# Lab: ORM and Entity Framework

You can check your solutions here: <https://judge.softuni.bg/Contests/3202/Additional-Exercises>.

**Use the provided skeleton from resources! Do not change its methods, classes and namespaces!**

## Import the SoftUni Database

Import the SoftUni DB into SQL Management Studio (if not yet imported) by **executing** the provided **.sql** script.

****

## Generate Database First ORM Model

Model the existing database by using “**Database First**” in Entity Framework Core.

First create a new empty **.Net Core** **Console Application** and after it is created open the **Package Manager Console**:



It will look something like this:



Use it to run the following commands **one by one**:

|  |
| --- |
| Install-Package Microsoft.EntityFrameworkCore.Tools |
| Install-Package Microsoft.EntityFrameworkCore.SqlServer |
| Install-Package Microsoft.EntityFrameworkCore.SqlServer.Design |

Graphical user interface, text, application

Description automatically generated

These are the **packages** you will need, in order to **scaffold** our **SoftUniContext** from the **SoftUni** **database**.

Next, we must **execute** the **command** to **scaffold** our **context** **class**. It will consist of 4 things:

* First, the name of the command:

|  |
| --- |
| Scaffold-DbContext |

* Second, the connection we will be using (our connection string):

|  |
| --- |
| -Connection "Server=<ServerName>;Database=<DatabaseName>;Integrated Security=True;" |

For **ServerName**, use the name of your local MS SQL Server instance or ".".

For **DatabaseName**, use the name of the database you want to use, in this case – **SoftUni**.

* Third, we need to declare our service provider, we’ll be using **Microsoft.EntityFrameworkCore.SqlServer**:

|  |
| --- |
| -Provider Microsoft.EntityFrameworkCore.SqlServer |

* And the fourth thing we’ll do, is to give it a directory where all of our models will go (e.g. **Models**):

|  |
| --- |
| -OutputDir Data/Models |

Our final command will look like this:

|  |
| --- |
| Scaffold-DbContext -Connection "Server=(localdb)\MSSQLLocalDB;Database=SoftUni;Integrated Security=True;" -Provider Microsoft.EntityFrameworkCore.SqlServer -OutputDir Data/Models |

Text

Description automatically generated

Execute the **whole command** on a **single line**.

Entity Framework Core has successfully **mapped the database schema to C# classes**. However, it isn't good enough with names – all classes have been **pluralized**.

* Use the **Solution Explorer** in Visual Studio to move the **SoftUniContext** class out of **Models** intothe **Data** folder and rename all of our classes properly.
* Use **right click →** [**Rename**] or the [**F2**] shortcut and press [**OK**] on this **pop** **up** **window** after each class:



This way Visual Studio will also **rename** the **classes** **everywhere** they’re used.

The final result should look like this:

Graphical user interface, application

Description automatically generated

Don’t forget to fix the **SoftUniContext’s** namespace after moving it and add a reference to the **Models** namespace:

**Make** **sure** that your namespaces are **exactly** the same as these:

|  |
| --- |
| SoftUni SoftUni.Data  SoftUni.Models |

Finally, we can clean up the packages we won’t be using anymore from the package manager GUI or by running these commands:

|  |
| --- |
| Uninstall-Package Microsoft.EntityFrameworkCore.Tools -r  Uninstall-Package Microsoft.EntityFrameworkCore.SqlServer.Design -RemoveDependencies |

## Find Employees with Job Title

Create a method **public static string FindEmployeesWithJobTitle(SoftUniContext context)** to print the **First Name** of all employees with **Job Title** equal to **“Design Engineer”**.

Graphical user interface, text, application

Description automatically generated

### Hints:

First, use the **context** in the method like this:

A picture containing graphical user interface

Description automatically generated

Get all employees and **filter** them using **context.Employees**. Then, select only the **First Name** of each employee and use **String.Join()** to return the array of names as a string to the method.

