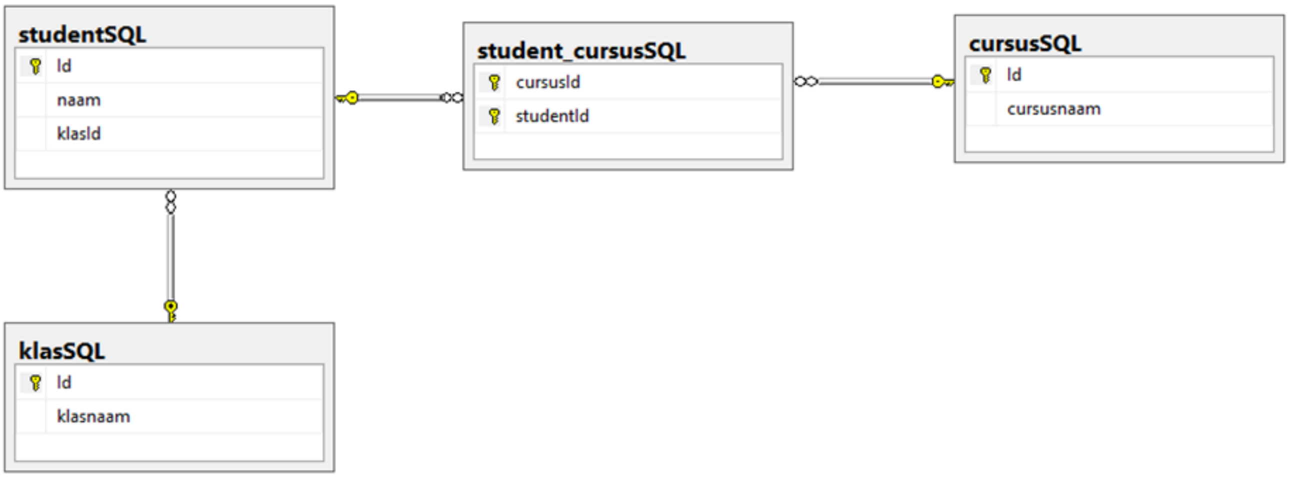
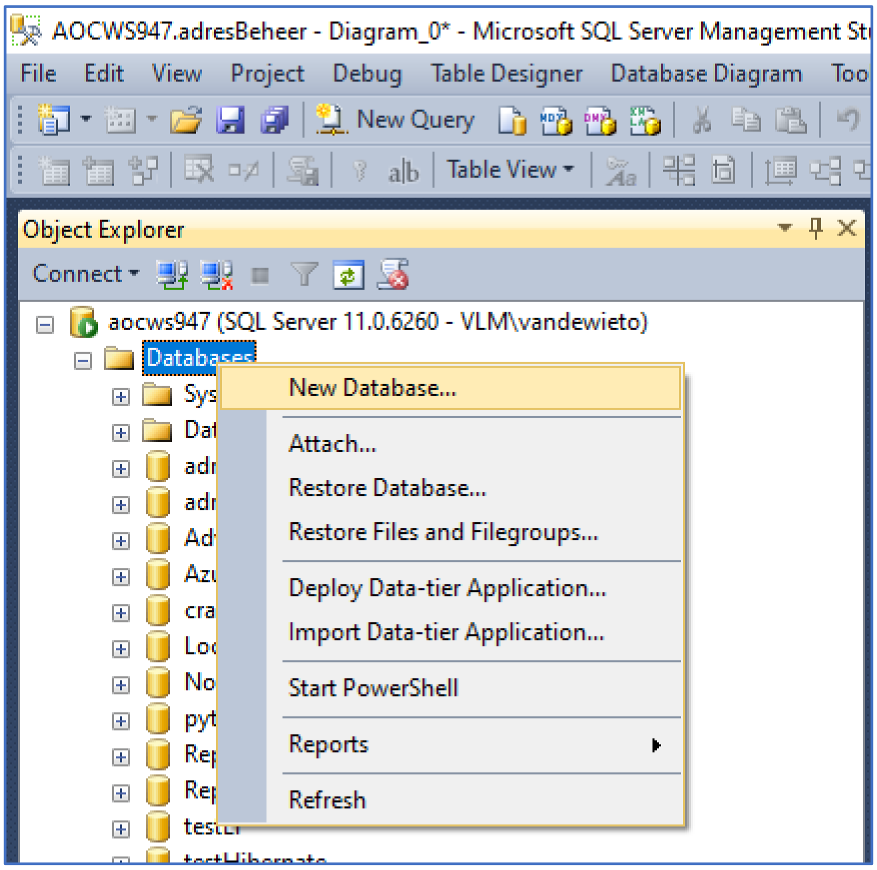
ADO.NET

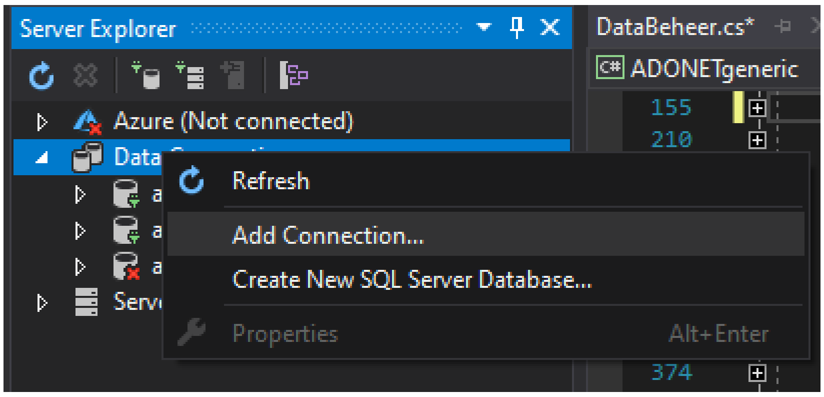
1.1 Data model

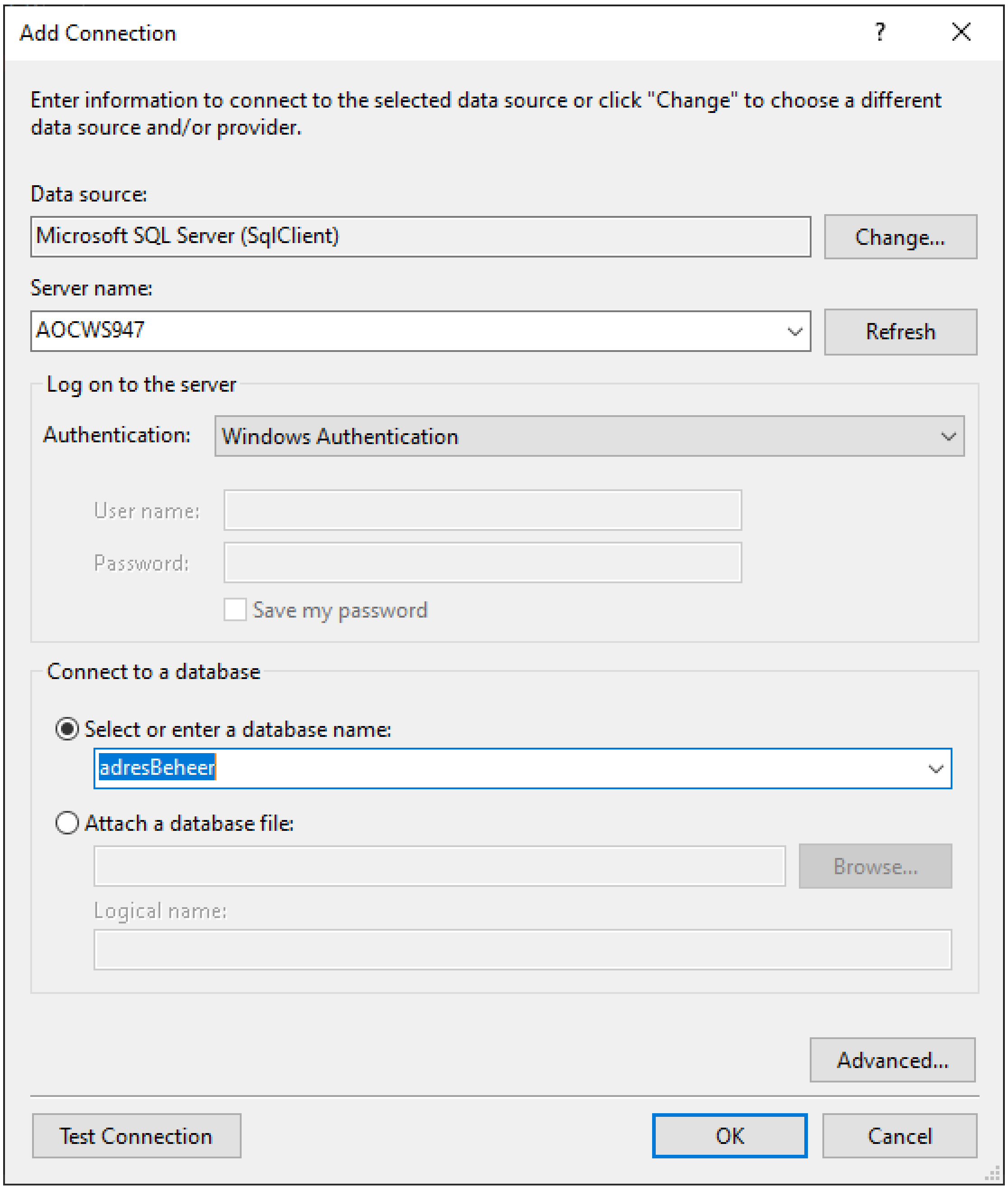


Stap 1 – aanmaken databank (Microsoft SQL Server Management Studio)


