Master the Basics of Querying Tables in MySQL

Richard Cornelius Suwandi

February 11, 2022

The Chinese University of Hong Kong, Shenzhen

Powered by MySQL®

Outline

Introduction to SQL

What is SQL?

- SQL stands for Structured Query Language
- SQL is a computer language for storing, manipulating, and retrieving data stored in a relational database
- SQL is the standard language for Relational Database Management Systems (RDMS) like MySQL, MS Access, Oracle, Postgres, etc.

Why SQL?

- SQL is easy to learn
- SQL is one of the most used languages for data analysis
- SQL can easily deal with big data
- SQL is one of the most in-demand skills for data-related jobs

Getting Started with MySQL

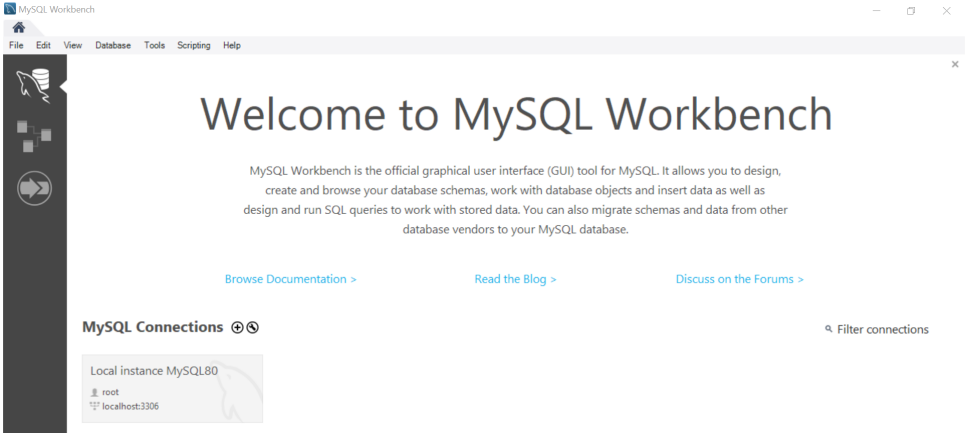
MySQL

- MySQL is a free and open-source RDMS developed by Oracle
- MySQL can run on a number of different operating systems such as Windows, Linux, Mac OS, etc.

Note

- In this workshop, we are going to use **MySQL Workbench** as our visual database designing and modeling access tool for MySQL
- Install MySQL Workbench here: *https://dev.mysql.com/downloads/workbench/*

MySQL Workbench (Demo)



Databases and Tables

Databases

- Database is a collection of tables that stores a specific set of structured data
- We can use the following SQL command to create a database:

Command
CREATE DATABASE [IF NOT EXISTS] database_name;

- *IF NOT EXISTS* is an optional parameter that can be used to avoid conflict with an existing database
- We can use the *SHOW DATABASES* command to see the list of existing databases

Tables

- Table is a collection of data organized in a tabular form (rows and columns)
- We can use the following SQL command to create a table:

Command

```
CREATE TABLE [IF NOT EXISTS] table_name (
    column_name data_type
);
```

- The *column_name* parameter specifies the name of the column
- The *data_type* parameter specifies the type of data that the column can hold

Numeric Data Types

Numeric data types are used to store numeric values

- *INT*: whole numbers (integers)
- *DOUBLE*: floating-point/rational numbers
- *BIT*: binary values (0 or 1)

Learn more

https://dev.mysql.com/doc/refman/8.0/en/numeric-types.html

String Data Types

String data types are used to store text values

- *CHAR*: fixed-length strings (up to 255 characters)
- *TEXT*: fixed-length strings (up to 65535 characters)
- *VARCHAR*: variable-length strings (up to 65535 characters)

Learn more

https://dev.mysql.com/doc/refman/8.0/en/string-types.html

Date and Time Data Types

Date and time data types are used to store date and time values

- *DATE*: date in the format YYYY-MM-DD
- *TIME*: time in the format HH:MM:SS
- *DATETIME*: date and time in the format YYYY-MM-DD HH:MM:SS

Learn more

https://dev.mysql.com/doc/refman/8.0/en/date-and-time-types.html

Inserting Data

- We can use the *INSERT* statement to insert one or more rows into a table:

Command

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

- The order of the values that are inserted must be in the same order as the columns in the table
- It is also possible to only insert data in specific columns:

Command

```
INSERT INTO table_name (column1, column2, …)
VALUES (value1, value2, …);
```

