**Widmarck Francois**

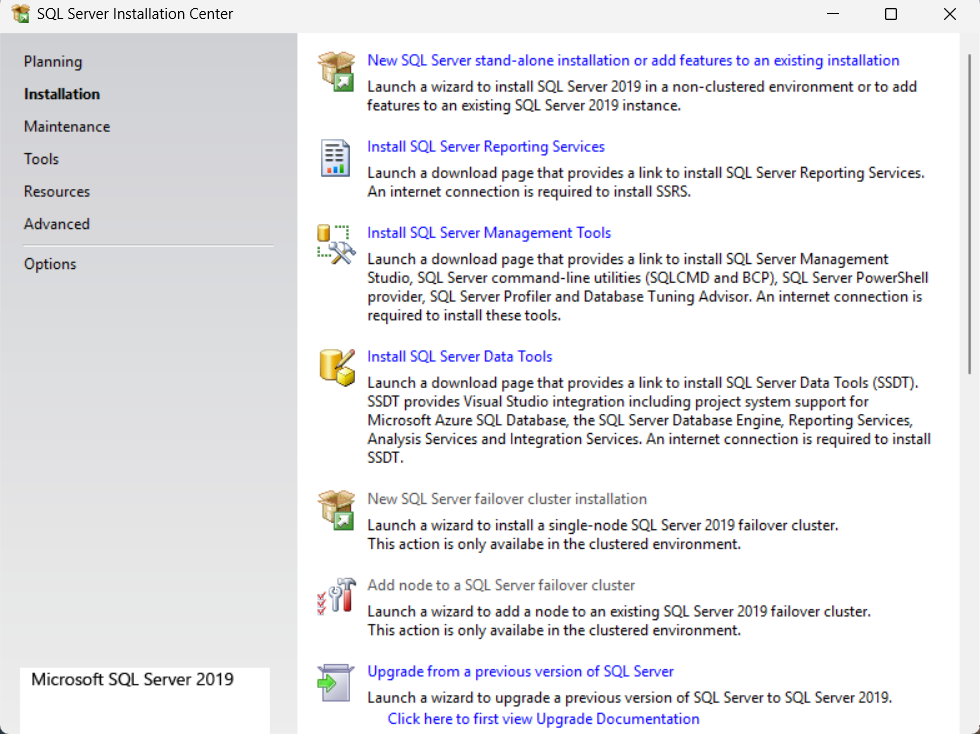
**Class NWIT 291**

**Professor Lemma**

**27 July 2025**

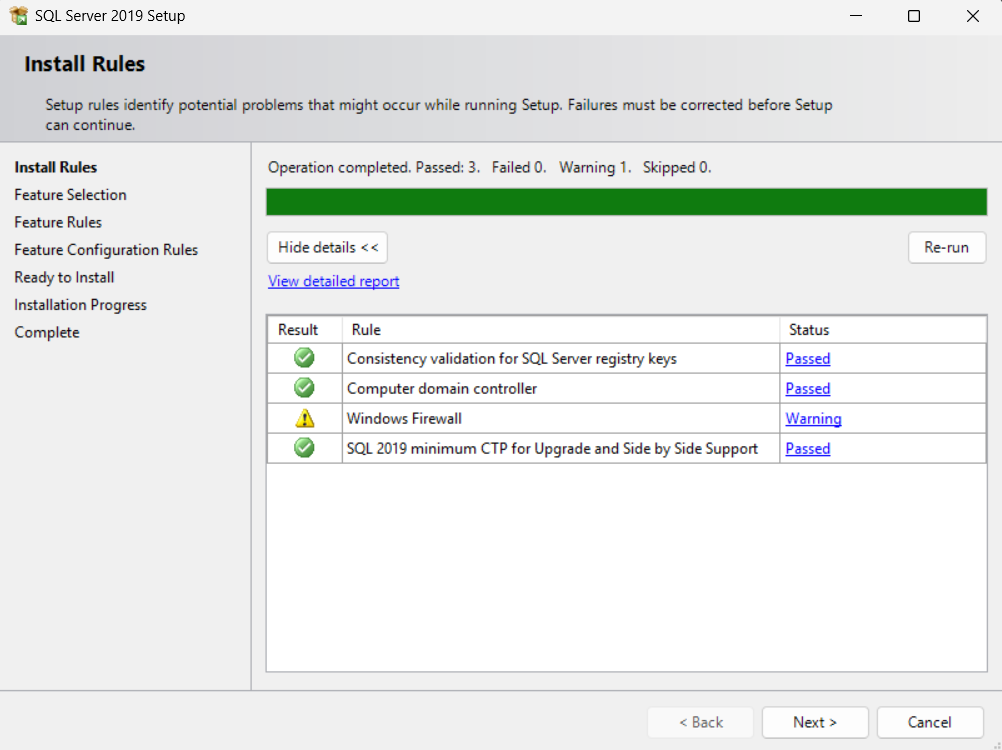
**Step 1: Installing SSMS SQL Server**

I started by downloading the SSMS installer file (**SSMS-setup-ENU.exe**) and ran it. During the setup, I chose Custom and clicked Install to get started.

On the left side menu, I clicked on Installation, then selected New SQL Server stand-alone installation, and clicked Next to continue.

