Sprawozdanie z przedmiotu Administrowanie internetowymi serwerami baz danych

Laboratorium № 9

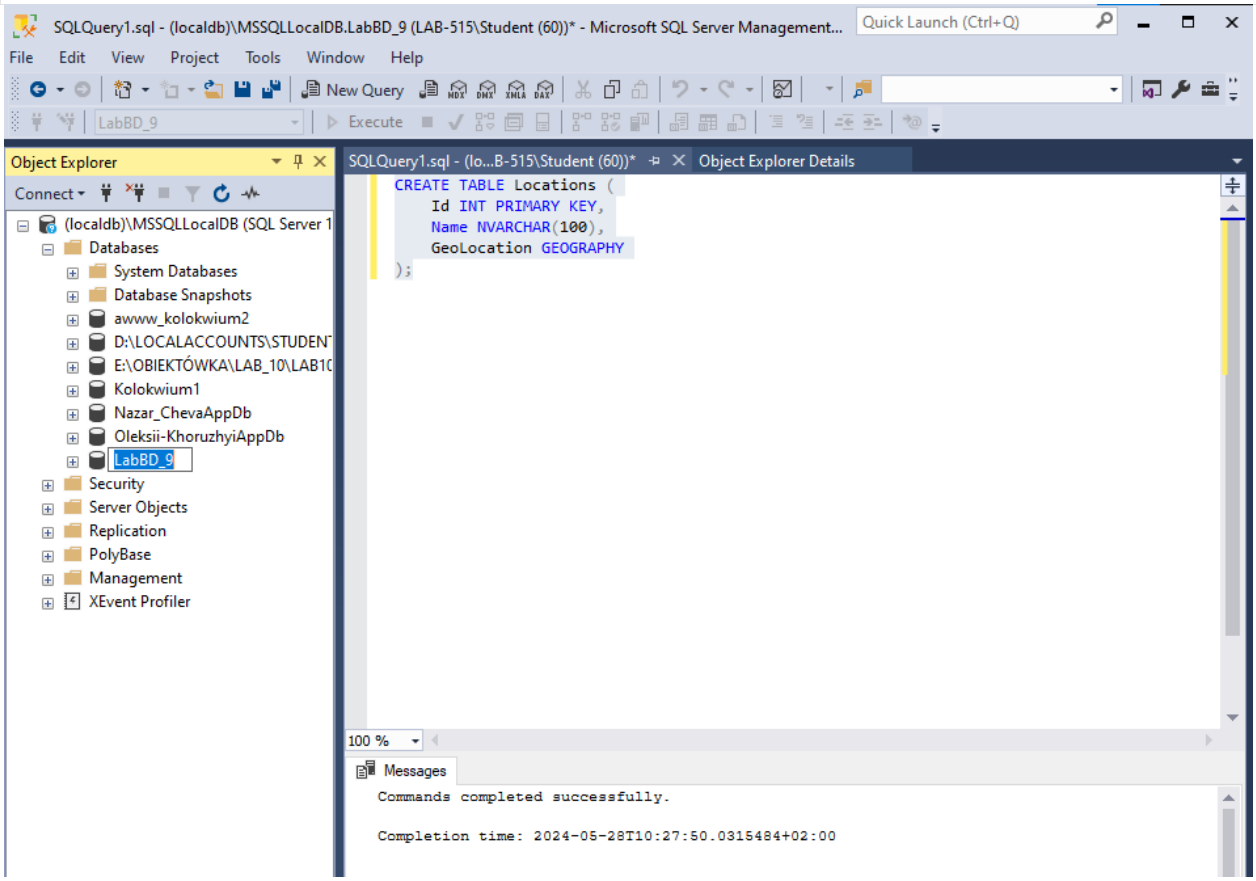
Khoruzhyi Oleksii

grupa nr.1

PAI WIMiI

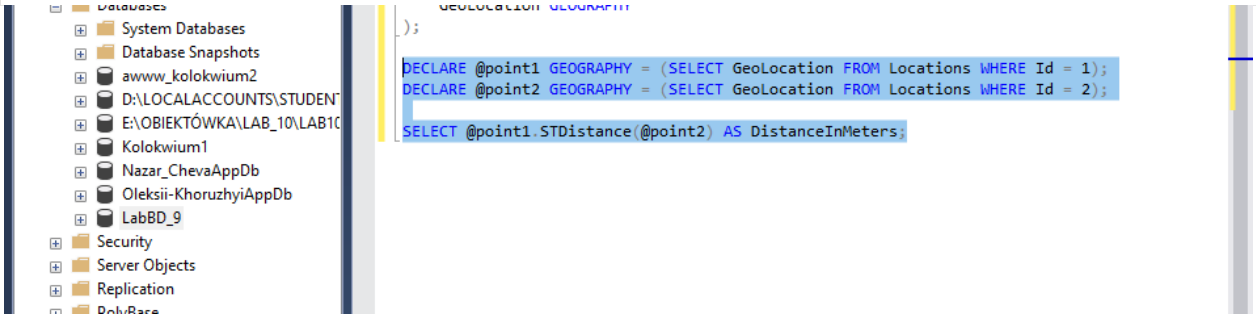
1. Utworzenie tabeli zawierającej pola typu geography

Stworzyliśmy tabelę, która zawiera pola typu **geography**, umożliwiające przechowywanie danych geograficznych takich jak punkty, linie czy poligony.



2. Obliczenie odległości pomiędzy punktami przy pomocy zapytania

