



# TÀI LIỆU HƯỚNG DẪN THỰC HÀNH MÔN HỌC IT004 - CƠ SỞ DỮ LIỆU (DATABASE)

## NỘI DUNG THỰC HÀNH TUẦN 1

*Hướng dẫn thực hành*

Lê Võ Đình Kha - [khalvd@uit.edu.vn](mailto:khalvd@uit.edu.vn)

# GIỚI THIỆU NỘI DUNG THỰC HÀNH TUẦN 1



## NỘI DUNG

1. Tổng quan về cơ sở dữ liệu.
2. Cài đặt SQL Server Management Studio.
3. Kiểu dữ liệu trong ngôn ngữ truy vấn SQL.
4. Các lệnh về Database, Table, Ràng buộc toàn vẹn.
5. Các lệnh về thao tác dữ liệu.
6. Bài tập thực hành và hỏi đáp.

# **GIỚI THIỆU NỘI DUNG THỰC HÀNH TUẦN 1**

**1**

## **TỔNG QUAN VỀ CƠ SỞ DỮ LIỆU**

# TỔNG QUAN VỀ CƠ SỞ DỮ LIỆU

## Giới thiệu về cơ sở dữ liệu

- **Cơ sở dữ liệu (CSDL)** là tập hợp các dữ liệu được tổ chức và lưu trữ theo cách có cấu trúc, thường dưới dạng bảng, để dễ dàng quản lý, truy xuất, và xử lý.
- **Tầm quan trọng:** Trong thời đại số hóa, mọi lĩnh vực như ngân hàng, y tế, giáo dục, và thương mại đều phụ thuộc vào CSDL để vận hành hiệu quả.
- **Ứng dụng:** Từ giao dịch tài chính, quản lý khách hàng, đến hệ thống thông tin doanh nghiệp, CSDL và hệ quản trị cơ sở dữ liệu (DBMS) đảm bảo tính toàn vẹn và khả năng truy xuất dữ liệu nhanh chóng.



# TỔNG QUAN VỀ CƠ SỞ DỮ LIỆU

## Ví dụ về dữ liệu

- **Giao dịch ngân hàng:** Lưu trữ và theo dõi lịch sử giao dịch, số dư tài khoản.
- **Đặt chỗ khách sạn:** Quản lý thông tin đặt phòng, tình trạng phòng trống.
- **Đặt chuyến bay:** Quản lý thông tin hành khách, lịch trình bay.
- **Thư viện:** Tìm kiếm và quản lý sách, trạng thái mượn trả.

## Ứng dụng của hệ thống cơ sở dữ liệu

- **CSDL đa phương tiện:** Lưu trữ và quản lý nội dung số (YouTube, Spotify).
- **Hệ thống thông tin địa lý (GIS):** Quản lý dữ liệu không gian (Google Maps, GPS).
- **Kho dữ liệu (Data warehouse):** Tích hợp và phân tích dữ liệu lớn (Amazon, Walmart).

# TỔNG QUAN VỀ CƠ SỞ DỮ LIỆU

## Hệ quản trị cơ sở dữ liệu (DBMS - Database Management System)

- Là phần mềm chuyên dụng cho việc tạo lập, quản lý, và vận hành cơ sở dữ liệu.
- Cho phép người dùng truy cập, điều khiển và quản lý dữ liệu một cách hiệu quả.



## Ví dụ về hệ quản trị cơ sở dữ liệu

- **SQL Server:** DBMS phổ biến của Microsoft, hỗ trợ các doanh nghiệp quản lý dữ liệu lớn.
- **MySQL:** DBMS mã nguồn mở, thường được sử dụng cho các ứng dụng web.
- **Oracle:** DBMS mạnh mẽ, hỗ trợ quản lý cơ sở dữ liệu phức tạp trong các tổ chức lớn

# TỔNG QUAN VỀ CƠ SỞ DỮ LIỆU

— Một số hệ quản trị cơ sở dữ liệu phổ biến



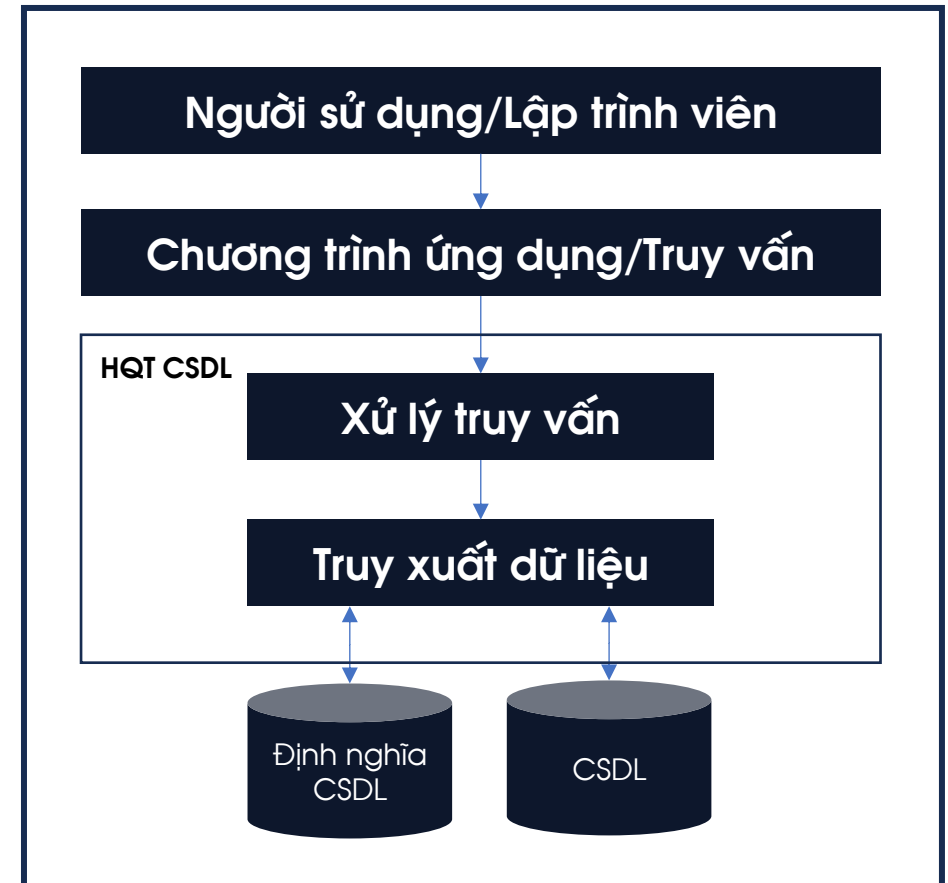
Các Hệ quản trị Cơ sở dữ liệu (Database Management System) phổ biến

Xem thêm tại: <https://db-engines.com/en/ranking>

# TỔNG QUAN VỀ CƠ SỞ DỮ LIỆU

## Hệ cơ sở dữ liệu

- Là sự tích hợp giữa Cơ sở dữ liệu (CSDL) và Hệ quản trị cơ sở dữ liệu (DBMS).
- **Cấu thành:**
  - Cơ sở dữ liệu: Là tập hợp các dữ liệu có cấu trúc, được tổ chức và lưu trữ một cách hệ thống.
  - Hệ quản trị cơ sở dữ liệu: Là phần mềm quản lý, truy cập và xử lý cơ sở dữ liệu.





## 2

### CÀI ĐẶT SQL SERVER MANAGEMENT STUDIO

# CÀI ĐẶT SQL SERVER MANAGEMENT STUDIO

## Giới thiệu Microsoft SQL Server

- SQL Server là một hệ quản trị cơ sở dữ liệu quan hệ (RDBMS) do Microsoft phát triển.
- Được sử dụng rộng rãi trong các tổ chức và doanh nghiệp để lưu trữ, quản lý và truy xuất dữ liệu.
- Hỗ trợ nhiều tính năng hiện đại như phân tích dữ liệu, tích hợp AI, và xử lý dữ liệu lớn (Big Data).



# CÀI ĐẶT SQL SERVER MANAGEMENT STUDIO

## Giới thiệu Microsoft SQL Server (tt)

Phiên Bản	Năm ra mắt	Tính năng đặc trưng
<b>SQL Server 1.0</b>	1989	Phiên bản đầu tiên, cung cấp các tính năng cơ bản về quản lý CSDL.
<b>SQL Server 7.0</b>	1998	Cải tiến giao diện đồ họa, hỗ trợ nhiều tính năng mới.
<b>SQL Server 2000 (ver 8)</b>	2000	Tăng cường hiệu suất, bảo mật, hỗ trợ XML.
<b>SQL Server 2019 (ver 15)</b>	2019	Hỗ trợ Big Data, tích hợp AI, bảo mật cao cấp.

# CÀI ĐẶT SQL SERVER MANAGEMENT STUDIO

## Giới thiệu Microsoft SQL Server (tt)

Ấn Bản	Mô tả
<b>Enterprise</b>	Phiên bản cao cấp với đầy đủ tính năng cho các tổ chức lớn.
<b>Developer</b>	Phiên bản dành cho các nhà phát triển, đầy đủ tính năng như Enterprise.
<b>Standard</b>	Phiên bản cơ bản cho các doanh nghiệp vừa và nhỏ.
<b>Web</b>	Phiên bản tối ưu hóa cho các ứng dụng web, chi phí thấp.
<b>Express</b>	Phiên bản miễn phí, giới hạn tính năng, phù hợp cho học tập và các ứng dụng nhỏ.

# CÀI ĐẶT SQL SERVER MANAGEMENT STUDIO

## Hướng dẫn tải SQL Server Developer:

- Truy cập vào đường dẫn:

[https://portal.azure.com/#view/Microsoft\\_Azure\\_Education/EducationMenuBlade/~/software](https://portal.azure.com/#view/Microsoft_Azure_Education/EducationMenuBlade/~/software)

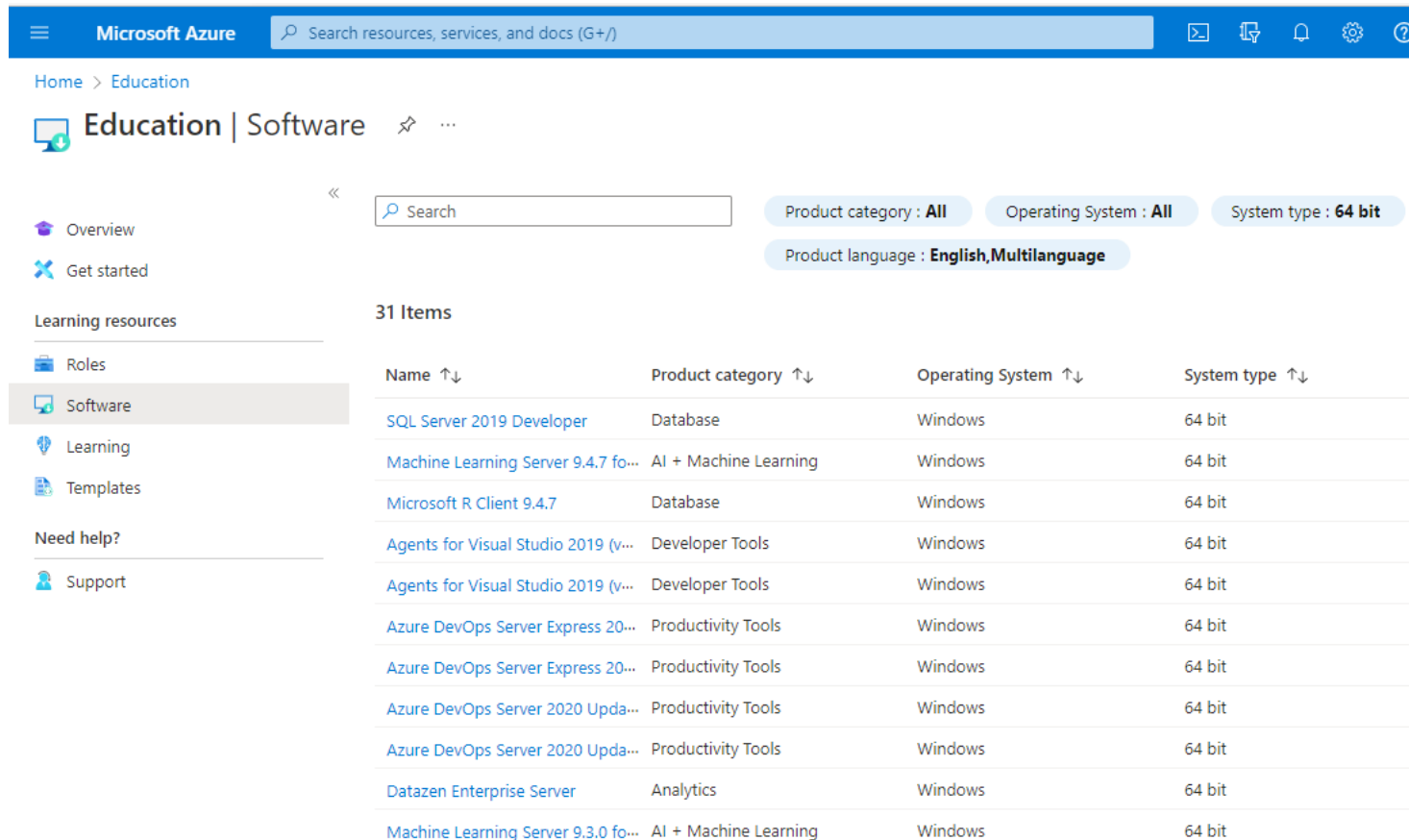
- Đăng nhập bằng tài khoản Microsoft do trường cấp (@ms.uit.edu.vn).
- Tìm và tải xuống phiên bản SQL Server 2019 (hoặc 2022) Developer từ cổng thông tin Azure.

