

MySQL



1.Thông tin cơ bản

1.1 Cơ sở dữ liệu là gì ?

- Một tập hợp dữ liệu được tổ chức có thể dễ dàng truy cập, khôi phục, quản lý và cập nhật được gọi là cơ sở dữ liệu. Nó giống như một thư viện lưu trữ nhiều loại sách thuộc nhiều thể loại khác nhau.
- Dữ liệu có thể được sắp xếp trong bảng cơ sở dữ liệu bằng cách sử dụng hàng và cột. Việc định vị và truy xuất dữ liệu đã được lập chỉ mục dễ dàng hơn nhiều.
- Cơ sở dữ liệu hiện đại được điều hành bởi DBMS. Dữ liệu trong cơ sở dữ liệu có thể được xử lý bằng SQL và cấu trúc cơ sở dữ liệu hình trụ là phổ biến nhất.
- Cơ sở dữ liệu thường được sử dụng cho ngân hàng, lưu lượng hàng không, trường đại học, nguồn nhân lực, doanh nghiệp thương mại điện tử, công ty viễn thông, giao dịch thẻ tín dụng, v.v. Ngoài ra, chúng có thể được sử dụng để quản lý số lượng lớn các trang web trên Internet.

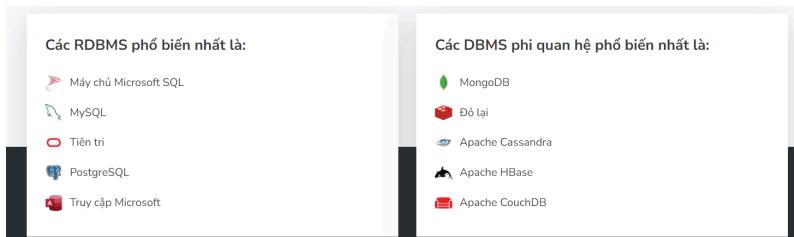
1.2 DBMS là gì?

- Hãy bắt đầu với định nghĩa. DBMS là viết tắt của hệ thống quản lý cơ sở dữ liệu, thường được định nghĩa là hệ thống phần mềm cho phép người dùng xác định, tạo, duy trì và kiểm soát quyền truy cập vào cơ sở dữ liệu.
- DBMS, cùng với các công cụ cơ sở dữ liệu chuyên dụng, tạo thành lớp giao diện giữa người dùng và dữ liệu cần được lưu trữ, truy xuất và xử lý.
- Các mô hình cơ sở dữ liệu chính, các thành phần của chúng, các trường hợp sử dụng và các ví dụ nổi tiếng nhất về hệ thống quản lý cơ sở dữ liệu đều được trình bày bên dưới. Cho dù bạn là người mới bắt đầu hay là một nhà phát triển phần mềm dày dạn kinh nghiệm, thông tin này có thể hữu ích.

1.3 RDBMS là gì?

- RDBMS là viết tắt của [hệ thống quản lý cơ sở dữ liệu](#) quan hệ —một hệ thống phần mềm cho phép bạn xác định, tạo, duy trì và kiểm soát quyền truy cập vào cơ sở dữ liệu quan hệ. Đây là phần cơ bản của lớp giao diện giúp bạn lưu trữ và làm việc với dữ liệu.
- Cơ sở dữ liệu quan hệ là loại cơ sở dữ liệu phổ biến nhất lưu trữ và cung cấp quyền truy cập vào các điểm dữ liệu có liên quan với nhau. Trong cơ sở dữ liệu quan hệ, dữ liệu được sắp xếp và lưu trữ trong các bảng bao gồm các cột và hàng. Và tất nhiên, bạn có ngôn ngữ truy vấn có cấu trúc trực quan và linh hoạt để thao tác cơ sở dữ liệu.

1.4 DBMS so với RDBMS



DBMS so với RDBMS: so sánh chi tiết

Tham số	DBMS phi quan hệ	Hệ quản trị cơ sở dữ liệu quan hệ (RDBMS)
Cấu trúc cơ sở dữ liệu	Hệ thống phân cấp (mỗi quan hệ cha-con), mạng (mỗi quan hệ nhiều-nhiều) hoặc một tập hợp các đối tượng	Định dạng bảng (hàng và cột)
Lưu trữ dữ liệu	Dữ liệu được lưu trữ trong các tập tin	Dữ liệu được lưu trữ trong các bảng có liên quan với nhau (xem ở trên)
Truy cập dữ liệu	Một người dùng tại một thời điểm	Truy cập đồng thời bởi nhiều người dùng với nhiều vai trò người dùng khác nhau
Mỗi quan hệ dữ liệu	Không có	Mỗi quan hệ giữa các bảng thông qua khóa ngoại

Tham số	DBMS phi quan hệ	Hệ quản trị cơ sở dữ liệu quan hệ (RDBMS)
Ngôn ngữ thống nhất	Không có	SQL (Ngôn ngữ truy vấn có cấu trúc)
Sự dư thừa dữ liệu	Chung	Đã loại bỏ bằng khóa và chỉ mục
Tính chất ACID: Nguyên tử, Đồng thời, Cô lập, Độ bền	Không tuân thủ	Tuân thủ đầy đủ
Kiến trúc máy khách-máy chủ	Không	Đúng
Cơ sở dữ liệu phân tán	Không	Đúng
Xử lý khôi phục lượng dữ liệu lớn	Không	Đúng
Chuẩn hóa	Không	Đúng
Yêu cầu về phần cứng và phần mềm	Thấp	Cao hơn

Các ưu điểm và nhược điểm của DBMS và RDBMS

Ưu điểm của DBMS phi quan hệ:

- Chi phí nói chung là thấp
- Tốc độ lựa chọn và chèn dữ liệu cao
- Bảo trì khá đơn giản
- Dễ dàng mở rộng theo chiều ngang
- Xử lý thông qua các thiết bị có tài nguyên thấp

Nhược điểm của DBMS không quan hệ:

-  Nói chung khả năng sử dụng kém
-  Không có khả năng xử lý các hoạt động phức tạp với dữ liệu
-  Lưu trữ dữ liệu không nhất quán
-  Không có biện pháp an ninh
-  Không có ngôn ngữ chuẩn hóa

Ưu điểm của RDBMS:

-  Tính toàn vẹn dữ liệu nhất quán
-  Sao lưu và phục hồi đáng tin cậy
-  Bảo mật dữ liệu nhạy cảm được duy trì tốt
-  Cộng đồng lớn
-  Tích hợp nhanh chóng với các sản phẩm phần mềm thương mại
-  [IDE cơ sở dữ liệu](#) đa chức năng cho mọi tác vụ

Nhược điểm của RDBMS:

-  Chi phí phần cứng và phần mềm tương đối cao
-  Sự phức tạp có thể có của quản lý
-  Thực hiện chậm các truy vấn phức tạp trong cơ sở dữ liệu được chuẩn hóa quá mức
-  Nhu cầu về chuyên môn nghiệp vụ để thực hiện các nhiệm vụ phát triển và quản lý thường xuyên

1.5 MySQL là gì?

Định nghĩa

- MySQL là một hệ thống quản lý cơ sở dữ liệu đa nền tảng mã nguồn mở dựa trên các truy vấn SQL. Ban đầu nó được phát triển bởi công ty MySQL AB của Thụy Điển. Ngày nay nó là chi nhánh của Oracle Corporation.
- Hệ thống này có thể được coi là RDBMS phổ biến và dễ nhận biết nhất trên toàn thế giới. Nhiều gã khổng lồ như Facebook, YouTube, Uber, Google và vô số các tập đoàn khác thích MySQL để lưu trữ và quản lý dữ liệu của họ. Và họ có tất cả lý do để gắn bó với RDBMS này.

Các tính năng đáng chú ý nhất của MySQL

- Kiến trúc máy khách/máy chủ
- Hỗ trợ ODBC
- Truy vấn và lệnh SQL
- Sao chép
- Giao dịch
- Ràng buộc khóa ngoài
- Tùy chỉnh dữ liệu

1.6 SQL so với NoSQL

Cơ sở dữ liệu SQL là gì?

- SQL là viết tắt của Structured Query Language, thực chất là một ngôn ngữ được sử dụng bởi [các hệ thống quản lý cơ sở dữ liệu quan hệ](#). Ban đầu nó được gọi là SEQUEL (Structured English Query Language) và sau đó được đổi tên thành SQL bằng cách bỏ nguyên âm để tránh xung đột về nhãn hiệu. SQL là một ngôn ngữ khai báo và bao gồm các lệnh có thể được chia thành bốn loại:

Ngôn ngữ truy vấn dữ liệu (DQL), Ngôn ngữ định nghĩa dữ liệu (DDL), Ngôn ngữ kiểm soát dữ liệu (DCL) và Ngôn ngữ thao tác dữ liệu (DML). Một số chuyên gia tách loại thứ năm—TCL—Ngôn ngữ kiểm soát giao dịch.

- SQL được sử dụng cho cơ sở dữ liệu quan hệ: SQL Server, MySQL, MariaDB, PostgreSQL và Oracle sử dụng SQL (với một số biến thể nhất định) để quản lý dữ liệu. Nói một cách đơn giản, **cơ sở dữ liệu SQL là một tập hợp dữ liệu có cấu trúc dựa trên bảng**. Bảng cơ sở dữ liệu SQL chứa một tập hợp các hàng, còn được gọi là bản ghi và các cột còn được gọi là thuộc tính. Mỗi cột lưu trữ một loại dữ liệu nhất định, ví dụ: văn bản, số, ngày, v.v.

NoSQL là gì?

- Cơ sở dữ liệu NoSQL, còn được gọi là cơ sở dữ liệu phi quan hệ, là cơ sở dữ liệu phi bảng lưu trữ dữ liệu theo cách khá khác so với cơ sở dữ liệu quan hệ. Cơ sở dữ liệu NoSQL không dựa vào các cấu trúc cứng nhắc và sử dụng các mô hình dữ liệu linh hoạt hơn. Thuật ngữ 'NoSQL' có nghĩa là 'không chỉ SQL' và được sử dụng để chỉ bất kỳ cơ sở dữ liệu phi quan hệ nào.

Bảng so sánh SQL và NoSQL

	Cơ sở dữ liệu SQL	Cơ sở dữ liệu SQL
Mô hình lưu trữ dữ liệu	Bảng hai chiều	Tùy thuộc vào loại cơ sở dữ liệu: Kho lưu trữ khóa-giá trị liên kết mỗi giá trị dữ liệu với một khóa duy nhất Cơ sở dữ liệu tài liệu lưu trữ dữ liệu dưới dạng tập hợp các tài liệu Cơ sở dữ liệu đồ thị lưu trữ dữ liệu dưới dạng các nút và cạnh Cơ sở dữ liệu họ cột lưu trữ dữ liệu trong các cột nhưng các cột được chia thành các nhóm gọi là họ cột
Kết cấu	Được xác định trước và cứng nhắc	Linh hoạt và năng động
Giao dịch cơ sở dữ liệu	ACID (Tính nguyên tử, Tính nhất quán, Tính cô lập và Tính bền vững) là một tiêu chuẩn cho cơ sở dữ liệu SQL	Cơ sở dữ liệu NoSQL thường tuân theo mô hình BASE (Cơ bản có sẵn, Trạng thái mềm, Tính nhất quán cuối cùng)

	Cơ sở dữ liệu SQL	Cơ sở dữ liệu SQL
Bản đồ	Cơ sở dữ liệu SQL yêu cầu ORM (ánh xạ quan hệ đối tượng)	Cơ sở dữ liệu NoSQL thường không yêu cầu ORM
Ngôn ngữ truy vấn	SQL (Ngôn ngữ truy vấn có cấu trúc)	Không có ngôn ngữ truy vấn khai báo
Tỷ lệ	Thẳng đứng	Nằm ngang
Các tính năng chính	Hỗ trợ đa nền tảng, bảo mật nhiều cấp độ	Khả năng mở rộng, linh hoạt, hiệu suất cao
Lựa chọn tốt nhất	Cơ sở dữ liệu SQL là lựa chọn tốt nhất nếu bạn cần hỗ trợ cho các truy vấn động	Cơ sở dữ liệu NoSQL là lựa chọn tốt nếu bạn có thể cần mở rộng quy mô do các yêu cầu thay đổi

Ưu điểm và nhược điểm của NoSQL và SQL

Ưu điểm của SQL

- Sự đơn giản
- Bảo mật dữ liệu
- Khả năng tiếp cận
- Độ chính xác và toàn vẹn của dữ liệu

Nhược điểm của SQL

- Trị giá
- Các vấn đề về hiệu suất
- Khả năng mở rộng kém

Ưu điểm của NoSQL

- Hiệu suất
- Khả năng mở rộng
- Tính linh hoạt

- Hiệu quả về chi phí
- **Nhược điểm của NoSQL**
- Sự thiếu chín chắn và thiếu sự hỗ trợ
- Các vấn đề về tính nhất quán
- Thách thức phân tích dữ liệu

1.7 Hướng dẫn về MariaDB

Định nghĩa

- MariaDB là một nhánh phần mềm của MySQL, đây là lý do chính khiến chúng có điểm tương đồng. Cốt lõi của nó là mã nguồn MySQL. Tuy nhiên, chúng không giống nhau. MariaDB đã phát triển đáng kể kể từ năm 2009 khi một số người sáng tạo MySQL tạo ra một sản phẩm riêng biệt từ cùng một nguồn gốc.

MariaDB so với MySQL

- MariaDB là một sự thay thế thường xuyên cho MySQL. Mỗi quan hệ chặt chẽ giữa hai hệ thống này giúp việc di chuyển từ MySQL sang MariaDB trở nên dễ dàng và đơn giản. Bạn có thể nhập và xuất dữ liệu từ một RDBMS này sang một RDBMS khác mà không cần thay đổi dữ liệu đó. Tất cả các công cụ và ứng dụng CMS thông thường đều hoạt động trên MariaDB ngay khi cài đặt.

Ưu điểm của MariaDB

- MariaDB có nhiều công cụ lưu trữ hơn, bao gồm PBXT, XtraDB, Maria và FederatedX.
- Nhóm kết nối của nó lớn hơn (lên tới hơn 200.000 kết nối).
- Việc sao chép có thể nhanh hơn gấp hai lần trong MariaDB.
- Nhìn chung, nó nhanh hơn và dễ sử dụng hơn.
- MariaDB cho phép tạo các bảng có phiên bản.

- Nó cung cấp các tính năng tương thích với Oracle giúp việc di chuyển từ Cơ sở dữ liệu Oracle trở nên dễ dàng.
- Máy chủ MariaDB hỗ trợ định dạng lưu trữ theo cột.
- MariaDB cung cấp giải pháp cơ sở dữ liệu SQL phân tán—MariaDB Xpand.

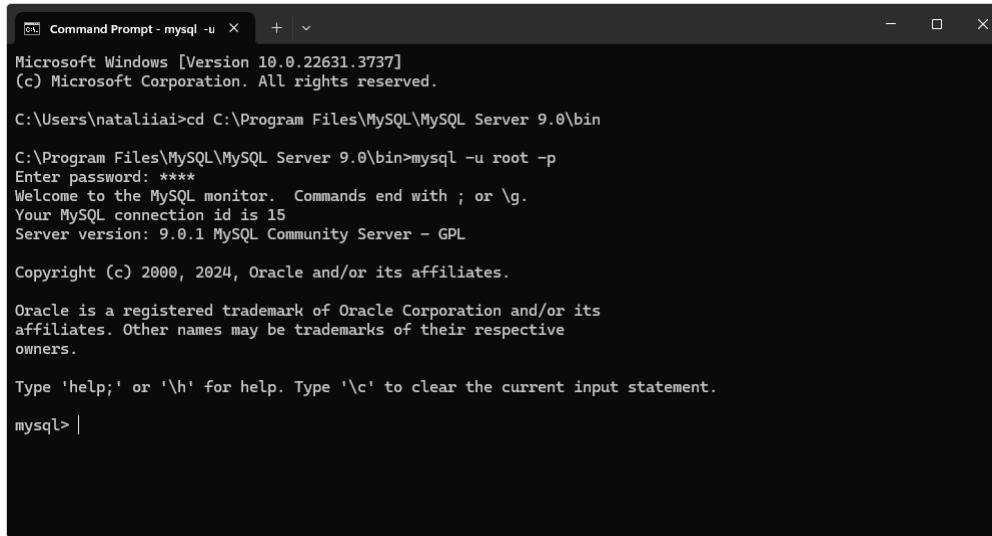
Các tính năng đáng chú ý nhất của MariaDB

- Khả năng tương thích
- Thực hiện song song các truy vấn
- Nhóm luồng
- Lướt xem cơ sở dữ liệu
- Cột ảo
- ColumnStore
- Lưu trữ Flash

1.8 Hướng dẫn sử dụng MySQL CLI

Cách sử dụng MySQL Command Line Client

1. Mở Dấu nhắc lệnh.
2. Điều hướng đến thư mục bin. Ví dụ: `cd C:\Program Files\MySQL\MySQL Server 9.0\bin`
3. Chạy lệnh. `mysql -u root -p`
4. Nhập mật khẩu.



```
Command Prompt - mysql -u
Microsoft Windows [Version 10.0.22631.3737]
(c) Microsoft Corporation. All rights reserved.

C:\Users\nataliai>cd C:\Program Files\MySQL\MySQL Server 9.0\bin
C:\Program Files\MySQL\MySQL Server 9.0\bin>mysql -u root -p
Enter password: ****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 15
Server version: 9.0.1 MySQL Community Server - GPL

Copyright (c) 2000, 2024, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> |
```

Cách bắt đầu quản lý cơ sở dữ liệu MySQL từ dòng lệnh

- Để bắt đầu quản lý và quản trị cơ sở dữ liệu MySQL từ dòng lệnh, bạn sẽ cần làm quen với cú pháp dòng lệnh. Hãy cùng xem xét chi tiết một loạt các lệnh đơn giản.

Cách tạo người dùng từ dòng lệnh

- Trước hết, chúng ta cần tạo một người dùng. Để thực hiện, chúng ta chạy lệnh sau:

```
CREATE USER 'username' IDENTIFIED BY 'password';
```

- Đừng quên thay thế chỗ giữ tên người dùng và mật khẩu bằng tên người dùng và mật khẩu mà bạn chọn.
- Hãy nhớ rằng, chỉ tạo một người dùng là không đủ, bạn cần cấp một số quyền nhất định cho người dùng này. Để làm được điều này, hãy chạy truy vấn MySQL:

```
GRANT SELECT ON *.* TO 'username';
```

- Lệnh này chỉ cấp quyền SELECT cho người dùng được chỉ định. Trong trường hợp bạn muốn cấp cho người dùng tất cả các quyền trên tất cả các cơ sở dữ liệu, hãy chạy lệnh sau:

```
GRANT ALL PRIVILEGES ON *.* TO 'username';
```



```
MySQL 9.0 Command Line Cli  X  +  ▾
mysql> CREATE USER 'JordanS' IDENTIFIED BY 'pass1235';
Query OK, 0 rows affected (0.01 sec)

mysql> GRANT SELECT ON *.* TO 'JordanS';
Query OK, 0 rows affected (0.02 sec)

mysql> GRANT ALL PRIVILEGES ON *.* TO 'JordanS';
Query OK, 0 rows affected (0.02 sec)

mysql> |
```

Cách tạo cơ sở dữ liệu từ dòng lệnh

- Để tạo cơ sở dữ liệu, hãy sử dụng lệnh sau. Thay thế chỗ giữ chỗ bằng tên cơ sở dữ liệu cần thiết.

```
CREATE DATABASE dbname;
```

- Để bắt đầu làm việc với cơ sở dữ liệu mới tạo, hãy thực hiện truy vấn:

```
USE dbname;
```



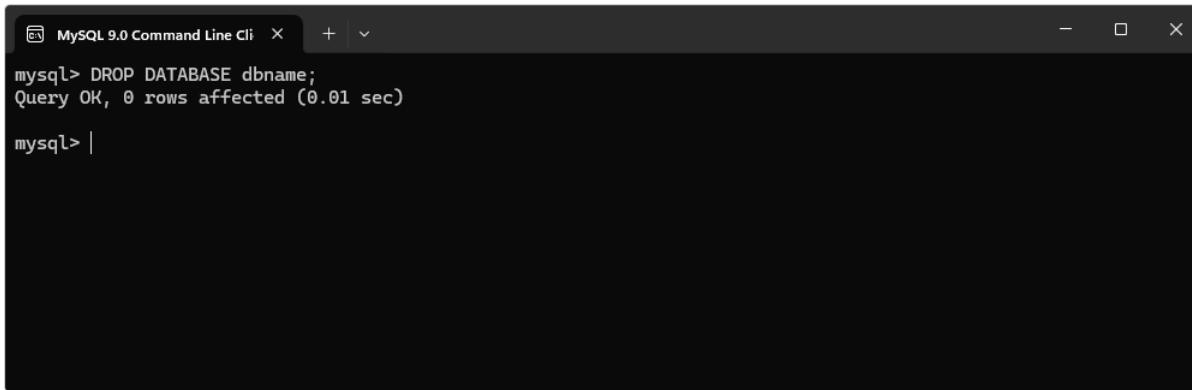
```
MySQL 9.0 Command Line Cli + x
mysql> CREATE DATABASE dbname;
Query OK, 1 row affected (0.01 sec)

mysql> USE dbname;
Database changed
mysql> |
```

Cách xóa cơ sở dữ liệu MySQL từ dòng lệnh

- Để xóa cơ sở dữ liệu, hãy chạy lệnh đơn giản sau. Hãy nhớ rằng bạn sẽ không thể khôi phục lại lệnh đã xóa, vì vậy hãy thực hiện thao tác này một cách thận trọng.

```
DROP DATABASE dbname;
```



```
MySQL 9.0 Command Line Cli + x
mysql> DROP DATABASE dbname;
Query OK, 0 rows affected (0.01 sec)

mysql> |
```

Cách xóa tài khoản người dùng MySQL

Trong blog của chúng tôi, chúng tôi đã nói về [cách tạo người dùng mới trong MySQL](#) . Để [xóa người dùng trong MySQL](#) , hãy thực hiện truy vấn:

```
DROP USER 'username';
```

A screenshot of the MySQL 9.0 Command Line Client window. The title bar says "MySQL 9.0 Command Line Cli". The main area shows the following text:

```
mysql> DROP USER 'JordanS';
Query OK, 0 rows affected (0.01 sec)

mysql> |
```

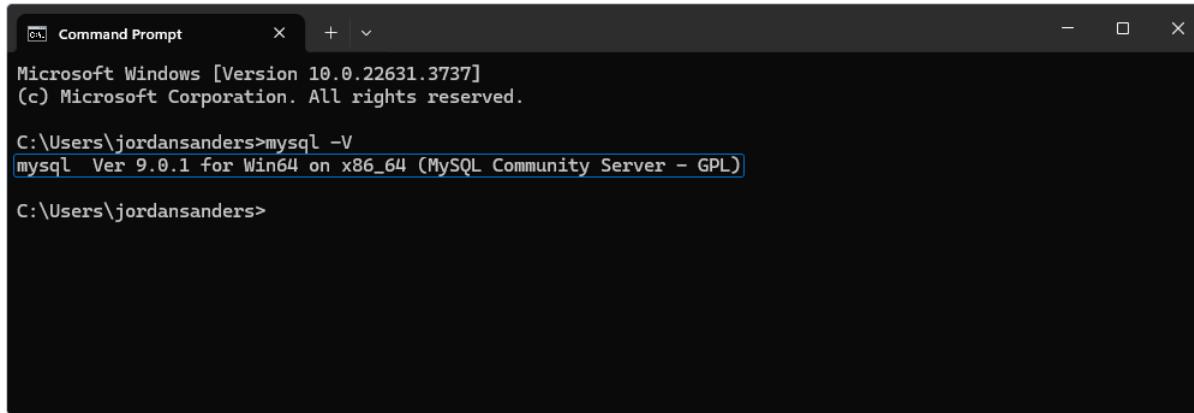
The window has a dark theme with white text.

1.9 Kiểm tra phiên bản MySQL

Cách kiểm tra phiên bản MySQL trong Windows Terminal

- Một trong những cách dễ nhất để kiểm tra phiên bản máy chủ MySQL cục bộ của bạn từ dòng lệnh trên [Windows](#) là sử dụng lệnh sau, lệnh này không chỉ hoạt động trên Windows mà còn trên [macOS](#) , [Linux](#) , [Ubuntu](#) và [Debian](#) :

```
mysql -V
```



```
Command Prompt
Microsoft Windows [Version 10.0.22631.3737]
(c) Microsoft Corporation. All rights reserved.

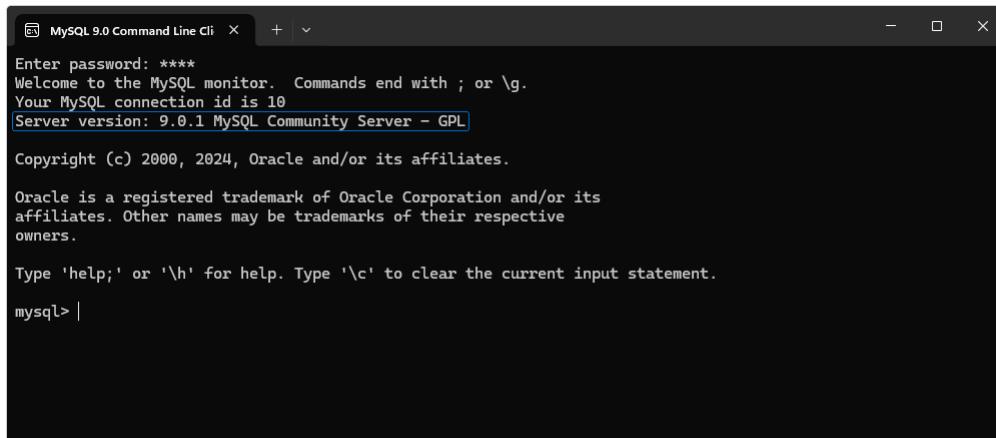
C:\Users\jordansanders>mysql -v
mysql Ver 9.0.1 for Win64 on x86_64 (MySQL Community Server - GPL)

C:\Users\jordansanders>
```

Cách tìm phiên bản MySQL từ máy khách dòng lệnh

Cách tìm phiên bản MySQL từ máy khách dòng lệnh

Chỉ cần mở ứng dụng MySQL và thông tin về phiên bản MySQL hiện tại sẽ có ngay lập tức.



```
MySQL 9.0 Command Line Cli
Enter password: ****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 10
Server version: 9.0.1 MySQL Community Server - GPL

Copyright (c) 2000, 2024, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

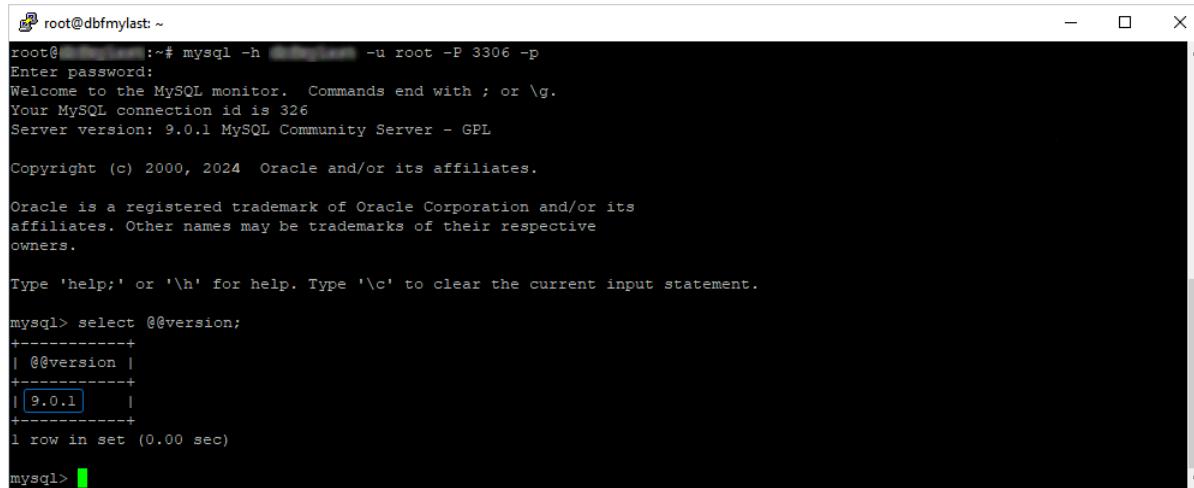
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> |
```

Ngoài ra còn có nhiều cách khác để tìm ra phiên bản máy chủ MySQL từ dòng lệnh:

- Sử dụng SSH

```
select @@version;
```



```
root@dbfmylast:~# mysql -h [REDACTED] -u root -P 3306 -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 326
Server version: 9.0.1 MySQL Community Server - GPL

Copyright (c) 2000, 2024 Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> select @@version;
+-----+
| @@version |
+-----+
| 9.0.1 |
+-----+
1 row in set (0.00 sec)

mysql> |
```

- **Với sự trợ giúp của truy vấn SHOW VARIABLES LIKE**

```
SHOW VARIABLES LIKE 'version';
```



```
MySQL 9.0 Command Line Cli + | ▾
mysql> SHOW VARIABLES LIKE 'version';
+-----+-----+
| Variable_name | Value |
+-----+-----+
| version | 9.0.1 |
+-----+-----+
1 row in set (0.03 sec)

mysql> |
```

- **Sử dụng lệnh SELECT VERSION**

```
SELECT VERSION();
```

Đừng quên sử dụng dấu chấm phẩy để phân cách câu lệnh khi làm việc với MySQL Client.

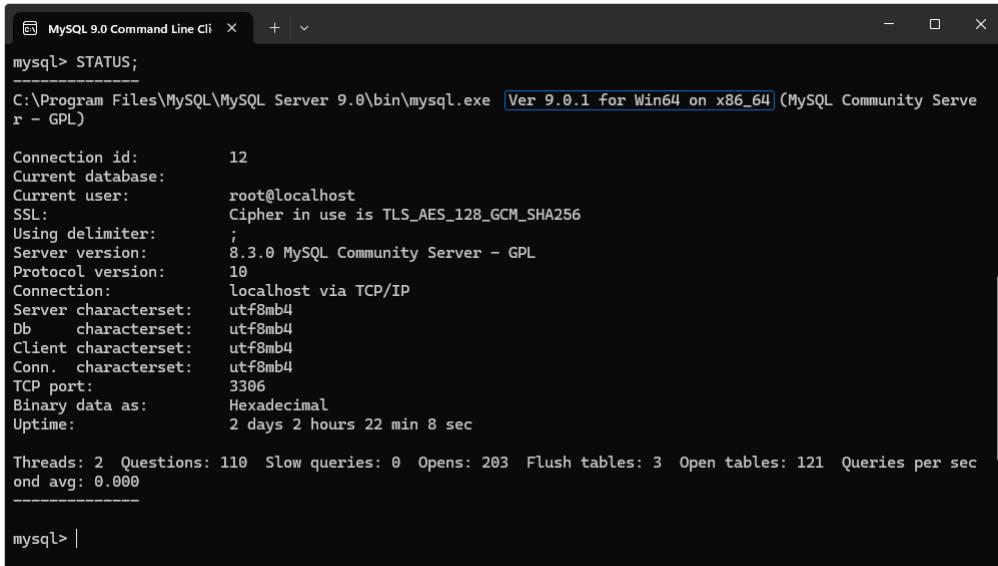


```
MySQL 9.0 Command Line Cli + - x
mysql> SELECT VERSION();
+-----+
| VERSION() |
+-----+
| 9.0.1 |
+-----+
1 row in set (0.01 sec)

mysql> |
```

- **Với lệnh MySQL STATUS**

```
STATUS;
```



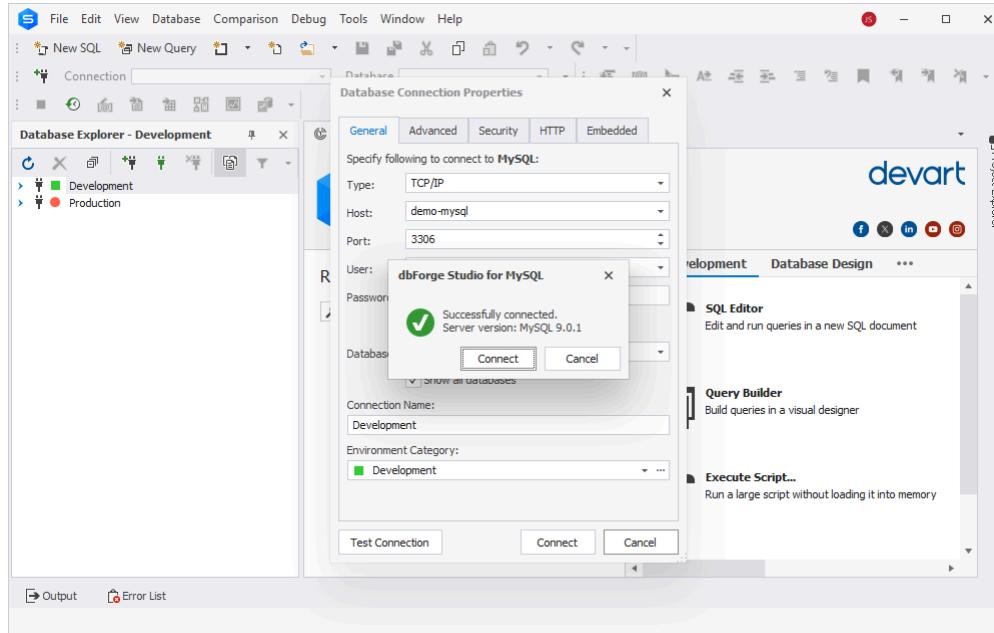
```
MySQL 9.0 Command Line Cli + - x
mysql> STATUS;
C:\Program Files\MySQL\MySQL Server 9.0\bin\mysql.exe [Ver 9.0.1 for Win64 on x86_64] (MySQL Community Server - GPL)

Connection id: 12
Current database:
Current user: root@localhost
SSL: Cipher in use is TLS_AES_128_GCM_SHA256
Using delimiter:
Server version: 8.3.0 MySQL Community Server - GPL
Protocol version: 10
Connection: localhost via TCP/IP
Server characterset: utf8mb4
Db characterset: utf8mb4
Client characterset: utf8mb4
Conn. characterset: utf8mb4
TCP port: 3306
Binary data as: Hexadecimal
Uptime: 2 days 2 hours 22 min 8 sec

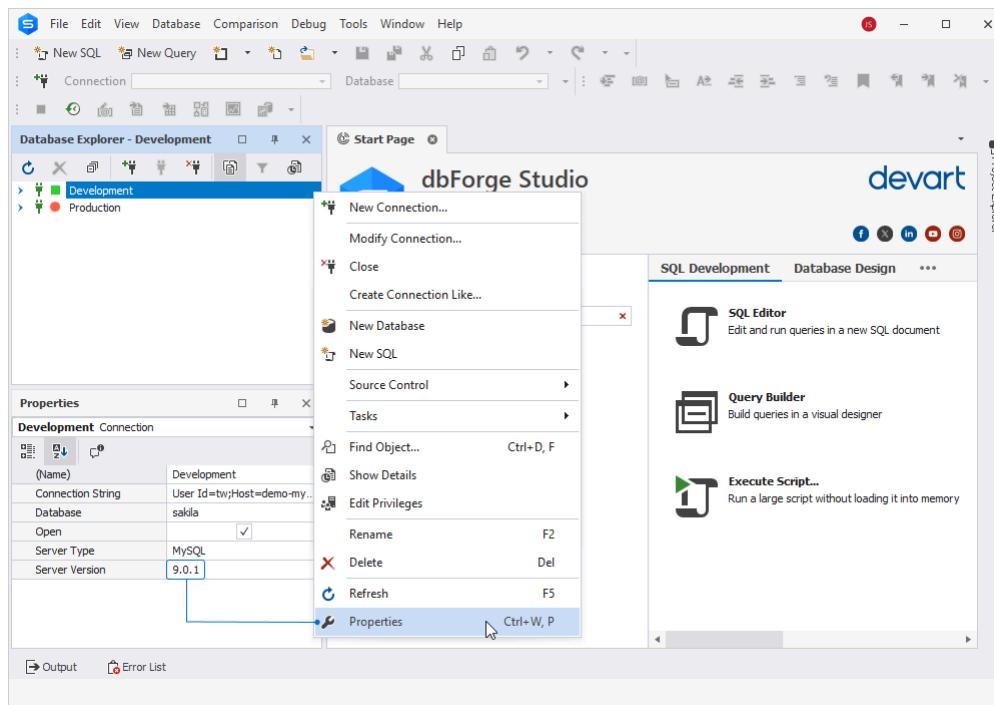
Threads: 2 Questions: 110 Slow queries: 0 Opens: 203 Flush tables: 3 Open tables: 121 Queries per sec
ond avg: 0.000
-----
```

Cách xác định phiên bản MySQL bằng dbForge Studio cho MySQL

- Đầu tiên, bạn sẽ nhận được thông tin về phiên bản máy chủ MySQL khi tùy chỉnh cài đặt kết nối. Trong cửa sổ Thuộc tính kết nối cơ sở dữ liệu, nhập cài đặt kết nối và nhấp vào **Kiểm tra kết nối**.

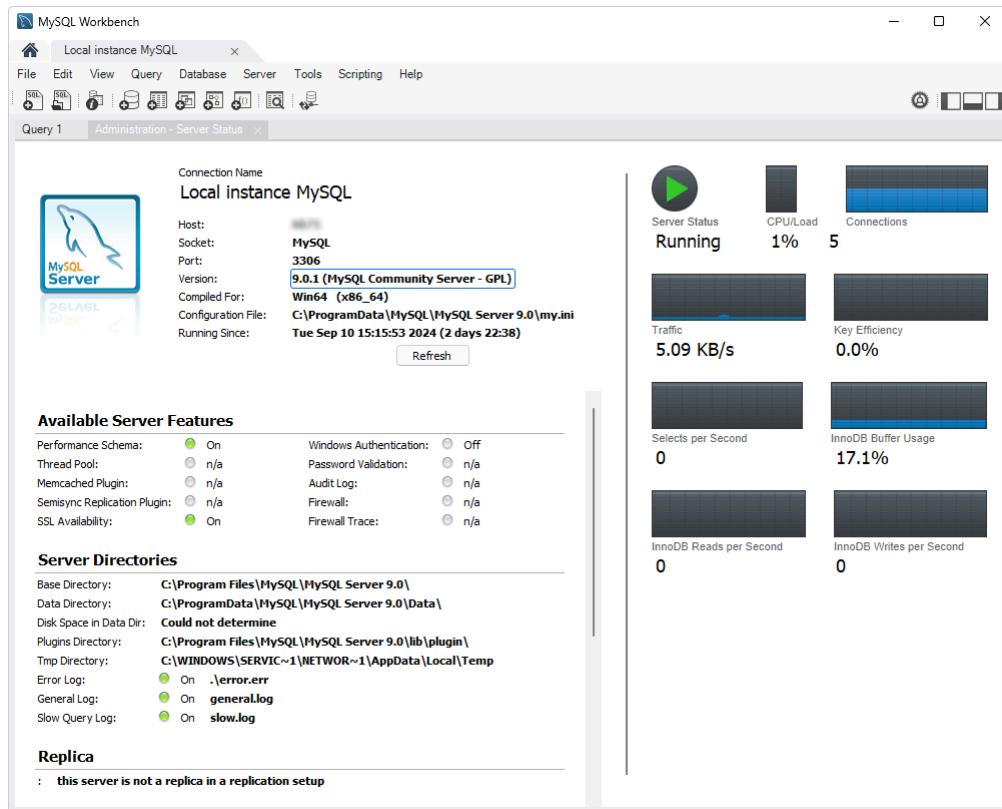


- Trong trường hợp bạn cần tìm hiểu phiên bản MySQL của mình sau khi đã kết nối, hãy nhấp chuột phải vào tên kết nối trong Database Explorer và chọn **Properties**.



Cách kiểm tra phiên bản MySQL trong Workbench

- Nếu bạn muốn biết cách kiểm tra phiên bản MySQL trong Workbench, chúng tôi sẽ cung cấp cho bạn một vài bước đơn giản. Đầu tiên, hãy mở Workbench và chọn máy chủ cơ sở dữ liệu của bạn trong menu chính, sau đó nhấp vào **Trạng thái máy chủ**. Tất cả thông tin liên quan đến lịch sử phiên bản của bạn được liệt kê trong cửa sổ này. Đây là cách bạn có thể kiểm tra xem mình có phiên bản MySQL mới nhất hay không và cập nhật nếu đã cũ.



1.10 InnoDB so với MyISAM

Công cụ InnoDB là gì?

- Công cụ lưu trữ đa năng InnoDB dành cho MySQL đã phát triển từ một hệ thống lưu trữ con thành một công cụ lưu trữ hoàn chỉnh. Đây là công cụ MySQL mặc định từ phiên bản 5.6 trở đi, nhờ sự kết hợp giữa hiệu suất cao và độ tin cậy. Công cụ này có thể dễ dàng tích hợp vào phần mềm. Trong khi MySQL xác định cách dữ liệu được lưu vào cơ sở dữ liệu, [công cụ lưu trữ InnoDB](#) lưu trữ dữ liệu trên đĩa hoặc giữ trong bộ nhớ chính để truy cập nhanh. Khi một giao dịch hoàn tất, dữ liệu được ghi vào phương tiện lưu trữ theo giao dịch. Điều quan trọng cần lưu ý là những thay đổi chưa hoàn tất không được lưu trữ trong cơ sở dữ liệu.

Công cụ MyISAM là gì?

- Trước đây, MyISAM được coi là lựa chọn tốt hơn, nhanh và được tối ưu hóa tốt cho các hoạt động đọc nặng. Nó được sử dụng cho nhiều mục đích khác nhau, từ thực hiện phân tích dữ liệu đến phát triển các hệ thống quản lý nội dung đơn giản cho diễn đàn hoặc xây dựng các công cụ tìm kiếm nhỏ hơn. Một số nhà phát triển đã tạo ra các công cụ và giải pháp được thiết kế riêng cho MyISAM, vì nó được coi là đơn giản hơn InnoDB. Trên thực tế, nhiều nhà phát triển đã bắt đầu hành trình đến thế giới MySQL bằng cách sử dụng MyISAM. Tuy nhiên, vào năm 2009, nó đã được thay thế bằng InnoDB làm công cụ lưu trữ MySQL mặc định. Tuy nhiên, nó vẫn cung cấp nhiều tiện ích mở rộng hữu ích cho các nhà phát triển MySQL.

MyISAM so với InnoDB: so sánh chi tiết

1. Loại động cơ lưu trữ

- Hãy bắt đầu với các loại công cụ MySQL. InnoDB là một công cụ lưu trữ giao dịch, trong khi MyISAM thuộc về danh mục không giao dịch. Cái trước có nghĩa là nếu thao tác dữ liệu của bạn liên quan đến giao dịch, thì lệnh khôi phục sẽ được kích hoạt tự động trong trường hợp giao dịch không hoàn tất. Rõ ràng là điều này được khuyến nghị hơn MyISAM, không hỗ trợ giao dịch và trong trường hợp cần thiết, bạn sẽ phải khôi phục các thay đổi theo cách thủ công.

2. Giao dịch

- InnoDB là một công cụ lưu trữ để xử lý giao dịch trực tuyến. Nó lưu trữ dữ liệu thực tế trong một vùng riêng biệt được gọi là vùng đệm InnoDB. Vùng này cho phép cơ sở dữ liệu MySQL định vị và truy cập dữ liệu một cách hiệu quả và nhanh chóng. Công cụ InnoDB là chìa khóa cho sự ổn định và hiệu suất của cơ sở dữ liệu MySQL. Ngược lại, công cụ MyISAM là một công cụ lưu trữ cũ hơn và ít phổ biến hơn để lưu trữ các bảng.

3. Tính chất của ACID

- Tính nguyên tử, tính nhất quán, tính cô lập và độ bền là các thuộc tính ACID nổi tiếng. Trong trường hợp xảy ra lỗi hoặc lỗi hệ thống, việc tuân thủ các thuộc tính này đảm bảo giao dịch sẽ được hoàn tất. Trong khi InnoDB cung cấp tính tuân thủ đầy đủ, MyISAM thì không.

4. So sánh hiệu suất

- InnoDB hỗ trợ các thuộc tính giao dịch. Nó cung cấp tốc độ viết mã cao hơn với các lệnh khôi phục và xác nhận. Hiệu suất của InnoDB với khối lượng dữ liệu lớn tốt hơn so với MyISAM. Trong khi đó, MyISAM đọc nhanh hơn, không hỗ trợ các thuộc tính giao

dịch và hiệu suất của nó với khối lượng dữ liệu lớn còn nhiều điều đáng mong đợi khi so sánh với InnoDB.

5. Hỗ trợ khóa ngoại

- Khóa [ngoại](#) của một bảng là một tập hợp các thuộc tính tham chiếu đến khóa chính của một bảng khác. Một bảng có thể có nhiều khóa ngoại và mỗi khóa ngoại có thể có một bảng cha khác nhau. Với MyISAM, bạn không thể thêm ràng buộc khóa ngoại, trong khi InnoDB hỗ trợ đầy đủ tính năng này.

6. Khóa cấp bảng so với khóa cấp hàng

- Khi tùy chọn khóa được kích hoạt, hai hoặc nhiều người dùng không thể sửa đổi cùng một dữ liệu cùng lúc. Tính năng này bảo toàn tính hợp lệ của dữ liệu. Khóa bảng là phương pháp khóa mặc định cho MyISAM, cho phép bạn sửa đổi các bảng trong một phiên duy nhất. Các bảng được khóa theo một thứ tự nhất định. Phương pháp khóa bảng có thể được sử dụng cho các cơ sở dữ liệu chỉ đọc không yêu cầu nhiều bộ nhớ.
- Đối với InnoDB, phương pháp khóa mặc định là khóa cấp hàng. Các hàng được khóa để hỗ trợ nhiều phiên trên các hàng đã chọn. Phương pháp này được ưa chuộng hơn nhiều đối với các cơ sở dữ liệu yêu cầu nhiều hơn một người dùng đang hoạt động.
- Tóm lại: InnoDB cung cấp khóa cấp hàng linh hoạt trong khi MyISAM chỉ có thể khóa cấp bảng. Phục hồi sự cố trong InnoDB cũng vượt trội hơn.

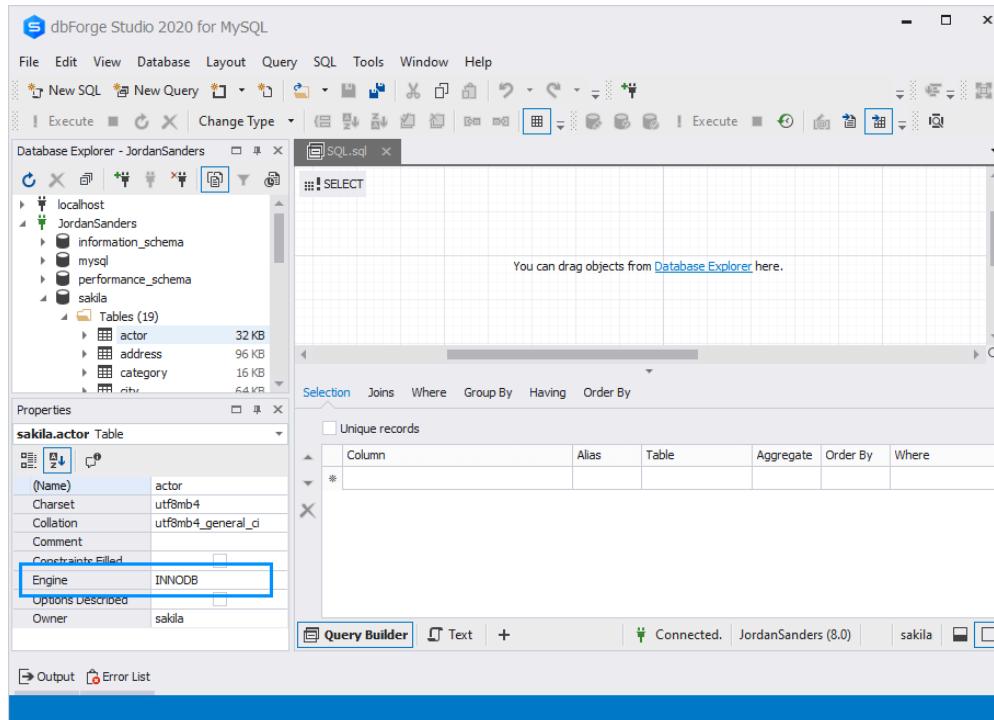
7. Lưu trữ đệm và lập chỉ mục

- Bây giờ là một vài lời về bộ nhớ đệm và lập chỉ mục trong InnoDB và MyISAM. Trong InnoDB, có một nhóm bộ đệm lớn có thể được sử dụng để lưu trữ cả dữ liệu và chỉ mục. Đối với MyISAM, có một bộ đệm khóa được sử dụng cho các chỉ mục. Trong khi đó, cơ chế bộ nhớ đệm chính là khóa bộ nhớ đệm, sử dụng các tệp MYI để lưu trữ các trang.

Cách kiểm tra xem bạn đang sử dụng MyISAM hay InnoDB

- Nếu bạn không biết cơ sở dữ liệu MySQL của mình hiện đang sử dụng storage engine nào, hãy kiểm tra và xem. Cách đầu tiên là khởi chạy MySQL Shell và chạy lệnh , sau đó định vị và chọn cơ sở dữ liệu cần thiết bằng lệnh. SHOW DATABASES;``USE database_name;
- Sau đó, một thông báo xác nhận sẽ hiển thị rằng cơ sở dữ liệu đã được định vị. Bây giờ bạn có thể sử dụng [lệnh SHOW CREATE TABLE](#) để hiển thị thông tin về bảng và công cụ lưu trữ. Nó trông như sau:
SHOW CREATE TABLE database_name.table_name;
- Đầu ra sẽ hiển thị công cụ lưu trữ mặc định cho bảng cần thiết.

- Có một cách dễ hơn để thực hiện nếu bạn đang sử dụng [dbForge Studio cho MySQL](#). Trong **Database Explorer**, chỉ cần nhấp chuột phải vào bảng cần thiết và nhấp vào **Properties** trên menu phím tắt. Như bạn có thể thấy, chúng ta có “INNODB” trong trường **Engine** của cửa sổ **Properties** ở góc dưới bên trái của màn hình.

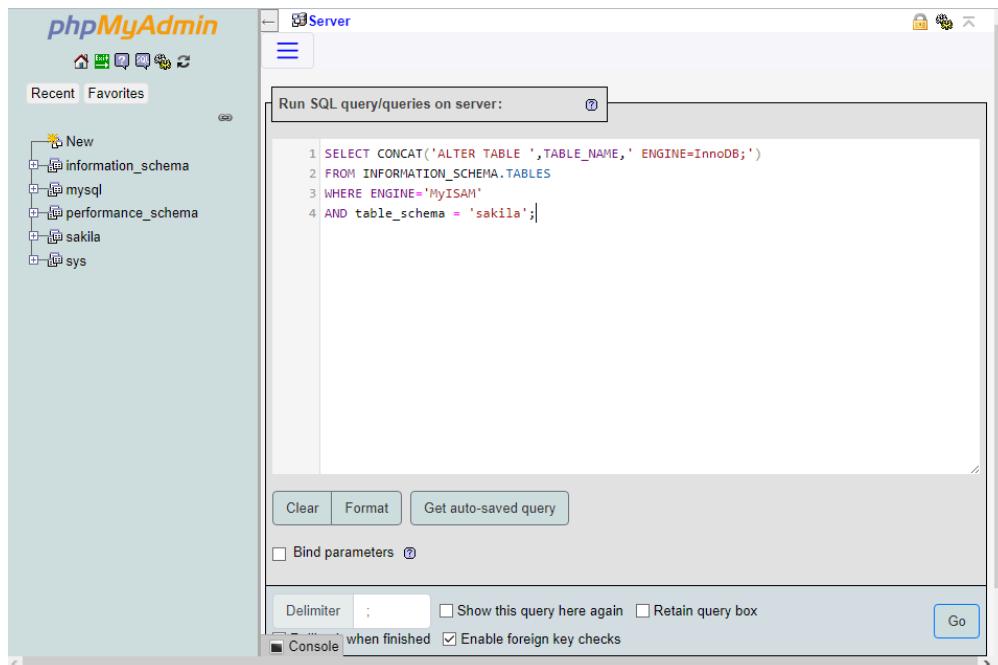


Cách chuyển đổi MyISAM sang InnoDB (và ngược lại)

- Sau khi khởi chạy và chọn cơ sở dữ liệu cần thiết, bạn có thể chạy truy vấn sau trên đó, thay thế `database_name` bằng tên thực của cơ sở dữ liệu của bạn bằng cách sử dụng [nối MySQL](#) :

```
SELECT CONCAT('ALTER TABLE ',TABLE_NAME,' ENGINE=InnoDB;') FROM INFORMATION_SCHEMA.TABLES WHERE ENGINE='MyISAM' AND table_schema = 'database_name';
```

Ví dụ, cơ sở dữ liệu sakila sẽ trông như thế này:



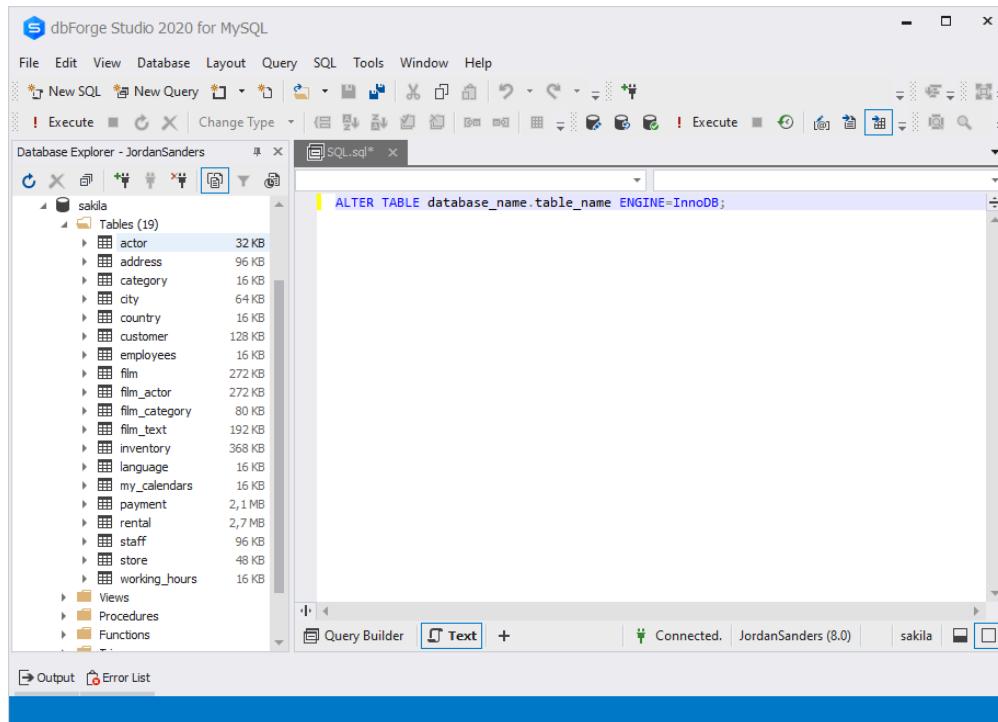
2. Sau khi bạn gửi truy vấn này, bạn sẽ thấy danh sách tất cả các bảng sẽ được chuyển đổi sang InnoDB.
3. Nhấp vào **+Tùy chọn** phía trên kết quả, chọn **Toàn văn** và nhấp vào **Đi** ở góc dưới bên phải màn hình.
4. Chọn hộp kiểm **Hiển thị tất cả** và sao chép tất cả truy vấn bằng cách sử dụng **Sao chép vào bảng tạm** trong hộp **Thao tác kết quả truy vấn**.
5. Dán kết quả vào trình soạn thảo văn bản và sao chép tất cả các dòng bắt đầu bằng ALTER TABLE vào bảng tạm.
6. Nhấp vào tab SQL phía trên kết quả và dán các câu lệnh ALTER TABLE vào trường văn bản, sau đó nhấp vào **Go** một lần nữa. Tất cả các bảng trong cơ sở dữ liệu của bạn sẽ được chuyển đổi thành InnoDB.

Cách chuyển đổi một bảng MyISAM duy nhất sang InnoDB

Nếu bạn muốn chuyển đổi một bảng duy nhất từ MyISAM sang InnoDB, hãy chạy lệnh ALTER TABLE, thay thế *database_name* và *table_name* bằng tên cơ sở dữ liệu và tên bảng thực tế:

```
ALTER TABLE database_name.table_name ENGINE=InnoDB;
```

Bạn cũng có thể thực hiện việc này bằng cách sử dụng [dbForge Studio cho MySQL](#) đã đề cập ở trên :



Cách chuyển đổi InnoDB sang MyISAM

Tương tự như vậy, bạn có thể muốn chuyển đổi một bảng InnoDB sang MyISAM. Phương pháp này giống như trong trường hợp trước:
ALTER TABLE database_name.table_name ENGINE=MyISAM;

1.11 Cơ sở dữ liệu mẫu MySQL

Cách tải xuống và cài đặt cơ sở dữ liệu mẫu MySQL

Nếu bạn chọn cơ sở dữ liệu mẫu sakila, [gói ZIP tải xuống](#) của nó có sẵn trên trang chính thức. Thư mục đã giải nén chứa ba tệp:

- **sakila-schema.sql** — tệp này chứa tất cả các câu lệnh CREATE cần thiết để tái tạo toàn bộ lược đồ sakila và do đó tạo cơ sở dữ liệu trên máy của bạn.

- **sakila-data.sql** — tệp chứa tất cả các câu lệnh INSERT để điền vào lược đồ và định nghĩa kích hoạt. Theo cách này, bạn nhận được tất cả dữ liệu cho cơ sở dữ liệu của mình.
- **sakila.mwb** — tệp chứa mô hình dữ liệu cho MySQL Workbench (IDE MySQL mặc định). Bạn có thể mở tệp bằng IDE đó và khám phá cấu trúc cơ sở dữ liệu.

Quá trình cài đặt cơ sở dữ liệu mẫu sakila MySQL rất đơn giản:

Bước 1. Giải nén tệp sakila-db.zip vào thư mục tạm thời (thường là C:\temp, nhưng cũng có thể là đường dẫn /tmp/ khác). Thư mục giải nén sẽ chứa ba tệp mà chúng tôi đã đặt tên trước đó.

Bước 2. Kết nối với máy chủ MySQL của bạn. Sử dụng máy khách dòng lệnh cho MySQL (tốt hơn là sử dụng tài khoản người dùng root, nhưng bạn cũng có thể sử dụng tài khoản không phải root nếu có quyền tạo cơ sở dữ liệu mới):

- \$> mysql -u root -p

Nhập mật khẩu khi được yêu cầu.

Bước 3. Thực thi sakila-schema.sql và sau đó thực thi sakila-data.sql. Các tập lệnh này sẽ tạo lược đồ cơ sở dữ liệu và điền dữ liệu vào đó.

- mysql> NGUỒN C:/temp/sakila-db/sakila-schema.sql;
- mysql> NGUỒN C:/temp/sakila-db/sakila-data.sql;

Ghi chú

Thay thế các đường dẫn đã chỉ định bằng đường dẫn thực tế tới các tệp tập lệnh.

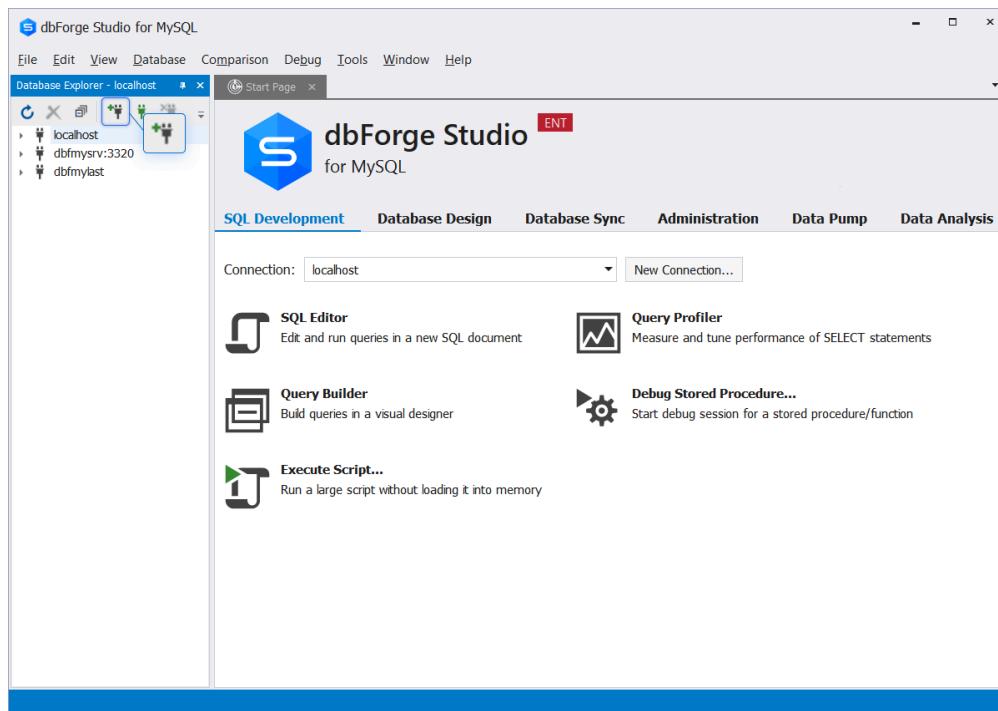
Khi cơ sở dữ liệu được cài đặt đúng cách, bạn sẽ nhận được thông báo xác nhận. Vậy là xong. Bạn có thể sử dụng cơ sở dữ liệu mẫu này để thiết kế và kiểm tra các truy vấn có độ phức tạp bất kỳ, theo cách thủ công hoặc với sự trợ giúp của các công cụ chuyên nghiệp.

Ở trên, chúng tôi đã đề cập đến MySQL IDE mặc định — Workbench. Nó phổ biến, nhưng không phải là lựa chọn duy nhất khả dụng. dbForge Studio for MySQL là một IDE toàn diện dựa trên GUI cho phép bạn quản lý và quản lý các tác vụ liên quan đến MySQL. Nó có

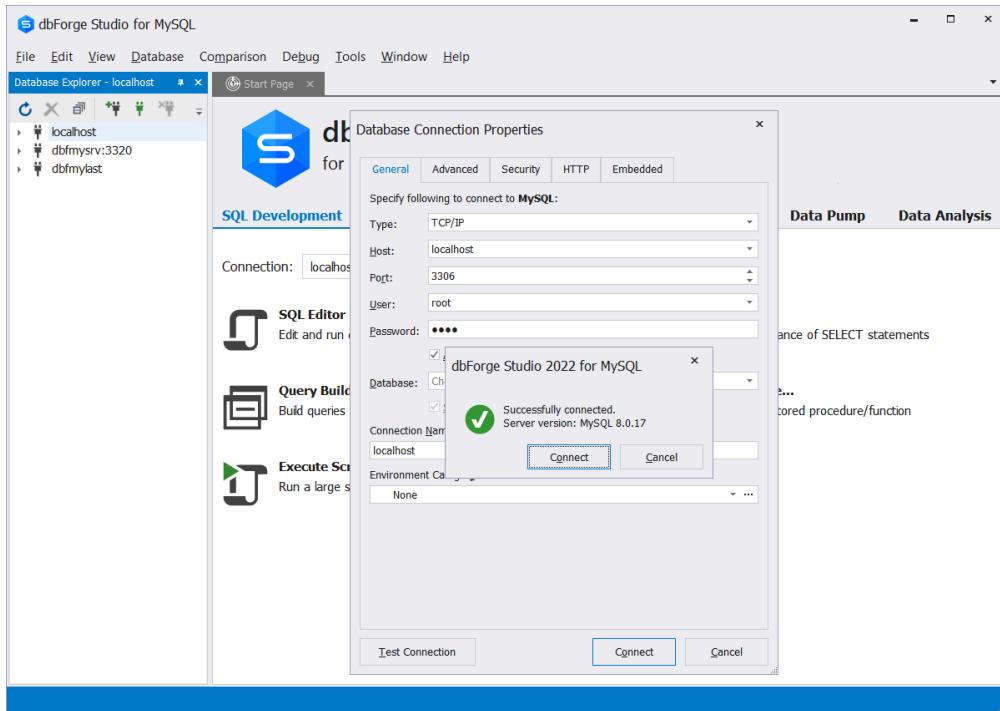
hiệu quả cao, nhanh như chớp và cực kỳ thân thiện với người dùng. Do đó, bạn có thể sử dụng nó để làm việc với sakila hoặc bất kỳ cơ sở dữ liệu thử nghiệm/sản xuất nào khác. Cách tiếp cận là giống nhau.

Kết nối với máy chủ MySQL trong dbForge Studio cho MySQL

Để bắt đầu sử dụng [dbForge Studio cho MySQL](#), trước tiên chúng ta cần kết nối đến máy chủ MySQL. Mở Studio và bạn sẽ thấy biểu tượng "kết nối" trong menu Database Explorer.

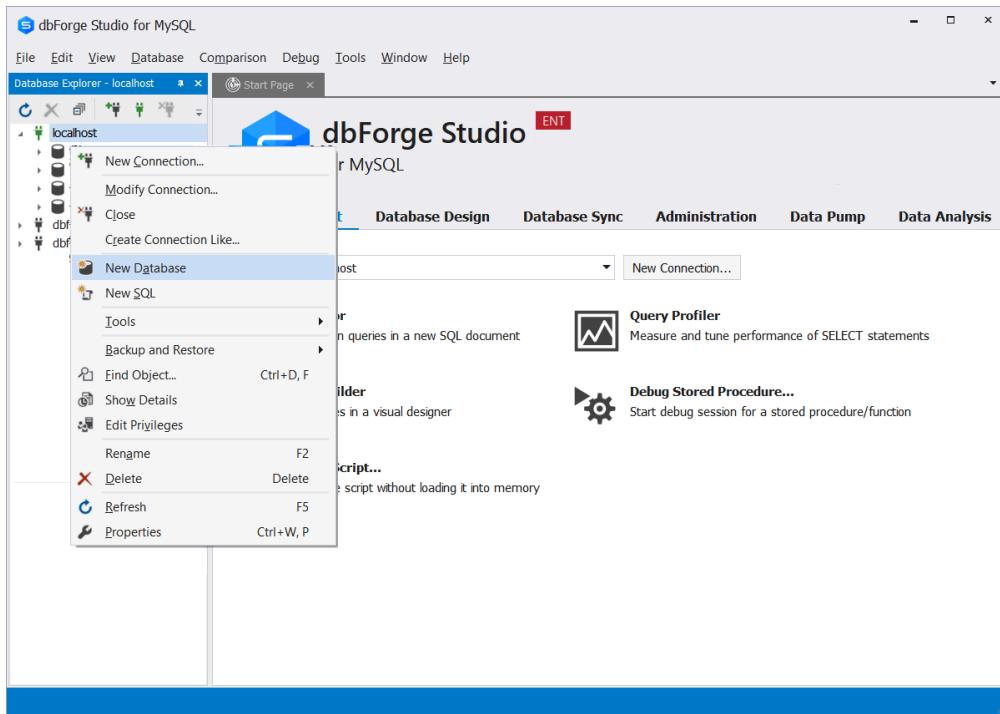


Trong cửa sổ mới sẽ xuất hiện, hãy cung cấp thông tin về phiên bản máy chủ MySQL để kết nối (tên người dùng gốc, mật khẩu, tên phiên bản cơ sở dữ liệu, v.v.). Sau đó nhấp vào **Kiểm tra kết nối**. Nếu kết nối thành công, bạn sẽ thấy thông báo xác nhận.

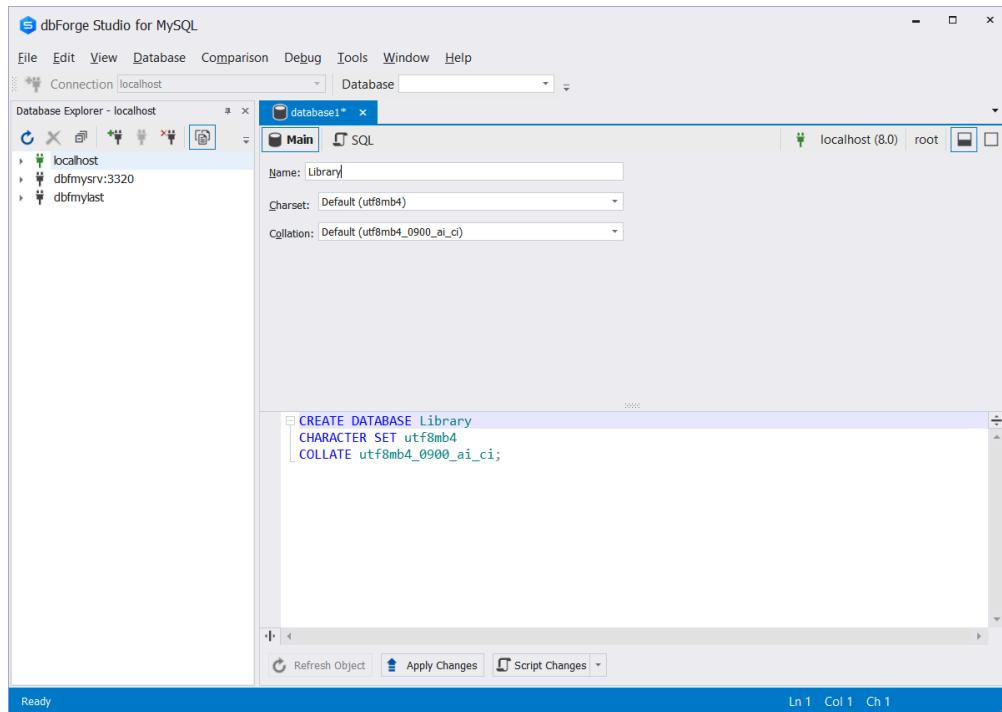


Tạo cơ sở dữ liệu mới

- Giả sử bạn không sử dụng cơ sở dữ liệu thử nghiệm do MySQL tạo ra vì một lý do nào đó. Thay vào đó, bạn có thể tự tạo cơ sở dữ liệu của riêng mình từ đầu. dbForge Studio for MySQL giúp thực hiện nhiệm vụ này nhanh chóng và dễ dàng.
- Để tạo cơ sở dữ liệu mới, hãy nhấp chuột phải vào tên máy chủ MySQL trong Database Explorer và chọn **New Database**.

