



BOOTCAMP '20

SESSION #26

What will be covered?

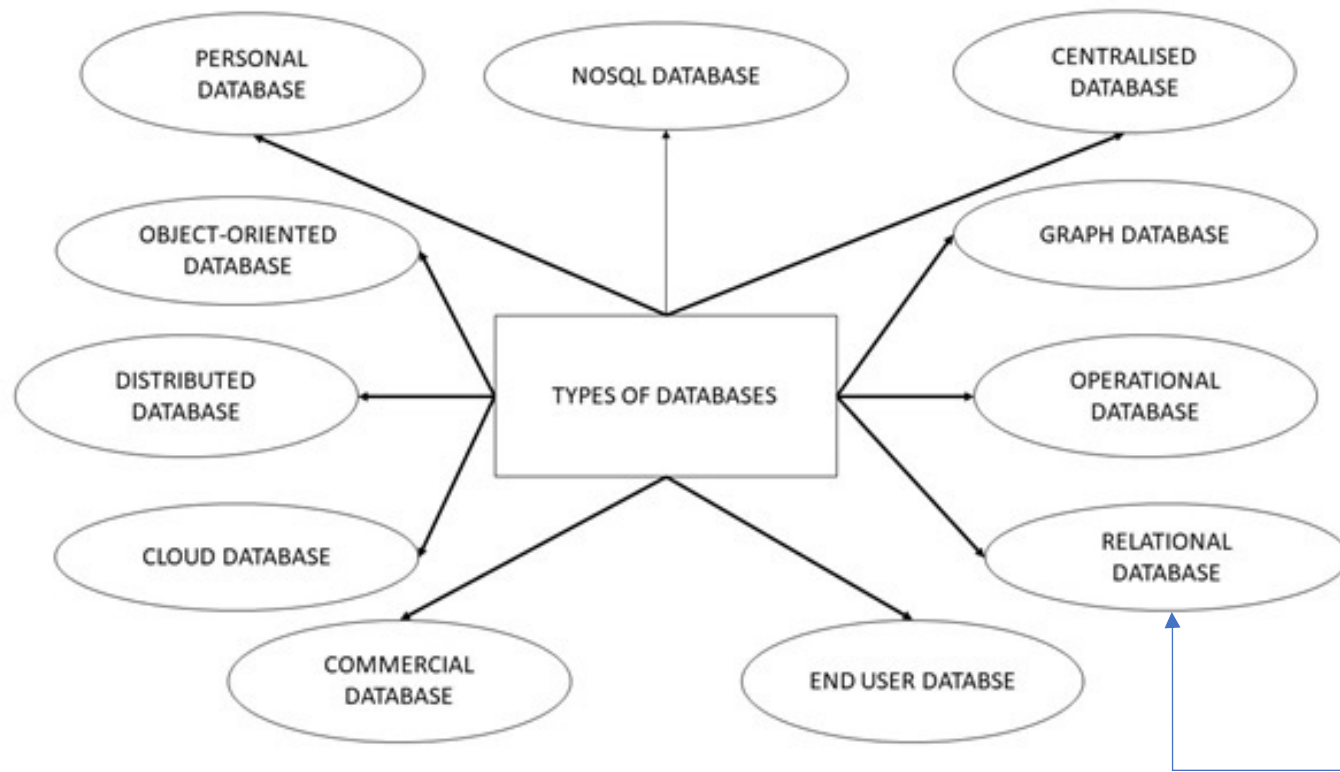
- MySQL and Databases
- Installing and Testing MySQL
- How Shall We Store it? — Datatypes
- Designing and Creating Tables
- Populating the Database
- Retrieving the Data
- Changing Data
- Deleting Data
- Aggregate Functions
- Working with Dates and Times

Intro

- MySQL is an open-source relational database management system. Its name is a combination of "My", the name of co-founder Michael Widenius's daughter, and "SQL", the abbreviation for Structured Query Language.
- Developed by: Oracle Corporation
- Stable release: 8.0.21 / 2020-07-13
- Initial release date: May 23, 1995
- License: GPLv2 or proprietary
- Original author: MySQL AB
- Written in: C, C++