AOCWS947.adresBeheer - Diagram_0* - Microsoft SQL Server Management St
File
Edit
View
Project
Debug
Table Designer
Database Diagram
Too
New Query
Table View
Object Explorer
Connect
aocws947 (SQL Server 11.0.6260 - VLM\vandewieto)
Databases
New Database...
Attach...
Restore Database...
Restore Files and Filegroups...
Deploy Data-tier Application...
Import Data-tier Application...
Start PowerShell
Reports
Refresh

Stap 2 – Voeg connectie toe naar databank




Add Connection

Enter information to connect to the selected data source or click "Change" to choose a different data source and/or provider.

Data source: Microsoft SQL Server (SqlClient)

Server name: AOCWS947

Log on to the server

Authentication: Windows Authentication

User name:

Password:

Save my password

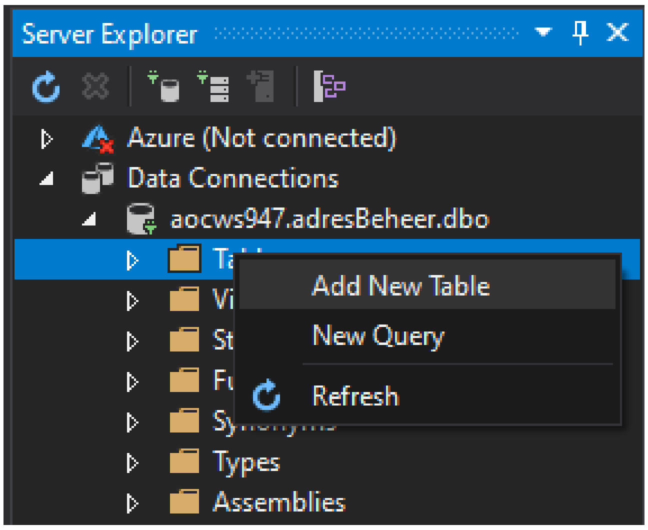
Connect to a database

(●) Select or enter a database name: adresBeheer

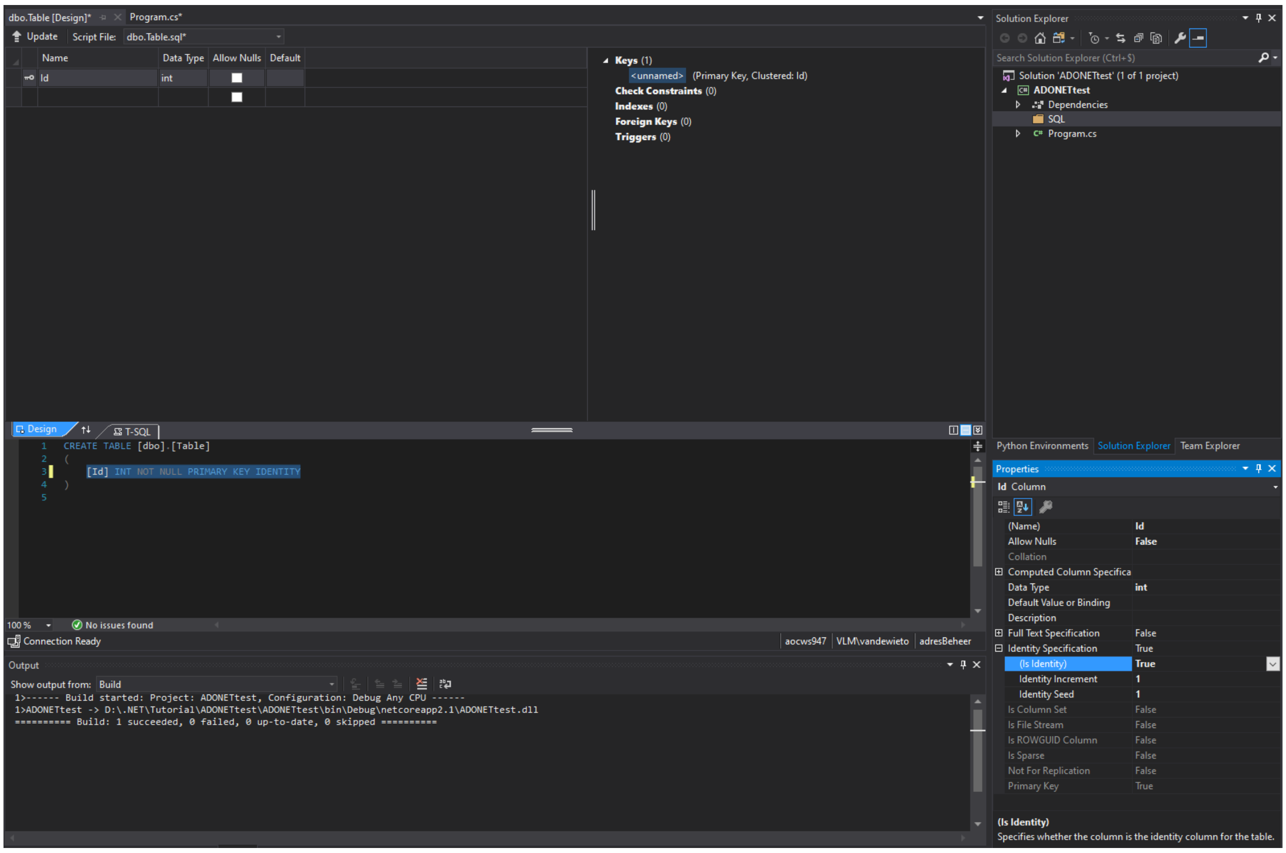
( ) Attach a database file:

Logical name:

Stap 3 – design tabellen



Opmerking identity column !



Name	Data Type	Allow Nulls	Default
Id	int	☐	
		☐	

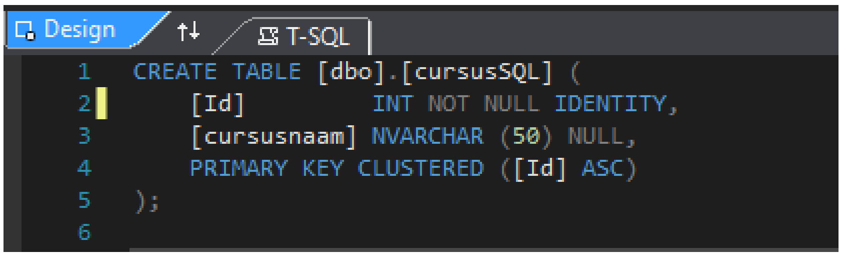
```
CREATE TABLE [dbo].[Table]
(
    [Id] INT NOT NULL PRIMARY KEY IDENTITY
)
```

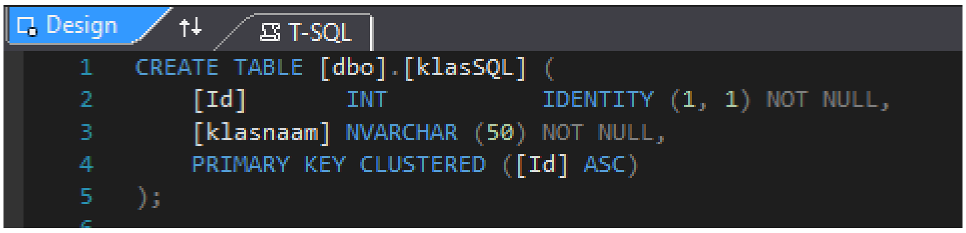
```
1>------ Build started: Project: ADONETtest, Configuration: Debug Any CPU ------
1>ADONETtest -> D:\.NET\Tutorial\ADONETtest\ADONETtest\bin\Debug\netcoreapp2.1\ADONETtest.dll
========== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped ==========
```