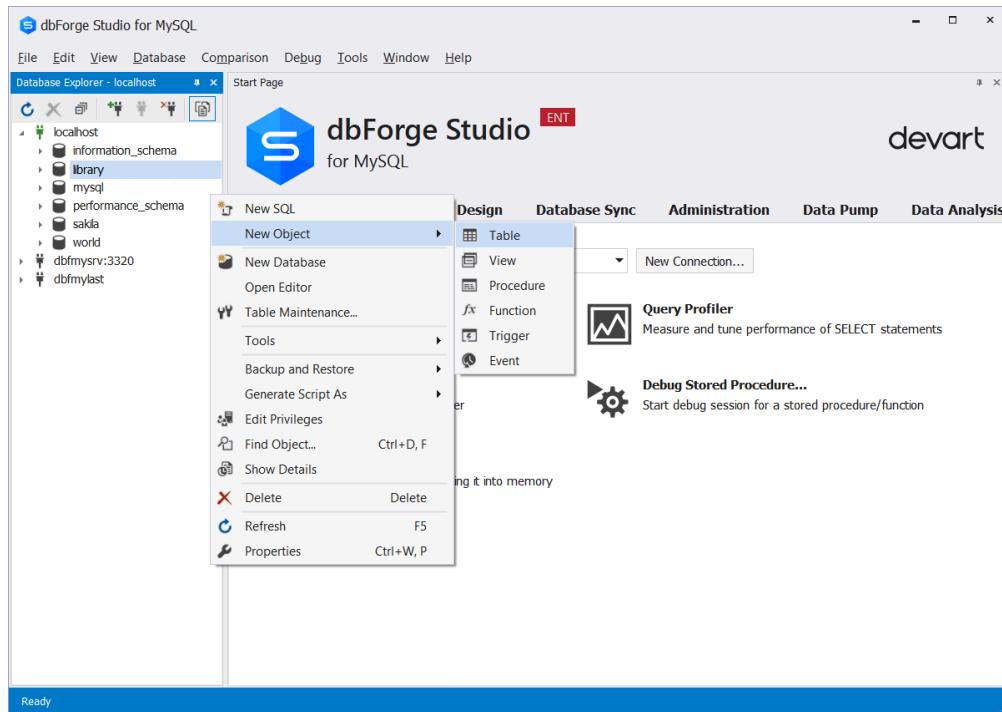


- Nhập tên cơ sở dữ liệu của bạn (trong trường hợp của chúng tôi là Thư viện). Ở cuối bảng điều khiển, bạn sẽ thấy truy vấn SQL được tạo để tạo cơ sở dữ liệu đó. Để hoàn tất việc tạo cơ sở dữ liệu, hãy nhấp vào **Áp dụng thay đổi**.



Tạo một bảng mới

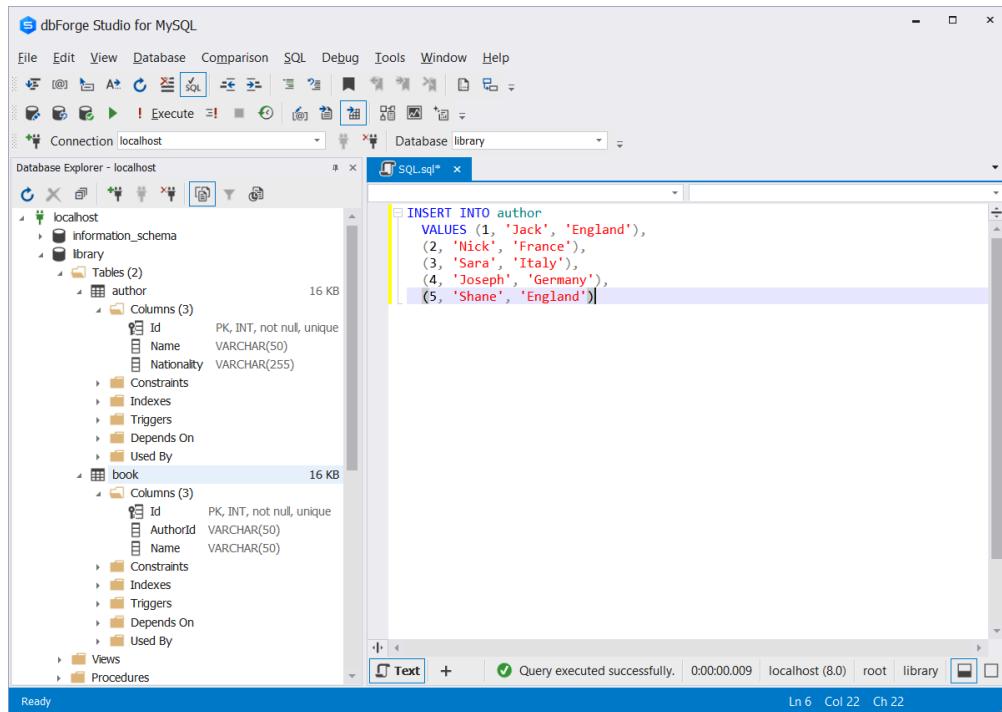
- Cơ sở dữ liệu của chúng tôi đang trống. Để thêm bảng, hãy nhấp chuột phải vào tên cơ sở dữ liệu đó rồi chọn **New Object > Table**.



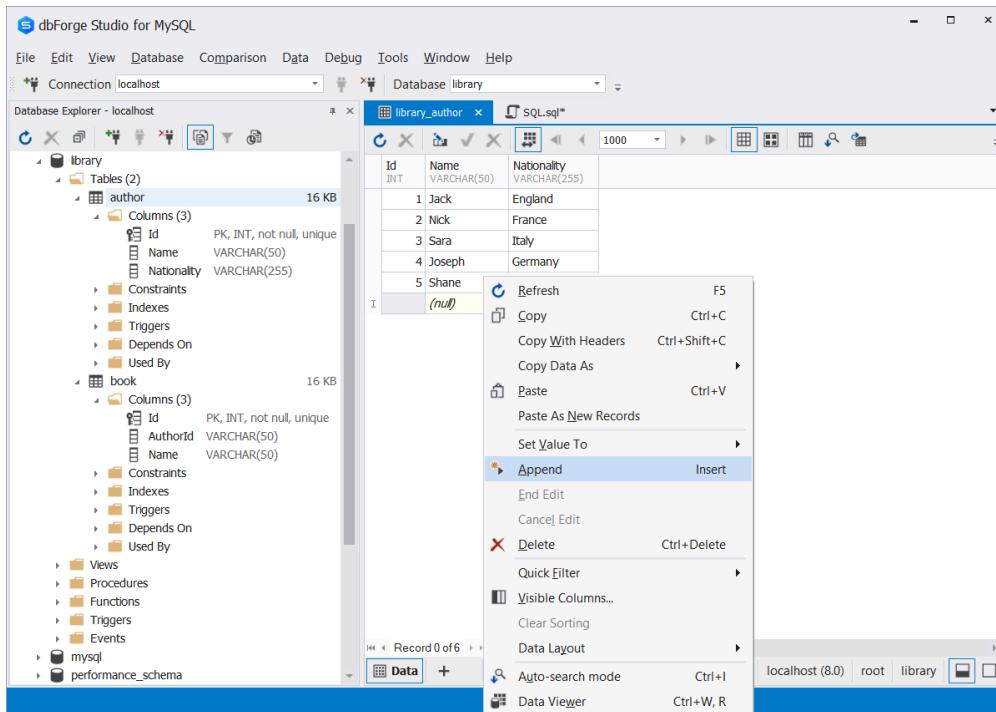
Thực hiện các hoạt động CRUD

- **Thêm hồ sơ**

Để tạo hoặc chèn bản ghi mới thông qua truy vấn SQL, hãy nhấp vào **New SQL** trên thanh công cụ chính. Trong cửa sổ SQL Editor sẽ xuất hiện, hãy nhập truy vấn SQL INSERT của bạn và nhấp vào **Execute** (hoặc nhấn **F5**).



Hoặc, bạn có thể sử dụng một cách tiếp cận khác. Nhấp chuột phải vào tên bảng và chọn **Retrieve Data**. Thao tác này sẽ mở cửa sổ chứa tất cả các bản ghi từ bảng đó. Để chèn các bản ghi mới, nhấp chuột phải vào phần thân SQL Editor và nhấp vào **Append**. Một hàng mới sẽ xuất hiện ở phía dưới. Tất cả những gì bạn cần làm là nhập các giá trị.



- **Đọc/Chọn hồ sơ**

Để chọn các hàng từ bảng MySQL, bạn có thể sử dụng các phương pháp sau:

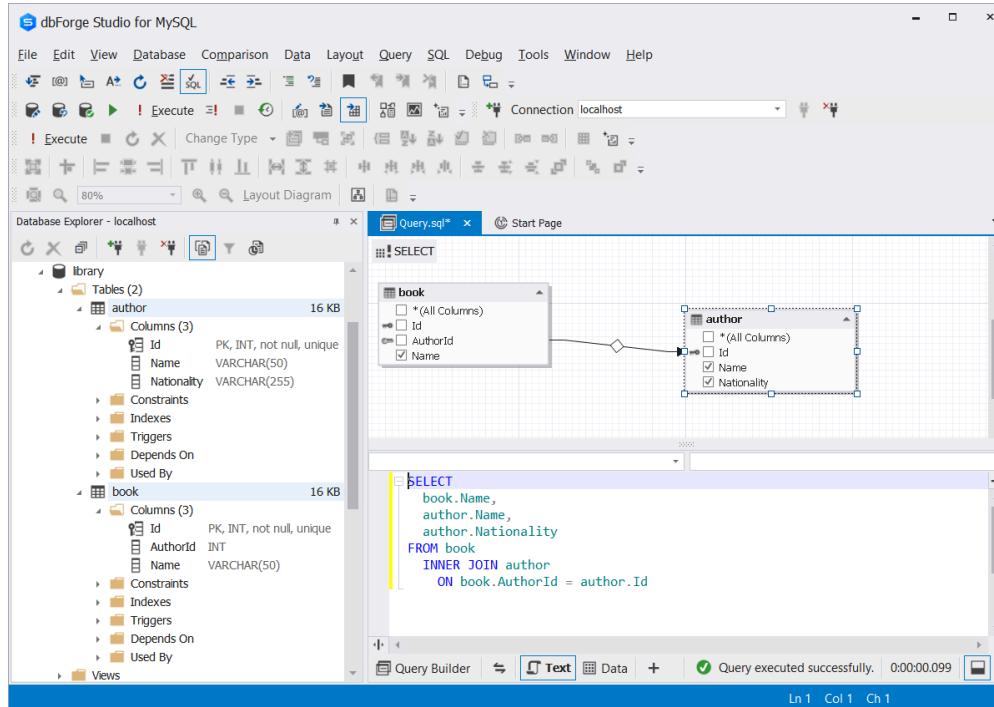
- Thực hiện truy vấn SQL
- Lấy dữ liệu trực quan
- Sử dụng chức năng Query Builder

Bạn có thể đọc dữ liệu MySQL theo cách truyền thống bằng cách viết và thực hiện truy vấn trong SQL Editor. Khi thực hiện điều đó trong Studio, bạn có thể hưởng lợi rất nhiều từ các tính năng tích hợp nâng cao: hoàn thành mã, chọn cột, kiểm tra cú pháp, v.v.

Để nhanh chóng lấy lại toàn bộ dữ liệu từ bảng MySQL, chỉ cần nhấp chuột phải vào bảng cần thiết trong Database Explorer rồi chọn **Retrive Data**.

Chức năng Query Builder cho phép linh hoạt hơn khi chọn các hàng MySQL. Chọn và nhấp chuột phải vào các bảng bạn muốn lấy dữ liệu trong Database Explorer, sau đó vào **Send to > Query Builder**. Tiếp theo, trên sơ đồ, chọn các cột bạn cần chọn dữ liệu và nhập

vào **Execute**. Nếu bạn chuyển sang chế độ xem Text, bạn sẽ thấy truy vấn được tạo tự động. Lưu ý rằng nó chứa một mệnh đề JOIN.



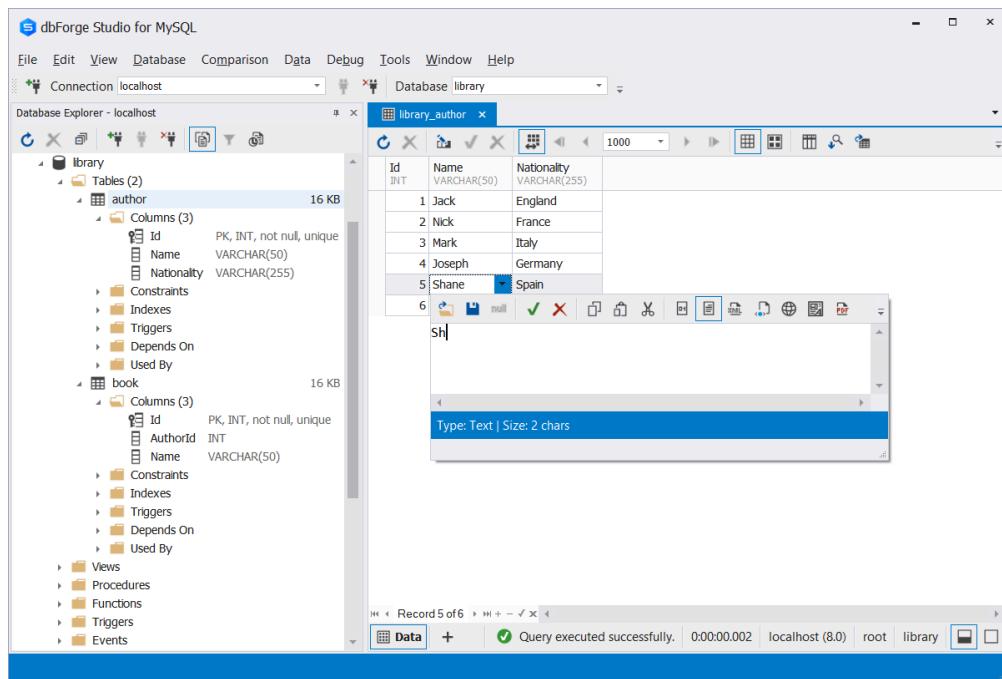
- **Cập nhật hồ sơ**

Bạn có thể cập nhật một bản ghi trong một bảng MySQL bằng cách sử dụng truy vấn SQL. Ví dụ:

CẬP NHẬT tác giả a ĐẶT Tên = "Mark"

NGÌ Id = 3;

Bạn cũng có thể cập nhật bản ghi trực quan trong chế độ xem lưới. Đầu tiên, chọn bản ghi bằng tùy chọn Truy xuất dữ liệu như đã trình bày trong các phần trước. Sau đó, ngay trong lưới kết quả, bạn có thể cập nhật bản ghi. Chỉ cần đảm bảo rằng bảng không ở chế độ đọc-một.



- **Xóa hồ sơ**

Có nhiều cách để xóa bản ghi trong bảng MySQL. Cách đầu tiên, tuy nhiên không dễ dàng, là viết và thực hiện truy vấn DELETE.

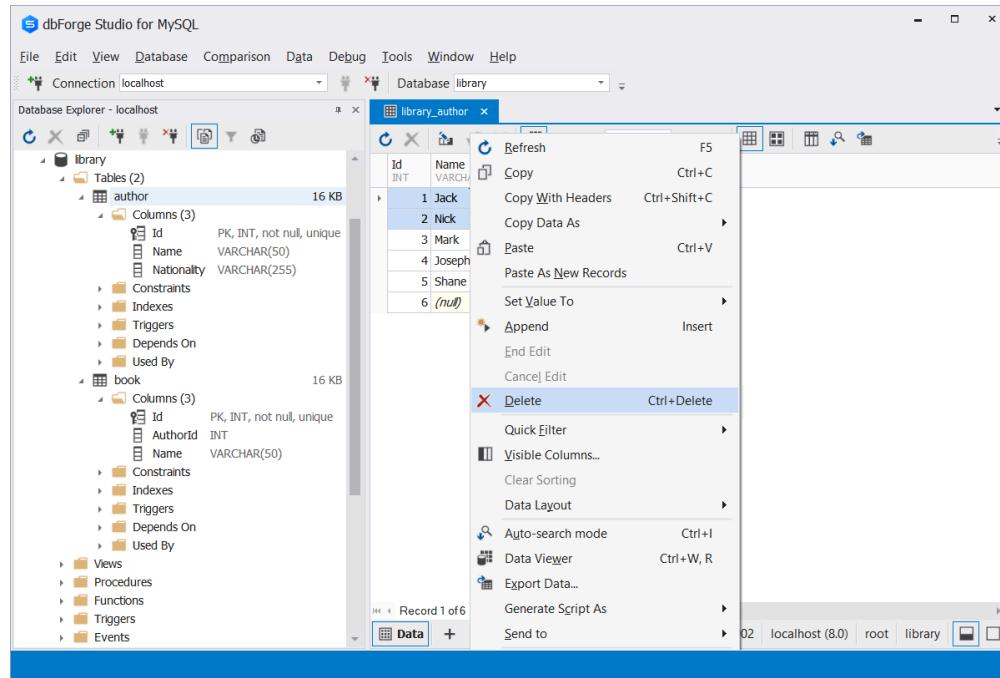
DELETE

FROM author

WHERE Name = 'Jack';

Để xóa bản ghi trực quan trong dbForge Studio for MySQL, trước tiên, hãy truy xuất dữ liệu. Tiếp theo, chọn và nhấp chuột phải vào

các bản ghi cần thiết, sau đó chọn **Xóa** từ menu phím tắt xuất hiện.



1.12 Bắt đầu với dbForge Studio for MySQL

5 lý do hàng đầu để sử dụng MySQL

1. Dễ sử dụng

Một điều nữa là MySQL khá đơn giản và thân thiện với người dùng. Vì vậy, nếu bạn biết một hoặc hai điều về SQL (ngôn ngữ truy vấn có cấu trúc) và quen thuộc với các câu lệnh SQL cơ bản, bạn sẽ có thể xây dựng và tương tác với MySQL.

2. Mức độ bảo mật cao

Một lợi thế quan trọng khác là MySQL được công nhận trên toàn cầu là hệ thống quản lý cơ sở dữ liệu an toàn và đáng tin cậy nhất. Cơ sở dữ liệu MySQL cung cấp lớp bảo mật dữ liệu vững chắc giúp bảo vệ dữ liệu nhạy cảm khỏi những kẻ xâm nhập.

3. Khả năng mở rộng

MySQL có thể hào về tính linh hoạt theo yêu cầu tuyệt vời: nó có khả năng quản lý hầu như bất kỳ khối lượng dữ liệu nào, lên đến

50 triệu hàng hoặc hơn trong một bảng. Nó cho phép tùy chỉnh gần như hoàn toàn: với giới hạn kích thước tệp ban đầu là 4 GB, bạn có thể tăng lên đến giới hạn lý thuyết là 8 TB dữ liệu.

4. Hiệu suất

Nhờ kiến trúc công cụ lưu trữ độc đáo, MySQL được thiết kế để hỗ trợ ngay cả những ứng dụng chuyên sâu nhất trong khi vẫn đảm bảo tốc độ phù hợp, chỉ mục toàn văn bản và bộ nhớ đệm để có kết quả hiệu suất cao.

5. Tính linh hoạt

Mặc dù MySQL là giải pháp mã nguồn mở, nhưng nó cung cấp hỗ trợ 24-7 tuyệt vời nhằm nâng cao trải nghiệm của người dùng. Bên cạnh đó, MySQL hỗ trợ rất nhiều ứng dụng nhúng, giúp cơ sở dữ liệu cực kỳ linh hoạt.

DbForge Studio for MySQL là gì?

- DbForge Studio for MySQL là một công cụ GUI đa chức năng cung cấp nhiều tính năng để phát triển, quản lý và quản trị cơ sở dữ liệu MySQL và MariaDB. Giao diện trực quan của IDE trao quyền cho bạn và tạo điều kiện cho các tác vụ như tạo và thực hiện truy vấn, phát triển và gỡ lỗi các quy trình được lưu trữ, tự động hóa quản lý đối tượng cơ sở dữ liệu và phân tích dữ liệu bảng. Nhưng hãy đợi đã, còn nhiều hơn thế nữa!

Bắt đầu với dbForge Studio cho MySQL

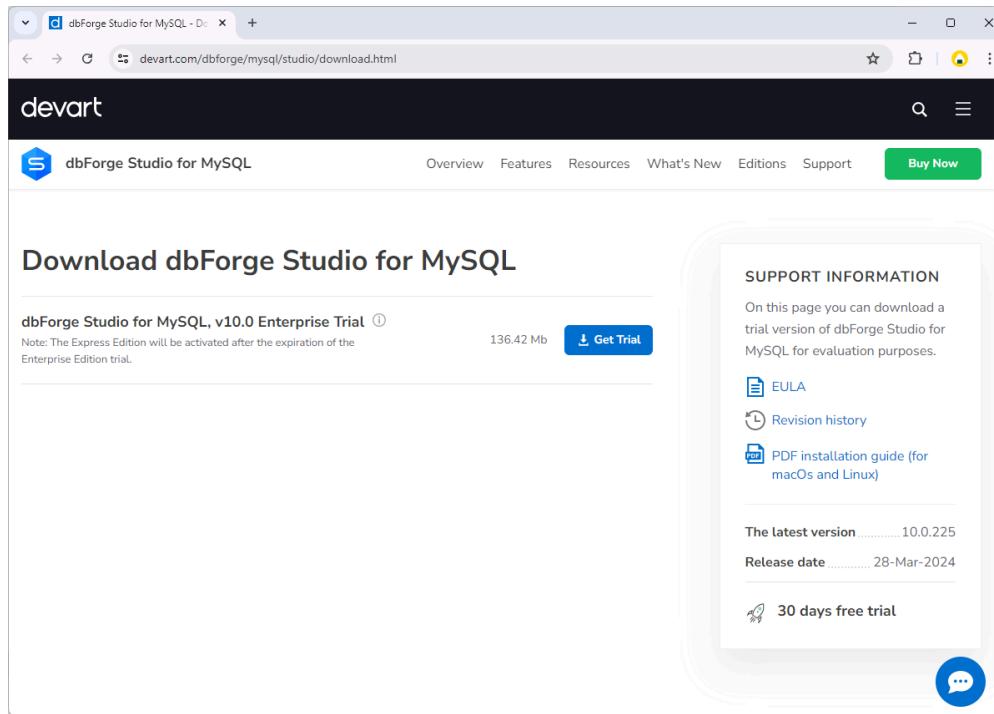
Quá trình cài đặt

Để bắt đầu, bạn nên [cài đặt](#) MySQL Server . Tùy thuộc vào hệ điều hành bạn đang sử dụng, quá trình cài đặt sẽ có đôi chút khác biệt:

Cửa sổ

1. [Tải xuống](#) dbForge Studio cho MySQL và chạy **dbforgemysql.exe** để bắt đầu quá trình cài đặt.
2. Mở dbForge Studio cho MySQL.

Bạn có thể tải xuống bản dùng thử miễn phí dbForge Studio for MySQL trong 30 ngày và tận hưởng những lợi ích mà phiên bản Enterprise của sản phẩm mang lại.



macOS và Linux

1. [Tải xuống và cài đặt CrossOver](#) cho macOS hoặc Linux.
2. Chọn một trong hai cách bạn sẽ thực hiện trong quá trình cài đặt: [dựa trên kịch bản](#) hoặc [thủ công](#) .
3. Chạy dbForge Studio for MySQL trên máy của bạn.

Bạn cũng có thể tham khảo [hướng dẫn cài đặt từng bước có minh họa](#) để biết hướng dẫn chi tiết hơn.

Làm quen

Sau khi hoàn tất cài đặt, bạn có thể tiến hành làm quen với chương trình. Để bắt đầu, hãy khởi chạy dbForge Studio for MySQL và bạn sẽ thấy màn hình sau, tức là Trang bắt đầu.

Phần trung tâm của Trang Bắt đầu hiển thị một số tab, đại diện cho các mô-đun chức năng chính: Phát triển SQL, Thiết kế Cơ sở dữ liệu, Đồng bộ Cơ sở dữ liệu, Quản trị, Bơm dữ liệu và Phân tích dữ liệu. Trước khi bắt đầu khám phá các chức năng có sẵn, bạn sẽ cần tạo kết

nối MySQL Server.

Tạo kết nối tới cơ sở dữ liệu

Trước khi thiết lập kết nối cơ sở dữ liệu, trước tiên bạn cần tạo kết nối máy chủ.

1. Mở hộp thoại **Thuộc tính kết nối cơ sở dữ liệu** bằng cách nhấp vào **Kết nối mới** trên menu Cơ sở dữ liệu hoặc bằng cách nhấp vào nút **Kết nối mới** trên thanh công cụ Kết nối:
 2. Trong hộp **Type**, chọn loại kết nối mong muốn: TCP/IP hoặc Named pipe. Tiếp theo, nhập **Host name** và thông tin **Port**. Số cổng mặc định là 3306. Đối với kết nối Named pipe, nhập tên pipe vào hộp Pipe.
 3. Nhập thông tin đăng nhập của bạn vào trường **Người dùng** và **Mật khẩu**.
 4. Sau đó, nhập hoặc chọn tên của **Cơ sở dữ liệu** mà bạn muốn kết nối. Lưu ý rằng **Tên kết nối** được tạo tự động trừ khi bạn chỉ định một tên cụ thể cho kết nối mới của mình. Nếu bạn chưa có cơ sở dữ liệu, bạn có thể tải xuống gói ZIP của [cơ sở dữ liệu mẫu MySQL](#) và giải nén.
 5. Tùy chọn, bạn có thể vào các tab khác và cấu hình các thuộc tính kết nối nâng cao, thuộc tính bảo mật (SSL hoặc SSH) và thuộc tính đường hầm HTTP.
 6. Bạn có thể nhấp vào **Kiểm tra kết nối** để xác minh rằng bạn có thể kết nối với cơ sở dữ liệu bằng cách sử dụng thông tin kết nối đã chỉ định.
- Nhấp vào Ok để hoàn tất việc tạo kết nối cơ sở dữ liệu và kiểm tra kết nối của bạn trong cửa sổ Database Explorer.

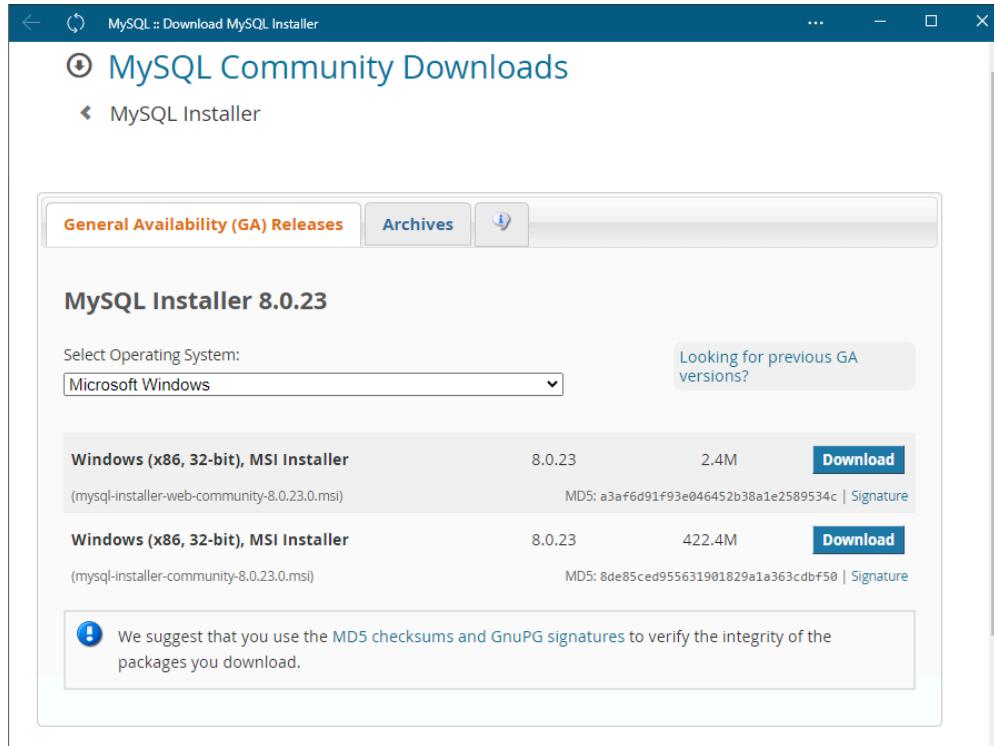
2. Cài đặt và kết nối MySQL

2.1 Cài đặt MySQL trên Windows

Tải xuống MySQL Installer

Để bắt đầu, hãy vào trang tải xuống [MySQL Installer](#). Như bạn thấy, có hai tùy chọn được đề xuất để tải xuống: phiên bản cộng đồng web và phiên bản đầy đủ. Sự khác biệt chính giữa hai phiên bản này là gói **web** chỉ bao gồm MySQL Installer và các tệp cấu hình, và nó yêu cầu kết nối Internet để tiến hành cài đặt. Trong phiên bản web, bạn có thể chọn các ứng dụng bạn muốn tải xuống ngoài MySQL Server.

Đối với gói **đầy đủ**, nó bao gồm tất cả các sản phẩm MySQL cho Windows (bao gồm cả MySQL Server) và phù hợp nếu bạn có ý định cài đặt MySQL ngoại tuyến.



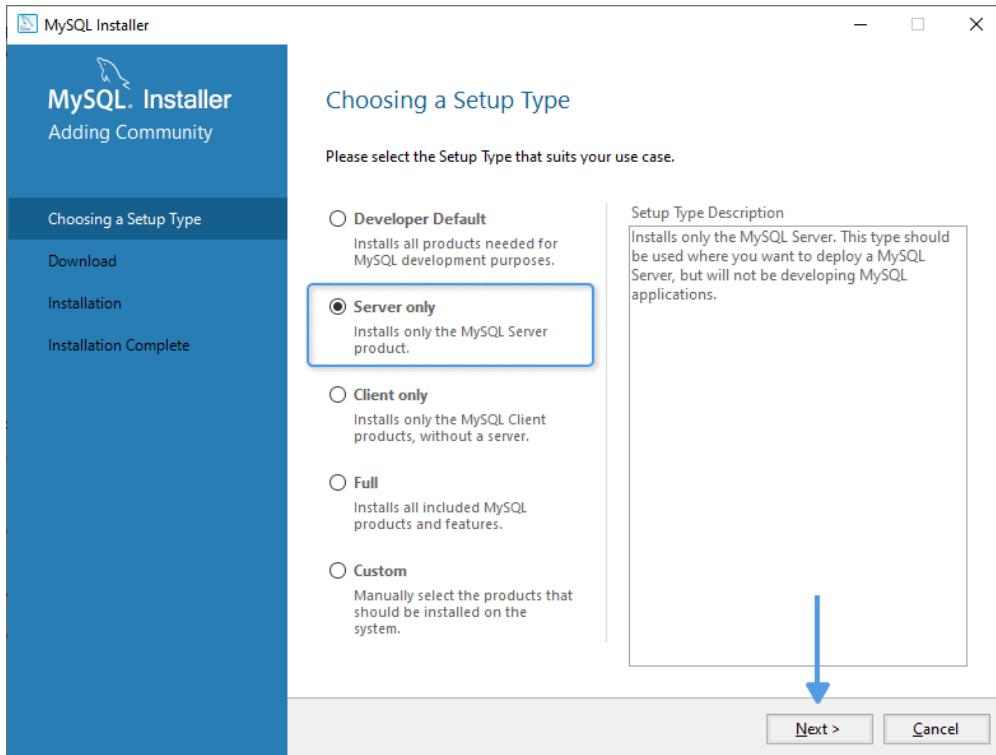
Thiết lập MySQL Installer cho Windows

Bây giờ, hãy chọn loại thiết lập phù hợp nhất với trường hợp sử dụng của bạn:

- **Developer Default** bao gồm cài đặt MySQL Server và tất cả các công cụ khác có thể được sử dụng để phát triển. Nếu bạn có kế hoạch xây dựng cơ sở dữ liệu MySQL từ đầu, loại này sẽ phù hợp nhất.
- **Server** chỉ để xuất cài đặt một phiên bản của MySQL Server. Loại này phù hợp nếu bạn không quản lý cơ sở dữ liệu trực tiếp.
- **Máy khách** chỉ cài đặt tất cả các ứng dụng và trình kết nối MySQL, ngoại trừ sản phẩm MySQL Server.
- Thiết lập **đầy đủ** sẽ cài đặt tất cả các sản phẩm MySQL có sẵn.

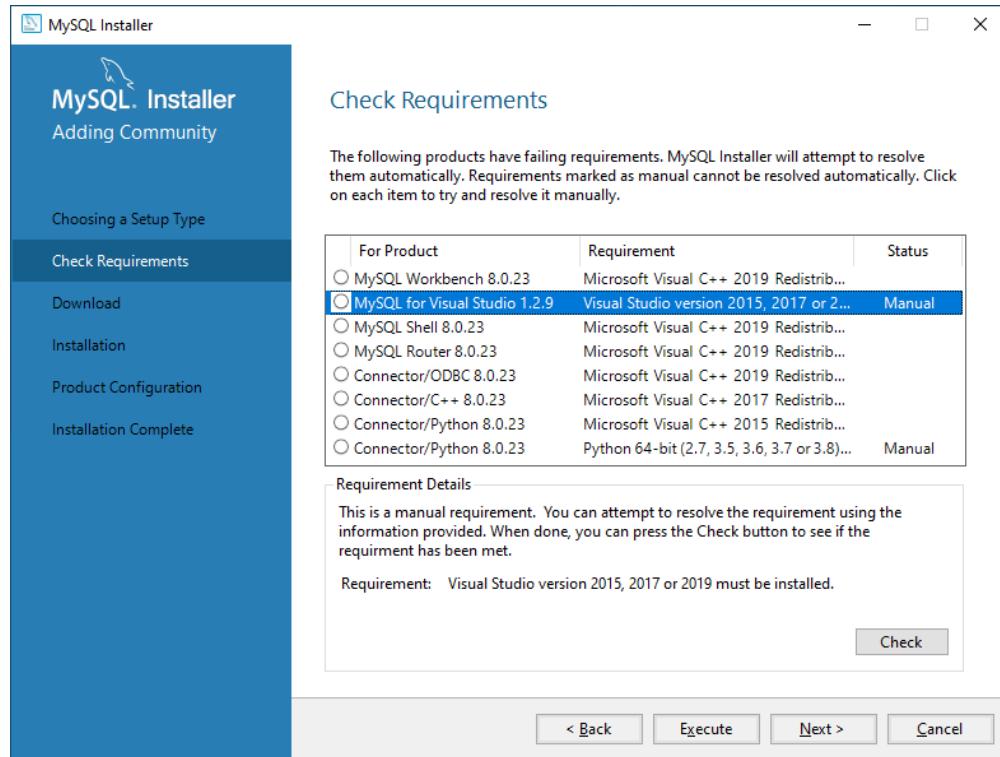
- **Kiểu Tùy chỉnh** cung cấp các tùy chọn rộng cho phép tùy chỉnh cài đặt và chọn các công cụ từ danh mục MySQL.

1. Sau khi chọn Kiểu thiết lập phù hợp, hãy nhấp vào **Tiếp theo** :



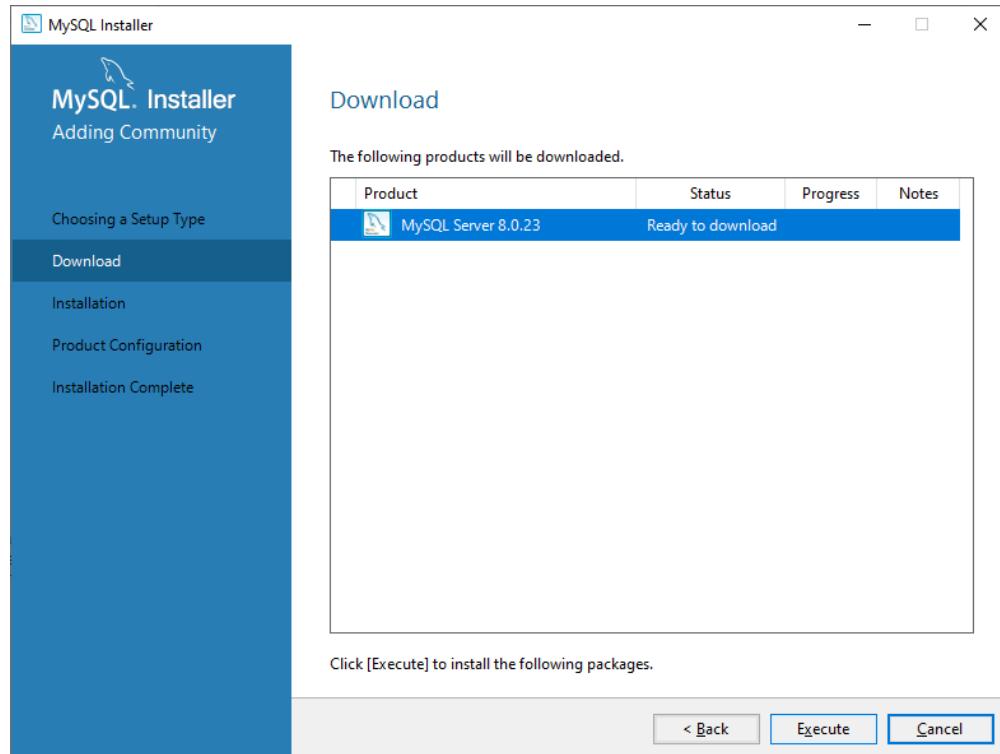
2. Sau đó, MySQL Installer đảm bảo rằng bạn đáp ứng các yêu cầu cần thiết cho quá trình cài đặt. Nếu có bất kỳ yêu cầu nào không thành công, trình cài đặt sẽ cố gắng tự động giải quyết các sự không nhất quán. Nếu không thực hiện được, bạn sẽ được yêu cầu giải quyết các vấn đề theo cách thủ công. Ví dụ, bạn có thể cần cài đặt một số ứng dụng hoặc gói bổ sung (ví dụ: Microsoft Visual C++ 2019 Redistributable Package). Bên cạnh đó, bạn có thể gặp phải sự cố về sự không nhất quán trong cài đặt Path trong trường

hợp bạn đã cài đặt MySQL trên Windows Server trước đó.



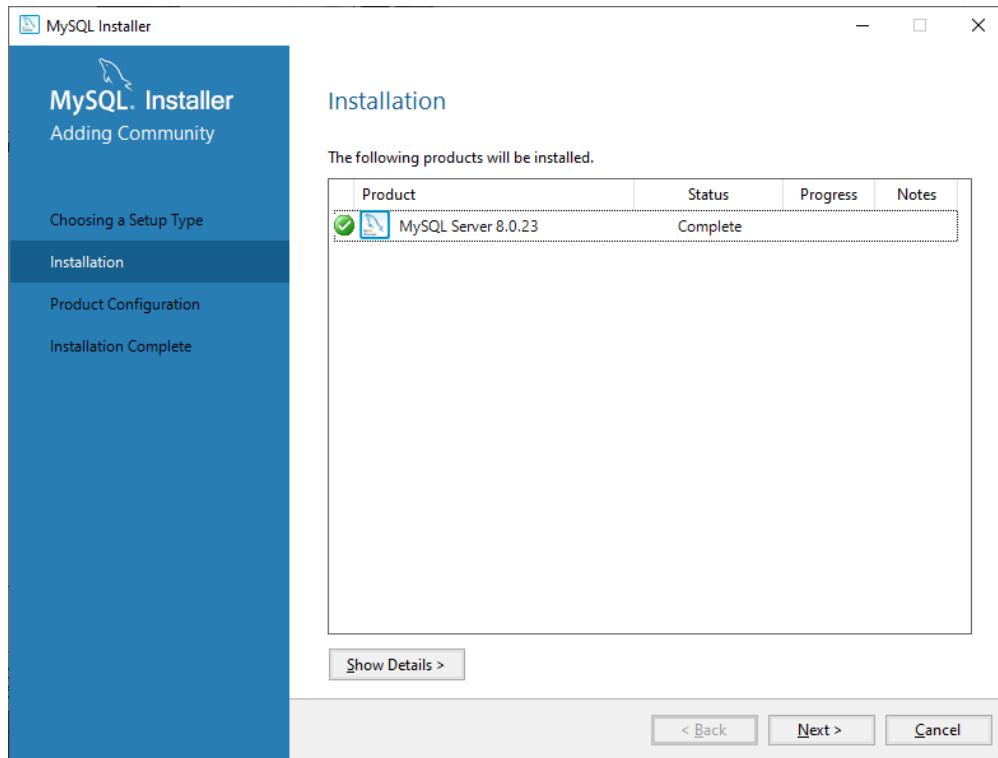
3. Kiểm tra trạng thái của các yêu cầu sản phẩm (nếu bạn có). Nếu bạn không có yêu cầu thủ công nào, hãy nhập vào **Thực thi** và Trình hướng dẫn sẽ cài đặt phần mềm cần thiết. Với điều kiện là bạn có các yêu cầu thủ công, hãy tự giải quyết chúng, nhập

vào **Kiểm tra** để xác minh, sau đó nhấp vào **Thực thi** để bắt đầu quá trình cài đặt:



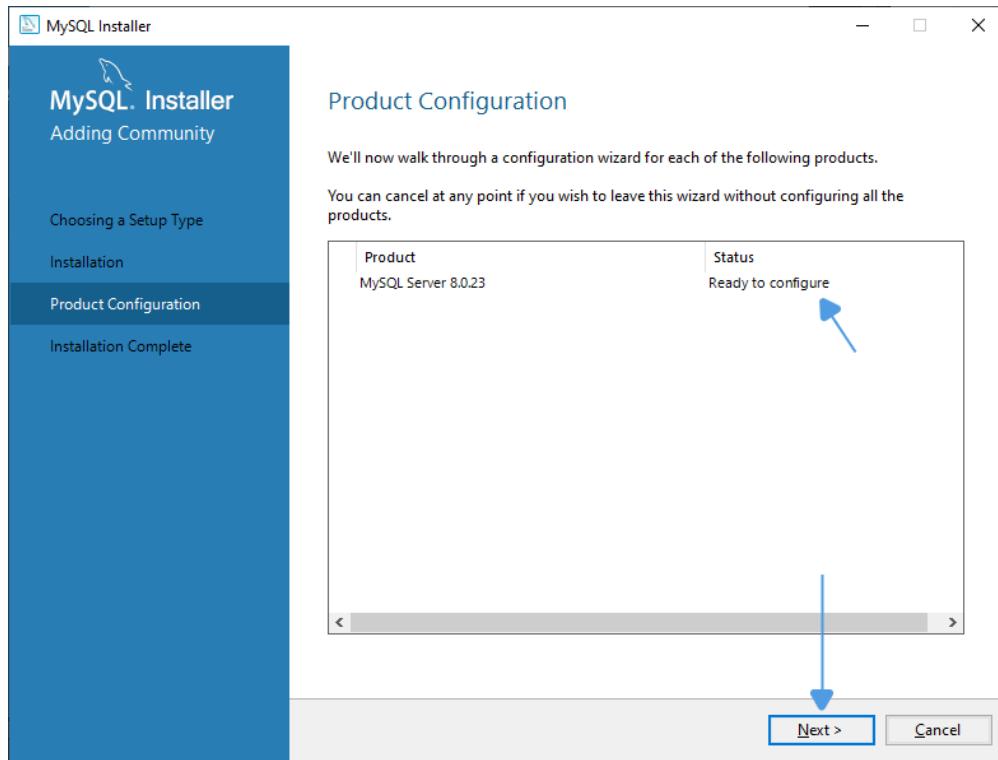
4. Cài đặt MySQL bằng trình cài đặt MySQL

Trong bước này, trình hướng dẫn tải xuống tất cả các sản phẩm MySQL đã chọn. Ngay khi trạng thái cài đặt được chỉ định là Hoàn tất, hãy nhấp vào **Tiếp theo** và tiến hành cấu hình cơ sở dữ liệu MySQL:



5. Cấu hình MySQL Server trên Windows

Bây giờ chúng ta đã đến phần cấu hình MySQL Server. Bắt đầu quá trình bằng cách nhập vào ****Next**** .

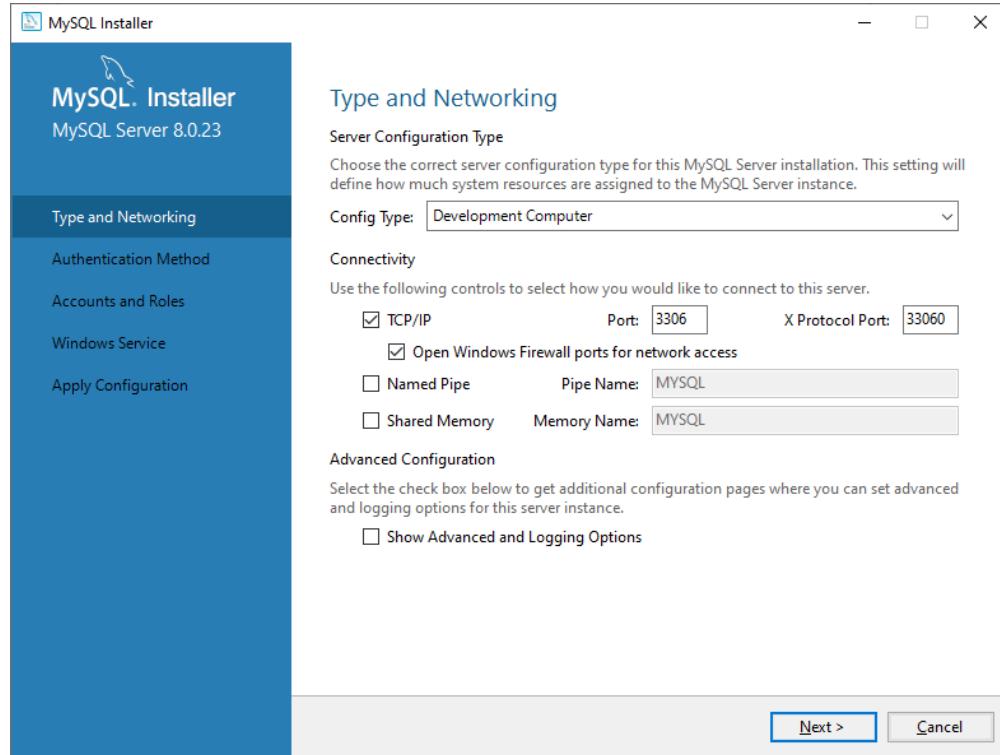


6. Loại và Mạng

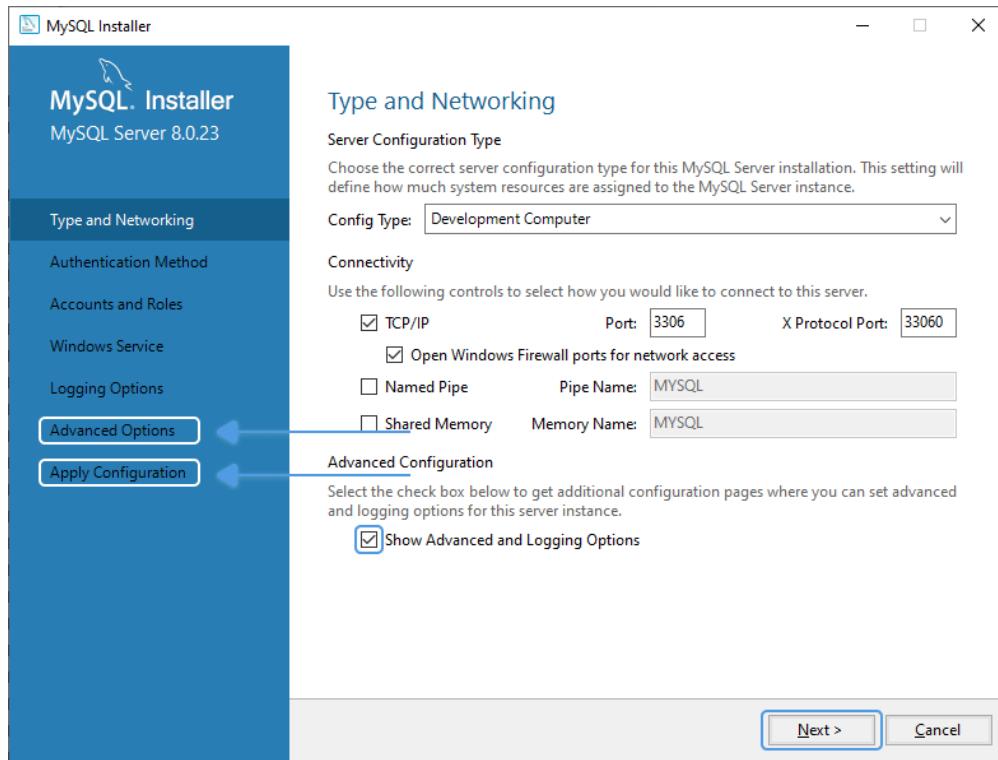
Trong phần Type and Networking, bạn có thể chọn giữa ba loại cấu hình máy chủ: Development Computer, Server Computer và Dedicated Computer để xác định phân bộ nhớ hệ thống sẽ được phân bổ cho phiên bản máy chủ MySQL của bạn. Nếu máy tính của bạn lưu trữ nhiều ứng dụng khác, hãy chọn Development Computer và nếu không có ứng dụng chính nào khác, hãy chọn loại Dedicated.

Chọn tùy chọn Connectivity đáp ứng nhu cầu của bạn. Bạn cũng có thể cấu hình tùy chỉnh ghi nhật ký và tùy chọn nâng cao trong các

bước sau nếu bạn chọn hộp tương ứng. Sau khi hoàn tất cấu hình, hãy nhấp vào **Next** :



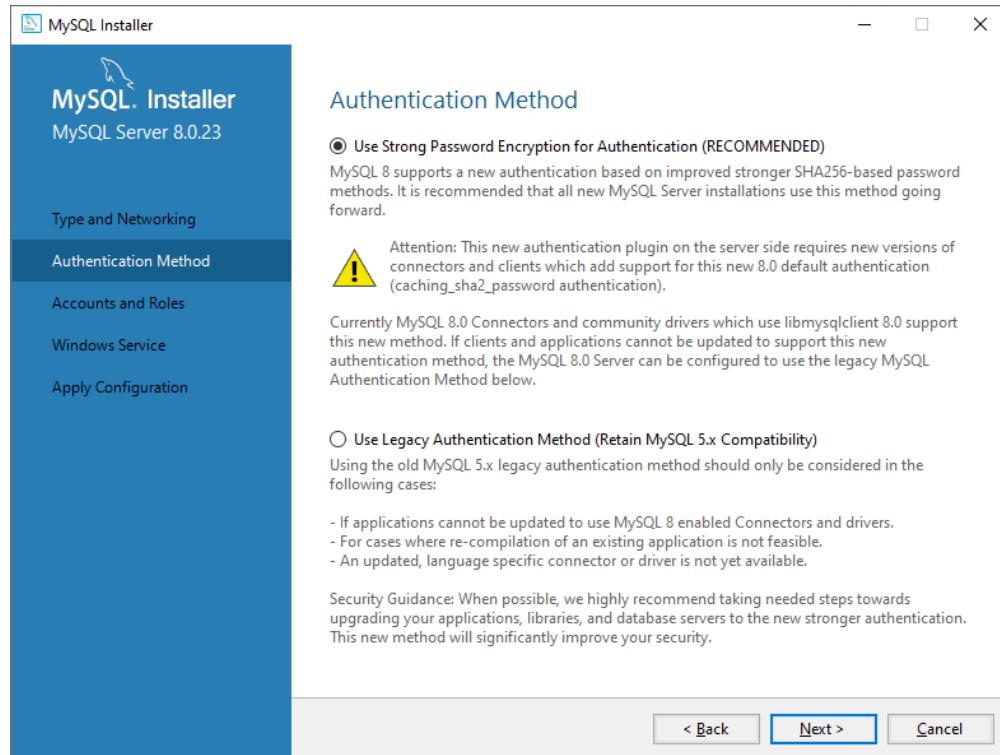
7. Chọn tùy chọn Kết nối đáp ứng nhu cầu của bạn. Bạn cũng có thể cấu hình tùy chọn ghi nhật ký tùy chỉnh và tùy chọn nâng cao trong các bước sau nếu bạn chọn hộp tương ứng. Sau khi hoàn tất cấu hình, hãy nhấp vào **Tiếp theo** :



8. Phương pháp xác thực

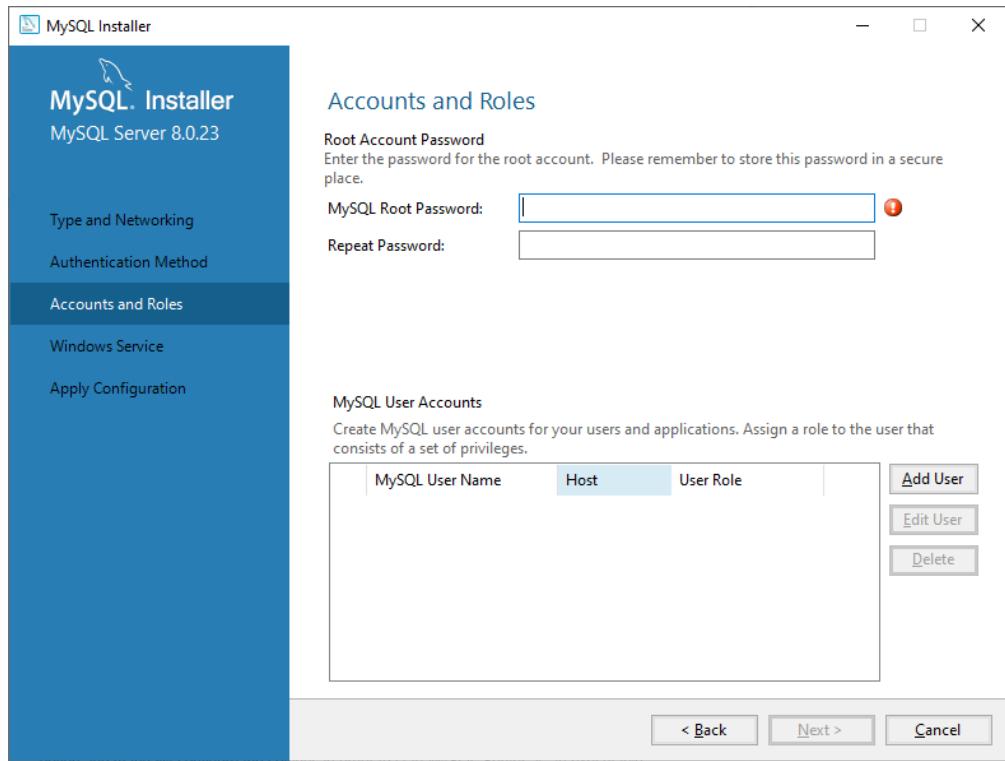
Trong bước này, bạn sẽ được lựa chọn giữa hai phương pháp xác thực phía máy chủ: Mã hóa mật khẩu mạnh và Xác thực cũ. Chọn

tùy chọn được đề xuất là **Sử dụng xác thực mật khẩu mạnh và nhập vào Tiếp theo:**



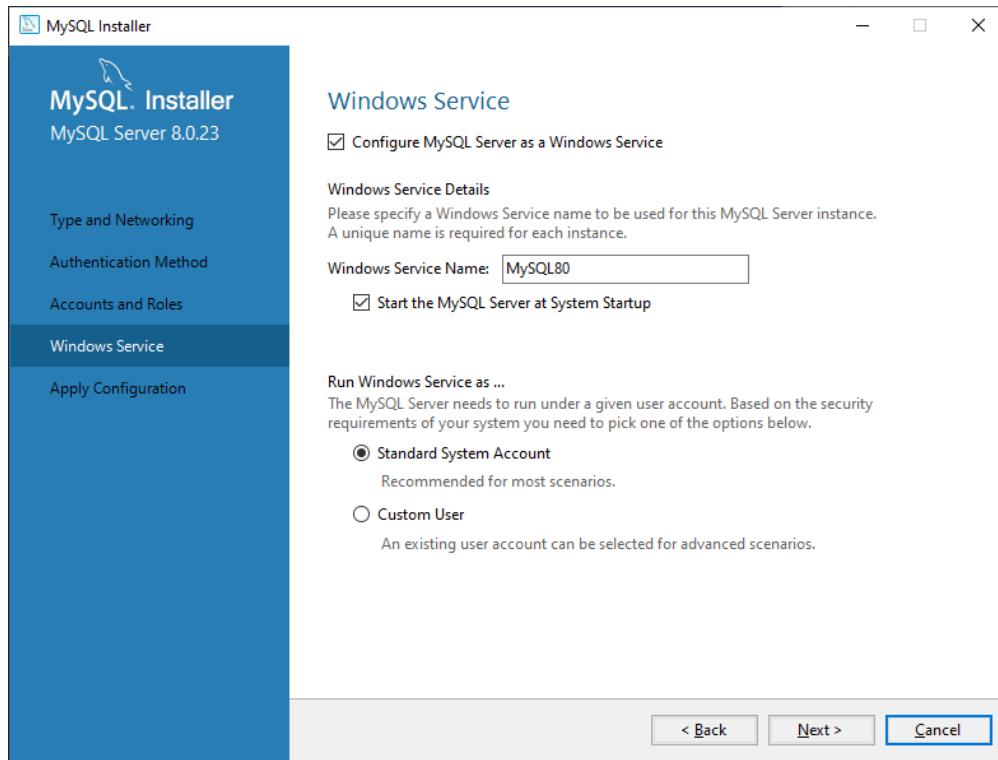
9. Tài khoản và vai trò

Tại đây, bạn sẽ cần cung cấp mật khẩu mạnh cho người dùng root MySQL của mình. Ngoài ra, hãy đảm bảo bạn lưu mật khẩu ở nơi an toàn để sử dụng sau. Tùy chọn, bạn có thể tạo thêm tài khoản người dùng MySQL với các vai trò được xác định trước. Nhập vào ****Tiếp theo**** ngay khi bạn hoàn tất:



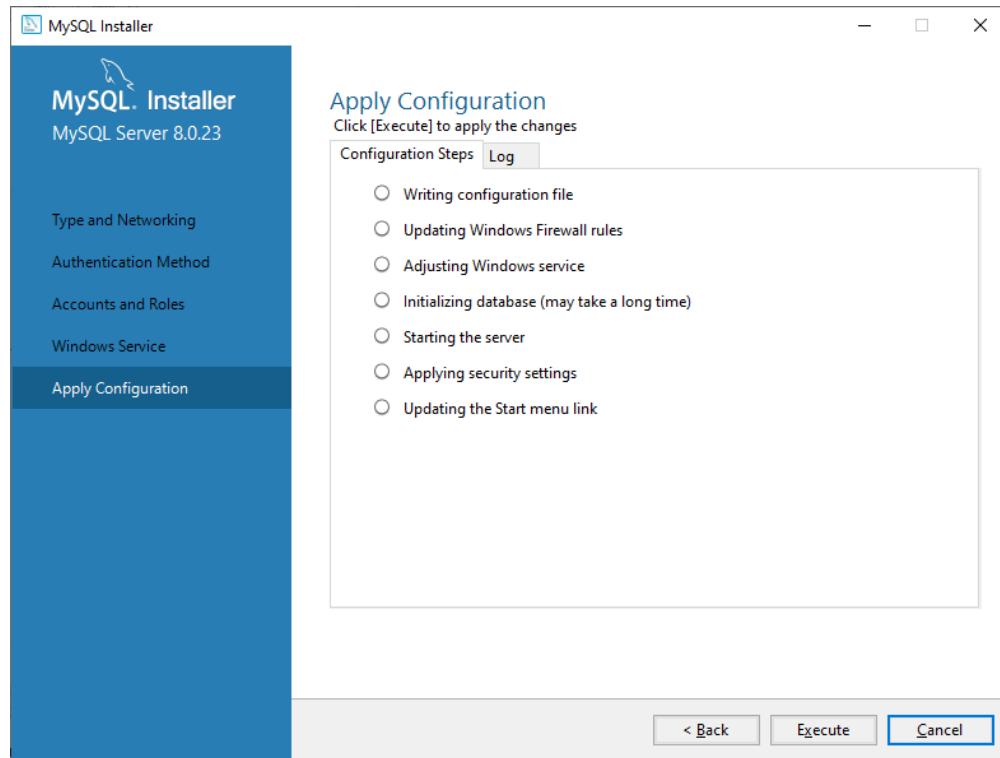
10. Dịch vụ Windows

Trong bước này, bạn có thể định nghĩa MySQL Server là một Dịch vụ Windows và thiết lập để tự động khởi động khi Windows khởi động. Ngoài ra, bạn có thể chọn tùy chọn Người dùng tùy chỉnh và cấu hình thủ công các thiết lập để khởi động MySQL Server như một chương trình thực thi.

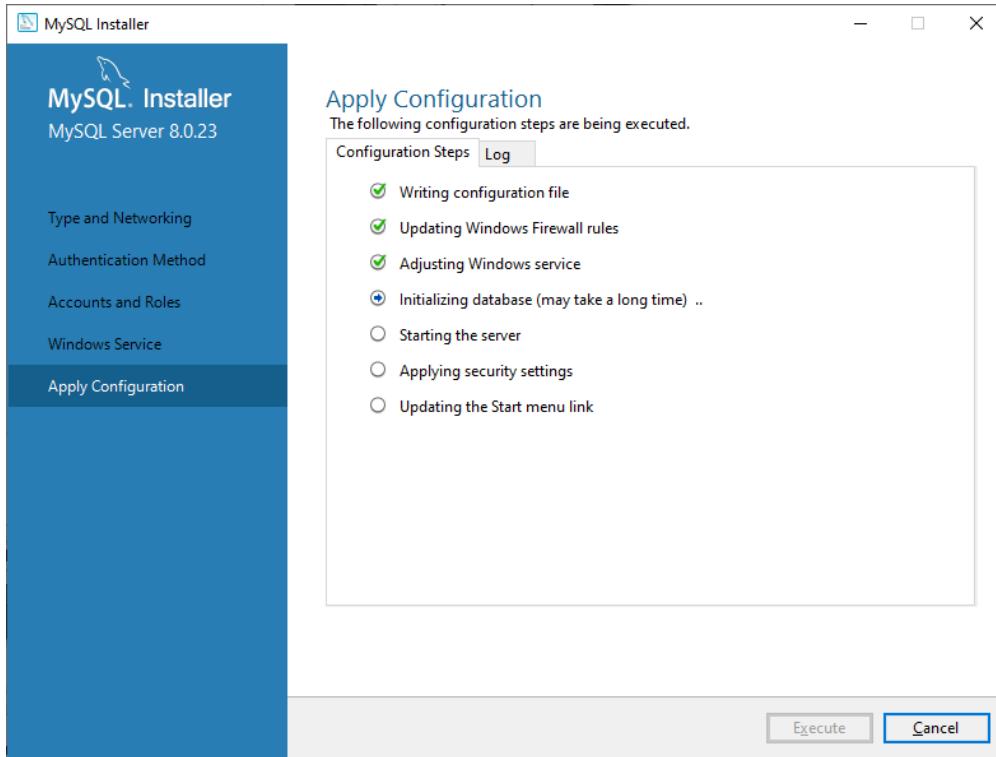


11. Áp dụng cấu hình

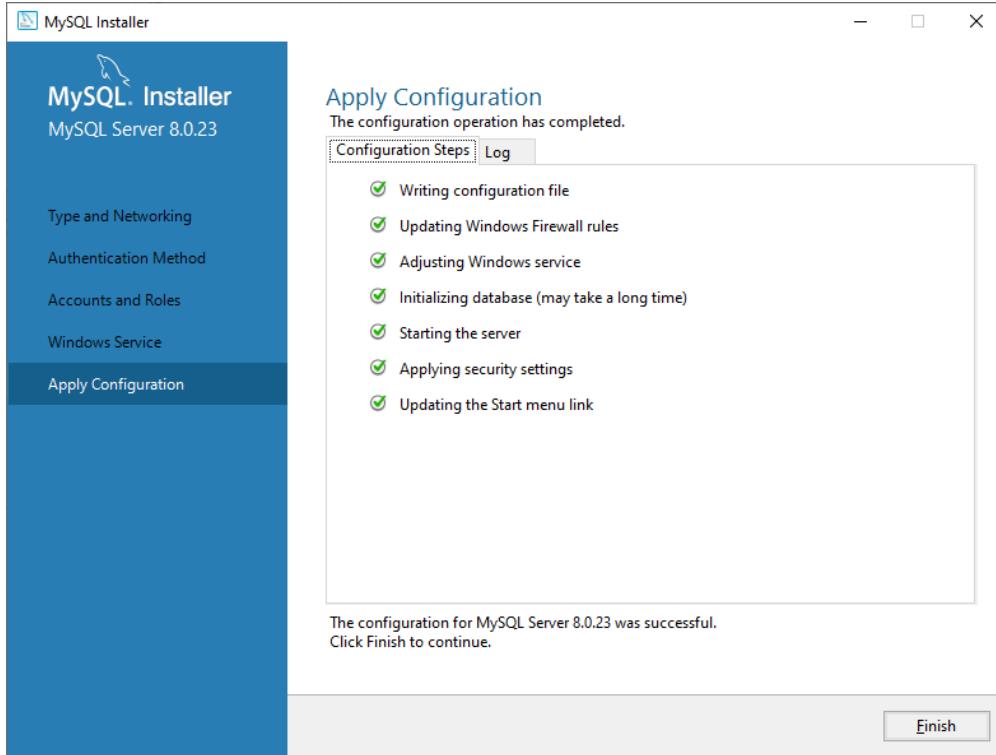
Bước cuối cùng này cung cấp tổng quan về các bước cấu hình:



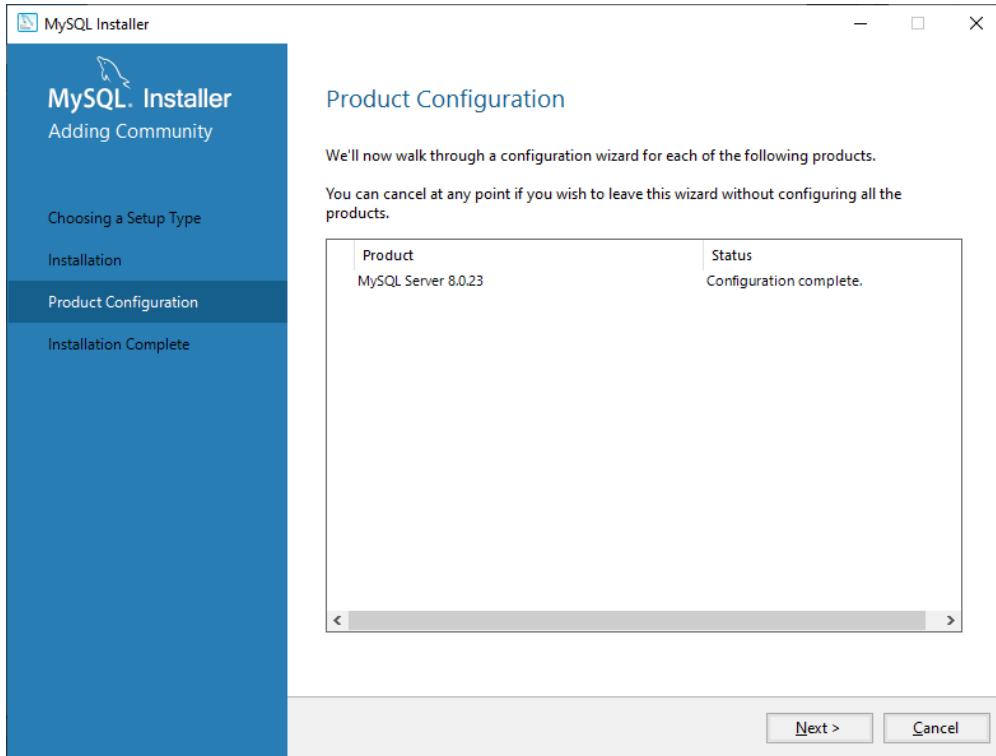
12. Bằng cách nhấp vào **Thực thi** , bạn sẽ thấy các thiết lập cấu hình được áp dụng từng cái một:



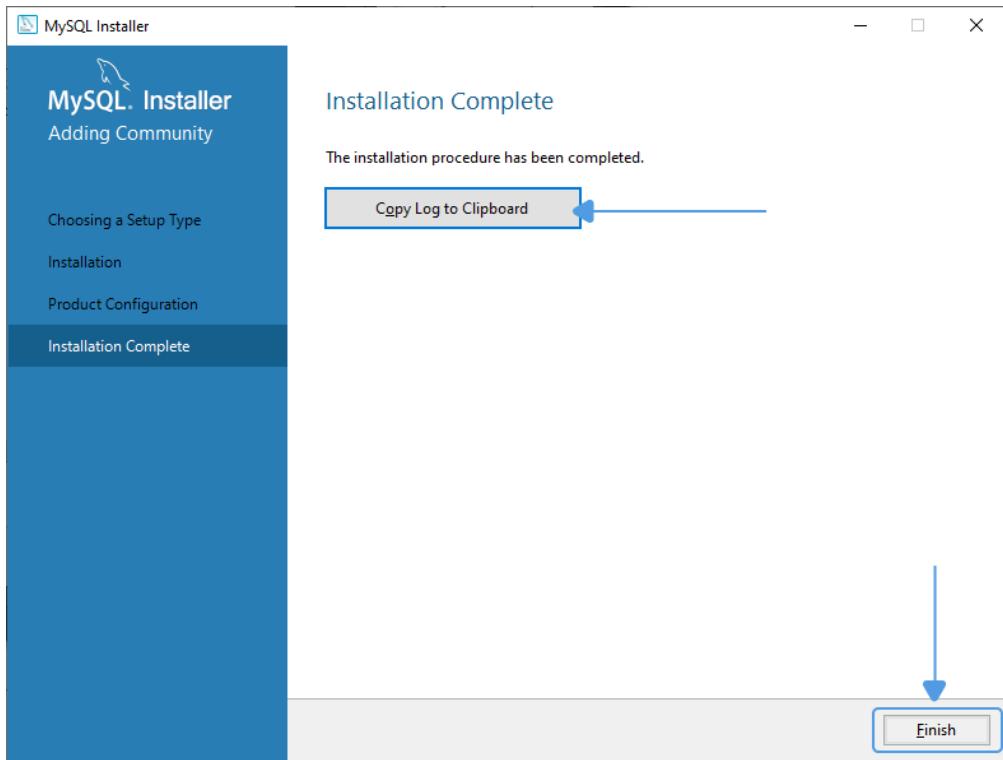
13. Các dấu kiểm màu xanh lá cây thông báo quá trình cấu hình đã hoàn tất thành công. Nhấp vào **Kết thúc** :



14. Cấu hình hoàn tất, nhấp vào **Tiếp theo** :



15. Xin chúc mừng! Bạn đã hoàn tất cài đặt MySQL Server. Bây giờ bạn có thể sao chép nhật ký quá trình cài đặt vào Windows Clipboard và nhấp vào **Finish** :



2.2 Cài đặt MySQL trên macOS

Tải xuống phiên bản mới nhất của MySQL cho Mac

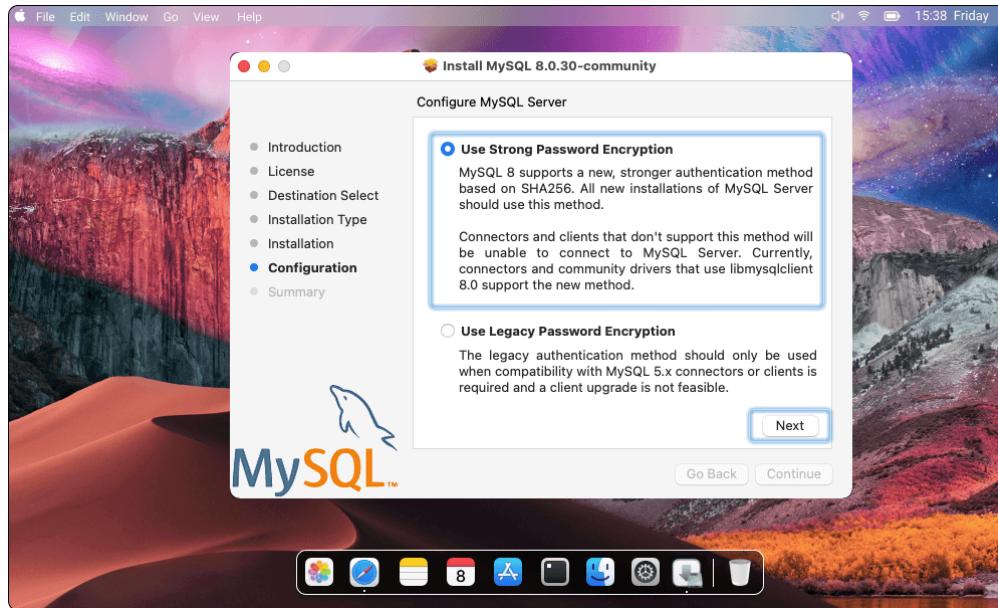
- Bạn có thể tải MySQL cho Mac từ [trang web chính thức của MySQL](#). Vì có nhiều tùy chọn, hướng dẫn sau đây có thể hữu ích và giúp bạn tìm thấy những gì bạn cần nhanh hơn.
- Bước 1: Trong phần đầu trang web MySQL, hãy chuyển đến tab **Tải xuống**.
- Bước 2: Cuộn xuống và chọn **Tải xuống cộng đồng MySQL (GPL)**.
- Bước 3: Chọn **MySQL Community Server**.
- Bước 4: Trong menu thả xuống **Chọn hệ điều hành**, chọn **macOS**.
- Bước 5: Tìm phiên bản bộ xử lý cần thiết và chọn **Tải xuống**.

