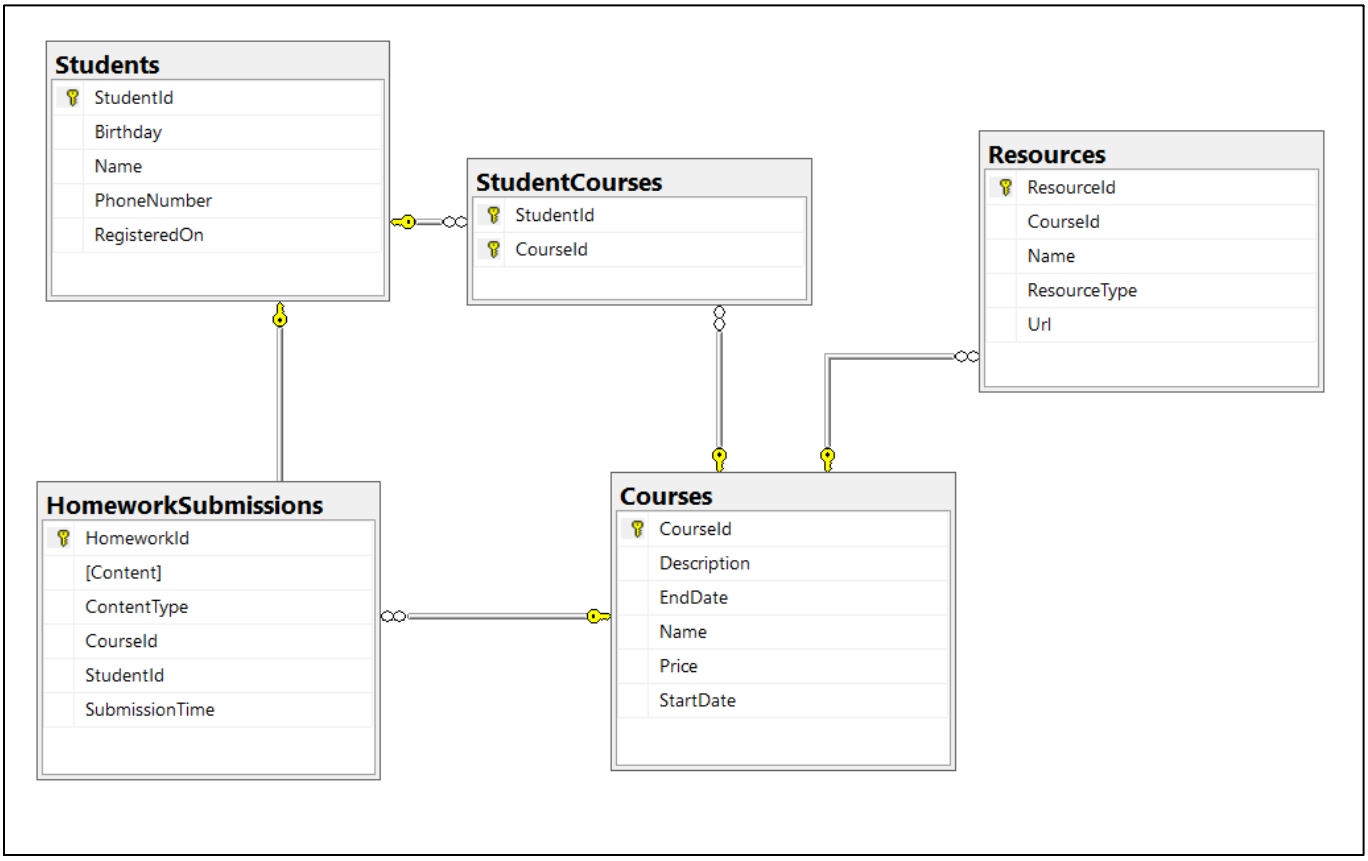
Exercises: Entity Framework Core

This document defines the **exercise assignments** for the entity framework core 02.

1. Student System

Your task is to create a database for the **Student System**, using the **EF Core Code First** approach. It should look like this:



Students
- StudentId
- Birthday
- Name
- PhoneNumber
- RegisteredOn

StudentCourses
- StudentId
- CourseId

Resources
- ResourceId
- CourseId
- Name
- ResourceType
- Url

HomeworkSubmissions
- HomeworkId
- [Content]
- ContentType
- CourseId
- StudentId
- SubmissionTime

Courses
- CourseId
- Description
- EndDate
- Name
- Price
- StartDate

Constraints

Your **namespaces** should be:

- **`P01_StudentSystem`** – Project Name
- **`P01_ StudentSystem.Data`** – (New folder named Data) for your DbContext
- **`P01_ StudentSystem.Models`** – (New folder named Models) for your models

Your **models** should be:

- **`StudentSystemContext`** – your DbContext
- **`Student`**:
  - StudentId
  - Name (up to 100 characters, unicode)
  - PhoneNumber (exactly 10 characters, not unicode, not required)
  - RegisteredOn (DateTime)
  - Birthday (not required)

- **`Course`**:
  - CourseId
  - Name (up to 80 characters, unicode)
  - Description (unicode, not required)
  - StartDate
  - EndDate
  - Price
- **`Resource`**:
  - ResourceId
  - Name (up to 50 characters, unicode)
  - Url (not unicode)
  - ResourceType (enum – can be Video, Presentation, Document or Other)
  - CourseId
- **`Homework`**:
  - HomeworkId
  - Content (string, linking to a file, not unicode)
  - ContentType (enum – can be Application, Pdf or Zip)
  - SubmissionTime
  - StudentId
  - CourseId
- **`StudentCourse`** – mapping class between **Students** and **Courses**

Table relations:

- **One student** can have **many CourseEnrollments**
- **One student** can have **many HomeworkSubmissions**
- **One course** can have **many StudentsEnrolled**
- **One course** can have **many Resources**
- **One course** can have **many HomeworkSubmissions**

2. Seed Some Data in the Database

Write a **seed method** that fills the database with sample data.
Fill a few **students**, **courses**, **resources** and **homework submissions**.