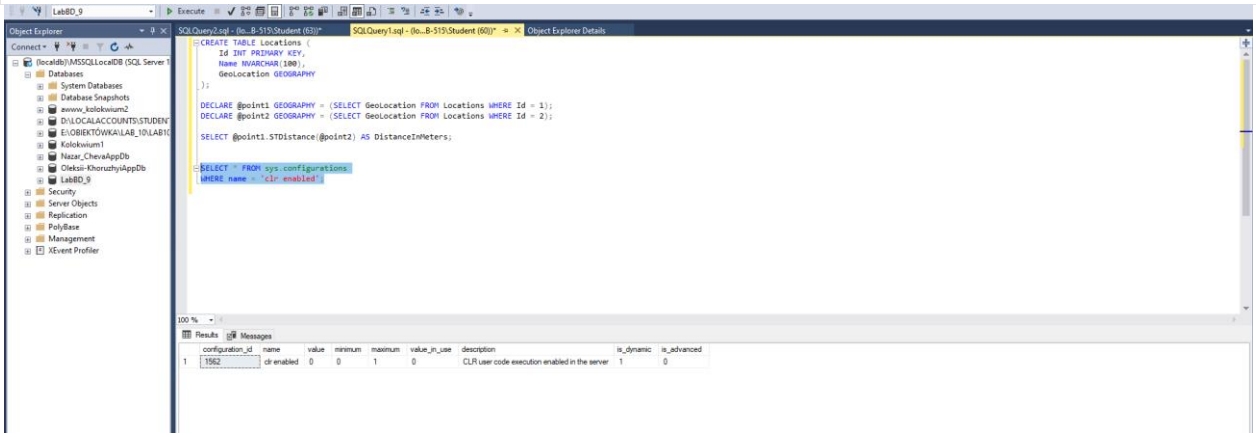
W tabeli **Locations** przechowujemy współrzędne geograficzne. Aby obliczyć odległość między dwoma punktami, możemy użyć metody **STDistance**.



3. Włączenie obsługi CLR na serwerze bazy danych

a) Sprawdzenie czy włączono obsługę CLR

Aby sprawdzić, czy obsługa CLR jest włączona, wykonaliśmy następujące zapytanie:



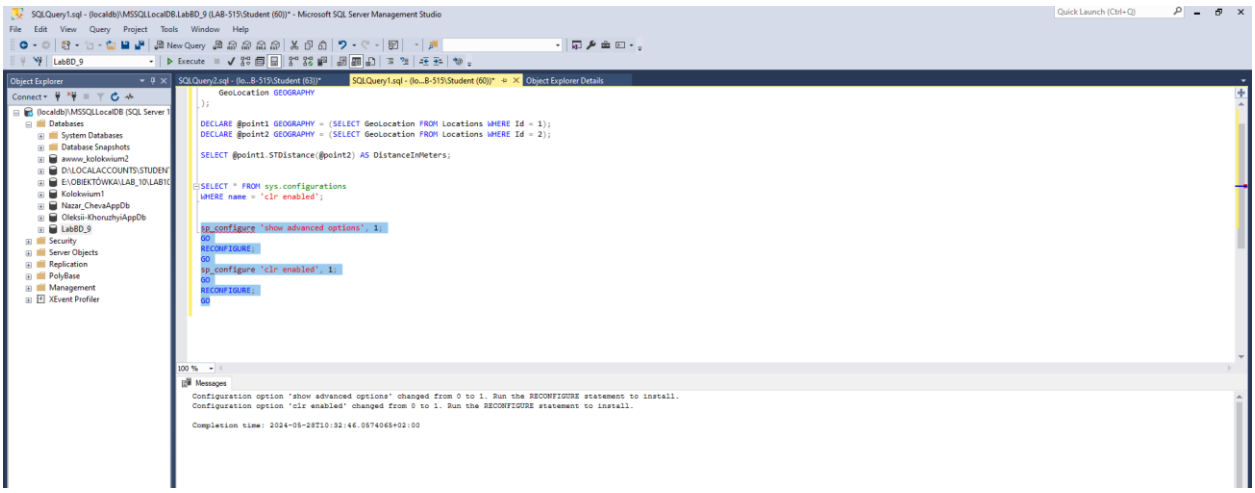
b) Opis poszczególnych pól otrzymanych w wyniku powyższego zapytania