MySQL and Databases

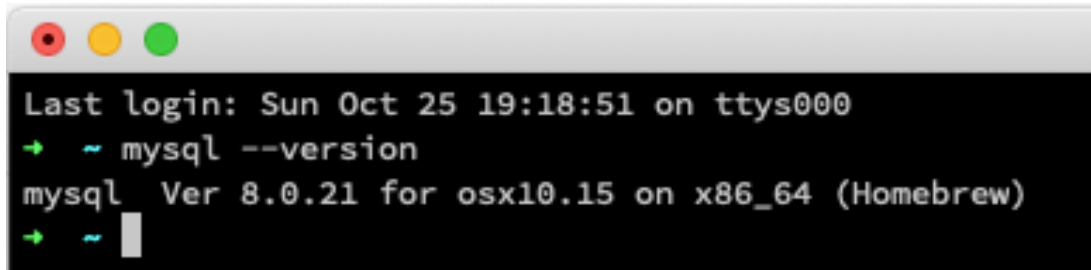


MySQL is an open-source relational database management system.



Installing and Testing MySQL

- Download link: <https://www.mysql.com/downloads/>
- Check the version from command line:

A screenshot of a macOS terminal window. The title bar shows three colored window control buttons (red, yellow, green). The terminal text shows the last login time as 'Sun Oct 25 19:18:51 on ttys000'. The user runs the command 'mysql --version', and the output is 'mysql Ver 8.0.21 for osx10.15 on x86_64 (Homebrew)'. The prompt is '~' followed by a cursor.

```
Last login: Sun Oct 25 19:18:51 on ttys000
➔ ~ mysql --version
mysql Ver 8.0.21 for osx10.15 on x86_64 (Homebrew)
➔ ~
```

- How to connect to MySQL via command line:

```
1  mysql --host=localhost --user=myname --password=password mydb
2  mysql -h localhost -u myname -ppassword mydb
```

Read more: <https://dev.mysql.com/doc/refman/8.0/en/connecting.html>

How Shall We Store it? — Datatypes

DATE TYPE	SPEC	DATA TYPE	SPEC
CHAR	String (0 - 255)	INT	Integer (-2147483648 to 2147483647)
VARCHAR	String (0 - 255)	BIGINT	Integer (-9223372036854775808 to 9223372036854775807)
TINYTEXT	String (0 - 255)	FLOAT	Decimal (precise to 23 digits)
TEXT	String (0 - 65535)	DOUBLE	Decimal (24 to 53 digits)
BLOB	String (0 - 65535)	DECIMAL	"DOUBLE" stored as string
MEDIUMTEXT	String (0 - 16777215)	DATE	YYYY-MM-DD
MEDIUMBLOB	String (0 - 16777215)	DATETIME	YYYY-MM-DD HH:MM:SS
LONGTEXT	String (0 - 4294967295)	TIMESTAMP	YYYYMMDDHHMMSS
LOBLOB	String (0 - 4294967295)	TIME	HH:MM:SS
TINYINT	Integer (-128 to 127)	ENUM	One of preset options
SMALLINT	Integer (-32768 to 32767)	SET	Selection of preset options
MEDIUMINT	Integer (-8388608 to 8388607)	BOOLEAN	TINYINT(1)

Source: <https://www.mysqltutorial.org/mysql-data-types.aspx/>

How Shall We Store it? — JSON data type

MySQL supported a native [JSON](#) data type since version 5.7.8 that allows you to store and manage JSON documents more effectively.

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

```
CREATE TABLE table_name (  
...  
json_column_name JSON,  
...  
);
```

Designing and Creating Tables

```
CREATE TABLE [IF NOT EXISTS] table_name (  
column_1_definition,  
column_2_definition,  
...,  
table_constraints )  
ENGINE=storage_engine;
```

DEFINITION:

```
column_name data_type(length) [NOT NULL] [DEFAULT value] [AUTO_INCREMENT]  
column_constraint;
```


Populating the Database

```
CREATE TABLE IF NOT EXISTS tasks (  
  task_id INT AUTO_INCREMENT PRIMARY KEY,  
  title VARCHAR(255) NOT NULL,  
  start_date DATE,  
  due_date DATE,  
  status BOOLEAN NOT NULL DEFAULT FALSE,  
  priority TINYINT NOT NULL,  
  description TEXT,  
  created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP  
) ENGINE=INNODB;
```

tasks
* task_id
title
start_date
due_date
status
priority
description
created_at

	Field	Type	Null	Key	Default	Extra
►	task_id	int(11)	NO	PRI	<small>NULL</small>	auto_increment
	title	varchar(255)	NO		<small>NULL</small>	
	start_date	date	YES		<small>NULL</small>	
	due_date	date	YES		<small>NULL</small>	
	status	tinyint(4)	NO		<small>NULL</small>	
	priority	tinyint(4)	NO		<small>NULL</small>	
	description	text	YES		<small>NULL</small>	
	created at	timestamp	YES		CURRENT_TIMESTAMP	DEFAULT GENERATED