**Step 2: Install Rules**

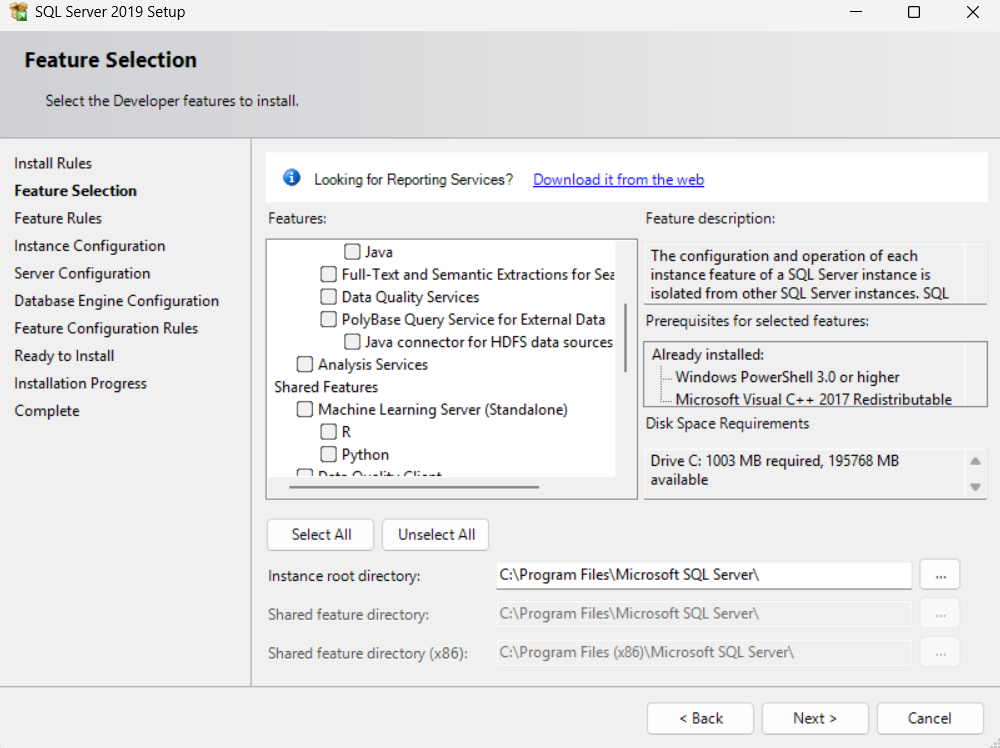
Once I got to the install rules screen, SQL Server ran a quick check to see if anything might cause problems.



There was one warning, but nothing major, so I clicked **Next** to move forward.

**Step 3: Features Selections**

Here I had several options to choose from, but I only needed the core features.  
So I selected **Database Engine Services** — this is the main service for running SQL databases — and clicked **Next.**



At this point, there was also an option to name the server, which I could personalize if needed.

**Step 4: Instance Configuration**

In this step, I gave the instance a custom name and ID.A screenshot of a computer

AI-generated content may be incorrect.

This name becomes part of the installation path, which helps later when accessing or backing up the server.

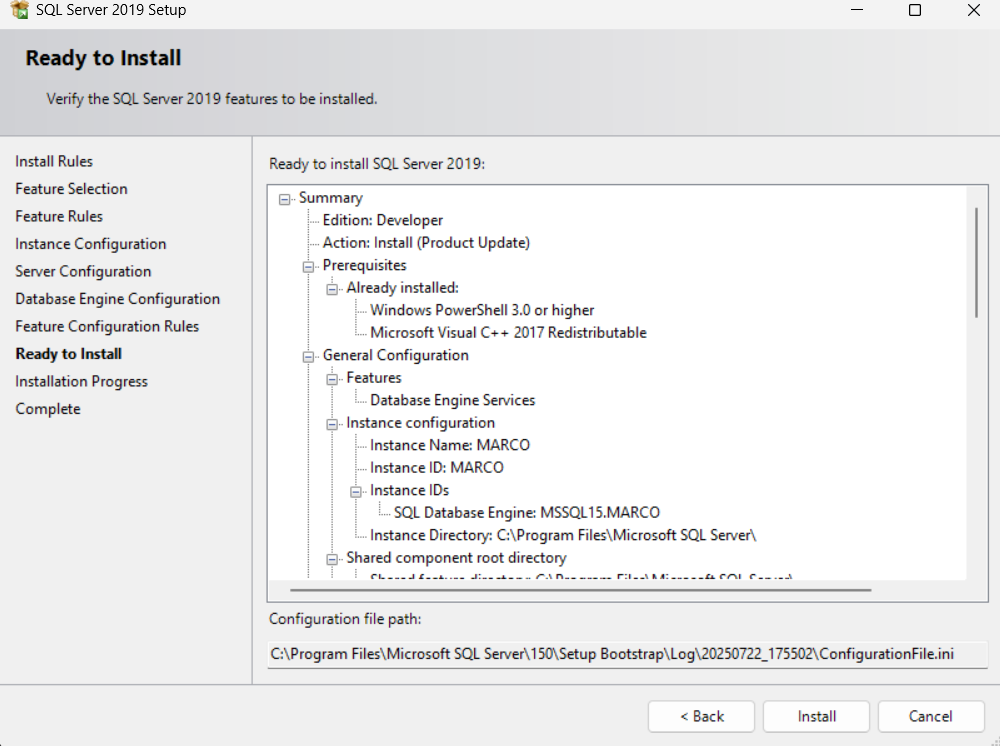
**Step 5: Configuring the Database Engine**

On the next screen, I added myself as a SQL Server administrator.  
This just means I’ll have full access to the server and can manage everything without restrictions.A screenshot of a computer

AI-generated content may be incorrect.