- Bước 6: Sau đó, bạn sẽ được nhắc đăng ký hoặc đăng nhập vào tài khoản Oracle Web của mình. Ngay bên dưới các nút, hãy chọn **Không, cảm ơn, chỉ cần bắt đầu tải xuống của tôi**. Quá trình tải xuống sẽ bắt đầu.



Cách cài đặt MySQL trên macOS

- Bước 1: Nhấp đúp vào tệp DMG đã tải xuống để mở trình cài đặt giống như trình hướng dẫn. Trình cài đặt sẽ cho bạn biết phải xác định xem phần mềm có trong đó có thể được cài đặt hay không. Nhấp vào **Cho phép**. Sau đó, bạn sẽ thấy các liên kết đến các tài nguyên liên quan đến MySQL, bao gồm cả tài liệu. Nhấp vào **Tiếp tục**.
- Bước 2: Trên trang **Giấy phép**, hãy chấp nhận Thỏa thuận cấp phép phần mềm bằng cách nhấp vào **Tiếp tục**.
- Bước 3: Tiếp theo, bạn cần chọn đích tải xuống. Theo mặc định, đó là ổ cứng chính của bạn. Nếu bạn muốn thay đổi, hãy nhấp vào **Thay đổi vị trí cài đặt**. Nếu không, hãy nhấp vào **Cài đặt**.
- Bước 4: Nhập mật khẩu, nhấp vào **Cài đặt phần mềm** và đợi trong khi các tệp được cài đặt trên máy Mac của bạn
- Bước 5: Trên trang **Cấu hình**, chọn **Sử dụng Mã hóa mật khẩu mạnh**. Nhấp vào **Tiếp theo**. Nhập mật khẩu gốc MySQL của bạn, nhấp vào **Kết thúc** và quá trình cài đặt sẽ hoàn tất



Cách tải xuống và cài đặt MySQL Server bằng Homebrew và Terminal

Ngoài ra, bạn có thể sử dụng trình quản lý gói Homebrew để tải xuống và cài đặt MySQL trên máy Mac của mình từ Terminal, CLI có sẵn trong macOS.

1. Nếu bạn chưa cài đặt Homebrew trên máy Mac, hãy mở Terminal và thực hiện lệnh sau:

```
/bin/bash -c "$(curl -fsSL 
```

Sau khi lệnh này được thực thi, bạn sẽ có các lệnh như `brew install`, `brew upgrade` và `brew uninstall` có sẵn trong Terminal.

2. Và như vậy, để cài đặt MySQL, bạn cần chạy lệnh sau:

- `brew install mysql`
3. Quá trình cài đặt sẽ mất một lúc. Sau khi hoàn tất, bạn có thể khởi động máy chủ MySQL của mình bằng lệnh sau:
- `brew services start mysql`
4. Để bảo mật MySQL của bạn bằng mật khẩu root, hãy chạy lệnh sau:
- `mysql_secure_installation`

Cách cấu hình MySQL từ dòng lệnh

- MySQL có thể được cấu hình thêm từ Terminal. Ví dụ, nó cho phép bạn quản lý người dùng. Để tạo người dùng cơ sở dữ liệu mới từ Terminal và cấp tất cả các quyền trên tất cả các cơ sở dữ liệu, hãy nhập lệnh sau, thay thế "username" bằng người dùng bạn muốn tạo và thay thế "password" bằng mật khẩu của người dùng.

```
GRANT ALL PRIVILEGES ON . TO 'username'@'localhost'  
IDENTIFIED BY 'password';
```

- Bạn cũng có thể cần cấp các quyền cụ thể. Ví dụ, sử dụng lệnh sau để cấp rõ ràng quyền SELECT cho người dùng.

```
GRANT SELECT ON . TO 'username'@'localhost'
```

- Nếu bạn muốn thu hẹp quyền truy cập của người dùng vào một cơ sở dữ liệu cụ thể, hãy nhập lệnh sau và thay thế "cơ sở dữ liệu" bằng tên cơ sở dữ liệu của bạn.

```
GRANT ALL PRIVILEGES ON database.* TO 'username'@'localhost';
```

- Để cho phép truy cập từ xa vào MySQL, chúng tôi đề xuất tạo một người dùng có quyền truy cập từ một địa chỉ IP cụ thể ('username'@'192.168.1.100') hoặc từ bất kỳ máy chủ nào ('username'@'%').

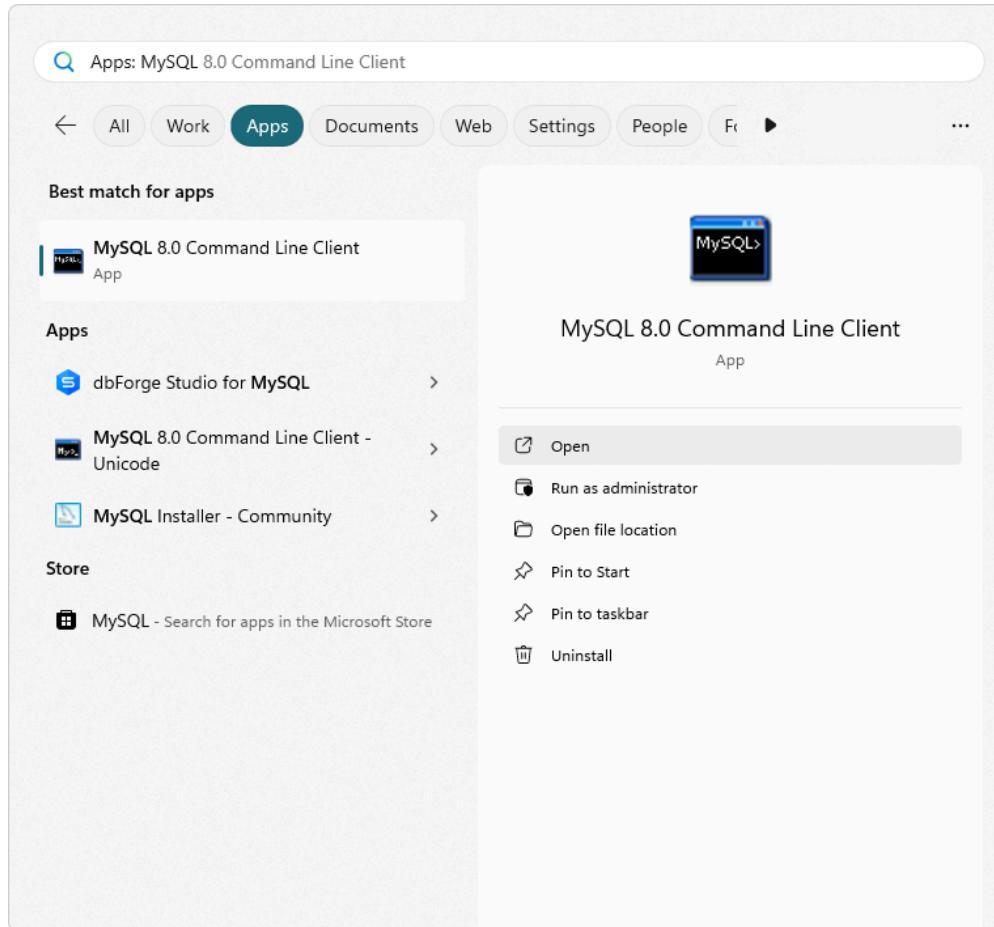
2.3 Kết nối với máy chủ MySQL

Cách kết nối với MySQL bằng Command-Line Client

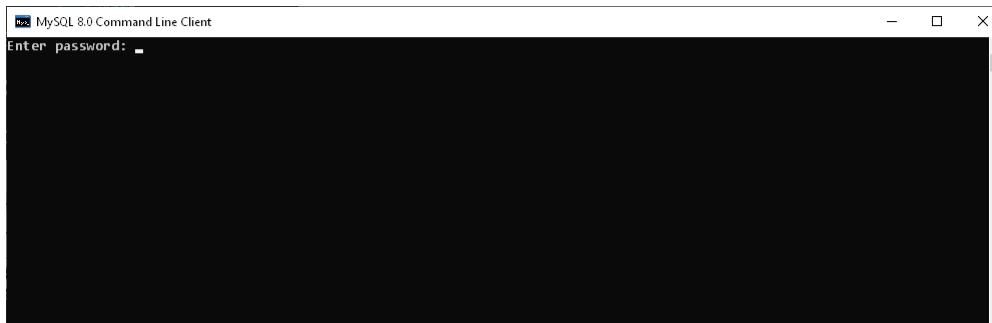
Trong bài viết đầu tiên của loạt bài này, chúng tôi đã cung cấp hướng dẫn chi tiết phác thảo các khía cạnh về cách cài đặt MySQL Server trên Windows. Trong hướng dẫn này, chúng tôi xây dựng trên máy chủ MySQL của bạn đang hoạt động.

[MySQL Command-Line Client](#) là tiện ích CLI mặc định đi kèm với mọi cài đặt MySQL. Giải pháp này cho phép bạn thực hiện tất cả các hoạt động chuẩn như kết nối với cơ sở dữ liệu, tạo, chỉnh sửa và xóa cơ sở dữ liệu và bảng, truy xuất và lọc dữ liệu, v.v.

Theo mặc định, MySQL Command-Line Client được cài đặt cùng với MySQL Server. Để kiểm tra xem bạn có cài đặt nó trên máy không, hãy tìm kiếm nó trong phần **Apps** (chúng tôi đang sử dụng Windows 11, nếu bạn có phiên bản Windows khác, hãy tìm kiếm tiện ích này bằng các phương pháp chuẩn):

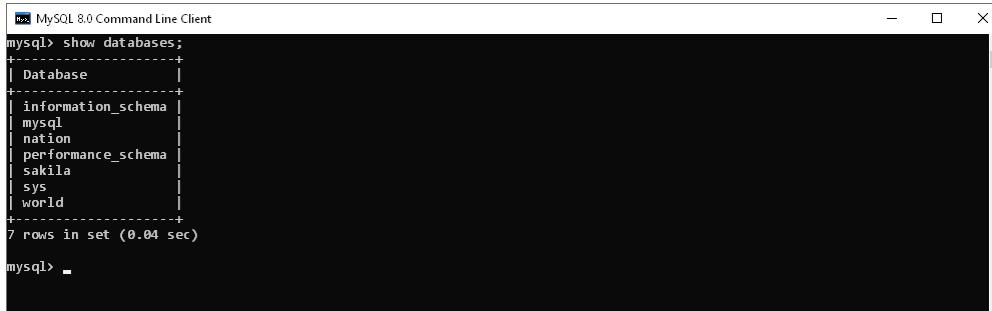


Khởi chạy MySQL Command-Line Client. Bạn sẽ được yêu cầu nhập mật khẩu người dùng root mà bạn đã đặt trong quá trình cài đặt MySQL. Cần phải kết nối với máy chủ MySQL của bạn.

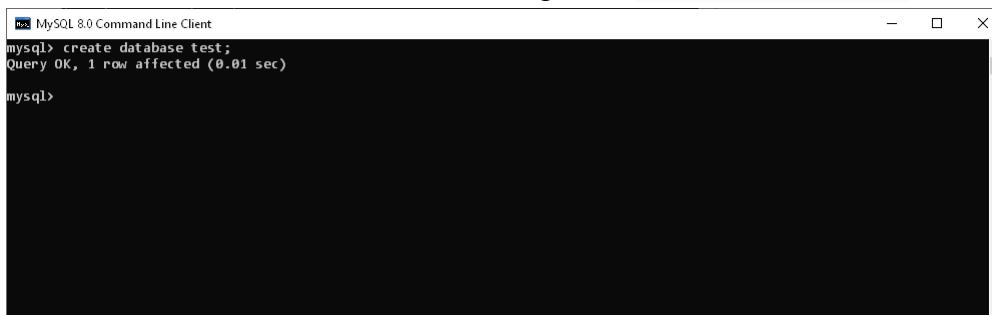


Sau khi kết nối thành công với máy chủ MySQL, bạn có thể thực hiện lệnh.

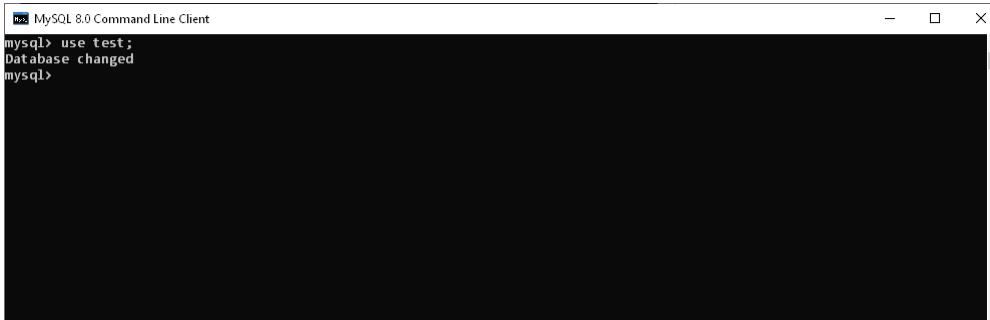
Giả sử bạn muốn [kiểm tra danh sách cơ sở dữ liệu](#) trên máy chủ. Thực hiện SHOW DATABASES lệnh:



Bạn có thể [tạo cơ sở dữ liệu mới](#) bằng lệnh CREATE DATABASE :

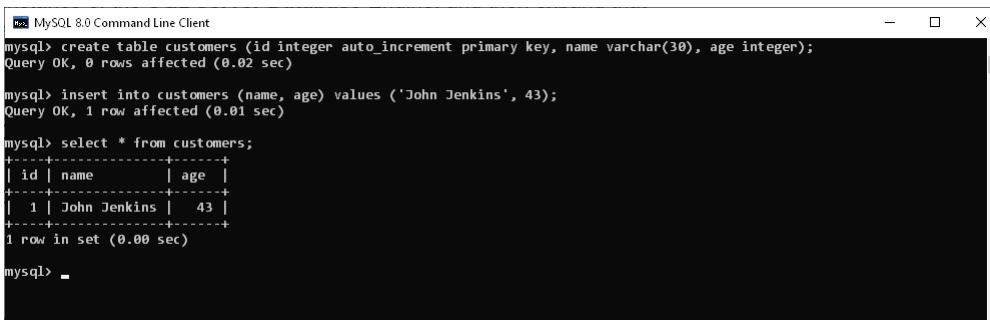


Để kết nối với một cơ sở dữ liệu MySQL cụ thể và làm việc với cơ sở dữ liệu đó, hãy thực hiện USE lệnh cơ sở dữ liệu và chỉ định tên cơ sở dữ liệu bạn muốn truy cập:



```
MySQL 8.0 Command Line Client
mysql> use test;
Database changed
mysql>
```

Bạn có thể [tạo một bảng mới](#) và sau đó điền dữ liệu vào đó bằng lệnh `CREATE TABLE` và `INSERT INTO` :



```
MySQL 8.0 Command Line Client
mysql> create table customers (id integer auto_increment primary key, name varchar(30), age integer);
Query OK, 0 rows affected (0.02 sec)

mysql> insert into customers (name, age) values ('John Jenkins', 43);
Query OK, 1 row affected (0.01 sec)

mysql> select * from customers;
+----+-----+-----+
| id | name  | age  |
+----+-----+-----+
|  1 | John Jenkins | 43 |
+----+-----+-----+
1 row in set (0.00 sec)

mysql> -
```

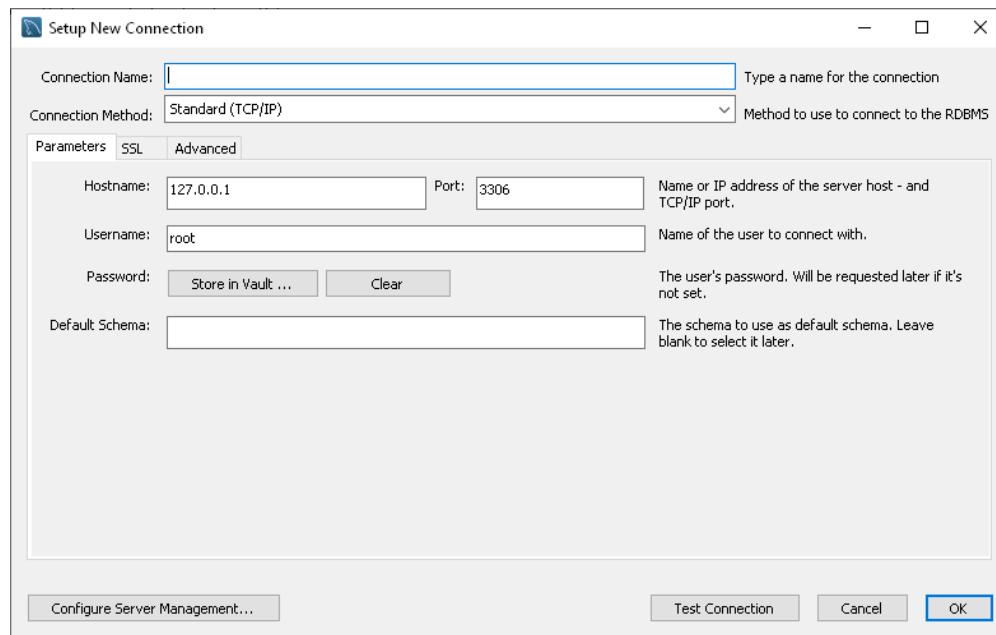
Cuối cùng, khi tất cả các tác vụ bạn muốn thực hiện trong phiên này đã hoàn tất, hãy nhập `QUIT` và nhấp `Enter` để thoát khỏi máy khách MySQL.

Cách kết nối bằng MySQL Workbench

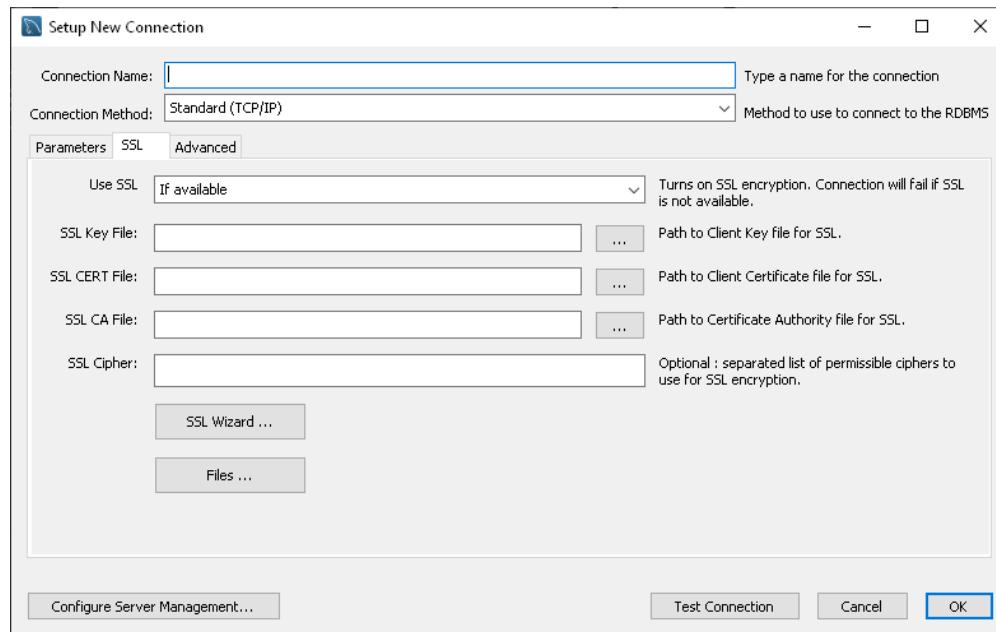
MySQL Workbench là một công cụ trực quan phổ biến dành cho các kiến trúc sư cơ sở dữ liệu, nhà phát triển và DBA. Đây là IDE mặc định cho MySQL, một công cụ miễn phí và có chức năng cao, mặc dù [chức năng của nó không mạnh mẽ](#) như trong dbForge Studio.

Để truy cập MySQL Server bằng Workbench:

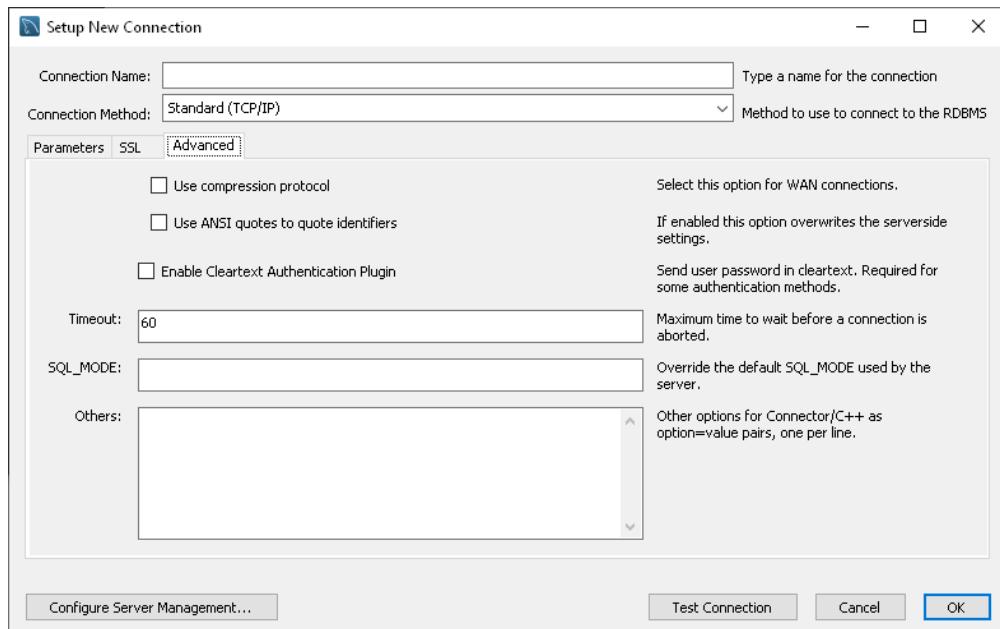
1. Chạy MySQL Workbench. Trên menu **Database** , nhấp vào **Connect to Database**.
Hoặc, nhấp vào biểu tượng dấu cộng bên cạnh nhãn **MySQL Connections** .
2. Trong cửa sổ **Thiết lập kết nối mới** , hãy chỉ định **Tên kết nối** và cung cấp tên máy chủ, cổng và tên người dùng.



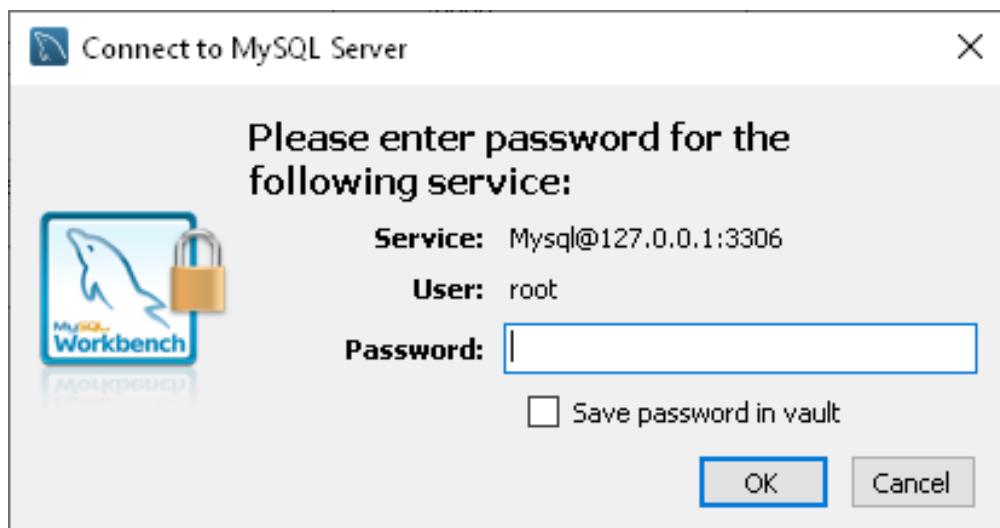
3. (Tùy chọn) Di tới tab **SSL** để cấu hình cài đặt kết nối SSL.



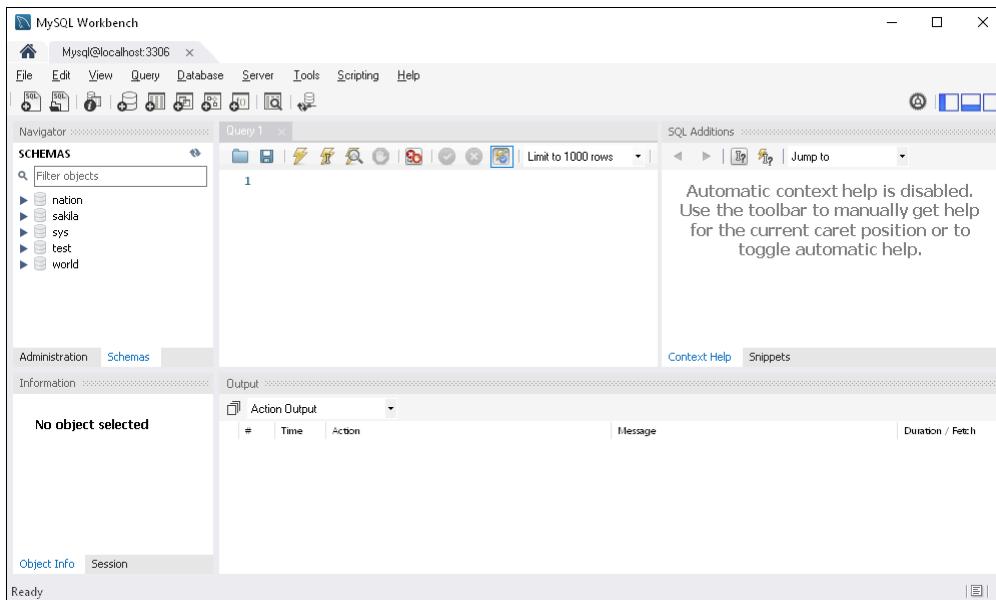
4. (Tùy chọn) Đi tới tab **Nâng cao** để cấu hình cài đặt kết nối nâng cao.



5. Bạn có thể nhấp vào **Kiểm tra kết nối** để kiểm tra các thông số bạn đã nhập. Trong trường hợp bạn chắc chắn rằng tất cả thông tin đăng nhập đều đúng, hãy nhấp vào **OK**. Nhập mật khẩu.



Sau khi kết nối, bạn sẽ thấy danh sách cơ sở dữ liệu ở bên trái.

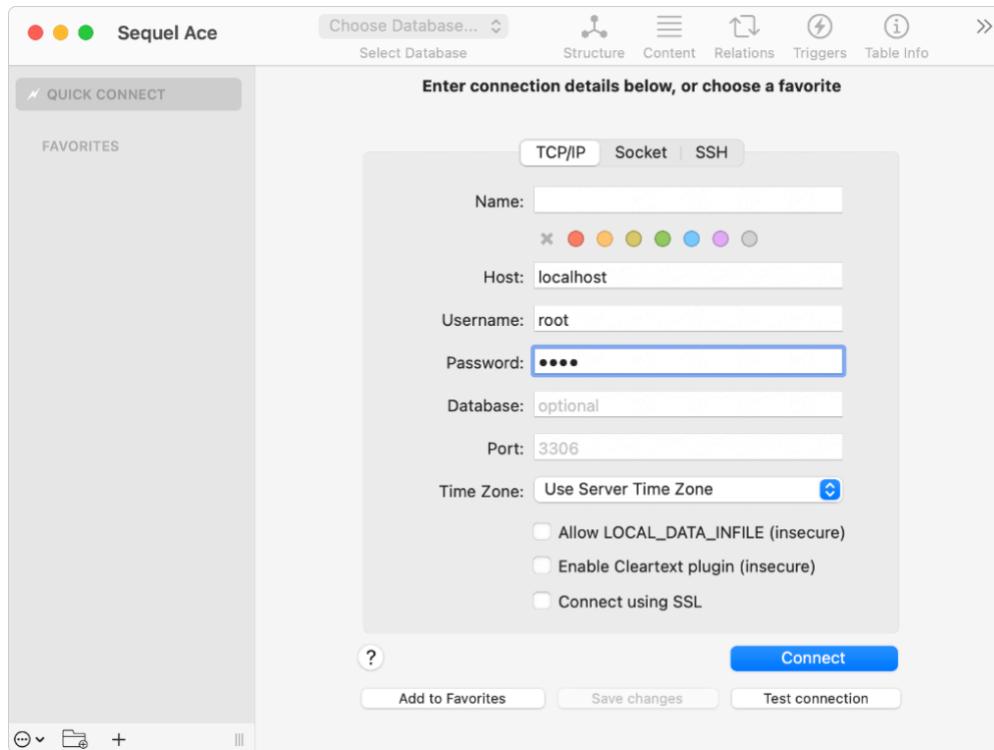


Cách kết nối bằng Sequel Ace

Sequel Ace là một công cụ GUI nguồn mở được sử dụng rộng rãi, được thiết kế để quản lý cơ sở dữ liệu MySQL và MariaDB trên macOS, được các chuyên gia MySQL đánh giá cao vì chức năng và tính dễ sử dụng.

Để kết nối với MySQL bằng Sequel Ace, hãy làm theo các bước sau:

1. Khởi chạy Sequel Ace và nhấp vào **Kết nối nhanh** .
2. Trong cửa sổ hộp thoại kết nối, chọn **TCP/IP** để thiết lập kết nối chuẩn.
3. Cung cấp tên cho kết nối và nhập thông tin đăng nhập sau: **Máy chủ** , **Tên người dùng** , **Mật khẩu** (nếu có), **Cơ sở dữ liệu** (tùy chọn), **Cổng** .
4. Nhấp vào **Kiểm tra kết nối** để xác minh thông tin chi tiết hoặc nhấp vào **Kết nối** để tiếp tục trực tiếp.



2.4 Kết nối cổng MySQL

Cách kết nối đến cổng MySQL từ dòng lệnh

Các tham số kết nối chính xác, chẳng hạn như tên máy chủ được gán cho máy tính của bạn, tên người dùng và mật khẩu liên kết với tài khoản MySQL của bạn, phải được sử dụng trong phần mềm máy khách để kết nối với máy chủ MySQL. Có một giá trị mặc định cho mỗi tham số kết nối, nhưng bạn có thể thay đổi chúng theo nhu cầu của mình bằng cách sử dụng các tùy chọn chương trình được cung cấp từ dòng lệnh hoặc trong tệp tùy chọn.

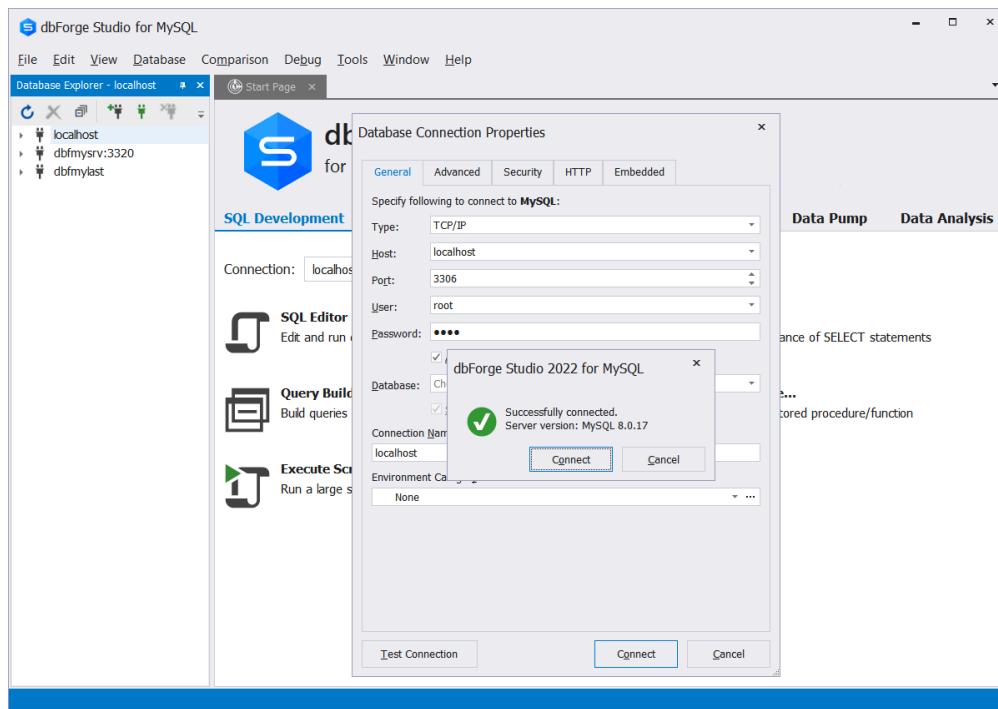
Không cung cấp bất kỳ tùy chọn kết nối cụ thể nào, lệnh sau sẽ khởi động MySQL: `mysql`

Các tham số không được chỉ định, do đó các giá trị mặc định được sử dụng:

- Localhost được sử dụng làm tên máy chủ mặc định.
- Trên Windows, tên người dùng mặc định là ODBC và trên Unix, đó là tên đăng nhập của bạn.
- Vì cả --password và -p đều không được sử dụng nên không có mật khẩu nào được cung cấp.
- Tên cơ sở dữ liệu mặc định cho MySQL được lấy từ tham số không phải tùy chọn đầu tiên. MySQL không chọn cơ sở dữ liệu mặc định vì không có đối số nào như vậy.

Xác định các tham số có liên quan trên dòng lệnh để chỉ rõ tên máy chủ, tên người dùng và mật khẩu. Bạn có thể sử dụng tùy chọn database-name để chọn cơ sở dữ liệu mặc định.

Kết nối và cấu hình cơ sở dữ liệu chưa bao giờ dễ dàng hơn với dbForge Studio for MySQL! Chỉ cần điền các thuộc tính kết nối cơ sở dữ liệu như trong video và tiết kiệm thời gian với trình hướng dẫn nhập và xuất. Tìm hiểu cách kết nối với Cơ sở dữ liệu MySQL bằng GUI MySQL đầy đủ này:



Cổng mặc định cho MySQL Server là gì?

Số cổng MySQL mặc định cho cơ sở dữ liệu của bạn là 3306. Mặc dù hầu hết các chương trình sẽ chỉ định số cổng theo mặc định, nhưng nó có thể thay đổi trong một số trường hợp. Vì lý do bảo mật, số cổng có khả năng bị thay đổi. Hơn nữa, nó cũng có thể bị thay đổi nếu số cổng mặc định đã được một chương trình khác sử dụng.

Giao thức MySQL kết nối với các tiện ích như mysqldump và [phần mềm máy khách](#) MySQL bằng cổng mặc định.

Có an toàn khi sử dụng Cổng MySQL mặc định 3306 không?

Máy chủ MySQL dễ bị tấn công khi cổng mặc định 3306 bị lộ. Nếu [người dùng muốn truy cập cơ sở dữ liệu từ xa](#), họ phải tìm kiếm các tùy chọn an toàn khác.

Nên cân nhắc sử dụng đường hầm SSH thay vì mở cổng MySQL 3306. Giải pháp thay thế khác là giới hạn các địa chỉ IP có thể truy cập vào cổng để ngăn chặn các máy chủ đáng ngờ kết nối. Mặc dù cổng mặc định là 3306, MySQL không phải lúc nào cũng sử dụng cổng này.

Bất kỳ phần mềm máy khách nào cố gắng kết nối với máy chủ đều phải có `-port=portNumber` tùy chọn được chỉ định nếu máy chủ MySQL của bạn đang lắng nghe trên cổng khác ngoài 3306.

Các loại cổng MySQL

Các tính năng của MySQL hỗ trợ nhiều loại cổng phục vụ cho nhiều mục đích khác nhau. Chúng ta hãy cùng xem xét kỹ hơn từng loại cổng đó:

Cổng kết nối giữa máy khách và máy chủ

Máy khách MySQL, các tiện ích như mysqldump và các trình kết nối MySQL đều sử dụng cổng 3306 làm cổng mặc định. Đây cũng là cổng chuẩn của giao thức MySQL. Các máy khách như MySQL router, MySQL Shell và MySQL connectors hỗ trợ giao thức này.

Cổng kết nối MySQL cho quản trị

Đối với [các kết nối quản trị](#), máy chủ MySQL hỗ trợ cài đặt cổng TCP/IP. Cổng này mở rộng các lựa chọn kết nối quản trị có sẵn trên giao diện mạng cho các kết nối thông thường.

Cổng Shell cho MySQL

MySQL Shell là trình soạn thảo mã và máy khách dành cho người dùng có kinh nghiệm. MySQL Shell hỗ trợ cả MySQL thông thường và X Protocol.

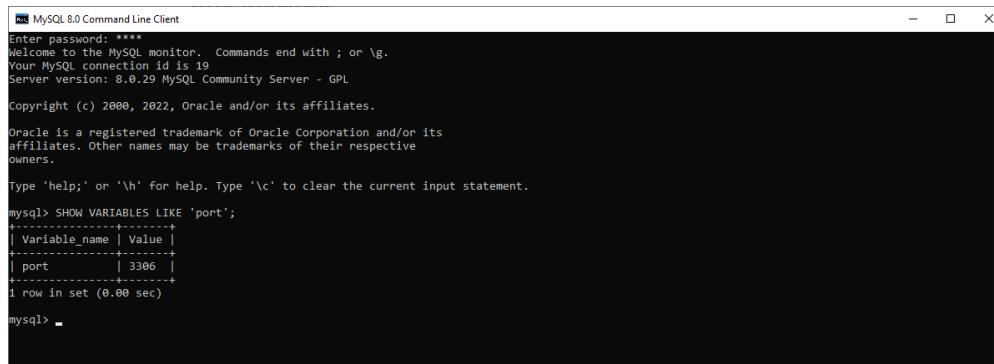
Ngoài ra còn có các loại như cổng có tính khả dụng cao, quản lý khóa, Giao thức Memcached và xác thực bên ngoài.

MySQL sử dụng cổng nào?

Cổng chủ yếu được kiểm soát bởi các thành phần đang sử dụng, ứng dụng nào đã được bật, cách chúng kết nối và các đặc điểm chung của hệ sinh thái MySQL.

Phương pháp tốt nhất để sử dụng cổng là thiết kế cổng đủ lớn để xử lý tất cả các thành phần khác nhau đồng thời chặn mọi máy chủ không đáng tin cậy.

Có một số kỹ thuật khác nhau để tìm ra cổng mà máy chủ MySQL của bạn đang lắng nghe. Khi bạn đã kết nối với máy chủ MySQL, bạn vẫn có thể kiểm tra số cổng đã được sử dụng cho kết nối cụ thể này. Thực hiện lệnh sau để thực hiện việc đó: SHOW VARIABLES LIKE 'port';



```
MySQL 8.0 Command Line Client
Enter password: ****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 19
Server version: 8.0.29 MySQL Community Server - GPL

Copyright (c) 2000, 2022, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

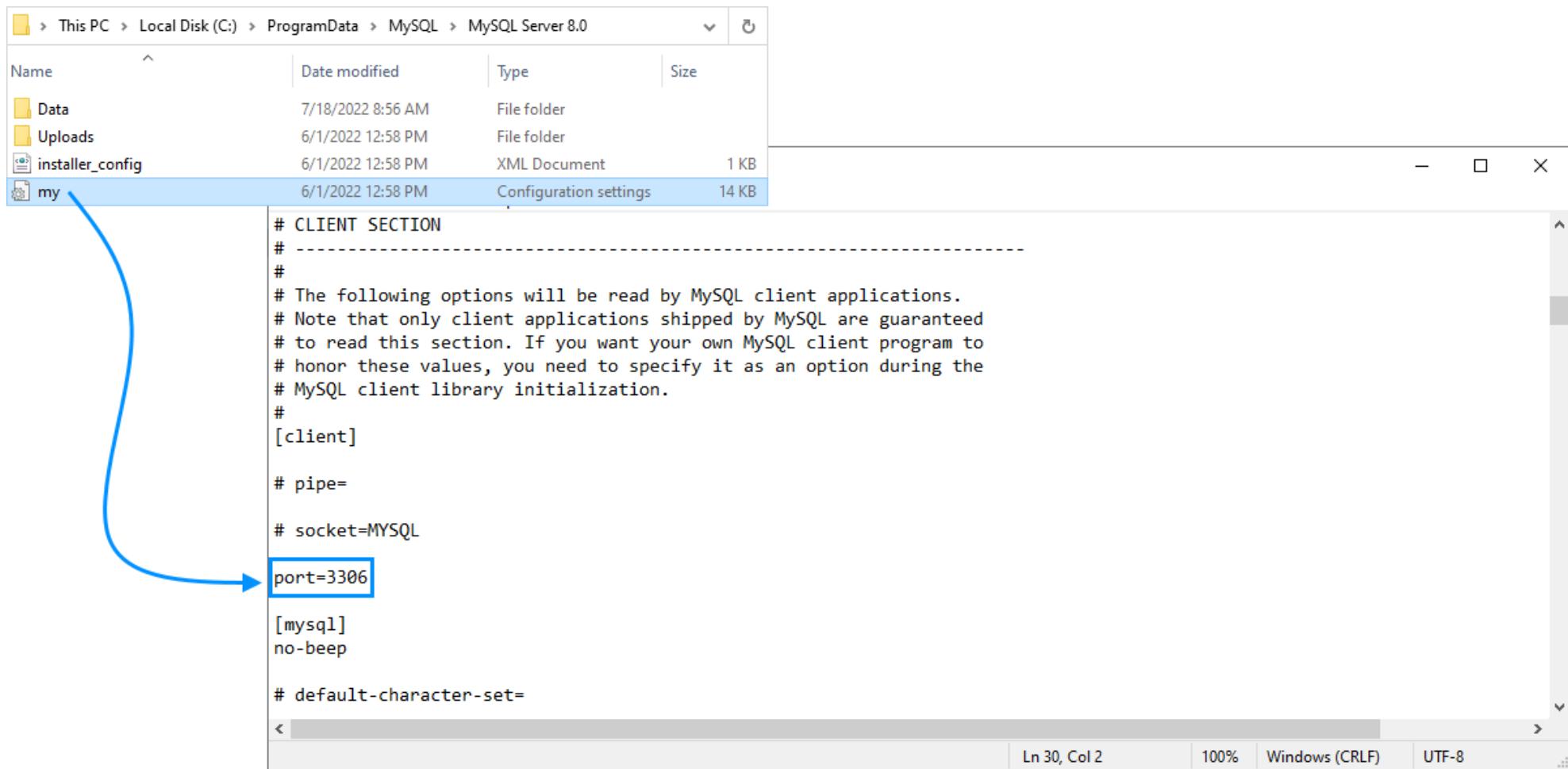
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> SHOW VARIABLES LIKE 'port';
+-----+-----+
| Variable_name | Value |
+-----+-----+
| port          | 3306  |
+-----+-----+
1 row in set (0.00 sec)

mysql> -
```

Nếu bạn đang sử dụng Windows, bạn có thể tìm số cổng MySQL trong tệp cài đặt:

1. Điều hướng đến tệp "my.ini". Tệp này sẽ nằm trong thư mục cài đặt MySQL, ví dụ: C:\ProgramData\MySQL\MySQL Server 8.0.
2. Mở tệp cấu hình .ini bằng Notepad.
3. Tìm cổng MySQL đang lắng nghe trên Windows.



Name	Date modified	Type	Size
Data	7/18/2022 8:56 AM	File folder	
Uploads	6/1/2022 12:58 PM	File folder	
installer_config	6/1/2022 12:58 PM	XML Document	1 KB
my	6/1/2022 12:58 PM	Configuration settings	14 KB

```
# CLIENT SECTION
# -----
#
# The following options will be read by MySQL client applications.
# Note that only client applications shipped by MySQL are guaranteed
# to read this section. If you want your own MySQL client program to
# honor these values, you need to specify it as an option during the
# MySQL client library initialization.
#
[client]

# pipe=

# socket=MYSQL

port=3306

[mysql]
no-beep

# default-character-set=
```

Thay đổi cổng MySQL mặc định của bạn có thể bảo vệ bạn khỏi các chương trình tấn công brute-force có được quyền truy cập không mong muốn. Mặc dù việc sử dụng các cổng không chuẩn có thể cải thiện bảo mật của bạn, nhưng nó chỉ làm như vậy bằng cách kéo dài thời gian tin tức thành công. Do đó, việc áp dụng các biện pháp bảo mật bổ sung ngoài việc thay đổi cổng sẽ có lợi. Một lý do khác khiến bạn có thể cần thay đổi cổng là nếu cổng 3306 thông thường đã được sử dụng.

Các cổng MySQL có nhiều cách sử dụng khác nhau

Cổng 3306 (TCP)

Máy khách MySQL kết nối với máy chủ MySQL thông qua cổng 3306 theo mặc định. Theo quy tắc, giao tiếp trên cổng này được mã hóa. Trừ khi giao thức X được sử dụng, giao tiếp trên cổng này phải theo hướng từ máy khách đến máy chủ.

Cổng 33060 (TCP)

Cổng này được sử dụng để giao tiếp giữa máy khách MySQL và máy chủ và nó cũng được mã hóa. Trừ khi sử dụng cổng mặc định 3306, cổng này cũng được yêu cầu để giao tiếp.

Cổng 33062 (Mặc định TCP/IP)

Giao tiếp trên cổng được mã hóa và diễn ra giữa máy khách và máy chủ. Cần lưu ý rằng cổng MySQL này được thiết lập cụ thể để tạo điều kiện thuận lợi cho kết nối của quản trị viên. Giao diện cho phép các hoạt động như quản lý người dùng, cấu hình máy chủ, xem nhật ký, thực hiện xuất và nhập.

Cổng 33061 (TCP/IP)

Cổng Shell là 33061/TCP, 33060/TCP và 3306/TCP, trong số những cổng khác. Chạy InnoDB Cluster khiến việc sử dụng cổng 33061/TCP trở nên bắt buộc. Giao tiếp của cổng cũng được mã hóa. Chức năng chính của nó là xác minh máy chủ trong quá trình cấu hình InnoDB Cluster.

Vậy là xong! Chúng tôi hy vọng hướng dẫn này hữu ích; và trong trường hợp bạn đang tìm kiếm một công cụ giúp bạn quản lý MySQL hiệu quả nhất, chúng tôi muốn giới thiệu [dbForge Studio for MySQL](#), IDE tối ưu của chúng tôi dành cho [quản trị cơ sở dữ liệu MySQL](#), có sẵn để dùng thử miễn phí trong 30 ngày.

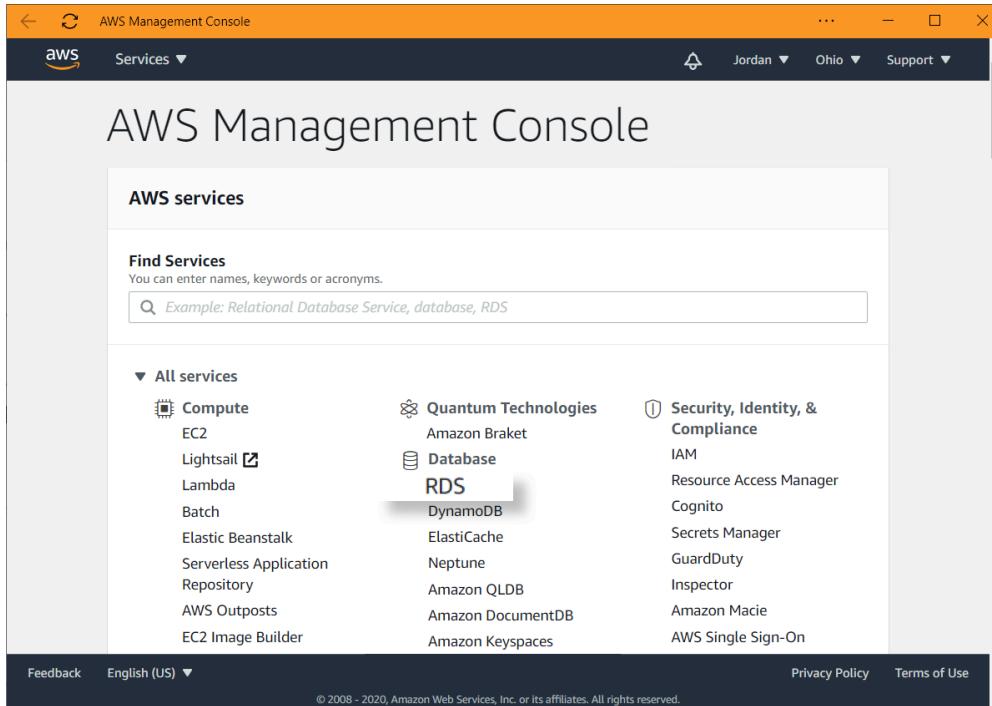
2.5 Kết nối với cơ sở dữ liệu Amazon RDS MySQL

Cách triển khai phiên bản cơ sở dữ liệu MySQL trong Amazon RDS

Bước 1: Mở bảng điều khiển Amazon RDS

Trước hết, bạn nên đăng nhập vào AWS Management Console như trong ảnh chụp màn hình bên dưới.

Tiếp theo, tìm RDS trong phần Cơ sở dữ liệu của Tất cả dịch vụ và nhấp để mở Amazon RDS Console.

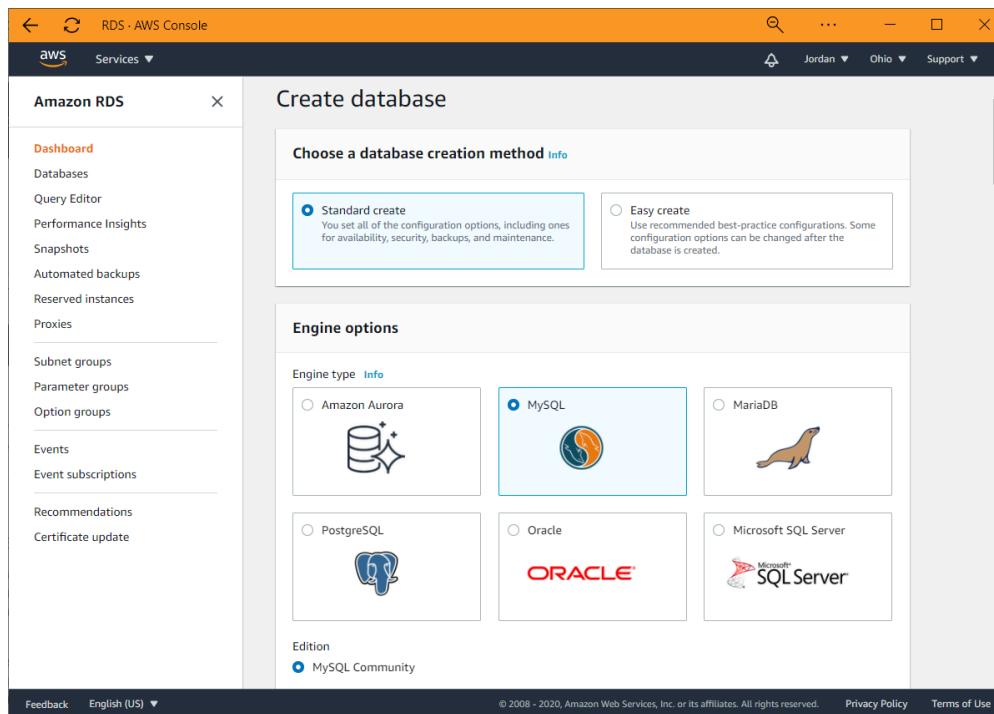


Bước 2: Tạo một phiên bản Amazon MySQL

Để tạo một phiên bản cơ sở dữ liệu MySQL trong AWS, hãy vào Databases trên menu bên và nhấp vào Create database. Trong bước này, bạn sẽ cần chọn phương pháp tạo cơ sở dữ liệu, cấu hình tùy chọn công cụ và chọn phiên bản cơ sở dữ liệu.

Xin lưu ý rằng khi chọn phương pháp tạo cơ sở dữ liệu Tiêu chuẩn, bạn sẽ có cơ hội cấu hình các tùy chọn cơ sở dữ liệu, bao gồm cả các tùy chọn nâng cao.

Sau đó, chọn công cụ MySQL và chọn phiên bản bên dưới.

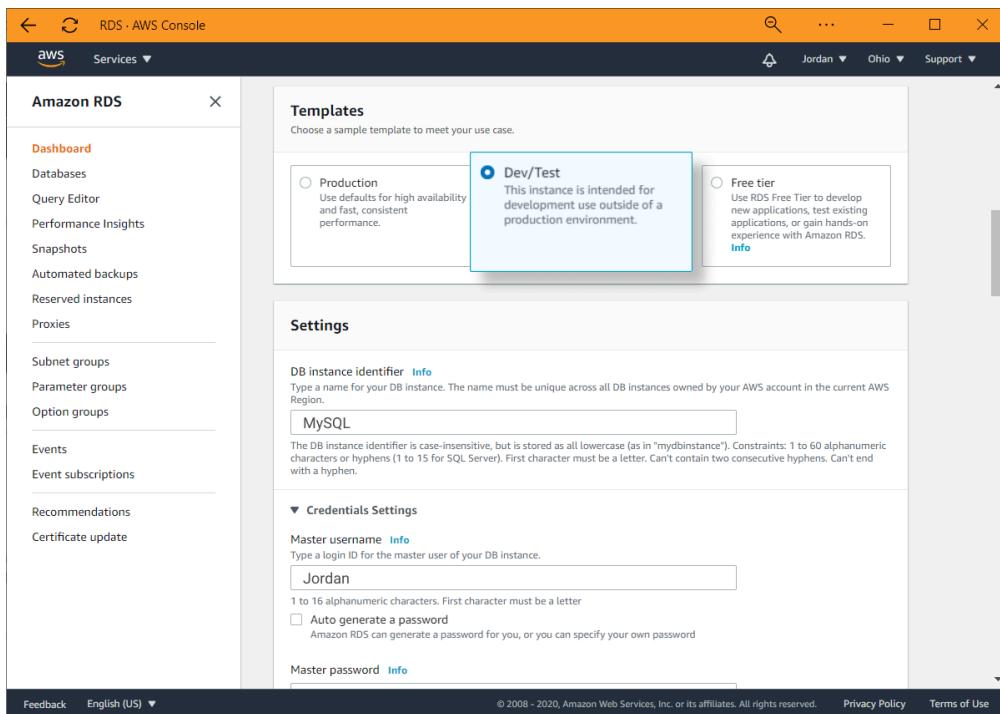


Bước 3: Chọn một mẫu

Tiếp theo, bạn sẽ cần chọn một mẫu cho phiên bản MySQL tương lai của mình trên Amazon RDS.

Tùy thuộc vào trường hợp sử dụng, hãy chọn giữa các tùy chọn gói Sản xuất, Phát triển/Kiểm thử và Miễn phí.

Xin lưu ý rằng Amazon RDS Free Tier có hiệu lực trong 12 tháng và theo ưu đãi này, bạn sẽ nhận được 750 giờ RDS, 20 GB dung lượng lưu trữ thông thường và 20 GB để sao lưu mỗi tháng.

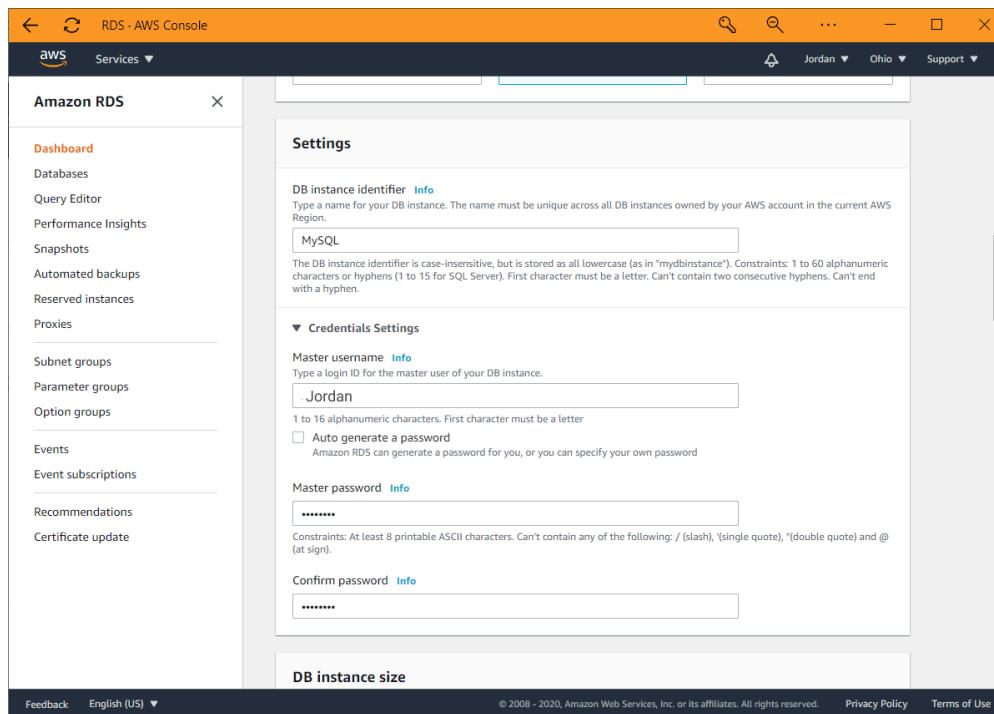


Bước 4.1: Cung cấp cài đặt cho cơ sở dữ liệu MySQL của bạn trên AWS

Ở bước này, bạn sẽ cần phải xác định cài đặt phiên bản.

Đầu tiên, hãy cung cấp một tên duy nhất cho phiên bản của bạn.

Sau đó, hãy nghĩ đến tên người dùng và mật khẩu chính phù hợp. Đảm bảo bạn lưu chúng vì bạn sẽ sử dụng chúng sau này để kết nối với Amazon RDS qua dbForge Studio cho MySQL.

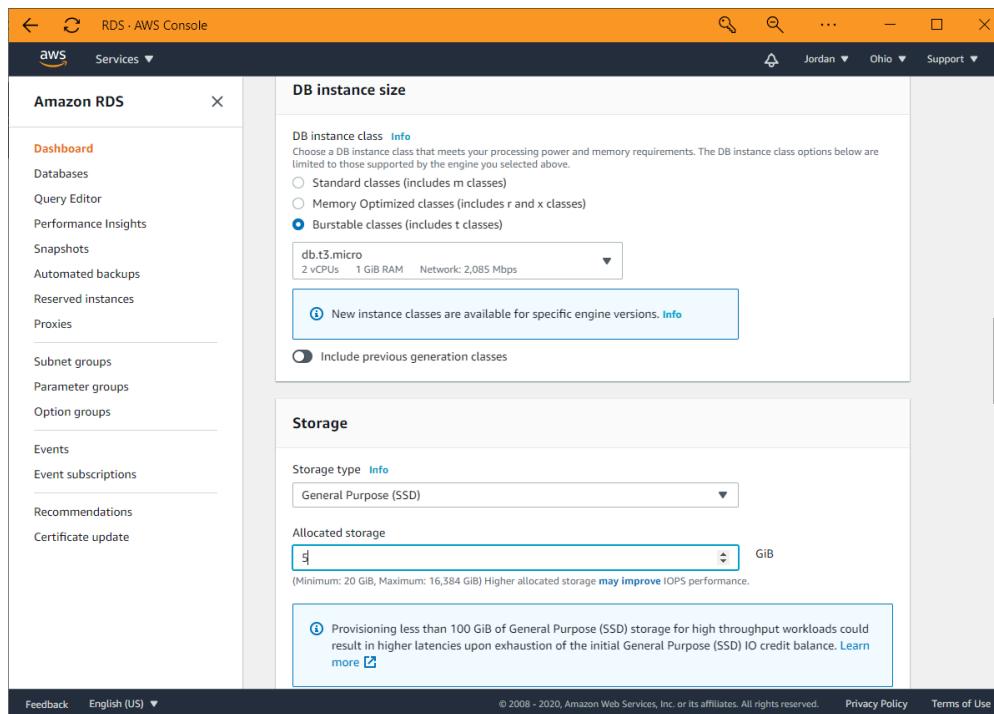


Bước 4.2: Tiếp tục cấu hình phiên bản MySQL của bạn

Sau khi bạn đã thiết lập thông tin xác thực cho phiên bản của mình, hãy tiếp tục và xác định kích thước phiên bản Amazon MySQL trong tương lai và các tùy chọn lưu trữ.

Chọn một lớp thể hiện DB, loại lưu trữ và dung lượng lưu trữ được phân bổ đáp ứng được yêu cầu khối lượng công việc của bạn.

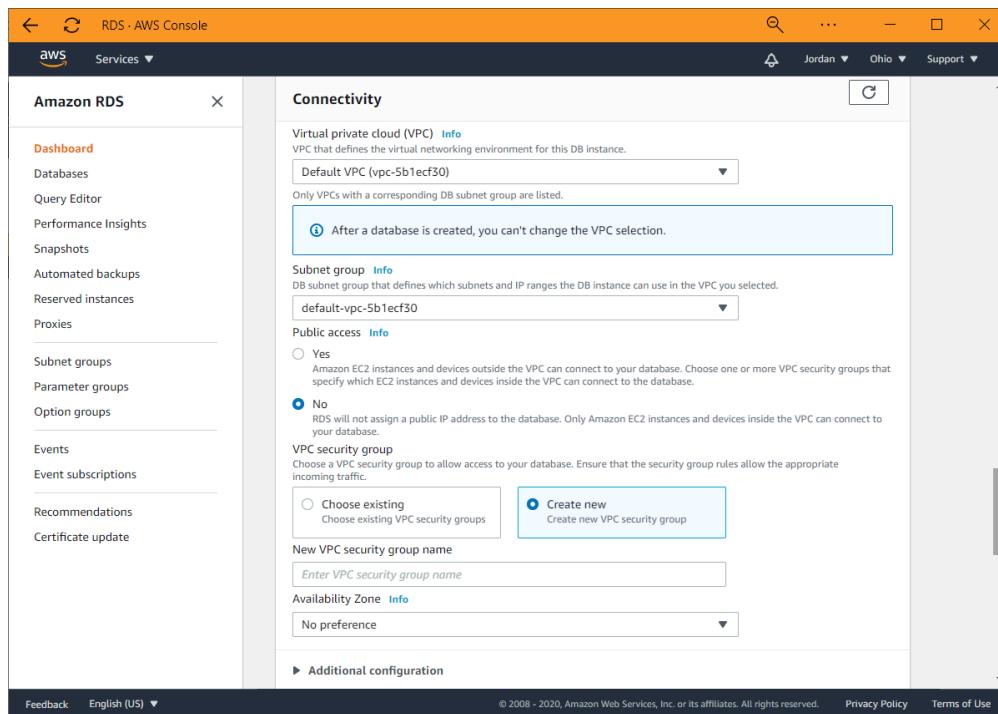
Lưu ý rằng bạn có thể mở rộng dung lượng lưu trữ được phân bổ lên tối đa 65 TB với Amazon RDS cho MySQL.



Bước 4.3: Xác định các tùy chọn kết nối

Bây giờ chúng ta đã đến phần Kết nối. Tại đây, bạn có thể cung cấp tất cả các cấu hình kết nối cần thiết cho phiên bản MySQL của mình trên AWS RDS.

Xin lưu ý rằng bạn không thể thay đổi Virtual Private Cloud sau khi tạo cơ sở dữ liệu.

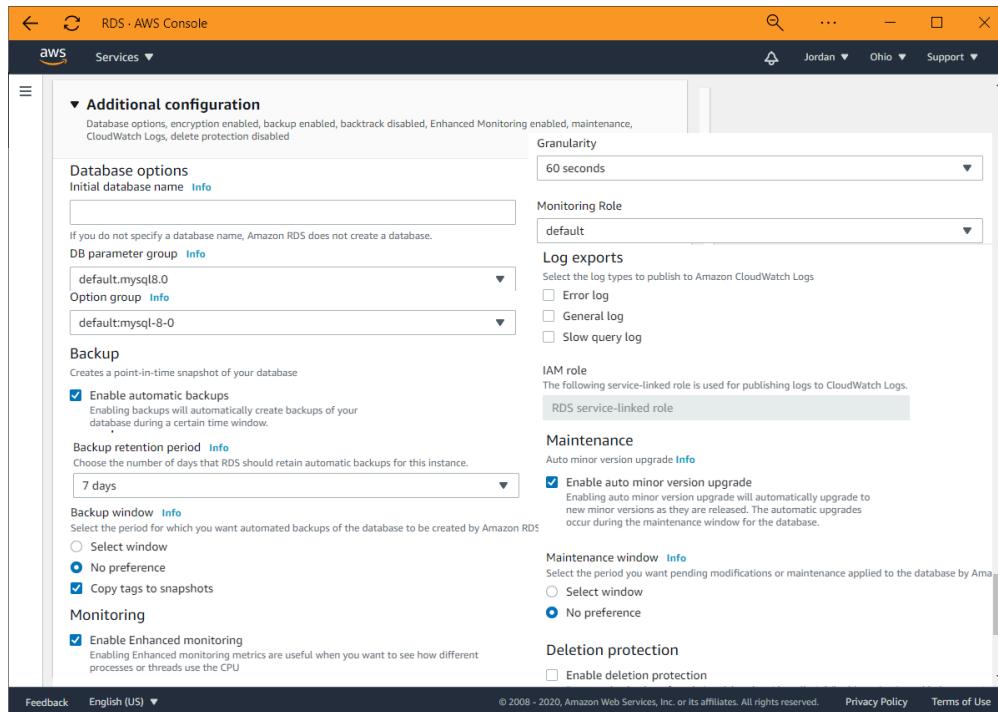


Bước 5: Xác định cài đặt nâng cao cho MySQL của bạn trên AWS

Trong bước này, bạn sẽ cần cấu hình các tùy chọn cơ sở dữ liệu bổ sung cần thiết cho AWS RDS để khởi chạy phiên bản MySQL DB của bạn. Bạn có thể bật sao lưu cơ sở dữ liệu, giám sát nâng cao và bảo trì.

Trong phần Bảo trì, chúng tôi khuyên bạn nên chọn **Bật nâng cấp phiên bản phụ tự động** để nhận bản cập nhật tự động ngay khi có sẵn.