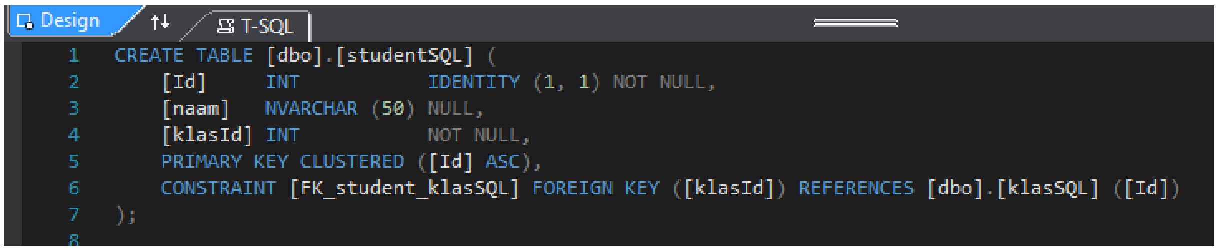
(Name)	Id
Allow Nulls	False
Collation	
Computed Column Specifica	
Data Type	int
Default Value or Binding	
Description	
Full Text Specification	False
Identity Specification	True
(Is Identity)	True
Identity Increment	1
Identity Seed	1
Is Column Set	False
Is File Stream	False
Is ROWGUID Column	False
Is Sparse	False
Not For Replication	False
Primary Key	True

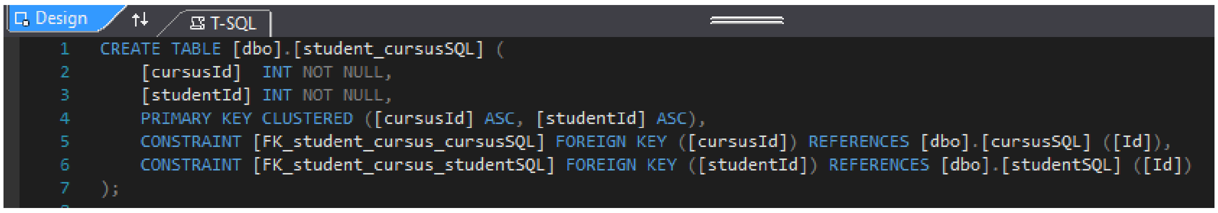
(Is Identity)
Specifies whether the column is the identity column for the table.

SQL statements

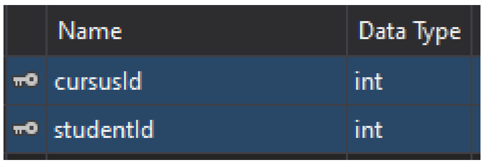

Design T-SQL
```
CREATE TABLE [dbo].[cursusSQL] (
    [Id]         INT NOT NULL IDENTITY,
    [cursusnaam] NVARCHAR (50) NULL,
    PRIMARY KEY CLUSTERED ([Id] ASC)
);
```


Design T-SQL
```
CREATE TABLE [dbo].[klasSQL] (
    [Id]       INT           IDENTITY (1, 1) NOT NULL,
    [klasnaam] NVARCHAR (50) NOT NULL,
    PRIMARY KEY CLUSTERED ([Id] ASC)
);
```

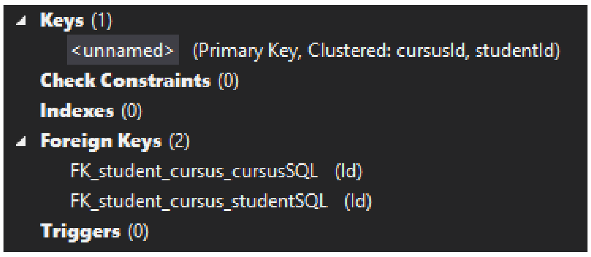

Design T-SQL
```
CREATE TABLE [dbo].[studentSQL] (
    [Id]     INT           IDENTITY (1, 1) NOT NULL,
    [naam]   NVARCHAR (50) NULL,
    [klasId] INT           NOT NULL,
    PRIMARY KEY CLUSTERED ([Id] ASC),
    CONSTRAINT [FK_student_klasSQL] FOREIGN KEY ([klasId]) REFERENCES [dbo].[klasSQL] ([Id])
);
```

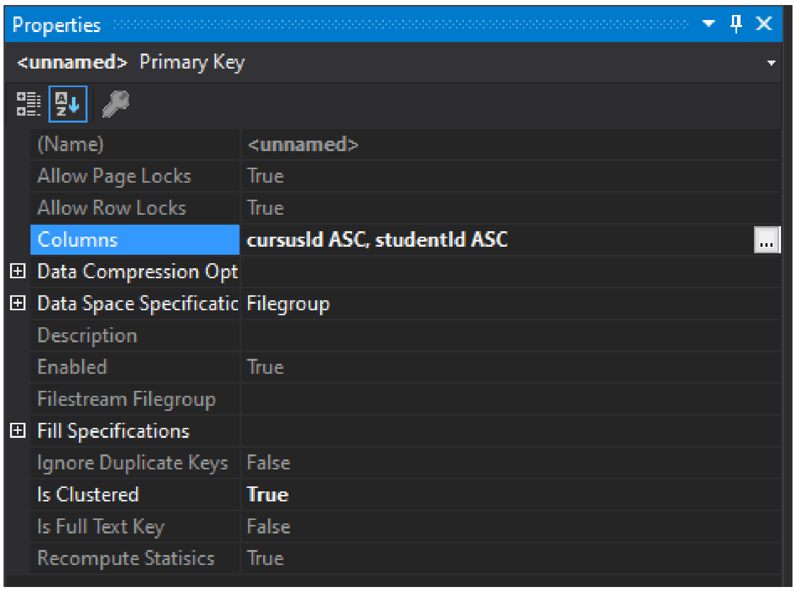

Design T-SQL
```
CREATE TABLE [dbo].[student_cursusSQL] (
    [cursusId]  INT NOT NULL,
    [studentId] INT NOT NULL,
    PRIMARY KEY CLUSTERED ([cursusId] ASC, [studentId] ASC),
    CONSTRAINT [FK_student_cursus_cursusSQL] FOREIGN KEY ([cursusId]) REFERENCES [dbo].[cursusSQL] ([Id]),
    CONSTRAINT [FK_student_cursus_studentSQL] FOREIGN KEY ([studentId]) REFERENCES [dbo].[studentSQL] ([Id])
);
```

Opmerking : composite Key



	Name	Data Type
🔑	cursusId	int
🔑	studentId	int

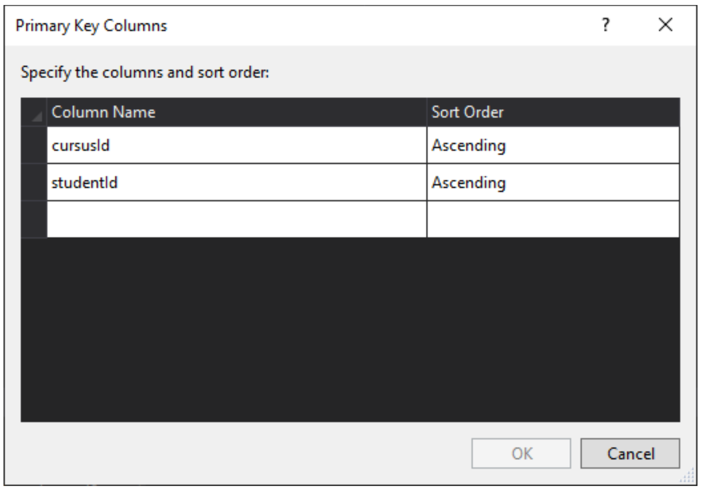

```
◢ Keys (1)
      <unnamed>  (Primary Key, Clustered: cursusId, studentId)
  Check Constraints (0)
  Indexes (0)
◢ Foreign Keys (2)
      FK_student_cursus_cursusSQL  (Id)
      FK_student_cursus_studentSQL  (Id)
  Triggers (0)
```



Properties

<unnamed> Primary Key

(Name)	<unnamed>
Allow Page Locks	True
Allow Row Locks	True
Columns	cursusId ASC, studentId ASC
⊞ Data Compression Opt	
⊞ Data Space Specificatic	Filegroup
Description	
Enabled	True
Filestream Filegroup	
⊞ Fill Specifications	
Ignore Duplicate Keys	False
Is Clustered	True
Is Full Text Key	False
Recompute Statisics	True