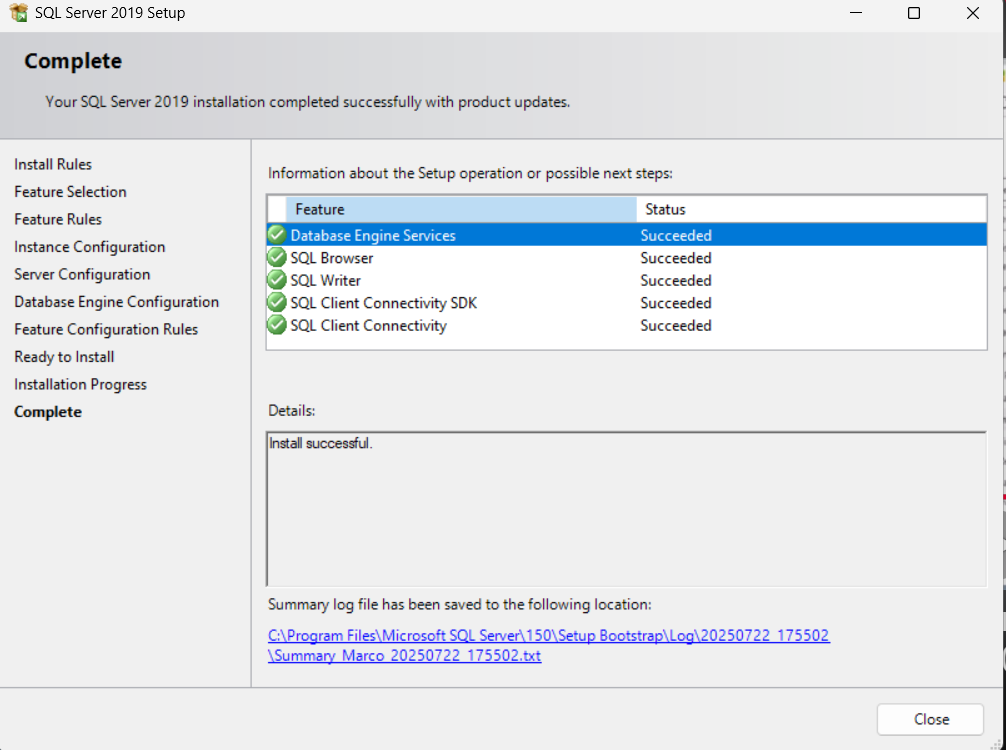
**Step 6: Final Setup**

I reached the “Ready to Install” screen, which gave a summary of all the features that were about to be installed.  
After reviewing, I clicked **Install** and let it run.

****

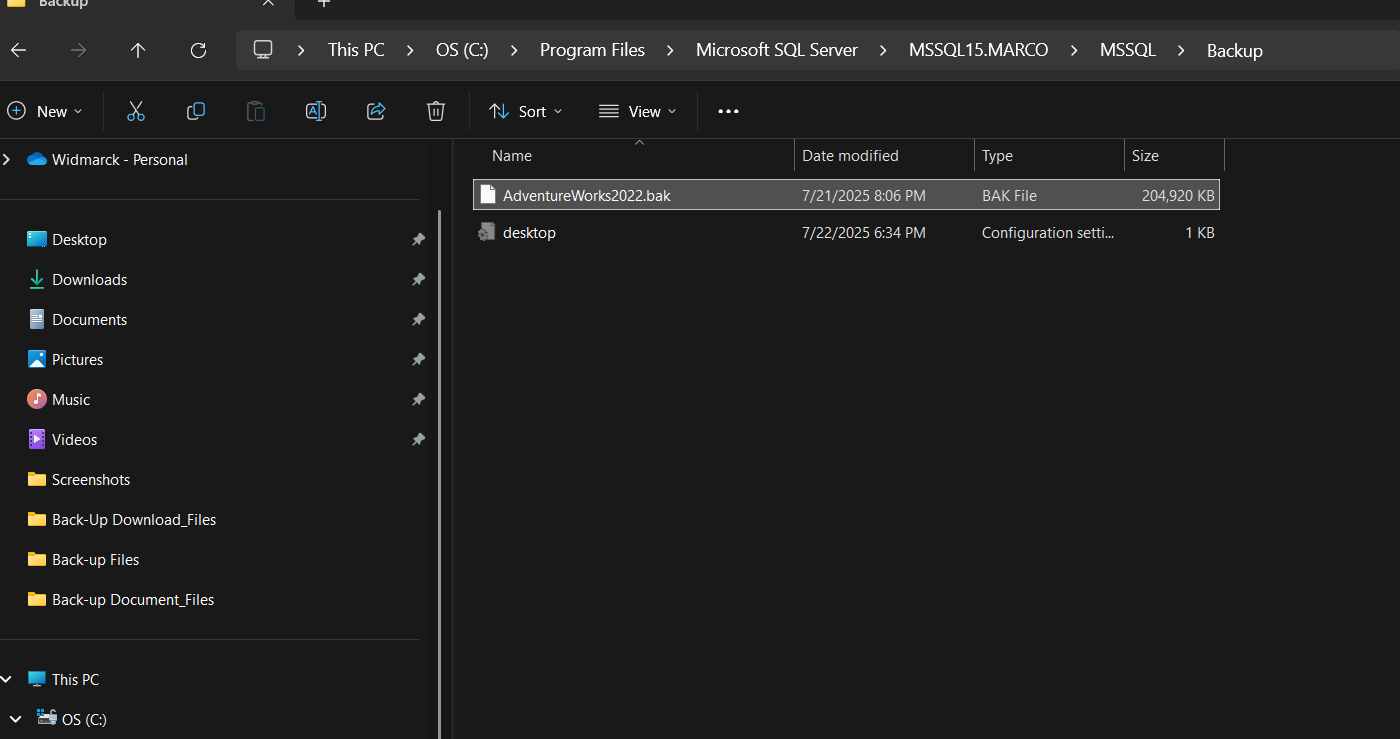
**Step 7: Installation Complete**

Once the installation finished, SQL Server was fully set up and ready to use.

