MySQL Database Design

By

Mohammed Mukthar Ahmed

1

Contents

- Create New MySQL Database
- Naming Conventions
- Available Databases
- Using a Database
  - Available Tables
- Create New Table
- Data Types
  - Numeric, Spatial Extensions
  - Date & Time
  - String, Enum & Set
- Column Attributes

2

Contents

- Column Attributes
  - AUTO_INCREMENT, DEFAULT, UNIQUE,
  - PRIMARY KEY, COMMENT
- Description of Table
- Insert Data into Table
  - Load Data
- Using MySQL in Batch Mode

Create New MySQL Database

- You need to use **mysqladmin** to create MySQL database from the DOS prompt.

```
C:\> mysqladmin create petstore
```

- You can also type the query in mysql> prompt like this:

```
mysql> CREATE DATABASE petstore;
Query OK, 0 rows affected (0.03sec)
```

Naming Conventions

- Database, table, index, column, and alias names are identifiers.
- The following table describes the maximum length for each identifier.

Identifier	Maximum Length
Database	64
Table	64
Column	64
Index	64
Alias	255

5

Naming Conventions

- There are some restrictions on the characters that may appear in identifiers:
  - No identifier can contain ASCII 0 (0x00) or a byte with a value of 255.
  - Before MySQL 4.1, identifier quote characters should not be used in identifiers. As of 4.1, the use of identifier quote characters in identifiers is permitted, although it is best to avoid.
  - Database, table, and column names should not end with space characters.
  - Database names cannot contain ‘/’, ‘\’, ‘.’, or characters that are not allowed in a directory name.
  - Table names cannot contain ‘/’, ‘\’, ‘.’, or characters that are not allowed in a filename.

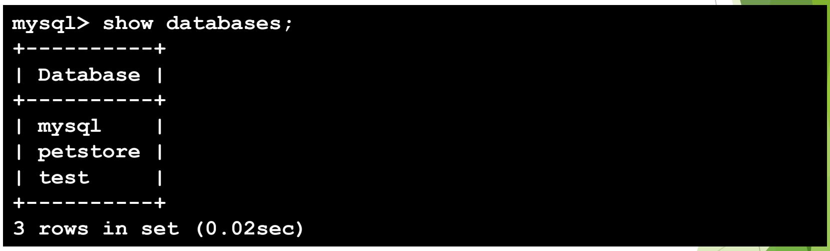
6

Naming Conventions

- There are some restrictions on the characters that may appear in identifiers:
  - Identifier quoting was introduced in MySQL 3.23.6 to allow use of identifiers that are reserved words or that contain special characters.
  - A name may consist of alphanumeric characters from the current character set, ‘_’, and ‘$’
  - A name may start with any character that is legal in a name. In particular, a name may start with a digit; this differs from many other database systems!
  - Database and table names are not case sensitive on some platforms, but this should be avoided.

Available Databases

- To show available databases in MySQL use the command **show databases**



```
mysql> show databases;
+----------+
| Database |
+----------+
| mysql    |
| petstore |
| test     |
+----------+
3 rows in set (0.02sec)
```

Using a Database

- Use the database by typing **USE petstore**
- Type **SHOW tables** to see what tables are available in the database.

```
mysql> USE petstore;
Database changed
mysql> SHOW tables;
Empty set (0.00sec)
```

Create New Table

- To create a table, the CREATE TABLE statement is used.
- You must have:
  - CREATE TABLE privilege.
  - A storage area.
- Syntax:

```
CREATE [TEMPORARY] TABLE [IF NOT EXIST] table
          (column datatype [ATTRIBUTES], ...)
```

- You specify:
  - Table Name, Column Name
  - Data Type and Attributes (if any).