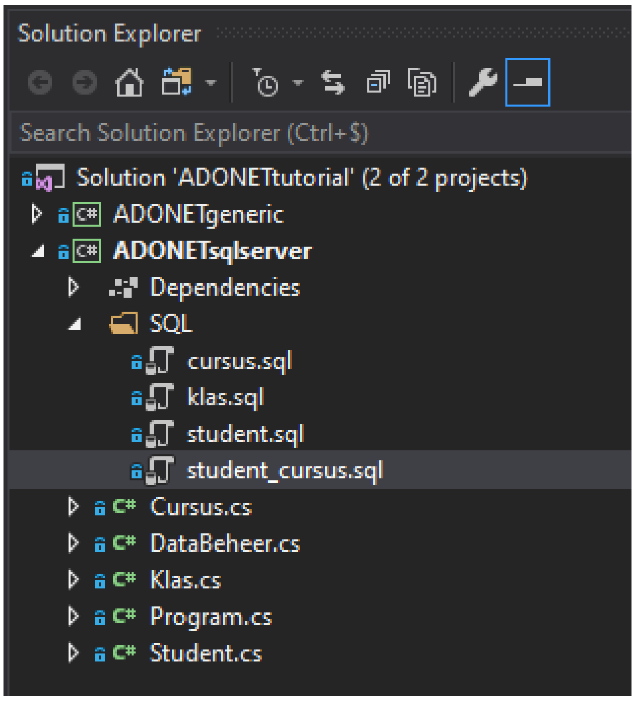


Primary Key Columns

Specify the columns and sort order:

Column Name	Sort Order
cursusId	Ascending
studentId	Ascending

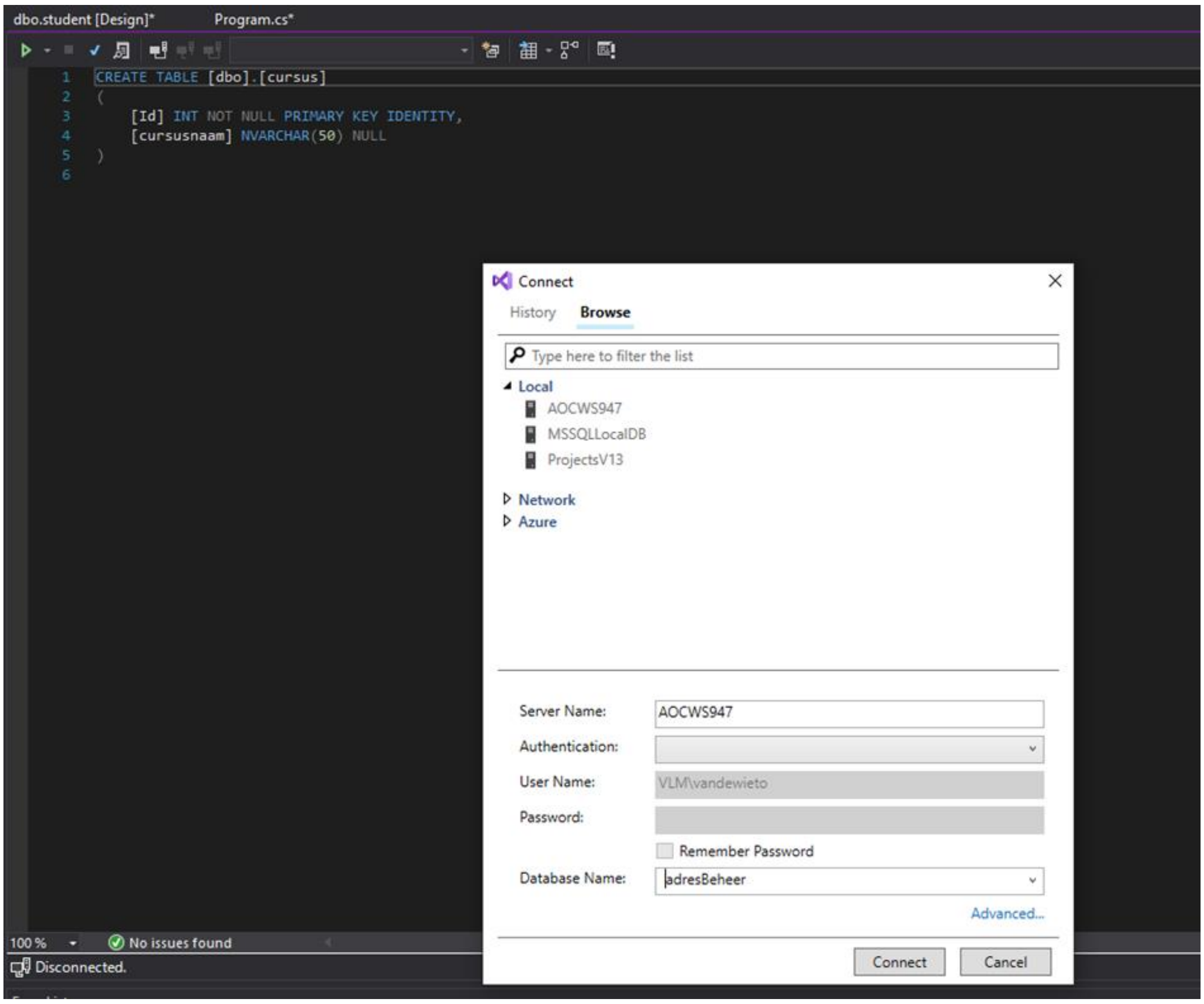
Opslaan SQL statements



Solution Explorer

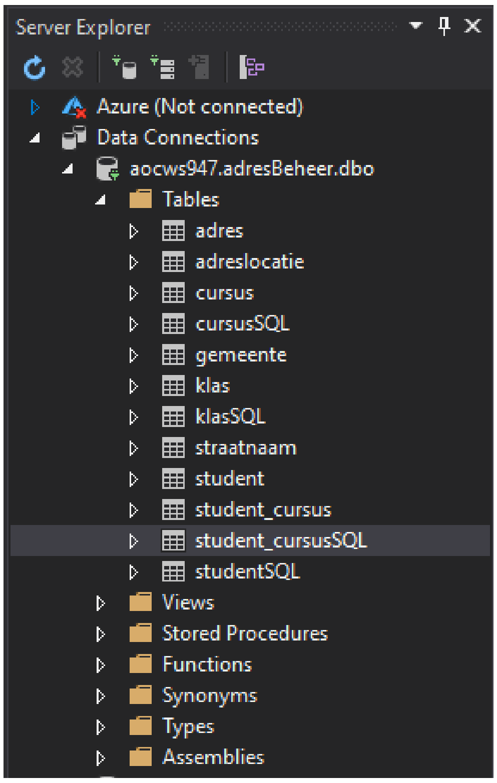
```
Solution 'ADONETtutorial' (2 of 2 projects)
  ADONETgeneric
  ADONETsqlserver
    Dependencies
    SQL
      cursus.sql
      klas.sql
      student.sql
      student_cursus.sql
    Cursus.cs
    DataBeheer.cs
    Klas.cs
    Program.cs
    Student.cs
```

Stap 4 – aanmaken tabellen in SQL Server



```
CREATE TABLE [dbo].[cursus]
(
    [Id] INT NOT NULL PRIMARY KEY IDENTITY,
    [cursusnaam] NVARCHAR(50) NULL
)
```

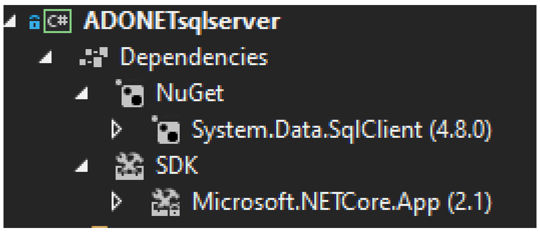
Server Name: AOCWS947
Authentication:
User Name: VLM\vandewieto
Password:
Remember Password
Database Name: adresBeheer