****

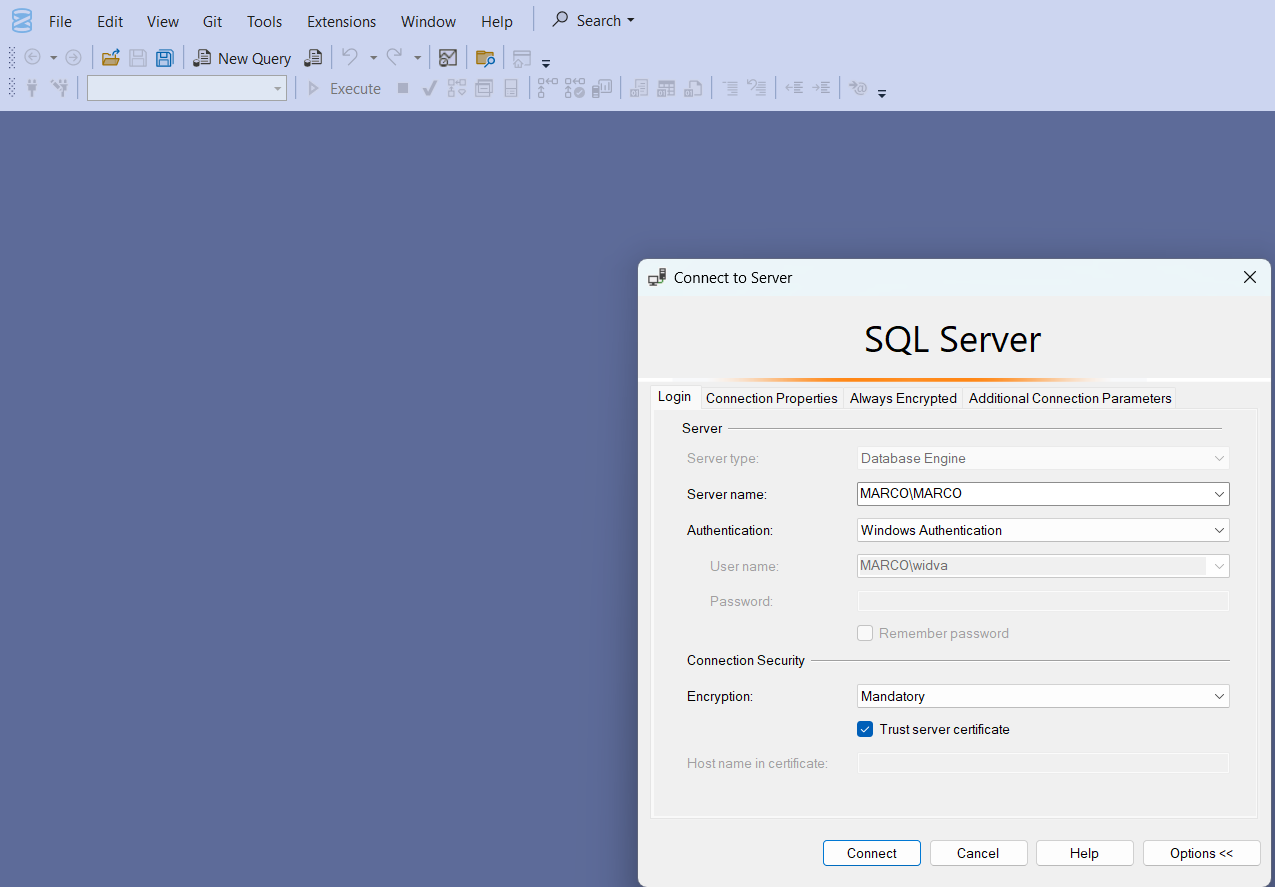
**Step 8: Moving the AdventureWorks2019 Backup**

Next, I downloaded the **AdventureWorks2019** database backup file.  
I moved it into the backup directory for my SQL Server instance:  
C:\Program Files\Microsoft SQL Server\MSSQL15.MARCO\MSSQL\Backup  
(In my case, the folder was named "MARCO" based on my instance name.)