Create New Table

- A TEMPORARY table is visible only to the current connection, and is dropped automatically when the connection is closed.
- The keywords IF NOT EXISTS prevent an error from occurring if the table exists.

```
mysql> CREATE TABLE species (id INT NOT NULL AUTO_INCREMENT,
name varchar(30) NOT NULL, primary key(id));
```

```
mysql> CREATE TABLE pet(id INT NOT NULL AUTO_INCREMENT,
sp_id INT NOT NULL, sex CHAR(1) NOT NULL, price DECIMAL(4,2)
NOT NULL, primary key(id));
```

11

Data Types

- MySQL supports a number of data types in the following categories :
  - Numeric Type,
  - Date & Time Type, and
  - String (Character) Type

12

Data Types - Numeric

- MySQL supports all of the standard SQL numeric data types.
  - These types include the exact numeric data types (INTEGER, SMALLINT, DECIMAL, and NUMERIC), as well as the approximate numeric data types (FLOAT, REAL, and DOUBLE PRECISION).
  - The keyword INT is a synonym for INTEGER, and the keyword DEC is a synonym for DECIMAL.
  - As an extension to the SQL standard, MySQL also supports the integer types TINYINT, MEDIUMINT, and BIGINT.

Data Types - Numeric

- Table with the required storage and range for each of the integer types.

Type	Bytes	Min. Value	Max. Value
		(Signed / Unsigned)	(Signed / Unsigned)
TINYINT or BIT	1	-128	127
		0	255
SMALLINT	2	-32768	32767
		0	65535
MEDIUMINT	3	-8388608	8388607
		0	16777215
INT	4	-2147483648	2147483647
		0	4294967295
BIGINT	8	-9223372036854775808	9223372036854775807

Data Types - Numeric

- The approximate numeric data types (FLOAT, REAL, and DOUBLE PRECISION) storage requirement are as follows:

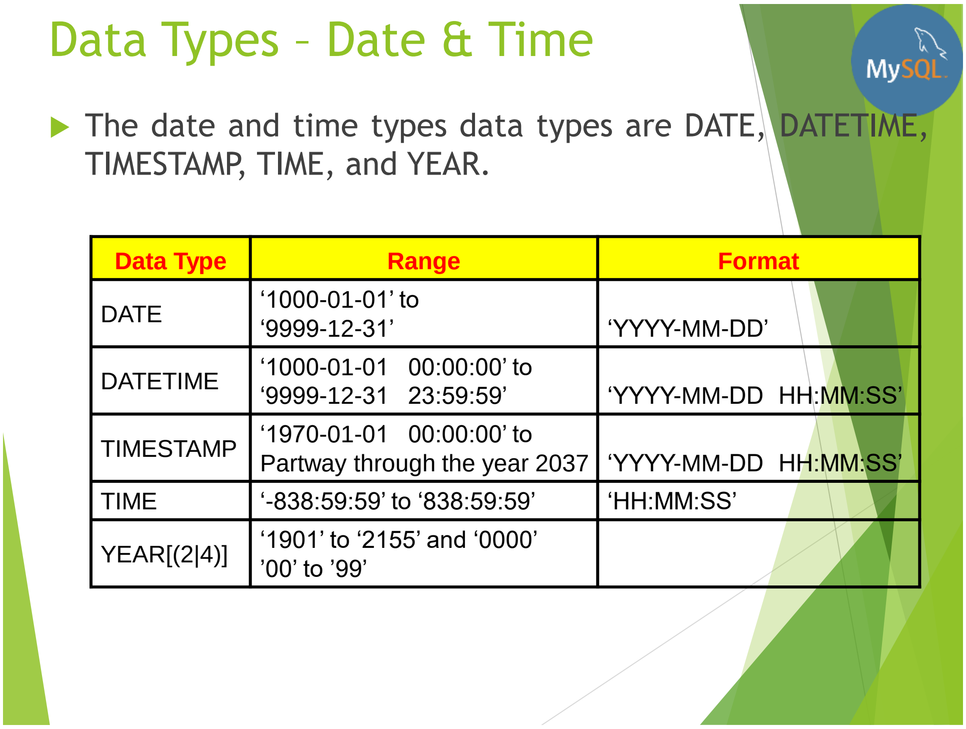
Data Type	Storage Requirement
FLOAT(p)	4 or 8 bytes
FLOAT	4 bytes
DOUBLE [PRECISION], REAL	8 bytes
DECIMAL(M,D), NUMERIC(M,D)	Varies

Data Types – Numeric (Spatial Extensions)

- MySQL 4.1 and up also supports extensions for handing **spatial** (pertaining to space) data .