## Cài đặt SQL Server 2019 Developer:

- Sau khi tải xuống, mở tệp cài đặt và làm theo các bước hướng dẫn.

# CÀI ĐẶT SQL SERVER MANAGEMENT STUDIO

## Cài đặt SQL Server 2019 Developer: (Có thể cài phiên bản năm 2022)



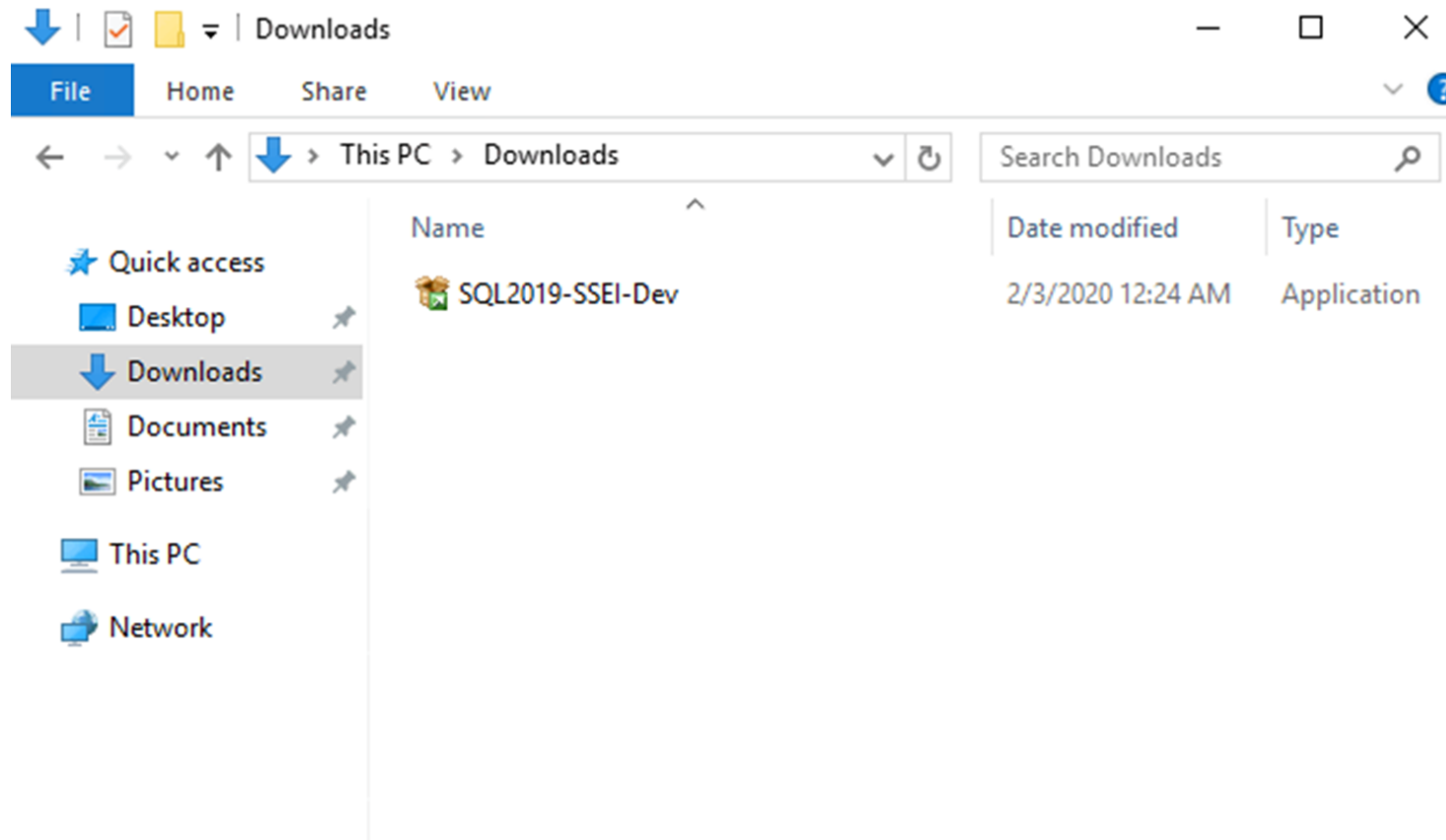
The screenshot shows the Microsoft Azure Education | Software page. The left sidebar contains navigation links: Overview, Get started, Learning resources, Roles, Software (selected), Learning, Templates, Need help?, and Support. The main content area displays a search bar and filter buttons for Product category (All), Operating System (All), System type (64 bit), and Product language (English, Multilanguage). Below the filters, a table lists 31 items. The first item is SQL Server 2019 Developer, which is highlighted in blue.

Name ↑↓	Product category ↑↓	Operating System ↑↓	System type ↑↓
SQL Server 2019 Developer	Database	Windows	64 bit
Machine Learning Server 9.4.7 fo...	AI + Machine Learning	Windows	64 bit
Microsoft R Client 9.4.7	Database	Windows	64 bit
Agents for Visual Studio 2019 (v...	Developer Tools	Windows	64 bit
Agents for Visual Studio 2019 (v...	Developer Tools	Windows	64 bit
Azure DevOps Server Express 20...	Productivity Tools	Windows	64 bit
Azure DevOps Server Express 20...	Productivity Tools	Windows	64 bit
Azure DevOps Server 2020 Upda...	Productivity Tools	Windows	64 bit
Azure DevOps Server 2020 Upda...	Productivity Tools	Windows	64 bit
Datazen Enterprise Server	Analytics	Windows	64 bit
Machine Learning Server 9.3.0 fo...	AI + Machine Learning	Windows	64 bit

➤ Chọn SQL Server 2019 Developer

# CÀI ĐẶT SQL SERVER MANAGEMENT STUDIO

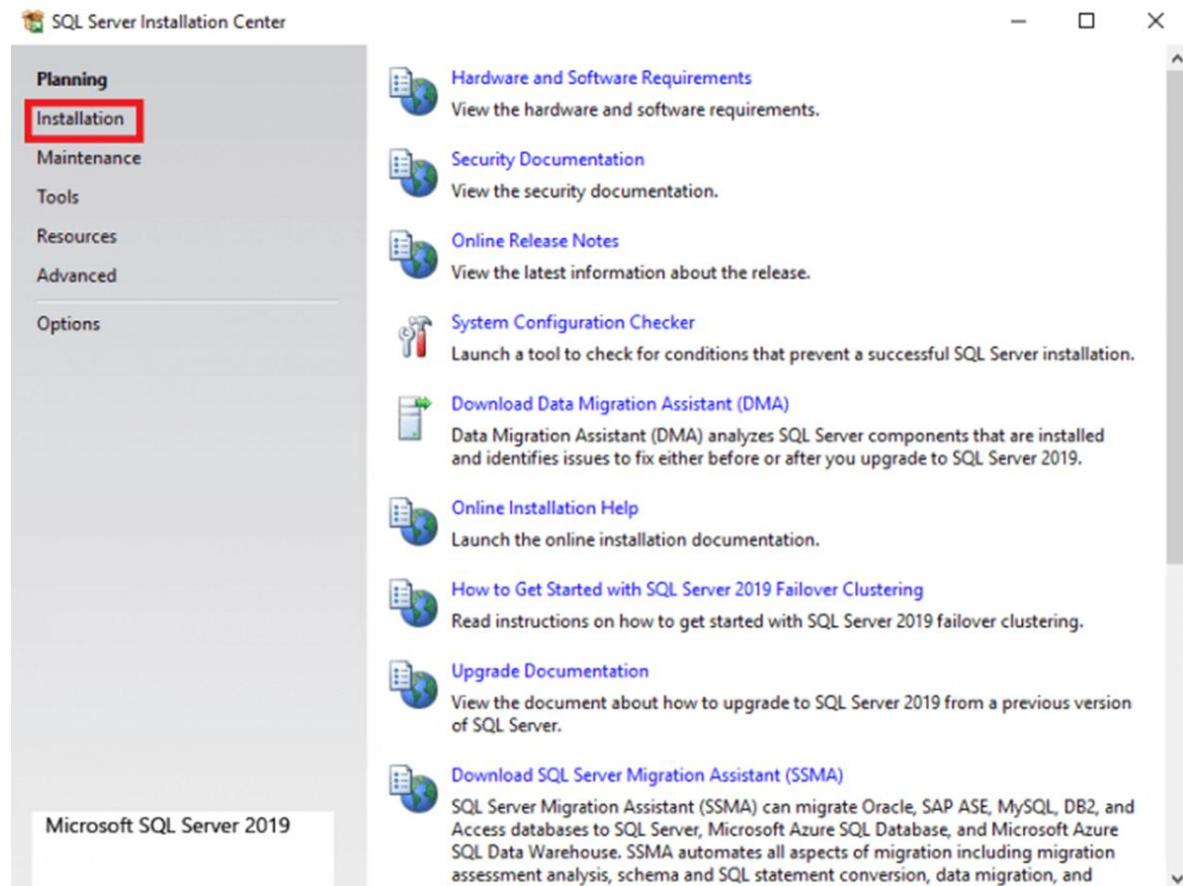
## Cài đặt SQL Server 2019 Developer:



➤ *Cài đặt SQL Server 2019 Developer*

# CÀI ĐẶT SQL SERVER MANAGEMENT STUDIO

## Cài đặt SQL Server 2019 Developer:

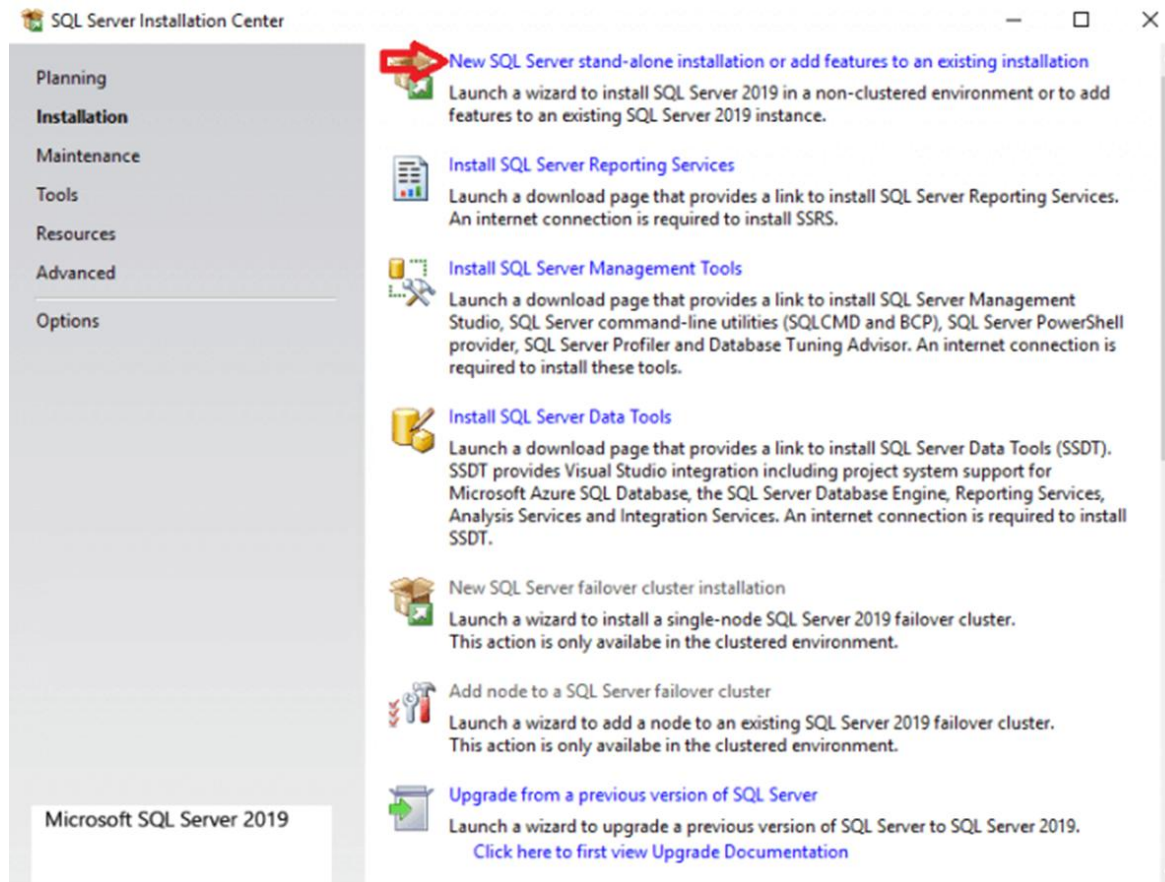


➤ *Chọn Installation*



# CÀI ĐẶT SQL SERVER MANAGEMENT STUDIO

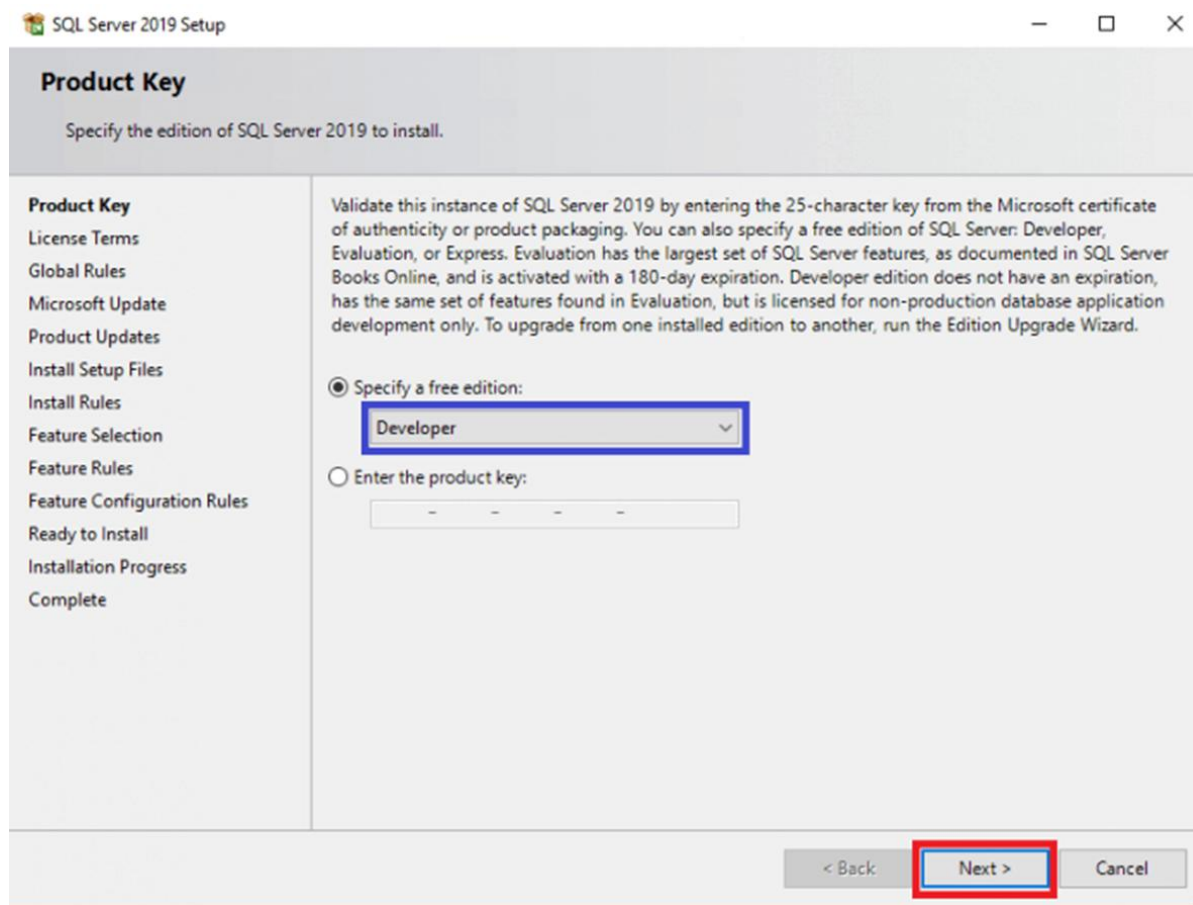
## Cài đặt SQL Server 2019 Developer:



➤ *Chọn New SQL Server stand-alone installation or add features to an existing installation*

# CÀI ĐẶT SQL SERVER MANAGEMENT STUDIO

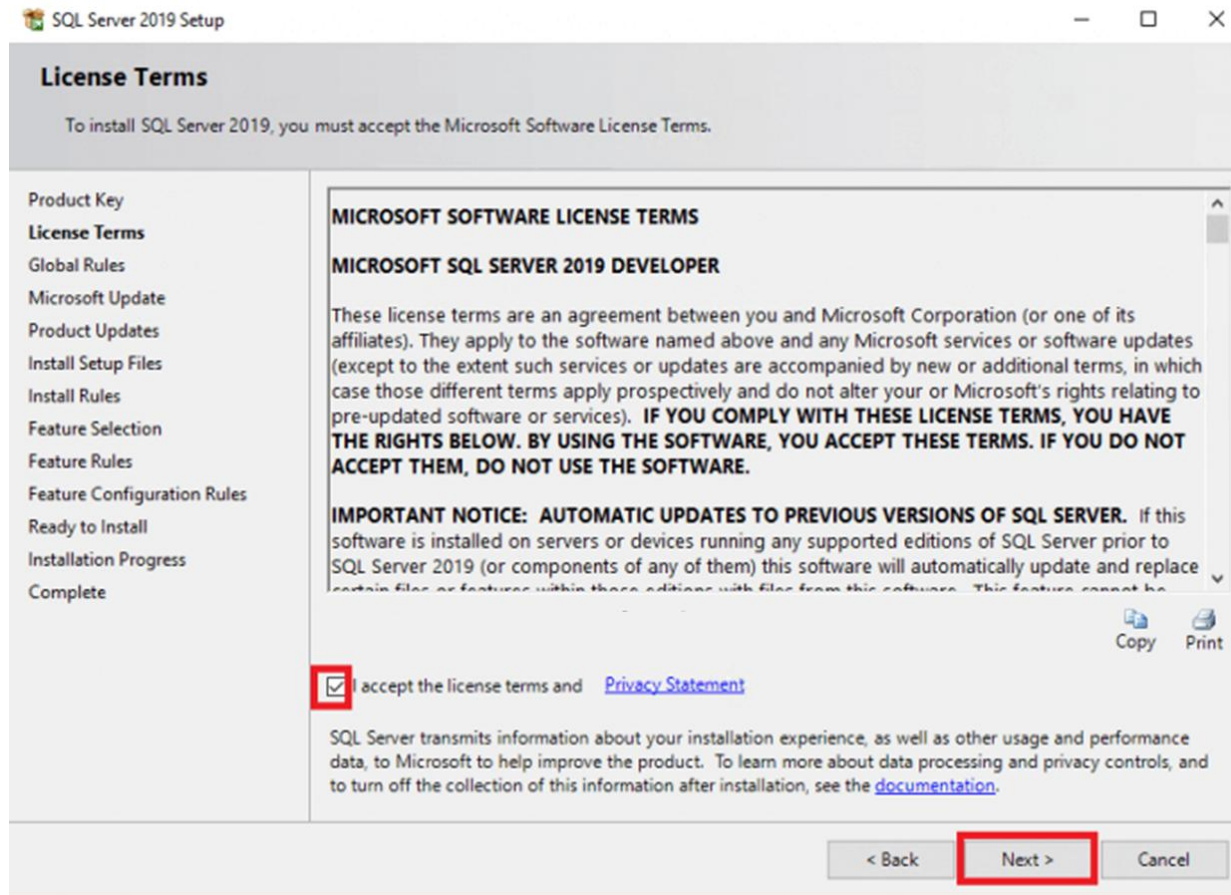
## Cài đặt SQL Server 2019 Developer:



➤ Điều chỉnh Sepecify a free edition thành Developer

# CÀI ĐẶT SQL SERVER MANAGEMENT STUDIO

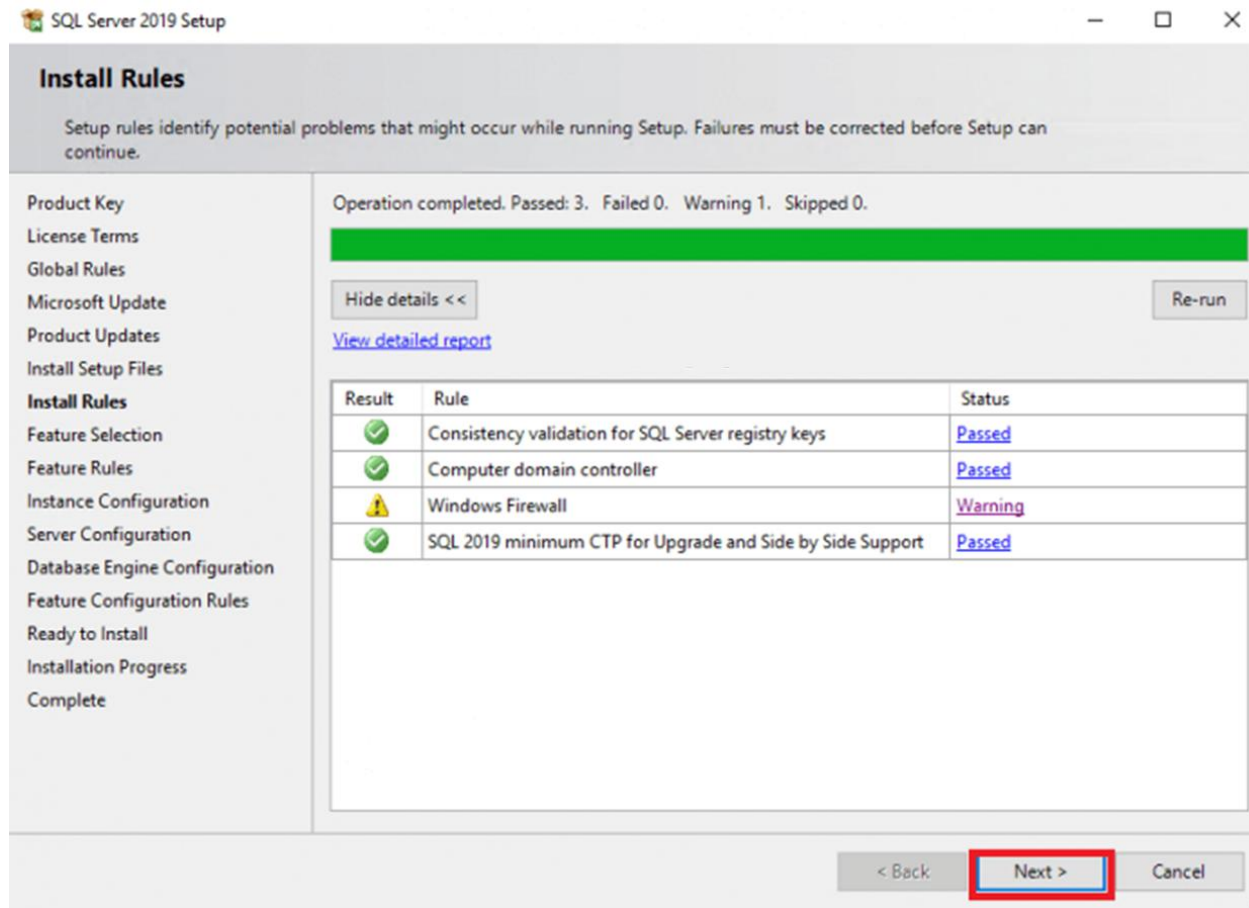
## Cài đặt SQL Server 2019 Developer:



- Tích vào I accept the license terms and privacy statement

# CÀI ĐẶT SQL SERVER MANAGEMENT STUDIO

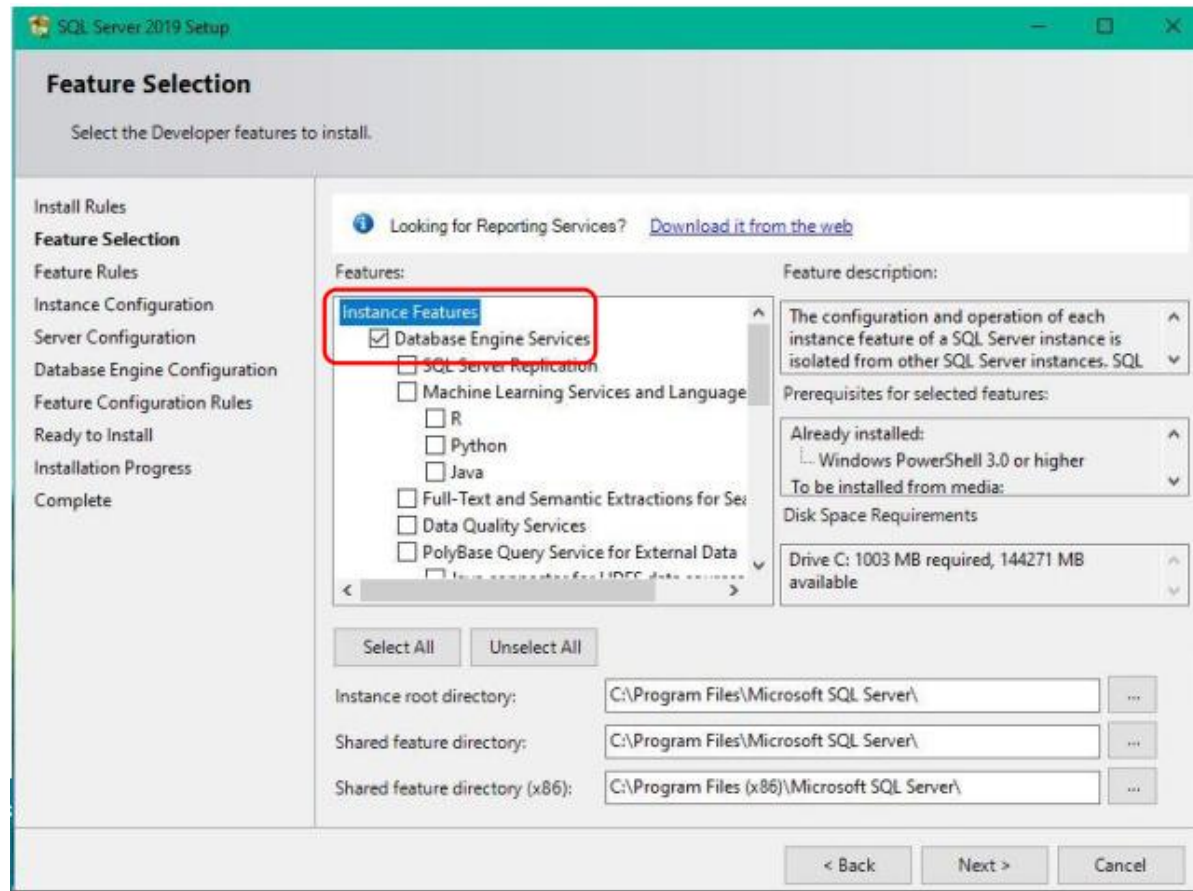
## Cài đặt SQL Server 2019 Developer:



➤ Kiểm tra như hình và nhấn Next

# CÀI ĐẶT SQL SERVER MANAGEMENT STUDIO

## Cài đặt SQL Server 2019 Developer:



➤ Tích vào cài đặt *Database Engine Services*

# CÀI ĐẶT SQL SERVER MANAGEMENT STUDIO

## Cài đặt SQL Server 2019 Developer:

**SQL Server 2019 Setup**

**Instance Configuration**

Specify the name and instance ID for the instance of SQL Server. Instance ID becomes part of the installation path.

Install Rules  
Feature Selection  
Feature Rules  
**Instance Configuration**  
Server Configuration  
Database Engine Configuration  
Feature Configuration Rules  
Ready to Install  
Installation Progress  
Complete

☒ Default instance  
☐ Named instance: MSSQLSERVER

Instance ID: MSSQLSERVER

SQL Server directory: C:\Program Files\Microsoft SQL Server\MSSQL15.MSSQLSERVER

Installed instances:

Instance Name	Instance ID	Features	Edition	Version
---------------	-------------	----------	---------	---------

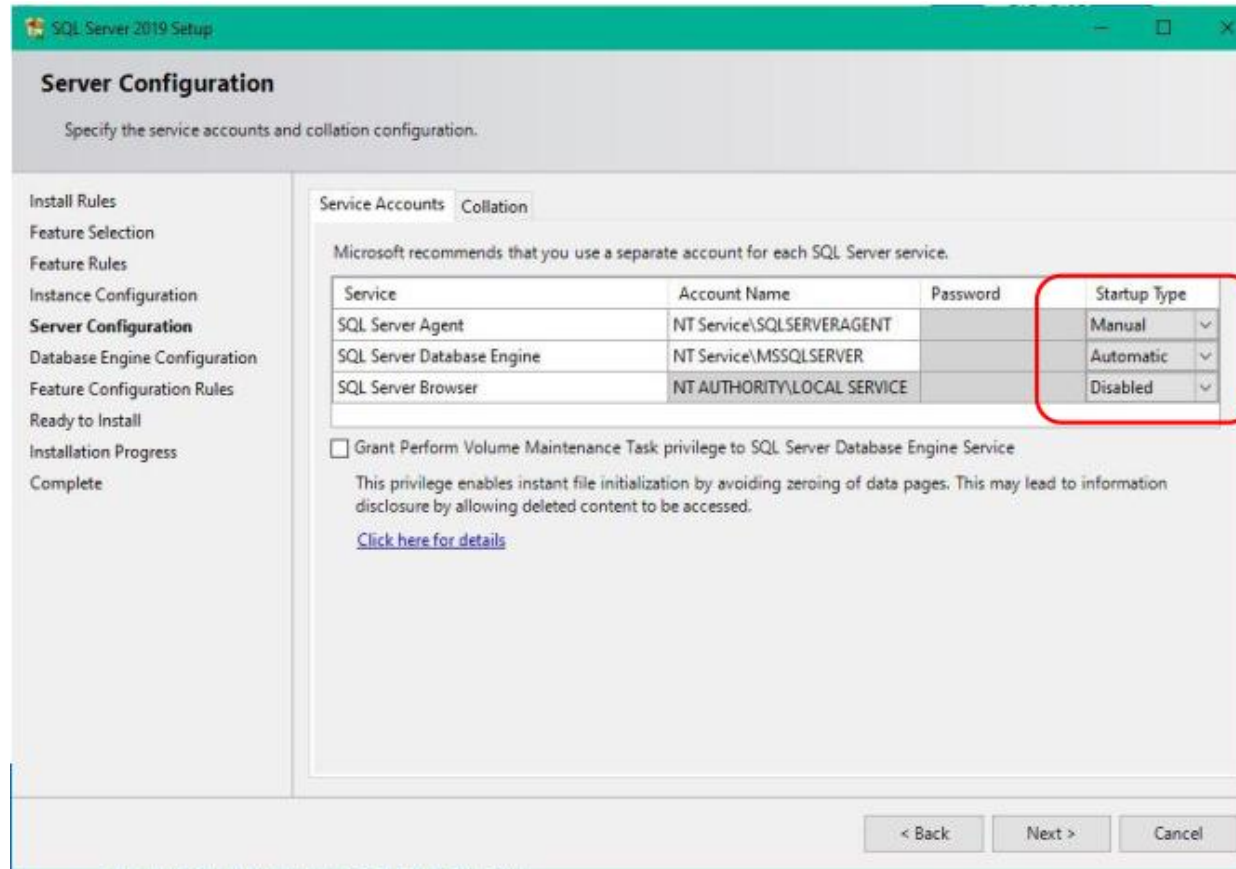
< Back   Next >   Cancel

➤ Kiểm tra Instance ID, kiểm tra xem đã tồn tại chưa



# CÀI ĐẶT SQL SERVER MANAGEMENT STUDIO

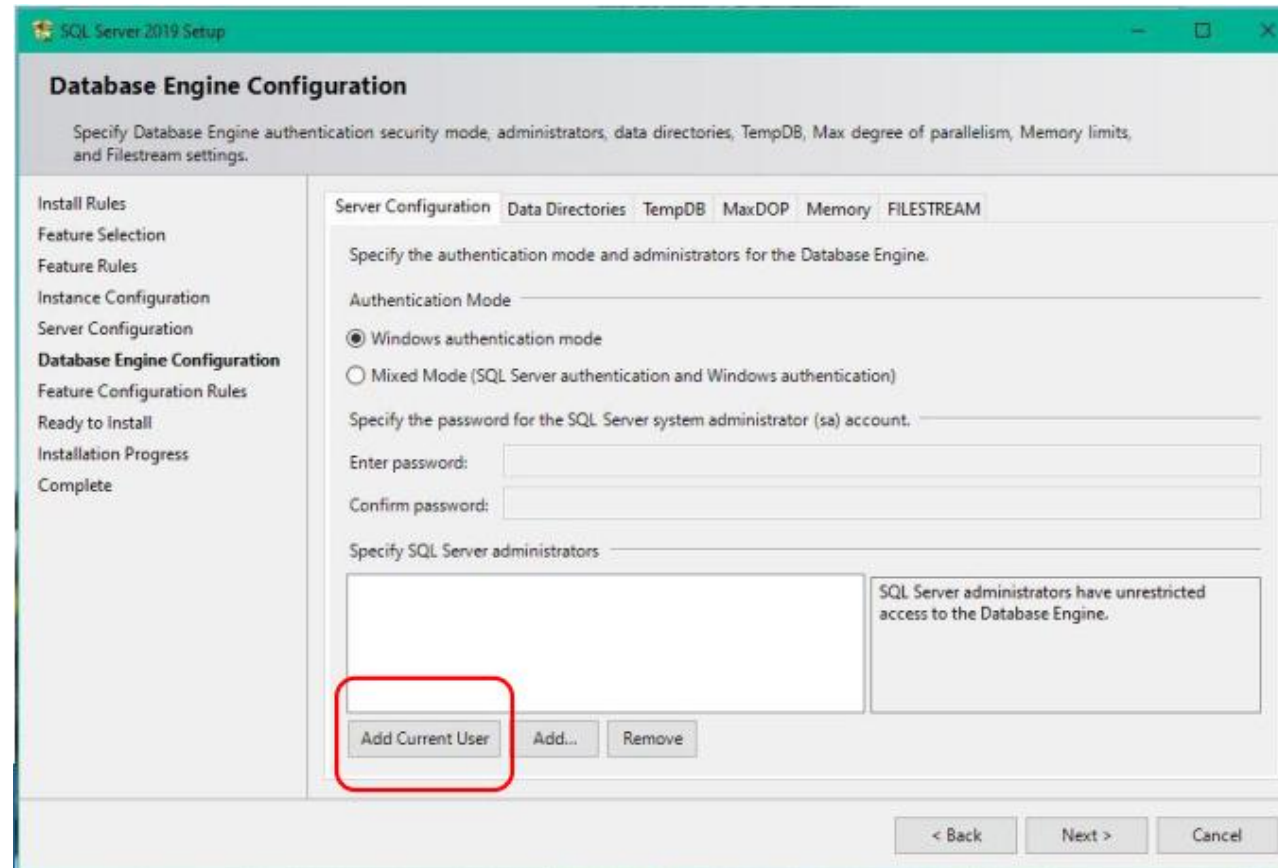
## Cài đặt SQL Server 2019 Developer:



➤ Điều chỉnh Startung Type như hình

# CÀI ĐẶT SQL SERVER MANAGEMENT STUDIO

## Cài đặt SQL Server 2019 Developer:

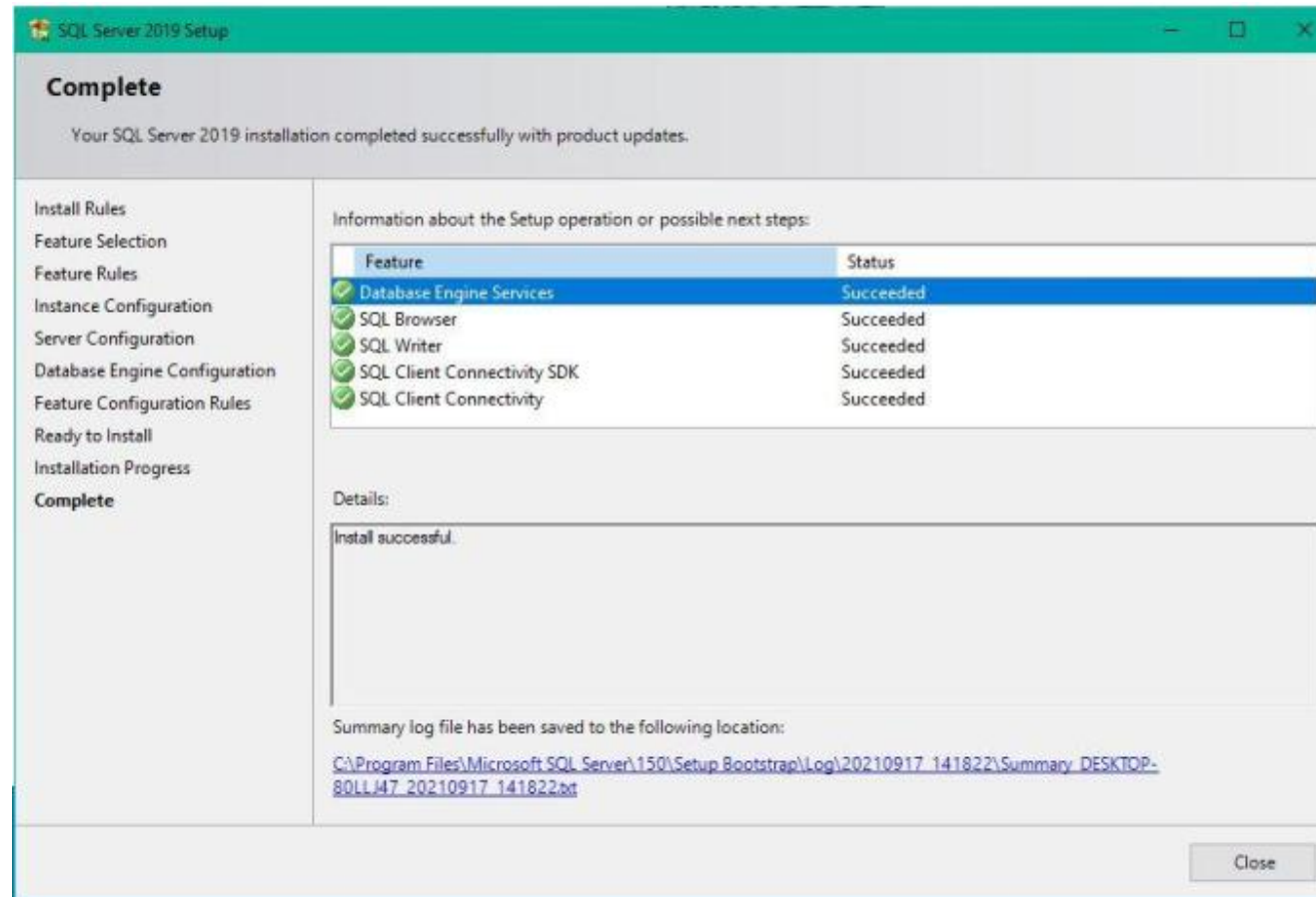


➤ *Chọn Add Current User*



# CÀI ĐẶT SQL SERVER MANAGEMENT STUDIO

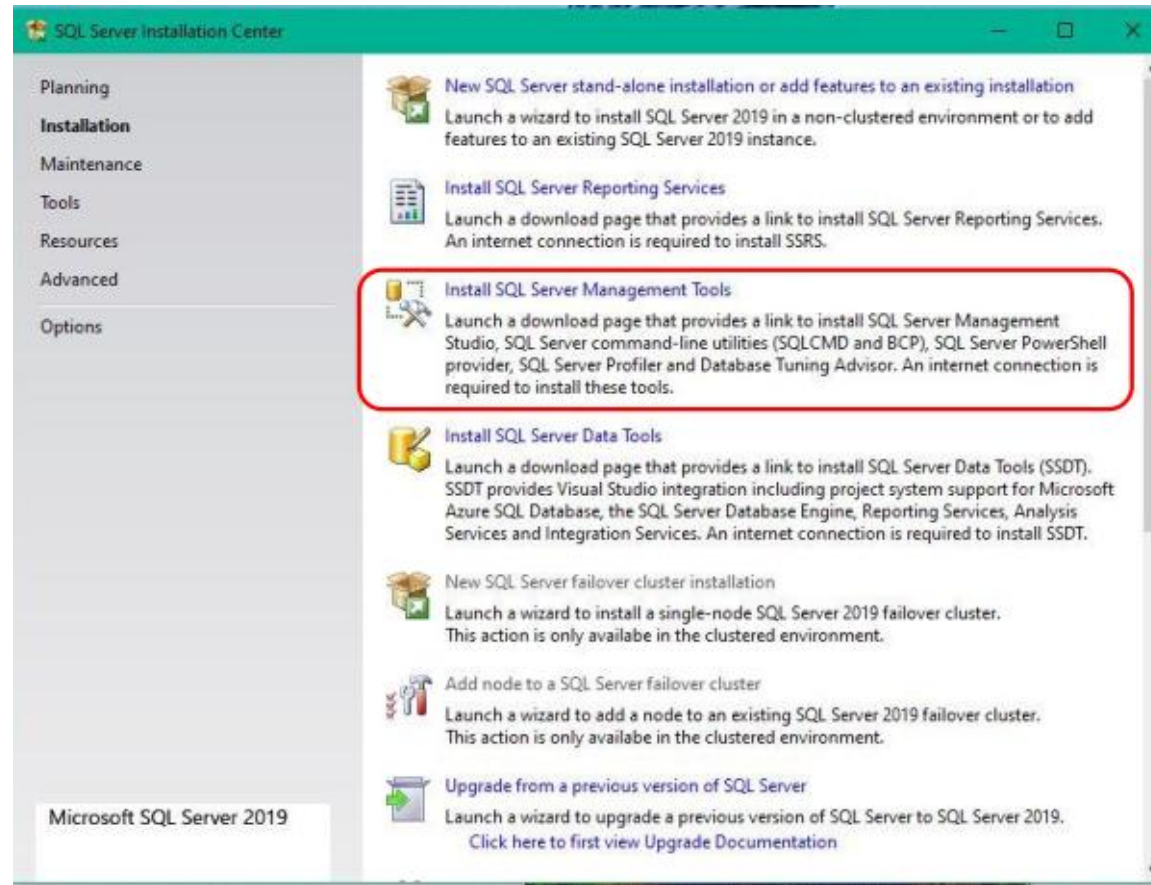
## Cài đặt SQL Server 2019 Developer:



➤ Cài đặt thành công thông báo như hình

# CÀI ĐẶT SQL SERVER MANAGEMENT STUDIO

## Cài đặt SQL Server 2019 Developer:



➤ *Tiếp tục cài đặt SQL Server Management Tools*

# CÀI ĐẶT SQL SERVER MANAGEMENT STUDIO

## Cài đặt SQL Server Management Studio (SSMS)

Microsoft | Docs | Documentation | Learn | Q&A | Code Samples

SQL Docs | Overview | Install | Secure | Develop | Administer | Analyze | Reference

Docs / SQL / Tools / SQL Server Management Studio (SSMS) / Download SSMS

### Download SQL Server Management Studio (SSMS)

07/15/2021 • 4 minutes to read

Applies to: SQL Server (all supported versions) Azure SQL Database Azure SQL Managed Instance Azure Synapse Analytics

SQL Server Management Studio (SSMS) is an integrated environment for managing any SQL infrastructure, from SQL Server to Azure SQL Database. SSMS provides tools to configure, monitor, and administer instances of SQL Server and databases. Use SSMS to deploy, monitor, and upgrade the data-tier components used by your applications, and build queries and scripts.

Use SSMS to query, design, and manage your databases and data warehouses, wherever they are - on your local computer, or in the cloud.

#### Download SSMS

[Download SQL Server Management Studio \(SSMS\) 18.9.2](#)

SSMS 18.9.2 is the latest general availability (GA) version. If you have a previous GA version of SSMS 18 installed, installing SSMS 18.9.2 upgrades it to 18.9.2.

- Release number: 18.9.2
- Build number: 15.0.18386.0
- Release date: July 15, 2021

If you have comments or suggestions, or you want to report issues, the best way to contact the SSMS team is at SQL Server user feedback. By using SQL Server Management Studio, you agree to its license terms and privacy statement.

The SSMS 18.x installation doesn't upgrade or replace SSMS versions 17.x or earlier. SSMS 18.x installs side by side with previous versions, so both versions are available for use. However, if you have a preview version of SSMS 18.x installed, you must uninstall it before installing SSMS 18.9.2. You can see if you have the preview version by going to the Help > About window.

If a computer contains side-by-side installations of SSMS, verify you start the correct version for your specific needs. The latest version is labeled Microsoft SQL Server Management Studio 18.



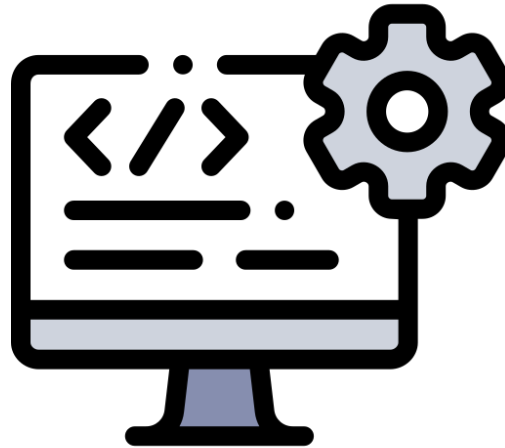
➤ Tải về và cài đặt

# CÀI ĐẶT SQL SERVER MANAGEMENT STUDIO

 Vận hành



User



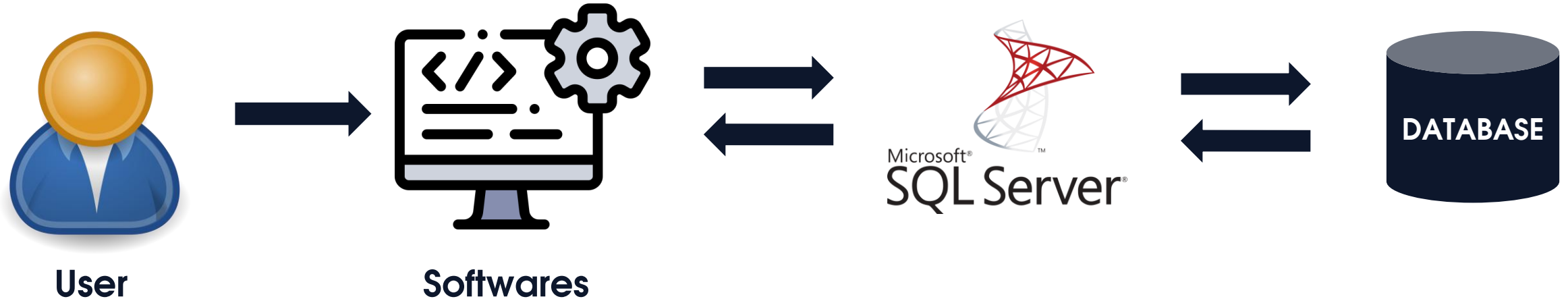
Softwares



Files

# CÀI ĐẶT SQL SERVER MANAGEMENT STUDIO

 Vận hành



## 3

### Kiểu dữ liệu trong ngôn ngữ truy vấn SQL

# Kiểu dữ liệu trong ngôn ngữ truy vấn SQL

Trong Microsoft SQL mỗi cột (column), biến cục bộ (local variable) biểu thức (expression) và tham số (parameter) đều có một kiểu dữ liệu liên quan. Sau đây là 6 kiểu dữ liệu trong SQL.

Kiểu dữ liệu	Mô tả
<b>Exact Numeric</b>	Lưu trữ số nguyên hoặc số thập phân với độ chính xác tuyệt đối.
<b>Approximate Numeric</b>	Lưu trữ số thực với độ chính xác tương đối, dùng trong tính toán khoa học.
<b>Date và Time</b>	Lưu trữ thông tin ngày tháng và thời gian.
<b>Character String</b>	Lưu trữ chuỗi ký tự có hoặc không có độ dài cố định.
<b>Unicode Character String</b>	Lưu trữ chuỗi ký tự hỗ trợ mã hóa Unicode.
<b>Binary</b>	Lưu trữ dữ liệu dạng nhị phân (binary) như hình ảnh, tệp tin.
<b>Các kiểu dữ liệu khác</b>	Lưu trữ dữ liệu chuyên biệt, như dữ liệu không gian, cây phân cấp.

Tài liệu tham khảo về các kiểu dữ liệu:

<https://learn.microsoft.com/vi-vn/sql/t-sql/data-types/data-types-transact-sql?view=sql-server-ver15>

# Kiểu dữ liệu trong ngôn ngữ truy vấn SQL

## — Kiểu dữ liệu **Exact numeric (Số chính xác, không sai số)** trong SQL

Kiểu dữ liệu	Mô tả	Phạm vi giá trị
<b>INT</b>	Lưu trữ số nguyên không dấu hoặc có dấu, kích thước cố định 4 byte.	-2,147,483,648 đến 2,147,483,647
<b>SMALLINT</b>	Lưu trữ số nguyên nhỏ hơn, kích thước cố định 2 byte.	-32,768 đến 32,767
<b>TINYINT</b>	Lưu trữ số nguyên nhỏ không dấu, kích thước cố định 1 byte.	0 đến 255
<b>BIGINT</b>	Lưu trữ số nguyên lớn hơn, kích thước cố định 8 byte.	-9,223,372,036,854,775,808 đến 9,223,372,036,854,775,807
<b>DECIMAL(p,s)</b>	Lưu trữ số thập phân với độ chính xác tuyệt đối, kích thước tùy thuộc vào p và s.	Độ chính xác lên đến 38 chữ số.
<b>NUMERIC(p,s)</b>	Tương tự như DECIMAL, lưu trữ số thập phân với độ chính xác tuyệt đối.	Độ chính xác lên đến 38 chữ số.
<b>BIT</b>	Lưu trữ giá trị nhị phân (0 hoặc 1), kích thước 1 bit.	0 hoặc 1



# Kiểu dữ liệu TRONG NGÔN NGỮ TRUY VẤN SQL

## ■ Kiểu dữ liệu **Approximate Numeric** trong SQL

Kiểu dữ liệu	Mô tả	Phạm vi giá trị
<b>FLOAT(n)</b>	Lưu trữ số thực với độ chính xác phụ thuộc vào giá trị của n (số bit).	-1.79E+308 đến 1.79E+308
<b>REAL</b>	Phiên bản FLOAT với độ chính xác cố định, kích thước 4 byte.	-3.4E+38 đến 3.4E+38

# Kiểu dữ liệu trong Ngôn ngữ Truy vấn SQL

## — Kiểu dữ liệu Date và Time trong SQL

Kiểu dữ liệu	Mô tả	Phạm vi giá trị	Định dạng
<b>DATE</b>	Lưu trữ ngày tháng, không có thời gian.	Từ năm 0001-01-01 đến 9999-12-31	'yyyy-mm-dd'
<b>TIME</b>	Lưu trữ thời gian trong ngày (giờ, phút, giây).	00:00:00.0000000 đến 23:59:59.9999999	'hh:mm:ss(.ffffff)'
<b>DATETIME</b>	Lưu trữ cả ngày và giờ, với độ chính xác đến mili giây.	Từ năm 1753-01-01 đến 9999-12-31, 00:00:00 đến 23:59:59.997	'yyyy-mm-dd hh:mm:ss'
<b>SMALLDATETIME</b>	Lưu trữ ngày và giờ, nhưng độ chính xác thấp hơn.	Từ năm 1900-01-01 đến 2079-06-06, 00:00:00 đến 23:59:59	'yyyy-mm-dd hh:mm'
<b>DATETIME2</b>	Lưu trữ ngày và giờ, với độ chính xác đến 100 nano giây.	Từ năm 0001-01-01 đến 9999-12-31, 00:00:00 đến 23:59:59.9999999	'yyyy-mm-dd hh:mm:ss(.ffffff)'
<b>DATETIMEOFFSET</b>	Lưu trữ ngày, giờ cùng với múi giờ (UTC offset).	Tương tự DATETIME2 với thêm múi giờ.	'yyyy-mm-dd hh:mm .ffffff (+

# Kiểu dữ liệu TRONG NGÔN NGỮ TRUY VẤN SQL

## Kiểu dữ liệu **Date** và **Time** trong SQL (tt)

- Lưu ý:

- Khi nhập dữ liệu ngày tháng, cần đặt trong cặp dấu ngoặc đơn và theo định dạng 'yyyy-mm-dd'.
- Ví dụ: '2024-12-24' là ngày 24 tháng 12 năm 2024.