1.2 SqlServer data provider

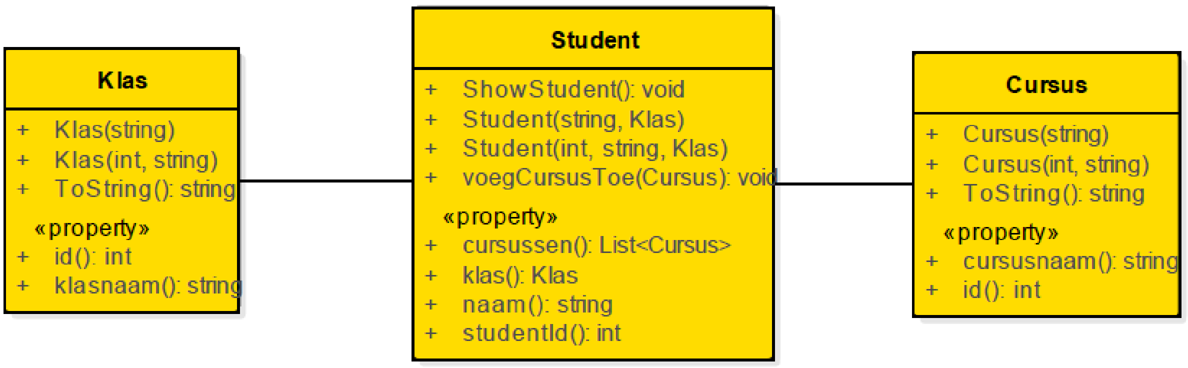
1.2.1 Aanmaken project

- Add .Net Core project
- Dependencies

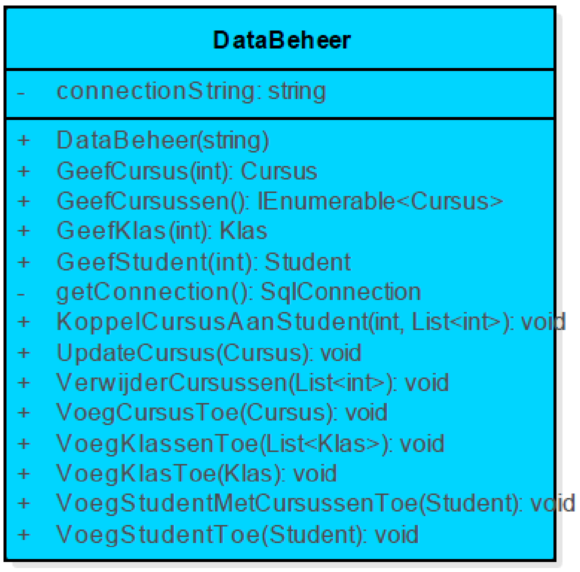


```
ADONETsqlserver
  Dependencies
    NuGet
      System.Data.SqlClient (4.8.0)
    SDK
      Microsoft.NETCore.App (2.1)
```

- Klassen

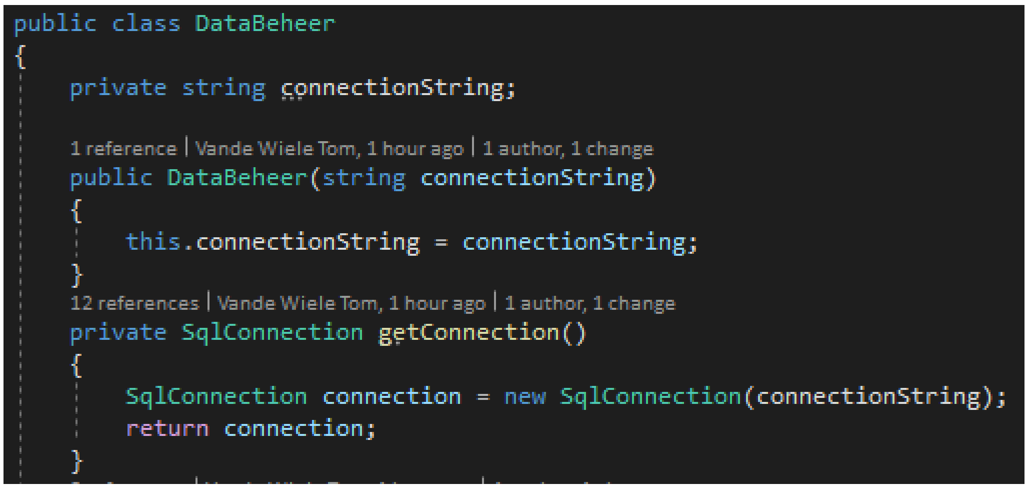


```
Klas
+ Klas(string)
+ Klas(int, string)
+ ToString(): string
«property»
+ id(): int
+ klasnaam(): string

Student
+ ShowStudent(): void
+ Student(string, Klas)
+ Student(int, string, Klas)
+ voegCursusToe(Cursus): void
«property»
+ cursussen(): List<Cursus>
+ klas(): Klas
+ naam(): string
+ studentId(): int

Cursus
+ Cursus(string)
+ Cursus(int, string)
+ ToString(): string
«property»
+ cursusnaam(): string
+ id(): int
```



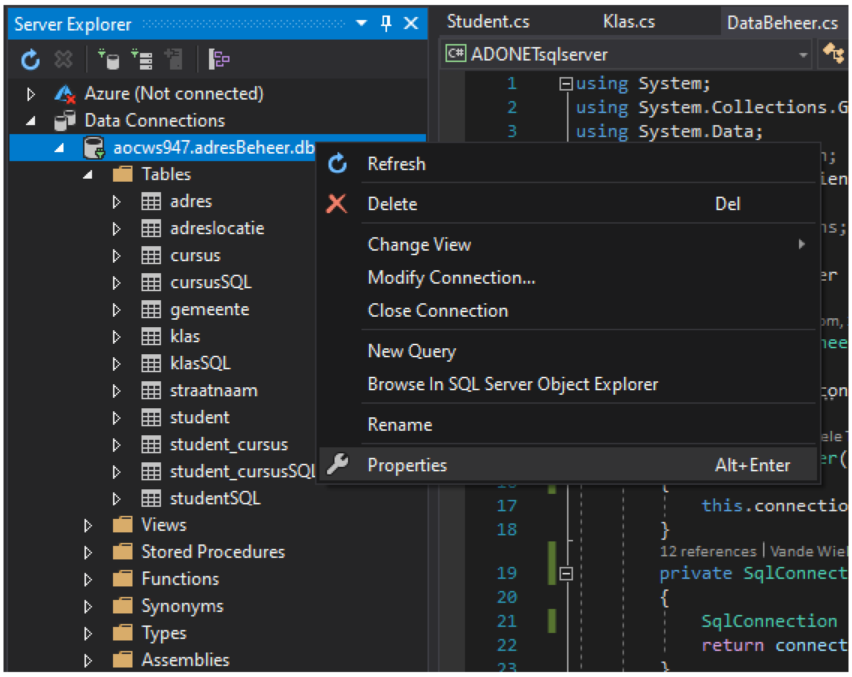
```
DataBeheer
- connectionString: string
+ DataBeheer(string)
+ GeefCursus(int): Cursus
+ GeefCursussen(): IEnumerable<Cursus>
+ GeefKlas(int): Klas
+ GeefStudent(int): Student
- getConnection(): SqlConnection
+ KoppelCursusAanStudent(int, List<int>): void
+ UpdateCursus(Cursus): void
+ VerwijderCursussen(List<int>): void
+ VoegCursusToe(Cursus): void
+ VoegKlassenToe(List<Klas>): void
+ VoegKlasToe(Klas): void
+ VoegStudentMetCursussenToe(Student): void
+ VoegStudentToe(Student): void
```

1.2.2 Connection



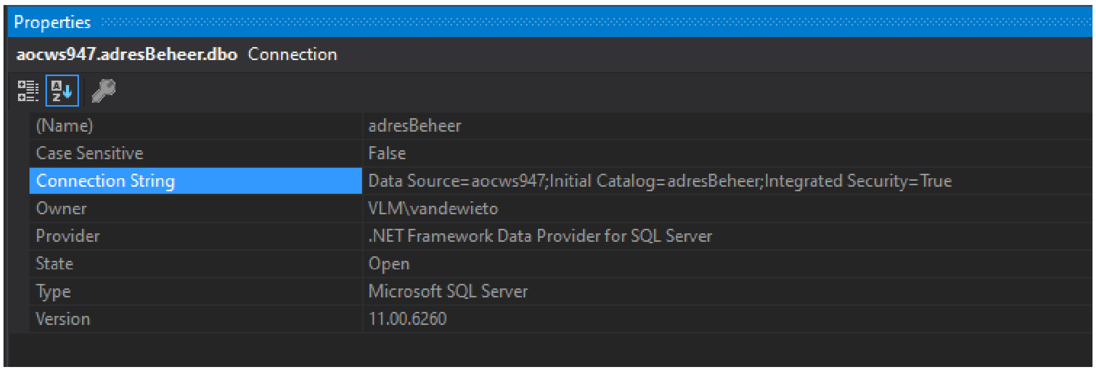
```
public class DataBeheer
{
    private string connectionString;

    1 reference | Vande Wiele Tom, 1 hour ago | 1 author, 1 change
    public DataBeheer(string connectionString)
    {
        this.connectionString = connectionString;
    }
    12 references | Vande Wiele Tom, 1 hour ago | 1 author, 1 change
    private SqlConnection getConnection()
    {
        SqlConnection connection = new SqlConnection(connectionString);
        return connection;
    }
```

ConnectionString – selecteer properties in server beheer



```
using System;
using System.Collections.G
using System.Data;
```

```
        this.connectio
    }
    12 references | Vande Wiel
    private SqlConnect
    {
        SqlConnection
        return connect
```



