MySQL Data Types

To see all data types of MySQL, visit:

MySQL :: MySQL 8.0 Reference Manual :: 13 Data Types

NUMERIC DATA TYPES

- TINYINT: Small integer (-128 to 127 or 0 to 255 UNSIGNED). Used for age, small counters.
- SMALLINT: Integer (-32,768 to 32,767 or 0 to 65,535 UNSIGNED). Used for quantities.
- MEDIUMINT: Medium integer (-8,388,608 to 8,388,607). Used for medium-sized values.
- INT / INTEGER: (-2,147,483,648to2,147,483,647) Standard integer. Used for IDs, salary, counts.
- BIGINT: (-9,223,372,036,854,775,808to 9,223,372,036,854,775,807) Large integer. Used for phone numbers, large IDs.
- DECIMAL / NUMERIC: Exact fixed-point number. Best for money.
- FLOAT: Decimal number - with precision to23 digits. Used for ratings.
- DOUBLE: Approximate decimal number with double precision.

CHARACTER DATA TYPES

- CHAR(n): Fixed-length string. Range: 0 to 255 characters. Used for gender, country codes.
- VARCHAR(n): Variable-length string. Range: 0 to 65,535 characters (depends on row size & charset). Used for names, emails.
- TINYTEXT: Very small text. Max length: 255 characters.
- TEXT: Large text. Max length: 65,535 characters (~64 KB). Used for descriptions.
- MEDIUMTEXT: Medium-length text. Max length: 16,777,215 characters (~16 MB).
- LONGTEXT: Very large text. Max length: 4,294,967,295 characters (~4 GB).

BINARY DATA TYPES

- BINARY(n): Fixed-length binary data. Range: 0 to 255 bytes.
- VARBINARY(n): Variable-length binary data. Range: 0 to 65,535 bytes.
- TINYBLOB: Very small binary data. Max length: 255 bytes.
- BLOB: Binary Large Object. Max length: 65,535 bytes (~64 KB). Used for images/files.
- MEDIUMBLOB: Medium binary data. Max length: 16,777,215 bytes (~16 MB).

- LONGBLOB: Very large binary data. Max length: 4,294,967,295 bytes (~4 GB).

DATE AND TIME DATA TYPES

- DATE: Stores date. Range: 1000-01-01 to 9999-12-31. Format: YYYY-MM-DD.
- TIME: Stores time. Range: -838:59:59 to 838:59:59. Format: HH:MM:SS.
- DATETIME: Stores date and time. Range: 1000-01-01 00:00:00 to 9999-12-31 23:59:59.
- TIMESTAMP: Stores date and time. Range: 1970-01-01 00:00:01 UTC to 2038-01-19 03:14:07 UTC.

OTHER DATA TYPES

- BOOLEAN / BOOL: Stores TRUE or FALSE.
- ENUM: Stores one value from predefined list.
- SET: Stores multiple values from predefined list.
- JSON: Stores JSON formatted data.

REGULAR EXPRESSION

- REGEXP is used to perform pattern matching using regular expressions in MySQL.

- SELECT column_name FROM table_name WHERE column_name REGEXP 'pattern';

- **Anchors**
- • ^ → Matches beginning of string
- • $ → Matches end of string

- **Character Classes**
- • [abc] → Matches a, b, or c
- • [a-z] → Matches any lowercase letter
- • [^abc] → Matches anything except a, b, c
- **Wildcard & Quantifiers**
- • . → Matches any single character
- • * → Zero or more occurrences
- • + → One or more occurrences
- • ? → Zero or one occurrence
- **Repetition Counts**
- • {n} → Exactly n times
- • {n,} → At least n times
- • {n,m} → Between n and m times
- **Alternation (OR)**
- • a|b → Matches either a or b
- **Grouping**
- • (abc) → Groups abc as single unit
- **Predefined Character Classes**
- • [[:digit:]] → Digits (0-9)
- • [[:alpha:]] → Letters
- • [[:alnum:]] → Letters & digits

- • [[:space:]] → White space
- • [[:lower:]] → Lowercase letters
- • [[:upper:]] → Uppercase letters
- **Common REGEXP Examples**
- • Names starting with A: WHERE name REGEXP '^A'
- • Ends with n: WHERE name REGEXP 'n$'
- • Contains digit: WHERE col REGEXP '[0-9]'
- • Starts with a or s: WHERE col REGEXP '^[as]'
- • Exact 5 characters: WHERE col REGEXP '^.{5}$'
- **REGEXP vs LIKE**
- • LIKE is simple wildcard matching
- • REGEXP supports complex patterns
- **Important Notes**
- MySQL REGEXP is case-insensitive by default (unless BINARY used)
- REGEXP is slower than LIKE for simple matches

MySQL LIKE Wildcards – Complete Notes

- This document explains all wildcards used with the LIKE operator in MySQL. These notes are suitable for learning, teaching, interviews, and quick revision.

- **1. Percent (%) Wildcard**

- % matches ZERO or MORE characters.

  Common Use Cases:
  - Starts with
  - Ends with
  - Contains
  - Any length of characters

- Examples:

  ```
  -- Names starting with A
  SELECT * FROM employees WHERE name LIKE
  'A%';

  -- Names ending with a
  SELECT * FROM employees WHERE name LIKE
  '%a';

  -- Names containing 'ha'
  SELECT * FROM employees WHERE name LIKE
  '%ha%';

  -- Matches all values
  SELECT * FROM employees WHERE name LIKE
  '%';
  ```

- Tip: % represents any length of characters (0 or more).

- **2. Underscore (_) Wildcard**
- _ matches EXACTLY ONE character.

  Common Use Cases:
  - Fixed-length patterns
  - Masking a single character

- Examples:

  ```
  -- Names with exactly 4 characters
  SELECT * FROM employees WHERE name LIKE '____';

  -- Names starting with A and having total 3 characters
  SELECT * FROM employees WHERE name LIKE 'A__';

  -- Names where second character is 'a'
  SELECT * FROM employees WHERE name LIKE '_a%';
  ```

- Tip: _ always represents one and only one character.

- **Percent (%) vs Underscore (_)**

  % → Zero or more characters
  _ → Exactly one character

Example:
'A%' → Any name starting with A
'A_' → Two-letter name starting with A
'%_%' → Any non-empty value

- **Common Interview Traps**

LIKE '%' → Returns all rows
LIKE '_%' → Matches all non-empty values
LIKE '__' → Matches values with exactly two characters

- **Important Notes**

- LIKE is case-insensitive by default in MySQL (depends on collation)
- LIKE is slower than = for exact matches
- LIKE does not support advanced patterns
- Use REGEXP for complex pattern matching

- **One-Line Summary**
- % matches any number of characters, _ matches exactly one character in MySQL LIKE.