- Sử dụng lệnh **SET DATEFORMAT**:

- Có thể thay đổi định dạng nhập ngày tháng bằng lệnh **SET DATEFORMAT**.
- Gợi ý: **SET DATEFORMAT dmy** cho phép nhập dữ liệu theo định dạng ngày tháng năm 'dd-mm-yyyy'.

# Kiểu dữ liệu TRONG NGÔN NGỮ TRUY VẤN SQL

## — Kiểu dữ liệu Character String (Kiểu chuỗi) trong SQL

Kiểu dữ liệu	Mô tả	Kích thước tối đa	Đặc điểm
<b>CHAR(n)</b>	Lưu trữ chuỗi ký tự có độ dài cố định n.	Tối đa 8,000 ký tự	Không thay đổi kích thước, padding nếu dữ liệu ngắn hơn.
<b>VARCHAR(n)</b>	Lưu trữ chuỗi ký tự có độ dài thay đổi, tối đa n.	Tối đa 8,000 ký tự (hoặc MAX)	Linh hoạt, không lãng phí không gian lưu trữ.
<b>VARCHAR(MAX)</b>	Lưu trữ chuỗi ký tự có độ dài thay đổi với kích thước lớn.	Tối đa 2.147.483.647 byte (khoảng 2GB)	Thích hợp cho các văn bản dài hoặc dữ liệu lớn.
<b>TEXT</b>	Lưu trữ chuỗi văn bản dài, hiện không khuyến khích sử dụng.	Tối đa 2.147.483.647 byte (khoảng 2GB)	Sử dụng VARCHAR(MAX) thay thế trong các phiên bản mới.

# Kiểu dữ liệu TRONG NGÔN NGỮ TRUY VẤN SQL

## — Kiểu dữ liệu **Unicode Character String** (Kiểu chuỗi có chứa Unicode) trong SQL

Kiểu dữ liệu	Mô tả	Kích thước tối đa	Đặc điểm
<b>NCHAR(n)</b>	Lưu trữ chuỗi ký tự Unicode có độ dài cố định n.	Tối đa 4,000 ký tự	Mỗi ký tự chiếm 2 byte, thích hợp cho dữ liệu cố định.
<b>NVARCHAR(n)</b>	Lưu trữ chuỗi ký tự Unicode có độ dài thay đổi, tối đa n.	Tối đa 4,000 ký tự (hoặc MAX)	Linh hoạt, không lãng phí không gian lưu trữ, hỗ trợ Unicode.
<b>NVARCHAR(MAX)</b>	Lưu trữ chuỗi ký tự Unicode có độ dài thay đổi với kích thước lớn.	Tối đa 2.147.483.647 byte (khoảng 2GB)	Thích hợp cho dữ liệu văn bản dài, hỗ trợ Unicode.
<b>TEXT</b>	Lưu trữ chuỗi văn bản dài bằng Unicode.	Tối đa 2.147.483.647 byte (khoảng 2GB)	Trước đây sử dụng cho văn bản dài, hiện không còn khuyến khích sử dụng, thay thế bằng NVARCHAR(MAX).

# Kiểu dữ liệu trong Ngôn ngữ Truy vấn SQL

## — Kiểu dữ liệu **Binary** trong SQL

Kiểu dữ liệu	Mô tả	Kích thước tối đa	Đặc điểm
<b>BINARY(n)</b>	Lưu trữ dữ liệu nhị phân có độ dài cố định n.	Tối đa 8.000 byte	Dùng khi biết trước độ dài dữ liệu, cố định kích thước.
<b>VARBINARY(n)</b>	Lưu trữ dữ liệu nhị phân có độ dài thay đổi, tối đa n.	Tối đa 8.000 byte	Linh hoạt, không lãng phí không gian lưu trữ.
<b>VARBINARY(MAX)</b>	Lưu trữ dữ liệu nhị phân có độ dài thay đổi với kích thước lớn.	Tối đa 2.147.483.647 byte (khoảng 2GB)	Thích hợp cho dữ liệu nhị phân lớn như hình ảnh, tệp tin, tài liệu.
<b>IMAGE</b>	Lưu trữ dữ liệu nhị phân lớn, hiện không khuyến khích sử dụng.	Tối đa 2.147.483.647 byte (khoảng 2GB)	Trước đây dùng cho hình ảnh hoặc dữ liệu nhị phân lớn, nên thay thế bằng VARBINARY(MAX).

# Kiểu dữ liệu trong ngôn ngữ truy vấn SQL

## Các kiểu dữ liệu khác

Kiểu dữ liệu	Mô tả	Đặc điểm
<b>BIT</b>	Lưu trữ giá trị nhị phân, chỉ nhận giá trị 0, 1 hoặc NULL.	Thích hợp cho các giá trị Boolean như True/False, On/Off.
<b>MONEY</b>	Lưu trữ giá trị tiền tệ.	Chính xác cao, hỗ trợ 4 số thập phân sau dấu phẩy.
<b>SMALLMONEY</b>	Lưu trữ giá trị tiền tệ với phạm vi nhỏ hơn MONEY.	Phạm vi từ -214,748.3648 đến 214,748.3647.
<b>UNIQUEIDENTIFIER</b>	Lưu trữ giá trị GUID (Global Unique Identifier).	Được dùng để lưu trữ các ID duy nhất toàn cầu.
<b>XML</b>	Lưu trữ dữ liệu dưới dạng XML.	Hỗ trợ truy vấn và thao tác dữ liệu XML trực tiếp trong SQL.

# Kiểu dữ liệu trong ngôn ngữ truy vấn SQL

## Các kiểu dữ liệu khác (tt)

Kiểu dữ liệu	Mô tả	Đặc điểm
<b>JSON</b>	Lưu trữ và truy vấn dữ liệu JSON.	Không phải kiểu dữ liệu riêng biệt, nhưng SQL Server hỗ trợ xử lý JSON.
<b>TABLE</b>	Lưu giữ tập hợp kết quả để xử lý vào lần sau.	Dùng trong biến bảng để lưu trữ kết quả truy vấn cho các phép toán sau.
<b>CURSOR</b>	Tham chiếu tới một đối tượng con trỏ (Cursor).	Dùng để xử lý từng hàng của tập hợp kết quả, hữu ích trong các xử lý tuần tự.
<b>TIMESTAMP</b>	Lưu giữ một số duy nhất, được cập nhật mỗi khi một hàng được cập nhật.	Dùng để theo dõi sự thay đổi dữ liệu, hiện không còn khuyến khích sử dụng, thay thế bằng ROWVERSION.
<b>SQL_VARIANT</b>	Lưu giữ các giá trị của các kiểu dữ liệu đa dạng, ngoại trừ TEXT, NTEXT, và TIMESTAMP.	Cho phép lưu trữ nhiều kiểu dữ liệu trong cùng một cột, linh hoạt nhưng cần cẩn thận khi thao tác.



## 4

### CÁC LỆNH VỀ DATABASE, TABLE, RÀNG BUỘC TOÀN VẠN

# CÁC LỆNH VỀ DATABASE

## Tạo Database mới trong SQL Server

- Để tạo một Database mới ta sử dụng cú pháp:

**CREATE DATABASE** <Tên Database>;

- Ví dụ: **CREATE DATABASE** QuanLyBanHang;

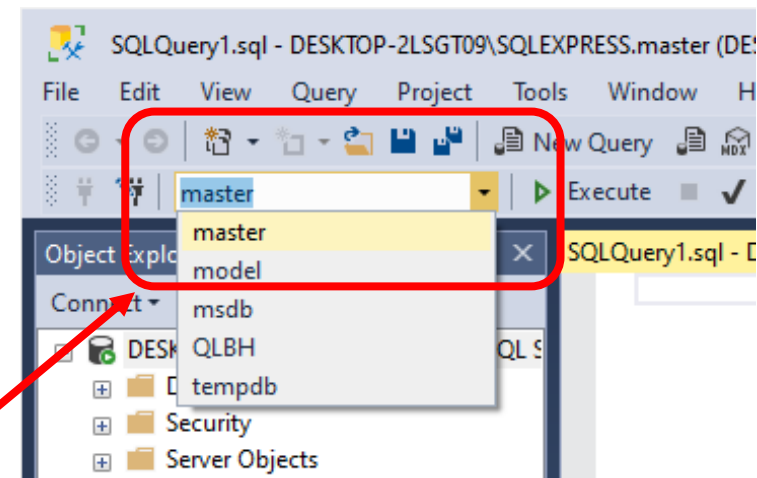
## Sử dụng Database trong SQL Server

- Sau khi tạo mới một Database ta cần dùng lệnh **USE** để chỉ định Database sẽ thao tác trên đó.

Cú pháp như sau:

**USE** <Tên Database>;

- Ví dụ: **USE** QuanLyBanHang;



Chú ý: "Available Database" có đúng Database cần thao tác chưa

# CÁC LỆNH VỀ DATABASE

## Xóa Database trong SQL Server

- Để xóa một Database ta sử dụng cú pháp:

```
DROP DATABASE <Tên Database>;
```

- Ví dụ: **DROP DATABASE** *QuanLyBanHang*;

## Xóa nhiều Database cùng một lúc

- Để xóa nhiều database cùng một lúc:

```
DROP DATABASE <Tên Database 1>, <Tên Database 2>, ..., <Tên Database N>;
```

- Ví dụ: **DROP DATABASE** *QuanLyGiaoVu, QuanLyCaSi, QuanLyBanHang*;

**Chú ý:** Cơ sở dữ liệu đang được sử dụng hoặc kết nối không thể bị xóa. Đảm bảo rằng cơ sở dữ liệu không còn kết nối trước khi thực hiện lệnh xóa.

# CÁC LỆNH VỀ TABLE

## Thuật ngữ SQL và cơ sở dữ liệu

- SQL sử dụng các thuật ngữ tương đương với thuật ngữ về cơ sở dữ liệu như sau:

SQL Term	Cơ sở dữ liệu
<b>Bảng</b>	Quan hệ (Relation)
<b>Cột</b>	Thuộc tính (Attribute)
<b>Dòng Dữ Liệu</b>	Bộ giá trị (Tuple)

- Để định nghĩa một bảng trong SQL ta cần: tên bảng, các cột, kiểu dữ liệu của cột và các ràng buộc toàn vẹn (**RBTV - Integrity constraint**) trên nó.