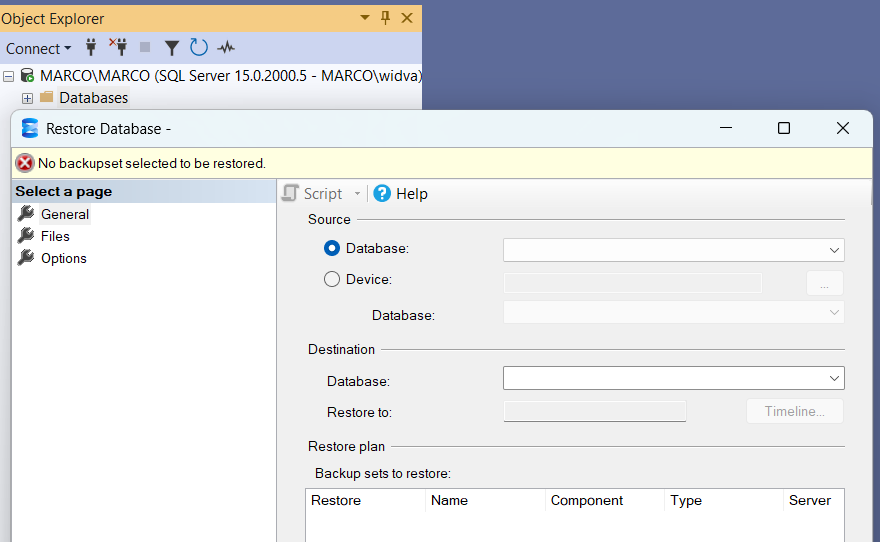
**Step 9: Logging into SSMS**

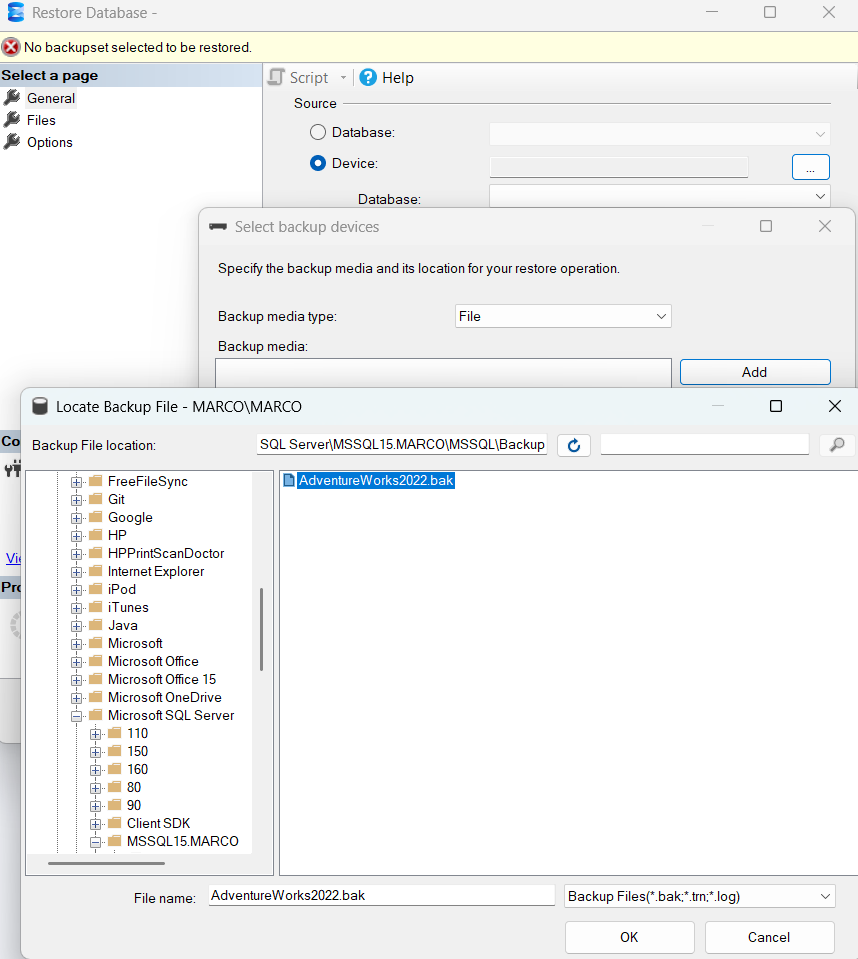
I opened SQL Server Management Studio (SSMS) and logged in to the server using the instance I had created earlier.

**** From here, I was ready to start restoring the AdventureWorks database

**Step 10: Starting the Restore**

I right-clicked on Databases, chose Restore Database, and selected the Device option to browse for my .bak file.  
I selected the AdventureWorks2019 backup file and clicked OK to start the restore.