Exercise 1

- Create a table named *`students`* that has the following columns: *`student_ID`*, *`student_name`*, *`major`*
- Insert the following data to the table:

student_ID	student_name	major
101	Alice	Accounting
102	Bob	Bioinformatics
103	Chris	Computer Science
104	Dave	Data Science
105	Ellen	Economics

- Use *`SELECT * FROM students`* to display the created table

Working with Real Data

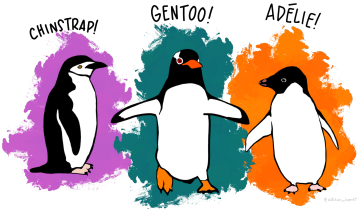
Importing Data (Demo)

- In practice, we often need to work with data from external files such CSV or JSON files
- We can use the Table Data Import Wizard in MySQL Workbench to import a table from a CSV or JSON file

Note

- In this workshop, we are going to use the **Palmer Penguins** dataset
- The dataset can be downloaded here: *https://bit.ly/sql-workshop-ppitsz*

About the Dataset



The dataset contains observations on 344 penguins with 3 different species from 3 islands in the Palmer Archipelago, Antarctica

Learn more

https://allisonhorst.github.io/palmerpenguins/

SQL Queries

SELECT

- *SELECT* is used to select data from a table

Command

```
SELECT column1, column2, …
FROM table_name;
```

- To select all the columns from the table, we use the following command:

Command

```
SELECT * FROM table_name;
```

- We can add the *DISTINCT* parameter to return only distinct (unique) values

WHERE

- *WHERE* is used to filter data from a table based on the specified condition

Command

```
SELECT column1, column2, …
FROM table_name
WHERE condition;
```

- The following operators can be used to specify a condition:
  - **Comparison operators**: =, !=, >, <, >=, <=
  - *BETWEEN*: values between a certain range
  - *IN*: multiple possible values

AND, OR, NOT

- The *AND* and *OR* operators are used to filter data based on more than one conditions

Command

```
SELECT column1, column2, …
FROM table_name
WHERE condition1 AND/OR condition2 AND/OR …;
```

- The *NOT* operator is used to display data when the condition is not true

Command

```
SELECT column1, column2, …
FROM table_name
WHERE NOT condition;
```

Exercise 2

- Select all data from the *`penguins`* table that satisfies the following conditions:
  1. The penguin is from Biscoe Island or Dream Island
  2. The penguin's body mass is between 3000 and 3500 grams
  3. The penguin's sex is not male

Aggregate Functions

COUNT

- The *COUNT()* function returns the number of rows that matches a specified criterion

Command

```
SELECT COUNT(column_name)
FROM table_name
WHERE condition;
```

- *NULL* values are ignored in the *COUNT()* function

SUM

- The *SUM()* function returns the total sum of a numeric column

Command

```
SELECT SUM(column_name)
FROM table_name
WHERE condition;
```

- *NULL* values are ignored in the *SUM()* function

AVG

- The *AVG()* function returns the average value of a numeric column

Command

```
SELECT AVG(column_name)
FROM table_name
WHERE condition;
```

- *NULL* values are ignored in the *AVG()* function

MIN(), MAX()

- The *MIN()* function returns the smallest value of the selected column
- The *MAX()* function returns the largest value of the selected column

Command

```
SELECT MIN/MAX(column_name)
FROM table_name
WHERE condition;
```

GROUP BY

- *GROUP BY* is used to group data that have the same values

Command

```
SELECT column1, column2, …
FROM table_name
GROUP BY column_name;
```

- *GROUP BY* is often used with aggregate functions like *COUNT( )*, *SUM( )*, *AVG( )*, etc.

HAVING

- *HAVING* is used to filter data from the groups based on the specified condition

Command

```
SELECT column1, column2, …
FROM table_name
GROUP BY column_name
HAVING condition;
```

- *HAVING* applies a filter condition to each group of rows, while *WHERE* applies the filter condition to each individual row

ORDER BY

- *ORDER BY* is used to sort the result-set in ascending or descending order

Command

```
SELECT column1, column2, …
FROM table_name
ORDER BY column_name;
```

- By default, *ORDER BY* sorts the data in ascending order
- To sort the data in descending order, we can add the *DESC* parameter

Exercise 3

- Write an SQL query to perform the following tasks:
  1. Display the number of penguins and the average bill length of the penguins in each island
  2. Filter the results to only show the islands where the average bill length of the penguins exceeds 40 millimeters
  3. Sort the results in descending order based on the number of the penguins in each island

Q&A

Thank you! Any questions?