- Several of the data type descriptions use these conventions.
  - *M* indicates the maximum display width for integer types. For floating-point and fixed-point types, *M* is the total number of digits.
  - *D* applies to floating-point and fixed-point types and indicates the number of digits following the decimal point. The maximum possible value is 30, but should be no greater than *M*–2.

- If you specify ZEROFILL for a numeric column, MySQL automatically adds the UNSIGNED attribute to the column.

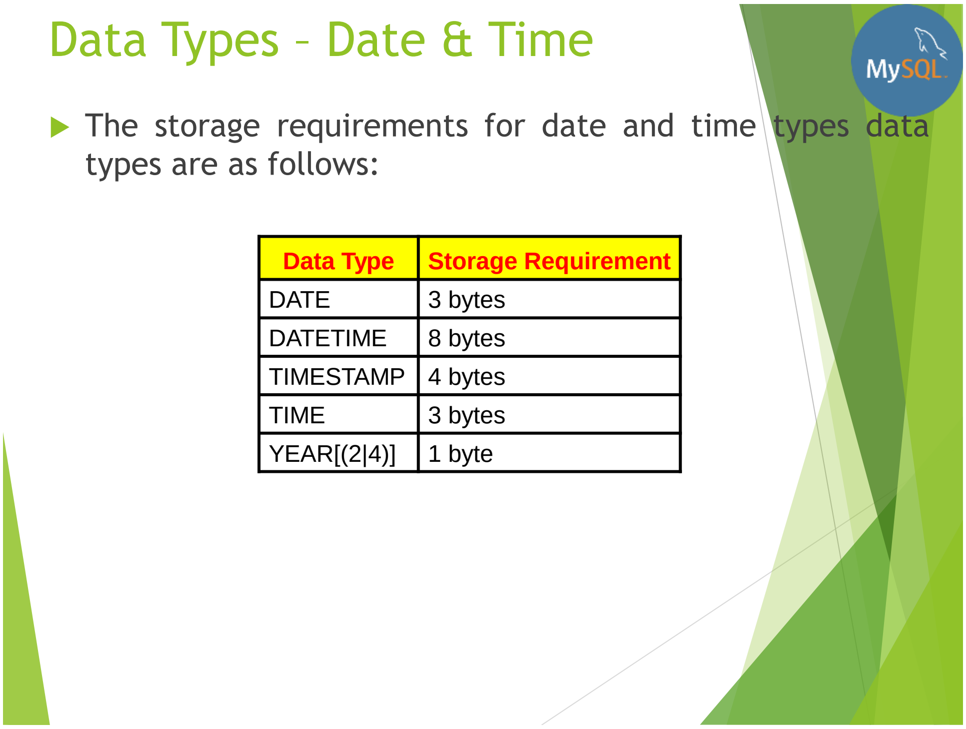


Data Types – Date & Time

- The date and time types data types are DATE, DATETIME, TIMESTAMP, TIME, and YEAR.

Data Type	Range	Format
DATE	‘1000-01-01’ to ‘9999-12-31’	‘YYYY-MM-DD’
DATETIME	‘1000-01-01 00:00:00’ to ‘9999-12-31 23:59:59’	‘YYYY-MM-DD HH:MM:SS’
TIMESTAMP	‘1970-01-01 00:00:00’ to Partway through the year 2037	‘YYYY-MM-DD HH:MM:SS’
TIME	‘-838:59:59’ to ‘838:59:59’	‘HH:MM:SS’
YEAR[(2|4)]	‘1901’ to ‘2155’ and ‘0000’ ’00’ to ’99’	