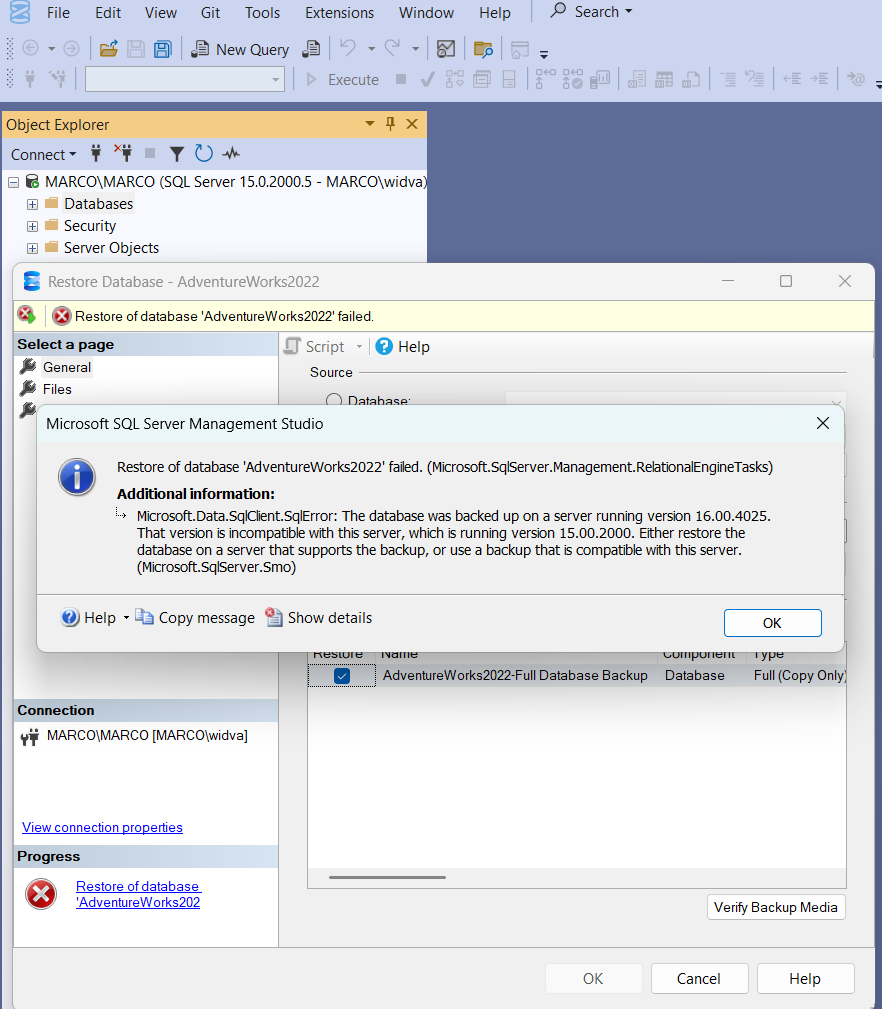
****

****

I selected the AdventureWorks2019 backup file and clicked OK to start the restore.

**Step 11: First Problem – Compatibility**

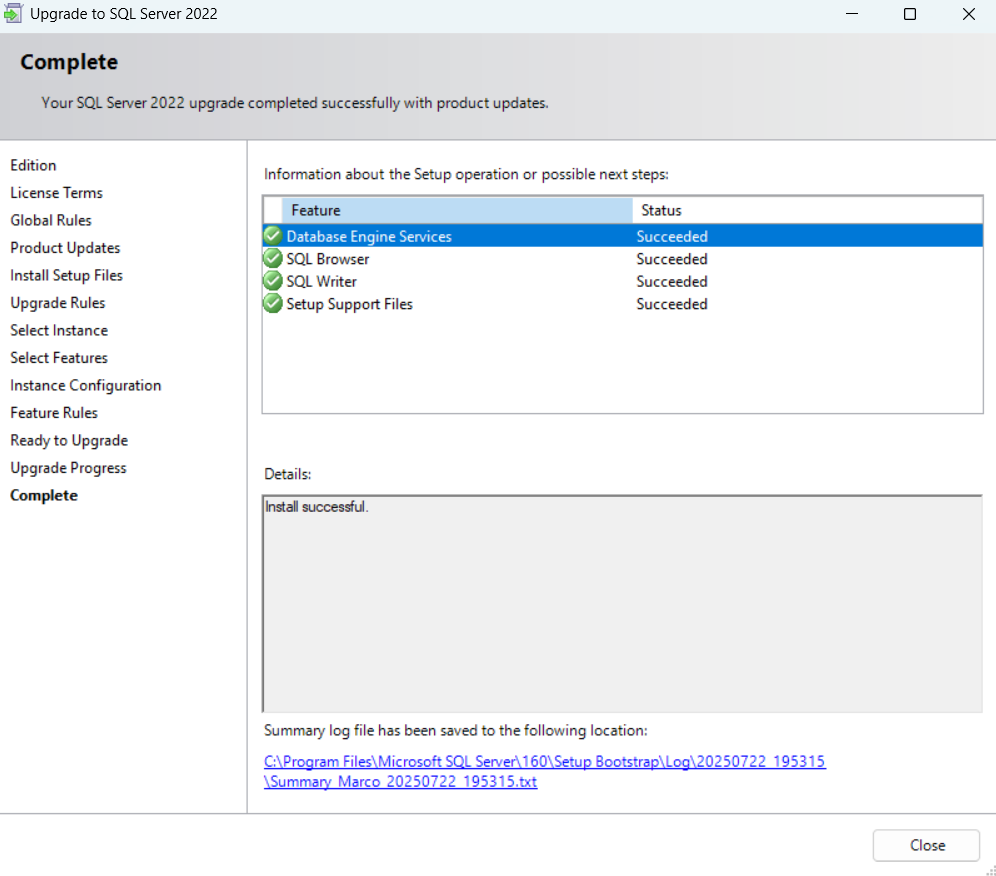
Unfortunately, the restore didn’t work.  
It turned out the version of SQL Server I had installed (2019) wasn’t compatible with the version of the backup file I was using.

