small non-binary string
TEXT	A small non-binary string
MEDIUMTEXT	A medium-sized non-binary string
LONGTEXT	A large non-binary string
ENUM	An enumeration; each column value may be assigned one enumeration member
SET	A set; each column value may be assigned zero or more SET members

MySQL date and time data types

MySQL provides types for date and time as well as the combination of date and time.

Date and Time Types	Description
DATE	A date value in CCYY-MM-DD format
TIME	A time value in hh:mm:ss format
DATETIME	A date and time value inCCYY-MM-DD hh:mm:ssformat
TIMESTAMP	A timestamp value in CCYY-MM-DD hh:mm:ss format
YEAR	A year value in CCYY or YY format

JSON data type

- The native JSON data type provides automatic validation of JSON documents and optimal storage format.

MySQL Constraints

Types of MySQL Constraints: Constraints in MySQL is classified into two types:

- **Column Level Constraints:** These constraints are applied only to the single column that limits the type of particular column data.
- **Table Level Constraints:** These constraints are applied to the entire table that limits the type of data for the whole table.
-

Constraint	Column	Table
CHECK	Yes	Yes
NOT NULL	Yes	No*
UNIQUE	Yes	Yes
PRIMARY KEY	Yes	Yes
FOREIGN KEY	No	Yes

How to create constraints in MySQL

- We can define the constraints during a table created by using the CREATE TABLE statement. The following are the most common constraints used in the MySQL:
- NOT NULL
- DEFAULT
- PRIMARY KEY
- UNIQUE
- FOREIGN KEY

SYNTAX:

CREATE TABLE new_table_name (

col_name1 datatype constraint,

col_name2 datatype constraint,

col_name3 datatype constraint,

.....

);