17



Data Types – Date & Time

- The storage requirements for date and time types data types are as follows:

Data Type	Storage Requirement
DATE	3 bytes
DATETIME	8 bytes
TIMESTAMP	4 bytes
TIME	3 bytes
YEAR[(2|4)]	1 byte

18

Data Types – String

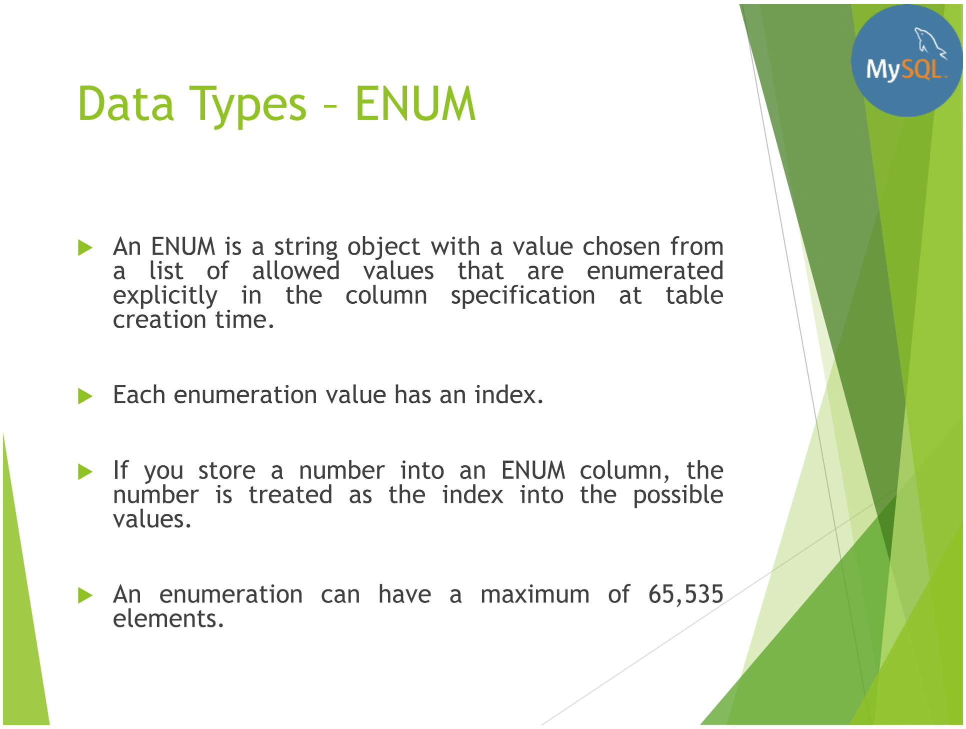
- The string types are CHAR, VARCHAR, BINARY, VARBINARY, BLOB, TEXT, ENUM, and SET.
- The CHAR and VARCHAR Types.
  - The CHAR and VARCHAR types are similar, but differ in the way they are stored and retrieved.
- The BINARY and VARBINARY Types.
  - The BINARY and VARBINARY types are similar to CHAR and VARCHAR, except that they contain binary strings rather than non-binary strings.
  - They contain byte strings rather than character strings.

19

Data Types – String

- The BOLB and TEXT Types.
  - Both BLOB and TEXT can hold a variable amount of data.
  - BLOB columns are treated as binary strings (byte strings). TEXT columns are treated as non-binary strings (character strings).
  - The four BLOB types are TINYBLOB, BLOB, MEDIUMBLOB, and LONGBLOB.
  - The four TEXT types are TINYTEXT, TEXT, MEDIUMTEXT, and LONGTEXT.

