MySQL Data Types

In MySQL, data types define the kind of data that can be stored in a column. Choosing the correct data type is crucial for optimizing storage and ensuring data integrity. MySQL data types are broadly categorized into three groups:

1. Numeric Data Types

These are used to store numbers, including integers, decimals, and floating-point numbers.

Integer Types

- **TINYINT**: Stores very small integers.
  - Range: `-128 to 127` (signed) or `0 to 255` (unsigned).
  - Example: Storing age categories (e.g., 0 for children, 1 for adults).
- **SMALLINT**: Stores small integers.
  - Range: `-32,768 to 32,767` (signed) or `0 to 65,535` (unsigned).
  - Example: Storing small counts like the number of items in stock.
- **MEDIUMINT**: Stores medium-sized integers.
  - Range: `-8,388,608 to 8,388,607` (signed) or `0 to 16,777,215` (unsigned).
- **INT (INTEGER)**: Stores standard integers.
  - Range: `-2,147,483,648 to 2,147,483,647` (signed) or `0 to 4,294,967,295` (unsigned).
  - Example: Storing user IDs or product IDs.
- **BIGINT**: Stores very large integers.
  - Range: `-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807` (signed).
  - Example: Storing large numbers like population or financial data.

Floating-Point Types

- **FLOAT**: Stores approximate decimal values.
  - Example: Storing measurements like weight or height.
- **DOUBLE (REAL)**: Stores double-precision floating-point numbers.
  - Example: Storing precise scientific calculations.
- **DECIMAL (NUMERIC)**: Stores exact decimal values.
  - Example: Storing financial data like prices or salaries.

2. String Data Types

These are used to store text, characters, or binary data.

Character Types

- **CHAR(size)**: Fixed-length string.
  - Example: Storing fixed-length codes like country codes (`'US'` , `'IN'`).
- **VARCHAR(size)**: Variable-length string.

  - Example: Storing names, email addresses, or descriptions.

Text Types

- **TINYTEXT**: Stores very small text (up to 255 characters).
  - Example: Storing short comments or tags.
- **TEXT**: Stores text up to 65,535 characters.
  - Example: Storing blog posts or articles.
- **MEDIUMTEXT**: Stores medium-length text (up to 16,777,215 characters).
  - Example: Storing long documents or reports.
- **LONGTEXT**: Stores very large text (up to 4GB).
  - Example: Storing books or large logs.

Binary Types

- **BINARY(size)**: Fixed-length binary data.
- **VARBINARY(size)**: Variable-length binary data.
- **BLOB (Binary Large Object)**: Stores binary data like images or files.
  - Types: TINYBLOB, BLOB, MEDIUMBLOB, LONGBLOB (varying sizes).

3. Date and Time Data Types

These are used to store dates, times, or both.

- **DATE**: Stores date values (`YYYY-MM-DD`).
  - Example: Storing birthdates or event dates.
- **DATETIME**: Stores date and time (`YYYY-MM-DD HH:MM:SS`).
  - Example: Storing timestamps for transactions.
- **TIMESTAMP**: Stores date and time, automatically updated to the current time when a record is modified.
  - Example: Storing last modified times.
- **TIME**: Stores time values (`HH:MM:SS`).
  - Example: Storing durations or time intervals.
- **YEAR**: Stores year values (`YYYY`).
  - Example: Storing manufacturing years or academic years.

Key Points for Beginners

1. **Choose the Right Data Type**: Use the smallest data type that fits your data to save storage.
2. **Signed vs. Unsigned**: Signed types allow negative values, while unsigned types only allow positive values.
3. **Length Specification**: For types like `VARCHAR(25)` , the number in parentheses specifies the maximum length.
4. **Default Values**: You can set default values for columns to ensure data consistency.

Example Table with Data Types

```
CREATE TABLE employees (
    employeeID INT UNSIGNED AUTO_INCREMENT PRIMARY KEY, -- Unique ID
    name VARCHAR(50),                                   -- Employee name
    age TINYINT UNSIGNED,                               -- Age (0-255)
    salary DECIMAL(10, 2),                              -- Salary with 2 decimal places
    hireDate DATE,                                      -- Date of hiring
    isActive BOOLEAN                                    -- Active status (1 or 0)
);
```

This table demonstrates how to use different data types effectively.