****

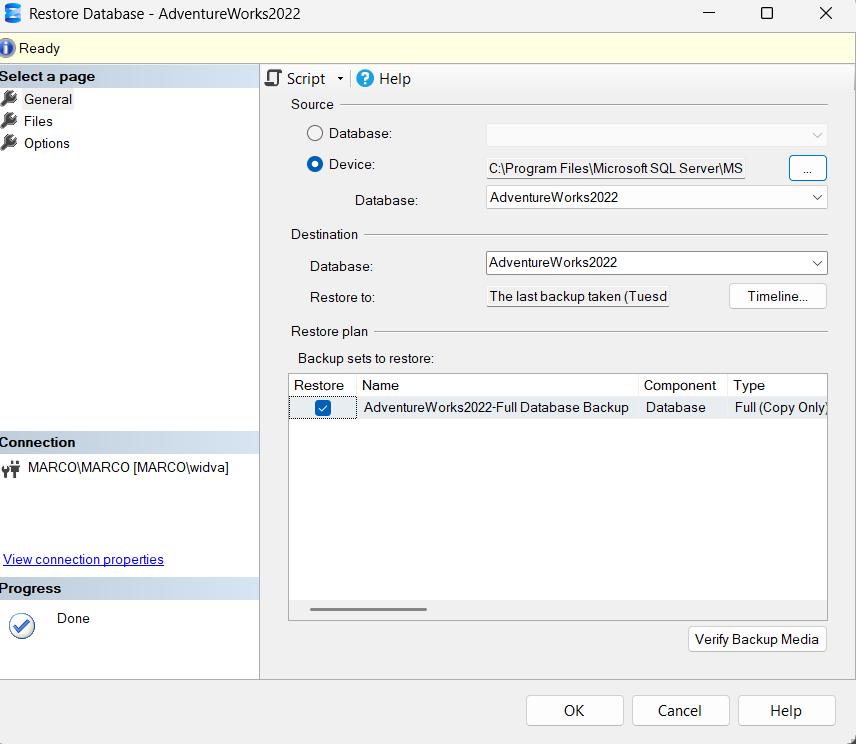
So I had to uninstall SQL Server 2019 and install **SQL Server 2022** instead. The upgradable process was successful, and below was the result.

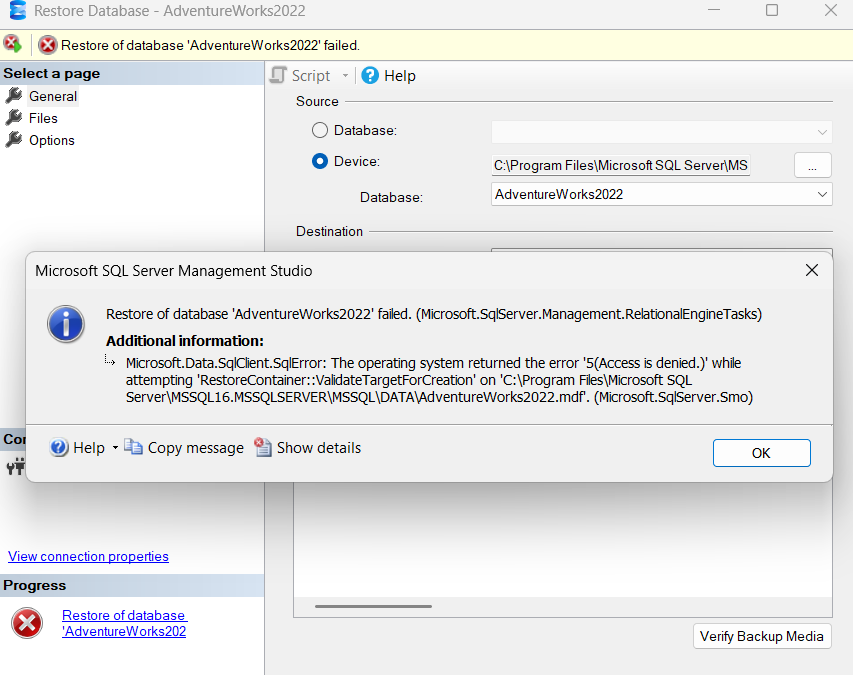
**Step 12: Upgrading to SQL Server 2022**

The upgrade went smoothly.  
Once I finished installing SQL Server 2022, I reopened SSMS and got ready to try the restore process again.

****

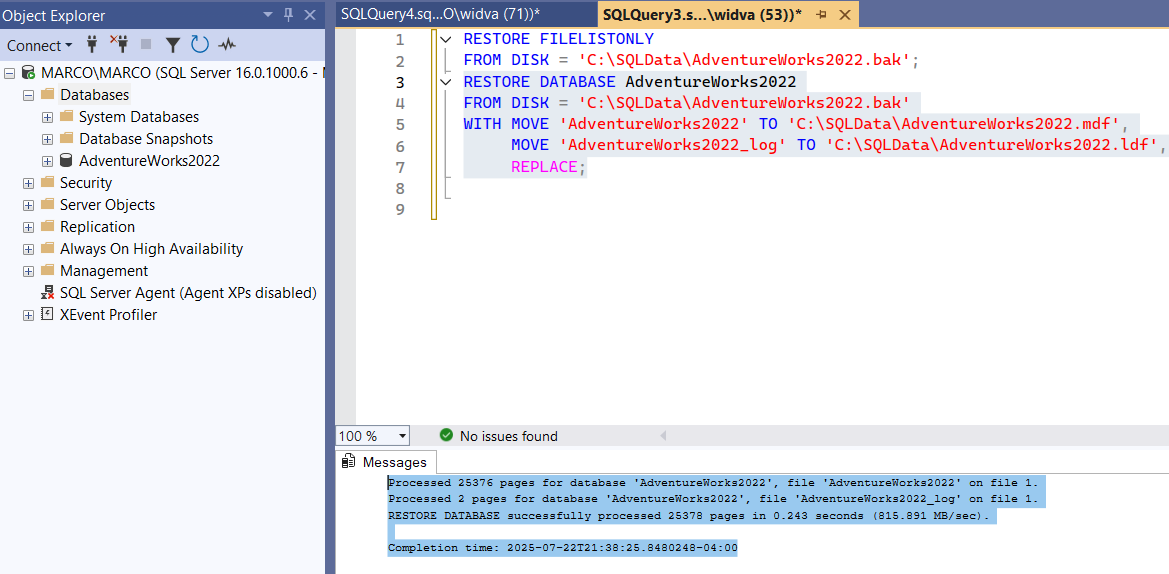
**Step 13: Second Problem – Restore Error**