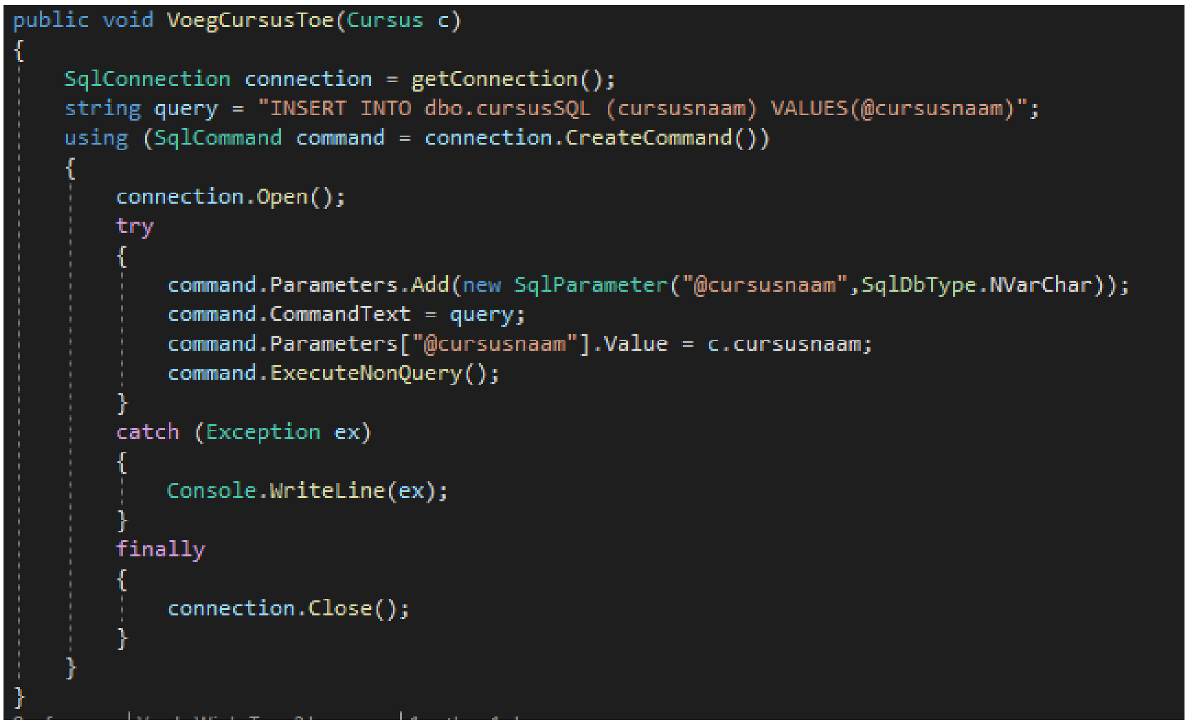
Properties

aocws947.adresBeheer.dbo Connection

(Name)	adresBeheer
Case Sensitive	False
Connection String	Data Source=aocws947;Initial Catalog=adresBeheer;Integrated Security=True
Owner	VLM\vandewieto
Provider	.NET Framework Data Provider for SQL Server
State	Open
Type	Microsoft SQL Server
Version	11.00.6260

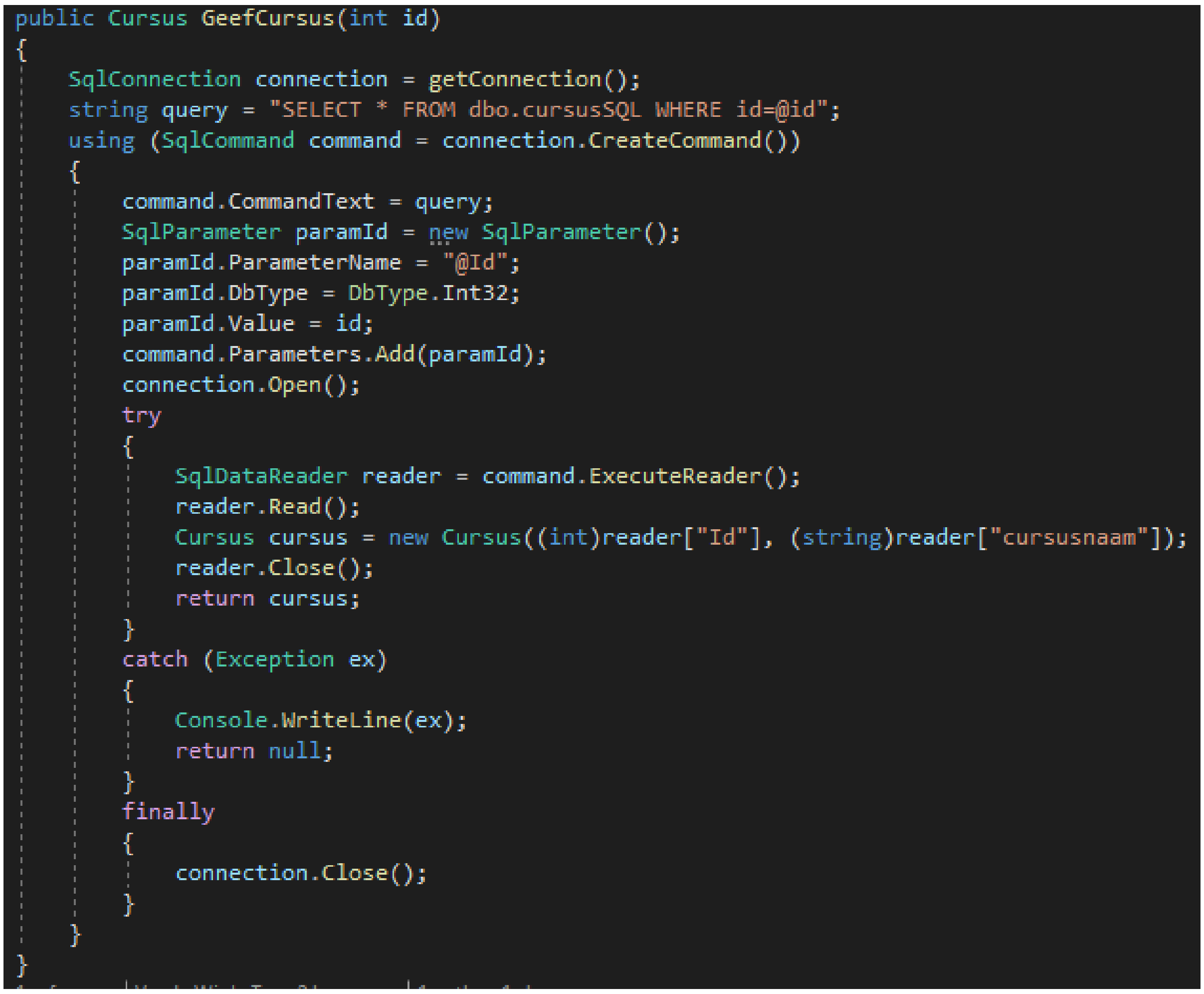
1.2.3 Command en Parameters

1.2.3.1 Insert

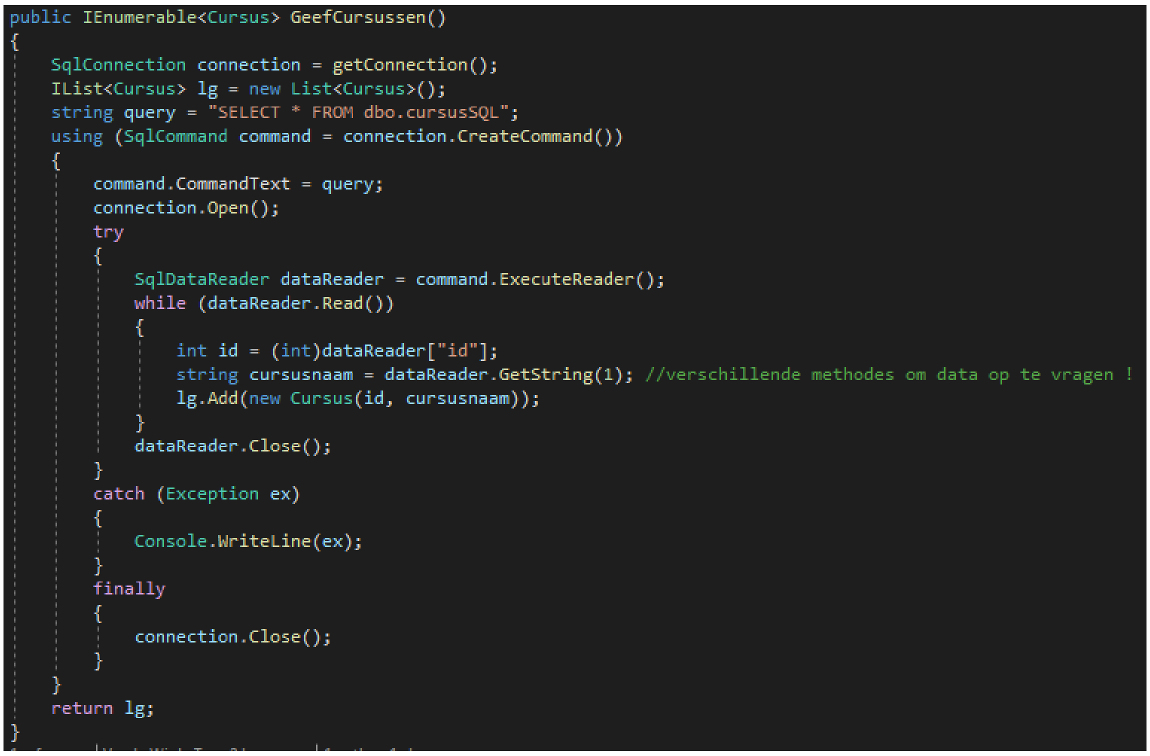


```
public void VoegCursusToe(Cursus c)
{
    SqlConnection connection = getConnection();
    string query = "INSERT INTO dbo.cursusSQL (cursusnaam) VALUES(@cursusnaam)";
    using (SqlCommand command = connection.CreateCommand())
    {
        connection.Open();
        try
        {
            command.Parameters.Add(new SqlParameter("@cursusnaam",SqlDbType.NVarChar));
            command.CommandText = query;
            command.Parameters["@cursusnaam"].Value = c.cursusnaam;
            command.ExecuteNonQuery();
        }
        catch (Exception ex)
        {
            Console.WriteLine(ex);
        }
        finally
        {
            connection.Close();
        }
    }
}
```