*Xin lưu ý rằng nếu tùy chọn **Bật bảo vệ xóa** được bật, bạn sẽ không thể xóa cơ sở dữ liệu đã tạo.*



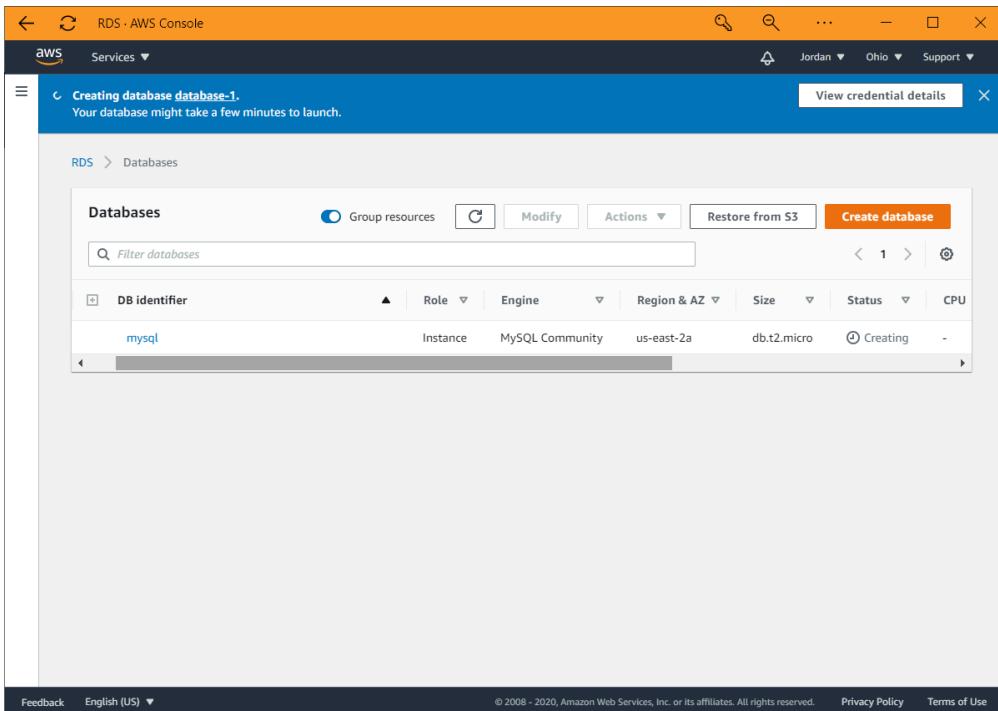
Bước 6: Khởi chạy phiên bản MySQL trên AWS

Sau khi bạn đã xác định tất cả các cài đặt, cuối cùng bạn có thể nhấp vào Khởi chạy DB Instance.

Ngay sau khi quá trình tạo phiên bản hoàn tất (có thể mất một thời gian), bạn sẽ có thể kiểm tra trạng thái cùng với một số thông tin cấu hình khác trong cửa sổ tương ứng như hiển thị trên ảnh chụp màn hình.

Để xem thông tin chi tiết về phiên bản MySQL của bạn, hãy nhấp vào tên phiên bản cơ sở dữ liệu.

Kiểm tra thông tin chi tiết và nhấp vào Tạo cơ sở dữ liệu để hoàn tất quy trình.



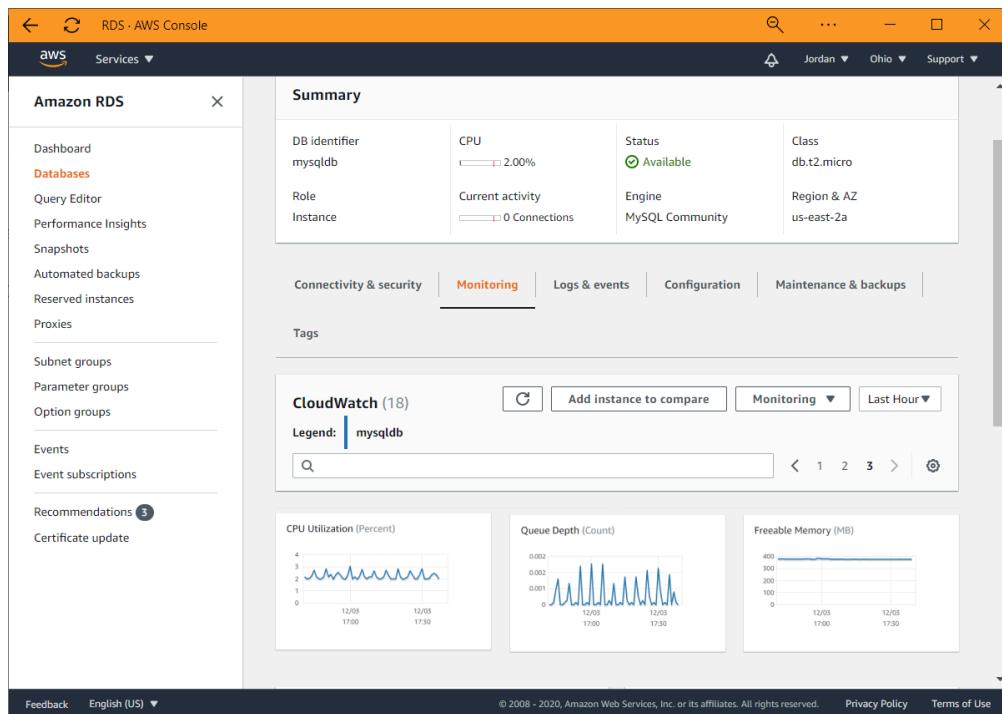
Bước 7: Xem chi tiết phiên bản MySQL

Bây giờ, khi chúng ta tiếp cận phần Tóm tắt, chúng ta có thể thấy bảng điều khiển, như minh họa trong ảnh chụp màn hình. Nó cung cấp thông tin chi tiết về phiên bản MySQL liên quan đến các sự kiện gần đây, bao gồm Sử dụng CPU, Kết nối DB, Không gian lưu trữ miễn phí.

Để kiểm tra thông tin cần thiết, hãy chuyển đổi giữa các tab. Ví dụ, trên tab Kết nối & bảo mật, bạn có thể truy cập thông tin liên quan đến MySQL Endpoint.

Lưu ý rằng sau này bạn sẽ cần chỉ định cả địa chỉ DNS từ điểm cuối của phiên bản (dưới dạng máy chủ) và số cổng (dưới dạng cổng) trong chuỗi kết nối để kết nối với phiên bản MySQL.

Khi trạng thái của phiên bản khả dụng, nghĩa là phiên bản cơ sở dữ liệu đã được triển khai, bạn có thể truy cập dbForge Studio for MySQL để kết nối với Amazon RDS.



The screenshot shows the AWS RDS console for a MySQL database named mysqlDb. The 'Monitoring' tab is selected. The 'CloudWatch' section shows 18 metrics for mysqlDb. Three charts are displayed: 'CPU Utilization (Percent)' with values between 1% and 3%; 'Queue Depth (Count)' with values between 0.001 and 0.002; and 'Freeable Memory (MB)' with a constant value of 400 MB. The time range for the charts is 'Last Hour'.

Sử dụng dbForge Studio cho MySQL để kết nối với phiên bản cơ sở dữ liệu MySQL của Amazon RDS

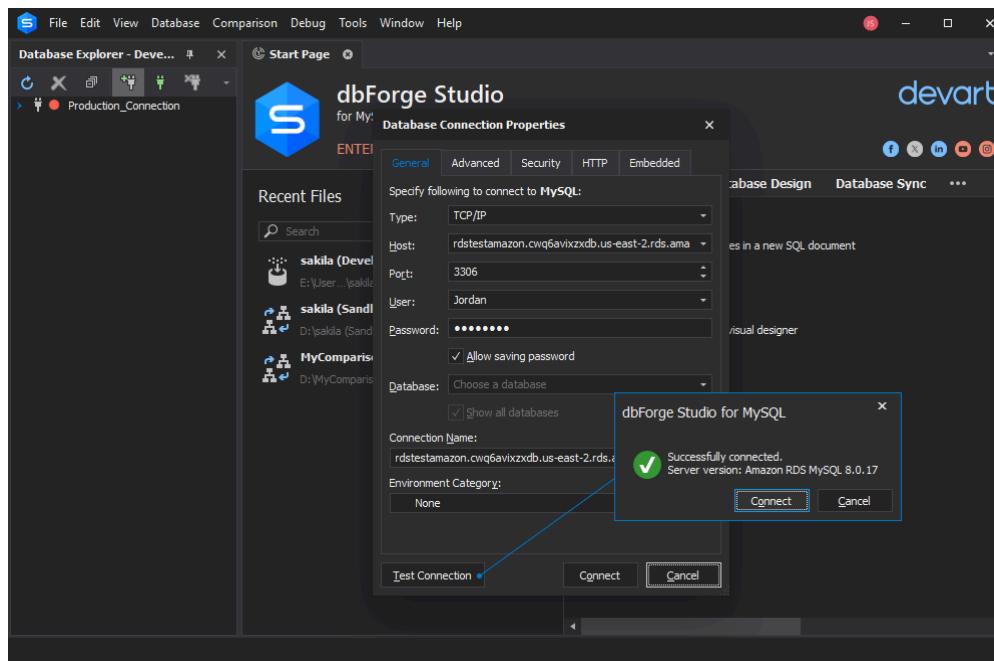
Kết nối với cơ sở dữ liệu MySQL của Amazon RDS

dbForge Studio là giải pháp tuyệt vời cho các nhà phát triển SQL vì nó cho phép bạn kết nối với phiên bản AWS RDS từ máy cục bộ và tạo điều kiện thuận lợi cho trải nghiệm cơ sở dữ liệu của bạn với các tính năng tích hợp mạnh mẽ của nó. Để cấu hình quyền truy cập từ xa vào phiên bản Amazon RDS của bạn thông qua dbForge Studio for MySQL, hãy thực hiện như sau:

1. Trong Database Explorer, nhập vào Database Connection. Tùy chọn, bạn có thể vào Database trên Main Menu và nhập vào New Connection.
2. Trong cửa sổ Thuộc tính kết nối cơ sở dữ liệu, hãy chỉ định Loại kết nối và nhập Tên máy chủ, Cổng, Người dùng và Mật khẩu.

3. Sau khi đã cấu hình xong tất cả các thiết lập, hãy nhập vào OK.

Bây giờ bạn đã thiết lập thành công kết nối từ xa tới AWS RDS, bạn có thể muốn tìm hiểu sâu hơn và cách [nhập dữ liệu MySQL vào Amazon RDS](#) và [xuất dữ liệu từ Amazon RDS](#) sang cơ sở dữ liệu MySQL tại chỗ.

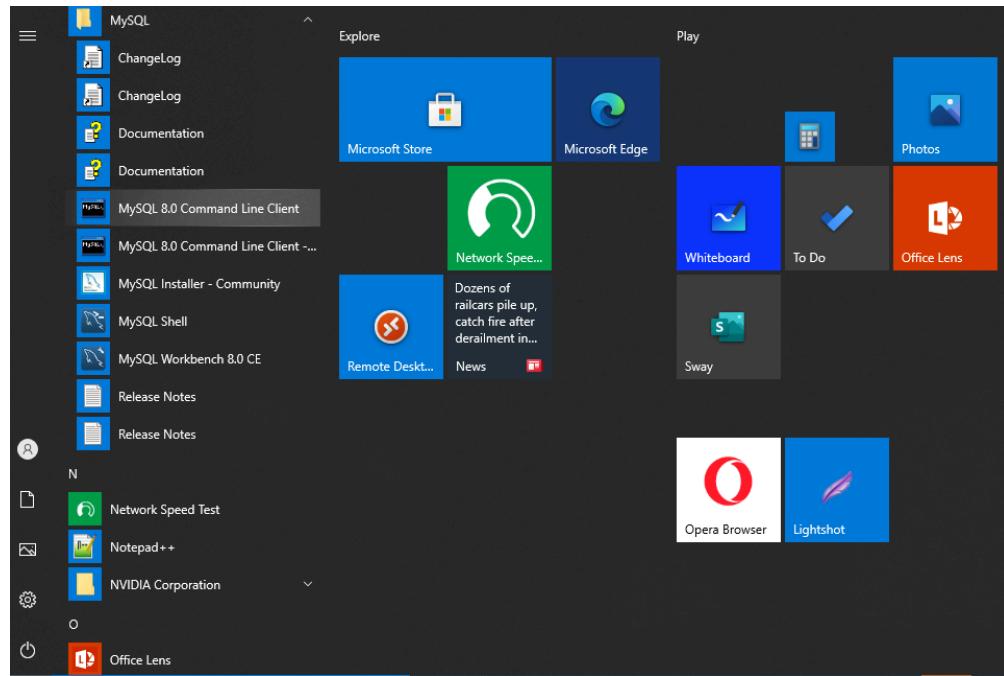


3. Quản lý MySQL

3.1 Tạo cơ sở dữ liệu MySQL mới

Tạo cơ sở dữ liệu từ Command Line Client

MySQL Command Line Client thường đi kèm với gói cài đặt MySQL Server. Nó được cài đặt theo hai phiên bản – có hỗ trợ UTF-8 và không hỗ trợ UTF-8. Bạn có thể chạy console client ngay từ menu Start.



Để tạo cơ sở dữ liệu MySQL mới thông qua MySQL Command Line Client:

1. Chạy client.
2. Nhập mật khẩu của bạn.
3. Thực hiện lệnh **create database** .

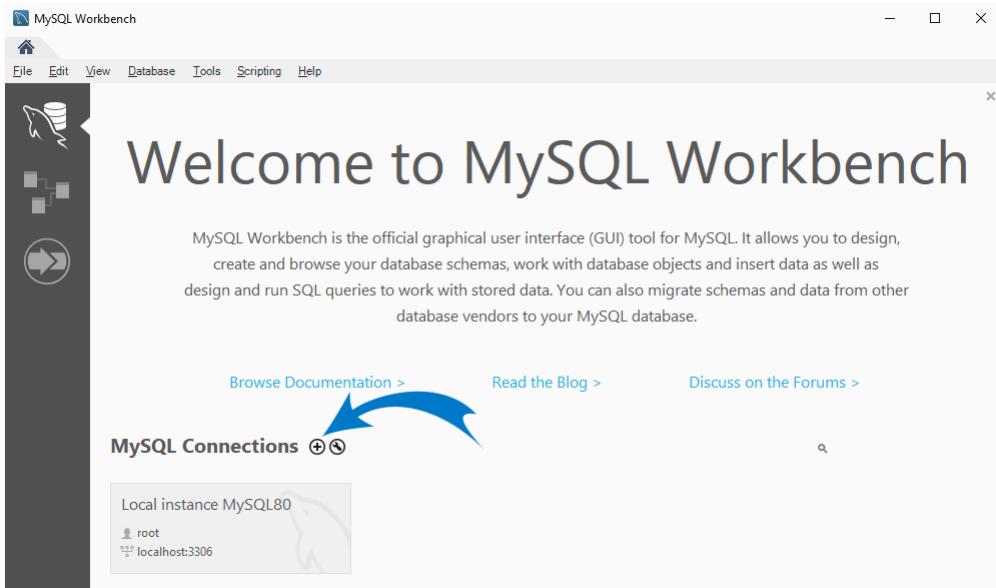
A screenshot of the MySQL 8.0 Command Line Client window. The title bar says 'MySQL 8.0 Command Line Client'. The main area is a black terminal window. The user has entered the command 'create database test;' and pressed Enter. The response is 'Query OK, 1 row affected (0.01 sec)'. The prompt 'mysql>' is visible at the bottom.

Cách tạo cơ sở dữ liệu trong MySQL Workbench

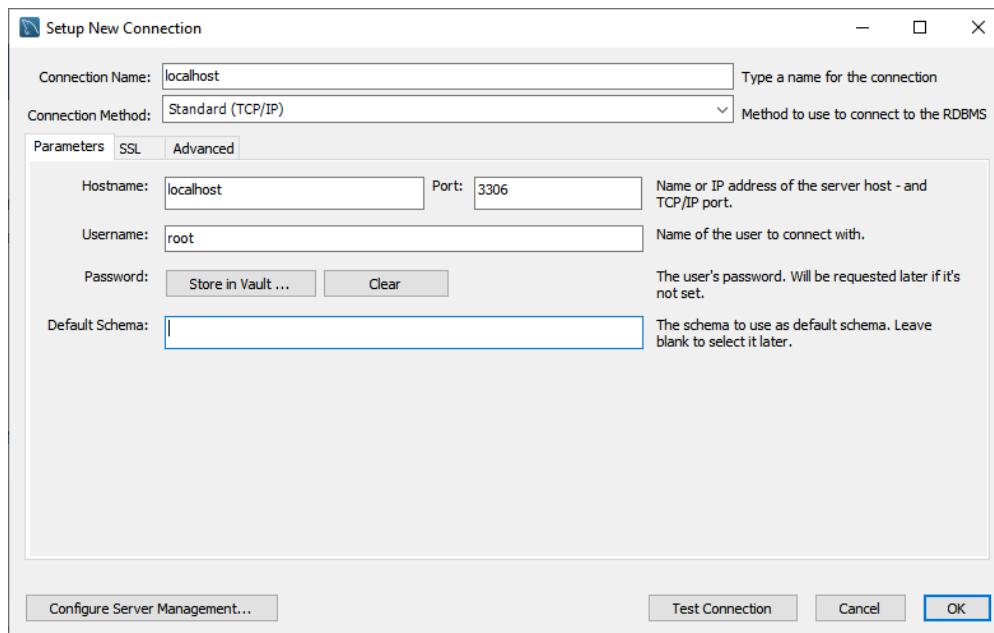
MySQL Workbench là một công cụ cơ sở dữ liệu trực quan phổ biến để thiết kế, phát triển và quản lý cơ sở dữ liệu MySQL.

Cách sử dụng MySQL Workbench để tạo cơ sở dữ liệu:

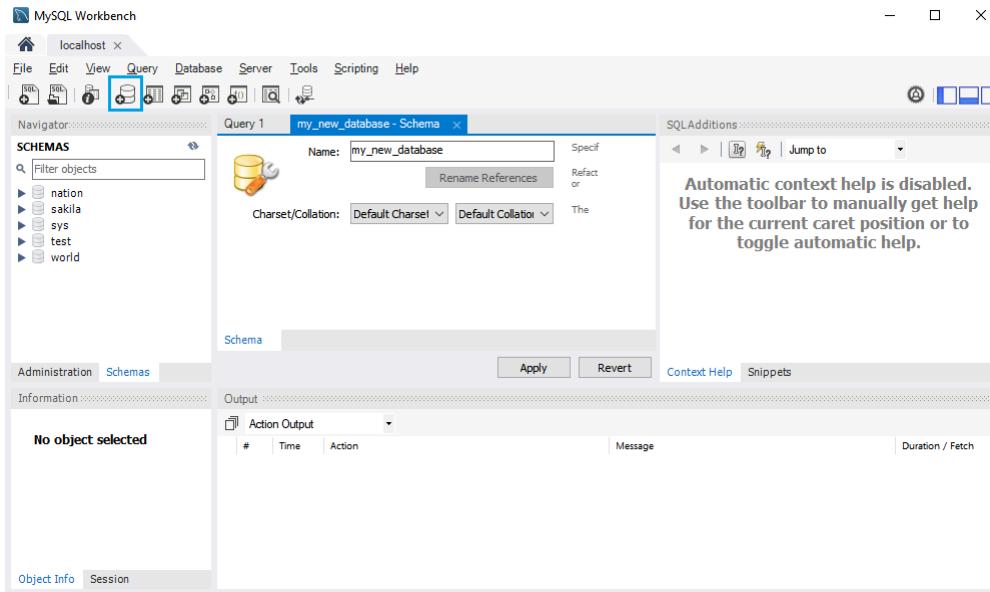
1. Khởi chạy MySQL Workbench và nhấp vào nút **+** để mở **trình hướng dẫn Thiết lập kết nối mới**.



2. Nhập tên cho kết nối và tên người dùng, sau đó nhấp vào **Kiểm tra kết nối**. Nhập mật khẩu vào hộp thoại yêu cầu nhập mật khẩu. Chúng tôi nhập *localhost* và *root*.



3. Nhập vào kết nối cần thiết trong phần **Kết nối MySQL** của trang bắt đầu Workbench.
4. Trong cửa sổ MySQL Workbench mở ra, nhập vào nút **Create a new schema in the connected server** trên thanh công cụ chính. Sau đó nhập tên schema, thay đổi bộ ký tự và đổi chiều nếu cần, rồi nhập vào **Apply**.



5. Trong cửa sổ **Apply SQL Script to Database** mở ra, hãy nhấp vào **Apply** . Sau đó nhấp vào **Finish** .



Review SQL Script

Apply SQL Script

Review the SQL Script to be Applied on the Database

Online DDL

Algorithm: Default

Lock Type: Default

```
1      CREATE SCHEMA `my_new_database` ;  
2
```

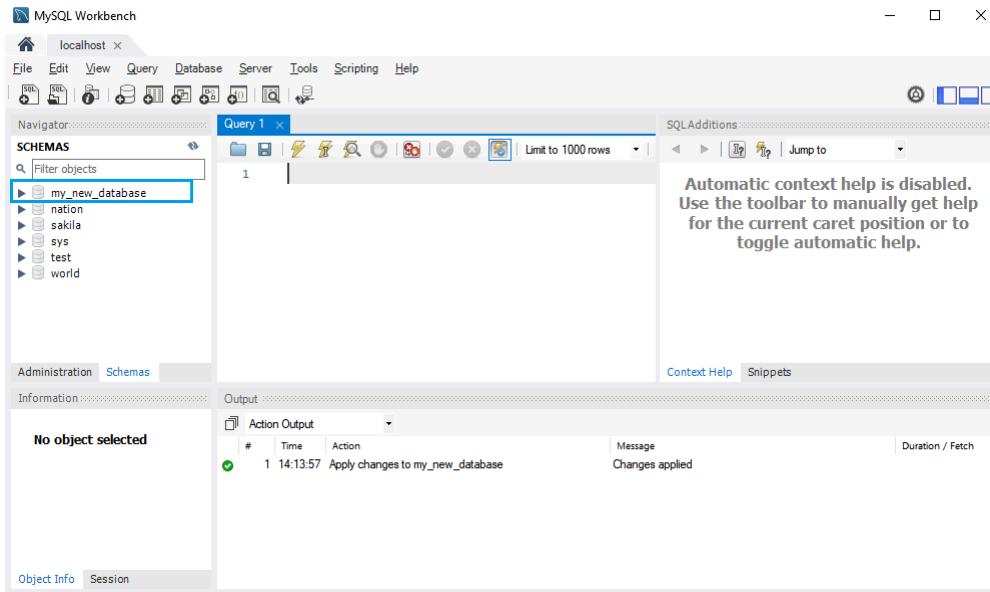
< >

Back

Apply

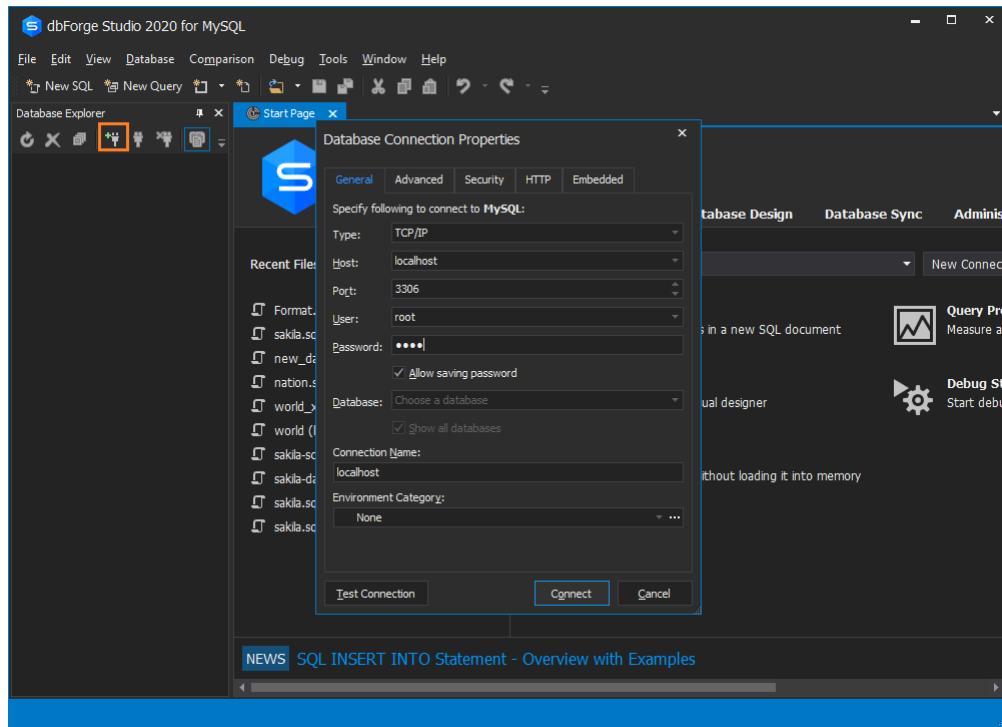
Cancel

6. Kiểm tra xem cơ sở dữ liệu đã xuất hiện trong Navigator chưa.

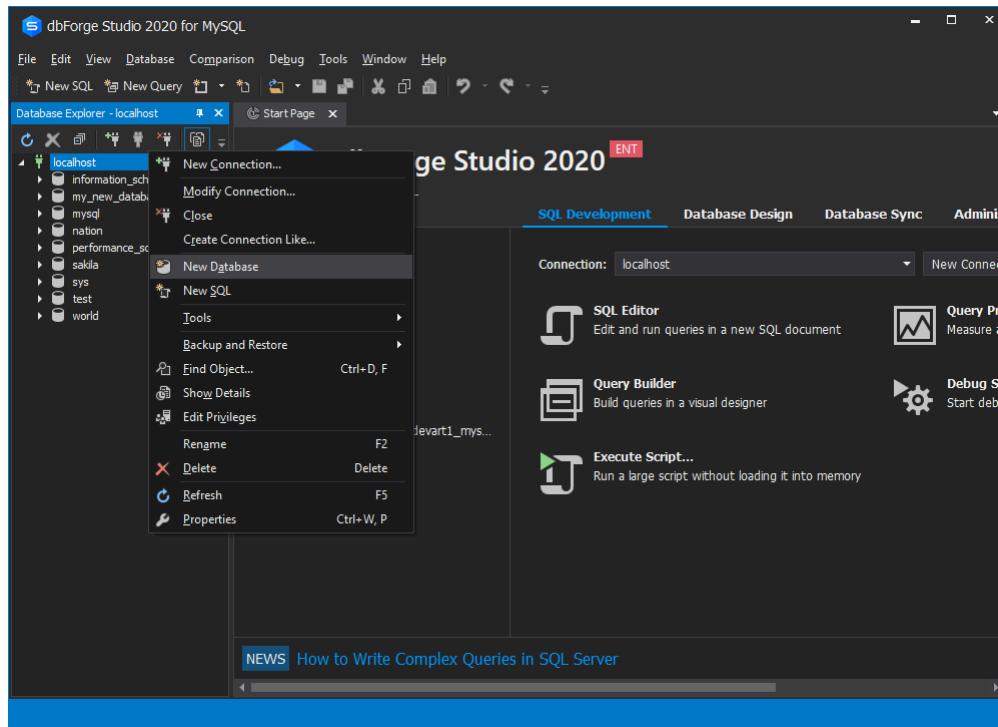


Cách tạo cơ sở dữ liệu bằng dbForge Studio cho MySQL

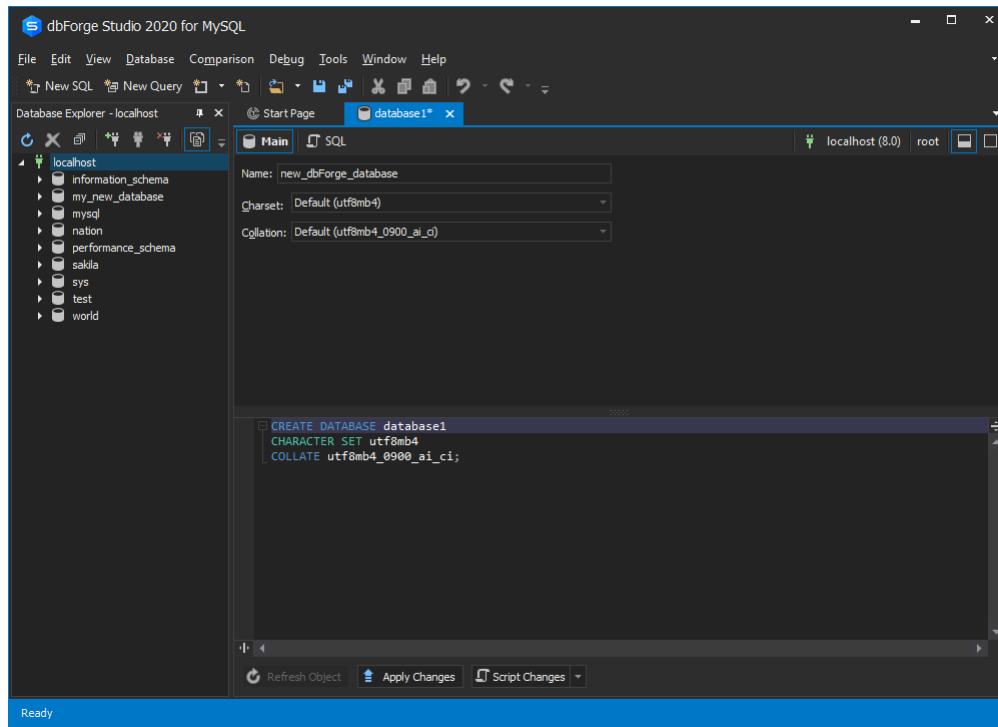
1. Trước tiên, bạn cần tạo một kết nối cần thiết. Nhấp vào nút **New Connection** trên thanh công cụ Database Explorer. Hoặc, hãy vào menu Database trên thanh công cụ chính và nhấp vào **New Connection** .
2. Trong cửa sổ Thuộc tính kết nối cơ sở dữ liệu mở ra, hãy cung cấp tất cả thông tin xác thực cần thiết cho kết nối của bạn.



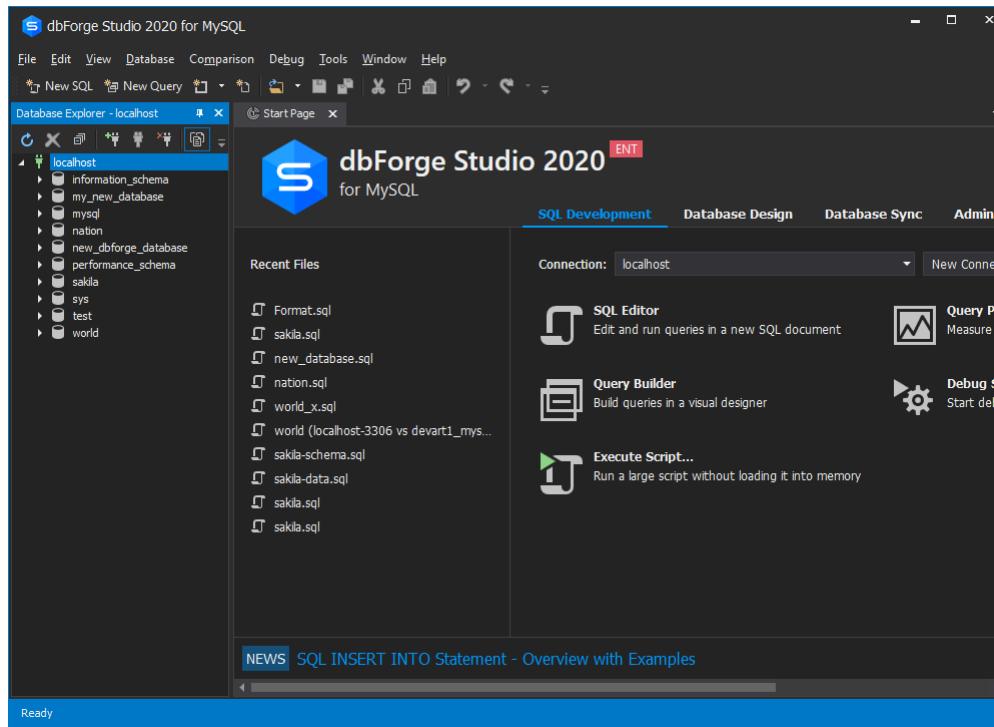
3. Kết nối mới sẽ xuất hiện trong Database Explorer. Nhấp chuột phải vào tên kết nối và chọn **New Database**. Hoặc, hãy vào menu **Database** trên thanh công cụ chính và nhấp vào **New Database**.



4. Trong tab Cơ sở dữ liệu mới sẽ mở ra, nhập tên cho cơ sở dữ liệu mới của bạn, chọn charset và collation. Bạn có thể kiểm tra tập lệnh cho cơ sở dữ liệu ở phần dưới của cửa sổ. Nhập vào **Áp dụng thay đổi**, sau khi bạn đã cấu hình mọi thứ theo yêu cầu.



5. Kiểm tra xem cơ sở dữ liệu mới tạo của bạn đã xuất hiện trên máy chủ MySQL chưa. Để thực hiện việc này, hãy nhấp chuột phải vào tên kết nối trong Database Explorer và nhấp vào **Refresh** .



Sau khi tạo cơ sở dữ liệu

Khi bạn có nhiều cơ sở dữ liệu trên máy chủ MySQL, để bắt đầu làm việc với cơ sở dữ liệu bạn đã tạo, hãy sử dụng câu lệnh sau:

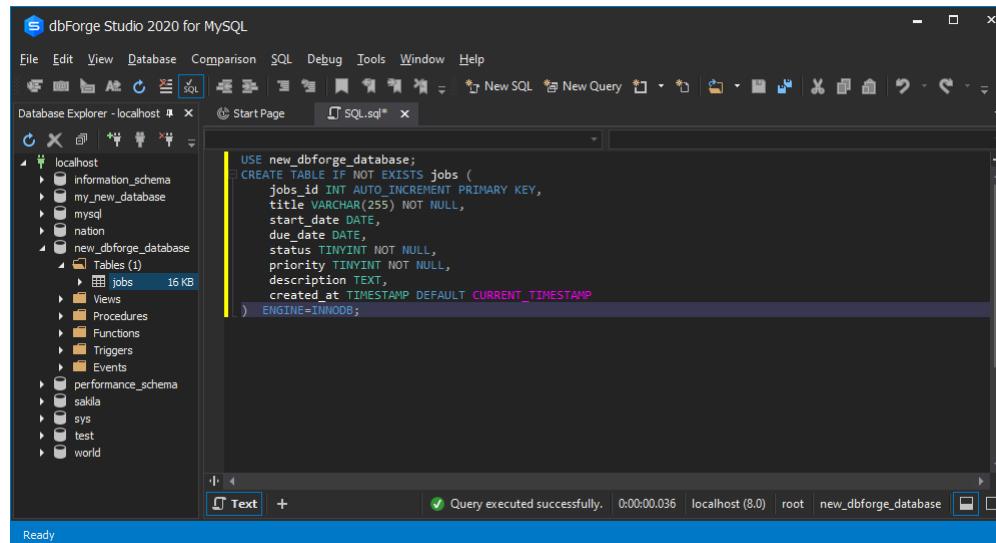
```
USE database_name;
```

Để tạo một bảng mới trong cơ sở dữ liệu, hãy sử dụng cú pháp sau:

```
CREATE TABLE [IF NOT EXISTS] table_name(  
    column_1_definition,  
    column_2_definition,  
    ...)
```

```
table_constraints  
) ENGINE=storage_engine;
```

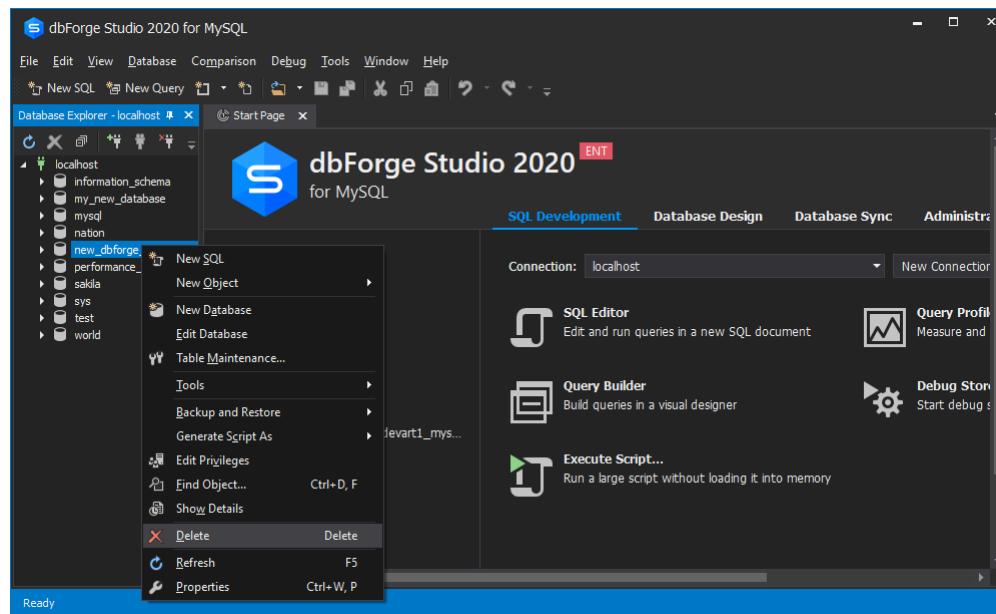
Trong hình ảnh sau, bạn có thể thấy cách tạo bảng MySQL trong dbForge Studio for MySQL. Studio có tính năng hoàn thành mã tự động, do đó không cần phải nhập toàn bộ mã.



Để xóa một bảng, hãy sử dụng câu lệnh sau:

```
DROP DATABASE databasename;
```

dbForge Studio for MySQL cho phép xóa cơ sở dữ liệu trực quan mà không cần phải viết mã. Chỉ cần nhấp chuột phải vào cơ sở dữ liệu cần thiết trong Database Explorer, sau đó nhấp vào **Delete**.



3.2 Tạo bảng MySQL

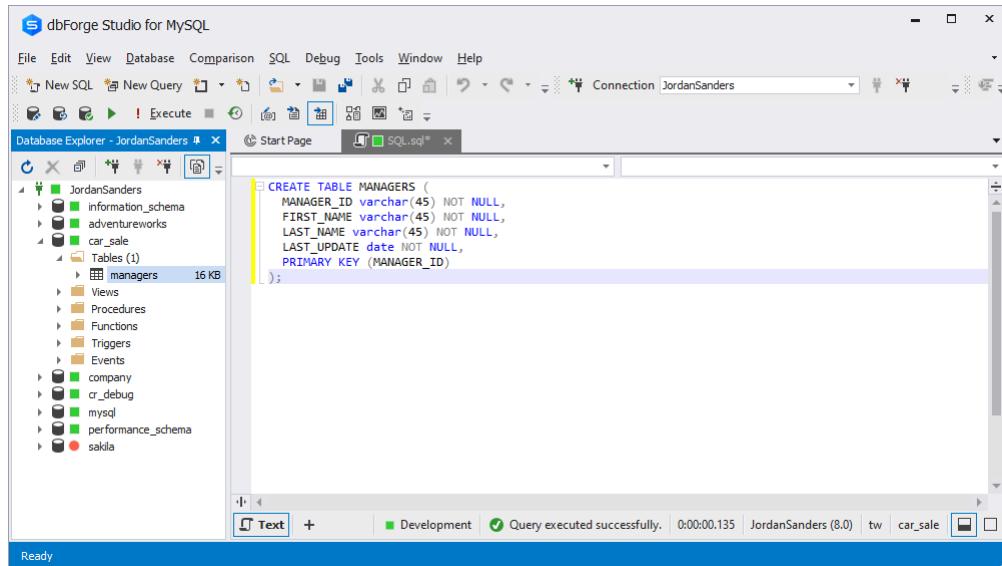
Tạo một bảng MySQL

Cách 1 – sử dụng câu lệnh CREATE TABLE

Chạy câu lệnh MySQL này và nhập vào **Thực thi** :

```
CREATE TABLE MANAGERS (
    MANAGER_ID varchar(45) NOT NULL,
    FIRST_NAME varchar(45) NOT NULL,
    LAST_NAME varchar(45) NOT NULL,
    LAST_UPDATE date NOT NULL,
    PRIMARY KEY (MANAGER_ID)
);
```

Và sau đó nhập vào **Làm mới đối tượng** :

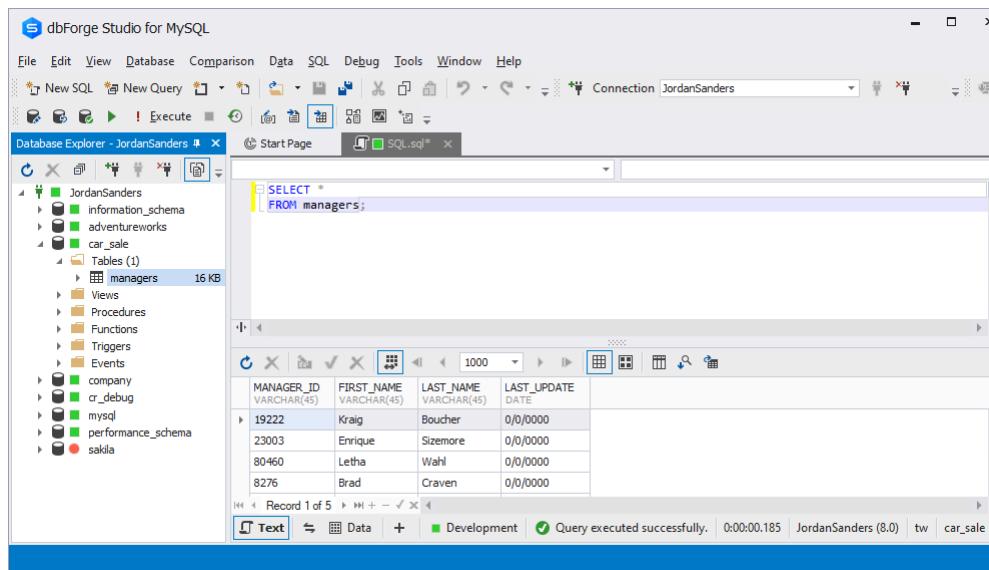


Hãy điền dữ liệu vào bảng:

```
INSERT INTO MANAGERS(MANAGER_ID,FIRST_NAME, LAST_NAME, LAST_UPDATE)
VALUES
(08276,'Brad','Craven',11/29/1976),
(19222,'Kraig','Boucher',10/10/1990),
(23003,'Enrique','Sizemore',5/26/1990),
(80460,'Letha','Wahl',3/9/1971),
(86849,'Harlan','Ludwig',1/11/1997);
```

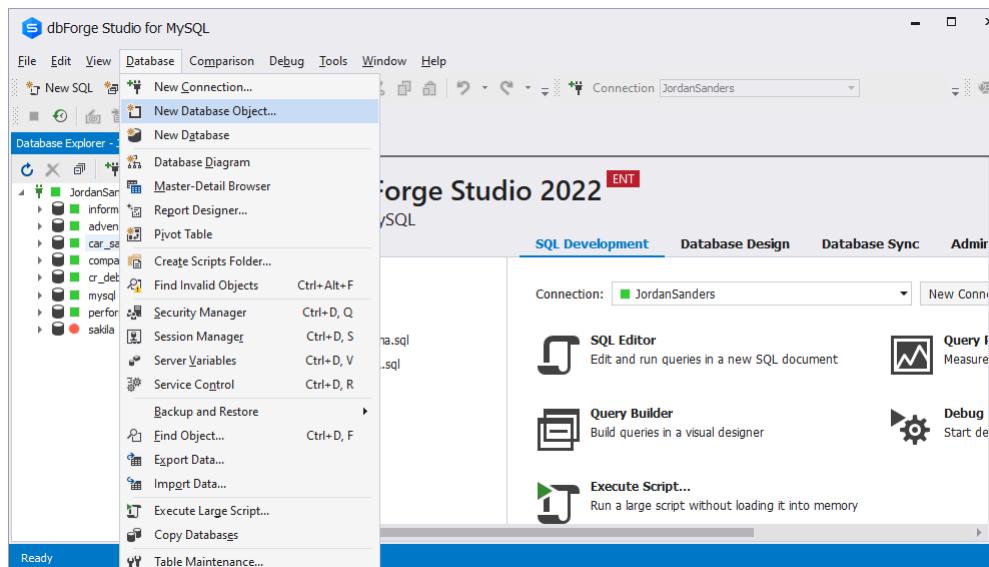
Để kiểm tra xem dữ liệu đã được thêm vào hay chưa, hãy thực hiện câu lệnh sau:

```
SELECT *
FROM MANAGERS;
```



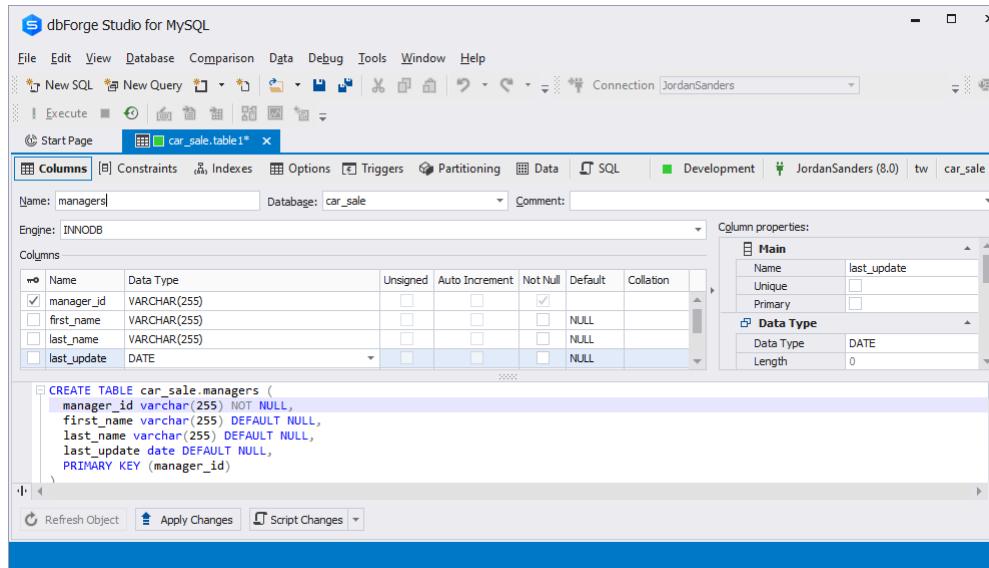
Cách 2 – sử dụng tùy chọn **Đối tượng cơ sở dữ liệu mới**

1. Điều hướng đến **Cơ sở dữ liệu > Đối tượng cơ sở dữ liệu mới** :

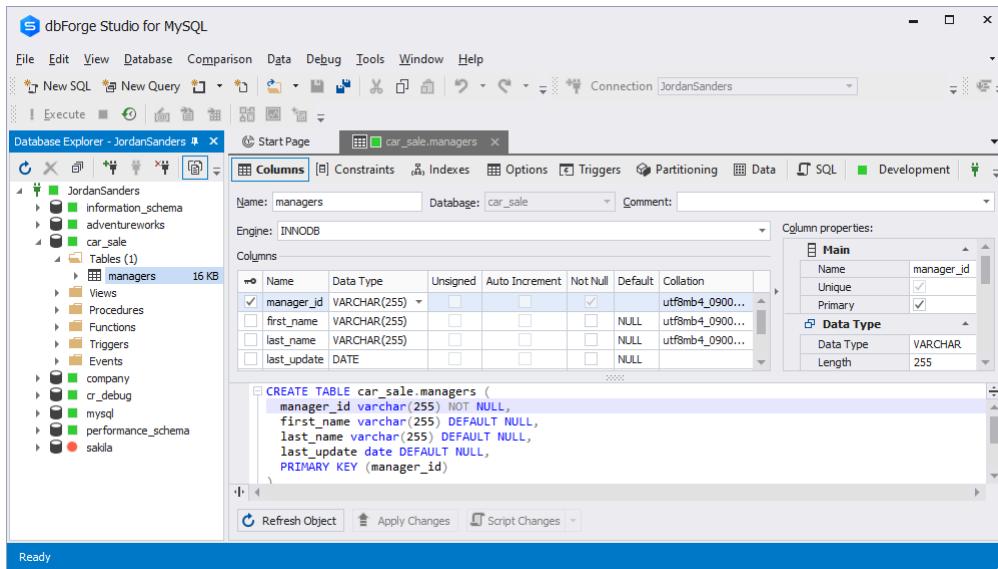


2. Chọn cơ sở dữ liệu mà bạn muốn tạo bảng mới và chọn **Bảng** trong danh sách các loại đối tượng ở bên phải, sau đó nhấp vào **Tạo** .

3. Một tab sẽ được mở ra cho phép bạn thiết kế và cấu hình bảng bạn muốn tạo. Về cơ bản, chức năng này cho phép đạt được cùng kết quả như bạn có thể nhận được với truy CREATE TABLE vấn, nhưng theo cách toàn diện và thuận tiện về mặt trực quan:



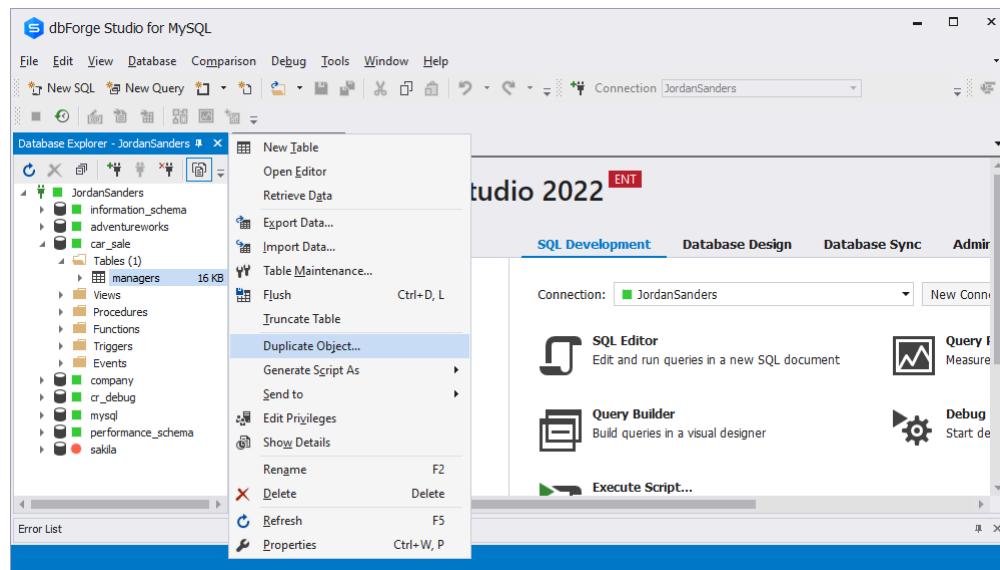
4. Sau khi bạn đã điều chỉnh tất cả các cột và cài đặt cần thiết, hãy nhấp vào **Apply Changes**. Vậy là xong, bảng sẽ được tạo trong cơ sở dữ liệu đã chỉ định. Nếu bạn cần cập nhật một số cài đặt của bảng, chỉ cần thay đổi các tùy chọn tương ứng và nhấp vào **Refresh Object** :



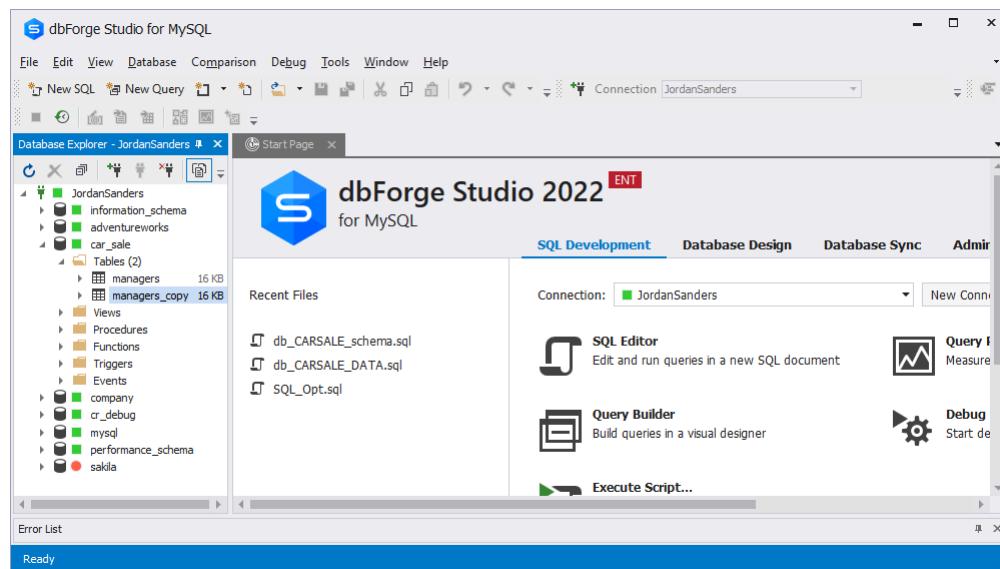
Tạo một bảng trùng lắp

Với dbForge Studio for MySQL, bạn có thể [sao chép bảng](#) một cách nhanh chóng và dễ dàng :

1. Nhấp chuột phải vào bảng cần thiết và nhấp vào **Sao chép đối tượng** :



2. Chọn cơ sở dữ liệu mà bạn muốn tạo bản sao bảng từ menu thả xuống **Cơ sở dữ liệu đích** .
3. Nhập tên cho bản sao của bảng hoặc giữ nguyên tên mặc định trong trường **Tên đối tượng mới** .
4. Nếu bạn muốn sao chép cả cấu trúc bảng và dữ liệu, hãy chọn **Sao chép dữ liệu** .
5. Cuối cùng, nhấp vào **OK** . Đừng quên nhấp vào **Refresh Object** . Bảng trùng lặp sẽ được tạo trong cơ sở dữ liệu được chỉ định:



Tạo một bảng tạm thời

- Người dùng phải được cấp quyền “*tạo bảng tạm thời*”
- Nếu bạn làm việc với công cụ cơ sở dữ liệu InnoDB, bạn không thể tạo bảng tạm thời
- Bạn có thể đặt tên cho các bảng tạm thời theo cùng cách như các bảng thông thường của MySQL
- Bảng tạm thời có mối quan hệ riêng biệt với lược đồ cơ sở dữ liệu. Do đó, nếu bạn xóa cơ sở dữ liệu, các bảng tạm thời sẽ không nhất thiết bị xóa trong cơ sở dữ liệu
- Câu `CREATE TEMPORARY TABLE` lệnh không gọi cam kết ngầm định

Hãy tạo một bảng tạm thời `Car_model` :

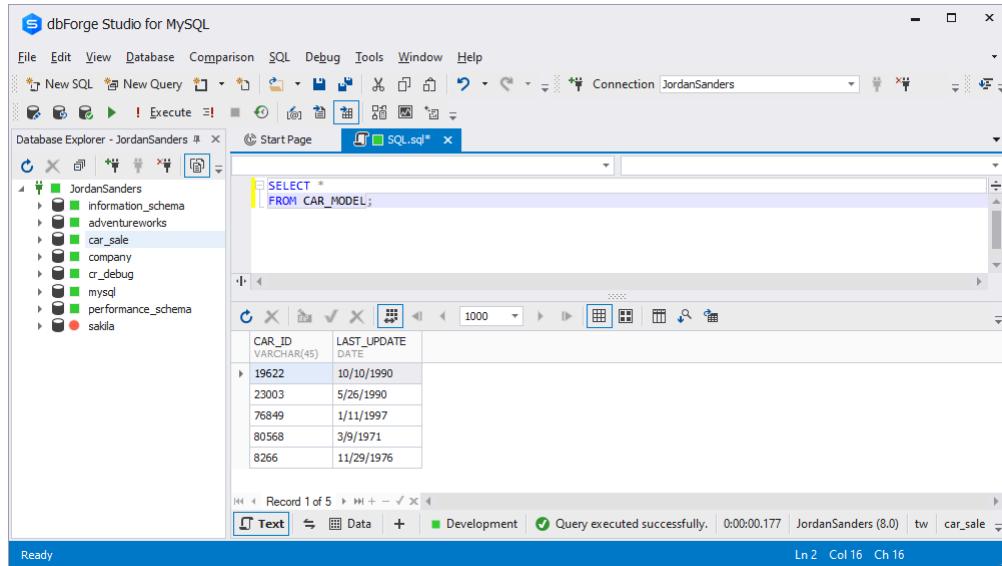
```
CREATE TEMPORARY TABLE CAR_MODEL
(CAR_ID varchar(45) NOT NULL,
LAST_UPDATE DATE NOT NULL,
PRIMARY KEY(CAR_ID)
);
```

Và thêm một số dữ liệu vào bảng đã tạo. Trong câu lệnh này, chúng ta chuyển đổi kiểu dữ liệu bằng `STR_TO_DATE` hàm:

```
INSERT INTO CAR_MODEL(CAR_ID, LAST_UPDATE)
VALUES
(08266,STR_TO_DATE('11/29/1976','%m/%d/%Y')),
(19622,STR_TO_DATE('10/10/1990','%m/%d/%Y')),
(23003,STR_TO_DATE('5/26/1990','%m/%d/%Y')),
(80568,STR_TO_DATE('3/9/1971','%m/%d/%Y')),
(76849,STR_TO_DATE('1/11/1997','%m/%d/%Y'));
```

Lưu ý rằng bảng có dữ liệu sẽ bị xóa sau khi bạn đóng phiên. Để kiểm tra xem bạn đã tạo bảng và thêm dữ liệu chưa, hãy thực hiện truy vấn này:

```
SELECT *
FROM CAR_MODEL;
```



The screenshot shows the dbForge Studio for MySQL interface. In the top navigation bar, the 'File', 'Edit', 'View', 'Database', 'Comparison', 'SQL', 'Debug', 'Tools', and 'Window' menus are visible. The 'Connection' dropdown is set to 'JordanSanders'. The 'Database Explorer' panel on the left lists databases: 'information_schema', 'adventureworks', 'car_sale', 'company', 'cr_debug', 'mysql', 'performance_schema', and 'sakila'. The 'SQL' tab in the center contains the query: 'SELECT * FROM CAR_MODEL;'. Below the query, the results are displayed in a table:

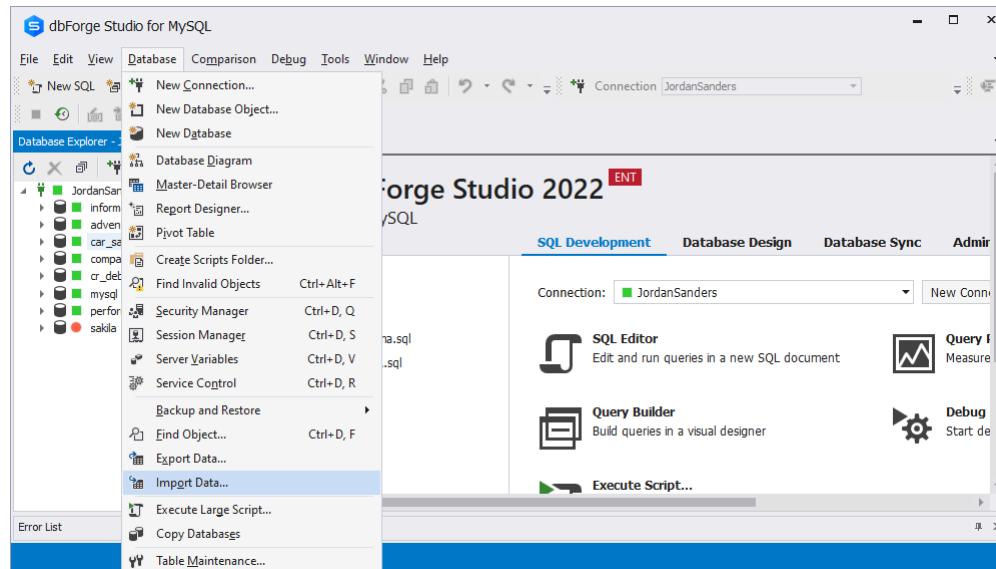
CAR_ID	LAST_UPDATE
VARCHAR(45)	DATE
19622	10/10/1990
23003	5/26/1990
76849	1/11/1997
80568	3/9/1971
8266	11/29/1976

At the bottom of the interface, the status bar shows 'Ready'.

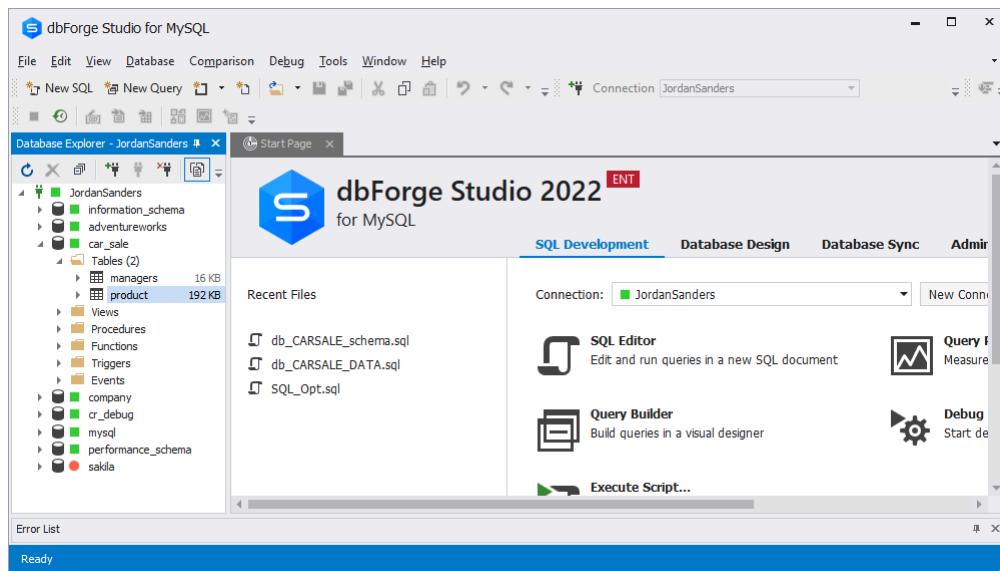
Tạo bảng MySQL từ CSV

Có một cách khác để tạo bảng trong dbForge Studio cho MySQL – [nhập bảng từ tệp CSV](#) :

1. Điều hướng đến **Cơ sở dữ liệu > Nhập dữ liệu** :



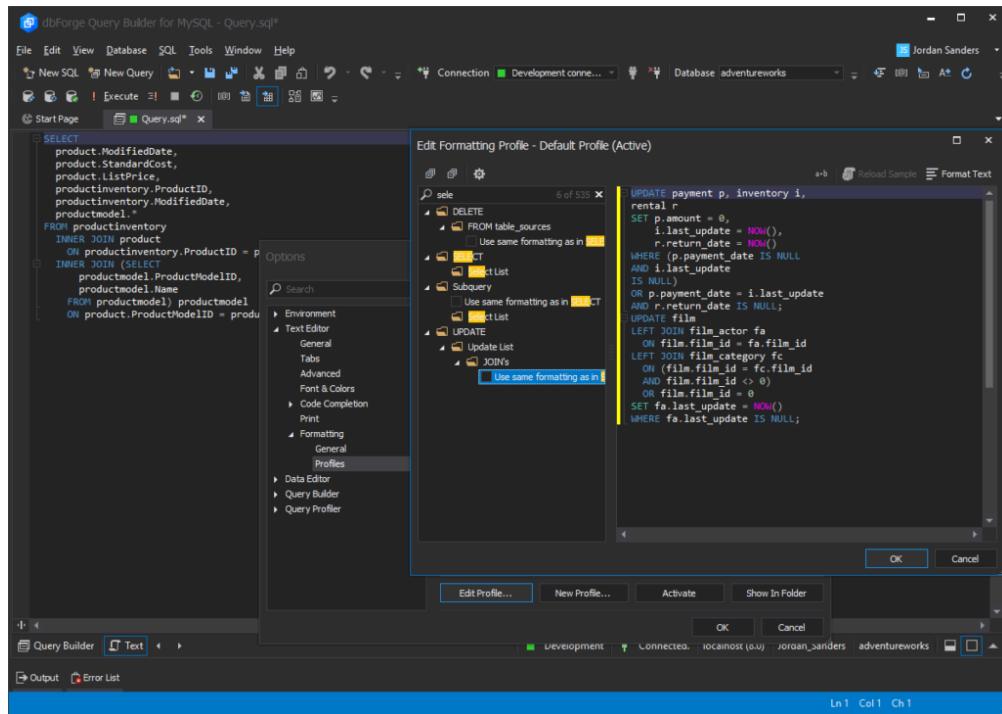
2. Nhập vào **CSV** và nhập vào ba dấu chấm bên cạnh trường **Tên tệp** để chọn tệp cần thiết.
3. Nhập vào **Tiếp theo** rồi chọn cơ sở dữ liệu mà bạn muốn nhập bảng từ menu thả xuống **Cơ sở dữ liệu** .
4. Chọn **Bảng mới** và nhập tên bảng bạn muốn nhập dữ liệu vào.
5. Để tùy chỉnh các tùy chọn khác, hãy nhấp vào **Tiếp theo** . Để nhập ngay lập tức, hãy nhấp vào **Nhập** .
6. Nhấp vào **Làm mới đối tượng** . Bạn có thể kiểm tra bảng đã nhập trong cơ sở dữ liệu được chỉ định:



3.3 Tao chế độ xem MySQL

Cách tạo chế độ xem trong dbForge Studio cho MySQL

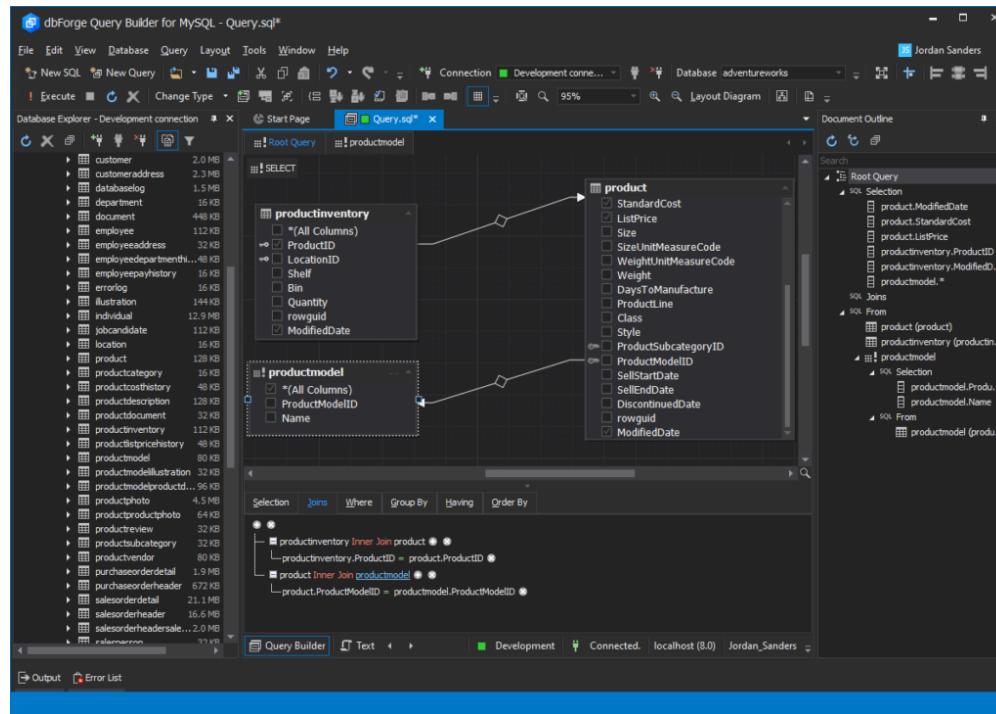
- Cách đầu tiên là viết và thực hiện truy vấn trong tài liệu SQL. Ở đây, dbForge Studio for MySQL [cung cấp tính năng hoàn thành mã theo ngữ cảnh, kiểm tra cú pháp tự động, đoạn mã, điều hướng nhanh qua các tập lệnh lớn](#) và các cấu hình định dạng có thể tùy chỉnh. Nói cách khác, bạn có mọi tính năng bạn có thể cần trong một IDE tiện lợi duy nhất.



Định dạng truy vấn trong tài liệu SQL

Nếu bạn muốn thành thạo chức năng này với hướng dẫn từng bước dễ dàng, hãy thoải mái kiểm tra phần [Viết và thực thi câu lệnh SQL](#) trong tài liệu của chúng tôi. Bạn sẽ tìm thấy mọi thứ ở đó, từ việc tạo tài liệu SQL mới đến việc thực thi tự động các truy vấn của bạn thông qua giao diện dòng lệnh.

- Cách thứ hai là một trong những công cụ đáng chú ý nhất của dbForge Studio — Query Builder. Nó trình bày các truy vấn của bạn dưới dạng sơ đồ, tạo ra các JOIN đã đề cập ở trên và cho phép xây dựng tương tác các câu lệnh INSERT, UPDATE và DELETE để cập nhật chế độ xem của bạn.



3.4 Khóa trong MySQL

Khóa chính trong MySQL là gì?

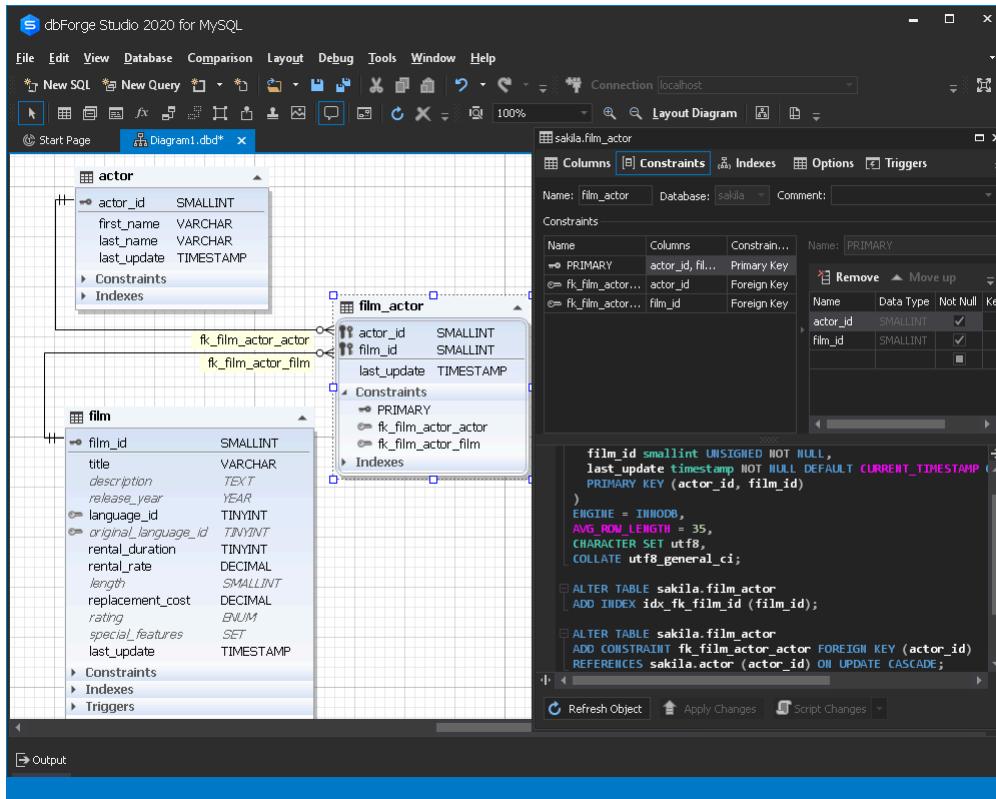
- **Khóa chính (PK)** bao gồm một cột hoặc một tổ hợp các cột có vai trò cung cấp một mã định danh duy nhất cho mỗi bản ghi trong một bảng. Các điểm chính đằng sau Khóa chính là:
 - Một bảng chỉ có thể có một Khóa chính nhưng nó có thể là Khóa **hợp thành** bao gồm nhiều cột.
 - Vì nó xác định duy nhất từng hàng trong bảng, nên một cột khóa chính không thể có giá trị NULL.
 - Một cột khóa chính phải chứa các giá trị duy nhất. Nếu Khóa chính là hợp thành, tổ hợp các giá trị trong các cột khóa chính phải là duy nhất.
 - Kiểu dữ liệu số nguyên là lựa chọn tốt nhất cho các cột khóa chính vì MySQL hoạt động nhanh hơn với các số nguyên.