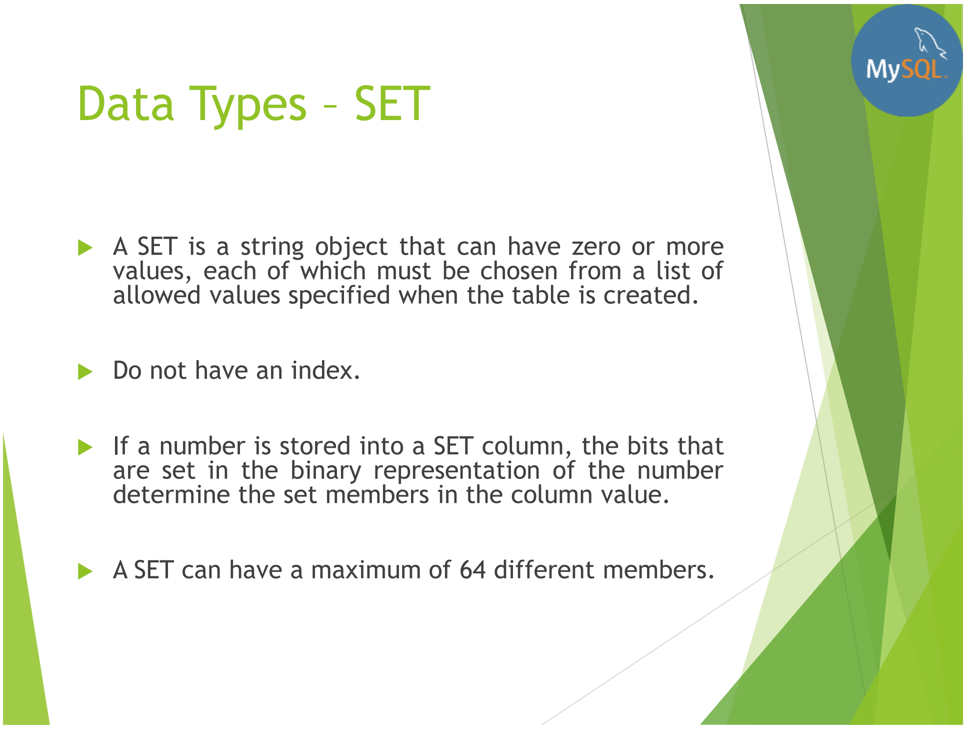
20



Data Types – ENUM

- An ENUM is a string object with a value chosen from a list of allowed values that are enumerated explicitly in the column specification at table creation time.
- Each enumeration value has an index.
- If you store a number into an ENUM column, the number is treated as the index into the possible values.
- An enumeration can have a maximum of 65,535 elements.

21



Data Types – SET

- A SET is a string object that can have zero or more values, each of which must be chosen from a list of allowed values specified when the table is created.
- Do not have an index.
- If a number is stored into a SET column, the bits that are set in the binary representation of the number determine the set members in the column value.
- A SET can have a maximum of 64 different members.

22

Data Types – String - Storage Requirement

Data Type	Storage Requirement
CHAR(M)	M bytes, 0 <= M <= 255
VARCHAR(M)	L+1 bytes, where L <= M and 0 <= M <= 255
BINARY(M)	M bytes, 0 <= M <= 255
VARBINARY(M)	L+1 bytes, where L <= M and 0 <= M <= 255
TINYBOLB, TINYTEXT	L+1 bytes, where L < 2^8
BOLB, TEXT	L+2 bytes, where L < 2^16
MEDIUMBOLB, MEDIUMTEXT	L+3 bytes, where L < 2^24
LONGBOLB, LONGTEXT	L+4 bytes, where L < 2^32
ENUM (val1, val2, …)	1 or 2 bytes. Max. 65535 values.
SET (val1, val2, …)	1, 2, 3, 4 or 8 bytes. Max. 64 members.

23

Column Attributes

- The column definition can have attribute(s) as follows:
  - [NOT NULL | NULL] [DEFAULT *default_value*] [AUTO_INCREMENT] [UNIQUE [KEY] | [PRIMARY] KEY] [COMMENT '*string*'] [*reference_definition*]
- AUTO_INCREMENT applies only to integer types. DEFAULT does not apply to the BLOB or TEXT types.
- If neither NULL nor NOT NULL is specified, the column is treated as though NULL had been specified.