1.2.3.2 Datareader

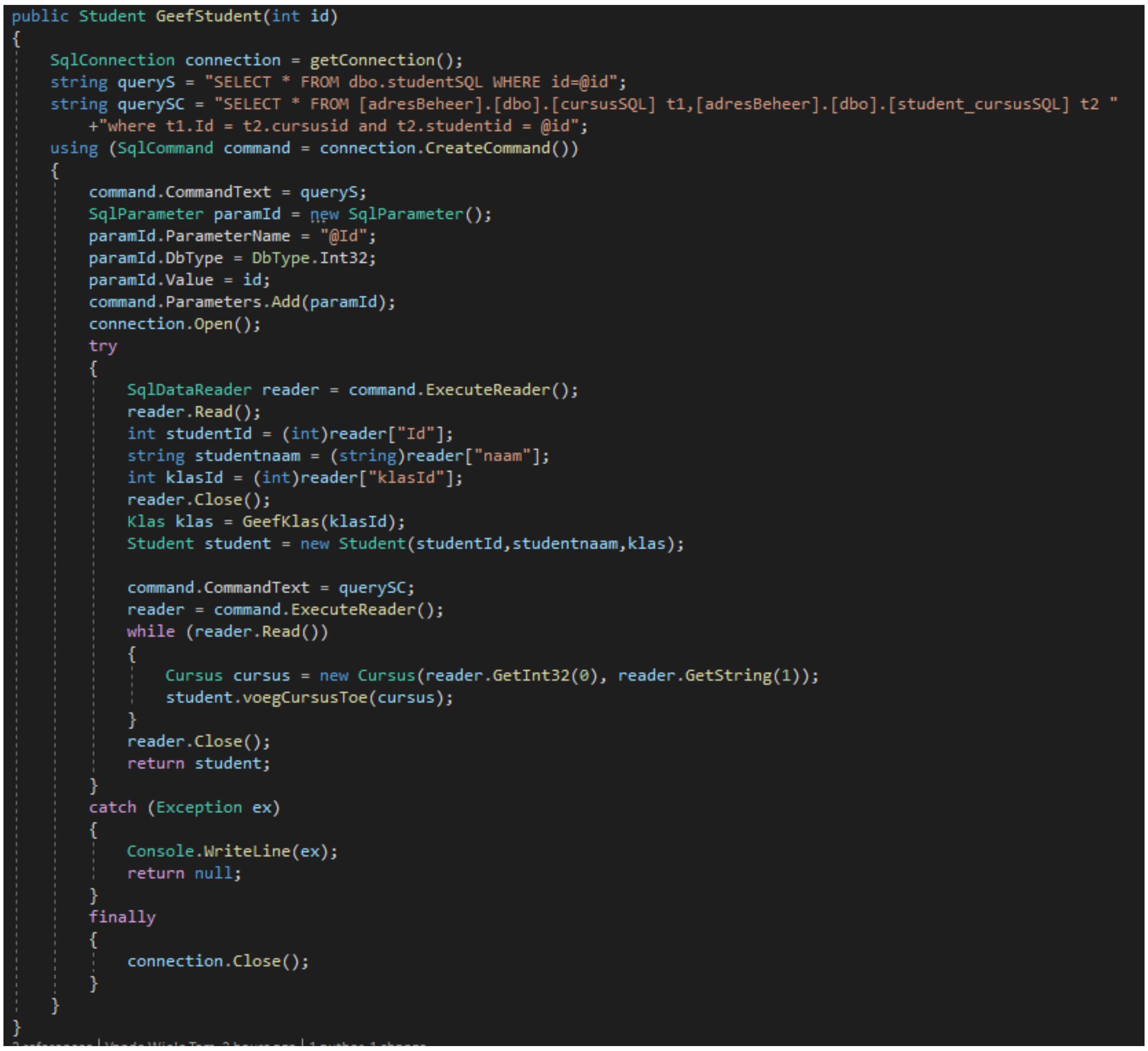


```
public Cursus GeefCursus(int id)
{
    SqlConnection connection = getConnection();
    string query = "SELECT * FROM dbo.cursusSQL WHERE id=@id";
    using (SqlCommand command = connection.CreateCommand())
    {
        command.CommandText = query;
        SqlParameter paramId = new SqlParameter();
        paramId.ParameterName = "@Id";
        paramId.DbType = DbType.Int32;
        paramId.Value = id;
        command.Parameters.Add(paramId);
        connection.Open();
        try
        {
            SqlDataReader reader = command.ExecuteReader();
            reader.Read();
            Cursus cursus = new Cursus((int)reader["Id"], (string)reader["cursusnaam"]);
            reader.Close();
            return cursus;
        }
        catch (Exception ex)
        {
            Console.WriteLine(ex);
            return null;
        }
        finally
        {
            connection.Close();
        }
    }
}
```



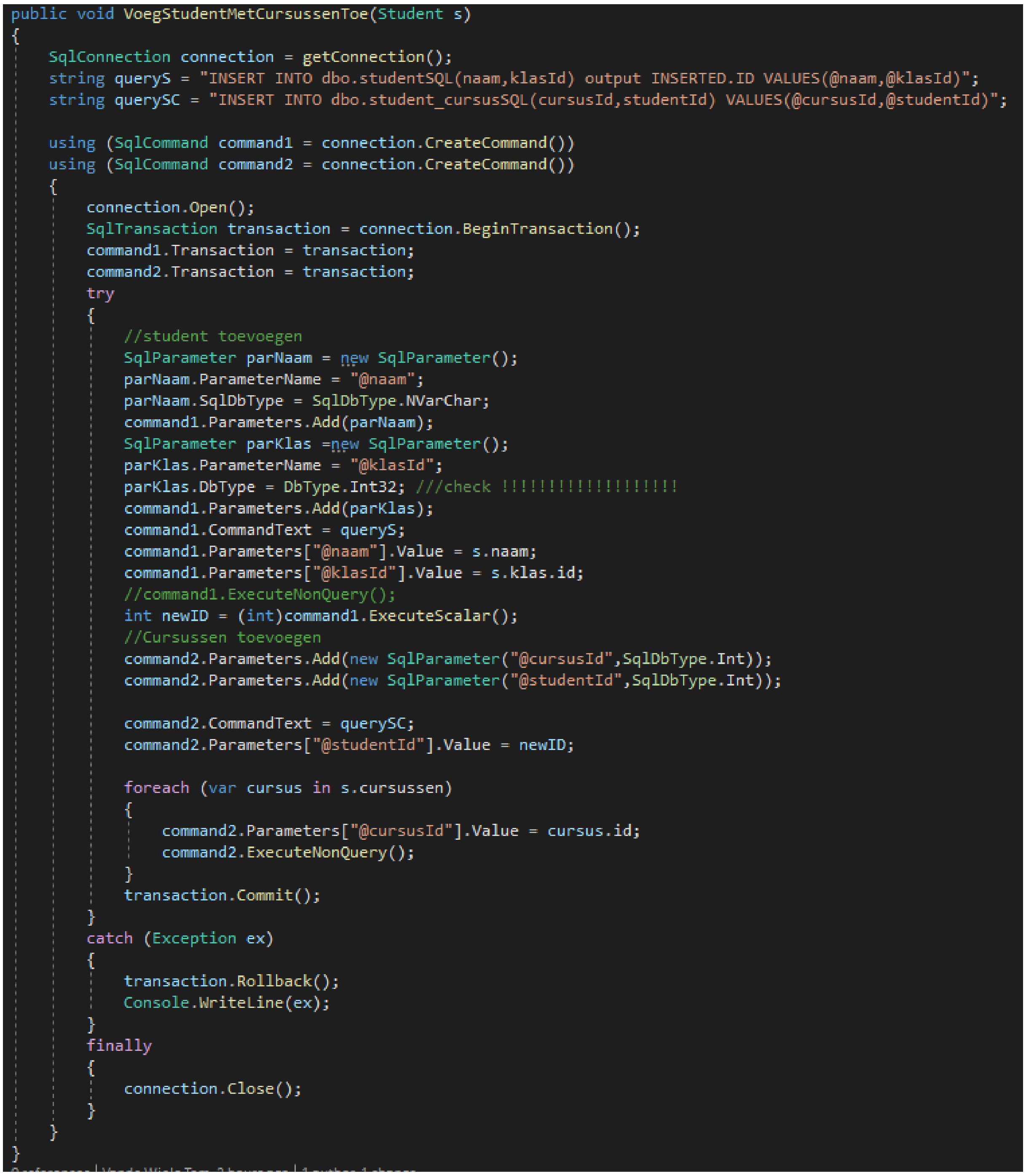
```
public IEnumerable<Cursus> GeefCursussen()
{
    SqlConnection connection = getConnection();
    IList<Cursus> lg = new List<Cursus>();
    string query = "SELECT * FROM dbo.cursusSQL";
    using (SqlCommand command = connection.CreateCommand())
    {
        command.CommandText = query;
        connection.Open();
        try
        {
            SqlDataReader dataReader = command.ExecuteReader();
            while (dataReader.Read())
            {
                int id = (int)dataReader["id"];
                string cursusnaam = dataReader.GetString(1); //verschillende methodes om data op te vragen !
                lg.Add(new Cursus(id, cursusnaam));
            }
            dataReader.Close();
        }
        catch (Exception ex)
        {
            Console.WriteLine(ex);
        }
        finally
        {
            connection.Close();
        }
    }
    return lg;
}
```

