Lab: Advanced Querying

This document defines the exercise assignments for the ["Databases Advanced – EF Core" course @ Software University.]()

Use the provided skeleton.

1. Add Employee to Project

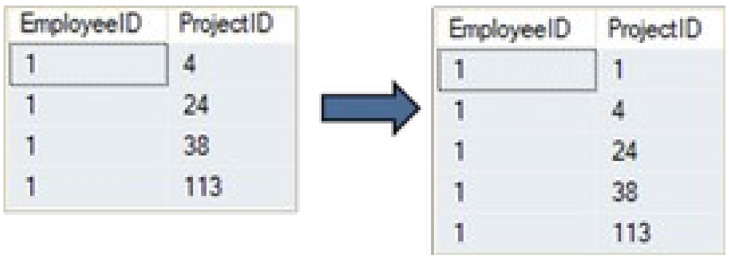
NOTE: You will need method public static string **AddEmployeeToProject**(SoftUniContext context, int employeeId, int projectId) in the public StartUp class.

Execute your **own SQL query** using **the stored procedure** to **add a project to an employee** in the SoftUni database.

Return the employee Id and the project id of his top 3 projects, ordered by Id in ascending order in the following format:

```
Employee Id:{Employee Id}, Projects:
{Project Id}
…
```

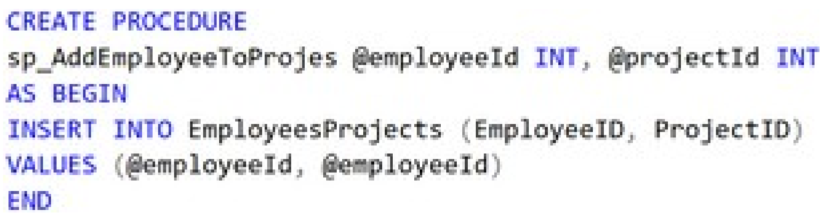
First, **create** a **stored** procedure `sp_AddEmployeeToProjest` that **accepts two parameters**: EmployeeID and ProjectID.



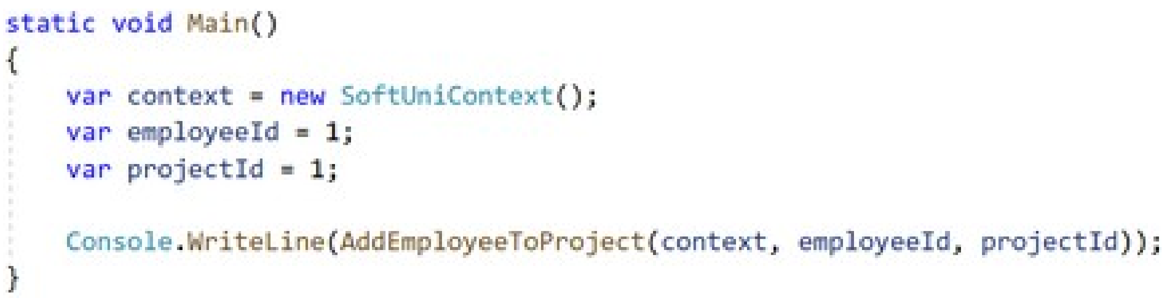
EmployeeID	ProjectID
1	4
1	24
1	38
1	113

EmployeeID	ProjectID
1	1
1	4
1	24
1	38
1	113

Hints:

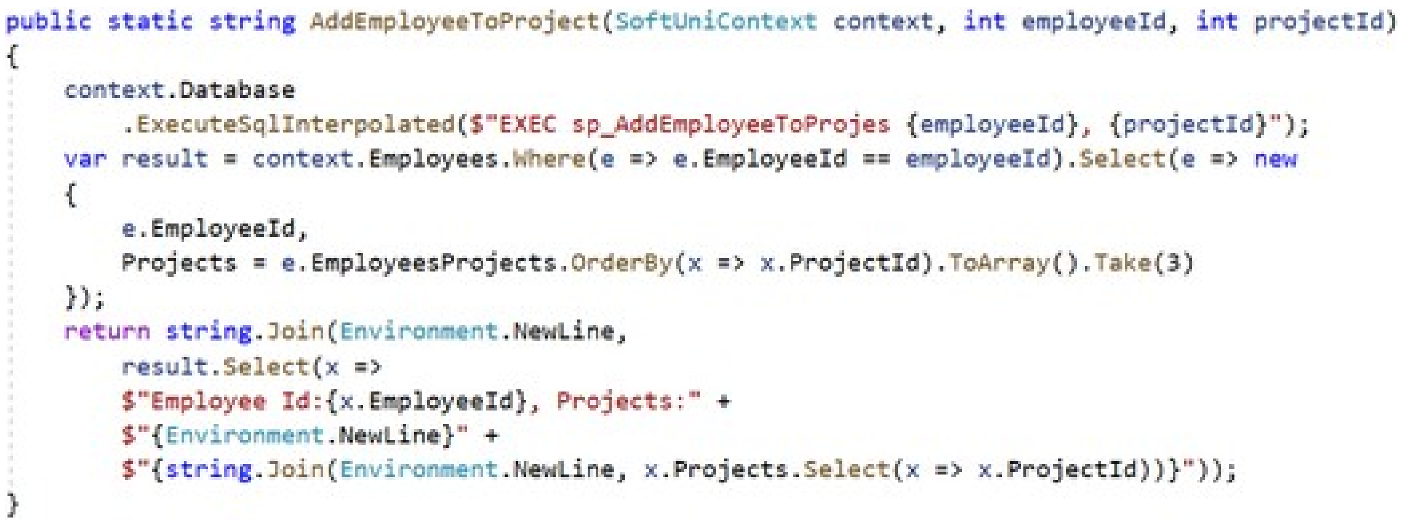


```
CREATE PROCEDURE
sp_AddEmployeeToProjes @employeeId INT, @projectId INT
AS BEGIN
INSERT INTO EmployeesProjects (EmployeeID, ProjectID)
VALUES (@employeeId, @employeeId)
END
```



```
static void Main()
{
    var context = new SoftUniContext();
    var employeeId = 1;
    var projectId = 1;

    Console.WriteLine(AddEmployeeToProject(context, employeeId, projectId));
}
```



```csharp
public static string AddEmployeeToProject(SoftUniContext context, int employeeId, int projectId)
{
    context.Database
        .ExecuteSqlInterpolated($"EXEC sp_AddEmployeeToProjes {employeeId}, {projectId}");
    var result = context.Employees.Where(e => e.EmployeeId == employeeId).Select(e => new
    {
        e.EmployeeId,
        Projects = e.EmployeesProjects.OrderBy(x => x.ProjectId).ToArray().Take(3)
    });
    return string.Join(Environment.NewLine,
        result.Select(x =>
        $"Employee Id:{x.EmployeeId}, Projects:" +
        $"{Environment.NewLine}" +
        $"{string.Join(Environment.NewLine, x.Projects.Select(x => x.ProjectId))}"));
}
```

Output(employeeId = 1, projectId = 1)
`Employee Id:1, Projects:`
1
4
24

2. Delete Records with ProjectId

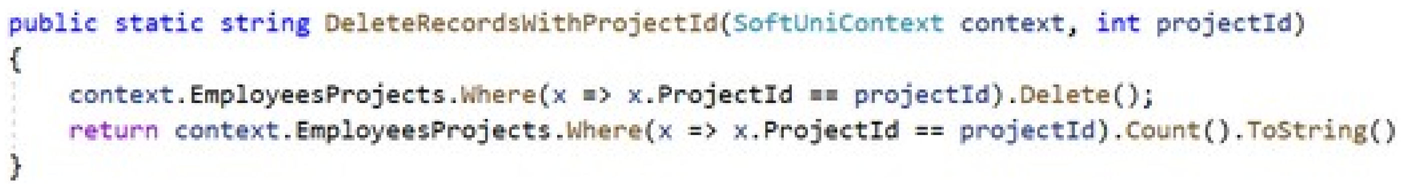
NOTE: You will need method public static string **DeleteRecordsWithProjectId**(SoftUniContext context, int projectId) in the public StartUp class.

Delete the **records** in the **EmployeesProjects** table in the SoftUni database, where **ProjectId** is **equal to** the given projectId and **return** the count of the **EmployeesProjects** with **ProjectId equal to** the given.

Output(projectId = 1)
0

Hints:

We can't delete in tables which don't have a primary key but **`Z.EntityFramework.Plus.EFCore`** and the **`using Z.EntityFramework.Plus`** makes that possible.



```csharp
public static string DeleteRecordsWithProjectId(SoftUniContext context, int projectId)
{
    context.EmployeesProjects.Where(x => x.ProjectId == projectId).Delete();
    return context.EmployeesProjects.Where(x => x.ProjectId == projectId).Count().ToString()
}
```