24

Column Attributes – AUTO_INCREMENT

- When you insert a value of NULL (recommended) or 0 into an indexed AUTO_INCREMENT column, the column is set to the next sequence value.
- To retrieve an AUTO_INCREMENT value after inserting a row, use the LAST_INSERT_ID() SQL function.
- The attribute SERIAL is an alias for BIGINT UNSIGNED NOT NULL AUTO_INCREMENT UNIQUE.

25

Column Attributes - DEFAULT

- The DEFAULT clause specifies a default value for a column.
- With one exception, the default value must be a constant; it cannot be a function or an expression.
- BLOB and TEXT columns cannot be assigned a default value.

26

Column Attributes - UNIQUE

- A UNIQUE index creates a constraint such that all values in the index must be distinct.
- A UNIQUE index allows multiple NULL values for columns that can contain NULL.

Column Attribute – PRIMARY KEY

- A PRIMARY KEY is a unique index where all key columns must be defined as NOT NULL.
- If they are not explicitly declared as NOT NULL, MySQL declares them so implicitly.
- A table can have only one PRIMARY KEY.
- A PRIMARY KEY can be a multiple-column index. However, you cannot create a multiple-column index using the PRIMARY KEY key attribute in a column specification. Doing so only marks that single column as primary. You must use a separate PRIMARY KEY(*index_col_name*, ...) clause.

Column Attribute – COMMENT

- A comment for a column can be specified with the COMMENT option.
- The comment is displayed by the SHOW CREATE TABLE and SHOW FULL COLUMNS statements.

29

- MySQL Server parses and ignores the FOREIGN KEY and REFERENCES syntax in CREATE TABLE statements.
- The CHECK clause is parsed but ignored by all storage engines.

30

Reference Definition

- The
  - *reference_definition*: REFERENCES *tbl_name* [(*index_col_name*,...)] [MATCH FULL | MATCH PARTIAL | MATCH SIMPLE] [ON DELETE *reference_option*] [ON UPDATE *reference_option*] *reference_option*: RESTRICT | CASCADE | SET NULL | NO ACTION

31

Description of Table

- The DESCRIBE or DESC statement is used to show a description of a table.

- You can also use EXPLAIN or SHOW COLUMNS.

```
mysql> DESCRIBE species;
mysql> DESC pet;
mysql> EXPLAIN pet;
mysql> SHOW COLUMNS FROM pet;
```

32

Insert Data Into Table

- You can insert data to the tables directly from mysql> prompt.
- To insert the data directly from mysql use the **INSERT** statement.
- The format for **INSERT** is INSERT INTO <table name> (column1, column2, ...) values ('value1', 'value2', ...).

33

Insert Data Into Table

- Remember to put single quotes around a value if it is a string.
- Don't have to set the value of id because id have AUTO_INCREMENT attribute.
- Whenever you insert a new record to the table the value of id is set automatically by mysql with increasing values.

```
mysql> INSERT INTO species (name) values ('Cat');
```

34

Insert Data – Load Data

- You can also insert data to the tables by loading a file containing the data.

```
mysql> LOAD DATA LOCAL INFILE "insert.txt" INTO TABLE
species;
```

- You can specify the field separator and line separator also.

```
mysql> LOAD DATA LOCAL INFILE “C:/dat/pets.txt" INTO TABLE
 -> pets
 -> FIELDS TERMINATED BY “~”;
```

Insert Data Into Table

- MySQL expects dates in 'YYYY-MM-DD‘.
- For missing values, you can use NULL values. To represent these in your text file, use \N (backslash, capital-N).

Using MySQL in Batch Mode

- So far we have used **mysql** interactively.
- We can also run **mysql** in batch mode. To do this, put the commands you want to run in a file, then tell **mysql** to read its input from the file.
- Execute scripts from the **mysql** prompt by using the **source** or **\.** command

```
mysql> source filename;
mysql> \. filename;
```