1.2.3.3 Query over meerdere tabellen



```
public Student GeefStudent(int id)
{
    SqlConnection connection = getConnection();
    string queryS = "SELECT * FROM dbo.studentSQL WHERE id=@id";
    string querySC = "SELECT * FROM [adresBeheer].[dbo].[cursusSQL] t1,[adresBeheer].[dbo].[student_cursusSQL] t2 "
        +"where t1.Id = t2.cursusid and t2.studentid = @id";
    using (SqlCommand command = connection.CreateCommand())
    {
        command.CommandText = queryS;
        SqlParameter paramId = new SqlParameter();
        paramId.ParameterName = "@Id";
        paramId.DbType = DbType.Int32;
        paramId.Value = id;
        command.Parameters.Add(paramId);
        connection.Open();
        try
        {
            SqlDataReader reader = command.ExecuteReader();
            reader.Read();
            int studentId = (int)reader["Id"];
            string studentnaam = (string)reader["naam"];
            int klasId = (int)reader["klasId"];
            reader.Close();
            Klas klas = GeefKlas(klasId);
            Student student = new Student(studentId,studentnaam,klas);

            command.CommandText = querySC;
            reader = command.ExecuteReader();
            while (reader.Read())
            {
                Cursus cursus = new Cursus(reader.GetInt32(0), reader.GetString(1));
                student.voegCursusToe(cursus);
            }
            reader.Close();
            return student;
        }
        catch (Exception ex)
        {
            Console.WriteLine(ex);
            return null;
        }
        finally
        {
            connection.Close();
        }
    }
}
```

1.2.4 Transaction

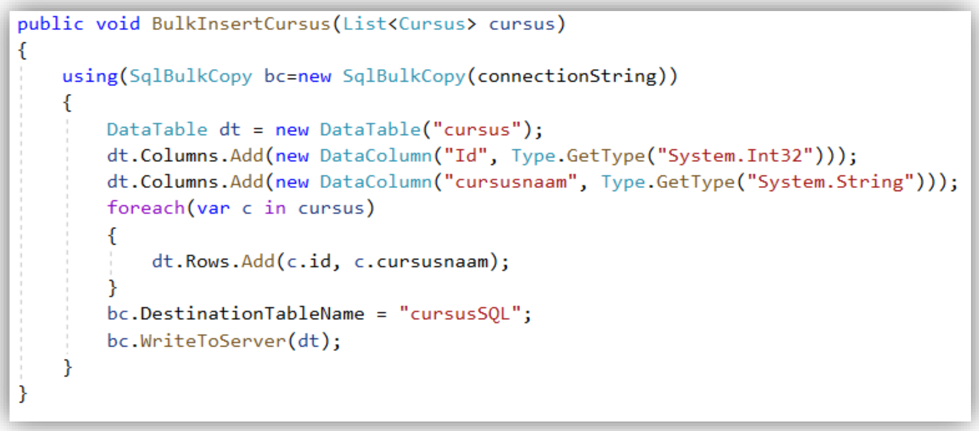


```
public void VoegStudentMetCursussenToe(Student s)
{
    SqlConnection connection = getConnection();
    string queryS = "INSERT INTO dbo.studentSQL(naam,klasId) output INSERTED.ID VALUES(@naam,@klasId)";
    string querySC = "INSERT INTO dbo.student_cursusSQL(cursusId,studentId) VALUES(@cursusId,@studentId)";

    using (SqlCommand command1 = connection.CreateCommand())
    using (SqlCommand command2 = connection.CreateCommand())
    {
        connection.Open();
        SqlTransaction transaction = connection.BeginTransaction();
        command1.Transaction = transaction;
        command2.Transaction = transaction;
        try
        {
            //student toevoegen
            SqlParameter parNaam = new SqlParameter();
            parNaam.ParameterName = "@naam";
            parNaam.SqlDbType = SqlDbType.NVarChar;
            command1.Parameters.Add(parNaam);
            SqlParameter parKlas =new SqlParameter();
            parKlas.ParameterName = "@klasId";
            parKlas.DbType = DbType.Int32; ///check !!!!!!!!!!!!!!!!!!!
            command1.Parameters.Add(parKlas);
            command1.CommandText = queryS;
            command1.Parameters["@naam"].Value = s.naam;
            command1.Parameters["@klasId"].Value = s.klas.id;
            //command1.ExecuteNonQuery();
            int newID = (int)command1.ExecuteScalar();
            //Cursussen toevoegen
            command2.Parameters.Add(new SqlParameter("@cursusId",SqlDbType.Int));
            command2.Parameters.Add(new SqlParameter("@studentId",SqlDbType.Int));

            command2.CommandText = querySC;
            command2.Parameters["@studentId"].Value = newID;

            foreach (var cursus in s.cursussen)
            {
                command2.Parameters["@cursusId"].Value = cursus.id;
                command2.ExecuteNonQuery();
            }
            transaction.Commit();
        }
        catch (Exception ex)
        {
            transaction.Rollback();
            Console.WriteLine(ex);
        }
        finally
        {
            connection.Close();
        }
    }
}
```

Opmerking : insert over meerdere tabellen met identity column !

1.3 BulkCopy

Wanneer we grote hoeveelheden data wensen te importeren in de databank is het aangewezen om gebruik te maken van de SqlBulkCopy class. Op deze manier kan er veel efficiënter (sneller) data worden weggeschreven in een tabel. Een SqlBulkCopy object kan worden aangemaakt met als paramter een connection string of een open connection object. Voor het schreven van de gegevens naar de databank, geven we aan in welke tabel dat moet gebeuren (DestinationTableName) en maken we gebruik van de methode WriteToServer. De data zelf geven we mee als een DataTable object, waarin we eerst de kolommen definiëren (zie voorbeeld) en daarna de rijen opvullen met onze gegevens.



```csharp
public void BulkInsertCursus(List<Cursus> cursus)
{
    using(SqlBulkCopy bc=new SqlBulkCopy(connectionString))
    {
        DataTable dt = new DataTable("cursus");
        dt.Columns.Add(new DataColumn("Id", Type.GetType("System.Int32")));
        dt.Columns.Add(new DataColumn("cursusnaam", Type.GetType("System.String")));
        foreach(var c in cursus)
        {
            dt.Rows.Add(c.id, c.cursusnaam);
        }
        bc.DestinationTableName = "cursusSQL";
        bc.WriteToServer(dt);
    }
}
```

(https://docs.microsoft.com/en-us/dotnet/api/system.data.sqlclient.sqlbulkcopy?view=dotnet-plat-ext-5.0)