Wynik zapytania zawiera następujące pola:

- **configuration_id**: Unikalny identyfikator ustawienia konfiguracji.
- **name**: Nazwa ustawienia konfiguracji.
- **value**: Aktualna wartość ustawienia (0 - wyłączone, 1 - włączone).
- **minimum**: Minimalna możliwa wartość ustawienia.
- **maximum**: Maksymalna możliwa wartość ustawienia.
- **value_in_use**: Aktualnie używana wartość ustawienia (może być różna od **value**, jeżeli wymagana jest ponowna konfiguracja serwera).
- **description**: Opis ustawienia konfiguracji.
- **is_dynamic**: Informacja, czy zmiana wartości ustawienia wymaga restartu serwera (1 - nie wymaga, 0 - wymaga).
- **is_advanced**: Informacja, czy ustawienie jest zaawansowane (1 - tak, 0 - nie).

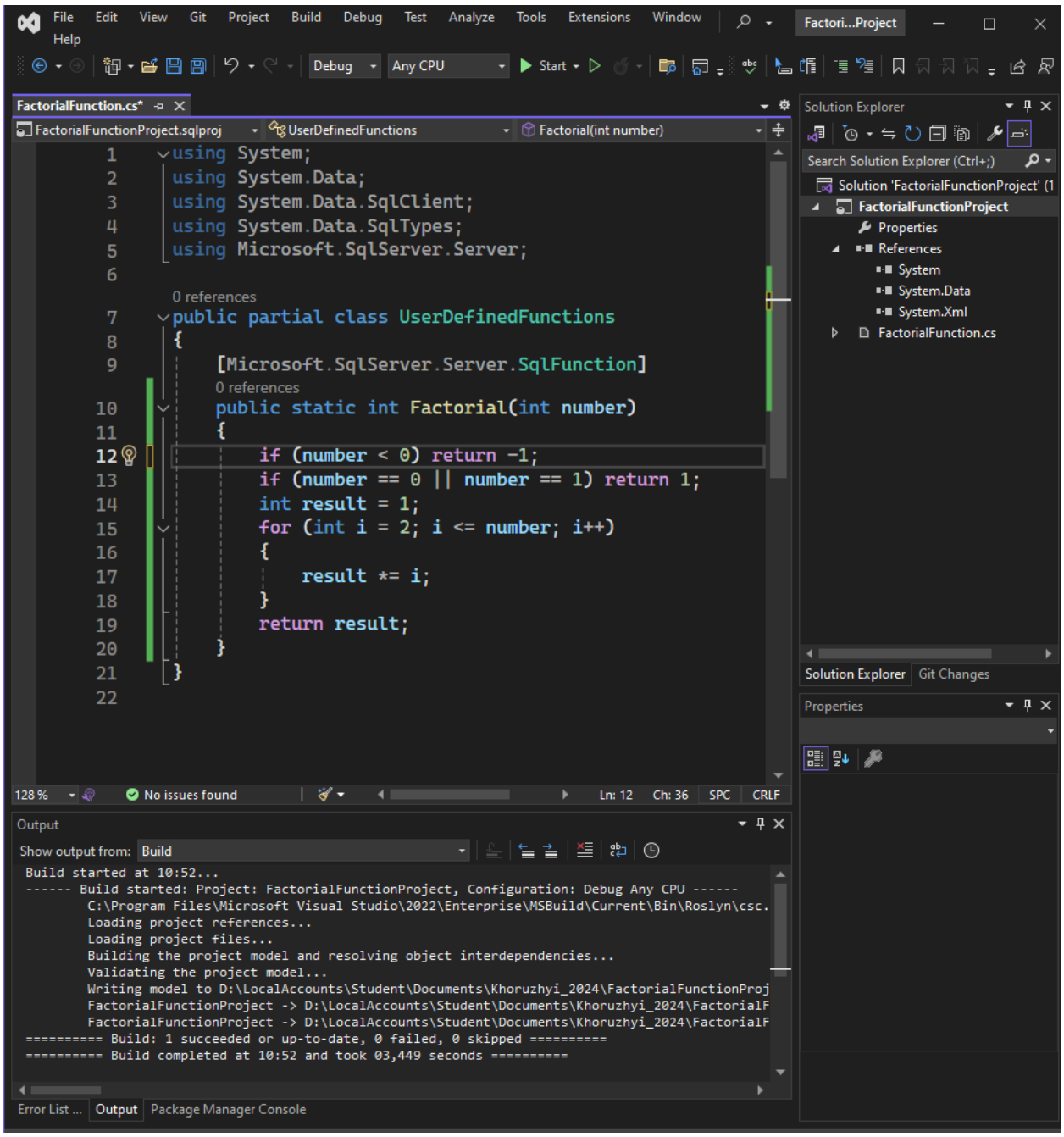
c) Włączenie obsługi CLR

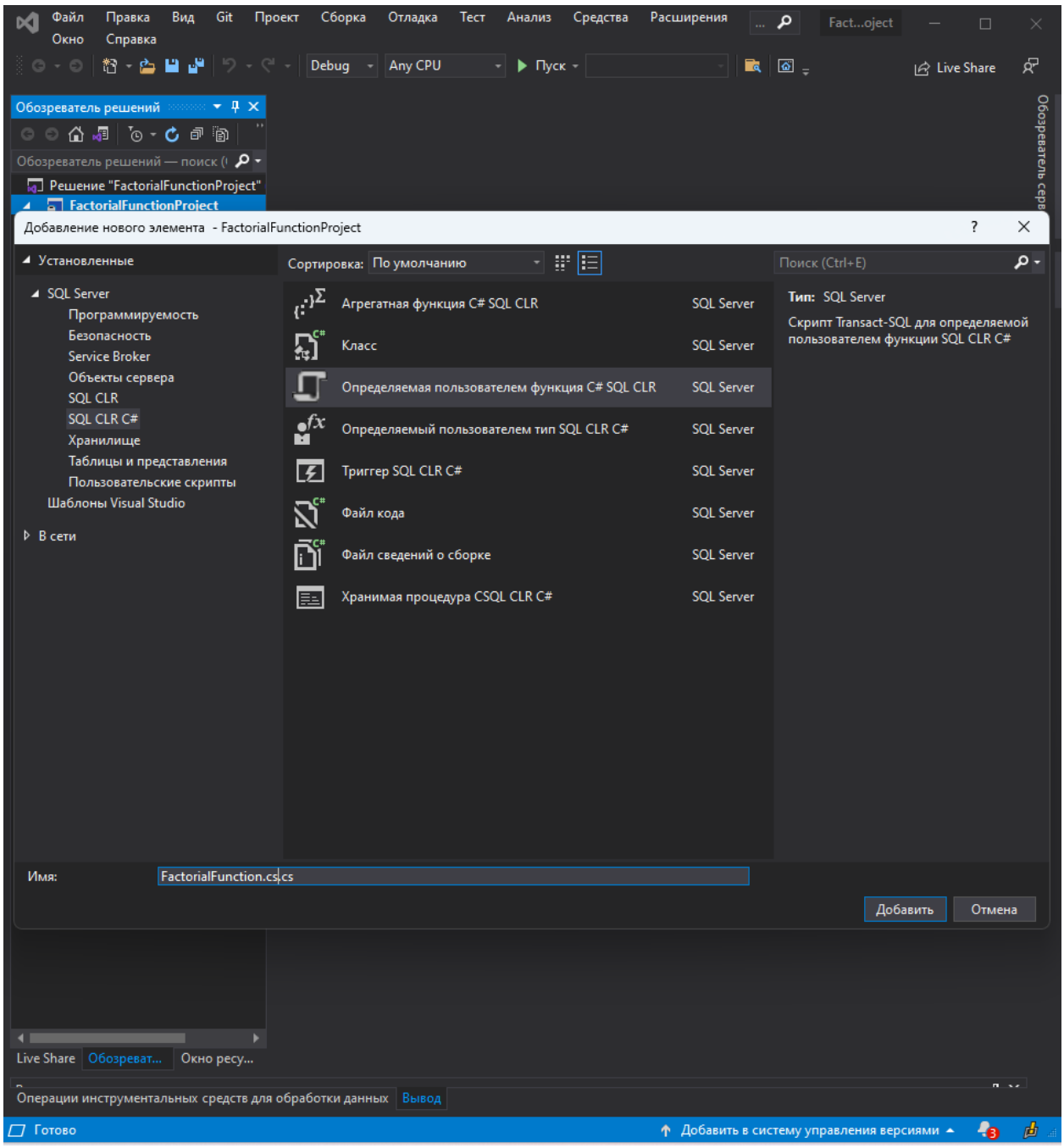
Aby włączyć obsługę CLR, wykonaliśmy poniższe komendy:

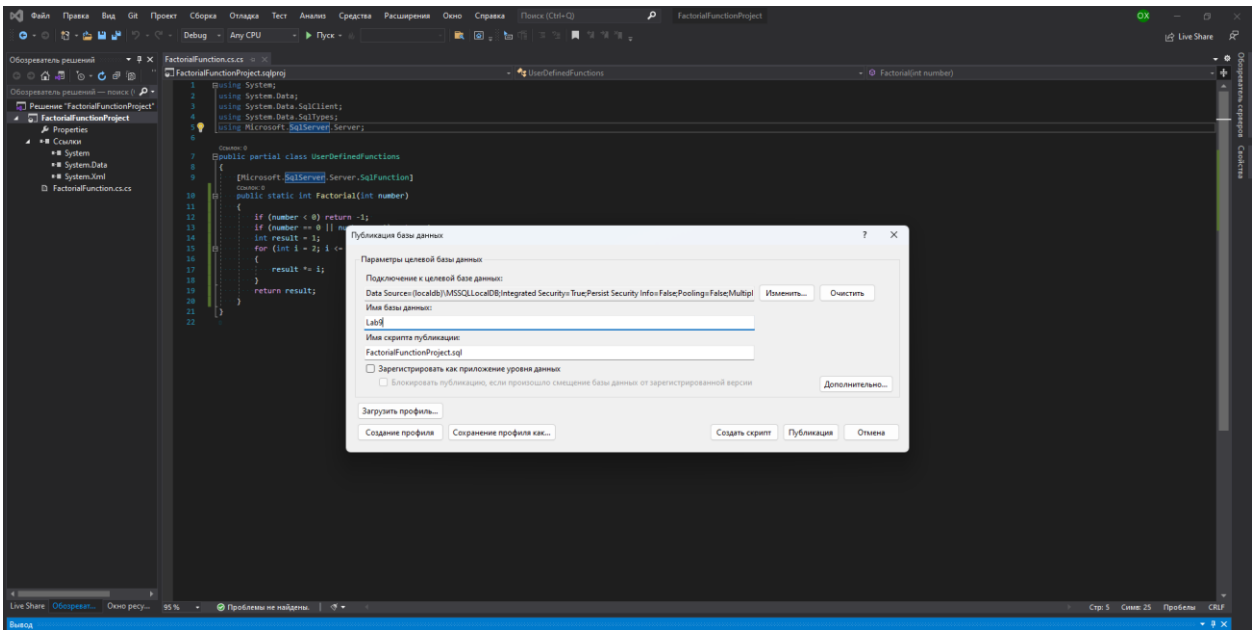


d) Stworzenie w Visual Studio nowego projektu SQL Server Database Project i dodanie własnej funkcji CLR

W Visual Studio stworzyliśmy nowy projekt SQL Server Database Project. Dodaliśmy do niego funkcję CLR obliczającą silnię.