# CÁC LỆNH VỀ TABLE

## Định nghĩa bảng trong SQL

- Để tạo bảng (**Table**), ta sử dụng cú pháp sau:

```
CREATE TABLE <Tên Table>
(
    <Tên column 1> <Kiểu dữ liệu> (integrity_constraint),
    <Tên column 2> <Kiểu dữ liệu> (integrity_constraint),
    ...
    (integrity_constraint)
);
```

# CÁC LỆNH VỀ TABLE

## Định nghĩa bảng trong SQL

- Ví dụ: Tạo bảng Khách hàng và bảng Sinh viên

### CREATE TABLE KHACHHANG

```
(  
    MaKH char(4) PRIMARY KEY,  
    HoTen varchar(40) NOT NULL,  
    DiaChi varchar(50) NOT NULL,  
    NgaySinh smalldatetime,  
    DoanhSo money  
);
```

### CREATE TABLE SINHVIEN

```
(  
    MaSV INT PRIMARY KEY,  
    HoTen NVARCHAR(50) NOT NULL,  
    NgaySinh DATE,  
    Lop NVARCHAR(20)  
);
```

# CÁC LỆNH VỀ TABLE

## Xóa bảng (Table) trong SQL

- Để xóa bảng (**Table**), ta sử dụng cú pháp sau:

```
DROP TABLE <Tên Table>;
```

- Ví dụ: Xóa bảng Khách hàng: **DROP TABLE KHACHHANG**

➤ *Khi bảng bị xóa, tất cả dữ liệu bên trong bảng đó cũng sẽ bị xóa vĩnh viễn.*

- Lưu ý:

- Không thể xóa bảng khi nó đang có liên kết khóa ngoại (Foreign Key).
- Trước khi xóa bảng, cần phải xóa các ràng buộc khóa ngoại liên kết với bảng đó.

```
ALTER TABLE <Tên Table> DROP CONSTRAINT <Tên khóa ngoại>;
```

# CÁC LỆNH VỀ TABLE

## Mở rộng các lệnh DROP trong SQL

- Để xóa một bảng View ta sử dụng cú pháp

```
DROP VIEW <Tên View>;
```

- Để xóa một Procedure ta sử dụng cú pháp

```
DROP PROCEDURE <Tên Procedure>;
```

- Ngoài ra còn các lệnh DROP khác như là 'DROP INDEX', 'DROP ROLE', 'DROP SCHEMA', 'DROP RULE'...



# CÁC LỆNH VỀ TABLE

## Chỉnh sửa bảng trong SQL với ALTER TABLE

- SQL hỗ trợ **ALTER TABLE** cho phép thêm, xóa hay sửa các thành phần của table.
- Để sửa Table, ta sử dụng cú pháp sau:

- ❖ Thêm cột mới:

**ALTER TABLE** <Tên Table> **ADD** <Tên cột> <Kiểu dữ liệu>;

➤ Ví dụ: **ALTER TABLE** SinhVien **ADD** Email NVARCHAR(50);

- ❖ Xóa cột

**ALTER TABLE** <Tên Table> **DROP COLUMN** <Tên cột>;

➤ Ví dụ: **ALTER TABLE** SinhVien **DROP COLUMN** Lop;

- ❖ Sửa cột (Thay đổi kiểu dữ liệu)

**ALTER TABLE** <Tên Table> **ALTER COLUMN** <Tên cột> <Kiểu dữ liệu mới>;

➤ Ví dụ: **ALTER TABLE** SinhVien **ALTER COLUMN** NgaySinh DATE;

# CÁC LỆNH VỀ TABLE

## Đổi tên và thông tin bảng trong SQL

- Để thực hiện **đổi tên bảng (Rename table)** ta sử dụng cú pháp:

```
EXEC sp_rename '<Tên Table>', '<Tên Table mới>';
```

➤ Ví dụ: **EXEC** sp\_rename 'SinhVien', 'HocVien';

- Để thực hiện **đổi tên cột (Rename Column)** ta sử dụng cú pháp:

```
EXEC sp_rename '<Tên Table.Tên Column>', '<Tên Column mới>', 'COLUMN';
```

➤ Ví dụ: **EXEC** sp\_rename 'SinhVien.HoTen', 'TenDayDu', '**COLUMN**';

- Ngoài ra, còn có các lệnh khác như sp\_help, sp\_tables, sp\_columns, sp\_pkeys, sp\_fkeys, sp\_helpconstraint, sp\_helpsql,...

- Lưu ý: Sinh viên có thể tham khảo thêm các lệnh sp\_ khác tại: **TẠI ĐÂY**

# RÀNG BUỘC TOÀN VỆN TRONG SQL

## Phân loại ràng buộc toàn vẹn

- Việc ràng buộc toàn vẹn trong SQL Server được chia làm 2 loại chính:
  - **Ràng buộc đơn giản (Simple constraints):** Sử dụng **CONSTRAINT** để mô tả các quy tắc toàn vẹn.
  - **Ràng buộc phức tạp (Complex constraints):** Sử dụng **TRIGGER** để thực hiện các quy tắc toàn vẹn phức tạp hơn mà **CONSTRAINT** không hỗ trợ.

# RÀNG BUỘC TOÀN VỆN TRONG SQL

## Các ràng buộc thường sử dụng trong SQL Server

- **PRIMARY KEY:** Ràng buộc khóa chính, kết hợp giữa **NOT NULL** và **UNIQUE**. Xác định duy nhất mỗi bản ghi trong bảng.
- **FOREIGN KEY:** Thiết lập mối quan hệ giữa các bảng, ràng buộc khóa ngoại.
- **NOT NULL:** Đảm bảo dữ liệu trong cột không được để trống.
- **UNIQUE:** Đảm bảo dữ liệu trong cột là duy nhất, không bị trùng lặp.
- **CHECK:** Kiểm tra dữ liệu nhập vào phải theo định dạng hoặc điều kiện nhất định.
- **DEFAULT:** Đặt giá trị mặc định cho cột nếu không có giá trị được cung cấp.

# RÀNG BUỘC TOÀN VỆN TRONG SQL

## Các ràng buộc thường sử dụng trong SQL Server (tt)

### ▪ PRIMARY KEY

- **Mô tả:** Đảm bảo giá trị của cột là duy nhất và không rỗng. Một bảng chỉ có thể có một khóa chính.
- **Ví dụ:**

```
CREATE TABLE SinhVien
```

```
(
```

```
    MaSV INT PRIMARY KEY,
```

```
    TenSV NVARCHAR(50)
```

```
);
```

# RÀNG BUỘC TOÀN VỆN TRONG SQL

## Các ràng buộc thường sử dụng trong SQL Server (tt)

### ▪ FOREIGN KEY

- **Mô tả:** Đảm bảo giá trị của cột phù hợp với giá trị của một cột trong bảng khác. Thiết lập mối quan hệ giữa các bảng.
- Ví dụ:

CREATE TABLE Lop

(  
    MaLop INT PRIMARY KEY,  
    TenLop NVARCHAR(50)  
);

CREATE TABLE SinhVien

(  
    MaSV INT PRIMARY KEY,  
    TenSV NVARCHAR(50),  
    MaLop INT,  
    FOREIGN KEY (MaLop) REFERENCES Lop(MaLop)  
);

# RÀNG BUỘC TOÀN VỆN TRONG SQL

## Các ràng buộc thường sử dụng trong SQL Server (tt)

- **NOT NULL**

- **Mô tả:** Đảm bảo dữ liệu trong cột không được để trống (không được phép có giá trị NULL).
- **Ví dụ:**

```
CREATE TABLE SinhVien
```

```
(
```

```
    MaSV INT PRIMARY KEY,
```

```
    TenSV NVARCHAR(50) NOT NULL
```

```
);
```

# RÀNG BUỘC TOÀN VỆN TRONG SQL

## Các ràng buộc thường sử dụng trong SQL Server (tt)

- **UNIQUE**

- **Mô tả:** Đảm bảo giá trị của cột là duy nhất trong bảng. Một bảng có thể có nhiều ràng buộc UNIQUE.
- **Ví dụ:**

```
CREATE TABLE SinhVien
```

```
(
```

```
    MaSV INT PRIMARY KEY,
```

```
    Email NVARCHAR(50) UNIQUE
```

```
);
```



# RÀNG BUỘC TOÀN VỆN TRONG SQL

## Các ràng buộc thường sử dụng trong SQL Server (tt)

- **CHECK**

- **Mô tả:** Đảm bảo giá trị của cột thỏa mãn một điều kiện nhất định.
- **Ví dụ:**

```
CREATE TABLE SinhVien
```

```
(
```

```
    MaSV INT PRIMARY KEY,
```

```
    TenSV NVARCHAR(50),
```

```
    Diem INT CHECK (Diem >= 0 AND Diem <= 10)
```

```
);
```

# RÀNG BUỘC TOÀN VỆN TRONG SQL

## Các ràng buộc thường sử dụng trong SQL Server (tt)

- **DEFAULT**

- **Mô tả:** Đặt giá trị mặc định cho cột nếu không có giá trị được cung cấp.
- **Ví dụ:**

```
CREATE TABLE SinhVien
```

```
(
```

```
    MaSV INT PRIMARY KEY,
```

```
    TenSV NVARCHAR(50),
```

```
    NgayNhapHoc DATE DEFAULT GETDATE()
```

```
);
```

# RÀNG BUỘC TOÀN VỆN TRONG SQL

## Nguyên tắc chung về ràng buộc (Constraint) trong SQL Server

- **Nguyên tắc 1: Ràng buộc gắn với bảng**
  - Mỗi constraint luôn được liên kết với một bảng cụ thể.
- **Nguyên tắc 2: Đặt tên cho Constraint**
  - Nếu không đặt tên, hệ thống sẽ tự động phát sinh tên cho constraint.
  - **Ví dụ:** PK\_TableName, FK\_TableName\_ColumnName.

# RÀNG BUỘC TOÀN VỆN TRONG SQL

## Nguyên tắc chung về ràng buộc (Constraint) trong SQL Server (tt)

### ▪ Nguyên tắc 3: Tạo Constraint

- Cùng thời điểm tạo bảng: Ràng buộc có thể được tạo ngay khi tạo bảng.

```
CREATE TABLE SinhVien (  
    MaSV INT PRIMARY KEY,  
    TenSV NVARCHAR(50) NOT NULL  
);
```

- Sau khi tạo bảng (dùng **ALTER**): Ràng buộc cũng có thể được thêm sau khi bảng đã được tạo.

```
ALTER TABLE SinhVien ADD CONSTRAINT PK_SinhVien PRIMARY KEY (MaSV);
```

# RÀNG BUỘC TOÀN VỆN TRONG SQL

## Nguyên tắc chung về ràng buộc (Constraint) trong SQL Server (tt)

### ▪ Nguyên tắc 4: Mức cột và mức bảng

- Mức cột: Constraint được khai báo trực tiếp trên một cột.

**MaSV** INT PRIMARY KEY

- Mức bảng: Constraint có thể được khai báo ở mức bảng, áp dụng cho nhiều cột.

**CONSTRAINT PK\_SinhVien** PRIMARY KEY (**MaSV**)

### ▪ Nguyên tắc 5: Xem các Constraint hiện có

- Có thể kiểm tra và liệt kê các constraint hiện có trong database.
- Sử dụng các câu lệnh như **sp\_helpconstraint** hoặc xem qua **Object Explorer** trong SQL Server Management Studio.

# RÀNG BUỘC TOÀN VỆN TRONG SQL

## Cú pháp chi tiết các loại Constraint trong SQL Server

Loại Constraint	Cú pháp
<b>PRIMARY KEY</b>	sql (CONSTRAINT ten_constraint) <b>PRIMARY KEY</b> (danh_sach_cot_khoa_chinh)
<b>FOREIGN KEY</b>	sql (CONSTRAINT ten_constraint) <b>FOREIGN KEY</b> (danh_sach_cot_khoangoai) <b>REFERENCES</b> bang_tham_chieu (ds_cot_tham_chieu)
<b>NOT NULL</b>	sql <Tên Cột> <Kiểu Dữ Liệu> <b>NOT NULL</b>
<b>UNIQUE</b>	sql (CONSTRAINT ten_constraint) <b>UNIQUE</b> (danh_sach_cot)
<b>CHECK</b>	sql (CONSTRAINT ten_constraint) <b>CHECK</b> (bieu_thuc_luan_ly)
<b>DEFAULT</b>	sql (CONSTRAINT ten_constraint) <b>DEFAULT</b> gia_tri_mac_dinh FOR ten_cot

# RÀNG BUỘC TOÀN VỆN TRONG SQL

## Các cách viết ràng buộc (Constraint) đơn giản trong SQL Server

- Cách 1: Thêm ràng buộc trong khi tạo bảng
  - Cú pháp:

```
CREATE TABLE <Tên Table>
```

```
(
```

```
    <Tên column 1> <Kiểu dữ liệu> (CONSTRAINT ten_constraint) <Loại ràng buộc> ,
```

```
    <Tên column 2> <Kiểu dữ liệu> (CONSTRAINT ten_constraint) <Loại ràng buộc> ,
```

```
    ...
```

```
);
```

➤ Lưu ý: Nên dùng với **PRIMARY KEY, NOT NULL, UNIQUE, DEFAULT.**

# RÀNG BUỘC TOÀN VỆN TRONG SQL

## Các cách viết ràng buộc (Constraint) đơn giản trong SQL Server

- Cách 1: Thêm ràng buộc trong khi tạo bảng (tt)
  - Cú pháp:

**CREATE TABLE** <Tên Table> (

<Tên column 1> <Kiểu dữ liệu> (**PRIMARY KEY**),

<Tên column 2> <Kiểu dữ liệu> (**NOT NULL**),

<Tên column 3> <Kiểu dữ liệu> (**UNIQUE**),

<Tên column 4> <Kiểu dữ liệu> (**FOREIGN KEY REFERENCES** Ten\_Bang\_Khac(Ten\_Cot\_Khac)),

<Tên column 5> <Kiểu dữ liệu> (**CHECK** (bieu\_thuc\_logic)),

<Tên column 6> <Kiểu dữ liệu> (**DEFAULT** gia\_tri\_mac\_dinh)