Retrieving the Data

- **SELECT** select_list **FROM** table_name;
- **SELECT** * **FROM** table_name;
- **SELECT** select_list **FROM** table_name **WHERE** search_condition;
- **SELECT** **DISTINCT** select_list **FROM** table_name;
- **SELECT** select_list **FROM** table_name **WHERE** c1 **AND** c2 ... **AND** cN;

Retrieving the Data

- **SELECT** select_list **FROM** table_name **WHERE** c1 **OR** c2 ... **OR** cN;
- **SELECT** c1, c2,... **FROM** table_name **WHERE** (expr|col) **IN** ('v1','v2',...);
- ... expr [**NOT**] **BETWEEN** begin_expr **AND** end_expr;
- ... expression **LIKE** pattern **ESCAPE** escape_character;
 - The percentage (%) wildcard matches any string of zero or more characters.
 - The underscore (_) wildcard matches any single character.
- **SELECT** select_list **FROM** table_name **LIMIT** [offset,] row_count;
- **SELECT** select_list **FROM** table_name **ORDER BY** col1 [**ASC**|**DESC**], col2 [**ASC**|**DESC**], ...;

Changing Data

- **UPDATE** [**LOW_PRIORITY**] [**IGNORE**] table_name **SET** column_name1 = expr1, column_name2 = expr2, ... [**WHERE** condition];

Example:

UPDATE employees

SET email = 'name.surname@domain.com'

WHERE employeeNumber = 10;

Deleting Data

- **DELETE FROM** table_name **WHERE** condition;
- **DELETE FROM** table_name **LIMIT** row_count;
- **DELETE FROM** table_name **ORDER BY** c1, c2, ... **LIMIT** row_count;

Example:

```
DELETE FROM customers  
WHERE country = 'France'  
ORDER BY creditLimit LIMIT 5;
```

Aggregate Functions

Aggregate function	Description
AVG()	Return the average of non-NULL values.
BIT_AND()	Return bitwise AND.
BIT_OR()	Return bitwise OR.
BIT_XOR()	Return bitwise XOR.
COUNT()	Return the number of rows in a group, including rows with NULL values.
GROUP_CONCAT()	Return a concatenated string.
JSON_ARRAYAGG()	Return result set as a single JSON array.
JSON_OBJECTAGG()	Return result set as a single JSON object.

MAX()	Return the highest value (maximum) in a set of non-NULL values.
MIN()	Return the lowest value (minimum) in a set of non-NULL values.
STDEV()	Return the population standard deviation.
STDDEV_POP()	Return the population standard deviation.
STDDEV_SAMP()	Return the sample standard deviation.
SUM()	Return the summation of all non-NULL values a set.
VAR_POP()	Return the population standard variance.
VARP_SAM()	Return the sample variance.
VARIANCE()	Return the population standard variance.

Aggregate Functions

- AVG()
 - SELECT AVG(buyprice) 'Average Price' FROM products;
- COUNT()
 - SELECT COUNT(*) FROM count_demos;
- MAX()
 - SELECT MAX(amount) FROM payments;
- MIN()
 - SELECT MAX(amount) FROM payments;
- SUM()
 - SELECT SUM(quantityOrdered) SalesQuantity FROM orderdetails;

Working with Dates and Times

- **SELECT NOW();**
- **SELECT DATE(NOW());**
- **SELECT CURDATE();**
- **SELECT DATE_FORMAT(CURDATE(), '%m/%d/%Y') today;**
- **SELECT DATEDIFF('2020-05-01','2020-10-25') days;**
- **SELECT WEEKDAY('2000-12-31') weekday,**
WEEK('2000-12-31') week,
WEEKOFYEAR('2000-12-31') weekofyear;

Working with Dates and Times

- **SET** time_zone = '+00:00';

- **SELECT TIME(NOW());**

- **SELECT**

- **HOUR(@dt),**
- **MINUTE(@dt),**
- **SECOND(@dt),**
- **DAY(@dt),**
- **WEEK(@dt),**
- **MONTH(@dt),**
- **QUARTER(@dt),**
- **YEAR(@dt);**

@dt = NOW()

Declare variable:
SET @dt = NOW();

QUESTIONS

