# How to bulk insert data to SQL Server database in .NET Core

# Introduction

This sample demonstrates how to bulk insert data to SQL Server database in .NET Core.

**Sample prerequisites**

* Visual Studio 2017 or above [[Visual Studio Home Page](https://www.visualstudio.com/)]
* SQL Server 2014
* Create database and table

CREATE DATABASE BulkInsertTestDB

GO

USE BulkInsertTestDB

GO

CREATE TABLE Student(

ID int identity primary key,

Name nvarchar(50),

Age int

)

GO

## Building the sample

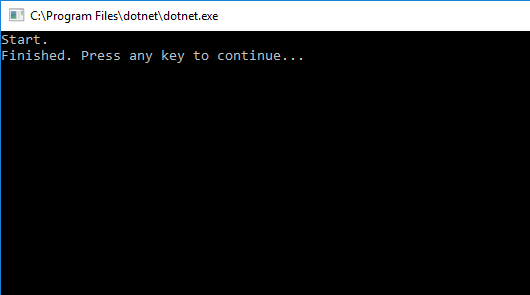
* Open the sample solution “CSNETCoreBulkInsert.sln” using Visual Studio
* Right click the project “CSNETCoreBulkInsert” and select Restore Packages
* Update the connection string in the **Main** function

BulkCopyHelper helper = new BulkCopyHelper("Server=.;Database= BulkInsertTestDB;Trusted\_Connection=True;");

* Press **F6** Key or select **Build -> Build Solution** from the menu to build the sample.

## Running the sample

* Open the sample solution using Visual Studio, then press F5 Key or select Debug -> Start Debugging from the menu.



* Go to sql server client, and run script:

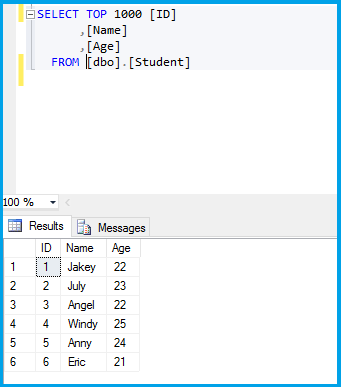
SELECT [ID]

,[Name]

,[Age]

FROM [dbo].[Student]

* You will see that student data has been inserted into database.



**Using the code**

CSNETCoreBulkInsert.cs:

public class StudentDbDataReader : DbDataReader

{

private List<Student> Students;

private List<Student>.Enumerator \_enumerator;

protected List<string> Fields;

protected List<Type> FieldTypes;

protected bool isClosed;

private Student \_current;

public StudentDbDataReader(List<Student> list)

{

Students = list;

\_enumerator = list.GetEnumerator();

IEnumerable<PropertyInfo> listProperty = typeof(Student).GetRuntimeProperties();

Fields = new List<string>();

FieldTypes = new List<Type>();

foreach (PropertyInfo p in listProperty)

{

Fields.Add(p.Name);

FieldTypes.Add(p.PropertyType);

}

}

public override object this[int ordinal] => GetValue(ordinal);

public override object this[string name] => GetValue(GetOrdinal(name));

public override int Depth => throw new NotImplementedException();

public override int FieldCount => Fields.Count;

public override bool HasRows => Students.Count>0;

public override bool IsClosed => isClosed;

public override int RecordsAffected => -1;

public override bool GetBoolean(int ordinal)

{

return (bool)GetValue(ordinal);

}

public override byte GetByte(int ordinal)

{

return (byte)GetValue(ordinal);

}

public override long GetBytes(int ordinal, long dataOffset, byte[] buffer, int bufferOffset, int length)

{

throw new NotImplementedException();

}

public override char GetChar(int ordinal)

{

return (char)GetValue(ordinal);

}

public override long GetChars(int ordinal, long dataOffset, char[] buffer, int bufferOffset, int length)

{

throw new NotImplementedException();

}

public override string GetDataTypeName(int ordinal)

{

return GetFieldType(ordinal).Name;

}

public override DateTime GetDateTime(int ordinal)

{

return (DateTime)GetValue(ordinal);

}

public override decimal GetDecimal(int ordinal)

{

return (decimal)GetValue(ordinal);

}

public override double GetDouble(int ordinal)

{

return (double)GetValue(ordinal);

}

public override IEnumerator GetEnumerator()

{

return Students.GetEnumerator();

}

public override Type GetFieldType(int ordinal)

{

return FieldTypes[ordinal];

}

public override float GetFloat(int ordinal)

{

return (float)GetValue(ordinal);

}

public override Guid GetGuid(int ordinal)

{

return (Guid)GetValue(ordinal);

}

public override short GetInt16(int ordinal)

{

return (short)GetValue(ordinal);

}

public override int GetInt32(int ordinal)

{

return (int)GetValue(ordinal);

}

public override long GetInt64(int ordinal)

{

return (long)GetValue(ordinal);

}

public override string GetName(int ordinal)

{

return Fields[ordinal];

}

public override int GetOrdinal(string name)

{

return Fields.IndexOf(name);

}

public override string GetString(int ordinal)

{

return (string)GetValue(ordinal);

}

public override object GetValue(int ordinal)

{

return typeof(Student).GetRuntimeProperty(GetName(ordinal)).GetValue(\_current);

}

public override int GetValues(object[] values)

{

throw new NotImplementedException();

}

public override bool IsDBNull(int ordinal)

{

throw new NotImplementedException();

}

public override bool NextResult()

{

throw new NotImplementedException();

}

public override bool Read()

{

bool result = \_enumerator.MoveNext();

\_current = result ? \_enumerator.Current : new Student();

return result;

}

}

## More information

[Bulk Copy](https://msdn.microsoft.com/en-us/library/system.data.sqlclient.sqlbulkcopy(v=vs.110).aspx)

[System.Data.Common.DbDataReader](https://msdn.microsoft.com/en-us/library/system.data.common.dbdatareader%28v=vs.110%29.aspx)