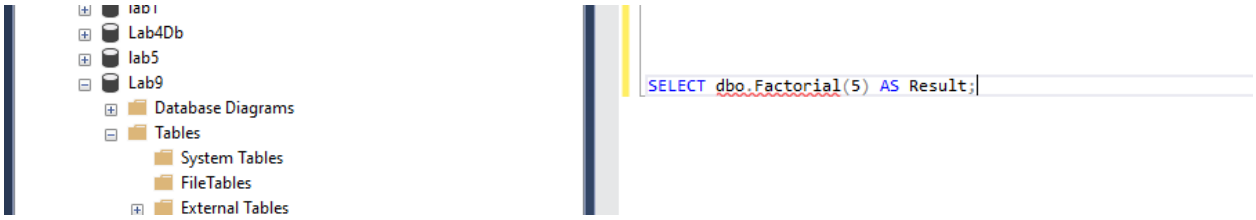

File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help
Factori...Project
Debug
Any CPU
Start
FactorialFunction.cs*
FactorialFunctionProject.sqlproj
UserDefinedFunctions
Factorial(int number)
using System;
using System.Data;
using System.Data.SqlClient;
using System.Data.SqlTypes;
using Microsoft.SqlServer.Server;
0 references
public partial class UserDefinedFunctions
{
[Microsoft.SqlServer.Server.SqlFunction]
0 references
public static int Factorial(int number)
{
if (number < 0) return -1;
if (number == 0 || number == 1) return 1;
int result = 1;
for (int i = 2; i <= number; i++)
{
result *= i;
}
return result;
}
}
Solution Explorer
Search Solution Explorer (Ctrl+;)
Solution 'FactorialFunctionProject' (1
FactorialFunctionProject
Properties
References
System
System.Data
System.Xml
FactorialFunction.cs
Solution Explorer
Git Changes
Properties
128 %
No issues found
Ln: 12
Ch: 36
SPC
CRLF
Output
Show output from: Build
Build started at 10:52...
------ Build started: Project: FactorialFunctionProject, Configuration: Debug Any CPU ------
C:\Program Files\Microsoft Visual Studio\2022\Enterprise\MSBuild\Current\Bin\Roslyn\csc.
Loading project references...
Loading project files...
Building the project model and resolving object interdependencies...
Validating the project model...
Writing model to D:\LocalAccounts\Student\Documents\Khoruzhyi_2024\FactorialFunctionProj
FactorialFunctionProject -> D:\LocalAccounts\Student\Documents\Khoruzhyi_2024\FactorialF
FactorialFunctionProject -> D:\LocalAccounts\Student\Documents\Khoruzhyi_2024\FactorialF
========== Build: 1 succeeded or up-to-date, 0 failed, 0 skipped ==========
========== Build completed at 10:52 and took 03,449 seconds ==========
Error List ...
Output
Package Manager Console


Файл Правка Вид Git Проект Сборка Отладка Тест Анализ Средства Расширения Окно Справка
Fact...oject
Debug
Any CPU
Пуск
Live Share
Обозреватель решений
Обозреватель решений — поиск (
Решение "FactorialFunctionProject"
FactorialFunctionProject
Добавление нового элемента - FactorialFunctionProject
Установленные
Сортировка:
По умолчанию
Поиск (Ctrl+E)
SQL Server
Программируемость
Безопасность
Service Broker
Объекты сервера
SQL CLR
SQL CLR C#
Хранилище
Таблицы и представления
Пользовательские скрипты
Шаблоны Visual Studio
В сети
Агрегатная функция C# SQL CLR SQL Server
Класс SQL Server
Определяемая пользователем функция C# SQL CLR SQL Server
Определяемый пользователем тип SQL CLR C# SQL Server
Триггер SQL CLR C# SQL Server
Файл кода SQL Server
Файл сведений о сборке SQL Server
Хранимая процедура CSQL CLR C# SQL Server
Тип: SQL Server
Скрипт Transact-SQL для определяемой пользователем функции SQL CLR C#
Имя:
FactorialFunction.cs
Добавить
Отмена
Live Share
Обозреват...
Окно ресу...
Операции инструментальных средств для обработки данных
Вывод
Готово
Добавить в систему управления версиями



f) Zapytanie z uwzględnieniem stworzonej funkcji

Aby użyć stworzonej funkcji CLR w zapytaniu SQL, wykonaliśmy poniższe zapytanie:



Result - 120

Podsumowanie

W sprawozdaniu pokazaliśmy proces tworzenia tabeli z polem typu **geography**, obliczanie odległości pomiędzy punktami, włączenie obsługi CLR na serwerze SQL, tworzenie i publikowanie funkcji CLR oraz użycie tej funkcji w zapytaniu SQL. Dzięki włączeniu CLR możemy rozszerzyć funkcjonalność SQL Server, dodając własne funkcje napisane w językach .NET, co pozwala na bardziej zaawansowane operacje i lepszą integrację z aplikacjami.