Explanation of MySQL Data Types with Storage Bytes

Below is a detailed explanation of MySQL data types along with their storage requirements in bytes. This will help you understand how much space each data type consumes.

Numeric Data Types

Data Type	Description	Storage (Bytes)	Range (Signed)	Range (Unsigned)
TINYINT	Very small integer	1	-128 to 127	0 to 255
SMALLINT	Small integer	2	-32,768 to 32,767	0 to 65,535
MEDIUMINT	Medium-sized integer	3	-8,388,608 to 8,388,607	0 to 16,777,215
INT (INTEGER)	Standard integer	4	-2,147,483,648 to 2,147,483,647	0 to 4,294,967,295
BIGINT	Large integer	8	-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807	0 to 18,446,744,073,709,551,615
FLOAT	Single-precision floating-point	4	Approx. ±3.402823466E+38	Approx. ±3.402823466E+38
DOUBLE	Double-precision floating-point	8	Approx. ±1.7976931348623157E+308	Approx. ±1.7976931348623157E+308
DECIMAL(M, D)	Exact fixed-point number	Varies (M+2 bytes)	Depends on precision	Depends on precision

String Data Types

Data Type	Description	Storage (Bytes)	Maximum Length
CHAR(M)	Fixed-length string	M bytes	0 to 255 characters
VARCHAR(M)	Variable-length string	L + 1 bytes (L = actual length)	0 to 65,535 characters (depends on row size)
TINYTEXT	Very small text	L + 1 bytes	Up to 255 characters
TEXT	Small text	L + 2 bytes	Up to 65,535 characters
MEDIUMTEXT	Medium-length text	L + 3 bytes	Up to 16,777,215 characters
LONGTEXT	Large text	L + 4 bytes	Up to 4,294,967,295 characters
BINARY(M)	Fixed-length binary data	M bytes	0 to 255 bytes
VARBINARY(M)	Variable-length binary data	L + 1 bytes	0 to 65,535 bytes
TINYBLOB	Very small binary object	L + 1 bytes	Up to 255 bytes
BLOB	Small binary object	L + 2 bytes	Up to 65,535 bytes
MEDIUMBLOB	Medium binary object	L + 3 bytes	Up to 16,777,215 bytes
LONGBLOB	Large binary object	L + 4 bytes	Up to 4,294,967,295 bytes

Date and Time Data Types

Data Type	Description	Storage (Bytes)	Range
DATE	Stores date (`YYYY-MM-DD`)	3	1000-01-01 to 9999-12-31
DATETIME	Stores date and time (`YYYY-MM-DD HH:MM:SS`)	8	1000-01-01 00:00:00 to 9999-12-31 23:59:59
TIMESTAMP	Stores timestamp	4	1970-01-01 00:00:01 UTC to 2038-01-19 03:14:07 UTC
TIME	Stores time (`HH:MM:SS`)	3	-838:59:59 to 838:59:59
YEAR	Stores year (`YYYY`)	1	1901 to 2155

Example from Your File

In your file, the `books` table uses the following data types:

```
CREATE TABLE books (
    bookID INT,                  -- 4 bytes
    title VARCHAR(25),           -- Up to 26 bytes (25 + 1 for length)
    author VARCHAR(25),          -- Up to 26 bytes
    genre VARCHAR(25),           -- Up to 26 bytes
    publicationYear INT          -- 4 bytes
);
```

- **bookID**: `INT` uses 4 bytes.
- **title, author, genre**: `VARCHAR(25)` uses up to 26 bytes each (25 for characters + 1 for length).
- **publicationYear**: `INT` uses 4 bytes.

Key Takeaways

1. **Storage Efficiency**: Choose the smallest data type that fits your data to save space.
2. **String Length**: For `VARCHAR` , the actual storage depends on the length of the stored string.
3. **Date and Time**: Use `DATE` or `TIMESTAMP` based on whether you need just the date or both date and time.