Graphical user interface, text, application

Description automatically generated

### Run Your Code in the Console

Invoke the **FindEmployeesWithJobTitle(SoftUniContext context)** method from the application entry point Main():

Text

Description automatically generated

Press **[Ctrl+F5]** to run the application. Check if the result on the console is correct:

Graphical user interface, text, application

Description automatically generated

### Submit Your Code to Judge

**Save your files** in Visual Studio. Delete the **“**bin**”** and **“**obj**”** folders from the **SoftUni** folder and create a **ZIP archive** of your solution:

Graphical user interface

Description automatically generated with low confidence Graphical user interface, application

Description automatically generated

**Submit** the ZIP file in Judge:

Graphical user interface, application, Word

Description automatically generated

You should get 100 / 100 score:

Graphical user interface, table

Description automatically generated

## Find Project with ID

Again, use the **context** and get all **Projects** from it. Use **.Find()** method to find the project with **ID 2** and return the **Name** of the project.

Graphical user interface, text, application

Description automatically generated

### Hints:

Graphical user interface, text

Description automatically generated

## Create New Project

Your task is to create a **new Project** in the **Projects** table.

### Hints:

To create a new database **row** use the **.Add()** method of the corresponding **DbSet**. First, create a new **Project** **object** and give values to **Name** and **StartDate** properties.

Graphical user interface, text, application

Description automatically generated

Then, add the object to the **DbSet** and do not forget to **save changes** the following way:

Text

Description automatically generated

**Run** the app. There is no result displayed on the console.

### Check the Result in the DB

In order to check the result, go to **SQL Server Management Studio -> Object Explorer -> Databases -> SoftUni -> dbo.Projects**. **Right-click** on it and choose **Select Top 1000 Rows**.

Graphical user interface, text, application, email

Description automatically generated

Scroll down to the **last entity**. It should be the one we added using a C# command in Visual Studio.



## Update First Employee

Get the **first employee** using **.FirstOrDefault()** method and change their **First Name** to **“Alex”**. Do not forget to **save changes**! In case there are no employees, return **empty string** to the method, else return the changed employee’s first name.

### Hints:

Graphical user interface, text, application

Description automatically generated

**Check Result in the DB**

This is the database entity **before** the code execution:



After the code execution, the entity **should be changed**:



## Delete First Project

Get the **first project** and delete it using the **.Remove()** method. Do not forget to **save changes**! The entity we should remove is the following:



### Hints:

Text

Description automatically generated

However, when the program is executed an **error message** appears.

Text

Description automatically generated

The reason for the error is that the **EmployeesProjects table** in the SoftUni DB contains a **ProjectID column**. So, entities from the Projects table **cannot be deleted** that way because some entities in the EmployeesProjects table contain the id of the project entity we want to delete. To solve that issue we may **first delete all enities** from the **EmployeesProjects** table, which contain our **ProjectId** (in our case with ProjectId=**1**). The command is the following:

Text

Description automatically generated

**Check Result in the DB**

Execute the program and see the result in the **Projects** table in the SoftUni DB.



You can also check the **EmployeesProjects** table. Now it does not contain entities with **ProjectId = 1**.

Graphical user interface, text, application, Word

Description automatically generated

## Update Addresses

Write a **method** to update **TownId** to **2** for all **Addresses** with **AddressText**, containing the word **“Drive”**.

Table, Excel

Description automatically generated Table, Excel

Description automatically generated

You can check the result in the **SoftUni DB** with this command:

Text

Description automatically generated

The method **UpdateAddresses(SoftUniContext context)** should return the **count** of **changed** addresses, converted to **string**.

Text

Description automatically generated

## Store Database

Use the **Code-First model** to create a **Store** **database** with two tables: **Orders** and **OrderDetails**. Use **Migrations** to create that database in **Microsoft SQL Server**.

Table **Orders:**

* int OrderID
* int Customer
* int Employee
* DateTime OrderDate
* List<OrderDetail> OrderDetails

Table **OrderDetails**:

* int OrderDetailID
* int OrderID
* Order Order
* int Product
* int Quantity

**Install** the following **Packages:**

|  |
| --- |
| Microsoft.EntityFrameworkCore.SqlServer |
| Microsoft.EntityFrameworkCore |
| Microsoft.EntityFrameworkCore.Tools |
| Microsoft.EntityFrameworkCore.Design |