# Exercises: C# Auto Mapping Objects

This document defines the **exercise assignments** for the ["Databases Advanced – EF Core" course @ Software University](https://softuni.bg/trainings/2251/databases-advanced-entity-framework-february-2019).

## Employees Mapping

Create a simple database with one table – Employees. Each employee should have properties: **first name**, **last name**, **salary**, **birthday** and **address**. Only **first** **name**, **last** **name** and **salary** are **required**.

Create **EmployeeDto** class that will keep synthesized information about instances of Employee class (only **id**, **first name**, **last name** and **salary**).

Create a console app for your database, which uses the **Automapper** package and the **EmployeeDto** class to **transfer** **data** from and back to the database. You should have the following commands:

* **AddEmployee** <**firstName**> <**lastName**> <**salary**> – adds a new Employee to the database
* **SetBirthday <employeeId> <date:** "dd-MM-yyyy"**>** – sets the birthday of the employee to the given date
* **SetAddress** <**employeeId**> <**address**> – sets the address of the employee to the given string
* **EmployeeInfo** <**employeeId**> – prints on the console the information for an employee in the format "ID: {employeeId} - {firstName} {lastName} - ${salary:f2}"
* **EmployeePersonalInfo <employeeId>** – prints all the information for an employee in the following format:

|  |
| --- |
| ID: 1 - Pesho Ivanov - $1000.00  Birthday: 15-04-1976  Address: Sofia, ul. Vitosha 15 |

* **Exit** – closes the application

#### Bonus

Only use **DTOs** in your application. Use a **service** to connect to the **database**.

## Manager Mapping

Add to the **Employee** model information about their **manager** and a list of **employees** that they **manage**. It is **possible** for an employee to have **no** **manager**. Create another data transfer object, which treats employees as managers:

* **ManagerDto** – first name, last name, list of EmployeeDtos that he/she is in charge of and their count

Add the following commands to your console application:

* **SetManager** <**employeeId**> <**managerId**> – sets the second employee to be a manager of the first employee
* **ManagerInfo** <**employeeId**> – prints on the console information about a manager in the following format:

### Example

|  |
| --- |
| **Sample output** |
| Steve Jobbsen | Employees: 2  - Stephen Bjorn - $4300.00  - Kirilyc Lefi - $4400.00 |
| Carl Kormac | Employees: 14  - Jurgen Straus - $1000.45  - Moni Kozinac - $2030.99  - Kopp Spidok - $2000.21  - … |

## Projection

Add a few employees to your database with their birthdays. Create a command "**ListEmployeesOlderThan** <**age**>" which lists all employees older than given age and their managers. Order them **by salary descending.** Add the necessary DTOs and commands to your application.

### Example

|  |
| --- |
| **Sample output** |
| Steve Jobbsen - $6000.20 - Manager: [no manager]  Kirilyc Lefi - $4400.00 - Manager: Jobbsen  Stephen Bjorn - $4300.00 - Manager: Jobbsen |