Khóa ngoại trong MySQL là gì?

- Khóa ngoại là một cột hoặc một nhóm các cột cho phép tham chiếu chéo dữ liệu liên quan giữa các bảng trong cơ sở dữ liệu. Mỗi quan hệ khóa ngoại bao gồm một bảng cha (chứa các giá trị cột ban đầu) và một bảng con (với các giá trị cột tham chiếu đến các giá trị của bảng cha). Một ràng buộc khóa ngoại được định nghĩa trên bảng con.

Sự khác biệt giữa Khóa chính và Khóa ngoại

- Khóa ngoại bao gồm một cột hoặc một tập hợp các cột trong một bảng cơ sở dữ liệu dùng để liên kết hai bảng. Thông thường, một cột *Khóa ngoại* trong bảng con là *Khóa chính* trong bảng cha. Không giống như *Khóa chính* và *Khóa duy nhất*, [Khóa ngoại MySQL](#) có thể chấp nhận nhiều giá trị NULL. Bạn có thể có nhiều hơn một *Khóa ngoại* trong một bảng. Một điểm khác biệt quan trọng khác cần xem xét là *Khóa ngoại* không tự động tạo chỉ mục, nhóm hoặc không nhóm. Bạn phải tự tạo chỉ mục trên các khóa ngoại.

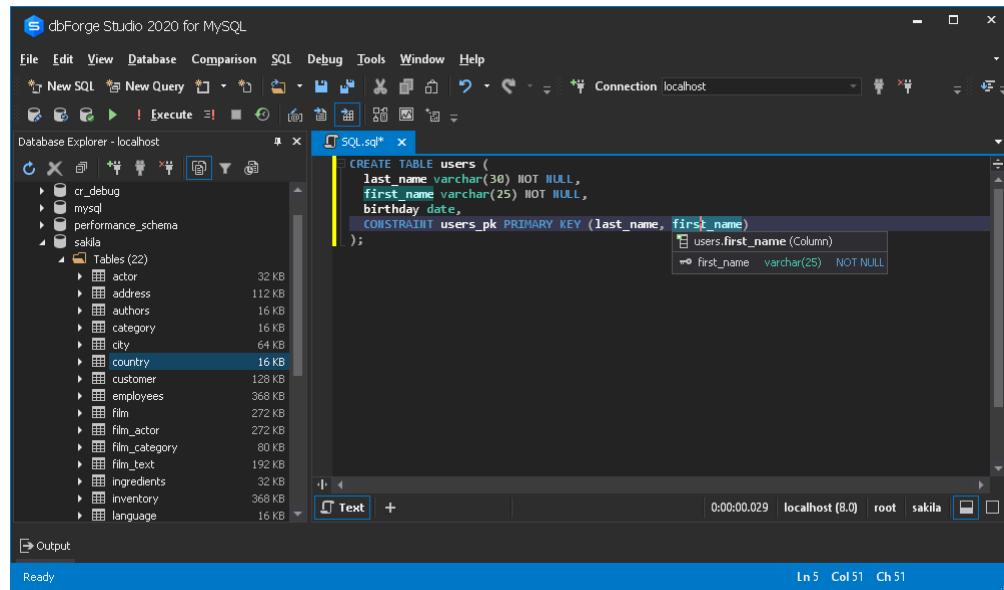


Cách tạo bảng trong MySQL với Khóa chính

Thông thường, khóa chính cho một bảng được định nghĩa trong [câu lệnh CREATE TABLE](#) . Cú pháp MySQL để tạo một bảng có khóa chính như sau:

```
CREATE TABLE table_name
(
    column1 column_definition,
    column2 column_definition,
    ...
    CONSTRAINT [constraint_name]
    PRIMARY KEY [ USING BTREE | HASH ] (column1, column2, ... column_n)
);
```

Ảnh chụp màn hình bên dưới minh họa một ví dụ thực tế về việc tạo bảng có Khóa chính trong MySQL với sự trợ giúp của dbForge Studio for MySQL. Trong số [các tính năng hữu ích](#) khác, IDE cho phép lấy **thông tin đối tượng nhanh chóng** và **kiểm tra cú pháp** ngay lập tức.



Cách tạo khóa ngoại trong MySQL

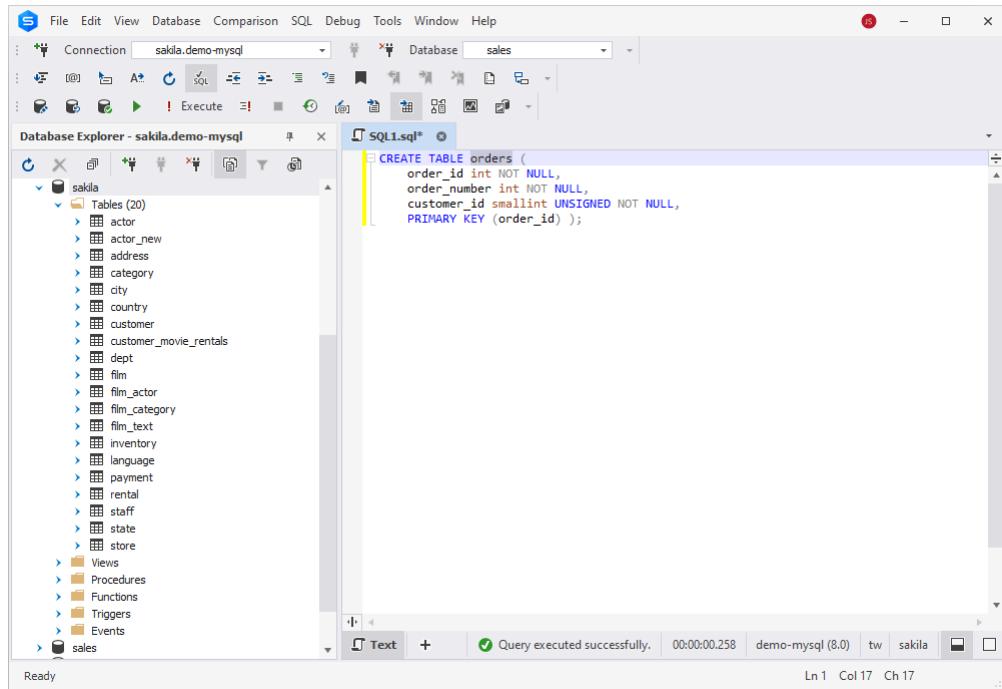
Cách tạo một bảng mới với khóa ngoại

Để làm rõ ngay từ đầu, chúng ta hãy minh họa cú pháp bằng một ví dụ dễ hiểu.

Ví dụ, hãy tưởng tượng rằng chúng ta có một bảng hiện có tên là *customer* với khóa chính là *customer_id*. Sau đó, chúng ta cần tạo một bảng thứ hai tên là *orders* và chúng ta muốn đảm bảo rằng mỗi order trong đó được liên kết với một customer hợp lệ. Vì vậy, khi chúng ta viết câu lệnh [CREATE TABLE](#) cho *orders*, chúng ta thêm một khóa ngoại tham chiếu đến cột *customer_id* của bảng *customer*. Đây là giao diện của nó.

```
CREATE TABLE orders (
    order_id int NOT NULL,
    order_number int NOT NULL,
    customer_id smallint UNSIGNED NOT NULL,
    PRIMARY KEY (order_id));
```

Để tạo bảng **đơn hàng**, chúng ta chỉ cần chạy truy vấn sau trong trình soạn thảo SQL của Studio.



The screenshot shows the MySQL Workbench interface. The 'Database Explorer' on the left lists the 'sakila' database with 20 tables. The 'SQL' tab on the right contains the following SQL code:

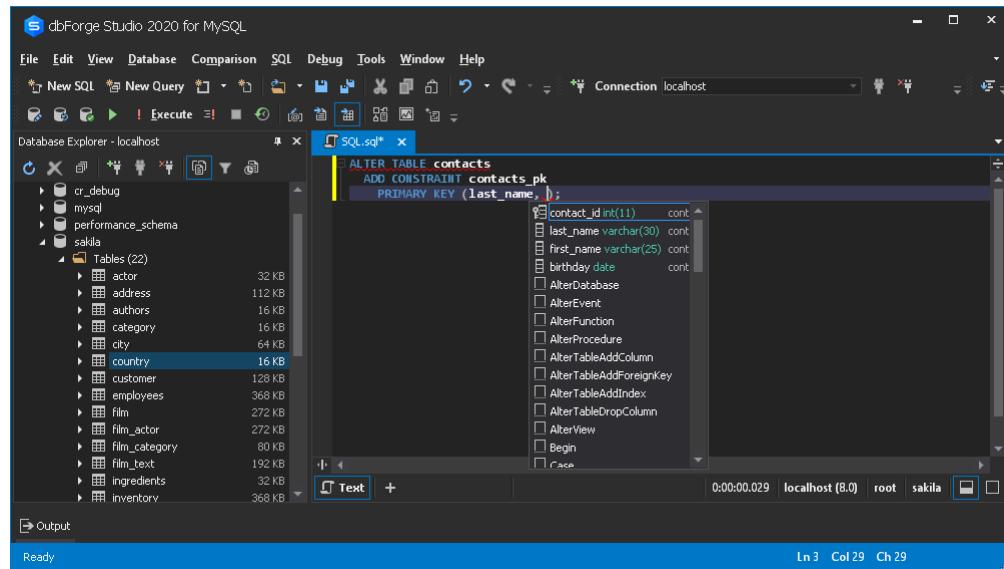
```
CREATE TABLE orders (
    order_id int NOT NULL,
    order_number int NOT NULL,
    customer_id smallint UNSIGNED NOT NULL,
    PRIMARY KEY (order_id));
```

The status bar at the bottom indicates 'Query executed successfully.' and '00:00:00.258'.

Cách thêm Khóa chính vào bảng hiện có trong MySQL

Bạn có thể thêm khóa chính vào bảng bằng cách sử dụng câu lệnh ALTER TABLE. Cú pháp MySQL để thêm khóa chính vào bảng hiện có như sau:

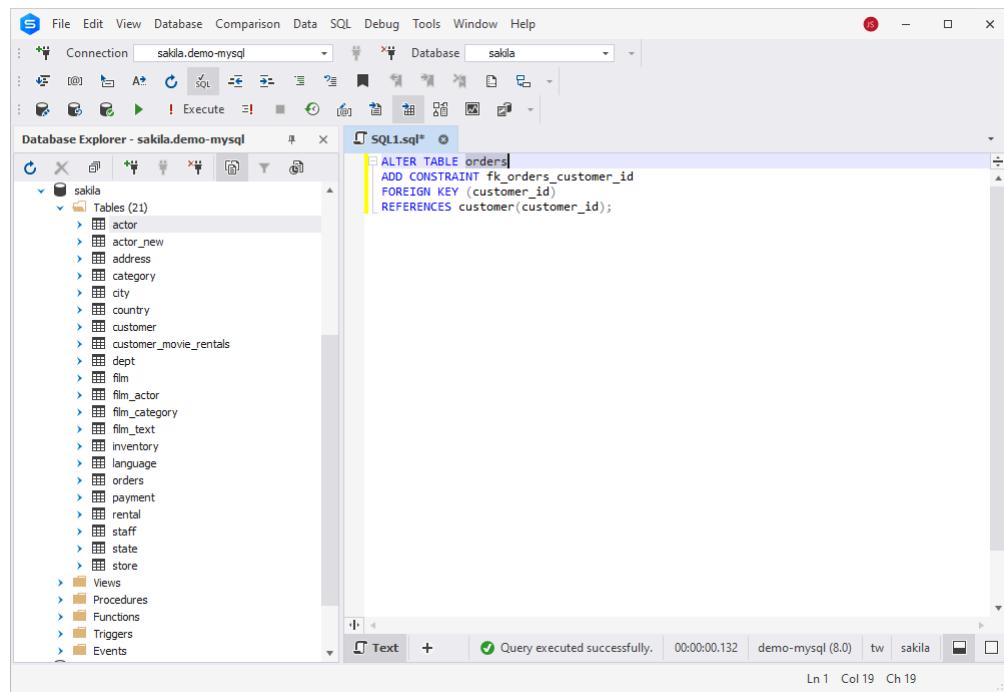
```
ALTER TABLE table_name
ADD CONSTRAINT [ constraint_name ]
PRIMARY KEY [ USING BTREE | HASH ] (column1, column2, ... column_n)
```



Cách thêm khóa ngoại vào bảng hiện có

Trong trường hợp chúng ta đã có bảng *orders* hiện tại, mọi thứ thậm chí còn dễ dàng hơn. Chúng ta chỉ cần sử dụng ADD CONSTRAINT...FOREIGN KEY trong câu lệnh ALTER TABLE.

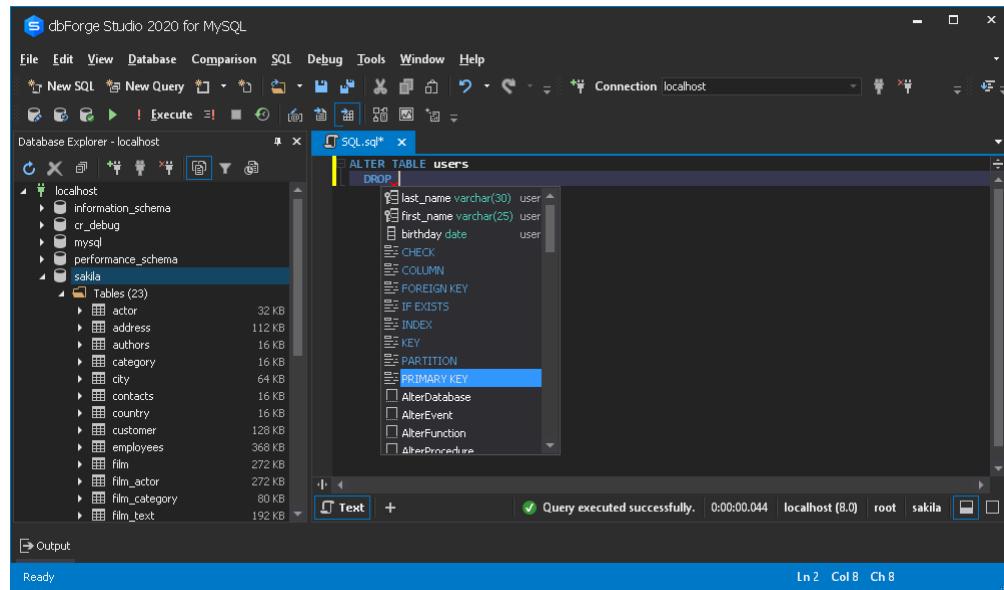
```
ALTER TABLE orders
ADD CONSTRAINT fk_orders_customer_id
FOREIGN KEY (customer_id)
REFERENCES customer(customer_id);
```



Làm thế nào để xóa Khóa trong MySQL

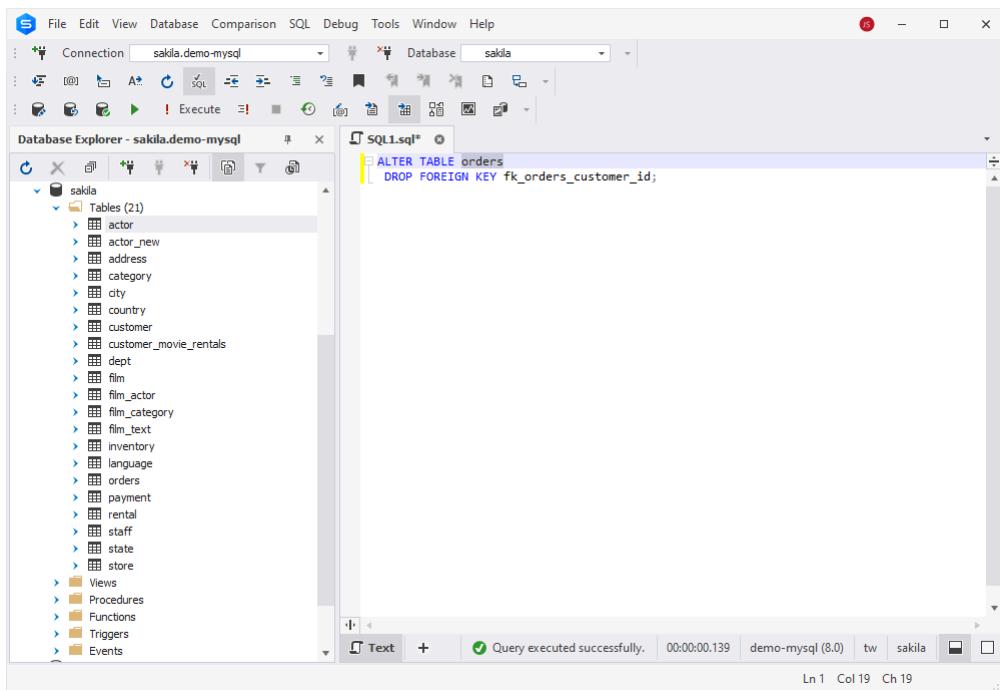
Xóa khóa chính

```
ALTER TABLE table_name
DROP PRIMARY KEY;
```



Xóa khóa ngoài

```
ALTER TABLE orders
DROP FOREIGN KEY fk_orders_customer_id;
```



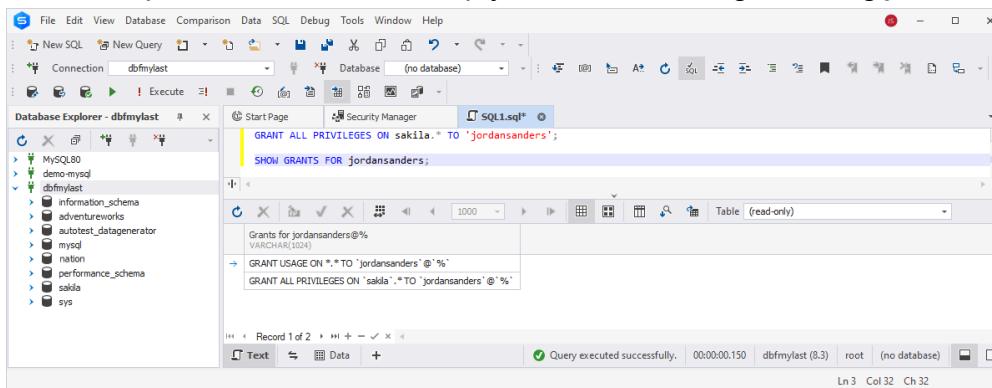
3.5 Cấp và thu hồi quyền người dùng

Cấp tất cả các quyền trong MySQL

- Bạn có thể thực hiện điều này bằng cách sử dụng câu lệnh GRANT ALL PRIVILEGES:
GRANT ALL PRIVILEGES ON . TO 'username'@'hostname';
- Thực hiện câu lệnh này sẽ cấp tất cả các quyền trên tất cả các cơ sở dữ liệu và bảng cho người dùng được chỉ định tại máy chủ đích. Tuy nhiên, nó không tự động cấp các quyền WITH GRANT OPTION hoặc PROXY.
- **WITH GRANT OPTION** cho phép người dùng chia sẻ hoặc xóa quyền khỏi người dùng khác. Khi bạn cấp quyền cho ai đó bằng mệnh đề WITH GRANT OPTION, họ có thể chia sẻ các đặc quyền đó với người dùng khác. Nhưng nếu bạn không bao gồm WITH GRANT OPTION, người dùng không thể chuyển các quyền đó thêm nữa, ngay cả khi họ đã được cấp tất cả các quyền.
- **PROXY** cho phép người dùng kết nối với tư cách là người dùng khác mà không cần cung cấp mật khẩu cho người dùng đó. Quyền này không được cấp theo mặc định với ALL PRIVILEGES. Nó phải được chỉ định rõ ràng nếu cần.

- Vì vậy, để gán các quyền WITH GRANT OPTION hoặc PROXY cùng với ALL PRIVILEGES, bạn sẽ cần phải bao gồm chúng một cách rõ ràng trong câu lệnh GRANT như sau:
GRANT ALL PRIVILEGES ON . TO 'username'@'hostname' WITH GRANT OPTION;
 - Nếu bạn muốn cấp tất cả các quyền đối với cơ sở dữ liệu đã chỉ định, hãy sửa đổi câu lệnh như sau:
GRANT ALL PRIVILEGES ON database_name.* TO 'username'@'hostname';
 - Bạn cũng có thể cung cấp quyền sở hữu đầy đủ đối với một bảng trong cơ sở dữ liệu bằng cách sử dụng câu lệnh sau:
GRANT ALL PRIVILEGES ON database_name.table_name TO 'username'@'hostname';
 - Ví dụ, gán tất cả các quyền trên cơ sở dữ liệu **sakila** cho người dùng - *jordansanders* :
GRANT ALL PRIVILEGES ON sakila.* TO 'jordansanders';
 - Sau đó, xác minh rằng quyền đã được cấp bằng cách chạy câu lệnh SHOW GRANTS FOR:
SHOW GRANTS FOR jordansanders;

Lưới kết quả hiển thị tất cả các quyền có sẵn cho người dùng *jordansanders* trên cơ sở dữ liệu **sakila** :



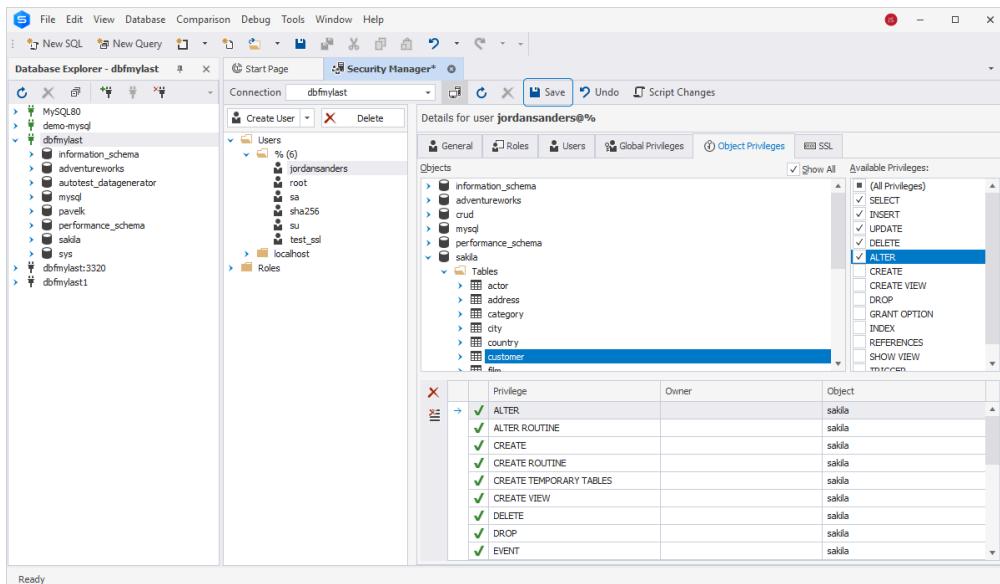
Cấp quyền bằng công cụ GUI

Security Manager cho phép cấp quyền toàn cục và đối tượng. Bây giờ chúng ta hãy xem cách cấp quyền cho đối tượng cơ sở dữ liệu. Bắt đầu bằng cách mở Studio.

1. Trên ribbon, chọn **Database > Security Manager** . Thao tác này sẽ mở Security Manager.
 2. Chọn người dùng mà bạn muốn cấp quyền và đi tới tab **Quyền đối tượng** .

3. Trong cây **Đối tượng**, chọn đối tượng cơ sở dữ liệu mà bạn muốn thiết lập quyền.
 4. Trong ngăn **Quyền hạn khả dụng**, danh sách các quyền hạn bạn có thể chỉ định sẽ được hiển thị - hãy chọn hộp kiểm bên cạnh các quyền hạn bắt buộc.
- Chọn hộp kiểm bên cạnh các quyền cần thiết.
6. Trên thanh công cụ, nhập vào **Lưu** để áp dụng các thay đổi.

Lưu ý rằng lướt bên dưới cây **Đối tượng** hiển thị các đặc quyền hiện tại của người dùng được chọn (tài khoản).



Làm thế nào để thu hồi tất cả các quyền trong MySQL

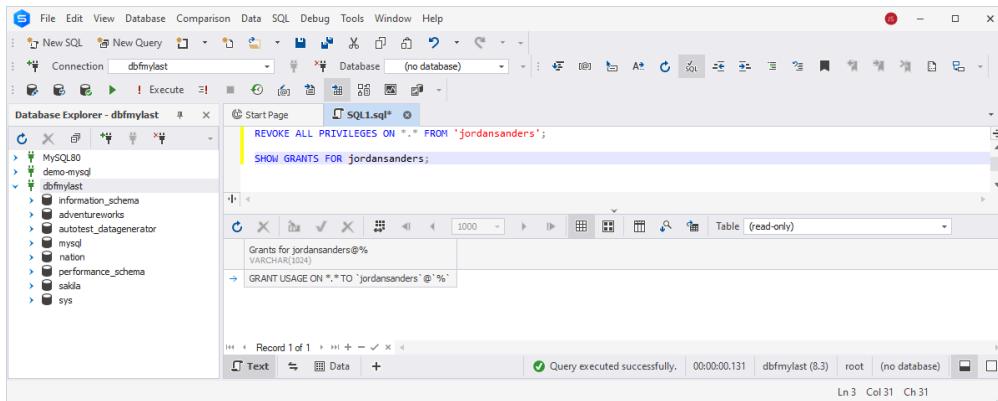
- Chúng ta đã xem xét cách cấp quyền cho người dùng và bây giờ là lúc xem xét cách thu hồi quyền đó.
 - Trong MySQL, lệnh REVOKE có thể được sử dụng để xóa quyền khỏi tài khoản người dùng. Cú pháp của lệnh như sau: `REVOKE ALL PRIVILEGES ON . FROM 'user_name'@'host_name';`
 - có nghĩa là các đặc quyền sẽ bị xóa khỏi bất kỳ cơ sở dữ liệu và đối tượng cơ sở dữ liệu nào.
 - `user_name` là tài khoản người dùng mà bạn muốn xóa quyền.
 - `hostname` là tên máy chủ hoặc địa chỉ IP mà người dùng được chỉ định đang kết nối.
- Sau khi thực hiện lệnh này, người dùng sẽ không còn bất kỳ quyền nào trong máy chủ MySQL. Tuy nhiên, điều quan trọng cần lưu ý

là lệnh này không xóa tài khoản người dùng; nó chỉ xóa các quyền được liên kết với tài khoản người dùng.

Chúng ta hãy thu hồi tất cả các quyền cho người dùng *jordansanders* mà chúng ta đã cấp trước đó, cụ thể là quyền truy cập chỉ đọc vào cơ sở dữ liệu **adventureworks** và tất cả các quyền đối với cơ sở dữ liệu **sakila**. Để thực hiện việc này, hãy thực hiện truy vấn sau:

```
REVOKE ALL PRIVILEGES ON . FROM 'jordansanders';
```

Sau đó, chạy truy vấn **SHOW GRANTS FOR** để xác minh rằng các đặc quyền đã bị xóa:



The screenshot shows the dbForge Studio for MySQL interface. The left sidebar shows the Database Explorer with the 'dbfmylast' database selected. The main pane contains a SQL editor with the following code:

```
REVOKE ALL PRIVILEGES ON . FROM 'jordansanders';
SHOW GRANTS FOR jordansanders;
```

The results of the 'SHOW GRANTS' command are displayed in a table:

Grants for jordansanders@%
VARCHAR(100)

Below the table, the status bar shows 'Query executed successfully.' and other details.

3.6 MySQL SHOW DATABASES

Lệnh MySQL SHOW DATABASES để lấy danh sách cơ sở dữ liệu

Chạy truy vấn sau để hiển thị danh sách cơ sở dữ liệu:

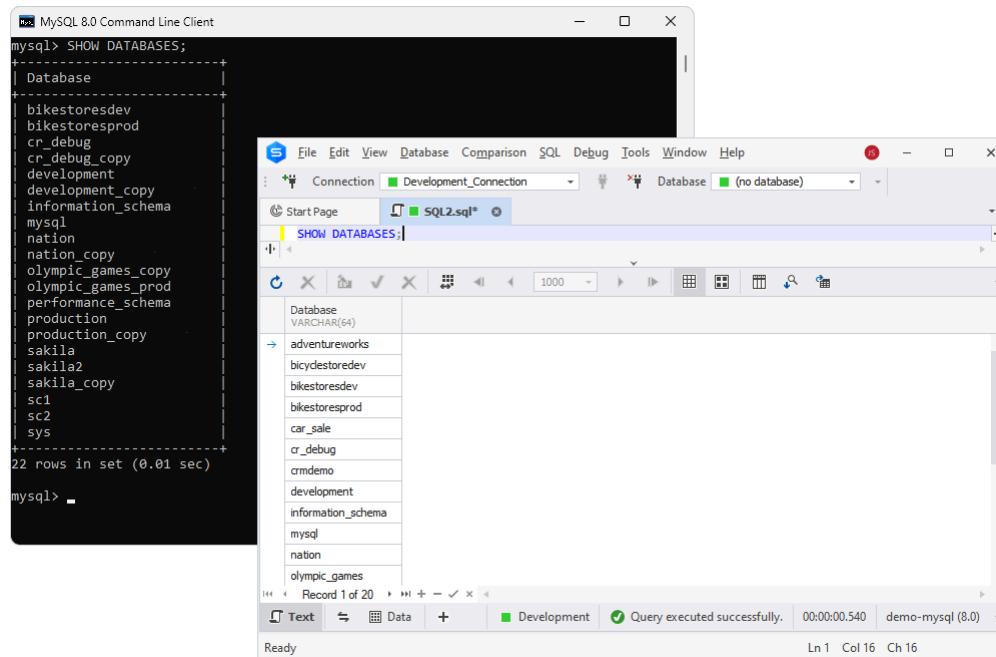
```
SHOW DATABASES;
```

Bạn có thể chạy câu lệnh này từ MySQL Command Line Client, MySQL Shell cũng như từ bất kỳ công cụ GUI nào hỗ trợ SQL—ví dụ: [dbForge Studio cho MySQL](#).

MySQL trả về kết quả trong một bảng có một cột—Database. Các cơ sở dữ liệu được sắp xếp theo thứ tự bảng chữ cái. Dòng tóm tắt cho bạn biết có bao nhiêu hàng (hoặc cơ sở dữ liệu).

Lưu ý

Trừ khi bạn có đặc quyền `SHOW DATABASES` toàn cục, bạn sẽ chỉ thấy những cơ sở dữ liệu mà bạn có một số đặc quyền. Bạn cũng có thể [cấp tất cả các đặc quyền trong MySQL](#) cho người dùng được yêu cầu để họ có thể thấy tất cả các cơ sở dữ liệu trên máy chủ.



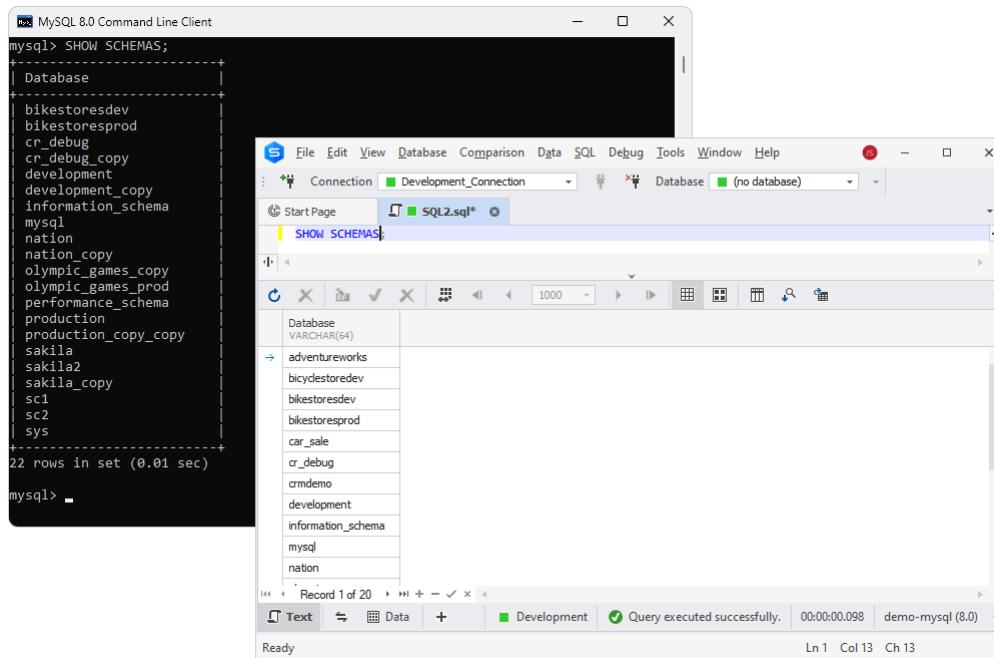
SHOW SCHEMAS để xem cơ sở dữ liệu MySQL

SHOW SCHEMAS là lệnh MySQL thay thế để xem cơ sở dữ liệu trên máy chủ lưu trữ.

SHOW SCHEMAS;

Tương tự như câu lệnh MySQL `SHOW DATABASES`, `SHOW SCHEMAS` có thể được chạy từ MySQL Command Line Client, MySQL Shell và dbForge Studio cho MySQL.

Kết quả trả về của MySQL sẽ giống hệt nhau.

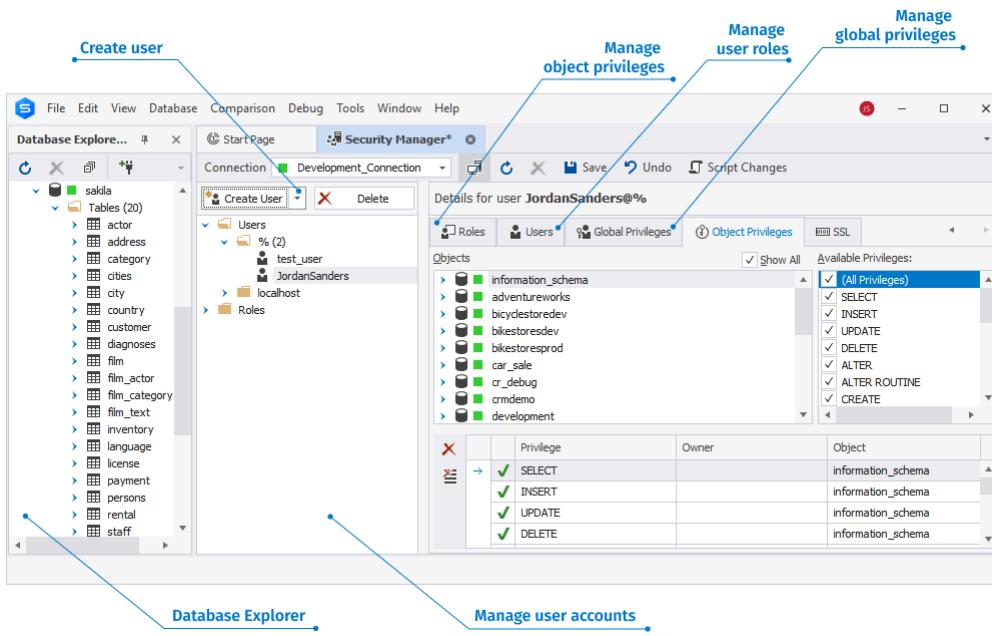


Hiển thị và quản lý danh sách cơ sở dữ liệu với dbForge Studio

Cách xem tất cả cơ sở dữ liệu MySQL trong dbForge Studio

- Sau khi bạn đã kết nối với máy chủ, các cơ sở dữ liệu được lưu trữ trên đó sẽ được hiển thị trong Database Explorer. Để xem các bảng cơ sở dữ liệu, chỉ cần mở rộng các nút cơ sở dữ liệu và bảng.
- Tất cả các lệnh chúng tôi đã đề cập ở trên đều hoạt động tốt trên Studio's Code Editor được trang bị chức năng kiểm tra cú pháp hoàn hảo và hoàn thành mã theo ngữ cảnh.

Hơn thế nữa, dbForge Studio còn đi kèm chức năng quản lý bảo mật tiên tiến cho phép kiểm soát toàn bộ tài khoản người dùng MySQL, vai trò và quyền hạn của họ.



Cách hiển thị danh sách tất cả các cơ sở dữ liệu trong MySQL Command line

1. Mở Command Prompt và điều hướng đến thư mục bin của thư mục cài đặt MySQL Server. Sau đó kết nối đến máy chủ bằng lệnh **mysql -u root -p**. Nhập mật khẩu và thực hiện lệnh **SHOW DATABASES**; mà chúng tôi đã thảo luận ở trên.
2. Mở Command Prompt và điều hướng đến thư mục bin của thư mục cài đặt MySQL Server. Sau đó chạy truy vấn sau:

```
mysql -u user -p -e "show databases;"
```

3. Mở Command Prompt và điều hướng đến thư mục bin của thư mục cài đặt MySQL Server. Sau đó chạy truy vấn:

```
mysqlshow -u user -p
```

```

C:\> Command Prompt - mysql -u root -p
C:\Program Files\MySQL\MySQL Server 8.0\bin>mysql -u root -p
Enter password: ****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 31
Server version: 8.0.30 MySQL Community Server - GPL

Copyright (c) 2000, 2022, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help.
mysql> use sakila
Database changed
mysql> SHOW DATABASES;
+-----+
| Database |
+-----+
| Database |
| bikestoresdev |
| bikestoresprod |
| cr_debug |
| cr_debug_copy |
| development |
| development_copy |
| development_copy |
| information_schema |
| mysql |
| nation |
| nation_copy |
+-----+
3 rows in set (0.00 sec)

mysql> use sakila
Database changed
mysql> SHOW TABLES;
+-----+
| Tables_in_database |
+-----+
| bikestoresdev |
| bikestoresprod |
| cr_debug |
| cr_debug_copy |
| development |
| development_copy |
| information_schema |
| mysql |
| nation |
| nation_copy |
| olympic_games_copy |
| olympic_games_prod |
| performance_schema |
| production |
| production_copy |
| sakila |
+-----+
14 rows in set (0.00 sec)

C:\> Command Prompt
C:\Program Files\MySQL\MySQL Server 8.0\bin>mysqlshow -u root -p
Enter password: ****
+-----+
| Databases |
+-----+
| bikestoresdev |
| bikestoresprod |
| cr_debug |
| cr_debug_copy |
| development |
| development_copy |
| information_schema |
| mysql |
| nation |
| nation_copy |
| olympic_games_copy |
| olympic_games_prod |
+-----+
11 rows in set (0.00 sec)
  
```

3.7 MySQL Show/List Tables

Cách hiển thị danh sách tất cả các bảng trong cơ sở dữ liệu MySQL

Chúng ta hãy bắt đầu với cú pháp cơ bản nhất. Truy vấn sau sẽ hiển thị tất cả các bảng trong cơ sở dữ liệu MySQL:

SHOW TABLES;

Để xem tất cả các bảng, bạn có thể chạy câu lệnh này từ MySQL Command Line Client, MySQL Shell cũng như từ bất kỳ công cụ GUI nào hỗ trợ SQL—ví dụ: [dbForge Studio cho MySQL](#) .

MySQL trả về kết quả trong một bảng có một cột— *Tables_in_DatabaseName* . Các bảng được sắp xếp theo thứ tự bảng chữ cái. Dòng tóm tắt cho bạn biết có bao nhiêu hàng (hoặc bảng) trong cơ sở dữ liệu đang xét.

MySQL 8.0 Command Line Client

```
mysql> use sakila;
Database changed
mysql> show tables;
+-----+
| Tables_in_sakila |
+-----+
| actor
| actor_info
| address
| category
| city
| country
| customer
| customer_list
| film
| film_actor
| film_category
| film_list
| film_text
| inventory
| language
| nicer_but_slower_film_list
| payment
| rental
| sales_by_film_category
| sales_by_store
| staff
| staff_list
| store
+-----+
23 rows in set (0.06 sec)

mysql> 
```

MySQL Workbench

SQL Editor: SQL1.sql*

SHOW TABLES

Tables_in_sakila

actor

actor_info

address

category

city

country

customer

customer_list

film

film_actor

film_category

film_list

film_text

inventory

language

nicer_but_slower_film_list

payment

rental

sales_by_film_category

sales_by_store

staff

staff_list

store

Ready

3.8 Cách đổi tên cơ sở dữ liệu MySQL

Phương pháp đổi tên bảng

Có một cách để thực hiện nhiệm vụ này khá dễ dàng. Trên thực tế, MySQL phiên bản 5.5 (và các phiên bản sau) có InnoDB Storage Engine là mặc định, có thể hữu ích.

Nói một cách ngắn gọn, chúng ta có thể áp dụng lệnh RENAME TABLE trong dấu nhắc MySQL để thay đổi tên cơ sở dữ liệu của một bảng cụ thể trong khi vẫn giữ nguyên tên bảng. Nhưng để làm như vậy, trước tiên chúng ta cần tạo một cơ sở dữ liệu mới bằng lệnh *mysqladmin* shell sau:

```
$ mysqladmin -u username -p"password" create newDbname
```

Bây giờ chúng ta đã tạo một cơ sở dữ liệu rỗng, chúng ta cần di chuyển từng bảng từ cơ sở dữ liệu cũ sang cơ sở dữ liệu mới tạo bằng lệnh sau:

```
RENAME TABLE oldDbname.table TO newDbname.table;
```

Như bạn thấy, truy vấn cho phép chúng ta di chuyển các bảng chỉ một lần, điều này không thực tế lăm đồi với các cơ sở dữ liệu lớn. Bên cạnh đó, lệnh RENAME TABLE không hoạt động đồi với các chế độ xem và trình kích hoạt. Thay vì chạy câu lệnh được đề cập ở trên, chúng ta sẽ phải xóa và tạo lại chúng.

Phương pháp đỗ

Một cách khác để đổi tên cơ sở dữ liệu trong MySQL là sử dụng lệnh *mysqldump* shell. Theo cách này, chúng ta có thể tạo một bản sao dumped của cơ sở dữ liệu và nhập toàn bộ nội dung cơ sở dữ liệu vào cơ sở dữ liệu mới. Sau đó, chúng ta có thể xóa cơ sở dữ liệu cũ nếu cần thiết.

Để bắt đầu, chúng ta cần xóa cơ sở dữ liệu cũ bằng cách chạy lệnh sau trong dấu nhắc shell:

```
$ mysqldump -u username -p"password" -R oldDbname > oldDbname.sql
```

Lệnh này tạo ra một bản sao lưu vật lý có chứa tất cả dữ liệu cơ sở dữ liệu cùng với các thủ tục và chức năng được lưu trữ.

Sau đó, chúng ta cần sử dụng lệnh trên để tạo cơ sở dữ liệu mới:

```
$ mysqladmin -u username -p"password" create newDbname
```

Cuối cùng, chúng ta cần nhập tệp dump đã tạo ở bước đầu tiên vào cơ sở dữ liệu mới:

```
$ mysql -u username -p"password" newDbname < oldDbname.sql
```

3.9 Đổi tên bảng trong cơ sở dữ liệu MySQL

Cách đổi tên bảng bằng các lệnh

MySQL RENAME TABLE

```
RENAME TABLE table_name_old TO table_name_new;
```

Trong trường hợp bạn quên tên chính xác của bảng mà bạn muốn đổi tên, bạn có thể sử dụng câu lệnh SHOW TABLES.

Sau đây là cú pháp để lấy tất cả tên bảng cơ sở dữ liệu:

```
SHOW TABLES
{FROM | IN} database_name
```

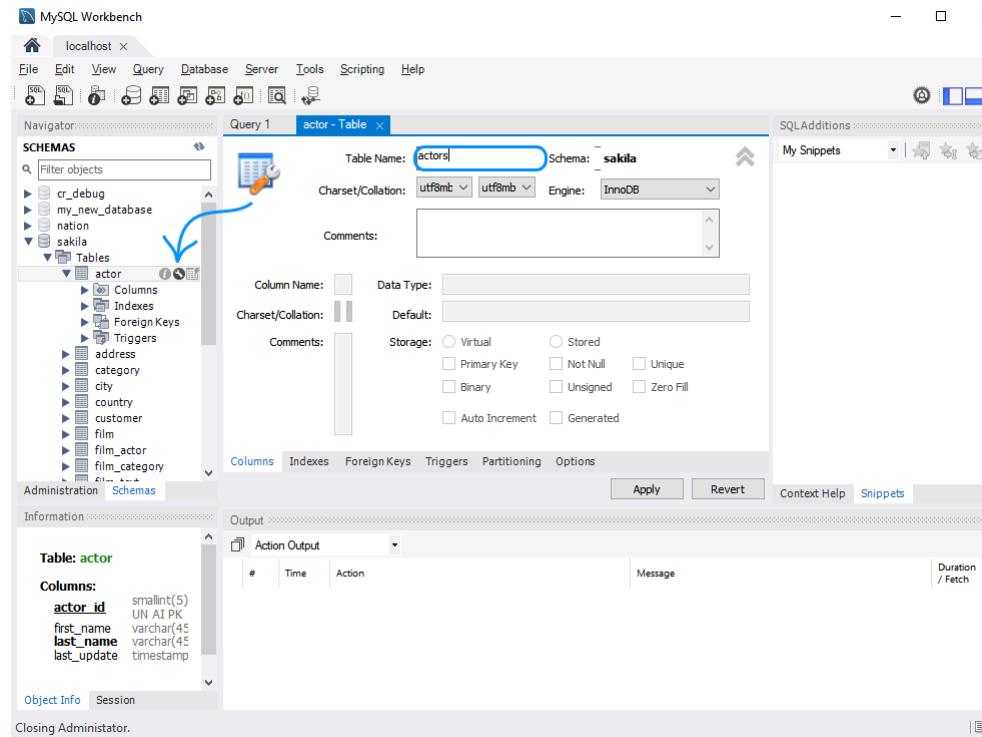
Giả sử chúng ta muốn tìm tên của tất cả các bảng trong cơ sở dữ liệu *sakila* của mình .

```
SHOW TABLES
FROM sakila
```

Cách đổi tên bảng trong MySQL Workbench

Để thay đổi tên của một bảng bằng công cụ MySQL Workbench:

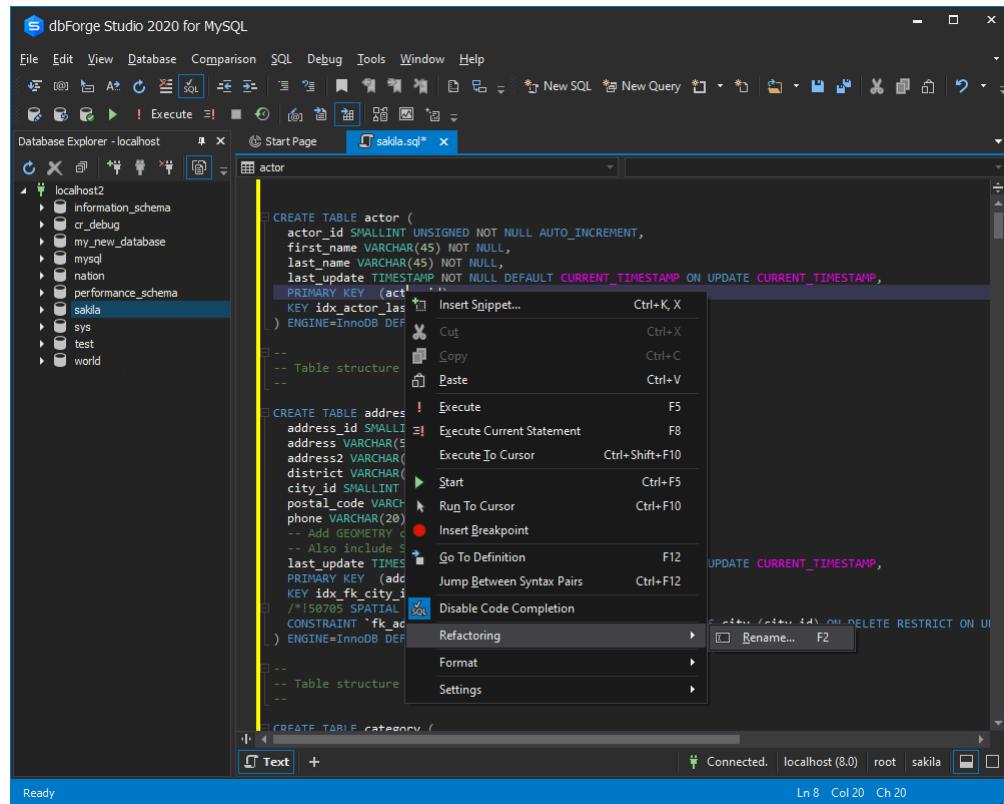
1. Trong MySQL Workbench Navigator, hãy tìm kiếm tên bảng mà bạn muốn thay đổi rồi nhấp vào tên đó.
2. Nhấp vào biểu tượng cờ lê bên cạnh bảng mà bạn muốn đổi tên.
Hoặc, nhấp chuột phải vào bảng mà bạn muốn đổi tên trong Navigator rồi nhấp vào **Alter Table** .
3. Trong cửa sổ Table Editor mở ra, hãy thay đổi tên của bảng như trong ảnh chụp màn hình bên dưới.
4. Nhấp vào **Apply** .



Cách thay đổi tên bảng trong dbForge Studio

Để đổi tên một đối tượng trong tập lệnh:

1. Nhấp chuột phải vào nó, trỏ đến **Refactoring**, rồi nhấp vào **Rename**.
Hoặc, đặt con trỏ vào đối tượng cơ sở dữ liệu và nhấn **F2**.



dbForge Studio 2020 for MySQL

File Edit View Database Comparison SQL Debug Tools Window Help

Database Explorer - localhost Start Page sakila.sql*

localhost2

information_schema
cr_debug
my_new_database
mysql
nation
performance_schema
sakila
sys
test
world

actor

```
CREATE TABLE actor (
    actor_id SMALLINT UNSIGNED NOT NULL AUTO_INCREMENT,
    first_name VARCHAR(45) NOT NULL,
    last_name VARCHAR(45) NOT NULL,
    last_update TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,
    PRIMARY KEY (actor_id),
    KEY idx_actor_last_name (last_name),
    KEY idx_actor_first_name (first_name),
    KEY idx_actor_id (actor_id),
    KEY idx_fk_city_id (city_id),
    KEY idx_fk_actor_id (actor_id),
    CONSTRAINT fk_actor_id FOREIGN KEY (city_id) REFERENCES city (city_id),
    CONSTRAINT fk_actor_id FOREIGN KEY (actor_id) REFERENCES employee (employee_id)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

-- Table structure

```
CREATE TABLE address (
    address_id SMALLINT UNSIGNED NOT NULL AUTO_INCREMENT,
    address VARCHAR(50) NOT NULL,
    address2 VARCHAR(50),
    district VARCHAR(20),
    city_id SMALLINT UNSIGNED NOT NULL,
    postal_code VARCHAR(10),
    phone VARCHAR(20),
    -- Add GEOMETRY column here
    -- Also include地理坐标
    last_update TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,
    PRIMARY KEY (address_id),
    KEY idx_fk_city_id (city_id),
    CONSTRAINT fk_address FOREIGN KEY (city_id) REFERENCES city (city_id)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

-- Table structure

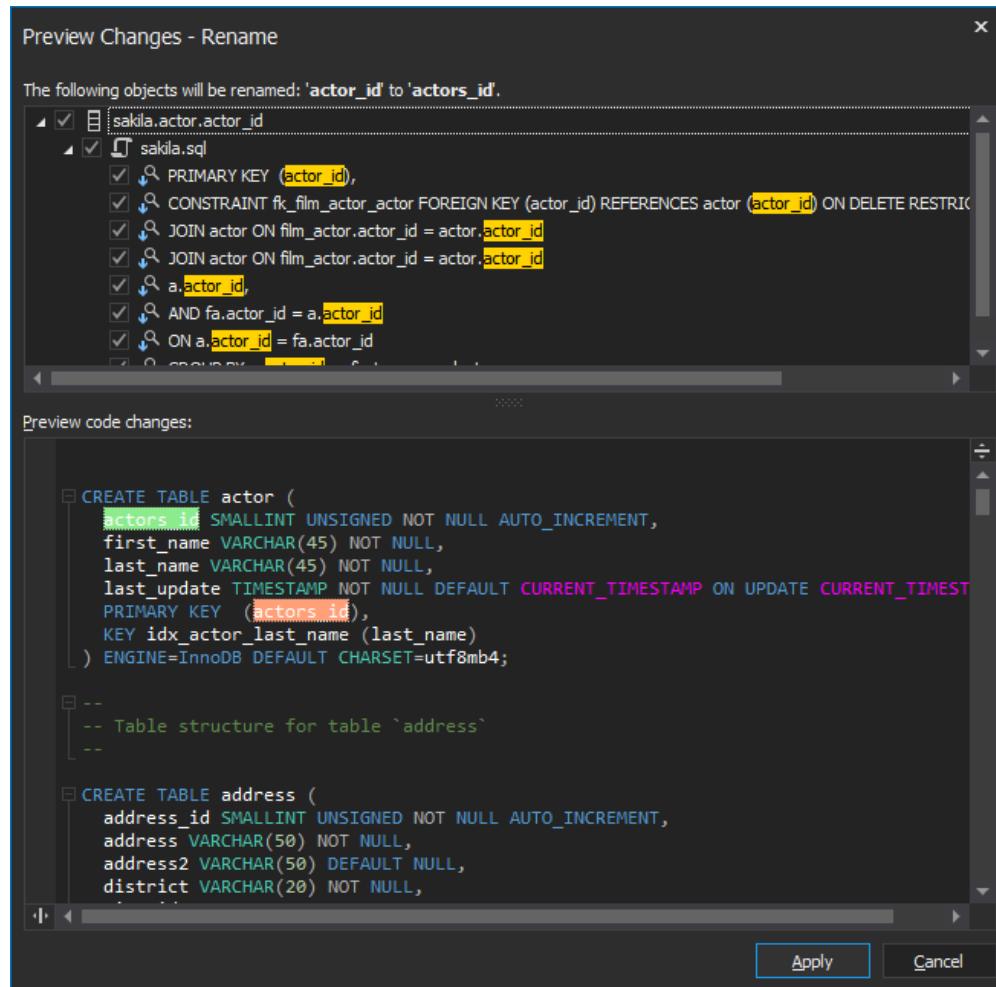
```
CREATE TABLE category (
    category_id SMALLINT UNSIGNED NOT NULL AUTO_INCREMENT,
    category_name VARCHAR(50) NOT NULL,
    description TEXT,
    last_update TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,
    PRIMARY KEY (category_id),
    KEY idx_fk_manager_id (manager_id),
    CONSTRAINT fk_category FOREIGN KEY (manager_id) REFERENCES employee (employee_id)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

Text +

Connected. localhost (8.0) root sakila

Ready Ln 8 Col 20 Ch 20

- Đổi tượng cơ sở dữ liệu sẽ được tô sáng màu xám. Đổi tên theo yêu cầu.
 - Nhấn **F2** để xem trước các thay đổi hoặc nhấn **Enter** / **Tab** để áp dụng các thay đổi mà không xem trước chúng.
- Trong hộp thoại **Xem trước các thay đổi – Đổi tên**, bạn có thể xem tất cả các tham chiếu đến đối tượng đã đổi tên.



4. Chọn các tài liệu tham khảo cần thiết và nhấn **Áp dụng**.

3.10 Thay đổi kiểu cột trong MySQL

Sửa đổi loại cột: Cú pháp cơ bản

Cú pháp cơ bản để thay đổi kiểu cột bằng ALTER TABLE lệnh:

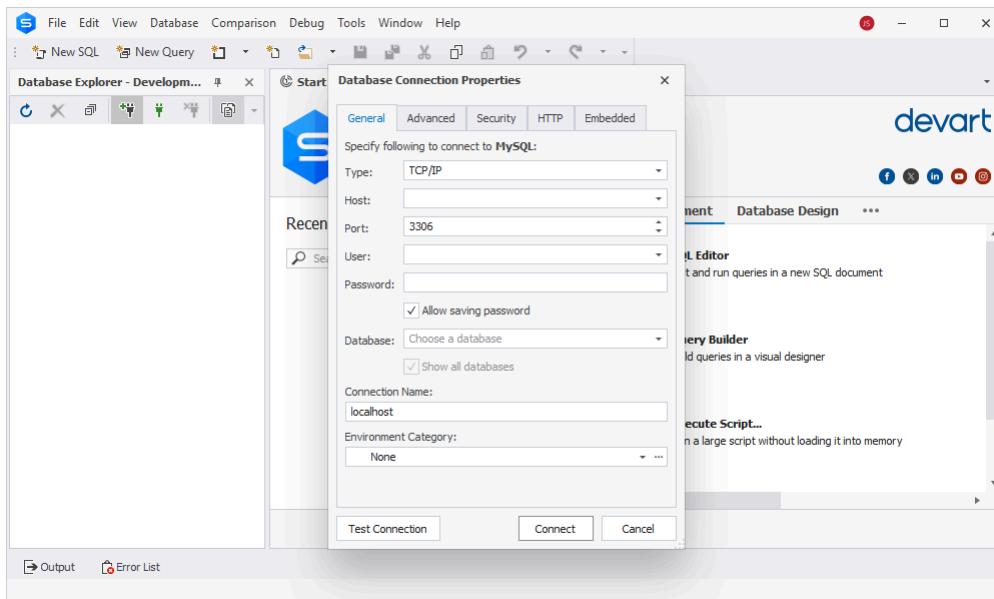
```
ALTER TABLE table_name  
MODIFY COLUMN column_name new_data_type;
```

Trong cú pháp này:

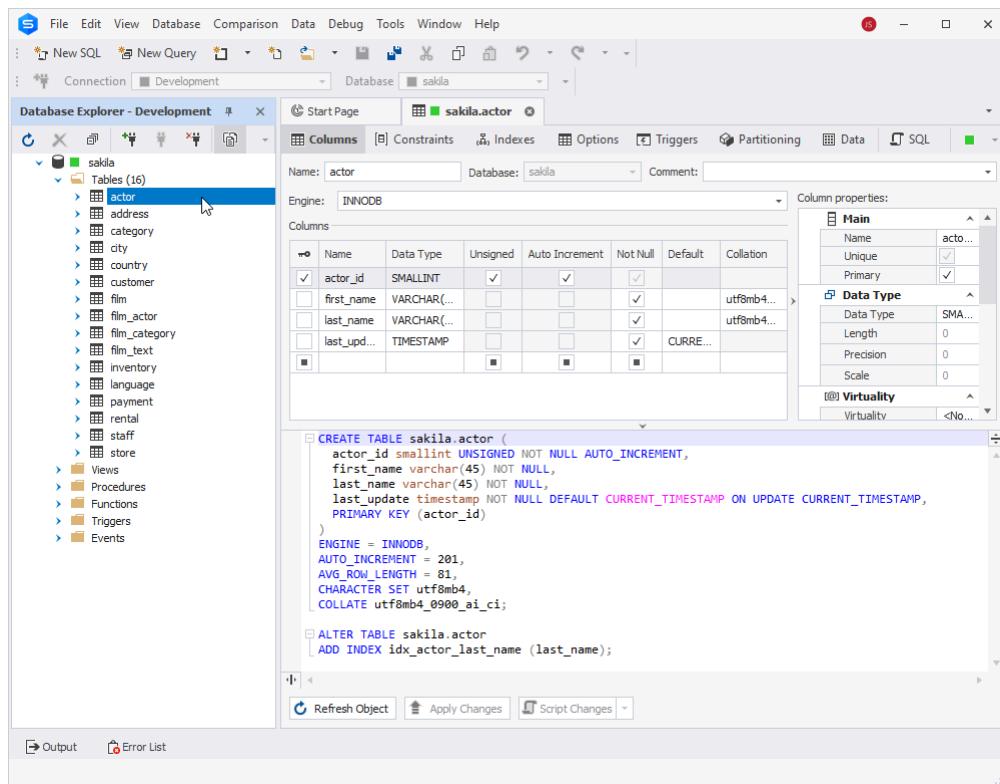
- `table_name` là tên của bảng bạn muốn sửa đổi.
- `column_name` là tên cột bạn muốn sửa đổi.
- `new_data_type` là kiểu dữ liệu mong muốn cho cột.

Thay đổi kiểu cột mà không cần mã MySQL

1. Khởi chạy dbForge Studio và kết nối với cơ sở dữ liệu MySQL của bạn.



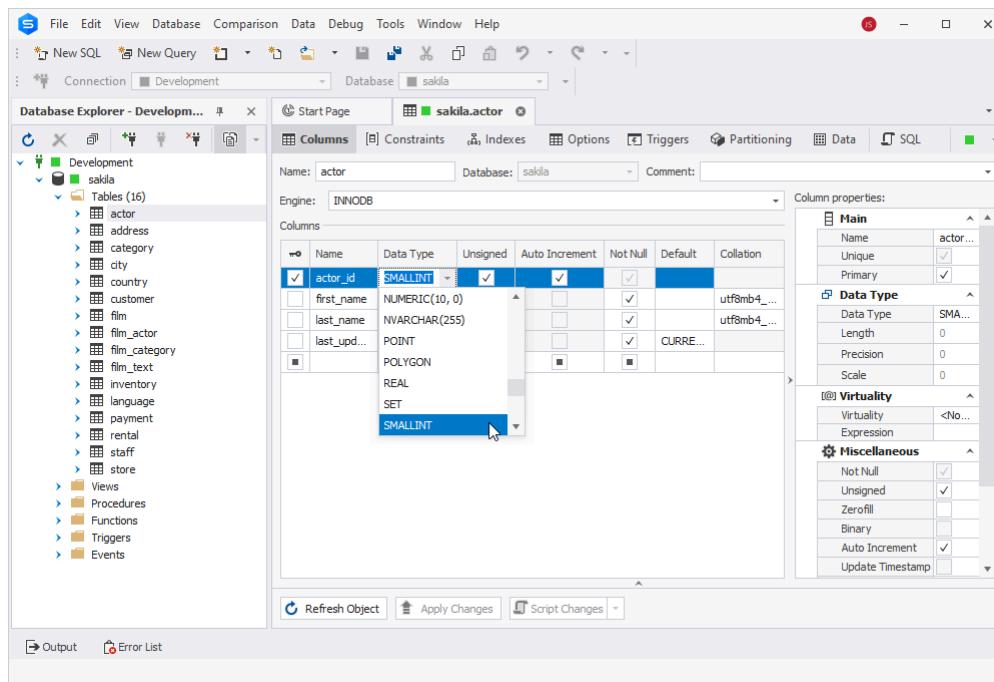
2. Trong **Database Explorer**, hãy nhấp đúp vào bảng mà bạn muốn thay đổi kiểu cột.



Trong [Trình chỉnh sửa bảng](#) mở ra, bạn sẽ thấy toàn bộ các tùy chọn bạn có thể sử dụng để chỉnh sửa dữ liệu của bảng trên lưới. Nó cho phép thêm các cột MySQL mới chỉ bằng một cú nhấp chuột, thay đổi tên cột và chỉ định các thuộc tính chính mà không cần mở bất kỳ cửa sổ bổ sung nào. Bằng cách chọn các hộp kiểm tương ứng, bạn có thể dễ dàng đặt và xóa khóa chính. Để ghi đè đối chiếu cơ sở dữ liệu mặc định, bạn có thể đặt đối chiếu khác cho một cột cụ thể. Ngoài ra, bạn có thể tạo một số ghi chú hoặc thêm một số thông tin về bảng bằng cách sử dụng trường Bình luận.

Trong bài viết này, chúng tôi chỉ tập trung vào một khía cạnh trong chức năng mở rộng của Studio: thay đổi kiểu dữ liệu cho một cột trong bảng.

3. Trong lưới, nhấp vào trường **Kiểu dữ liệu** cho cột bạn muốn thay đổi kiểu. Sau đó, chọn kiểu dữ liệu mong muốn từ danh sách thả xuống.



4. Khi hoàn tất, nhấp vào **Áp dụng thay đổi**.

3.11 Câu lệnh MySQL SHOW INDEXes

Hiển thị chỉ mục trong bảng MySQL

- Để hiển thị các chỉ mục trong cơ sở dữ liệu MySQL, hãy sử dụng câu lệnh SHOW INDEX. Nó trả về thông tin về các chỉ mục được liên kết với bảng được chỉ định trong mệnh đề FROM. Cú pháp của câu lệnh như sau:

SHOW INDEXES FROM table_name;

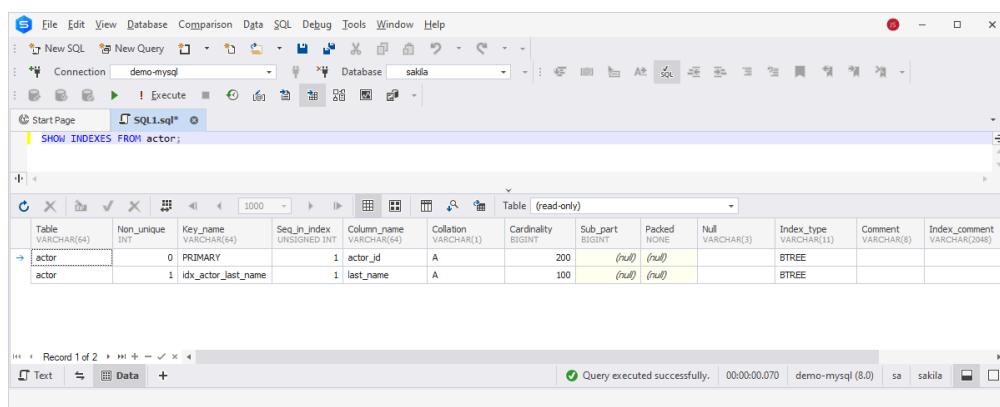
- Để lấy thông tin chỉ mục từ một cơ sở dữ liệu khác, hãy thêm mệnh đề IN vào câu lệnh:

SHOW INDEXES FROM table_name

IN database_name;

Để minh họa, chúng ta hãy xem tất cả các chỉ mục cho bảng **actor** . Truy vấn trả về các chi tiết chỉ mục sau:

- **Bảng:** Tên của bảng được chỉ định.
- **Không duy nhất:** Chỉ ra liệu chỉ mục có thể có bản sao (1) hay không (0).
- **Key_name:** Tên của chỉ mục, trong đó chỉ mục khóa chính luôn được gọi là PRIMARY.
- **Seq_in_index:** Số thứ tự của cột trong chỉ mục, bắt đầu từ 1.
- **Column_name:** Tên của cột.
- **Sắp xếp:** Thứ tự sắp xếp của cột trong chỉ mục (A theo thứ tự tăng dần, B theo thứ tự giảm dần hoặc NULL theo thứ tự chưa được sắp xếp).
- **Số lượng:** Số lượng ước tính các giá trị duy nhất trong chỉ mục.
- **Sub_part:** Tiền tố chỉ mục, hiển thị NULL cho lập chỉ mục toàn cột hoặc số ký tự được lập chỉ mục cho lập chỉ mục một phần.
- **Đóng gói:** Chỉ ra cách khóa được đóng gói.
- **Null:** Chỉ ra liệu cột có chứa giá trị NULL (CÓ) hay không (trống).
- **Index_type:** Xác định phương pháp được sử dụng cho chỉ mục: BTREE, HASH, RTREE hoặc FULLTEXT.
- **Bình luận:** Thông tin bổ sung về chỉ mục.
- **Index_comment:** Hiển thị bất kỳ bình luận nào được chỉ định khi tạo chỉ mục bằng thuộc tính COMMENT.
- **Hiển thị:** Hiển thị liệu chỉ mục có hiển thị (CÓ) hay không hiển thị với trình tối ưu hóa truy vấn (KHÔNG).
- **Biểu thức:** Biểu thị phần khóa nếu chỉ mục sử dụng biểu thức thay vì giá trị tiền tố cột hoặc cột; cột column_name là NULL trong những trường hợp như vậy.



The screenshot shows the MySQL Workbench interface with the following details:

- File Edit View Database Comparison Data SQL Debug Tools Window Help**
- Connection:** demo-mysql
- Database:** sakila
- Start Page:** SQL1.sql
- Query:** SHOW INDEXES FROM actor;
- Table:** (read-only)
- Results:**

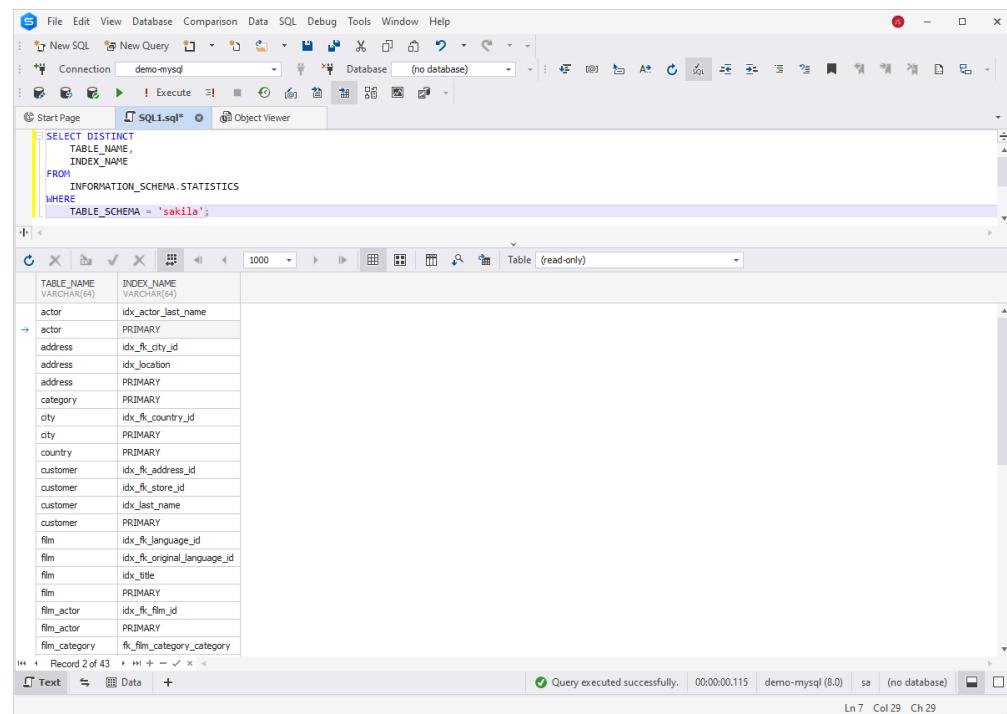
Table	Non_unique	Key_name	Seq_in_index	Column_name	Collation	Cardinality	Sub_part	Packed	Null	Index_type	Comment	Index_comment
actor	0	PRIMARY	1	actor_id	A	200	(null)	(null)	NULL	BTREE		
actor	1	idx_actor_last_name	1	last_name	A	100	(null)	(null)	NULL	BTREE		
- Bottom Status Bar:** Record 1 of 2, Text, Data, +, Query executed successfully, 00:00:00.070, demo-mysql (8.0), sa, sakila

Liệt kê tất cả các chỉ mục trong MySQL theo mã

Bây giờ, chúng ta hãy kiểm tra tất cả các chỉ mục tồn tại trong cơ sở dữ liệu MySQL cụ thể bằng cách truy vấn câu lệnh SELECT sau:

```
SELECT DISTINCT
  TABLE_NAME,
  INDEX_NAME
FROM
  INFORMATION_SCHEMA.STATISTICS
WHERE
  TABLE_SCHEMA = 'your_database';
```

Trong truy vấn, thay thế `your_database` bằng tên cơ sở dữ liệu MySQL mà bạn muốn xem chỉ mục.



The screenshot shows the MySQL Workbench interface with a query editor and a results table.

Query Editor (SQL tab):

```
SELECT DISTINCT
  TABLE_NAME,
  INDEX_NAME
FROM
  INFORMATION_SCHEMA.STATISTICS
WHERE
  TABLE_SCHEMA = 'sakila';
```

Results Table:

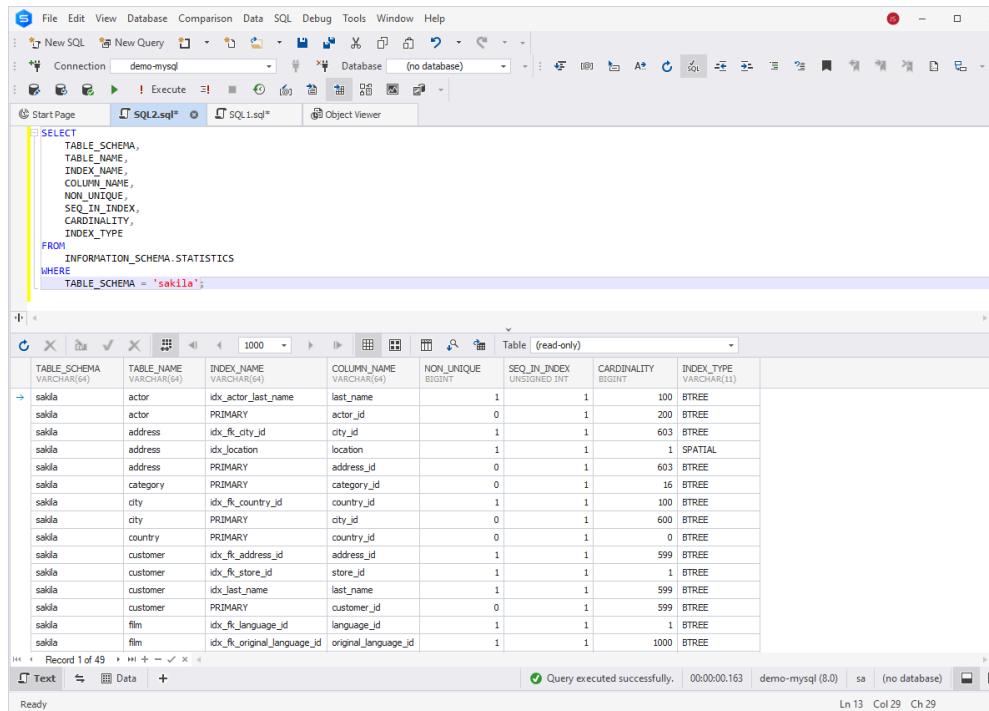
TABLE_NAME	INDEX_NAME
actor	idx_actor_last_name
actor	PRIMARY
address	idx_fk_city_id
address	PRIMARY
category	PRIMARY
city	idx_fk_country_id
city	PRIMARY
country	PRIMARY
customer	idx_fk_address_id
customer	idx_fk_store_id
customer	idx_last_name
customer	PRIMARY
film	idx_fk_language_id
film	idx_fk_original_language_id
film	idx_title
film	PRIMARY
film_actor	idx_fk_film_id
film_actor	PRIMARY
film_category	fk_film_category_category

Truy vấn trả về danh sách các chỉ mục trên tất cả các bảng trong cơ sở dữ liệu được chỉ định. Đầu ra hiển thị một lưới trong đó mỗi hàng chứa tên bảng và tên chỉ mục tương ứng.

Hơn nữa, bạn có thể lấy thông tin cụ thể về chỉ mục của tất cả các bảng. Để thực hiện việc này, hãy sửa đổi truy vấn SELECT bằng cách chọn các cột bổ sung từ bảng INFORMATION_SCHEMA.STATISTICS:

```
SELECT
  TABLE_SCHEMA,
  TABLE_NAME,
  INDEX_NAME,
  COLUMN_NAME,
  NON_UNIQUE,
  SEQ_IN_INDEX,
  CARDINALITY,
  INDEX_TYPE
FROM
  INFORMATION_SCHEMA.STATISTICS
WHERE
  TABLE_SCHEMA = 'your_database';
```

Đầu ra hiển thị thông tin chỉ mục sau: bảng nơi chỉ mục tồn tại, tên chỉ mục, cột mà chỉ mục dựa trên và tính duy nhất, cho biết chỉ mục có cho phép các giá trị trùng lặp hay không (0 cho các chỉ mục duy nhất, 1 cho các chỉ mục không duy nhất). Ngoài ra, nó hiển thị số thứ tự của cột trong chỉ mục, số lượng giá trị duy nhất trong cột được lập chỉ mục và loại chỉ mục.