I went back through the same steps to restore the database, but this time I got an error related to the backup folder.****

****  
To fix it, I created a new folder called SQLDATA directly on my C:\ drive and moved the .bak file there.

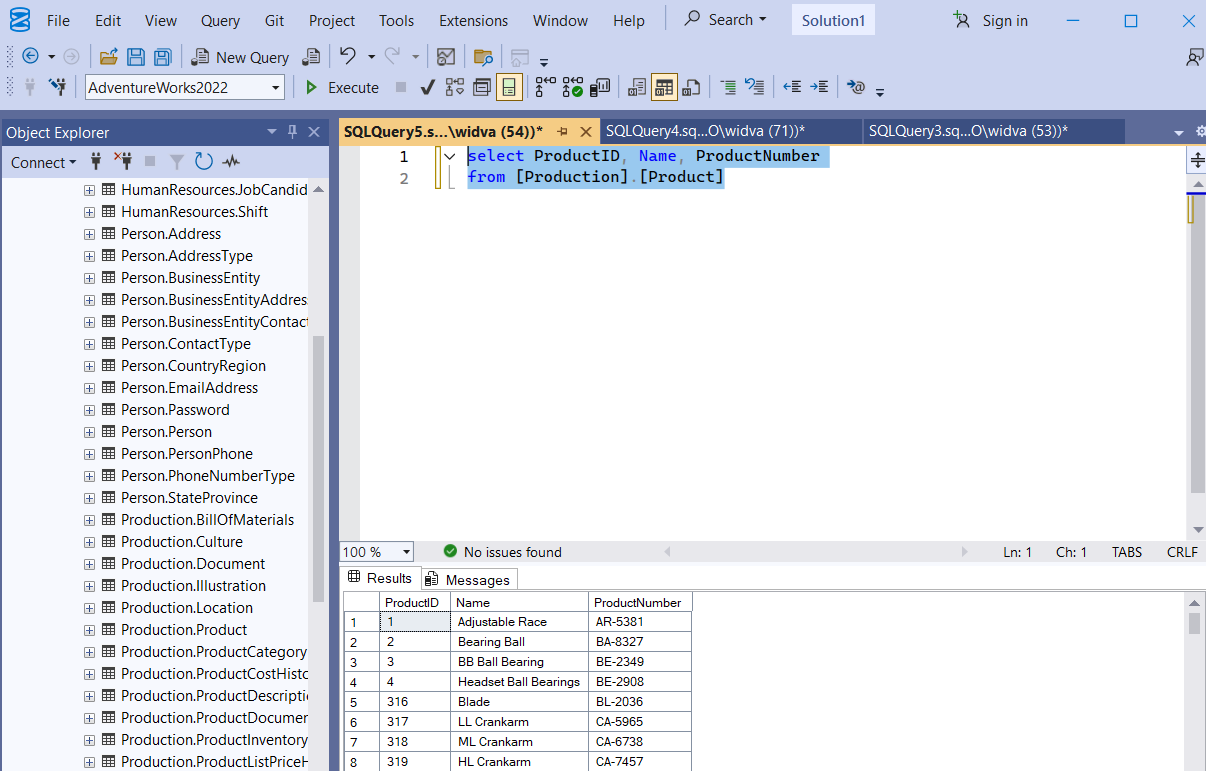
**Step 14: Successful Restore**

After changing the backup path, the restore finally worked!

I could see AdventureWorks2022 listed in my databases on the left-hand side in SSMS.

To double-check that it was working, I opened a new query window and ran a quick test.

**Step 15: Testing the Database**

I ran this simple SELECT statement to pull some data from the Production.Product table:

The query worked, and I saw the results right away — everything was now up and running.

**Lab 10b – Database Security: Audit Log and Alert System**

**Screenshot 1: Checking the Value of OrderID**

After running the update code (lines 61–66), I checked the row for OrderID 10248. The new values show that the OrderDate was successfully changed to 2025-04-12, and the Freight was updated to 0.99, as expected.**A screenshot of a computer

AI-generated content may be incorrect.**

**Screenshot 2: Verifying That the Data Was Changed**

Looking at the table again, I confirmed the original values were updated. The OrderDate went from 2023-04-02 to 2025-04-12, and the Freight changed from 179.99 to 0.99. This confirms that the update worked correctly.A screenshot of a computer

AI-generated content may be incorrect.

**Screenshot 3: Identifying the Current User**

I tried to identify which user made the changes using the SYSTEM\_USER command. Instead of a username, the result came back as NULL. So, it didn’t show the expected user information this time.

A screenshot of a computer

AI-generated content may be incorrect.

**Screenshot 4: Updating the Orders Table**

After running the UPDATE statement on the Orders table, I got a message from the procedure that says:

"The Order table in Northwind DB was modified!"

This shows that the trigger worked and successfully called the procedure to raise the alert.**A screenshot of a computer

AI-generated content may be incorrect.**

**Screenshot 5: Verifying That the Change Was Logged**

Finally, I ran a query to check the AuditLog table. It showed a new row that logged the event. The table now contains the AuditID, EventType, EventMessage, and EventTime, confirming that the update to the Orders table was tracked properly.

**A screenshot of a computer

AI-generated content may be incorrect.**