);



# RÀNG BUỘC TOÀN VỆN TRONG SQL

## Các cách viết ràng buộc (Constraint) đơn giản trong SQL Server

- Cách 1: Thêm ràng buộc trong khi tạo bảng (tt)
  - Ví dụ:

**CREATE TABLE SinhVien**

(

**MaSV** INT **CONSTRAINT** pk\_MaSV PRIMARY KEY,

**TenSV** NVARCHAR(50) **CONSTRAINT** uq\_TenSV UNIQUE,

**MaLop** INT **CONSTRAINT** fk\_MaLop FOREIGN KEY REFERENCES Lop(MaLop),

**NgaySinh** DATE **CONSTRAINT** chk\_NgaySinh CHECK (NgaySinh >= '1900-01-01'),

**GioiTinh** CHAR(1) **CONSTRAINT** df\_GioiTinh DEFAULT 'M'

);

# RÀNG BUỘC TOÀN VỆN TRONG SQL

## Các cách viết ràng buộc (Constraint) đơn giản trong SQL Server

- Cách 1: Thêm ràng buộc trong khi tạo bảng (tt)

- Ví dụ:

```
CREATE TABLE SinhVien
```

```
(
```

```
    MaSV INT CONSTRAINT pk_MaSV PRIMARY KEY,
```

```
    TenSV NVARCHAR(50) UNIQUE,
```

```
    MaLop INT FOREIGN KEY REFERENCES Lop(MaLop),
```

```
    NgaySinh DATE CHECK (NgaySinh >= '1900-01-01'),
```

```
    GioiTinh CHAR(1) DEFAULT 'M'
```

```
);
```

# RÀNG BUỘC TOÀN VỆN TRONG SQL

## Các cách viết ràng buộc (Constraint) đơn giản trong SQL Server

- Cách 2: Thêm ràng buộc ngay tại dòng cuối cùng của lệnh tạo bảng
  - Cú pháp:

```
CREATE TABLE <Tên Table>
(
    <Tên cột 1> <Kiểu dữ liệu>,
    <Tên cột 2> <Kiểu dữ liệu>,
    ...
    (CONSTRAINT ten_constraint) <Loại ràng buộc>
);
```

- **Lưu ý:** Nên dùng với Primary Key (Khóa chính có nhiều thuộc tính).

# RÀNG BUỘC TOÀN VỆN TRONG SQL

Các cách viết ràng buộc (Constraint) đơn giản trong SQL Server

- Cách 2: Thêm ràng buộc ngay tại dòng cuối cùng của lệnh tạo bảng (tt)
  - Ví dụ:

```
CREATE TABLE CTHD  
(  
    SOHD INT,  
    MASP CHAR(4),  
    SL INT,  
    CONSTRAINT pk_cthd PRIMARY KEY (SOHD, MASP)  
);
```

# RÀNG BUỘC TOÀN VỆN TRONG SQL

## Các cách viết ràng buộc (Constraint) đơn giản trong SQL Server

- Cách 3: Thêm ràng buộc sau khi tạo bảng (dùng ALTER TABLE)

- Cú pháp:

**ALTER TABLE** Ten\_Bang **ADD CONSTRAINT** ten\_constraint PRIMARY KEY  
(danh\_sach\_cot);

**ALTER TABLE** Ten\_Bang **ADD CONSTRAINT** ten\_constraint UNIQUE (danh\_sach\_cot);

**ALTER TABLE** Ten\_Bang **ADD CONSTRAINT** ten\_constraint FOREIGN KEY  
(danh\_sach\_cot) **REFERENCES** Ten\_Bang\_Khac(danh\_sach\_cot\_khac);

**ALTER TABLE** Ten\_Bang **ADD CONSTRAINT** ten\_constraint CHECK (bieu\_thuc\_logic);

**ALTER TABLE** Ten\_Bang **ADD CONSTRAINT** ten\_constraint DEFAULT gia\_tri\_mac\_dinh  
**FOR** ten\_cot;

# RÀNG BUỘC TOÀN VỆN TRONG SQL

## Các cách viết ràng buộc (Constraint) đơn giản trong SQL Server

- Cách 3: Thêm ràng buộc sau khi tạo bảng (dùng ALTER TABLE) (tt)
  - Ví dụ:

```
ALTER TABLE HoaDon ADD CONSTRAINT fk_HD_KH FOREIGN KEY (MaKH) REFERENCES  
KhachHang(MaKH);
```

❖ ALTER TABLE HoaDon ADD FOREIGN KEY (MaKH) REFERENCES KhachHang(MaKH);

```
ALTER TABLE SinhVien ADD CONSTRAINT chk_NgaySinh CHECK (NgaySinh >= '1900-01-01');
```

❖ ALTER TABLE SinhVien ADD CHECK (NgaySinh >= '1900-01-01');

➤ Lưu ý: Nên dùng với **Foreign Key**, **Check**. Khóa ngoại ở bảng nào thì sửa bảng đó.

# RÀNG BUỘC TOÀN VỆN TRONG SQL

## Ràng buộc CHECK

- Ràng buộc CHECK được sử dụng để kiểm tra dữ liệu nhập vào bảng phải thỏa mãn một điều kiện nhất định.
- Áp dụng cho:
  - Ràng buộc miền giá trị: Đảm bảo rằng các giá trị của một cột nằm trong một phạm vi hoặc tập hợp giá trị cụ thể.
  - Ràng buộc liên thuộc tính: Đảm bảo rằng một hoặc nhiều cột tuân thủ các điều kiện phức tạp hơn, thường liên quan đến nhiều cột.
- **Cú pháp:**

```
ALTER TABLE Ten_Bang ADD CONSTRAINT ten_constraint CHECK (bieu_thuc_logic);
```

# RÀNG BUỘC TOÀN VỆN TRONG SQL

## Ràng buộc CHECK (tt)

### ▪ Ví dụ:

- Đảm bảo doanh số (DOANH SỐ) trong bảng KHACHHANG luôn lớn hơn hoặc bằng 0:

```
ALTER TABLE KHACHHANG ADD CONSTRAINT CHK_DS CHECK (DOANH SỐ >= 0);
```

- Đảm bảo đơn vị tính (DVT) trong bảng SANPHAM chỉ có các giá trị 'CAY', 'HOP', hoặc 'CAI':

```
ALTER TABLE SANPHAM ADD CONSTRAINT CHK_DVT  
CHECK (DVT = 'CAY' OR DVT = 'HOP' OR DVT = 'CAI');
```



# RÀNG BUỘC TOÀN VẠN TRONG SQL

## Toán tử LIKE

- Toán tử LIKE được sử dụng để kiểm tra dữ liệu kiểu chuỗi (string) dựa trên một mẫu (pattern) nhất định.
- Dấu gạch dưới '\_' : thay thế cho 1 ký tự.
- Dấu phần trăm '%': thay thế cho 0, 1, hoặc n ký tự.
- Ví dụ:

LIKE 'a__'	Chuỗi bắt đầu là a , sau a có 2 ký tự.
LIKE 'a%'	Chuỗi bắt đầu là a , sau a có thể có 0 hoặc n ký tự.
LIKE '_a___'	Ký tự thứ 2 là a, sau a có 3 ký tự

**ALTER TABLE** SINHVIEN **ADD** CHECK (MSSV LIKE '\_\_52\_\_')

## 5

### CÁC LỆNH THAO TÁC DỮ LIỆU

# CÁC LỆNH THAO TÁC DỮ LIỆU

## Các lệnh thao tác dữ liệu Data Manipulation Language (DML)

- DML (Data Manipulation Language) là tập hợp các lệnh SQL được sử dụng để thao tác và quản lý dữ liệu trong cơ sở dữ liệu.
- Các lệnh chính trong DML bao gồm:
  - **INSERT**: Thêm dữ liệu vào bảng.
  - **DELETE**: Xóa dữ liệu khỏi bảng.
  - **UPDATE**: Sửa dữ liệu trong bảng.
  - **SELECT INTO**: Sao chép dữ liệu từ bảng này sang bảng khác.

# CÁC LỆNH THAO TÁC DỮ LIỆU

## Các lệnh thao tác dữ liệu Data Manipulation Language (DML)

### ▪ Thêm mới dữ liệu (INSERT)

- Cú pháp: **INSERT INTO** <Tên bảng> **VALUES** (<Danh sách giá trị>);

➤ Ví dụ 1: Thêm dữ liệu không chỉ định cụ thể các cột

**INSERT INTO** NhanVien **VALUES** ('NV01', 'Nguyen Nhu Nhat', '0927345678', '2006-04-13');

- Hoặc có thể chỉ định cụ thể các cột, với cú pháp:

**INSERT INTO** <Tên bảng> (Tên cột 1, Tên cột 2, Tên cột 3)  
**VALUES** ('Giá trị 1', 'Giá trị 2', Giá trị 3);

➤ Ví dụ 2: Thêm dữ liệu chỉ định cụ thể các cột

**INSERT INTO** NHANVIEN (MANV, HOTEN, SODT, NGVL) **VALUES** ('NV01',  
'Nguyen Nhu Nhat', '0927345678', '2006-04-13');

# CÁC LỆNH THAO TÁC DỮ LIỆU

## Các lệnh thao tác dữ liệu Data Manipulation Language (DML)

- Xóa dữ liệu (DELETE)

- Cú pháp:

```
DELETE FROM <Tên bảng>  
WHERE <Điều kiện>;
```

**Lưu ý:** Nếu không có điều kiện (WHERE), tất cả các hàng trong bảng sẽ bị xóa.

- Ví dụ 1: Xóa một dòng dữ liệu cụ thể

```
DELETE FROM NhanVien WHERE MaNV = 'NV01';
```

- Ví dụ 2: Xóa các dòng dữ liệu theo một mẫu nhất định

```
DELETE FROM SinhVien WHERE MSSV LIKE '2352____';
```

# CÁC LỆNH THAO TÁC DỮ LIỆU

## Các lệnh thao tác dữ liệu Data Manipulation Language (DML)

- Cập nhật dữ liệu (UPDATE)

- Cú pháp:

```
UPDATE <Tên bảng>
```

```
SET <Tên cột 1> = <Giá trị mới>, <Tên cột 2> = <Giá trị mới>
```

```
WHERE <Điều kiện>;
```

- Ví dụ: Cập nhật tên và số điện thoại của nhân viên

```
UPDATE NHANVIEN
```

```
SET HoTen = 'Nguyen Van Tien', SoDT = '083568711'
```

```
WHERE MaNV = 'NV02';
```

# CÁC LỆNH THAO TÁC DỮ LIỆU

## Các lệnh thao tác dữ liệu Data Manipulation Language (DML)

- Sao chép dữ liệu (SELECT INTO)

- Cú pháp:

```
SELECT * INTO <Tên bảng mới>  
FROM <Tên bảng cũ>;
```

- Ví dụ: Sao chép toàn bộ dữ liệu từ bảng KHACHHANG sang bảng mới KHACHHANG\_NEW

```
SELECT * INTO KHACHHANG_NEW  
FROM KHACHHANG;
```

*Toàn bộ dữ liệu từ bảng cũ sẽ được sao chép sang bảng mới. Nếu chỉ muốn sao chép một phần dữ liệu, có thể sử dụng điều kiện **WHERE** để lọc dữ liệu.*

## 6

### BÀI TẬP THỰC HÀNH VÀ HỎI ĐÁP



# BÀI TẬP THỰC HÀNH VÀ HỎI ĐÁP

## Yêu cầu 1: (Xem thêm đề bài tại file Bài tập thực hành tuần 1)

Phần bài làm lưu trữ trên một file script có tên **<MSSV>\_<HoVaTen>\_BTTH1\_YC1.sql** (trong đó **MSSV** là mã số sinh viên, **HoVaTen** là họ và tên của sinh viên).

Sử dụng các câu lệnh SQL trong công cụ SQL Server Management Studio để thực hiện yêu cầu sau:

**Câu 1.** Tạo cơ sở dữ liệu đặt tên **TrungTam\_TDTT**, gồm có 6 quan hệ **PHONGTAP**, **HUANLUYENVIENT**, **HOCVIEN**, **LOPTAP**, **DANGKY**, **LICHTAP**.

**Câu 2.** Tạo khóa chính, khóa ngoại cho các quan hệ đó.

**Câu 3.** Nhập dữ liệu cho 6 quan hệ theo như dữ liệu đề bài đã cung cấp.

# BÀI TẬP THỰC HÀNH VÀ HỎI ĐÁP

## Yêu cầu 2: (Xem thêm đề bài tại file Bài tập thực hành tuần 1)

Phần bài làm lưu trữ trên một file script có tên **<MSSV>\_<HoVaTen>\_BTTH1\_YC2.sql** (trong đó MSSV là mã số sinh viên, HoVaTen là họ và tên của sinh viên).

Sử dụng các câu lệnh SQL trong công cụ **SQL Server Management Studio** để thực hiện yêu cầu.



# HỎI ĐÁP



*Liên hệ hỗ trợ*

**Lê Võ Đình Kha - [khalvd@uit.edu.vn](mailto:khalvd@uit.edu.vn)**