```

SELECT
    TABLE_SCHEMA,
    TABLE_NAME,
    INDEX_NAME,
    COLUMN_NAME,
    NON_UNIQUE,
    SEQ_IN_INDEX,
    CARDINALITY,
    INDEX_TYPE
FROM
    INFORMATION_SCHEMA.STATISTICS
WHERE
    TABLE_SCHEMA = 'sakila';

```

TABLE_SCHEMA	TABLE_NAME	INDEX_NAME	COLUMN_NAME	NON_UNIQUE	SEQ_IN_INDEX	CARDINALITY	INDEX_TYPE
sakila	actor	idx_actor_last_name	last_name	1	1	100	BTREE
sakila	actor	PRIMARY	actor_id	0	1	200	BTREE
sakila	address	idx_fk_city_id	city_id	1	1	603	BTREE
sakila	address	idx_location	location	1	1	1	SPATIAL
sakila	address	PRIMARY	address_id	0	1	603	BTREE
sakila	category	PRIMARY	category_id	0	1	16	BTREE
sakila	city	idx_fk_country_id	country_id	1	1	100	BTREE
sakila	city	PRIMARY	city_id	0	1	600	BTREE
sakila	country	PRIMARY	country_id	0	1	0	BTREE
sakila	customer	idx_fk_address_id	address_id	1	1	599	BTREE
sakila	customer	idx_fk_store_id	store_id	1	1	1	BTREE
sakila	customer	idx_last_name	last_name	1	1	599	BTREE
sakila	customer	PRIMARY	customer_id	0	1	599	BTREE
sakila	film	idx_fk_language_id	language_id	1	1	1	BTREE
sakila	film	idx_fk_original_language_id	original_language_id	1	1	1000	BTREE

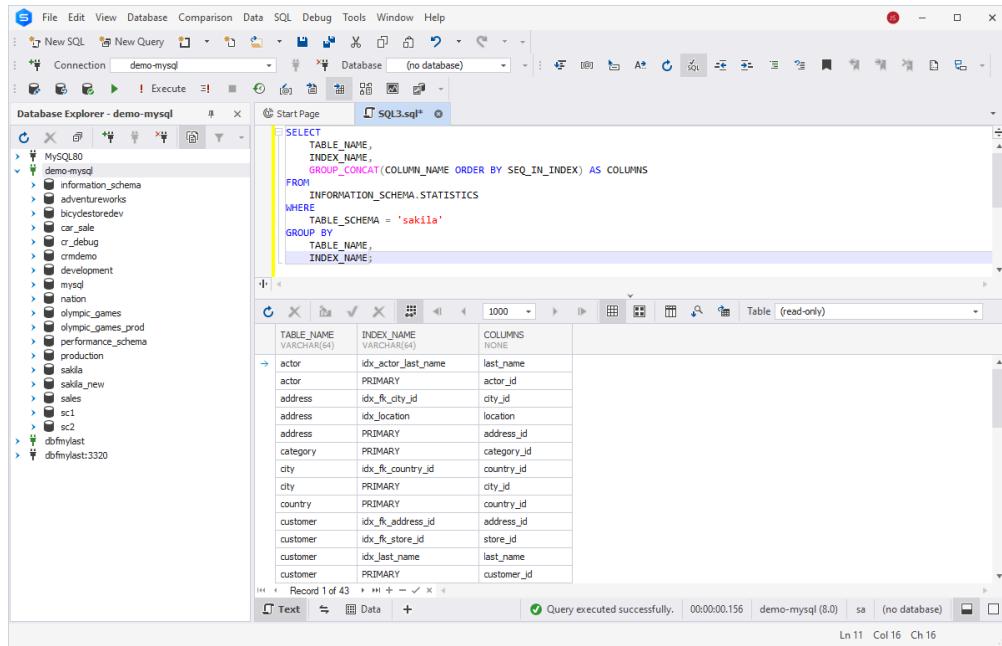
Nếu bạn cần hiển thị các cột được lập chỉ mục từ tất cả các bảng trong một cơ sở dữ liệu cụ thể, hãy sử dụng tập lệnh sau:

```

SELECT
    TABLE_NAME,
    INDEX_NAME,
    GROUP_CONCAT(COLUMN_NAME ORDER BY SEQ_IN_INDEX) AS COLUMNS
FROM
    INFORMATION_SCHEMA.STATISTICS
WHERE
    TABLE_SCHEMA = 'your_database'
GROUP BY
    TABLE_NAME,
    INDEX_NAME;

```

Trong ví dụ, hãy nhập cơ sở dữ liệu cần thiết thay vì `your_database` và thực hiện truy vấn:



The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'Database Explorer' with a tree view of MySQL databases, including 'MySQL80', 'demo-mysql', and various schema like 'information_schema', 'adventureworks', 'bicycleshopdev', 'car_sale', 'cr_debug', 'crmdemo', 'development', 'mysql', 'nation', 'olympic_games', 'olympic_games_prod', 'performance_schema', 'production', 'sakila', 'sakila_new', 'sales', 'sc1', 'sc2', 'dbfmylast', and 'dbfmylast:3320'. The main area shows a SQL editor with the following query:

```
SELECT
    TABLE_NAME,
    INDEX_NAME,
    GROUP_CONCAT(COLUMN_NAME ORDER BY SEQ_IN_INDEX) AS COLUMNS
FROM
    INFORMATION_SCHEMA.STATISTICS
WHERE
    TABLE_SCHEMA = 'sakila'
    GROUP BY
        TABLE_NAME,
        INDEX_NAME;
```

Below the query, a results table is displayed with the following data:

TABLE_NAME	INDEX_NAME	COLUMNS
actor	idx_actor_last_name	last_name
actor	PRIMARY	actor_id
address	idx_fk_city_id	city_id
address	idx_location	location
category	PRIMARY	category_id
city	idx_fk_country_id	country_id
city	PRIMARY	city_id
country	PRIMARY	country_id
customer	idx_fk_address_id	address_id
customer	idx_fk_store_id	store_id
customer	idx_last_name	last_name
customer	PRIMARY	customer_id

At the bottom of the interface, the status bar shows: 'Query executed successfully.', '00:00:00.156', 'demo-mysql (8.0)', 'sa (no database)', 'Ln 11 Col 16 Ch 16'.

3.12 Xóa các hàng trùng lặp trong MySQL

Sử dụng DELETE JOIN

Một trong những phương pháp phổ biến nhất để xóa các bản ghi trùng lặp khỏi các bảng MySQL là sử dụng `DELETE JOIN` – mệnh đề `INNER JOIN` trong câu lệnh `DELETE`. Phương pháp `DELETE JOIN` cho phép xóa các hàng khớp với các hàng khác từ cùng một bảng.

Cú pháp cơ bản của truy vấn này như sau:

```
DELETE t1
FROM table_name AS t1
INNER JOIN table_name AS t2
```

```
WHERE t1.unique_column < t2.unique_column  
AND t1.duplicate_column = t2.duplicate_column;
```

Các thông số:

t1 và *t2* là các bí danh cho *bảng* chứa các hàng trùng lặp. Các bí danh này là cần thiết để biểu diễn hai trường hợp logic của cùng một bảng để cho phép thao tác SELF JOIN.

INNER JOIN là cần thiết để tìm các hàng trùng lặp trong bảng đang đề cập.

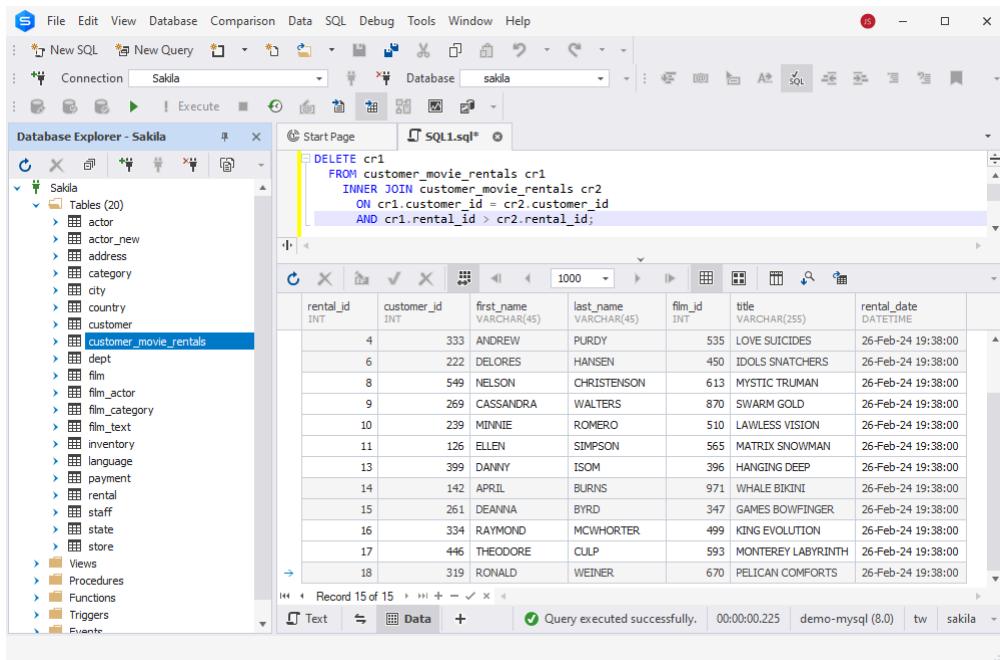
Mệnh đề WHERE là điều kiện chỉ định những hàng nào sẽ bị xóa. Trong trường hợp này, truy vấn sẽ giữ bản ghi đầu tiên và xóa các bản ghi trùng lặp tiếp theo.

Bây giờ, mục tiêu của chúng ta là xóa các bản sao khỏi bảng *customer_movie_rentals* bằng cách sử dụng câu lệnh *DELETE JOIN*. Truy vấn như sau:

```
DELETE cr1  
FROM customer_movie_rentals cr1  
INNER JOIN customer_movie_rentals cr2  
ON cr1.customer_id = cr2.customer_id  
AND cr1.rental_id > cr2.rental_id;
```

Trong truy vấn đó, **cr1** và **cr2** là các bí danh cho bảng *customer_movie_rentals* và điều kiện **cr1.rental_id > cr2.rental_id** có tác dụng chỉ giữ lại hàng có giá trị *rental_id* thấp nhất cho mỗi nhóm bản sao.

Sau đây là kết quả đầu ra của truy vấn:



The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'Database Explorer - Sakila' with a list of tables under the 'Sakila' database, including 'customer_movie_rentals'. The central area shows a SQL editor with the following query:

```
DELETE cr1
  FROM customer_movie_rentals cr1
  INNER JOIN customer_movie_rentals cr2
  ON cr1.customer_id = cr2.customer_id
  AND cr1.rental_id > cr2.rental_id;
```

Below the query is a data grid showing the results of the delete operation. The data includes columns: rental_id, customer_id, first_name, last_name, film_id, title, and rental_date. The results show 15 rows of data that have been deleted.

rental_id	customer_id	first_name	last_name	film_id	title	rental_date
4	333	ANDREW	PURDY	535	LOVE SUICIDES	26-Feb-24 19:38:00
6	222	DELORES	HANSEN	450	IDOLS SNATCHERS	26-Feb-24 19:38:00
8	549	NELSON	CHRISTENSON	613	MYSTIC TRUMAN	26-Feb-24 19:38:00
9	269	CASSANDRA	WALTERS	870	SWARM GOLD	26-Feb-24 19:38:00
10	239	MINNIE	ROMERO	510	LAWLESS VISION	26-Feb-24 19:38:00
11	126	ELLEN	SIMPSON	565	MATRIX SNOWMAN	26-Feb-24 19:38:00
13	399	DANNY	ISOM	396	HANGING DEEP	26-Feb-24 19:38:00
14	142	APRIL	BURNS	971	WHALE BIKINI	26-Feb-24 19:38:00
15	261	DEANINA	BYRD	347	GAMES BOWFINGER	26-Feb-24 19:38:00
16	334	RAYMOND	MCWHORTER	499	KING EVOLUTION	26-Feb-24 19:38:00
17	446	THEODORE	CULP	593	MONTEREY LABYRINTH	26-Feb-24 19:38:00
18	319	RONALD	WEINER	670	PELICAN COMFORTS	26-Feb-24 19:38:00

At the bottom, a message indicates the query was executed successfully.

Sử dụng bảng tạm thời

1. Tạo một bảng tạm thời mới có cùng cấu trúc với bảng gốc (có mục trùng lặp cần xóa).
2. Chèn các hàng duy nhất từ bảng gốc vào bảng tạm thời.
3. Cắt bớt bảng gốc và chèn lại các hàng duy nhất từ bảng tạm thời.

Phương pháp này hữu ích khi các hàng trùng lặp giống hệt nhau trong tất cả các giá trị cột. Sau đó, chúng ta có thể sử dụng lệnh SELECT DISTINCT để chỉ sao chép các hàng duy nhất vào bảng tạm thời.

Xem bảng kiểm tra trong ảnh chụp màn hình bên dưới. Như bạn có thể thấy, một số hàng có giá trị giống hệt nhau trong tất cả các cột.

MySQL Workbench Database Explorer showing the Sakila database. The 'Tables (20)' section is expanded, and the 'customer_movie_rentals' table is selected. The Data tab shows the first 1000 rows of the table, which lists rental details for various customers. The table has columns: rental_id, customer_id, first_name, last_name, film_id, title, and rental_date.

rental_id	customer_id	first_name	last_name	film_id	title	rental_date
1	130	CHARLOTTE	HUNTER	80	BLANKET BEVERLY	26-Feb-24 19:38:00
2	459	TOMMY	COLLAZO	333	FREAKY POCUS	26-Feb-24 19:38:00
3	408	MANUEL	MURRELL	373	GRADUATE LORD	26-Feb-24 19:38:00
4	333	ANDREW	PURDY	535	LOVE SUICIDES	26-Feb-24 19:38:00
5	222	DELORES	HANSEN	450	IDOL SNATCHERS	26-Feb-24 19:38:00
5	222	DELORES	HANSEN	450	IDOL SNATCHERS	26-Feb-24 19:38:00
6	549	NELSON	CHRISTENSON	613	MYSTIC TRUMAN	26-Feb-24 19:38:00
7	269	CASSANDRA	WALTERS	870	SWARM GOLD	26-Feb-24 19:38:00
8	239	MINNIE	ROMERO	510	LAWLESS VISION	26-Feb-24 19:38:00
9	126	ELLEN	SIMPSON	565	MATRIX SNOWMAN	26-Feb-24 19:38:00
9	126	ELLEN	SIMPSON	565	MATRIX SNOWMAN	26-Feb-24 19:38:00
11	399	DANNY	ISOM	396	HANGING DEEP	26-Feb-24 19:38:00
12	142	APRIL	BURNS	971	WHALE BIKINI	26-Feb-24 19:38:00
12	142	APRIL	BURNS	971	WHALE BIKINI	26-Feb-24 19:38:00
13	261	DEANNA	BYRD	347	GAMES BOWFINGER	26-Feb-24 19:38:00
14	334	RAYMOND	MCWHORTER	499	KING EVOLUTION	26-Feb-24 19:38:00

Đầu tiên, chúng ta tạo một bảng tạm thời và chỉ sao chép các hàng duy nhất từ bảng *customer_movie_rentals* vào đó:

-- 1. Create a temporary table

```
CREATE TEMPORARY TABLE temp_customer_movie_rentals LIKE customer_movie_rentals;
```

-- 2. Insert distinct rows into the temporary table

```
INSERT INTO temp_customer_movie_rentals
```

```
SELECT DISTINCT *
```

```
FROM customer_movie_rentals;
```

-- 3. View the temp table

```
SELECT * FROM temp_customer_movie_rentals;
```

The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'Database Explorer - Sakila' with a list of tables: actor, actor_new, address, category, city, country, customer, customer_movie_rentals, dept, film, film_actor, film_category, film_text, inventory, language, payment, rental, staff, state, store, Views, Procedures, Functions, Triggers, and Events. The main area shows a SQL editor with the following code:

```
-- 1. Create a temporary table
CREATE TEMPORARY TABLE temp_customer_movie_rentals LIKE customer_movie_rentals;

-- 2. Insert distinct rows into the temporary table
INSERT INTO temp_customer_movie_rentals
SELECT DISTINCT *
FROM customer_movie_rentals;

-- 3. View the temp table
SELECT * FROM temp_customer_movie_rentals;
```

Below the code, a data grid displays the 15 rows of data from the temporary table:

rental_id	customer_id	first_name	last_name	film_id	title	rental_date
1	130	CHARLOTTE	HUNTER	80	BLANKET BEVERLY	26-Feb-24 19:38:00
2	459	TOMMY	COLLAZO	333	FREAKY POCUS	26-Feb-24 19:38:00
3	408	MANUEL	MURRELL	373	GRADUATE LORD	26-Feb-24 19:38:00
4	333	ANDREW	PURDY	535	LOVE SUICIDES	26-Feb-24 19:38:00
5	222	DELORES	HANSEN	450	IDOLS SNATCHERS	26-Feb-24 19:38:00
6	549	NELSON	CHRISTENSON	613	MYSTIC TRUMAN	26-Feb-24 19:38:00
7	269	CASSANDRA	WALTERS	870	SWARM GOLD	26-Feb-24 19:38:00
8	239	MINNIE	ROMERO	510	LAWLESS VISION	26-Feb-24 19:38:00
9	126	ELLEN	SIMPSON	565	MATRIX SNOWMAN	26-Feb-24 19:38:00

At the bottom of the SQL editor, the status bar shows 'Query executed successfully.' and the execution time '00:00:01.452'.

Chúng ta có thể thấy rằng bảng tạm thời chỉ có 15 hàng và tất cả đều là duy nhất. Bây giờ chúng ta có thể chuyển tập dữ liệu này trở lại bảng gốc.

-- 1. Truncate the original table

```
TRUNCATE TABLE customer_movie_rentals;
```

-- 2. Insert records back into the original table

```
INSERT INTO customer_movie_rentals
```

```
SELECT * FROM temp_customer_movie_rentals;
```

-- 3. View the original table

```
SELECT * FROM customer_movie_rentals;
```

```

-- 1. Truncate the original table
TRUNCATE TABLE customer_movie_rentals;

-- 2. Insert records back into the original table
INSERT INTO customer_movie_rentals
SELECT * FROM temp_customer_movie_rentals;

-- 3. View the original table
SELECT * FROM customer_movie_rentals;

```

rental_id	customer_id	first_name	last_name	film_id	title	rental_date
1	130	CHARLOTTE	HUNTER	80	BLANKET BEVERLY	26-Feb-24 19:38:00
2	459	TOMMY	COLLAZO	333	FREAKY POCUS	26-Feb-24 19:38:00
3	408	MANUEL	MURRELL	373	GRADUATE LORD	26-Feb-24 19:38:00
4	333	ANDREW	PURDY	535	LOVE SUICIDES	26-Feb-24 19:38:00
5	222	DELORES	HANSEN	450	IDOLS SNATCHERS	26-Feb-24 19:38:00
6	549	NELSON	CHRISTENSON	613	MYSTIC TRUMAN	26-Feb-24 19:38:00
7	269	CASSANDRA	WALTERS	870	SWARM GOLD	26-Feb-24 19:38:00
8	239	MINNIE	ROMERO	510	LAWLESS VISION	26-Feb-24 19:38:00
9	126	ELLEN	SIMPSON	565	MATRIX SNOWMAN	26-Feb-24 19:38:00
11	399	DANNY	ISOM	396	HANGING DEEP	26-Feb-24 19:38:00

Sử dụng GROUP BY với tổng hợp

1. Xác định tiêu chí để xác định các hàng trùng lặp. Ví dụ, khi bạn có một tập hợp các hàng có giá trị giống hệt nhau trong tất cả các cột có liên quan, các hàng này là hàng trùng lặp.
2. Xác định tiêu chí cho hàng cần giữ lại khi xóa các mục trùng lặp. Thông thường, đó là mã định danh duy nhất nhỏ nhất hoặc lớn nhất.
3. Xóa tất cả các hàng ngoại trừ những hàng khớp với tiêu chí cụ thể.

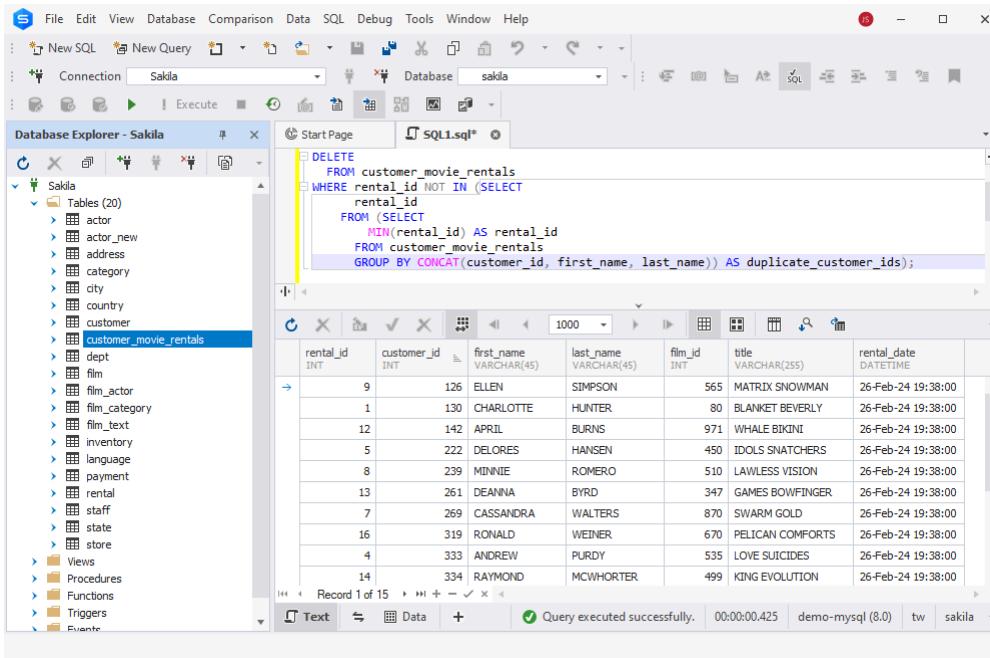
Tập lệnh bên dưới sử dụng hàm MIN() tổng hợp để xác định mã định danh duy nhất nhỏ nhất cho nhóm trùng lặp, sau đó xác định phạm vi giá trị *rental_id* cho các hàng duy nhất. Cuối cùng, truy vấn xóa các hàng khớp với tiêu chí không duy nhất.

Đoạn mã bên dưới sẽ xóa các hàng trùng lặp khỏi bảng *customer_movie_rentals* bằng cách sử dụng mệnh đề GROUP BY với hàm tổng hợp MIN():

```

DELETE
FROM customer_movie_rentals
WHERE rental_id NOT IN (SELECT
    rental_id
FROM (SELECT
    MIN(rental_id) AS rental_id
    FROM customer_movie_rentals
    GROUP BY CONCAT(customer_id, first_name, last_name)) AS duplicate_customer_ids);

```



The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'Database Explorer - Sakila' with the 'Tables (20)' section expanded, showing the 'customer_movie_rentals' table. The main area shows the SQL editor with the DELETE query and its results. The results table shows 15 rows of data from the sakila database.

rental_id	customer_id	first_name	last_name	film_id	title	rental_date
9	126	ELLEN	SIMPSON	565	MATRIX SNOWMAN	26-Feb-24 19:38:00
1	130	CHARLOTTE	HUNTER	80	BLANKET BEVERLY	26-Feb-24 19:38:00
12	142	APRIL	BURNS	971	WHALE BIKINI	26-Feb-24 19:38:00
5	222	DELORES	HANSEN	450	IDOLS SNATCHERS	26-Feb-24 19:38:00
8	239	MINNIE	ROMERO	510	LAWLESS VISION	26-Feb-24 19:38:00
13	261	DEANNA	BYRD	347	GAMES BOWFINGER	26-Feb-24 19:38:00
7	269	Cassandra	WALTERS	870	SWARM GOLD	26-Feb-24 19:38:00
16	319	RONALD	WEINER	670	PELICAN COMFORTS	26-Feb-24 19:38:00
4	333	ANDREW	PURDY	535	LOVE SUICIDES	26-Feb-24 19:38:00
14	334	RAYMOND	MCWHORTER	499	KING EVOLUTION	26-Feb-24 19:38:00

Sử dụng hàm cửa sổ ROW_NUMBER()

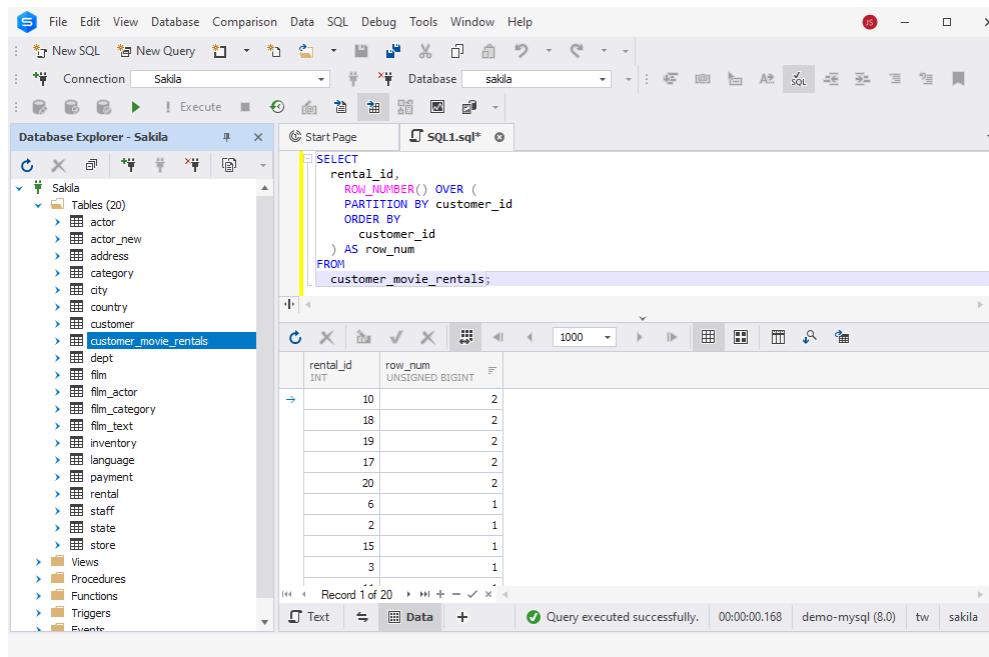
- Hàm ROWNUMBER() là một trong [những hàm cửa sổ MySQL](#) cơ bản, gán các số duy nhất cho mỗi hàng, bắt đầu từ 1 và tiếp theo là tuần tự. Nếu chúng ta có các bản ghi trùng lặp, các số duy nhất của chúng sẽ > 1. Điều này cho phép chúng ta nhanh chóng xác

định các bản ghi trùng lặp.

Hãy áp dụng `ROW_NUMBER()` vào bảng `customer_movie_rentals` của chúng ta :

```
SELECT
    rental_id,
    ROW_NUMBER() OVER (
        PARTITION BY customer_id
        ORDER BY customer_id
    ) AS row_num
FROM customer_movie_rentals;
```

- Đầu ra cho chúng ta thấy danh sách tất cả các giá trị `rental_id` và chúng ta có thể nhanh chóng xác định các giá trị trùng lặp bằng cách sắp xếp kết quả theo cột `row_num` :



The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'Database Explorer - Sakila' with the 'customer_movie_rentals' table selected. The main area shows the SQL query:

```
SELECT
    rental_id,
    ROW_NUMBER() OVER (
        PARTITION BY customer_id
        ORDER BY
            customer_id
    ) AS row_num
FROM
    customer_movie_rentals;
```

The results pane displays the following data:

rental_id	row_num
10	2
18	2
19	2
17	2
20	2
6	1
2	1
15	1
3	1

At the bottom, the status bar shows 'Query executed successfully.' and 'demo-mysql (8.0)'. The results pane also indicates 'Record 1 of 20'.

- Bây giờ chúng ta có thể xóa các bản sao bằng lệnh `DELETE` với truy vấn phụ:

```

DELETE
  FROM customer_movie_rentals
 WHERE rental_id IN (SELECT
    rental_id
  FROM (SELECT
    rental_id,
    ROW_NUMBER() OVER (
      PARTITION BY customer_id
      ORDER BY customer_id
    ) AS row_num
  FROM customer_movie_rentals cmr) t
 WHERE row_num > 1);

```

Kết quả đầu ra cho thấy không có dữ liệu trùng lặp trong bảng.

The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'Database Explorer - Sakila' with the 'Tables (20)' section expanded, showing the 'customer_movie_rentals' table selected. The main area contains the SQL query:

```

DELETE
  FROM customer_movie_rentals
 WHERE rental_id IN (SELECT
    rental_id
  FROM (SELECT
    rental_id,
    ROW_NUMBER() OVER (
      PARTITION BY customer_id
      ORDER BY customer_id
    ) AS row_num
  FROM customer_movie_rentals cmr) t
 WHERE row_num > 1);

```

The results pane shows the deleted rows:

rental_id	customer_id	first_name	last_name	film_id	title	rental_date
9	126	ELLEN	SIMPSON	565	MATRIX SNOWMAN	26-Feb-24 19:38:00
1	130	CHARLOTTE	HUNTER	80	BLANKET BEVERLY	26-Feb-24 19:38:00
12	142	APRIL	BURNS	971	WHALE BIKINI	26-Feb-24 19:38:00
5	222	DELORES	HANSEN	450	IDOLS SNATCHERS	26-Feb-24 19:38:00
8	239	MINNIE	ROMERO	510	LAWLESS VISION	26-Feb-24 19:38:00
13	261	DEANNA	BYRD	347	GAMES BOWFINGER	26-Feb-24 19:38:00
7	269	CASSANDRA	WALTERS	870	SWARM GOLD	26-Feb-24 19:38:00

At the bottom, the status bar shows 'Query executed successfully.' and the execution time '00:00:00.379'.

4.Tối ưu hóa hiệu suất MySQL

4.1 MySQL Query EXPLAIN Plan

MySQL EXPLAIN là gì?

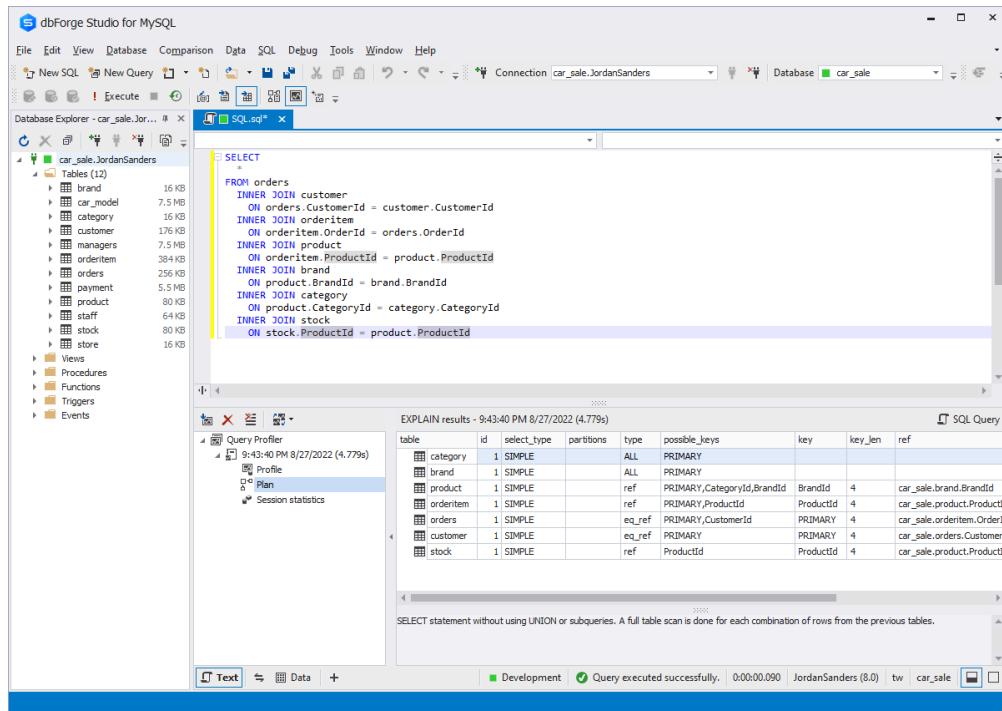
Câu lệnh EXPLAIN cung cấp thông tin chi tiết về hiệu suất truy vấn, chẳng hạn như ID truy vấn, loại, phân vùng, loại JOIN và cách sử dụng chỉ mục. Nó cho phép bạn phân tích và cải thiện các truy vấn chạy chậm.

EXPLAIN có thể được sử dụng với các truy vấn SELECT, DELETE, INSERT, REPLACE và UPDATE. Luôn đặt nó ở đầu truy vấn.

Những vấn đề EXPLAIN có thể giải quyết

EXPLAIN hữu ích khi bạn muốn biết liệu một truy vấn có thực hiện đúng chức năng của nó hay không. Trước tiên, hãy kiểm tra dữ liệu trong mỗi cột của đầu ra EXPLAIN:

- *select_type* : theo dõi dữ liệu trong cột này nếu bạn muốn đảm bảo rằng các truy vấn của bạn tham gia vào các hoạt động JOIN
- *phân vùng* : dữ liệu trong cột này có thể hữu ích nếu bạn thêm phân vùng vào bảng và cần kiểm tra phân vùng nào mà truy vấn sử dụng
- *loại* : nếu bạn thiết kế truy vấn, hãy tập trung vào dữ liệu trong cột này
- *possible_keys* : dữ liệu trong cột này có thể cho bạn biết MySQL đã sử dụng chỉ mục nào
- *key* : nếu bạn cần biết MySQL đã chọn chỉ mục nào, hãy theo dõi dữ liệu trong cột
- *key_len* : cột hiển thị độ dài của chỉ mục đã chọn
- *ref* : dữ liệu trong cột có thể có giá trị để cải thiện hiệu suất truy vấn bằng cách sử dụng chỉ mục
- *hàng* : chú ý đến dữ liệu trong cột này nếu bạn thiết kế chỉ mục bên trong các phiên bản cơ sở dữ liệu
- *đã lọc* : nó hiển thị phần trăm gần đúng của các hàng trong một bảng được lọc theo một điều kiện cụ thể

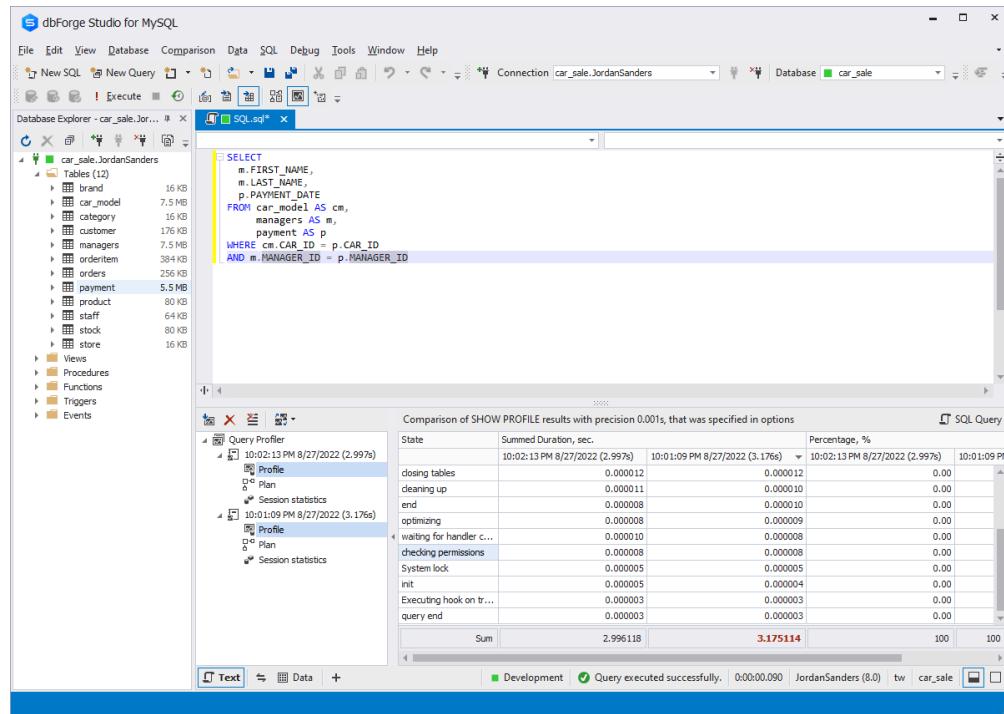


Thực hành tốt nhất để tối ưu hóa truy vấn chậm với MySQL EXPLAIN

Trong phần này, chúng tôi giải thích cách dễ dàng điều chỉnh truy vấn chạy chậm bằng Query Profiler được cung cấp trong dbForge Studio cho MySQL. Mỗi quan hệ FOREIGN KEY đóng vai trò chính trong quá trình này vì các bảng phải có mối quan hệ với nhau.

Chúng ta hãy xem xét kịch bản cho MySQL EXPLAIN với ví dụ: chúng ta đã tạo khóa ngoài cho một bảng mà chúng ta muốn thực hiện truy vấn. Sau đó, chúng ta chạy truy vấn với chế độ Query Profiling được bật.

Như bạn có thể thấy trong ảnh chụp màn hình, có một sự khác biệt rất lớn về thời gian thực hiện: kết quả đầu tiên là thời gian thực hiện truy vấn có khóa và kết quả thứ hai là thời gian thực hiện truy vấn không có khóa.



4.2 Tối ưu hóa bảng MySQL

Tối ưu hóa bảng MySQL

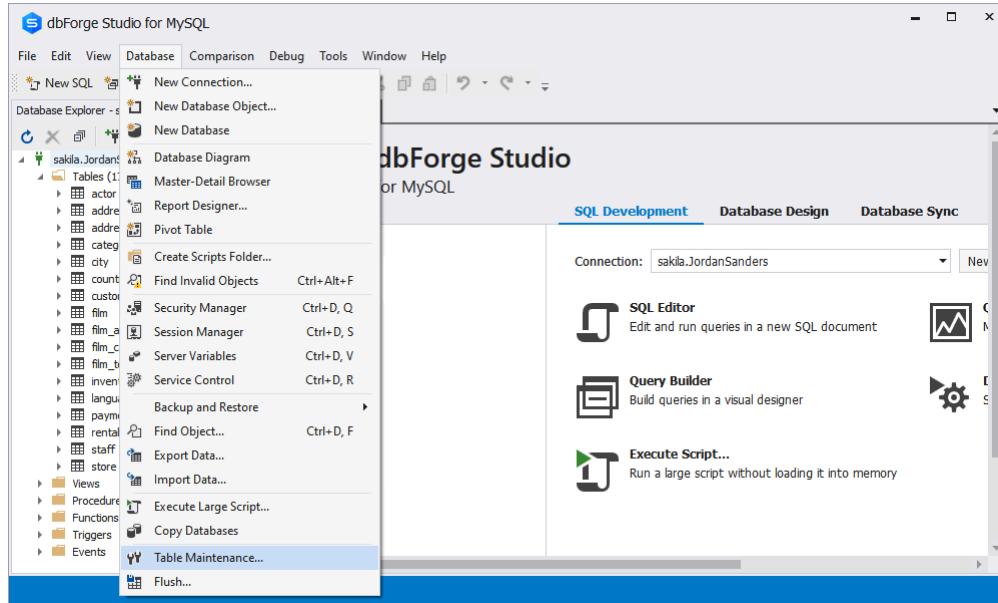
Công cụ Bảo trì bảng từ dbForge Studio cho MySQL có thể được sử dụng để thực hiện nhiều tác vụ tối ưu hóa khác nhau như được liệt kê dưới đây:

- Phân tích bảng
- Tối ưu hóa bảng
- Kiểm tra lỗi bảng
- Tổng kiểm tra
- Sửa

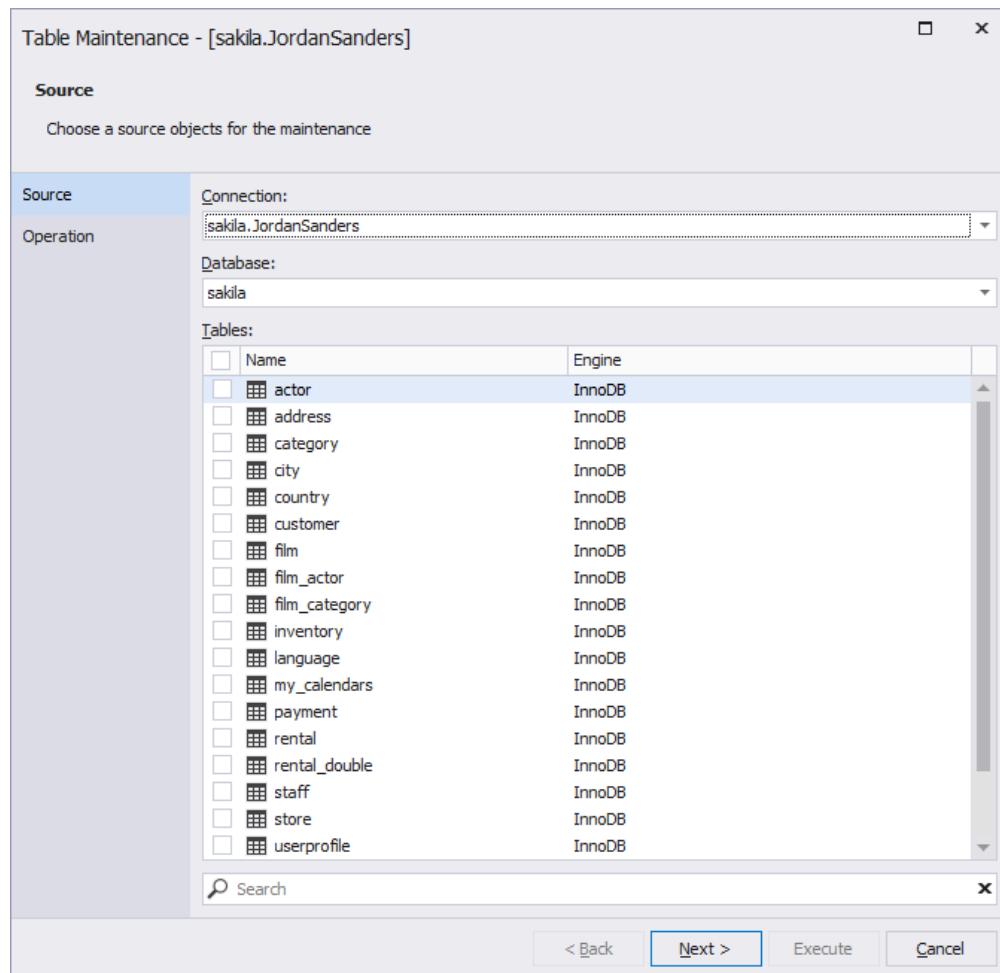
Có một số phương pháp để truy cập công cụ Bảo trì bảng từ bên trong dbForge Studio cho MySQL.

Phương pháp 1

Trong menu **Cơ sở dữ liệu** , chọn tùy chọn **Bảo trì bảng** từ danh sách thả xuống như hiển thị trong ảnh chụp màn hình sau.

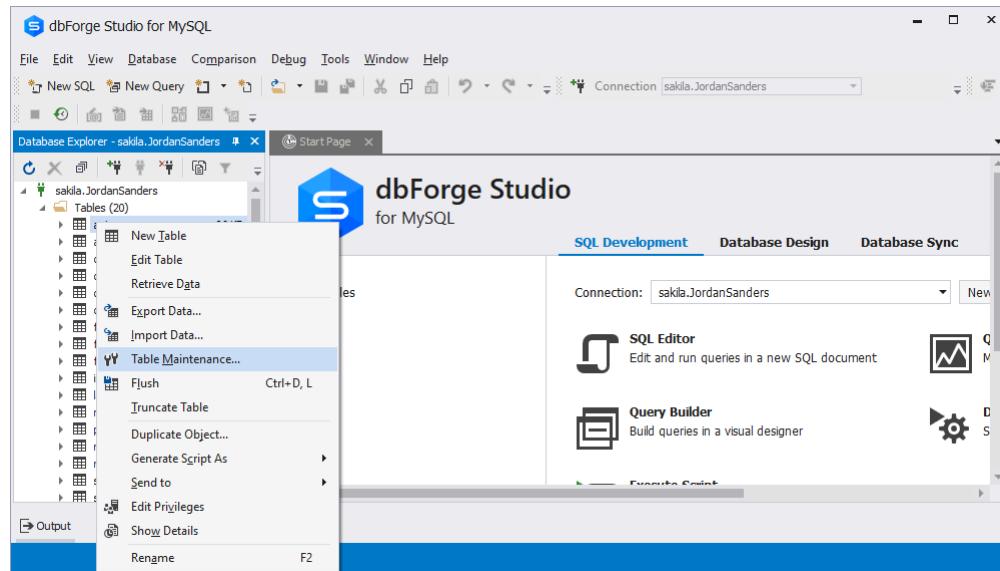


Công cụ Bảo trì bảng sẽ được hiển thị như trong ảnh chụp màn hình sau. Tại đây, bạn phải chỉ định kết nối cơ sở dữ liệu và tên của cơ sở dữ liệu, và chọn một hoặc nhiều bảng để tối ưu hóa.

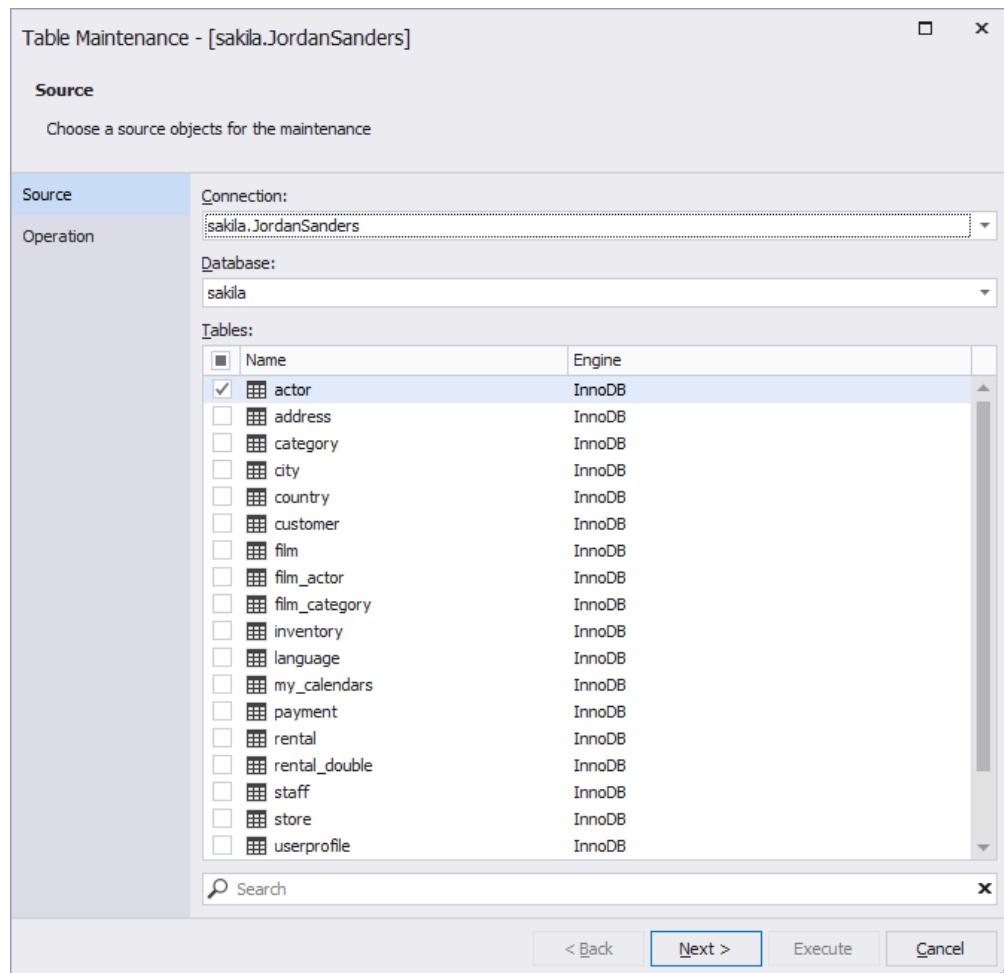


Phương pháp 2

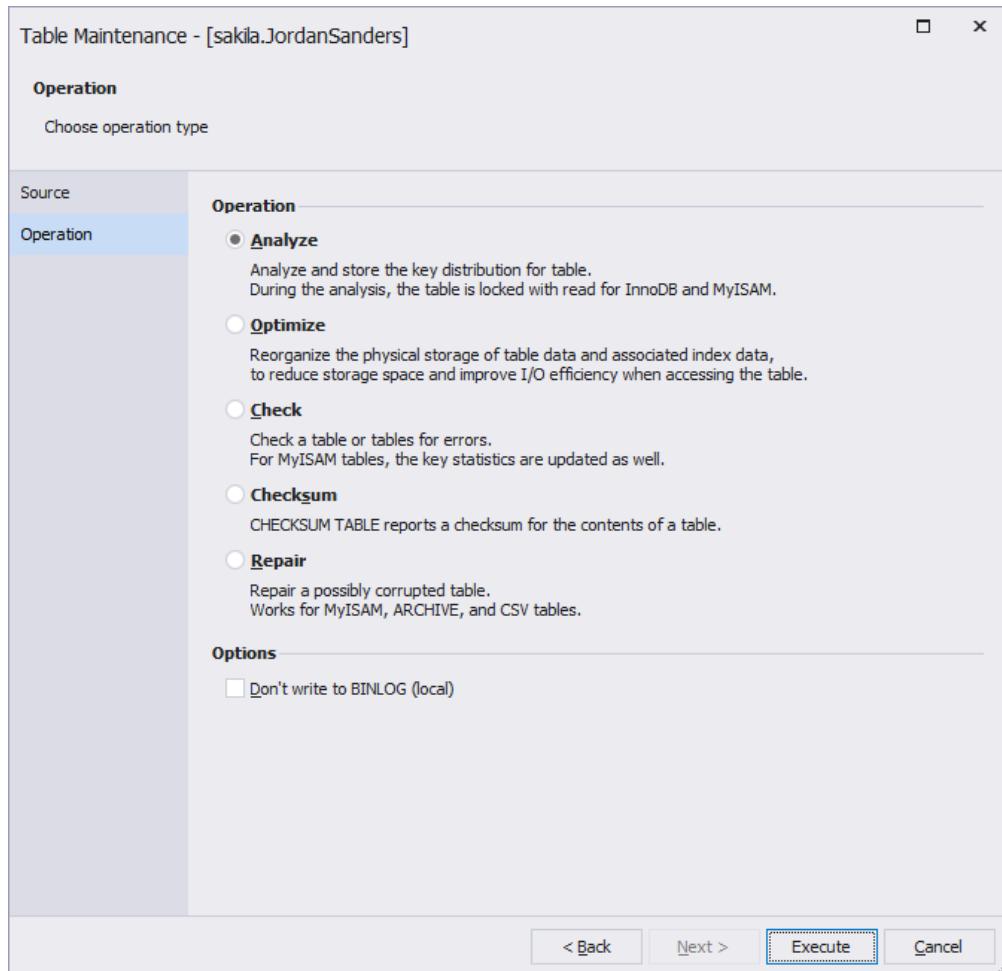
Một cách khác để mở công cụ Bảo trì bảng là nhấp chuột phải vào tên bảng trong Database Explorer rồi chọn **Bảo trì bảng** như trong ảnh chụp màn hình sau.



Công cụ Bảo trì bảng sẽ mở ra và bảng bạn nhấp chuột phải sẽ được chọn để bảo trì, như hiển thị trong ảnh chụp màn hình sau.



Nhập vào **Tiếp theo** . Bạn sẽ được chuyển đến trang Tùy chọn. Tại đây, bạn cần chọn loại hoạt động bảo trì.



Cách nhanh chóng để cập nhật số liệu thống kê phân phối chính

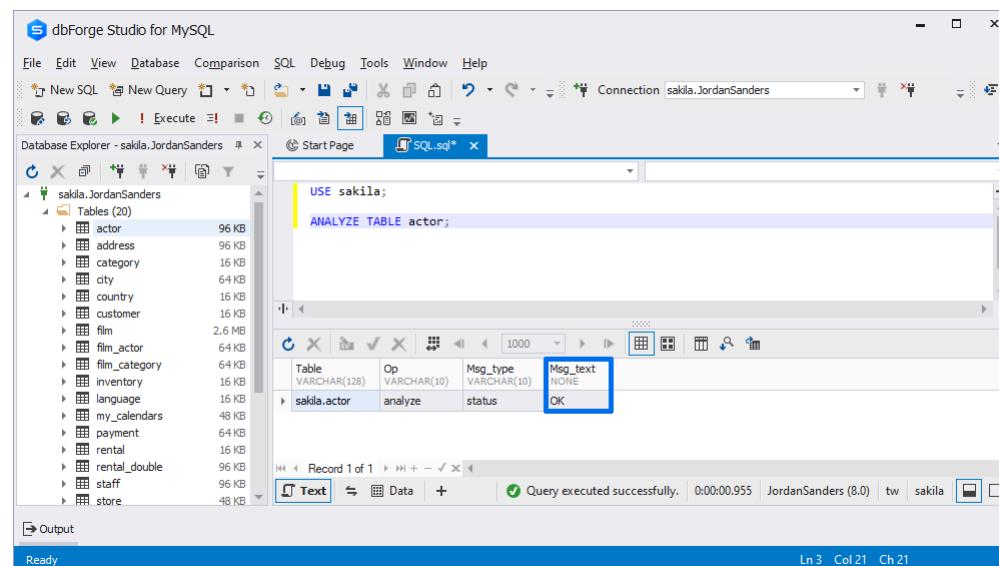
- Trình tối ưu hóa truy vấn MySQL sử dụng số liệu thống kê phân phối của các khóa trong bảng để tạo một kế hoạch truy vấn được tối ưu hóa. Nếu số liệu thống kê phân phối khóa không được phân tích trong một thời gian dài, trình tối ưu hóa truy vấn phải làm việc với số liệu thống kê phân phối khóa lỗi thời để tối ưu hóa các kế hoạch truy vấn. Điều này xảy ra khi một số lượng lớn các hoạt động *UPDATE*, *DELETE* hoặc *INSERT* xảy ra mà không có sự phân tích số liệu thống kê phân phối khóa. Các kế hoạch truy vấn như vậy không tối ưu và có thể dẫn đến các truy vấn bị trì hoãn.

- Để tối ưu hóa hiệu suất cơ sở dữ liệu của bạn hơn nữa, hãy sử dụng lệnh cập nhật số liệu thống kê về phân phối giá trị trong các cột bảng và giúp trình tối ưu hóa truy vấn MySQL đưa ra quyết định tốt hơn về cách thực hiện truy vấn:

PHÂN TÍCH BẢNG tên bảng;

Lệnh **ANALYZE TABLE** từ công cụ Table Maintenance của dbForge Studio được sử dụng để phân tích và lưu trữ nội bộ số liệu thống kê phân phối khóa của một bảng. Điều này giúp trình tối ưu hóa truy vấn tạo ra các kế hoạch truy vấn tối ưu dẫn đến thực hiện truy vấn nhanh chóng và hiệu quả.

Để chạy lệnh, hãy chọn **Analyze** từ công cụ Table Maintenance và nhấp vào **Execute**. Tập lệnh SQL cho lệnh sẽ được thực thi và đầu ra sẽ được hiển thị, như hiển thị bên dưới. Nếu cột *Msg_text* hiển thị **OK**, điều đó có nghĩa là lệnh **ANALYZE TABLE** đã được thực thi thành công.



Tối ưu hóa các bảng MySQL để chống phân mảnh

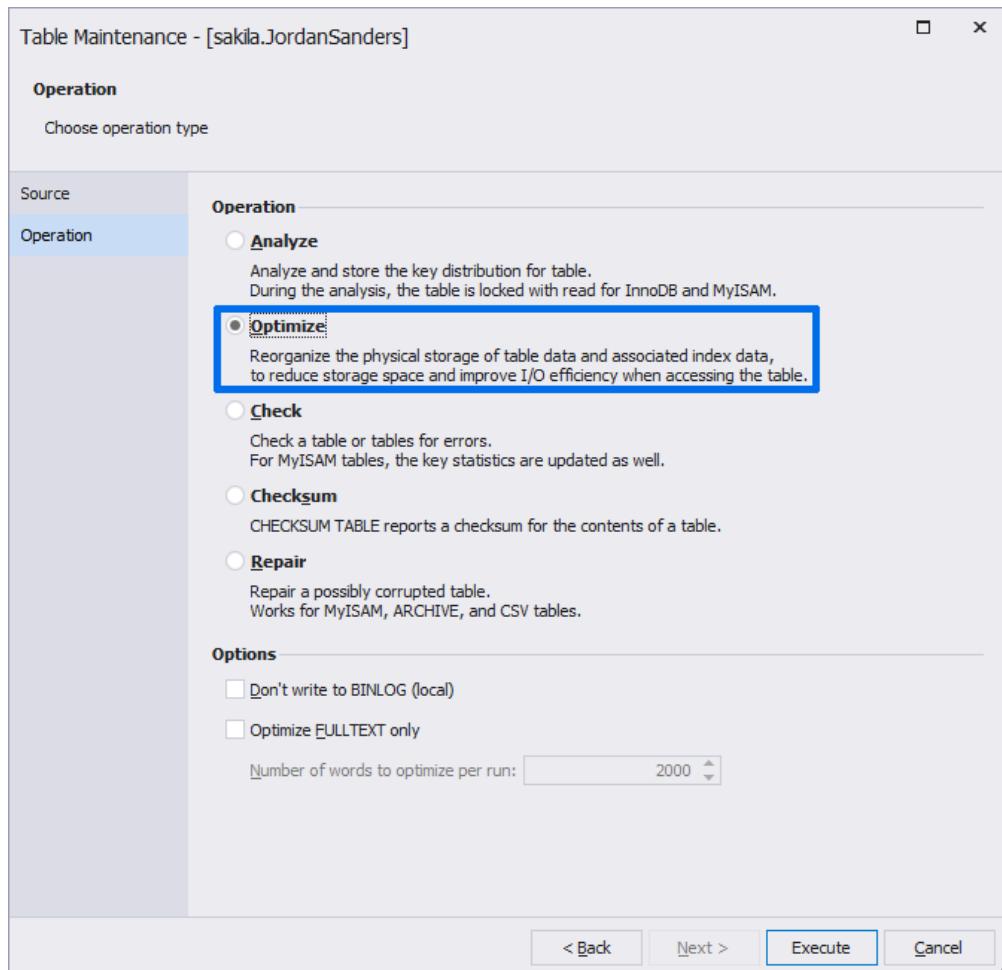
Trong một tệp bình thường, hệ thống cài đặt/gỡ cài đặt ứng dụng và sao chép/xóa tệp dẫn đến phân mảnh đĩa. Điều tương tự cũng xảy ra với các bảng cơ sở dữ liệu nếu một số lượng lớn các truy vấn DML (*CREATE* , *INSERT* , *DELETE*) được thực hiện trên các bảng cơ sở

dữ liệu. Phân mảnh lưu trữ bảng làm chậm nghiêm trọng hiệu suất truy vấn.

Trong MySQL, có một lệnh phân mảnh lưu trữ bảng bằng cách sắp xếp lại dữ liệu trên các khối dữ liệu liền kề, giúp cải thiện hiệu suất I/O. Câu lệnh này chỉ có thể chạy trên các bảng MyISAM:

OPTIMIZE TABLE tablename;

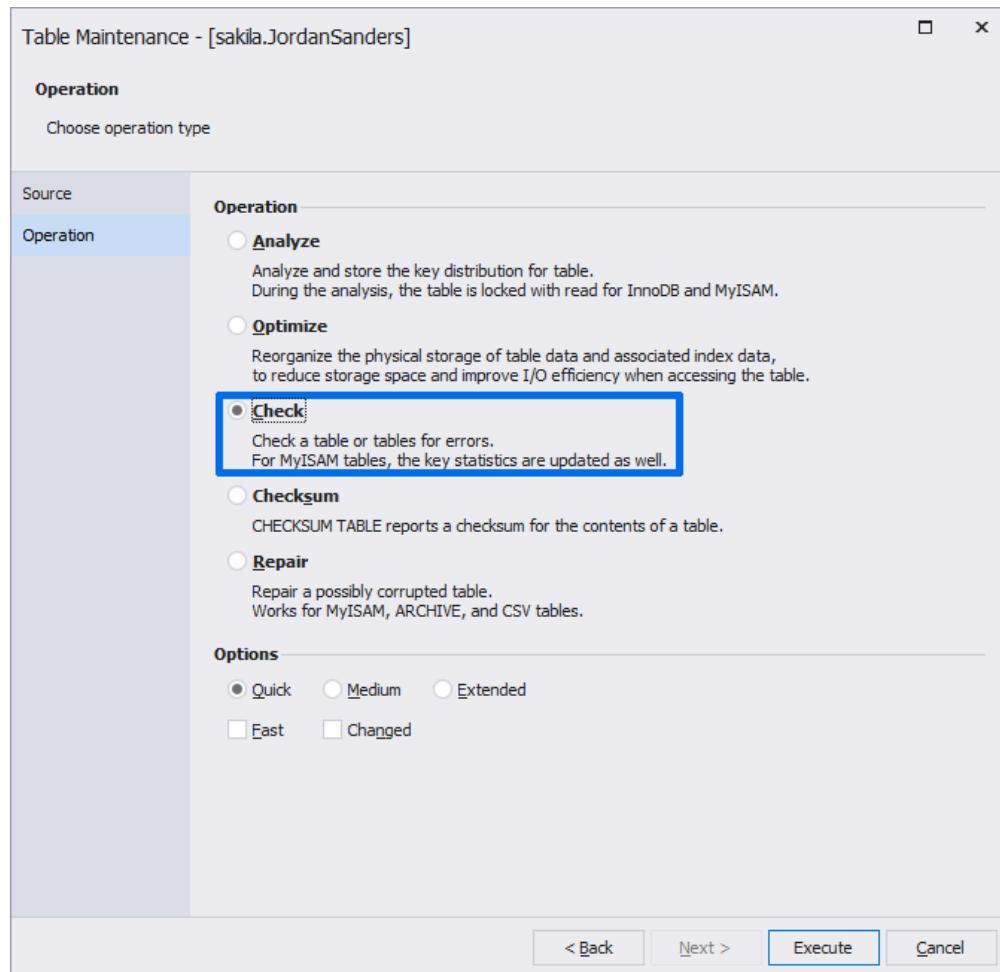
Để tối ưu hóa bảng MySQL cho quá trình phân mảnh, hãy vào công cụ Table Maintenance trong dbForge Studio và nhấp vào **Optimize** như trong ảnh chụp màn hình bên dưới. Nhấp vào **Execute** để chạy lệnh.



Kiểm tra bảng để chạy thử nghiệm chẩn đoán

Lệnh *CHECK TABLE* được sử dụng để chạy các thử nghiệm chẩn đoán trên các bảng để tìm lỗi và báo cáo bất kỳ lỗi nào được tìm thấy. Câu lệnh này cũng có thể được sử dụng để tìm các chế độ xem liên kết với các bảng không còn tồn tại. Lệnh này có thể được sử dụng để tối ưu hóa [các bảng InnoDB và MyISAM](#) trong MySQL.

Để chạy thử nghiệm chẩn đoán bằng lệnh, hãy nhập vào **Kiểm tra** trên trang Hoạt động của công cụ Bảo trì bảng như hiển thị bên dưới:



Có năm tùy chọn có sẵn cho lệnh **CHECK TABLE** . Ba trong số đó (**Nhanh** , **Trung bình** , **Mở rộng**) có sẵn dưới dạng các nút radio. Trong khi hai tùy chọn (**Nhanh** , **Đã thay đổi**) có sẵn dưới dạng hộp kiểm.

Khi sử dụng các nút radio, bạn có thể chọn các tùy chọn sau:

- Tùy chọn **Nhanh** bỏ qua việc kiểm tra các liên kết bị hỏng trong các hàng bảng để đẩy nhanh quá trình
- Tùy chọn **Medium** đảm bảo rằng các liên kết đã xóa trong các hàng bảng là hợp lệ
- Tùy chọn **Mở rộng** chạy tìm kiếm khóa đầy đủ cho tất cả các khóa trong tất cả các hàng của bảng. Lệnh **CHECK TABLE** với tùy chọn **Mở rộng** có thể mất nhiều thời gian để thực thi, tuy nhiên nó đảm bảo rằng bảng nhất quán 100%

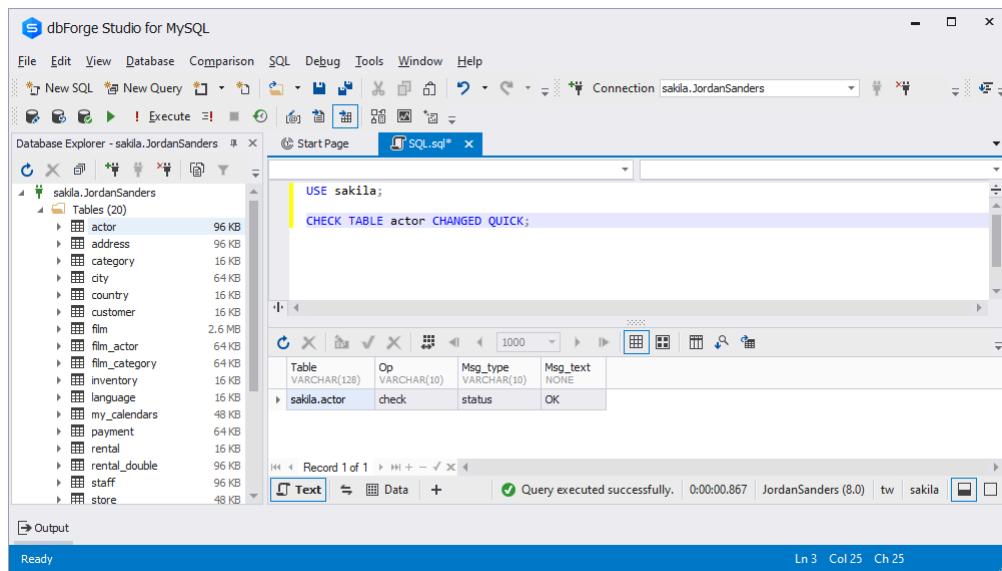
Khi sử dụng hộp kiểm, bạn có thể chọn các tùy chọn sau:

- Tùy chọn **Nhanh** chạy các thử nghiệm chẩn đoán trên các bảng chưa được đóng đúng cách. Tùy chọn này chỉ có thể áp dụng cho các bảng MyISAM và không có tác dụng đối với các bảng InnoDB
- Tùy chọn **Changed** kiểm tra các lệnh bảng đã thay đổi kể từ lệnh **CHECK TABLE** cuối cùng hoặc các bảng chưa được đóng đúng cách

Bây giờ, chúng ta hãy xem lệnh này hoạt động như thế nào khi được sử dụng bên ngoài giao diện người dùng đồ họa:

CHECK TABLE tablename CHANGED QUICK;

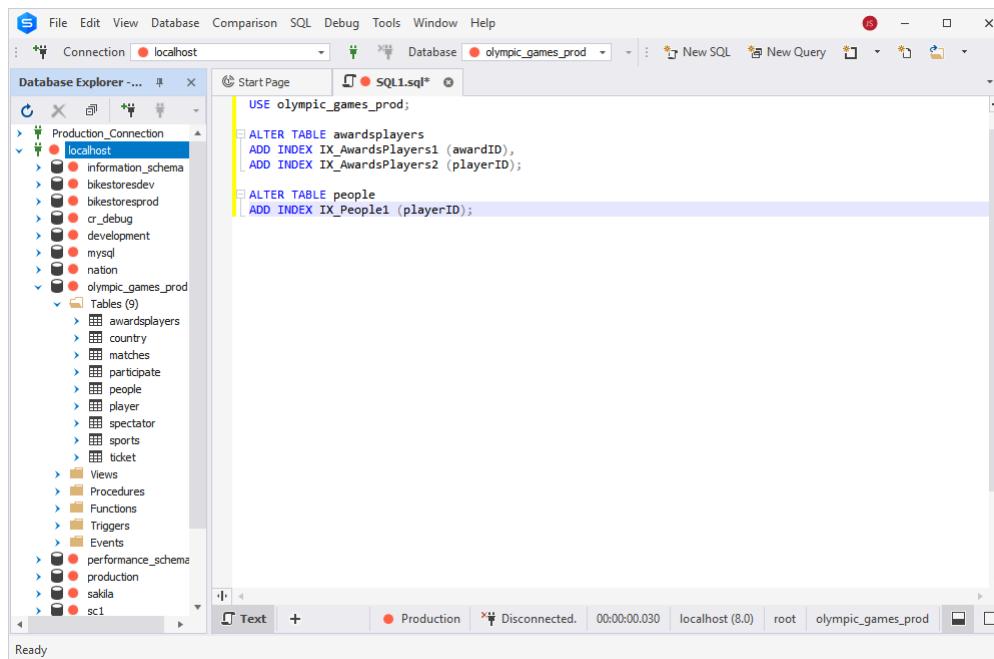
Để chạy lệnh **CHECK TABLE** , hãy nhập vào **Execute** . Sử dụng tập lệnh bên dưới, chúng tôi chạy lệnh bằng cách chọn tùy chọn **Changed** và **Quick** .



Điều chỉnh hiệu suất MySQL

Sử dụng chỉ mục MySQL để tăng hiệu suất

- Việc lập chỉ mục phù hợp để nâng cao hiệu suất không hề dễ dàng và đòi hỏi một mức độ chuyên môn nhất định, tuy nhiên đây là một trong những cải tiến hiệu suất tốt nhất mà bạn có thể thực hiện cho cơ sở dữ liệu của mình.
- MySQL sử dụng các chỉ mục như một chỉ mục sách hoặc bộ trinh để nhanh chóng tìm giá trị cho một truy vấn nhất định. Nếu không có chỉ mục, MySQL sẽ quét toàn bộ bảng theo từng hàng để tìm dữ liệu có liên quan. Do đó, tối ưu hóa chỉ mục nhằm mục đích tăng tốc độ truy xuất dữ liệu. Chỉ mục không hiển thị với người dùng và chứa thông tin về nơi lưu trữ dữ liệu thực tế. Cũng cần lưu ý rằng độ dài chỉ mục MySQL cho các bảng InnoDB có giới hạn tùy thuộc vào định dạng hàng.
- Chỉ mục MySQL cực kỳ hữu ích cho các tập dữ liệu lớn và điều chỉnh chỉ mục là điều đúng đắn cần làm nếu cơ sở dữ liệu của bạn đang phát triển nhanh chóng. Chỉ mục đặc biệt có lợi cho các hoạt động sau: tìm các hàng khớp với mệnh đề WHERE, truy xuất dữ liệu [bằng JOIN](#), sắp xếp và nhóm dữ liệu với sự trợ giúp của ORDER BY và GROUP BY.
- Vậy tại sao không chèn càng nhiều chỉ mục càng tốt? Đó sẽ là một ý tưởng tồi—các chỉ mục không cần thiết chiếm không gian và lãng phí thời gian của hệ thống, chưa kể chúng còn làm tăng chi phí cho các truy vấn vì các chỉ mục cần được cập nhật. Vì vậy, bạn phải tìm ra sự cân bằng phù hợp để đạt được mục đích sử dụng chỉ mục MySQL tối ưu.



Tối ưu hóa truy vấn MySQL

Bây giờ chúng ta hãy xem cách tối ưu hóa truy vấn MySQL để có hiệu suất và tốc độ tốt hơn. Đối với những ai muốn cải thiện truy vấn MySQL, sẽ là một ý tưởng hay nếu làm theo các kỹ thuật tối ưu hóa sau.

Thêm chỉ mục vào các cột được sử dụng trong các mệnh đề WHERE, ORDER BY và GROUP BY.

Theo cách này, bạn sẽ tăng hiệu suất truy vấn MySQL vì máy chủ MySQL sẽ lấy kết quả từ cơ sở dữ liệu nhanh hơn đáng kể.

Chỉ định các cột cần thiết trong các câu lệnh SELECT

Cố gắng tránh sử dụng `SELECT * FROM` vì nó sẽ truy xuất tất cả các cột của bảng và do đó gây thêm tải cho máy chủ và làm chậm hiệu suất của máy chủ. Hãy đặt ra quy tắc là luôn chỉ định các cột trong các câu lệnh `SELECT`.

Sử dụng DISTINCT và UNION một cách tiết kiệm

Một mẹo hay khác để điều chỉnh truy vấn là chỉ sử dụng toán tử `DISTINCT` và `UNION` khi cần thiết vì các truy vấn có chúng sẽ làm tăng

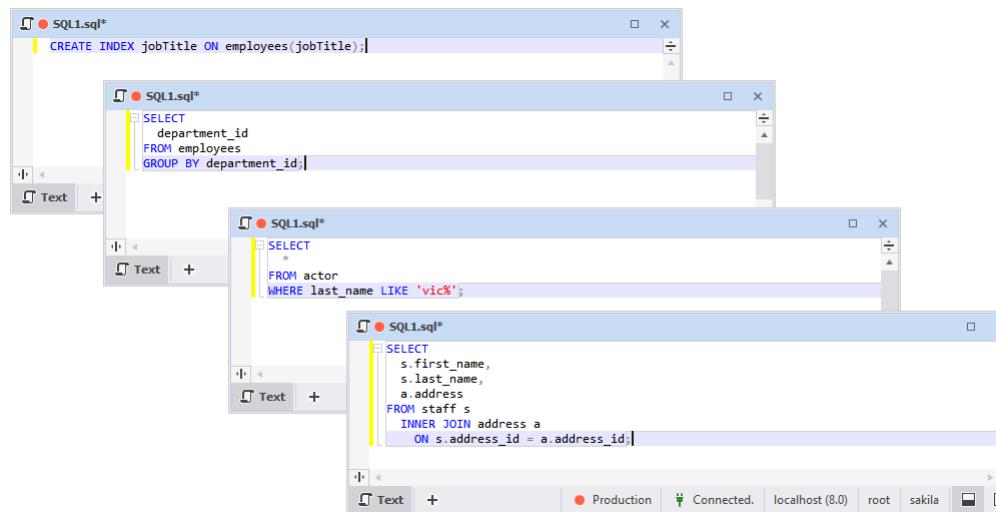
chi phí máy chủ và thường làm tăng thời gian phản hồi. Hãy cân nhắc thay thế UNION bằng UNION ALL và DISTINCT bằng GROUP BY để mang lại hiệu quả hơn cho quy trình.

Tránh sử dụng ký tự đại diện ở đầu các mẫu LIKE

Các truy vấn MySQL với toán tử LIKE thường dẫn đến giảm hiệu suất máy chủ nên chúng phải được sử dụng cẩn thận. MySQL không thể sử dụng chỉ mục khi mẫu LIKE bắt đầu bằng ký tự đại diện, ví dụ: '%xyz' và thực hiện quét toàn bộ bảng trong trường hợp này. Bạn nên ghi nhớ điều này khi tối ưu hóa các truy vấn MySQL và hãy thử sử dụng 'xyz%' thay thế bắt đầu khi nào có thể.

Sử dụng INNER JOIN thay vì OUTER JOIN

Chỉ sử dụng OUTER JOIN khi cần thiết. MySQL thực hiện nhiều công việc hơn để lấy kết quả cho OUTER JOIN so với INNER JOIN. Chúng tôi khuyên bạn nên kiểm tra hiệu suất của các truy vấn JOIN và trong trường hợp không thỏa mãn—hãy bắt đầu chuyển đổi OUTER JOIN thành INNER JOIN khi có thể. Tối ưu hóa [MySQL JOIN](#) có thể dẫn đến cải thiện hiệu suất đáng kể.



The screenshot shows four separate MySQL Workbench windows, each with a tab labeled 'SQL1.sql*'. The content of each window is as follows:

- Top window: `CREATE INDEX jobTitle ON employees(jobTitle);`
- Second window: `SELECT department_id FROM employees GROUP BY department_id;`
- Third window: `SELECT * FROM actor WHERE last_name LIKE 'vick%';`
- Bottom window: `SELECT s.first_name, s.last_name, a.address FROM staff s INNER JOIN address a ON s.address_id = a.address_id;`

5. So Sánh

5.1 SQL Server so với MySQL

SQL Server so với MySQL: Tổng quan về điểm tương đồng

Ngôn ngữ SQL: Cả SQL Server và MySQL đều sử dụng Ngôn ngữ truy vấn có cấu trúc (SQL) làm ngôn ngữ chính để quản lý và truy vấn dữ liệu. Ngôn ngữ chuẩn hóa này cho phép các nhà phát triển tạo, đọc, cập nhật và xóa dữ liệu, cũng như quản lý các đối tượng cơ sở dữ liệu như bảng, chỉ mục và chế độ xem.

Tuân thủ ACID: Cả hai RDBMS đều tuân thủ ACID, đảm bảo tính toàn vẹn và độ tin cậy của các giao dịch. ACID là viết tắt của Atomicity, Consistency, Isolation và Durability. Các thuộc tính này đảm bảo rằng các giao dịch cơ sở dữ liệu được xử lý đáng tin cậy và duy trì tính nhất quán của dữ liệu ngay cả trong trường hợp hệ thống bị lỗi.

Kiểu dữ liệu và lập chỉ mục: SQL Server và MySQL hỗ trợ các kiểu dữ liệu tương tự như số nguyên, số dấu phẩy động, ngày tháng và chuỗi ký tự. Chúng cũng cung cấp các tùy chọn lập chỉ mục để cải thiện hiệu suất truy vấn, bao gồm các chỉ mục chính, duy nhất và toàn văn bản.

Quy trình lưu trữ, trình kích hoạt và chế độ xem: Cả hai hệ thống đều hỗ trợ việc sử dụng quy trình lưu trữ, trình kích hoạt và chế độ xem, cho phép các nhà phát triển tạo mã mô-đun, có thể tái sử dụng, tự động hóa các hành động cụ thể và trình bày dữ liệu theo định dạng có cấu trúc.

Tính năng bảo mật: SQL Server và MySQL cung cấp nhiều tính năng bảo mật khác nhau để bảo vệ dữ liệu, chẳng hạn như xác thực người dùng, kiểm soát truy cập dựa trên vai trò và các tùy chọn mã hóa cho dữ liệu lưu trữ và dữ liệu đang truyền.

Khả năng mở rộng và tính khả dụng cao: Cả hai RDBMS đều cung cấp các tùy chọn mở rộng theo chiều ngang và chiều dọc để xử lý lượng dữ liệu và nhu cầu ngày càng tăng của người dùng. Chúng cũng cung cấp các giải pháp có tính khả dụng cao, bao gồm sao chép và phân cụm, để đảm bảo dữ liệu có thể truy cập được ngay cả khi hệ thống gặp sự cố hoặc bảo trì.

Hệ sinh thái và cộng đồng: SQL Server và MySQL có hệ sinh thái rộng lớn, bao gồm tài liệu, thư viện, công cụ mở rộng và cộng đồng lớn các nhà phát triển và người dùng đóng góp vào quá trình phát triển và hỗ trợ liên tục của họ.

Sự khác biệt giữa MySQL và SQL Server

	Mãy chủ SQL	MySQL
Hàm độ dài	SELECT LEN(data_string) FROM _table_name_	SELECT CHARACTER_LENGTH(data_string) FROM _table_name_
Hàm nối	SELECT ('SQL' + 'Server')	SELECT CONCAT ('My', 'SQL')
Chọn N bản ghi hàng đầu từ một bảng	SELECT TOP 10 * FROM _table_name_ WHERE _condition_	SELECT * FROM _table_name_ WHERE`_condition_` LIMIT 10

Tạo GUID	SELECT NEWID()	SELECT UUID()
Trả về ngày và giờ hiện tại	SELECT GETDATE()	SELECT NOW()
Trả về phiên bản cơ sở dữ liệu	SELECT @@VERSION	SELECT VERSION()

Thao tác với csdl Sakila theo chương 3