3. Migration

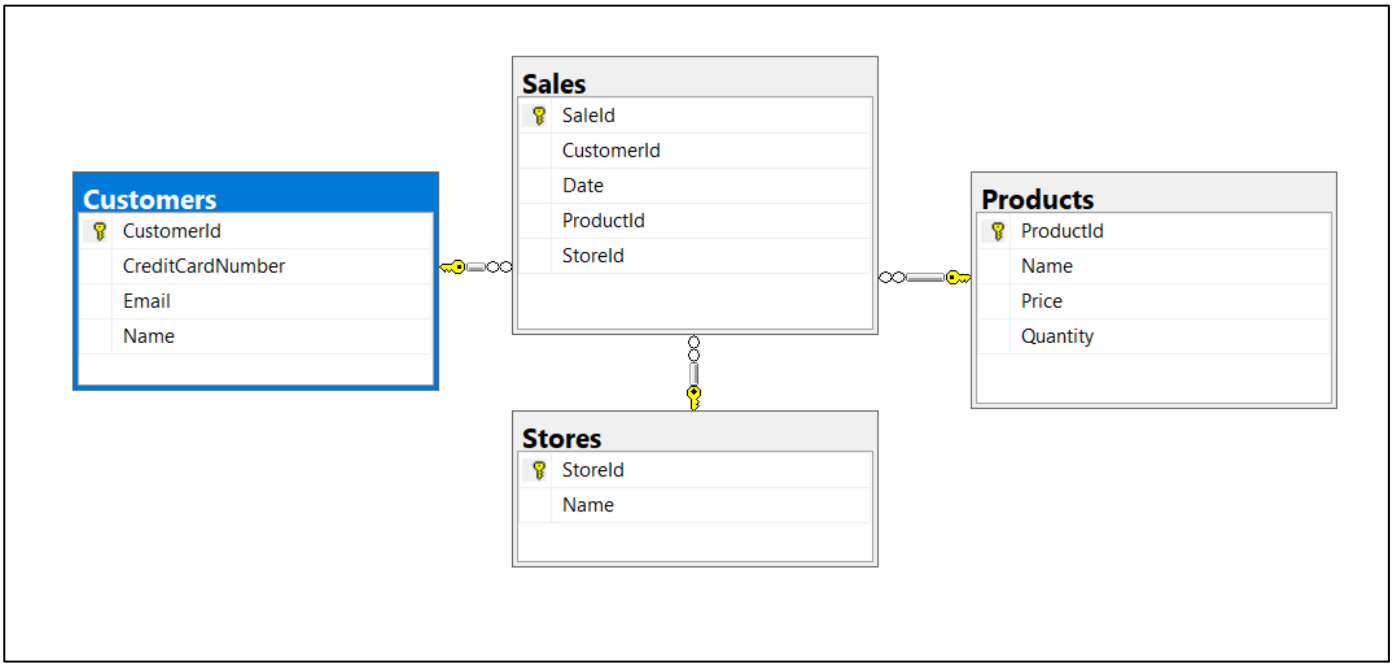
Add new migration. The migration should be named: **"InitialCreate" and run** the project.

Bonus

Create a console application that reads information about **courses** and **students** and store it.

4. Sales Database (New Separated Project)

Create a database for storing data about sales using the Code First approach. The database should look like this:



Sales
- SaleId
- CustomerId
- Date
- ProductId
- StoreId

Customers
- CustomerId
- CreditCardNumber
- Email
- Name

Products
- ProductId
- Name
- Price
- Quantity

Stores
- StoreId
- Name

Constraints

Your **namespaces** should be:

- `P02_SalesDatabase` – Project Name
- `P02_SalesDatabase.Data` – (New folder named Data) for your DbContext
- `P02_SalesDatabase.Models` – (New folder named Models) for your models

Your **classes** should be:

- `SalesContext` – your DbContext
- `Product`:
  - ProductId
  - Name (up to 50 characters, unicode)
  - Quantity (real number)
  - Price
  - Sales
- `Customer`:
  - CustomerId
  - Name (up to 100 characters, unicode)
  - Email (up to 80 characters, not unicode)
  - CreaditCardNumber (string)
  - Sales
- `Store`:
  - StoreId
  - Name (up to 80 characters, unicode)
  - Sales

- **Sale**:
    - SaleId
    - Date
    - Product
    - Customer
    - Store

Bonus Task

Write a **seed method** that fills the database with sample data (randomly generated).

5. Migration

Add new migration. The migration should be named: **"InitialCreate" and run** the project.

6. Products Migration

For table **Products** add string column **Description**, up to 250 symbols. Use migrations. The migration should be named: **"ProductsAddColumnDescription"**. Add a default value for the description property: **"No description"**.

7. Sales Migration

For table **Sales** make **Date** column with default value **GETDATE()** function, called from the database, not the application. Use explicit migration. Do **not** use **DateTime.Now**! Name the migration "**SalesAddDateDefault**".

After that, open your table data and see if the default value is applied or not.

8. Upload Projects

Use GitHub to upload your two projects.