```

CREATE DATABASE DSSINHVIEN;
USE DSSINHVIEN;
CREATE TABLE KHOAHOC (
MAKHOA VARCHAR(10) PRIMARY KEY,
TENKHOA VARCHAR (50)
)CHARACTER SET utf8mb4;
CREATE TABLE LOP
(
MALOP VARCHAR (10) PRIMARY KEY,
TENLOP VARCHAR (50) ,
SISODK INT ,
MAKHOA VARCHAR(10) ,
FOREIGN KEY (MAKHOA) REFERENCES KHOAHOC(MAKHOA)CHARACTER SET utf8mb4;

CREATE TABLE SINHVIEN
(
MASV VARCHAR (10) PRIMARY KEY,
HOTEN VARCHAR (50) ,
NGSINH DATE ,
DCHI VARCHAR (50),
GIOITINH VARCHAR (50),

```

```
MALOP CHAR(10),
FOREIGN KEY (MALOP) REFERENCES LOP(MALOP))
CHARACTER SET utf8mb4;
CREATE TABLE SINHVIENMOI
(
MASV VARCHAR (10) PRIMARY KEY,
HOTEN VARCHAR (50) ,
NGSINH DATE ,
DCHI VARCHAR (50),
GIOITINH VARCHAR (50),
MALOP CHAR(10),
FOREIGN KEY (MALOP) REFERENCES LOP(MALOP)
)CHARACTER SET utf8mb4;
CREATE TABLE MONHOC
(
MAMH CHAR (10) PRIMARY KEY,
TENMH VARCHAR (50) ,
SOTIET INT )
CHARACTER SET utf8mb4;
CREATE TABLE KETQUA
(
MASV CHAR (10) ,
MAMH CHAR (10) ,
DIEM FLOAT (10),
CONSTRAINT PK_KETQUA PRIMARY KEY (MASV,MAMH),
FOREIGN KEY (MASV) REFERENCES SINHVIEN (MASV),
FOREIGN KEY (MAMH) REFERENCES MONHOC (MAMH))
CHARACTER SET utf8mb4;
CREATE TABLE DIEMTB
(
```

```
MASV CHAR (10) ,
MAMH CHAR (10) ,
DIEM FLOAT (10),
CONSTRAINT PK_KETQUA PRIMARY KEY (MASV,MAMH),
FOREIGN KEY (MASV) REFERENCES SINHVIEN (MASV),
FOREIGN KEY (MAMH) REFERENCES MONHOC (MAMH))
CHARACTER SET utf8mb4;
```

```
INSERT INTO KHOAHOC
```

```
VALUES ('TH','TIN HOC'),
('VL','VAT LY'),
('DT','ĐIEN TU'),
('TP','THUC PHAM'),
('HH','HOA HOC'),
('SH','SINH HOC');
```

```
INSERT INTO LOP
```

```
VALUES ('L001','24THTH1','30','TH'),
('L002','24THTH2','30','TH'),
('L003','05CDTH','60','TH'),
('L004','06CDTH1','60','TH'),
('L005','06CDTH2','60','TH'),
('L006','24THTP1','30','TP'),
('L007','24THTP2','30','TP'),
('L008','24THTP3','30','TP'),
('L009','06CDDT','60','DT'),
('L010','05CDVL','45','VL'),
('L011','24THHH','60','HH'),
('L012','06CDSH','60','SH');
```

```
INSERT INTO SINHVIENMOI
```

```
VALUES ('SV01',N'Nguyễn Thị Lan',NULL,'TPHCM',N'Nam','L001'),
```

('SV25',N'Trần Văn Chính','1980/04/02','VUNG TAU',N'Nam','L001'),
('SV33',N'Truong Văn Thọ','1980/10/11','DA NANG',N'Nữ','L001'),
('SV04',N'Lê Văn Khánh','1985/07/04','VUNG TAU',N'Nam','L002'),
('SV45',N'Ngô Viết Nam',NULL,'DA NANG',N'Nam','L002'),
('SV06',N'Trần Thị Liễu','1985/07/05','TPHCM',N'Nữ','L003');

INSERT INTO SINHVIEN

VALUES ('SV01',N'Nguyễn Thị Lan',NULL,'TPHCM',N'Nam','L001'),
('SV02',N'Trần Thanh Tùng','1980/04/02','VUNG TAU',N'Nam','L001'),
('SV03',N'Truong Thị Hué','1980/10/11','DA NANG',N'Nữ','L001'),
('SV04',N'Lê Văn Khánh','1985/07/04','VUNG TAU',N'Nam','L002'),
('SV05',N'Ngô Viết Nam',NULL,'DA NANG',N'Nam','L002'),
('SV06',N'Trần Thị Liễu','1985/07/05','TPHCM',N'Nữ','L003'),
('SV07',N'Trần Thành Nam','1988/04/02','DONG NAI',N'Nam','L004'),
('SV08',N'Phạm Hoài Phong','1979/05/09','TIEN GIANG',N'Nam','L004'),
('SV09',N'Trần Thị Tố Anh','1981/07/08','TPHCM',N'Nữ','L004'),
('SV10',N'Đỗ Thị Hạnh','1982/03/15','DONG NAI',N'Nữ','L004'),
('SV11',N'Hà Thanh Tùng','1983/04/26','VUNG TAU',N'Nam','L005'),
('SV12',N'Trần Quốc Tuấn','1982/07/15','TIEN GIANG',N'Nam','L006'),
('SV13',N'Nguyễn Minh Trí',NULL,'DONG THAP',N'Nam','L007'),
('SV14',N'Trần Thanh Phước','1984/12/17','TPHCM',N'Nam','L008'),
('SV15',N'Nguyễn Phước Sang','1986/02/18','HA NOI',N'Nam','L008'),
('SV16',N'Lê Thị Anh','1987/05/15','HA NOI',N'Nữ','L009'),
('SV17',N'Truong Thị Nhàn',NULL,'TPHCM',N'Nữ','L010'),
('SV18',N'Nguyễn Kim Phụng','1983/06/14','HA NOI',N'Nữ','L011'),
('SV19',N'Trần Thị Thắm','1985/02/11','DONG NAI',N'Nữ','L012'),
('SV20',N'Nguyễn Thị Thu Trang','1988/05/17','TPHCM',N'Nữ','L012');

INSERT INTO MONHOC

VALUES ('M001','TOAN CAO CAP A1','60'),
('M002','LICH SU DANG','45'),

('M003','CHINH TRI','45'),
(‘M004’,‘CO SO DU LIEU’,‘90’),
(‘M005’,‘SQL SERVER’,‘60’),
(‘M006’,‘LAP TRINH C’,‘60’),
(‘M007’,‘XU LY ANH’,‘90’),
(‘M008’,‘TIN HOC CAN BAN’,‘60’),
(‘M009’,‘MANG MAY TINH’,‘60’),
(‘M010’,‘TOAN ROI RAC’,‘45’),
(‘M011’,‘LAP TRINH QUAN LY’,‘90’),
(‘M012’,‘VISUAL BASIC’,‘60’);

INSERT INTO KETQUA

VALUES ('SV01','M001','8'),
(‘SV01’,‘M002’,‘4’),
(‘SV01’,‘M003’,‘6’),
(‘SV02’,‘M001’,‘7’),
(‘SV04’,‘M001’,‘5’),
(‘SV04’,‘M002’,‘7’),
(‘SV05’,‘M003’,‘9’),
(‘SV06’,‘M008’,‘10’),
(‘SV07’,‘M005’,‘6’),
(‘SV07’,‘M007’,‘9’),
(‘SV08’,‘M004’,‘7’),
(‘SV09’,‘M010’,‘3’),
(‘SV10’,‘M001’,‘8’),
(‘SV10’,‘M006’,‘6’),
(‘SV11’,‘M004’,‘5’),
(‘SV12’,‘M004’,‘8’),
(‘SV13’,‘M007’,‘8’),
(‘SV14’,‘M005’,‘4’),

```
('SV14','M006','7'),  
('SV15','M009','7'),  
('SV16','M002','10'),  
('SV17','M006','8'),  
('SV18','M007','9'),  
('SV19','M006','7'),  
('SV20','M001','5'),  
('SV20','M002','7'),  
('SV20','M004','6'),  
('SV20','M005','9'),  
('SV20','M009','10');
```

```
INSERT INTO DIEMTB
```

```
VALUE ('SV01','M001','8.4'),  
('SV01','M002','4.5'),  
('SV01','M003','6.6'),  
('SV02','M001','7'),  
('SV04','M001','5.76'),  
('SV04','M002','7.9'),  
('SV05','M003','9.2'),  
('SV06','M008','10'),  
('SV07','M005','6.7'),  
('SV07','M007','9.82'),  
('SV08','M004','7.3'),  
('SV09','M010','3.9'),  
('SV10','M001','8.2'),  
('SV10','M006','6.4');
```

```
SELECT FROM KHOAHOC;
```

```
SELECT * FROM KHOAHOC;
SELECT * FROM LOP;
SELECT * FROM MONHOC;
```

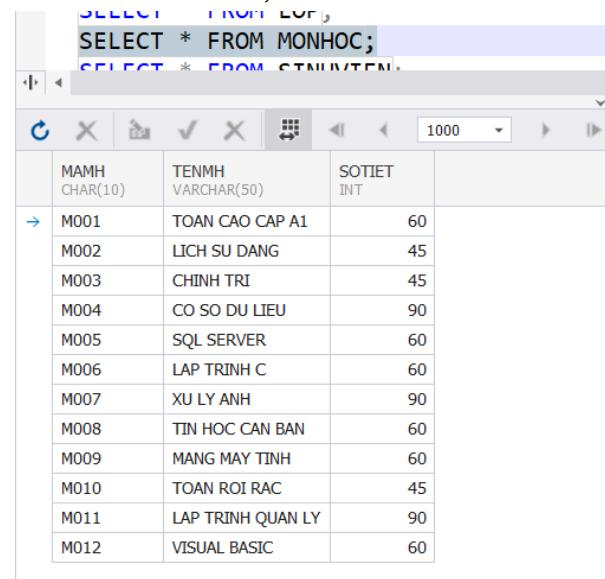
	MAKHOA VARCHAR(10)	TENKHOA VARCHAR(50)
→	DT	ĐIEN TU
	HH	HOA HOC
	SH	SINH HOC
	TH	TIN HOC
	TP	THUC PHAM
	VL	VAT LY

SELECT

```
SELECT * FROM KHOAHOC;
SELECT * FROM LOP;
SELECT * FROM MONHOC;
```

	MALOP VARCHAR(10)	TENLOP VARCHAR(50)	SISODK INT	MAKHOA VARCHAR(10)
→	L001	24THTH1	30	TH
	L002	24THTH2	30	TH
	L003	05CDTH	60	TH
	L004	06CDTH1	60	TH
	L005	06CDTH2	60	TH
	L006	24THTP1	30	TP
	L007	24THTP2	30	TP
	L008	24THTP3	30	TP
	L009	06CDDT	60	DT
	L010	05CDVL	45	VL
	L011	24THHH	60	HH
	L012	06CDSH	60	SH

FROM MONHOC;

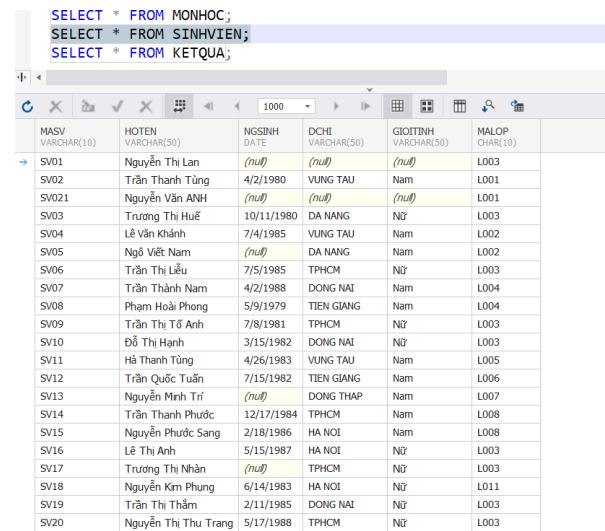


SELECT * FROM MONHOC;

SELECT * FROM SINHVIEN;

MAMH CHAR(10)	TENMH VARCHAR(50)	SOTIET INT
M001	TOAN CAO CAP A1	60
M002	LICH SU DANG	45
M003	CHINH TRI	45
M004	CO SO DU LIEU	90
M005	SQL SERVER	60
M006	LAP TRINH C	60
M007	XU LY ANH	90
M008	TIN HOC CAN BAN	60
M009	MANG MAY TINH	60
M010	TOAN ROI RAC	45
M011	LAP TRINH QUAN LY	90
M012	VISUAL BASIC	60

SELECT



SELECT * FROM MONHOC;

SELECT * FROM SINHVIEN;

SELECT * FROM KETQUA;

MASV VARCHAR(10)	HOTEN VARCHAR(50)	NGSINH DATE	DCHI VARCHAR(50)	GIOITINH VARCHAR(50)	MALOP CHAR(10)
SV01	Nguyễn Thị Lan	(null)	(null)	(null)	L003
SV02	Trần Thanh Tùng	4/2/1980	VUNG TAU	Nam	L001
SV021	Nguyễn Văn ANH	(null)	(null)	(null)	L001
SV03	Trương Thị Huệ	10/11/1980	DA NANG	Nữ	L003
SV04	Lê Văn Khanh	7/4/1985	VUNG TAU	Nam	L002
SV05	Ngô Việt Nam	(null)	DA NANG	Nam	L002
SV06	Trần Thị Liễu	7/5/1985	TPHCM	Nữ	L003
SV07	Trần Thành Nam	4/2/1988	DONG NAI	Nam	L004
SV08	Phạm Hòa Phong	5/9/1979	TIEN GIANG	Nam	L004
SV09	Trần Thị Tố Anh	7/8/1981	TPHCM	Nữ	L003
SV10	Đỗ Thị Hạnh	3/15/1982	DONG NAI	Nữ	L003
SV11	Hà Thanh Tùng	4/26/1983	VUNG TAU	Nam	L005
SV12	Trần Quốc Tuấn	7/15/1982	TIEN GIANG	Nam	L006
SV13	Nguyễn Minh Trí	(null)	DONG THAP	Nam	L007
SV14	Trần Thanh Phước	12/17/1984	TPHCM	Nam	L008
SV15	Nguyễn Phước Sang	2/18/1986	HA NOI	Nam	L008
SV16	Lê Thị Anh	5/15/1987	HA NOI	Nữ	L003
SV17	Trương Thị Nhàn	(null)	TPHCM	Nữ	L003
SV18	Nguyễn Kim Phung	6/14/1983	HA NOI	Nữ	L011
SV19	Trần Thị Thắm	2/11/1985	DONG NAI	Nữ	L003
SV20	Nguyễn Thị Thu Trang	5/17/1988	TPHCM	Nữ	L003

SELECT * FROM KETQUA;

	MASV CHAR(10)	MAMH CHAR(10)	DIEM FLOAT
→	SV01	M001	8
	SV01	M002	4
	SV01	M003	6
	SV02	M001	7
	SV04	M001	5
	SV04	M002	7
	SV05	M003	9
	SV06	M008	10
	SV07	M005	6
	SV07	M007	9
	SV08	M004	7
	SV09	M010	3
	SV10	M001	8
	SV10	M006	6
	SV11	M004	5
	SV12	M004	8
	SV13	M007	8
	SV14	M005	4
	SV14	M006	7
	SV15	M009	7
	SV16	M002	10
	SV17	M006	8

Record 1 of 29

Text Data + ✓ Query executed successfully.

-- Điều kiện WHERE của MySQL

-- Toán tử 'and' và 'or'

-- VD : Cho biết họ tên những sinh viên nữ có địa chỉ ở 'TPHCM' hay ở 'VUNG TAU'.

SELECT HOTEN

FROM SINHVIEN

WHERE GIOITINH = 'NU'

AND DCHI = 'TPHCM' OR DCHI = 'VUNG TAU'

```
-- Toán tử "and' và 'or"
-- VD : Cho biết họ tên những sinh viên nữ có địa chỉ ở 'TPHCM' hay ở 'VUNG TAU'.
SELECT HOTEN
FROM SINHVIEN
WHERE GIOITINH = 'NU'
AND DCHI = 'TPHCM' OR DCHI = 'VUNG TAU'
-- Toán RFTWFFN
```

HOTEN VARCHAR(50)
Trần Thanh Tùng
Lê Văn Khánh
Trần Thị Liễu
Trần Thị Tố Anh
Hà Thành Tùng
Trương Thị Nhàn
Nguyễn Thị Thu Trang

-- Toán BETWEEN

-- VD : Cho biết họ tên và số điểm của sinh viên có điểm từ 7 đến 9

```
SELECT HOTEN , DIEM
FROM SINHVIEN , KETQUA
WHERE DIEM BETWEEN 7 AND 9 AND KETQUA.MASV=SINHVIEN.MASV
ORDER BY DIEM
```

```
-- Toán BETWEEN
-- VD : Cho biết họ tên và số điểm của sinh viên có điểm từ 7 đến 9
SELECT HOTEN , DIEM
FROM SINHVIEN , KETQUA
WHERE DIEM BETWEEN 7 AND 9 AND KETQUA.MASV=SINHVIEN.MASV
ORDER BY DIEM
-- Toán tử LIKE
-- VD : tìm các sinh viên có họ là "Nguyễn"
SELECT *
```

HOTEN VARCHAR(50)	DIEM FLOAT
Trần Thanh Tùng	7
Lê Văn Khánh	7
Phạm Hoài Phong	7
Trần Thành Phước	7
Nguyễn Phước Sang	7
Trần Thị Thắm	7
Nguyễn Thị Thu Trang	7
Nguyễn Thị Lan	8
Đỗ Thị Hạnh	8

-- Toán tử LIKE

-- VD : tìm các sinh viên có họ là "Nguyễn"

```
SELECT *
FROM SINHVIEN
WHERE HOTEN LIKE 'Nguyễn%';
```

-- Toán tử LIKE
-- VD : tìm các sinh viên có họ là "Nguyễn"
SELECT *
FROM SINHVIEN
WHERE HOTEN LIKE 'Nguyễn%';

-- Toán tử In
-- Vd : Tìm các sinh viên có mã lớp là "L001", "L002", hoặc "L003".
SELECT *
FROM SINHVIEN
WHERE HOTEN LIKE 'Nguyễn%';

MASV VARCHAR(10)	HOTEN VARCHAR(50)	NGSINH DATE	DCHI VARCHAR(50)	GIOITINH VARCHAR(50)	MALOP CHAR(10)
SV01	Nguyễn Thị Lan	(null)	TPHCM	Nam	L001
SV13	Nguyễn Minh Trí	(null)	DONG THAP	Nam	L007
SV15	Nguyễn Phước Sang	2/18/1986	HA NOI	Nam	L008
SV18	Nguyễn Kim Phung	6/14/1983	HA NOI	Nữ	L011
SV20	Nguyễn Thị Thu Trang	5/17/1988	TPHCM	Nữ	L012

-- Toán tử In
-- Vd : Tìm các sinh viên có mã lớp là "L001", "L002", hoặc "L003".

```
SELECT *
FROM SINHVIEN
WHERE MALOP IN ('L001', 'L002', 'L003');
```

-- Toán tử In
-- Vd : Tìm các sinh viên có mã lớp là "L001", "L002", hoặc "L003".
SELECT *
FROM SINHVIEN
WHERE MALOP IN ('L001', 'L002', 'L003');

-- Toán tử Not In
-- Vd : Tìm các sinh viên không có mã lớp là "L001", "L002", hoặc "L003".
SELECT *
FROM SINHVIEN
WHERE MALOP NOT IN ('L001', 'L002', 'L003');

MASV VARCHAR(10)	HOTEN VARCHAR(50)	NGSINH DATE	DCHI VARCHAR(50)	GIOITINH VARCHAR(50)	MALOP CHAR(10)
SV01	Nguyễn Thị Lan	(null)	TPHCM	Nam	L001
SV02	Trần Thanh Tùng	4/2/1980	VUNG TAU	Nam	L001
SV03	Trương Thị Huế	10/11/1980	DA NANG	Nữ	L001
SV04	Lê Văn Khánh	7/4/1985	VUNG TAU	Nam	L002
SV05	Ngô Việt Nam	(null)	DA NANG	Nam	L002
SV06	Trần Thị Liễu	7/5/1985	TPHCM	Nữ	L003

-- Toán tử Not In

-- Vd : Tìm các sinh viên không có mã lớp là "L001", "L002", hoặc "L003".

SELECT *

FROM SINHVIEN

WHERE MALOP NOT IN ('L001', 'L002', 'L003');

```
-- Toán tử Not In
-- Vd : Tìm các sinh viên không có mã lớp là "L001", "L002", hoặc "L003".
SELECT *
FROM SINHVIEN
WHERE MALOP NOT IN ('L001', 'L002', 'L003');
-- Toán tử NULL
-- VD : tìm tất cả các sinh viên chưa có địa chỉ
```

MASV VARCHAR(10)	HOTEN VARCHAR(50)	NGSINH DATE	DCHI VARCHAR(50)	GIOITINH VARCHAR(50)	MALOP CHAR(10)
SV01	Nguyễn Thị Lan	(null)	TPHCM	Nam	L001
SV02	Trần Thanh Tùng	4/2/1980	VUNG TAU	Nam	L001
SV03	Trương Thị Huế	10/1/1980	DA NANG	Nữ	L001
SV04	Lê Văn Khanh	7/4/1985	VUNG TAU	Nam	L002
SV05	Ngô Việt Nam	(null)	DA NANG	Nam	L002
SV06	Trần Thị Liễu	7/5/1985	TPHCM	Nữ	L003

-- Toán tử NULL

-- VD : tìm tất cả các sinh viên chưa có địa chỉ

SELECT * FROM SINHVIEN

WHERE DCHI IS NULL;

-- Toán tử NULL

-- VD : tìm tất cả các sinh viên chưa có địa chỉ

SELECT * FROM SINHVIEN

WHERE DCHI IS NULL;

-- Toán tử not NULL

-- VD : tìm các sinh viên đã có địa chỉ được nhập

SELECT * FROM SINHVIEN

```
-- Toán tử NULL
-- VD : tìm tất cả các sinh viên chưa có địa chỉ
SELECT * FROM SINHVIEN
WHERE DCHI IS NULL;
-- Toán tử not NULL
-- VD : tìm các sinh viên đã có địa chỉ được nhập
SELECT * FROM SINHVIEN
```

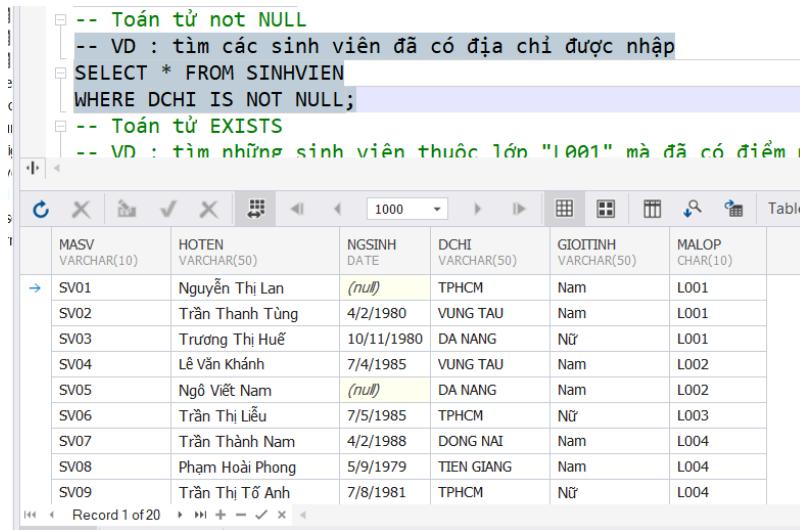
MASV VARCHAR(10)	HOTEN VARCHAR(50)	NGSINH DATE	DCHI VARCHAR(50)	GIOITINH VARCHAR(50)	MALOP CHAR(10)
---------------------	----------------------	----------------	---------------------	-------------------------	-------------------

-- Toán tử not NULL

-- VD : tìm các sinh viên đã có địa chỉ được nhập

SELECT * FROM SINHVIEN

WHERE DCHI IS NOT NULL;



The screenshot shows a database query results window. The query is:

```
-- Toán tử not NULL
-- VD : tìm các sinh viên đã có địa chỉ được nhập
SELECT * FROM SINHVIEN
WHERE DCHI IS NOT NULL;
```

The results table has the following columns:

	MASV VARCHAR(10)	HOTEN VARCHAR(50)	NGSINH DATE	DCHI VARCHAR(50)	GIOITINH VARCHAR(50)	MALOP CHAR(10)
→	SV01	Nguyễn Thị Lan	(null)	TPHCM	Nam	L001
	SV02	Trần Thanh Tùng	4/2/1980	VUNG TAU	Nam	L001
	SV03	Trương Thị Huế	10/11/1980	DA NANG	Nữ	L001
	SV04	Lê Văn Khánh	7/4/1985	VUNG TAU	Nam	L002
	SV05	Ngô Việt Nam	(null)	DA NANG	Nam	L002
	SV06	Trần Thị Liễu	7/5/1985	TPHCM	Nữ	L003
	SV07	Trần Thành Nam	4/2/1988	DONG NAI	Nam	L004
	SV08	Phạm Hoài Phong	5/9/1979	TIEN GIANG	Nam	L004
	SV09	Trần Thị Tố Anh	7/8/1981	TPHCM	Nữ	L004

Record 1 of 20

-- Toán tử EXISTS

-- VD : tìm những sinh viên thuộc lớp "L001" mà đã có điểm môn học

SELECT *

FROM SINHVIEN

WHERE MALOP = 'L001'

AND EXISTS (

SELECT 1

FROM KETQUA

WHERE SINHVIEN.MASV = KETQUA.MASV);

```
-- Toán tử EXISTS
-- VD : tìm những sinh viên thuộc lớp "L001" mà đã có điểm môn học
SELECT *
FROM SINHVIEN
WHERE MALOP = 'L001'
AND EXISTS (
    SELECT 1
    FROM KETQUA
    WHERE SINHVIEN.MASV = KETQUA.MASV);
```

-- Toán tử NOT EXISTS

-- VD : tìm các sinh viên chưa có kết quả học tập

MASV VARCHAR(10)	HOTEN VARCHAR(50)	NGSINH DATE	DCHI VARCHAR(50)	GIOITINH VARCHAR(50)	MALOP CHAR(10)
SV01	Nguyễn Thị Lan	(null)	TPHCM	Nam	L001
SV02	Trần Thành Tùng	4/2/1980	VUNG TAU	Nam	L001

-- Toán tử NOT EXISTS

-- VD : tìm các sinh viên chưa có kết quả học tập

```
SELECT *
FROM SINHVIEN
WHERE NOT EXISTS (
    SELECT 1
    FROM KETQUA
    WHERE SINHVIEN.MASV = KETQUA.MASV);
```

```
-- Toán tử NOT EXISTS
-- VD : tìm các sinh viên chưa có kết quả học tập
SELECT *
FROM SINHVIEN
WHERE NOT EXISTS (
    SELECT 1
    FROM KETQUA
    WHERE SINHVIEN.MASV = KETQUA.MASV);
```

MASV VARCHAR(10)	HOTEN VARCHAR(50)	NGSINH DATE	DCHI VARCHAR(50)	GIOITINH VARCHAR(50)	MALOP CHAR(10)
SV03	Trương Thị Huế	10/11/1980	DA NANG	Nữ	L001

-- Mệnh đề ORDER BY của MySQL ((sắp xếp))

-- ASC sắp xếp thứ tự tăng dần

-- sắp xếp danh sách sinh viên theo họ tên (HOTEN) theo thứ tự t

SELECT * FROM SINHVIEN

ORDER BY HOTEN ASC;

```
-- ASC sắp xếp thứ tự tăng dần
-- sắp xếp danh sách sinh viên theo họ tên (HOTEN) theo thứ tự t
SELECT * FROM SINHVIEN
ORDER BY HOTEN ASC;
-- DESC sắp xếp theo thứ tự giảm dần
-- sắp xếp danh sách sinh viên theo họ tên (HOTEN) theo thứ tự giảm
SELECT * FROM SINHVIEN
ORDER BY HOTEN DESC;
```

Table (read-only)

MASV	HOTEN	NGSINH	DCHI	GIOITINH	MALOP
SV10	Đỗ Thị Hạnh	3/15/1982	DONG NAI	Nữ	L004
SV11	Hà Thành Tùng	4/26/1983	VUNG TAU	Nam	L005
SV16	Lê Thị Anh	5/15/1987	HA NOI	Nữ	L009
SV04	Lê Văn Khánh	7/4/1985	VUNG TAU	Nam	L002
SV05	Ngô Việt Nam	(null)	DA NANG	Nam	L002

Record 1 of 20

-- DESC sắp xếp theo thứ tự giảm dần

-- sắp xếp danh sách sinh viên theo họ tên (HOTEN) theo thứ tự giảm dần

SELECT * FROM SINHVIEN

ORDER BY HOTEN DESC;

```
SELECT * FROM SINHVIEN
ORDER BY HOTEN ASC;
-- DESC sắp xếp theo thứ tự giảm dần
-- sắp xếp danh sách sinh viên theo họ tên (HOTEN) theo thứ tự giảm dần
SELECT * FROM SINHVIEN
ORDER BY HOTEN DESC;
-- ORDER BY với các thuộc tính ASC hoặc DESC
```

Table (read-only)

MASV	HOTEN	NGSINH	DCHI	GIOITINH	MALOP
SV17	Trương Thị Nhàn	(null)	TPHCM	Nữ	L010
SV03	Trương Thị Huế	10/11/1980	DA NANG	Nữ	L001
SV09	Trần Thị Tố Anh	7/8/1981	TPHCM	Nữ	L004
SV19	Trần Thị Thắm	2/11/1985	DONG NAI	Nữ	L012
SV06	Trần Thị Liễu	7/5/1985	TPHCM	Nữ	L003

Record 1 of 20

-- ORDER BY với các thuộc tính ASC hoặc DESC

-- sắp xếp theo lớp (MALOP) tăng dần và sau đó theo ngày sinh (NGSINH) giảm dần

SELECT * FROM SINHVIEN

ORDER BY MALOP ASC, NGSINH DESC;

```
-- ORDER BY với các thuộc tính ASC hoặc DESC
-- sắp xếp theo lớp (MALOP) tăng dần và sau đó theo ngày sinh (NGSINH) giảm dần
SELECT * FROM SINHVIEN
ORDER BY MALOP ASC, NGSINH DESC;
-- Với nhiều cột
-- sắp xếp danh sách sinh viên theo DIỄM (KETQUA) trước và sau đó theo tên (HOTEN)
```

Table (read-only)

MASV VARCHAR(10)	HOTEN VARCHAR(50)	NGSINH DATE	DCHI VARCHAR(50)	GIOTINH VARCHAR(50)	MALOP CHAR(10)
SV03	Trương Thị Huế	10/11/1980	DA NANG	Nữ	L001
SV02	Trần Thành Tùng	4/2/1980	VUNG TAU	Nam	L001
SV01	Nguyễn Thị Lan	(null)	TPHCM	Nam	L001
SV04	Lê Văn Khánh	7/4/1985	VUNG TAU	Nam	L002
SV05	Ngô Việt Nam	(null)	DA NANG	Nam	L002

-- Với nhiều cột

-- sắp xếp danh sách sinh viên theo DIỄM (KETQUA) trước và sau đó theo tên (HOTEN)

SELECT * FROM SINHVIEN, KETQUA

WHERE KETQUA.MASV=SINHVIEN.MASV

ORDER BY DIEM ASC, HOTEN ASC;

```
-- Với nhiều cột
-- sắp xếp danh sách sinh viên theo DIỄM (KETQUA) trước và sau đó theo tên (HOTEN)
SELECT * FROM SINHVIEN, KETQUA
WHERE KETQUA.MASV=SINHVIEN.MASV
ORDER BY DIEM ASC, HOTEN ASC;
-- ORDER BY random
-- sử dụng hàm NEWID() để sắp xếp ngẫu nhiên.
```

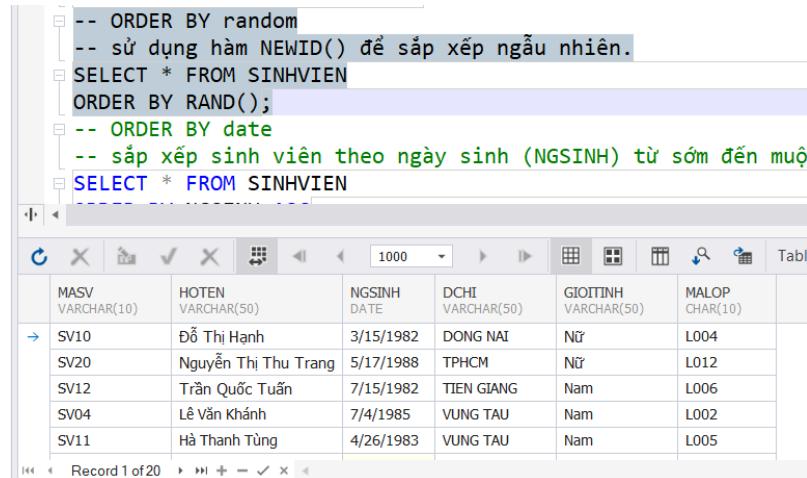
Table (read-only)

MASV VARCHAR(10)	HOTEN VARCHAR(50)	NGSINH DATE	DCHI VARCHAR(50)	GIOTINH VARCHAR(50)	MALOP CHAR(10)	MASV1 CHAR(10)	MAMH CHAR(10)	DIEM FLOAT
SV09	Trần Thị Tố Anh	7/8/1981	TPHCM	Nữ	L004	SV09	M010	3
SV01	Nguyễn Thị Lan	(null)	TPHCM	Nam	L001	SV01	M002	4
SV14	Trần Thành Phước	12/17/1984	TPHCM	Nam	L008	SV14	M005	4
SV11	Hà Thành Tùng	4/26/1983	VUNG TAU	Nam	L005	SV11	M004	5
SV04	Lê Văn Khánh	7/4/1985	VUNG TAU	Nam	L002	SV04	M001	5

-- ORDER BY random

-- sử dụng hàm NEWID() để sắp xếp ngẫu nhiên.

```
SELECT * FROM SINHVIEN
ORDER BY RAND();
```



-- ORDER BY random
-- sử dụng hàm NEWID() để sắp xếp ngẫu nhiên.

```
SELECT * FROM SINHVIEN
ORDER BY RAND();
```

-- ORDER BY date
-- sắp xếp sinh viên theo ngày sinh (NGSINH) từ sớm đến muộn

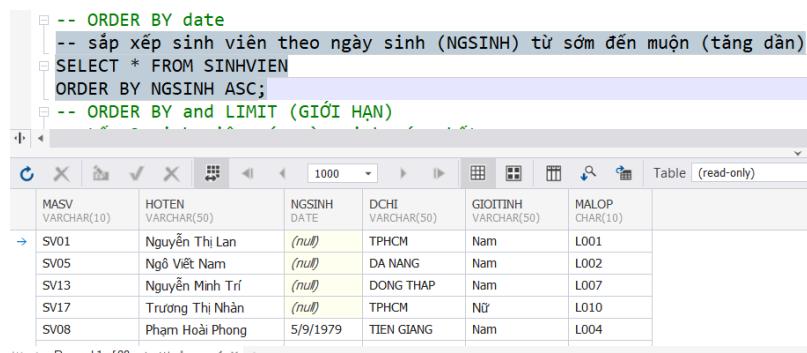
```
SELECT * FROM SINHVIEN
```

MASV VARCHAR(10)	HOTEN VARCHAR(50)	NGSINH DATE	DCHI VARCHAR(50)	GIOITINH VARCHAR(50)	MALOP CHAR(10)
SV10	Đỗ Thị Hạnh	3/15/1982	DONG NAI	Nữ	L004
SV20	Nguyễn Thị Thu Trang	5/17/1988	TPHCM	Nữ	L012
SV12	Trần Quốc Tuấn	7/15/1982	TIEN GIANG	Nam	L006
SV04	Lê Văn Khánh	7/4/1985	VUNG TAU	Nam	L002
SV11	Hà Thành Tùng	4/26/1983	VUNG TAU	Nam	L005

Record 1 of 20

-- ORDER BY date
-- sắp xếp sinh viên theo ngày sinh (NGSINH) từ sớm đến muộn (tăng dần)

```
SELECT * FROM SINHVIEN
ORDER BY NGSINH ASC;
```



-- ORDER BY date
-- sắp xếp sinh viên theo ngày sinh (NGSINH) từ sớm đến muộn (tăng dần)

```
SELECT * FROM SINHVIEN
ORDER BY NGSINH ASC;
```

-- ORDER BY and LIMIT (GIỚI HẠN)

MASV VARCHAR(10)	HOTEN VARCHAR(50)	NGSINH DATE	DCHI VARCHAR(50)	GIOITINH VARCHAR(50)	MALOP CHAR(10)
SV01	Nguyễn Thị Lan	(null)	TPHCM	Nam	L001
SV05	Ngô Việt Nam	(null)	DA NANG	Nam	L002
SV13	Nguyễn Minh Trí	(null)	DONG THAP	Nam	L007
SV17	Trương Thị Nhàn	(null)	TPHCM	Nữ	L010
SV08	Phạm Hoài Phong	5/9/1979	TIEN GIANG	Nam	L004

Record 1 of 20

-- ORDER BY and LIMIT (GIỚI HẠN)
-- Lấy 3 sinh viên có ngày sinh sớm nhất

```
SELECT * FROM SINHVIEN
```

ORDER BY NGSINH ASC

LIMIT 3;

```
-- ORDER BY and LIMIT (GIỚI HẠN)
-- Lấy 3 sinh viên có ngày sinh sớm nhất
SELECT * FROM SINHVIEN
ORDER BY NGSINH ASC
LIMIT 3;
```

MASV VARCHAR(10)	HOTEN VARCHAR(50)	NGSINH DATE	DCHI VARCHAR(50)	GIOITINH VARCHAR(50)	MALOP CHAR(10)
SV01	Nguyễn Thị Lan	(null)	TPHCM	Nam	L001
SV13	Nguyễn Minh Trí	(null)	DONG THAP	Nam	L007
SV05	Ngô Việt Nam	(null)	DA NANG	Nam	L002

-- ORDER BY and NULL

-- sắp xếp sinh viên theo ngày sinh và muôn các giá trị NULL xuất hiện trước

SELECT * FROM SINHVIEN

ORDER BY NGSINH ASC;

```
-- ORDER BY and NULL
-- sắp xếp sinh viên theo ngày sinh và muôn các giá trị NULL xuất hiện trước
SELECT * FROM SINHVIEN
ORDER BY NGSINH ASC;
```

MASV VARCHAR(10)	HOTEN VARCHAR(50)	NGSINH DATE	DCHI VARCHAR(50)	GIOITINH VARCHAR(50)	MALOP CHAR(10)
SV01	Nguyễn Thị Lan	(null)	TPHCM	Nam	L001
SV05	Ngô Việt Nam	(null)	DA NANG	Nam	L002
SV13	Nguyễn Minh Trí	(null)	DONG THAP	Nam	L007
SV17	Trương Thị Nhàn	(null)	TPHCM	Nữ	L010
SV08	Phạm Hải Phong	5/9/1979	TIEN GIANG	Nam	L004

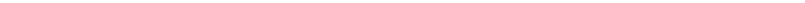
-- ORDER BY length of characters (độ dài của kí tự)

-- sắp xếp độ dài giảm dần của tên khoa

SELECT * FROM KHOAHOC

ORDER BY LENGTH(TENKHOA) DESC

```
-- ORDER BY length of characters (độ dài của kí tự)
-- sắp xếp độ dài giảm dần của tên khoa
SELECT * FROM KHOAHOC
ORDER BY LENGTH(TENKHOA) DESC
-- ORDER BY and arithmetic operators (toán tử số học)
-- Điểm trung bình sắp xếp theo sinh viên
SELECT MASV,
```



MAKHOA VARCHAR(10)	TENKHOA VARCHAR(50)
TP	THUC PHAM
DT	ĐIEN TU
SH	SINH HOC
HH	HOA HOC
TH	TIN HOC

Record 1 of 6

```
-- ORDER BY and arithmetic operators (toán tử số học)
-- Điểm trung bình sắp xếp theo sinh viên
SELECT MASV,
AVG(DIEM) AS DIEM_TRUNG_BINH
FROM KETQUA
GROUP BY MASV
ORDER BY DIEM_TRUNG_BINH DESC;
```

```
-- ORDER BY and arithmetic operators (toán tử số học)
-- Điểm trung bình sắp xếp theo sinh viên
SELECT MASV,
AVG(DIEM) AS DIEM_TRUNG_BINH
FROM KETQUA
GROUP BY MASV
ORDER BY DIEM_TRUNG_BINH DESC;
-- ORDER BY using a custom list (danh sách tùy chỉnh )
```

MASV	DIEM_TRUNG_BINH
SV06	10
SV16	10
SV05	9
SV18	9
SV12	8

```
-- ORDER BY using a custom list (danh sách tùy chỉnh )
-- sắp xếp bảng KHOAHOC theo thứ tự ưu tiên như sau: TH, VL, DT, TP, HH, SH
SELECT *
FROM KHOAHOC
ORDER BY
CASE MAKHOA
WHEN 'TH' THEN 2
WHEN 'VL' THEN 4
WHEN 'DT' THEN 6
WHEN 'TP' THEN 1
WHEN 'HH' THEN 5
WHEN 'SH' THEN 3
END;
```

```
-- ORDER BY using a custom list (danh sách tùy chỉnh )
-- sắp xếp bảng KHOAHOC theo thứ tự ưu tiên như sau: TH, VL, DT, TP, HH, SH
SELECT *
FROM KHOAHOC
ORDER BY
CASE MAKHOA
    WHEN 'TH' THEN 2
    WHEN 'VL' THEN 4
    WHEN 'DT' THEN 6
    WHEN 'TP' THEN 1
    WHEN 'HH' THEN 5
    WHEN 'SH' THEN 3
END;
-- ORDER BY with GROUP BY and aggregate functions (các hàm tổ hợp)
-- tìm điểm cao nhất của mỗi môn học và sắp xếp theo môn học
SELECT MAMH,
```

MAKHOA	TENKHOA
TP	THUC PHAM
TH	TIN HOC
SH	SINH HOC
VL	VAT LY
HH	HOA HOC

-- ORDER BY with GROUP BY and aggregate functions (các hàm tổ hợp)

-- tìm điểm cao nhất của mỗi môn học và sắp xếp theo môn học

```
SELECT MAMH,
MAX(DIEM) AS DIEM_CAO_NHAT
FROM KETQUA
GROUP BY MAMH
ORDER BY MAMH;
```

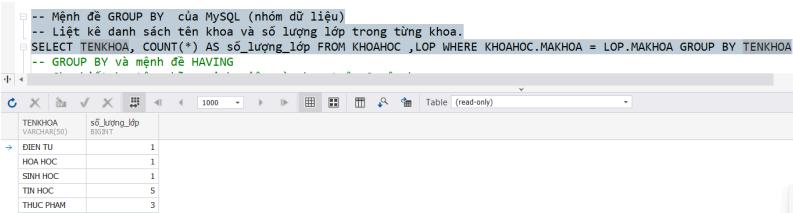
```
-- ORDER BY with GROUP BY and aggregate functions (các hàm tổ hợp)
-- tìm điểm cao nhất của mỗi môn học và sắp xếp theo môn học
SELECT MAMH,
MAX(DIEM) AS DIEM_CAO_NHAT
FROM KETQUA
GROUP BY MAMH
ORDER BY MAMH;
```

MAMH	DIEM_CAO_NHAT
M001	8
M002	10
M003	9
M004	8
M005	9

-- Mệnh đề GROUP BY của MySQL (nhóm dữ liệu)

-- Liệt kê danh sách tên khoa và số lượng lớp trong từng khoa.

```
SELECT TENKHOA, COUNT(*) AS số_lượng_lớp FROM KHOAHOC ,LOP WHERE KHOAHOC.MAKHOA = LOP.MAKHOA GROUP BY TENKHOA
```

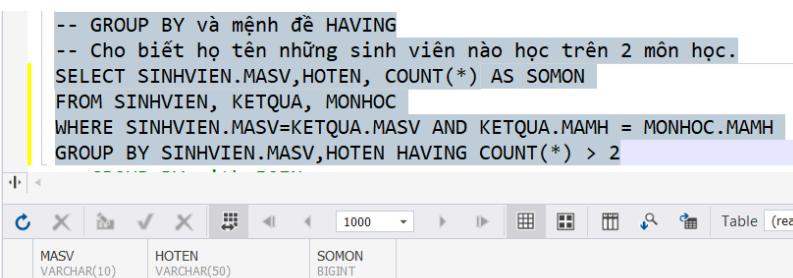


TENKHOA	số_lượng_lớp
ĐIỀN TỰ	1
HÓA HỌC	1
SINH HỌC	1
TIN HỌC	5
THỰC PHẨM	3

-- GROUP BY và mệnh đề HAVING

-- Cho biết họ tên những sinh viên nào học trên 2 môn học.

```
SELECT SINHVIEN.MASV,HOTEN, COUNT() AS SOMON FROM SINHVIEN, KETQUA, MONHOC WHERE SINHVIEN.MASV=KETQUA.MASV AND KETQUA.MAMH = MONHOC.MAMH GROUP BY SINHVIEN.MASV,HOTEN HAVING COUNT() > 2
```



MASV	HOTEN	SOMON
SV01	Nguyễn Thị Lan	3
SV20	Nguyễn Thị Thu Trang	5

-- GROUP BY with JOIN

-- Tính điểm trung bình của mỗi sinh viên cho các môn học đã thi

```
SELECT HOTEN, AVG(DIEM) AS AverageScore
```

```

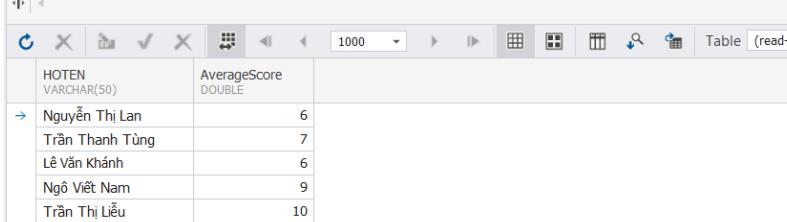
FROM SINHVIEN
JOIN KETQUA ON SINHVIEN.MASV = KETQUA.MASV
GROUP BY HOTEN;

```

```

-- GROUP BY with JOIN
-- Tính điểm trung bình của mỗi sinh viên cho các môn học đã thi
SELECT HOTEN, AVG(DIEM) AS AverageScore
FROM SINHVIEN
JOIN KETQUA ON SINHVIEN.MASV = KETQUA.MASV
GROUP BY HOTEN;
-- GROUP BY và trình sửa đổi WITH ROLLUP

```



-- GROUP BY và trình sửa đổi WITH ROLLUP

-- Tính số lượng sinh viên trong mỗi lớp và tổng số sinh viên trong tất cả các lớp

```

SELECT TENLOP, COUNT(MASV) AS SOLUONGSINHVIEN

```

```

FROM LOP

```

```

JOIN SINHVIEN ON SINHVIEN.MALOP = LOP.MALOP

```

```

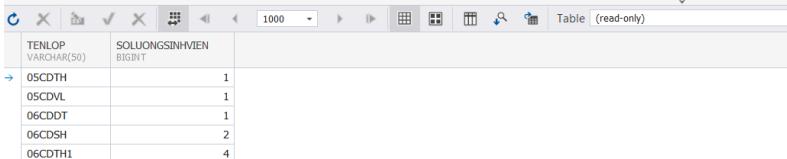
GROUP BY TENLOP WITH ROLLUP;

```

```

-- GROUP BY và trình sửa đổi WITH ROLLUP
-- Tính số lượng sinh viên trong mỗi lớp và tổng số sinh viên trong tất cả các lớp
SELECT TENLOP, COUNT(MASV) AS SOLUONGSINHVIEN
FROM LOP
JOIN SINHVIEN ON SINHVIEN.MALOP = LOP.MALOP
GROUP BY TENLOP WITH ROLLUP;
-- GROUP BY and ORDER BY

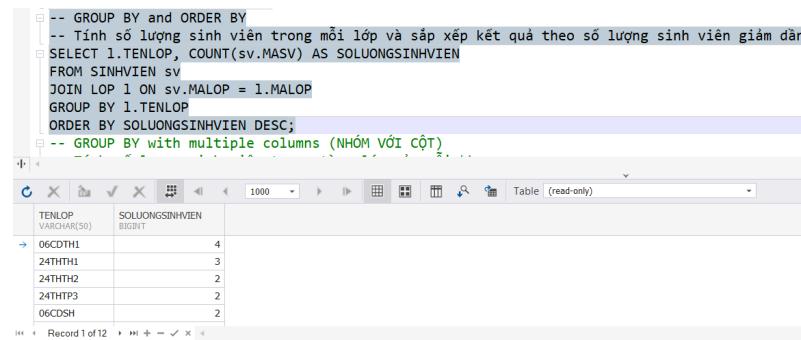
```



-- GROUP BY and ORDER BY

-- Tính số lượng sinh viên trong mỗi lớp và sắp xếp kết quả theo số lượng sinh viên giảm dần

```
SELECT I.TENLOP, COUNT(sv.MASV) AS SOLUONGSINHVIEN
FROM SINHVIEN sv
JOIN LOP I ON sv.MALOP = I.MALOP
GROUP BY I.TENLOP
ORDER BY SOLUONGSINHVIEN DESC;
```



```
-- GROUP BY and ORDER BY
-- Tính số lượng sinh viên trong mỗi lớp và sắp xếp kết quả theo số lượng sinh viên giảm dần
SELECT I.TENLOP, COUNT(sv.MASV) AS SOLUONGSINHVIEN
FROM SINHVIEN sv
JOIN LOP I ON sv.MALOP = I.MALOP
GROUP BY I.TENLOP
ORDER BY SOLUONGSINHVIEN DESC;
-- GROUP BY with multiple columns (NHÓM VỚI CỘT)
```

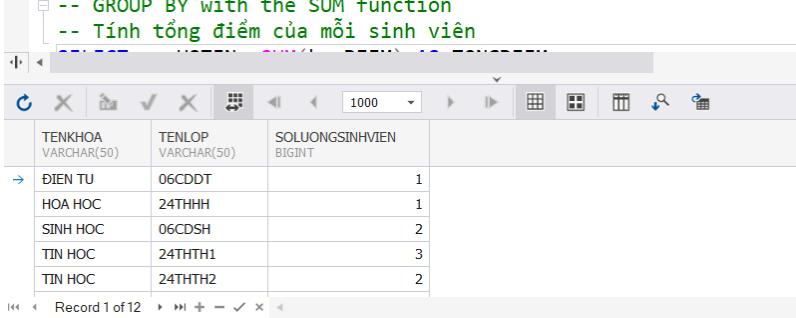
TENLOP	SOLUONGSINHVIEN
06CDTH1	4
24HTH1	3
24HTH2	2
24HTP3	2
06CDSH	2

-- GROUP BY with multiple columns (NHÓM VỚI CỘT)

-- Tính số lượng sinh viên trong từng lớp của mỗi khoa

```
SELECT k.TENKHOA, I.TENLOP, COUNT(sv.MASV) AS SOLUONGSINHVIEN
FROM SINHVIEN sv
JOIN LOP I ON sv.MALOP = I.MALOP
JOIN KHOAHOC k ON I.MAKHOA = k.MAKHOA
GROUP BY k.TENKHOA, I.TENLOP;
```

```
-- GROUP BY with multiple columns (NHÓM VỚI CỘT)
-- Tính số lượng sinh viên trong từng lớp của mỗi khoa
SELECT k.TENKHOA, l.TENLOP, COUNT(sv.MASV) AS SOLUONGSINHVIEN
FROM SINHVIEN sv
JOIN LOP l ON sv.MALOP = l.MALOP
JOIN KHOAHOC k ON l.MAKHOA = k.MAKHOA
GROUP BY k.TENKHOA, l.TENLOP;
-- GROUP BY with the SUM function
-- Tính tổng điểm của mỗi sinh viên
```



TENKHOA VARCHAR(50)	TENLOP VARCHAR(50)	SOLUONGSINHVIEN BIGINT
ĐIEN TU	06CDDT	1
HOA HOC	24THHH	1
SINH HOC	06CDSH	2
TIN HOC	24THTH1	3
TIN HOC	24THTH2	2

-- GROUP BY with the SUM function

-- Tính tổng điểm của mỗi sinh viên

```
SELECT sv.HOTEN, SUM(kq.DIEM) AS TONGDIEM
FROM SINHVIEN sv
JOIN KETQUA kq ON sv.MASV = kq.MASV
GROUP BY sv.HOTEN
ORDER BY TONGDIEM ASC;
```

```
-- GROUP BY with the SUM function
-- Tính tổng điểm của mỗi sinh viên
SELECT sv.HOTEN, SUM(kq.DIEM) AS TONGDIEM
FROM SINHVIEN sv
JOIN KETQUA kq ON sv.MASV = kq.MASV
GROUP BY sv.HOTEN
ORDER BY TONGDIEM ASC;
-- GROUP BY với hàm MAX,MIN
-- Tìm điểm cao nhất của mỗi sinh viên
```

HOTEN VARCHAR(50)	TONGDIEM DOUBLE
Trần Thị Tố Anh	3
Hà Thanh Tùng	5
Trần Thanh Tùng	7
Phạm Hoài Phong	7
Nguyễn Phước Sang	7

-- GROUP BY với hàm MAX,MIN

-- Tìm điểm cao nhất của mỗi sinh viên

```
SELECT sv.HOTEN, MAX(kq.DIEM) AS DIEMCAONHAT
FROM SINHVIEN sv
JOIN KETQUA kq ON sv.MASV = kq.MASV
GROUP BY sv.HOTEN;
```

```

-- GROUP BY với hàm MAX,MIN
-- Tìm điểm cao nhất của mỗi sinh viên
SELECT sv.HOTEN, MAX(kq.DIEM) AS DIEMCAONHAT
FROM SINHVIEN sv
JOIN KETQUA kq ON sv.MASV = kq.MASV
GROUP BY sv.HOTEN;
-- GROUP BY và mệnh đề DISTINCT (loại bỏ các

```

Record 1 of 19

HOTEN VARCHAR(50)	DIEMCAONHAT FLOAT
Nguyễn Thị Lan	8
Trần Thanh Tùng	7
Lê Văn Khánh	7
Ngô Viết Nam	9
Trần Thị Liễu	10

-- GROUP BY và mệnh đề DISTINCT (loại bỏ các vòng lặp)

```

SELECT DISTINCT sv.MALOP,
COUNT(DISTINCT sv.MASV) AS SOSINHVIENKHACBIET
FROM SINHVIEN sv
GROUP BY sv.MALOP;

```

```

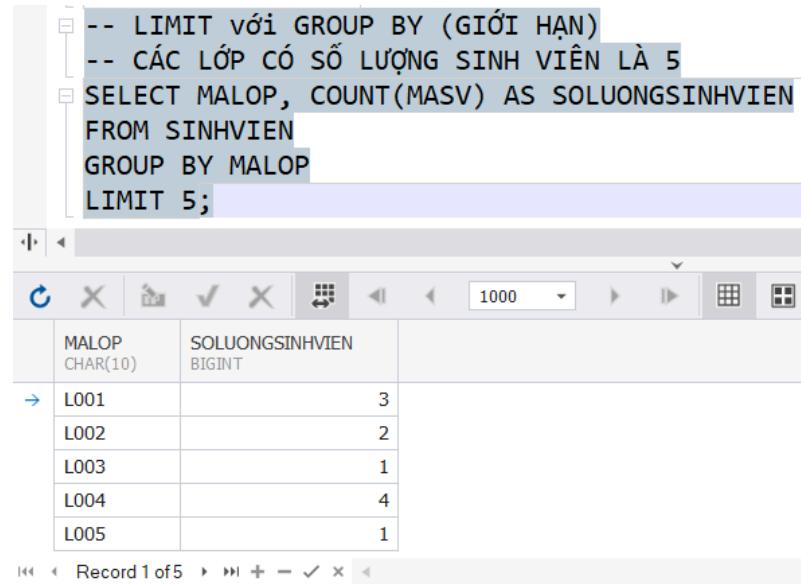
-- GROUP BY và mệnh đề DISTINCT (loại bỏ các vòng lặp)
SELECT DISTINCT sv.MALOP,
COUNT(DISTINCT sv.MASV) AS SOSINHVIENKHACBIET
FROM SINHVIEN sv
GROUP BY sv.MALOP;
-- LIMIT với GROUP BY (GIỚI HẠN)

```

Record 1 of 12

MALOP CHAR(10)	SOSINHVIENKHACBIET BIGINT
L001	3
L002	2
L003	1
L004	4
L005	1

```
-- LIMIT với GROUP BY (GIỚI HẠN)
-- CÁC LỚP CÓ SỐ LƯỢNG SINH VIÊN LÀ 5
SELECT MALOP, COUNT(MASV) AS SOLUONGSINHVIEN
FROM SINHVIEN
GROUP BY MALOP
LIMIT 5;
```



The screenshot shows the MySQL Workbench interface. The SQL editor contains the query:

```
-- LIMIT với GROUP BY (GIỚI HẠN)
-- CÁC LỚP CÓ SỐ LƯỢNG SINH VIÊN LÀ 5
SELECT MALOP, COUNT(MASV) AS SOLUONGSINHVIEN
FROM SINHVIEN
GROUP BY MALOP
LIMIT 5;
```

The results grid shows the following data:

MALOP	SOLUONGSINHVIEN
L001	3
L002	2
L003	1
L004	4
L005	1

At the bottom, the status bar indicates "Record 1 of 5".

-- Câu lệnh IF của MySQL

IF condition THEN

-- các câu lệnh thực hiện nếu điều kiện đúng

ELSE

-- các câu lệnh thực hiện nếu điều kiện sai

END IF;

-- Hàm IF của MySQL

SELECT

```

sv.MASV,
sv.HOTEN,
kq.MAMH,
kq.DIEM,
IF(kq.DIEM >= 5, 'Đậu', 'Rớt') AS KETQUA
FROM
SINHVIEN sv
JOIN
KETQUA kq ON sv.MASV = kq.MASV;

```

-- Hàm IF của MySQL

```

SELECT
    sv.MASV,
    sv.HOTEN,
    kq.MAMH,
    kq.DIEM,
    IF(kq.DIEM >= 5, 'Đậu', 'Rớt') AS KETQUA
FROM
    SINHVIEN sv
JOIN
    KETQUA kq ON sv.MASV = kq.MASV;

```

-- Hàm NULLIF

```

SELECT HOTEN,
    NULLIF(DIEM, 0) AS DiemKhongBangKhong
FROM
    KETQUA, SINHVIEN;

```

-- IFNULL

HOTEN	DiemKhongBangKhong
VARCHAR(50)	FLOAT
Nguyễn Thị Thu Trang	8
Trần Thị Thắm	8
Nguyễn Kim Phụng	8
Trần Thị Nhàn	8

Record 1 of 580

-- Hàm NULLIF

```
SELECT HOTEN,
NULLIF(DIEM, 0) AS DiemKhongBangKhong
FROM
KETQUA,SINHVIEN;
```

```
-- Toán tử In
-- Vd : Tìm các sinh viên có mã lớp là "L001", "L002", hoặc "L003".
SELECT *
FROM SINHVIEN
WHERE MALOP IN ('L001', 'L002', 'L003');

-- Toán tử Not In
-- Vd : Tìm các sinh viên không có mã lớp là "L001", "L002", hoặc "L003".
```

MASV VARCHAR(10)	HOTEN VARCHAR(50)	NGSINH DATE	DCHI VARCHAR(50)	GIOITINH VARCHAR(50)	MALOP CHAR(10)
SV01	Nguyễn Thị Lan	(null)	TPHCM	Nam	L001
SV02	Trần Thanh Tùng	4/2/1980	VUNG TAU	Nam	L001
SV03	Trương Thị Huệ	10/11/1980	DA NANG	Nữ	L001
SV04	Lê Văn Khánh	7/4/1985	VUNG TAU	Nam	L002
SV05	Ngô Việt Nam	(null)	DA NANG	Nam	L002
SV06	Trần Thị Liễu	7/5/1985	TPHCM	Nữ	L003

-- IFNULL

```
SELECT
MASV,
HOTEN,
IFNULL(DCHI, 'Chưa có địa chỉ') AS DiaChi
FROM
SINHVIEN;
```

```
-- IFNULL
SELECT
    MASV,
    HOTEN,
    IFNULL(DCHI, 'Chưa có địa chỉ') AS DiaChi
FROM
    SINHVIEN;
```

MASV VARCHAR(10)	HOTEN VARCHAR(50)	DiaChi VARCHAR(50)
SV01	Nguyễn Thị Lan	TPHCM
SV02	Trần Thanh Tùng	VUNG TAU
SV03	Trương Thị Huế	DA NANG
SV04	Lê Văn Khánh	VUNG TAU
SV05	Ngô Việt Nam	DA NANG

-- SELECT của MySQL

-- Cho biết họ tên, ngày sinh, địa chỉ của những sinh viên nam học lớp '24THTH1'.

```
SELECT HOTEN,NGSINH,DCHI
```

```
FROM SINHVIEN,LOP
```

```
WHERE GIOITINH = 'NAM' AND TENLOP = '24THTH1'
```

-- SELECT của MySQL		
-- Cho biết họ tên, ngày sinh, địa chỉ của những sinh viên nam học lớp '24THTH1'		
HOTEN VARCHAR(50)	NGSINH DATE	DCHI VARCHAR(50)
Nguyễn Thị Lan	(null)	TPHCM
Trần Thanh Tùng	4/2/1980	VUNG TAU
Lê Văn Khánh	7/4/1985	VUNG TAU
Ngô Việt Nam	(null)	DA NANG
Trần Thành Nam	4/2/1988	DONG NAI

-- Hoạt động UPSERT của MySQL (Kết hợp các bản cập nhật dữ liệu hiện có và thêm các bản ghi mới.)

-- MySQL UPSERT using INSERT IGNORE (CHÈN THÔNG TIN VÀO BẢNG)

```
INSERT IGNORE INTO SINHVIEN (MASV, HOTEN, MALOP)  
VALUES ('SV021', 'Nguyễn Văn ANH', 'L001');
```

-- MySQL UPSERT sử dụng REPLACE (THAY THẾ)

```
REPLACE INTO SINHVIEN (MASV, HOTEN, MALOP)  
VALUES ('SV01', 'Nguyễn Văn B', 'L002');
```

-- MySQL UPSERT sử dụng INSERT ON DUPLICATE KEY UPDATE (chèn một sinh viên mới vào bảng)

```
INSERT INTO SINHVIEN (MASV, HOTEN, MALOP)  
VALUES ('SV22', 'Nguyễn Thị C', 'L003') AS new_val  
ON DUPLICATE KEY UPDATE  
HOTEN = new_val.HOTEN,  
MALOP = new_val.MALOP;
```

-- Truy vấn MySQL UPDATE (cập nhật)

-- câu lệnh UPDATE của MySQL

```
UPDATE SINHVIEN  
SET HOTEN = 'Nguyễn Thị Lan'  
WHERE MASV = 'SV01';
```

-- UPDATE DỮ LIỆU TRONG MỘT BẢNG BẰNG DỮ LIỆU TỪ BẢNG KHÁC

-- cập nhật địa chỉ sinh viên (cột DCHI) thành "Hà Nội" cho tất cả sinh viên học lớp thuộc khoa có mã khoa TP

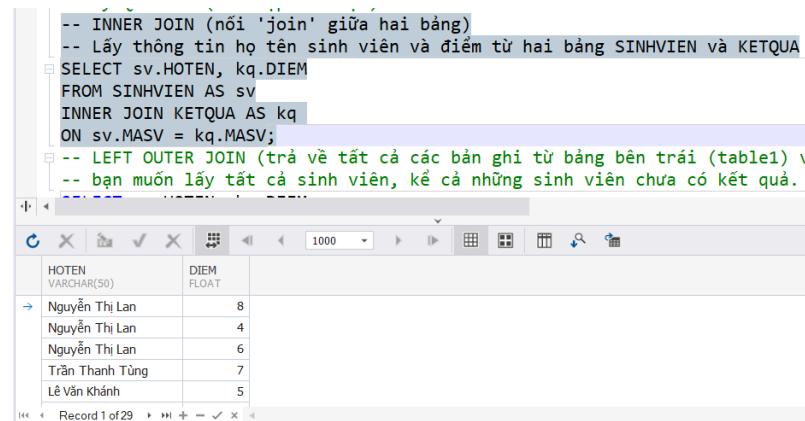
```
UPDATE SINHVIEN AS SV  
JOIN LOP AS L ON SV.MALOP = L.MALOP  
JOIN KHOAHOC AS KH ON L.MAKHOA = KH.MAKHOA
```

```
SET SV.DCHI = 'HÀ NỘI'  
WHERE KH.MAKHOA = 'TP'
```

-- MySQL JOIN (kết hợp dữ liệu)

-- INNER JOIN (nối 'join' giữa hai bảng)
-- Lấy thông tin họ tên sinh viên và điểm từ hai bảng SINHVIEN và KETQUA

```
SELECT sv.HOTEN, kq.DIEM  
FROM SINHVIEN AS sv  
INNER JOIN KETQUA AS kq  
ON sv.MASV = kq.MASV;
```



The screenshot shows the MySQL Workbench interface. The query editor contains the SQL code for an INNER JOIN. The results pane displays a table with two columns: HOTEN (VARCHAR(50)) and DIEM (FLOAT). The data shows five rows: Nguyễn Thị Lan with a score of 8, Nguyễn Thị Lan with a score of 4, Nguyễn Thị Lan with a score of 6, Trần Thanh Tùng with a score of 7, and Lê Văn Khánh with a score of 5. The status bar at the bottom indicates 'Record 1 of 29'.

HOTEN	DIEM
Nguyễn Thị Lan	8
Nguyễn Thị Lan	4
Nguyễn Thị Lan	6
Trần Thanh Tùng	7
Lê Văn Khánh	5

-- LEFT OUTER JOIN (trả về tất cả các bản ghi từ bảng bên trái (table1) và các bản ghi khớp từ bảng bên phải (table2).)
-- bạn muốn lấy tất cả sinh viên, kể cả những sinh viên chưa có kết quả.

```
SELECT sv.HOTEN, kq.DIEM  
FROM SINHVIEN AS sv  
LEFT OUTER JOIN KETQUA AS kq ON sv.MASV = kq.MASV;
```

```
-- LEFT OUTER JOIN (trả về tất cả các bản ghi từ bảng bên trái (table1) và
-- bạn muốn lấy tất cả sinh viên, kể cả những sinh viên chưa có kết quả.
SELECT sv.HOTEN, kq.DIEM
FROM SINHVIEN AS sv
LEFT OUTER JOIN KETQUA AS kq ON sv.MASV = kq.MASV;
-- RIGHT OUTER JOIN (rả về tất cả các bản ghi từ bảng bên phải (table2) và
-- lấy tất cả kết quả trong bảng KETQUA, kể cả những môn học không có sinh
-- viên nào.

```

HOTEN VARCHAR(50)	DIEM FLOAT
Nguyễn Thị Lan	8
Nguyễn Thị Lan	4
Nguyễn Thị Lan	6
Trần Thanh Tùng	7
Nguyễn Văn ANH	(null)

-- RIGHT OUTER JOIN (rả về tất cả các bản ghi từ bảng bên phải (table2) và các bản ghi khớp từ bảng bên trái (table1).)
-- lấy tất cả kết quả trong bảng KETQUA, kể cả những môn học không có sinh viên nào.

```
SELECT sv.HOTEN, kq.DIEM
FROM SINHVIEN AS sv
RIGHT OUTER JOIN KETQUA AS kq
ON sv.MASV = kq.MASV;
```

```
-- RIGHT OUTER JOIN (rả về tất cả các bản ghi từ bảng bên phải (table2) và
-- lấy tất cả kết quả trong bảng KETQUA, kể cả những môn học không có sinh
-- viên nào.
SELECT sv.HOTEN, kq.DIEM
FROM SINHVIEN AS sv
RIGHT OUTER JOIN KETQUA AS kq
ON sv.MASV = kq.MASV;
-- FULL OUTER JOIN (Trả về tất cả các bản ghi từ cả hai bảng)

```

HOTEN VARCHAR(50)	DIEM FLOAT
Nguyễn Thị Lan	8
Nguyễn Thị Lan	4
Nguyễn Thị Lan	6
Trần Thanh Tùng	7
Lê Văn Khanh	5

-- FULL OUTER JOIN (Trả về tất cả các bản ghi từ cả hai bảng)
-- lấy tất cả sinh viên và tất cả kết quả, kể cả những sinh viên không có kết quả và những kết quả không có sinh viên

```
SELECT
FROM SINHVIEN AS sv
LEFT JOIN KETQUA AS kq ON sv.MASV = kq.MASV
```

UNION ALL

SELECT

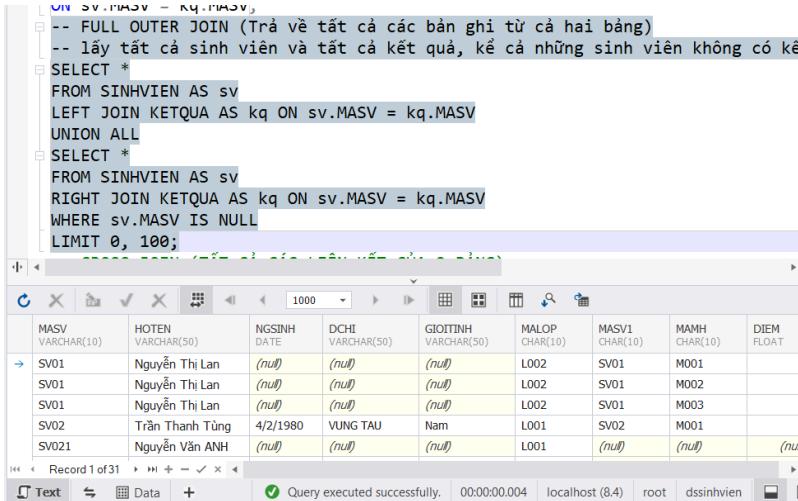
FROM SINHVIEN AS sv

RIGHT JOIN KETQUA AS kq ON sv.MASV = kq.MASV

WHERE sv.MASV IS NULL

LIMIT 0, 100;

```
UNION sv.MASV = kq.MASV;
-- FULL OUTER JOIN (Trả về tất cả các bản ghi từ cả hai bảng)
-- lấy tất cả sinh viên và tất cả kết quả, kể cả những sinh viên không có kết
SELECT *
FROM SINHVIEN AS sv
LEFT JOIN KETQUA AS kq ON sv.MASV = kq.MASV
UNION ALL
SELECT *
FROM SINHVIEN AS sv
RIGHT JOIN KETQUA AS kq ON sv.MASV = kq.MASV
WHERE sv.MASV IS NULL
LIMIT 0, 100;
```



-- CROSS JOIN (TẤT CẢ CÁC LIÊN KẾT CỦA 2 BẢNG)

-- LẤY TẤT CẢ CÁC MÔN HỌC MÀ CÁC SINH VIÊN ĐANG HỌC

SELECT sv.HOTEN, mh.TENMH

FROM SINHVIEN AS sv

CROSS JOIN MONHOC AS mh;

```
-- CROSS JOIN (TẤT CẢ CÁC LIÊN KẾT CỦA 2 BẢNG)
-- LẤY TẤT CẢ CÁC MÔN HỌC MÀ CÁC SINH VIÊN ĐANG HỌC
SELECT sv.HOTEN, mh.TENMH
FROM SINHVIEN AS sv
CROSS JOIN MONHOC AS mh;
```

HOTEN VARCHAR(50)	TENMH VARCHAR(50)
Nguyễn Thị Lan	VISUAL BASIC
Nguyễn Thị Lan	LAP TRINH QUAN LY
Nguyễn Thị Lan	TOAN ROI RAC
Nguyễn Thị Lan	MANG MAY TINH
Nguyễn Thị Lan	TIN HOC CAN BAN

Record 1 of 252

Text Data + ✓ Query executed successfully. 00:00:00.004 local

-- MySQL UNION (kết hợp kết quả của hai hoặc nhiều câu lệnh SELECT thành một tập hợp duy nhất)

-- UNION cơ bản

```
SELECT MASV, HOTEN
FROM SINHVIEN
UNION
SELECT MASV, HOTEN
FROM SINHVIENMOI ;
```

```
-- MySQL UNION (kết hợp kết quả của hai hoặc nhiều câu lệnh SELECT thành một
-- UNION cơ bản
SELECT MASV, HOTEN
FROM SINHVIEN
UNION
SELECT MASV, HOTEN
FROM SINHVIENMOI ;
-- UNION ALL
```


-- UNION ALL

```
SELECT MASV,HOTEN
FROM SINHVIEN
UNION ALL
SELECT MASV, HOTEN
FROM SINHVIENMOI;
```

```
-- UNION ALL
SELECT MASV,HOTEN
FROM SINHVIEN
UNION ALL
SELECT MASV, HOTEN
FROM SINHVIENMOI;
```


-- UNION có điều kiện

```
SELECT MASV,HOTEN,GIOITINH
FROM SINHVIEN AS SV
WHERE SV.MALOP = 'L003'
```

UNION

```
SELECT MASV,HOTEN,GIOITINH
FROM SINHVIENMOI AS SVM
WHERE SVM.MALOP = 'L001';
```

The screenshot shows a database query interface with the following content:

```
-- UNION có điều kiện
SELECT MASV,HOTEN,GIOITINH
FROM SINHVIEN AS SV
WHERE SV.MALOP = 'L003'
UNION
SELECT MASV,HOTEN,GIOITINH
FROM SINHVIENMOI AS SVM
WHERE SVM.MALOP = 'L001';
-- UNION với DISTINCT
SELECT DISTINCT SINHVIEN.MASV, HOTEN, DIEM
FROM SINHVIEN,KETQUA
```

Below the code, there is a table with the following data:

	MASV VARCHAR(10)	HOTEN VARCHAR(50)	GIOITINH VARCHAR(50)
→	SV06	Trần Thị Liễu	Nữ
	SV01	Nguyễn Thị Lan	Nam
	SV25	Trần Văn Chính	Nam
	SV33	Trương Văn Thọ	Nữ

At the bottom, the status bar shows: Record 1 of 4, Text, Data, +, ✓ Query executed successfully, 00:00:00.007, local.

-- UNION với DISTINCT

```
SELECT DISTINCT SINHVIEN.MASV, HOTEN, DIEM
FROM SINHVIEN,KETQUA
UNION
SELECT DISTINCT SINHVIENMOI.MASV, HOTEN, DIEM
FROM SINHVIENMOI,KETQUA;
```

```
WHERE SINHVIEN.MASV = 'LOU1' ;
-- UNION với DISTINCT
SELECT DISTINCT SINHVIEN.MASV, HOTEN, DIEM
FROM SINHVIEN,KETQUA
UNION
SELECT DISTINCT SINHVIENMOI.MASV, HOTEN, DIEM
FROM SINHVIENMOI,KETQUA;
-- UNION đơn giản để tìm kiếm từ khóa
```

MASV VARCHAR(10)	HOTEN VARCHAR(50)	DIEM FLOAT
SV20	Nguyễn Thị Thu Trang	8
SV19	Trần Thị Thắm	8
SV18	Nguyễn Kim Phung	8
SV17	Trương Thị Nhàn	8
SV16	Lê Thị Anh	8

Record 1 of 192

Text Data + ✓ Query executed successfully. 00:00:00.008

-- UNION đơn giản để tìm kiếm từ khóa

```
SELECT MASV, HOTEN
FROM SINHVIEN
WHERE HOTEN LIKE '%Trần%'
UNION
SELECT MASV, HOTEN
FROM SINHVIENMOI
WHERE HOTEN LIKE '%Trần%';
```

```
-- UNION đơn giản để tìm kiếm từ khóa
SELECT MASV, HOTEN
FROM SINHVIEN
WHERE HOTEN LIKE '%Trần%'
UNION
SELECT MASV, HOTEN
FROM SINHVIENMOI
WHERE HOTEN LIKE '%Trần%';
-- UNION cho nhiều tìm kiếm từ khóa
```

	MASV VARCHAR(10)	HOTEN VARCHAR(50)
→	SV02	Trần Thanh Tùng
	SV06	Trần Thị Liễu
	SV07	Trần Thành Nam
	SV09	Trần Thị Tố Anh
	SV12	Trần Quốc Tuấn

Record 1 of 9

Text Data + ✓ Query executed successfully.

-- UNION cho nhiều tìm kiếm từ khóa

```
SELECT MASV, HOTEN
FROM SINHVIEN
WHERE HOTEN LIKE '%Nguyễn%'
OR HOTEN LIKE '%Trần%'
UNION
SELECT MASV, HOTEN
FROM SINHVIENMOI
WHERE HOTEN LIKE '%Nguyễn%'
OR HOTEN LIKE '%Trần%';
```

```
-- UNION cho nhiều tìm kiếm từ khóa
SELECT MASV, HOTEN
FROM SINHVIEN
WHERE HOTEN LIKE '%Nguyễn%'
    OR HOTEN LIKE '%Trần%'
UNION
SELECT MASV, HOTEN
FROM SINHVIENMOI
WHERE HOTEN LIKE '%Nguyễn%'
    OR HOTEN LIKE '%Trần%';
```

	MASV VARCHAR(10)	HOTEN VARCHAR(50)
→	SV01	Nguyễn Thị Lan
	SV02	Trần Thanh Tùng
	SV021	Nguyễn Văn ANH
	SV06	Trần Thị Liễu
	SV07	Trần Thành Nam

-- MySQL LIKE (tìm kiếm dữ liệu)

-- MySQL LIKE: (%) phần trăm ký tự đại diện

```
SELECT * FROM SINHVIEN WHERE HOTEN LIKE '%Trần%';
```

```
-- MySQL LIKE (tìm kiếm dữ liệu)
-- MySQL LIKE: (%) phần trăm ký tự đại diện
SELECT * FROM SINHVIEN WHERE HOTEN LIKE '%Trần%';



| MASV<br>VARCHAR(10) | HOTEN<br>VARCHAR(50) | NGSINH<br>DATE | DCHI<br>VARCHAR(50) | GIOITINH<br>VARCHAR(50) | MALOP<br>CHAR(10) |
|---------------------|----------------------|----------------|---------------------|-------------------------|-------------------|
| SV02                | Trần Thanh Tùng      | 4/2/1980       | VUNG TAU            | Nam                     | L001              |
| SV06                | Trần Thị Liễu        | 7/5/1985       | TPHCM               | Nữ                      | L003              |
| SV07                | Trần Thành Nam       | 4/2/1988       | DONG NAI            | Nam                     | L004              |
| SV09                | Trần Thị Tố Anh      | 7/8/1981       | TPHCM               | Nữ                      | L004              |
| SV12                | Trần Quốc Tuấn       | 7/15/1982      | TIEN GIANG          | Nam                     | L006              |



Record 1 of 8



Text Data + ✓ Query executed successfully. 00:00:00.008 localhost (8.4)


```

-- MySQL LIKE: (_) ký tự gạch dưới

```
SELECT * FROM SINHVIEN WHERE HOTEN LIKE 'Tr__';
```

```
-- MySQL LIKE: (_) ký tự gạch dưới
SELECT * FROM SINHVIEN WHERE HOTEN LIKE 'Tr__';
-- MySQL LIKE: kết hợp các ký tự đại diện (%) và (_)
```

MASV VARCHAR(10)	HOTEN VARCHAR(50)	NGSINH DATE	DCHI VARCHAR(50)	GIOITINH VARCHAR(50)	MALOP CHAR(10)

--MySQL LIKE: kết hợp các ký tự đại diện (%) và (_)

```
SELECT * FROM SINHVIEN WHERE HOTEN LIKE '__ Văn%';
```

```

SELECT * FROM SINHVIEN WHERE HOTEN LIKE '_ _ %';
-- MySQL LIKE: kết hợp các ký tự đại diện (%) và (_)
SELECT * FROM SINHVIEN WHERE HOTEN LIKE '__ Văn%';
-- MySQL NOT LIKE Syntax
SELECT * FROM SINHVIEN WHERE HOTEN NOT LIKE '%Thị%';

```

MASV VARCHAR(10)	HOTEN VARCHAR(50)	NGSINH DATE	DCHI VARCHAR(50)	GIOITINH VARCHAR(50)	MALOP CHAR(10)
SV04	Lê Văn Khánh	7/4/1985	VUNG TAU	Nam	L002

Record 1 of 1

Text Data + ✓ Query executed successfully. 00:00:00.008 localhost (

-- MySQL NOT LIKE Syntax

```
SELECT * FROM SINHVIEN WHERE HOTEN NOT LIKE '%Thị%';
```

```

-- MySQL NOT LIKE Syntax
SELECT * FROM SINHVIEN WHERE HOTEN NOT LIKE '%Thị%';
-- MySQL LIKE: tìm kiếm không phân biệt chữ hoa chữ thường
SELECT * FROM SINHVIEN WHERE UPPER(HOTEN) LIKE UPPER('%LAN%');
-- MySQL UPDATE với LIKE

```

MASV VARCHAR(10)	HOTEN VARCHAR(50)	NGSINH DATE	DCHI VARCHAR(50)	GIOITINH VARCHAR(50)	MALOP CHAR(10)
SV02	Trần Thanh Tùng	4/2/1980	VUNG TAU	Nam	L001
SV01	Nguyễn Văn ANH	(null)	(null)	(null)	L001
SV04	Lê Văn Khánh	7/4/1985	VUNG TAU	Nam	L002
SV05	Ngô Việt Nam	(null)	DA NANG	Nam	L002
SV07	Trần Thành Nam	4/2/1988	DONG NAI	Nam	L004

Record 1 of 12

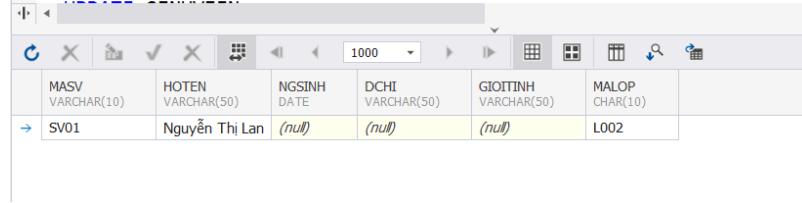
Text Data + ✓ Query executed successfully. 00:00:00.041 localhost (8.4) root

-- MySQL LIKE: tìm kiếm không phân biệt chữ hoa chữ thường

```
SELECT * FROM SINHVIEN WHERE UPPER(HOTEN) LIKE UPPER('%LAN%');
```

```
-- MySQL LIKE: tìm kiếm không phân biệt chữ hoa chữ thường
SELECT * FROM SINHVIEN WHERE UPPER(HOTEN) LIKE UPPER('%LAN%');

-- MySQL UPDATE với LIKE
```



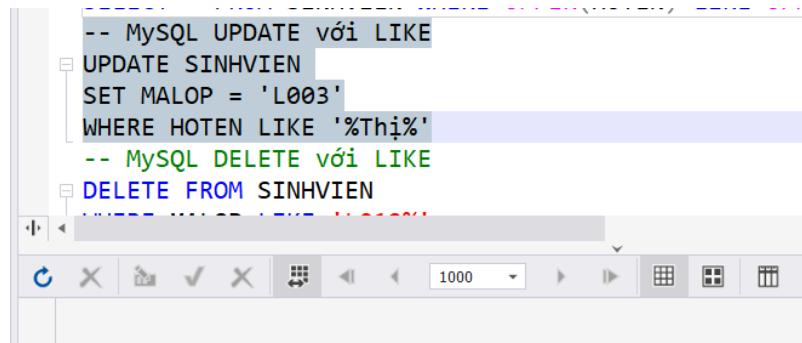
MASV VARCHAR(10)	HOTEN VARCHAR(50)	NGSINH DATE	DCHI VARCHAR(50)	GIOITINH VARCHAR(50)	MALOP CHAR(10)
SV01	Nguyễn Thị Lan	(null)	(null)	(null)	L002

-- MySQL UPDATE với LIKE

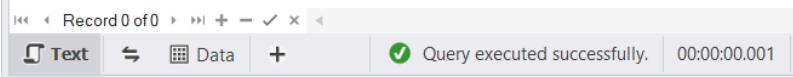
UPDATE SINHVIEN

SET MALOP = 'L003'

WHERE HOTEN LIKE '%Thị%'



MASV	HOTEN	NGSINH	DCHI	GIOITINH	MALOP
SV01	Nguyễn Thị Lan	(null)	(null)	(null)	L003



Record 0 of 0

Text Data + ✓ 00:00:00.001

Query executed successfully.

-- MySQL DELETE với LIKE

DELETE FROM SINHVIEN

WHERE MALOP LIKE 'L012%';

```
-- MySQL DELETE với LIKE
DELETE FROM SINHVIEN
WHERE MALOP LIKE 'L012%';
```

Text Data + ✓ Query executed successfully. 00:00:00.002

-- Cú pháp của MySQL ROUND (làm tròn)

-- Làm tròn khi số thập phân không được chỉ định

```
SELECT ROUND(15.678) AS rounded_number;
```

```
-- Cú pháp của MySQL ROUND (làm tròn)
-- Làm tròn khi số thập phân không được chỉ định
SELECT ROUND(15.678) AS rounded_number;
-- Làm tròn khi số thập phân là số dương
SELECT ROUND(10.345, 2) AS rounded_number;
```



rounded_number	DECIMAL(3, 0)
16	



Record 1 of 1 ✓ 00:00:00.009

Text Data + ✓ Query executed successfully. 00:00:00.009

-- Làm tròn khi số thập phân là số dương
SELECT ROUND(10.345, 2) AS rounded_number;

```
SELECT ROUND(15.678) AS rounded_number;
-- Làm tròn khi số thập phân là số dương
SELECT ROUND(10.345, 2) AS rounded_number;
```



rounded_number	DECIMAL(5, 2)
10.35	



Record 1 of 1 ✓ 00:00:00.000

Text Data + ✓ Query executed successfully. 00:00:00.000

-- Làm tròn khi số thập phân là số âm
SELECT ROUND(25.23, -1) AS rounded_number;

```
SELECT ROUND(10.575, 2) AS rounded_number;
-- Làm tròn khi số thập phân là số âm
SELECT ROUND(25.23, -1) AS rounded_number;
-- Làm tròn đến 0 chữ số thập phân
SELECT ROUND(7.9, 0) AS rounded_number;
-- Làm tròn với hàm AVG
```

Output	
rounded_number	DECIMAL(3, 0)
→	30

Output	
rounded_number	DECIMAL(3, 0)
→	30

-- Làm tròn đến 0 chữ số thập phân

```
SELECT ROUND(7.9, 0) AS rounded_number;
```

Output	
rounded_number	DECIMAL(2, 0)
→	8

Output	
rounded_number	DECIMAL(2, 0)
→	8

-- Làm tròn với hàm AVG

```
SELECT ROUND(AVG(DIEM),2 ) AS DiemTrungBinh
FROM SINHVIEN,DIEMTB;
```

```
-- Làm tròn với hàm AVG
SELECT ROUND(AVG(DIEM),2) AS DiemTrungBinh
FROM SINHVIEN,DIEMTB;
-- Làm tròn bằng hàm SUM
SELECT ROUND(SUM(DIEM), 2) AS TongDiem
FROM SINHVIEN,DIEMTB;
```

	DiemTrungBinh
→	7.26

Record 1 of 1

Text Data + ✓ Query executed successfully. 00:00:00.

-- Làm tròn bằng hàm SUM

```
SELECT ROUND(SUM(DIEM), 2) AS TongDiem
FROM SINHVIEN,DIEMTB;
```

```
-- Làm tròn bằng hàm SUM
SELECT ROUND(SUM(DIEM), 2) AS TongDiem
FROM SINHVIEN,DIEMTB;
```

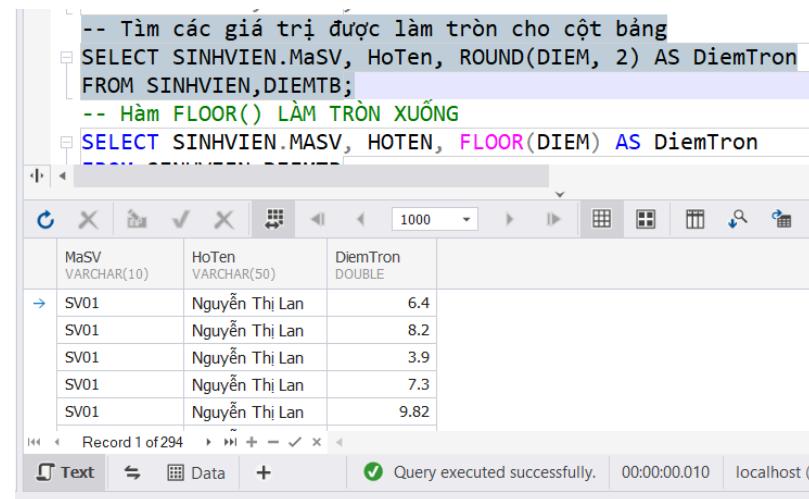
	TongDiem
→	2135.28

Record 1 of 1

Text Data + ✓ Query executed successfully.

-- Tìm các giá trị được làm tròn cho cột bảng

```
SELECT SINHVIEN.MaSV, HoTen, ROUND(DIEM, 2) AS DiemTron
FROM SINHVIEN,DIEMTB;
```



-- Tìm các giá trị được làm tròn cho cột bảng

```
SELECT SINHVIEN.MaSV, HoTen, ROUND(DIEM, 2) AS DiemTron
FROM SINHVIEN,DIEMTB;
-- Hàm FLOOR() LÀM TRÒN XUỐNG
```

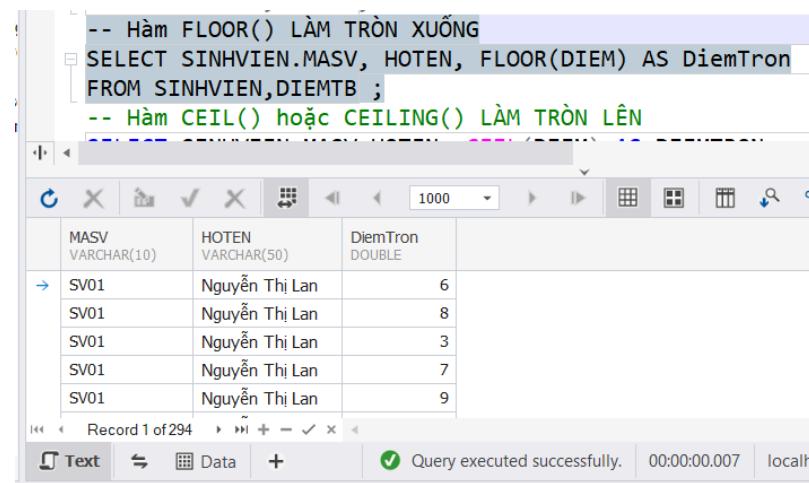
MaSV	HoTen	DiemTron
SV01	Nguyễn Thị Lan	6.4
SV01	Nguyễn Thị Lan	8.2
SV01	Nguyễn Thị Lan	3.9
SV01	Nguyễn Thị Lan	7.3
SV01	Nguyễn Thị Lan	9.82

Record 1 of 294

Text Data + ✓ Query executed successfully. 00:00:00.010 localhost (8)

-- Hàm FLOOR() LÀM TRÒN XUỐNG

```
SELECT SINHVIEN.MASV, HOTEN, FLOOR(DIEM) AS DiemTron
FROM SINHVIEN,DIEMTB ;
```



-- Hàm FLOOR() LÀM TRÒN XUỐNG

```
SELECT SINHVIEN.MASV, HOTEN, FLOOR(DIEM) AS DiemTron
FROM SINHVIEN,DIEMTB ;
-- Hàm CEIL() hoặc CEILING() LÀM TRÒN LÊN
```

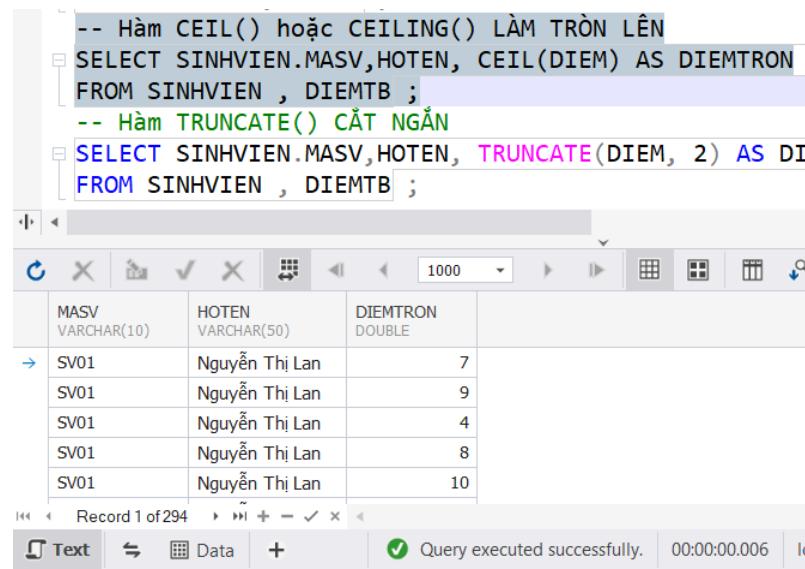
MASV	HOTEN	DiemTron
SV01	Nguyễn Thị Lan	6
SV01	Nguyễn Thị Lan	8
SV01	Nguyễn Thị Lan	3
SV01	Nguyễn Thị Lan	7
SV01	Nguyễn Thị Lan	9

Record 1 of 294

Text Data + ✓ Query executed successfully. 00:00:00.007 localhost

-- Hàm CEIL() hoặc CEILING() LÀM TRÒN LÊN

```
SELECT SINHVIEN.MASV,HOTEN, CEIL(DIEM) AS DIEMTRON
FROM SINHVIEN , DIEMTB ;
```



-- Hàm CEIL() hoặc CEILING() LÀM TRÒN LÊN

```
SELECT SINHVIEN.MASV,HOTEN, CEIL(DIEM) AS DIEMTRON
FROM SINHVIEN , DIEMTB ;
```

-- Hàm TRUNCATE() CẮT NGẮN

```
SELECT SINHVIEN.MASV,HOTEN, TRUNCATE(DIEM, 2) AS DIEMCAT
FROM SINHVIEN , DIEMTB ;
```

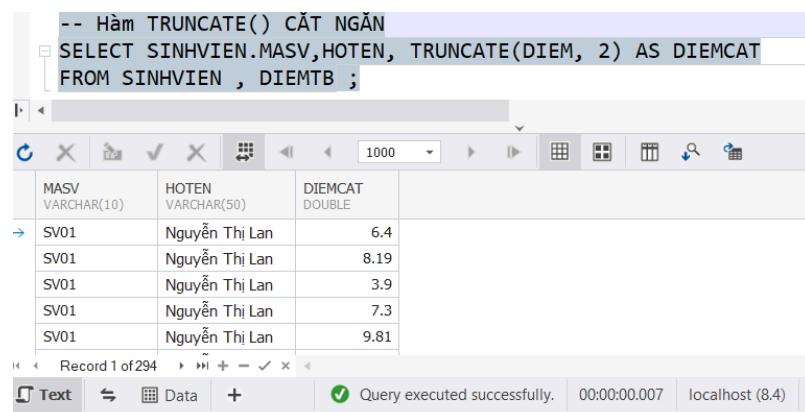
MASV VARCHAR(10)	HOTEN VARCHAR(50)	DIEMTRON DOUBLE
SV01	Nguyễn Thị Lan	7
SV01	Nguyễn Thị Lan	9
SV01	Nguyễn Thị Lan	4
SV01	Nguyễn Thị Lan	8
SV01	Nguyễn Thị Lan	10

Record 1 of 294

Text Data + ✓ Query executed successfully. 00:00:00.006

-- Hàm TRUNCATE() CẮT NGẮN

```
SELECT SINHVIEN.MASV,HOTEN, TRUNCATE(DIEM, 2) AS DIEMCAT
FROM SINHVIEN , DIEMTB ;
```



-- Hàm TRUNCATE() CẮT NGẮN

```
SELECT SINHVIEN.MASV,HOTEN, TRUNCATE(DIEM, 2) AS DIEMCAT
FROM SINHVIEN , DIEMTB ;
```

MASV VARCHAR(10)	HOTEN VARCHAR(50)	DIEMCAT DOUBLE
SV01	Nguyễn Thị Lan	6.4
SV01	Nguyễn Thị Lan	8.19
SV01	Nguyễn Thị Lan	3.9
SV01	Nguyễn Thị Lan	7.3
SV01	Nguyễn Thị Lan	9.81

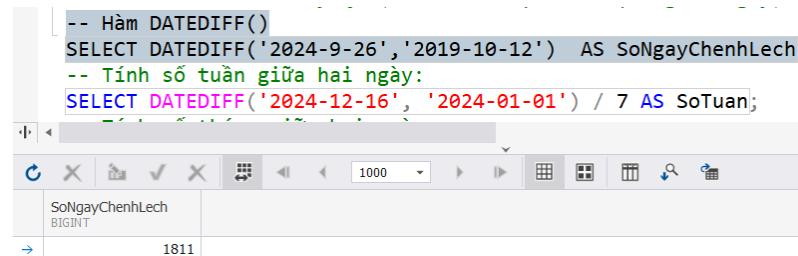
Record 1 of 294

Text Data + ✓ Query executed successfully. 00:00:00.007 localhost (8.4)

-- Hàm DATEDIFF của MySQL (tính toán sự trễ trên lịch giữa ngày)

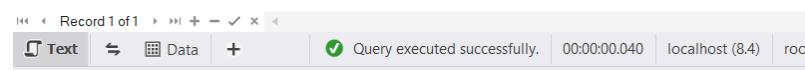
-- Hàm DATEDIFF()

```
SELECT DATEDIFF('2024-9-26','2019-10-12') AS SoNgayChenhLech;
```



```
-- Hàm DATEDIFF()
SELECT DATEDIFF('2024-9-26','2019-10-12') AS SoNgayChenhLech;
-- Tính số tuần giữa hai ngày:
SELECT DATEDIFF('2024-12-16', '2024-01-01') / 7 AS SoTuan;
```

SoNgayChenhLech	BIGINT
1811	

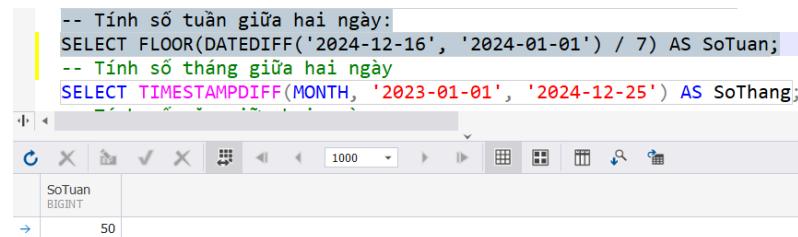


```
Record 1 of 1 ✓ 00:00:00.040 localhost (8.4) root
```

```
-- Tính số tuần giữa hai ngày:
SELECT FLOOR(DATEDIFF('2024-12-16', '2024-01-01') / 7) AS SoTuan;
```

-- Tính số tuần giữa hai ngày:

```
SELECT FLOOR(DATEDIFF('2024-12-16', '2024-01-01') / 7) AS SoTuan;
```



```
-- Tính số tháng giữa hai ngày
SELECT FLOOR(DATEDIFF('2024-12-16', '2024-01-01') / 7) AS SoTuan;
-- Tính số tháng giữa hai ngày
SELECT TIMESTAMPDIFF(MONTH, '2023-01-01', '2024-12-25') AS SoThang;
```

SoTuan	BIGINT
50	



```
Record 1 of 1 ✓ 00:00:00.005 localhost (8.4) root dssinhv
```

-- Tính số tháng giữa hai ngày

```
SELECT TIMESTAMPDIFF(MONTH, '2023-01-01', '2024-12-25') AS SoThang;
```

```
SELECT FLOOR(DATEDIFF('2024-12-10', '2024-01-01')) / 30 AS SoThang;
-- Tính số tháng giữa hai ngày
SELECT TIMESTAMPDIFF(MONTH, '2023-01-01', '2024-12-25') AS SoThang;
```

SoThang	BIGINT
23	

Record 1 of 1 ✓ 00:00:00.009 localhost (8.4) root dssinl

Text Data + ✓ Query executed successfully. 00:00:00.009 localhost (8.4) root dssinl

-- Tính số năm giữa hai ngày

```
SELECT TIMESTAMPDIFF(YEAR, '2019-10-12', '2024-9-26') AS SoNam;
```

```
-- Tính số năm giữa hai ngày
SELECT TIMESTAMPDIFF(YEAR, '2019-10-12', '2024-9-26') AS SoNam;
-- Tính toán thời lượng của một sự kiện theo giờ và phút
SELECT
    HOUR(TIMEDIFF('2024-09-25 16:45', '2024-09-25 14:30')) AS SoG:
```

SoNam	BIGINT
4	

Record 1 of 1 ✓ 00:00:00.004 localhost (8.4) root dssinl

Text Data + ✓ Query executed successfully. 00:00:00.004 localhost (8.4) root dssinl

-- Tính toán thời lượng của một sự kiện theo giờ và phút

```
SELECT
    HOUR(TIMEDIFF('2024-09-25 16:45', '2024-09-25 14:30')) AS SoGio,
    MINUTE(TIMEDIFF('2024-09-25 16:45', '2024-09-25 14:30')) AS SoPhut;
```

```
-- Tính toán thời lượng của một sự kiện theo giờ và phút
SELECT
    HOUR(TIMEDIFF('2024-09-25 16:45', '2024-09-25 14:30')) AS SoGio,
    MINUTE(TIMEDIFF('2024-09-25 16:45', '2024-09-25 14:30')) AS SoPhut;
-- Sử dụng hàm DATEDIFF() và loại bỏ ngày cuối tuần
SELECT
    DATEDIFF('2024-09-30', '2024-09-01') + 1 AS TongSoNgay,
    (DATEDIFF('2024-09-30', '2024-09-01') + 1)
```

SoGio	SoPhut
2	15

Record 1 of 1

Text Data + ✓ Query executed successfully. 00:00:00.027 localhost (8.4) root dssinhvien

-- Sử dụng hàm DATEDIFF() và loại bỏ ngày cuối tuần

```
SELECT
    DATEDIFF('2024-09-30', '2024-09-01') + 1 AS TongSoNgay,
    (DATEDIFF('2024-09-30', '2024-09-01') + 1)
    - (FLOOR(DATEDIFF('2024-09-30', '2024-09-01') / 7) * 2)
    - (CASE WHEN WEEKDAY('2024-09-01') = 6 THEN 1 ELSE 0 END)
    - (CASE WHEN WEEKDAY('2024-09-30') = 6 THEN 1 ELSE 0 END)
AS SoNgayLamViec;
```

```
-- Sử dụng hàm DATEDIFF() và loại bỏ ngày cuối tuần
SELECT
    DATEDIFF('2024-09-30', '2024-09-01') + 1 AS TongSoNgay,
    (DATEDIFF('2024-09-30', '2024-09-01') + 1)
    - (FLOOR(DATEDIFF('2024-09-30', '2024-09-01') / 7) * 2)
    - (CASE WHEN WEEKDAY('2024-09-01') = 6 THEN 1 ELSE 0 END)
    - (CASE WHEN WEEKDAY('2024-09-30') = 6 THEN 1 ELSE 0 END)
    AS SoNgayLamViec;
```

-- Tính số ngày cho đến khi có sự kiện trong tương lai

TongSoNgay	SoNgayLamViec
30	21

←	Record 1 of 1	→	1000	◀	▶	✚	✖
Text	Data	+	Query executed successfully.	00:00:00.042	localhost (8.4)	root	

-- Tính số ngày cho đến khi có sự kiện trong tương lai

```
SELECT DATEDIFF('2024-10-12', CURRENT_DATE) AS SoNgayDenSuKien;
```

-- Tính số ngày cho đến khi có sự kiện trong tương lai	
SELECT DATEDIFF('2024-10-12', CURRENT_DATE) AS SoNgayDenSuKien;	
SoNgayDenSuKien	BIGINT
→	10

←	Record 1 of 1	→	1000	◀	▶	✚	✖
Text	Data	+	Query executed successfully.	00:00:00.011	localhost (8.4)	root	d

-- Hàm CONCAT của MySQL (nối các chuỗi)

-- MySQL CONCAT với SELECT

```
SELECT CONCAT('My','SQL') AS ketqua;
```

```
-- MySQL CONCAT với SELECT
SELECT CONCAT('My', 'SQL') AS ketqua;
-- CONCAT() with NULL values
SELECT CONCAT('Student: ', NULL, ' A') AS result;
```


ketqua	VARCHAR(5)
MySQL	

Record 1 of 1

Text Data + ✓ Query executed successfully. 00:00:00.004

-- CONCAT() with NULL values

```
SELECT CONCAT('Student: ', NULL, ' A') AS result;
```

```
-- CONCAT() Với FROM
SELECT CONCAT('Ho', 'Ten') AS HoTen FROM SINHVIEN;
-- Cách sử dụng CONCAT với WHERE
SELECT * FROM SINHVIEN
```


HoTen	VARCHAR(5)
HoTen	

Record 1 of 21

Text Data + ✓ Query executed successfully. 00:00:00.007

-- CONCAT() VỚI FROM

```
SELECT CONCAT('Ho', 'Ten') AS HoTen FROM SINHVIEN;
```

```
SELECT * FROM SINHVIEN
WHERE CONCAT('Ho', 'Ten') = N'Nguyễn Thị Lan';
-- MySQL CONCAT so với CONCAT_WS
-- CONCAT(string1, string2, ..., stringN), NULL = RỖNG
```

MASV VARCHAR(10)	HOTEN VARCHAR(50)	NGSINH DATE	DCHI VARCHAR(50)	GIOTINH VARCHAR(50)	MALOP CHAR(10)	

Record 0 of 0

-- Cách sử dụng CONCAT với WHERE

```
SELECT * FROM SINHVIEN
```

```
WHERE CONCAT('Ho','Ten') = 'Nguyen Thị Lan';
```

```
SELECT * FROM SINHVIEN
WHERE CONCAT('Ho', 'Ten') = N'Nguyễn Thị Lan';
-- MySQL CONCAT so với CONCAT_WS
-- CONCAT(string1, string2, ..., stringN), NULL = RỖNG
```

MASV VARCHAR(10)	HOTEN VARCHAR(50)	NGSINH DATE	DCHI VARCHAR(50)	GIOTINH VARCHAR(50)	MALOP CHAR(10)	

Record 0 of 0

-- MySQL CONCAT so với CONCAT_WS

-- CONCAT(string1, string2, ..., stringN), NULL = RỖNG

-- CONCAT_WS(giải phân cách, string1, string2, ..., stringN), BỎ QUA NULL

```
SELECT CONCAT_WS(' ', 'ĐÀO', NULL, 'DƯA', 'CHUỐI') AS KETQUA;
```

```
-- MySQL CONCAT so với CONCAT_WS
-- CONCAT(string1, string2, ..., stringN), NULL = RỖNG
-- CONCAT_WS(phân cách, string1, string2, ..., stringN), BỎ QUA NULL
SELECT CONCAT_WS(' ', 'ĐÀO', NULL, 'DƯA', 'CHUỐI') AS KETQUA;
```

KETQUA	VARCHAR(17)
ĐÀO, DƯA, CHUỐI	

Record 1 of 1

Text Data + ✓ Query executed successfully. 00:00:00.011 localhost (8.4) root dssinhvien

-- Hàm COUNT của MySQL (ĐÉM DỮ LIỆU)

-- Liệt kê danh sách tên khoa và số lượng lớp trong từng khoa.

```
SELECT TENKHOA, COUNT(*) AS số_lượng_lớp
FROM KHOAHOC,LOP
WHERE KHOAHOC.MAKHOA = LOP.MAKHOA GROUP BY TENKHOA
```

```
-- Hàm COUNT của MySQL (ĐÉM DỮ LIỆU)
-- Liệt kê danh sách tên khoa và số lượng lớp trong từng khoa.
SELECT TENKHOA, COUNT(*) AS số_lượng_lớp
FROM KHOAHOC,LOP
WHERE KHOAHOC.MAKHOA = LOP.MAKHOA GROUP BY TENKHOA
-- COUNT(biểu thức DISTINCT)
SELECT COUNT(DISTINCT TENKHOA) AS SoKhoaKhac FROM SINHVIEN,KHOAHOC
```

TENKHOA	số_lượng_lớp
DIEN TU	1
HOA HOC	1
SINH HOC	1
TIN HOC	5
THUC PHAM	3
VAT LY	1

Record 1 of 6

-- COUNT(biểu thức DISTINCT)

```
SELECT COUNT(DISTINCT TENKHOA) AS SoKhoaKhac FROM SINHVIEN,KHOAHOC;
```

```
-- COUNT(biểu thức DISTINCT)
SELECT COUNT(DISTINCT TENKHOA) AS SoKhoaKhac FROM SINHVIEN,KHOAHOC;
-- Sử dụng COUNT với ví dụ mệnh đề WHERE
-- Liệt kê danh sách tên khoa và số lượng sinh viên trong từng khoa.
```

SoKhoaKhac	BIGINT
6	

-- Sử dụng COUNT với ví dụ mệnh đề WHERE

-- Liệt kê danh sách tên khoa và số lượng sinh viên trong từng khoa.

```
SELECT TENKHOA, COUNT(*) AS SOLUONGSINHVIEN
FROM KHOAHOC,SINHVIEN,LOP
WHERE KHOAHOC.MAKHOA = LOP.MAKHOA AND LOP.MALOP = SINHVIEN.MALOP
GROUP BY TENKHOA
```

```
-- Sử dụng COUNT với ví dụ mệnh đề WHERE
-- Liệt kê danh sách tên khoa và số lượng sinh viên trong từng khoa.
SELECT TENKHOA, COUNT(*) AS SOLUONGSINHVIEN
FROM KHOAHOC,SINHVIEN,LOP
WHERE KHOAHOC.MAKHOA = LOP.MAKHOA AND LOP.MALOP = SINHVIEN.MALOP GROUP BY TENKHOA
-- sử dụng COUNT với GROUP BY
-- Liệt kê danh sách tên lớp và số lượng sinh viên trong từng lớp.
```

TENKHOA	SOLUONGSINHVIEN
HOA HOC	1
TIN HOC	16
THUC PHAM	4

-- sử dụng COUNT với GROUP BY

-- Liệt kê danh sách tên lớp và số lượng sinh viên trong từng lớp.

```
SELECT TENLOP, COUNT(*) AS SOLUONGSINHVIEN
FROM LOP,SINHVIEN
WHERE LOP.MALOP=SINHVIEN.MALOP GROUP BY TENLOP
```

```
-- sử dụng COUNT với GROUP BY
-- Liệt kê danh sách tên lớp và số lượng sinh viên trong từng lớp.
SELECT TENLOP, COUNT(*) AS SOLUONGSINHVIEN
FROM LOP,SINHVIEN
WHERE LOP.MALOP=SINHVIEN.MALOP GROUP BY TENLOP
-- sử dụng COUNT với HAVING
-- Cho biết tên những khoa có ít nhất 3 lớp.
```

TENLOP	SOLUONGSINHVIEN
24HTTH1	2
24HTTH2	2
05CDTH	9
06CDTH1	2
06CDTH2	1
24HTPT1	1
	Record 1 of 9

-- sử dụng COUNT với HAVING

-- Cho biết tên những khoa có ít nhất 3 lớp.

```
SELECT KHOAHOC.MAKHOA,TENKHOA, COUNT()
```

```
FROM KHOAHOC,LOP
```

```
WHERE KHOAHOC.MAKHOA = LOP.MAKHOA GROUP BY KHOAHOC.MAKHOA,TENKHOA
```

```
HAVING COUNT() >= 3
```

```
SELECT KHOAHOC.MAKHOA,TENKHOA, COUNT(*)
FROM KHOAHOC,LOP
WHERE KHOAHOC.MAKHOA = LOP.MAKHOA GROUP BY KHOAHOC.MAKHOA,TENKHOA
HAVING COUNT(*) >= 3
```

MAKHOA	TENKHOA	COUNT(*)
TH	TIN HOC	5
TP	THUC PHAM	3

-- Hàm LEAD và LAG của MySQL (truy cập các giá trị của hàng phía sau và hàng trước trong một bảng)

-- Hàm LEAD

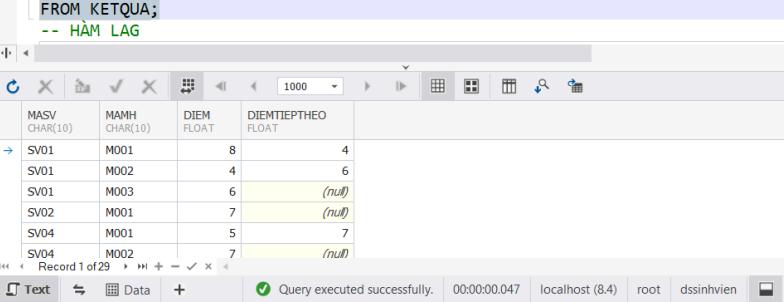
-- Lấy điểm số và điểm số của lần kiểm tra tiếp theo

SELECT MASV,MAMH,DIEM,

LEAD (DIEM) OVER (PARTITION BY MASV ORDER BY MAMH)AS DIEMTIEPTHEO

FROM KETQUA;

```
-- Hàm LEAD và LAG của MySQL (truy cập các giá trị của hàng phía sau và hàng
-- Hàm LEAD
-- Lấy điểm số và điểm số của lần kiểm tra tiếp theo
SELECT MASV,MAMH,DIEM,
LEAD (DIEM) OVER (PARTITION BY MASV ORDER BY MAMH)AS DIEMTIEPTHEO
FROM KETQUA;
-- HÀM LAG
```



MASV CHAR(10)	MAMH CHAR(10)	DIEM FLOAT	DIEMTIEPTHEO FLOAT
SV01	M001	8	4
SV01	M002	4	6
SV01	M003	6	(null)
SV02	M001	7	(null)
SV04	M001	5	7
SV04	M002	7	(null)

Record 1 of 29

Text Data + ✓ Query executed successfully. 00:00:00.047 localhost (8.4) root dssinhvien

-- HÀM LAG

SELECT MASV,MAMH,DIEM,

LAG(DIEM) OVER (PARTITION BY MASV ORDER BY MAMH) AS DiemTruoc

FROM KETQUA;

```
-- HÀM LAG
SELECT MASV,MAMH,DIEM,
LAG(DIEM) OVER (PARTITION BY MASV ORDER BY MAMH) AS DiemTruoc
FROM KETQUA;
-- FIRST_VALUE
SELECT MASV,MAMH,DIEM,
```

Table Data View

MASV CHAR(10)	MAMH CHAR(10)	DIEM FLOAT	DiemTruoc FLOAT
SV01	M001	8	(null)
SV01	M002	4	8
SV01	M003	6	4
SV02	M001	7	(null)
SV04	M001	5	(null)
SV04	M002	7	5

Record 1 of 29

Text Data + ✓ Query executed successfully. 00:00:00.009 localhost (8.4) root

-- FIRST_VALUE

```
SELECT MASV,MAMH,DIEM,
FIRST_VALUE(DIEM) OVER (PARTITION BY MASV ORDER BY MAMH) AS DIEMDAUTIEN
FROM KETQUA;
```

```
-- FIRST_VALUE
SELECT MASV,MAMH,DIEM,
FIRST_VALUE(DIEM) OVER (PARTITION BY MASV ORDER BY MAMH) AS DIEMDAUTIEN
FROM KETQUA;
-- LAST_VALUE
SELECT MASV,MAMH,DIEM,
LAST_VALUE(DIEM) OVER (PARTITION BY MASV ORDER BY MAMH) AS DIEMDAUTIEN
```

Table Data View

MASV CHAR(10)	MAMH CHAR(10)	DIEM FLOAT	DIEMDAUTIEN FLOAT
SV01	M001	8	8
SV01	M002	4	8
SV01	M003	6	8
SV02	M001	7	7
SV04	M001	5	5
SV04	M002	7	5

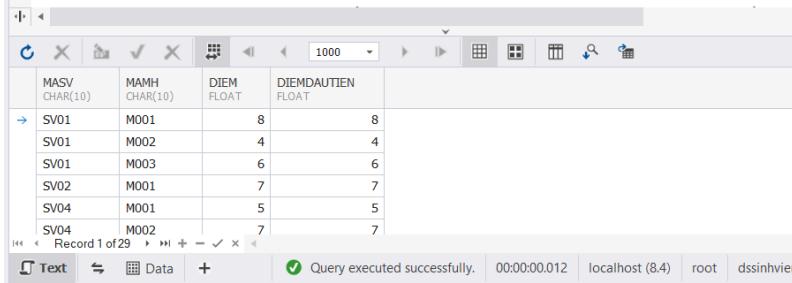
Record 1 of 29

Text Data + ✓ Query executed successfully. 00:00:00.010 localhost (8.4) root dssinhvien

-- LAST_VALUE

```
SELECT MASV,MAMH,DIEM,
LAST_VALUE(DIEM) OVER (PARTITION BY MASV ORDER BY MAMH) AS DIEMDAUTIEN
FROM KETQUA;
```

```
FROM KETQUA;
-- LAST_VALUE
SELECT MASV,MAMH,DIEM,
LAST_VALUE(DIEM) OVER (PARTITION BY MASV ORDER BY MAMH) AS DIEMDAUTIEN
FROM KETQUA;
```



The screenshot shows the MySQL Workbench interface. At the top, there is a code editor window with the query shown above. Below it is a results grid showing the output of the query. The results grid has four columns: MASV, MAMH, DIEM, and DIEMDAUTIEN. The data is as follows:

MASV	MAMH	DIEM	DIEMDAUTIEN
CHAR(10)	CHAR(10)	FLOAT	FLOAT
SV01	M001	8	8
SV01	M002	4	4
SV01	M003	6	6
SV02	M001	7	7
SV04	M001	5	5
SV04	M002	7	7

At the bottom of the results grid, it says "Record 1 of 29". Below the results grid is a toolbar with buttons for Text, Data, and a plus sign, followed by a message "Query executed successfully." and some status information: "00:00:00.012", "localhost (8.4)", "root", and "dssinhvien".

-- MySQL Window Functions (Sử dụng phân tích nâng cao trong tập kết quả truy vấn)

-- Các hàm cửa sổ tổng hợp của MySQL

-- HÀM SUM()

SELECT MASV,MAMH,DIEM,

SUM(DIEM) OVER (PARTITION BY MASV) AS TongDiem

FROM KETQUA;

```
-- Các hàm cửa sổ tổng hợp của MySQL
-- HÀM SUM()
SELECT MASV, MAMH, DIEM,
SUM(DIEM) OVER (PARTITION BY MASV) AS TongDiem
FROM KETQUA;
-- HÀM AVG()
SELECT MASV, MAMH, DIEM,
```

MASV CHAR(10)	MAMH CHAR(10)	DIEM FLOAT	TongDiem DOUBLE
SV01	M001	8	18
SV01	M002	4	18
SV01	M003	6	18
SV02	M001	7	7
SV04	M001	5	12
SV04	M002	7	12

Record 1 of 29

Text Data + ✓ Query executed successfully. 00:00:00.006

-- HÀM AVG()

```
SELECT MASV, MAMH, DIEM,
AVG(DIEM)OVER(PARTITION BY MASV)AS DIEMTRUNGBINH
FROM KETQUA;
```

```
-- HÀM AVG()
SELECT MASV, MAMH, DIEM,
AVG(DIEM)OVER(PARTITION BY MASV)AS DIEMTRUNGBINH
FROM KETQUA;
-- Hàm MIN() và MAX()
SELECT MASV, MAMH, DIEM,
MIN(DIEM) OVER (PARTITION BY MASV) AS DiemThapNhat
```

MASV CHAR(10)	MAMH CHAR(10)	DIEM FLOAT	DIEMTRUNGBINH DOUBLE
SV01	M001	8	6
SV01	M002	4	6
SV01	M003	6	6
SV02	M001	7	7
SV04	M001	5	6
SV04	M002	7	6

Record 1 of 29

Text Data + ✓ Query executed successfully. 00:00:00.016

-- Hàm MIN() và MAX()

```
SELECT MASV, MAMH, DIEM,
MIN(DIEM) OVER (PARTITION BY MASV) AS DiemThapNhat,
MAX(DIEM) OVER (PARTITION BY MASV) AS DiemCaoNhat
FROM KETQUA;
```

```
-- Hàm MIN() và MAX()
SELECT MASV, MAMH, DIEM,
MIN(DIEM) OVER (PARTITION BY MASV) AS DiemThapNhat,
MAX(DIEM) OVER (PARTITION BY MASV) AS DiemCaoNhat
FROM KETQUA;
-- Hàm COUNT()
SELECT MASV, MAMH, DIEM,
COUNT(DIEM)OVER(PARTITION BY MASV)AS DIEMTRUNGBINH
```

Text Data + ✓ Query executed successfully. 00:00:00.006 localh

-- Hàm COUNT()

```
SELECT MASV, MAMH, DIEM,
COUNT(DIEM)OVER(PARTITION BY MASV)AS DIEMTRUNGBINH
FROM KETQUA;
```

```
-- Hàm COUNT()
SELECT MASV, MAMH, DIEM,
COUNT(DIEM)OVER(PARTITION BY MASV)AS DIEMTRUNGBINH
FROM KETQUA;
-- ROW_NUMBER(ĐÁNH SỐ THỨ TỰ)
```

Text Data + ✓ Query executed successfully. 00:00:00.007 localh

```
-- ROW_NUMBER(ĐÁNH SỐ THỨ TỰ)
SELECT MASV,MAMH,DIEM,
ROW_NUMBER() OVER (PARTITION BY MASV ORDER BY MAMH) AS STT
FROM KETQUA;
```

```
-- ROW_NUMBER(ĐÁNH SỐ THỨ TỰ)
SELECT MASV,MAMH,DIEM,
ROW_NUMBER() OVER (PARTITION BY MASV ORDER BY MAMH) AS STT
FROM KETQUA;
-- Hàm RANK(XẾP THỨ HẠNG BỎ QUA SỐ TIẾP THEO NẾU TRÙNG LẶP)
SELECT MASV,MAMH,DIEM,
```

MASV CHAR(10)	MAMH CHAR(10)	DIEM FLOAT	STT UNSIGNED BIGINT
SV01	M001	8	1
SV01	M002	4	2
SV01	M003	6	3
SV02	M001	7	1
SV04	M001	5	1
SV04	M002	7	2

Record 1 of 29

Text Data + Query executed successfully. 00:00:00.004 localhost (8.4)

```
-- Hàm RANK(XẾP THỨ HẠNG BỎ QUA SỐ TIẾP THEO NẾU TRÙNG LẶP)
SELECT MASV,MAMH,DIEM,
RANK() OVER (PARTITION BY MAMH ORDER BY DIEM DESC) AS XepHang
FROM KETQUA;
```

```
-- Hàm RANK(XẾP THỨ HẠNG BỎ QUA SỐ TIẾP THEO NẾU TRÙNG LẶP)
SELECT MASV,MAMH,DIEM,
RANK() OVER (PARTITION BY MAMH ORDER BY DIEM DESC) AS XepHang
FROM KETQUA;
-- Hàm DENSE_RANK(XẾP THỨ HẠNG KHÔNG BỎ QUA GIÁ TRỊ NÀO )
SELECT MASV,MAMH,DIEM,
DENSE_RANK() OVER (PARTITION BY MAMH ORDER BY DIEM DESC) AS Xep

```

Query executed successfully. 00:00:00.007 localhost (8.4) root

-- Hàm DENSE_RANK(XẾP THỨ HẠNG KHÔNG BỎ QUA GIÁ TRỊ NÀO)

```
SELECT MASV,MAMH,DIEM,
DENSE_RANK() OVER (PARTITION BY MAMH ORDER BY DIEM DESC) AS XepHang
FROM KETQUA;
```

```
-- Hàm DENSE_RANK(XẾP THỨ HẠNG KHÔNG BỎ QUA GIÁ TRỊ NÀO )
SELECT MASV,MAMH,DIEM,
DENSE_RANK() OVER (PARTITION BY MAMH ORDER BY DIEM DESC) AS XepHang
FROM KETQUA;
```

Query executed successfully. 00:00:00.010 localhost (8.4) root dssinh

Tài liệu tham khảo

<https://blog.devart.com/mysql-tutorial.html>

<https://www.w3schools.com/MySQL/default.asp>

<https://chatgpt.com>