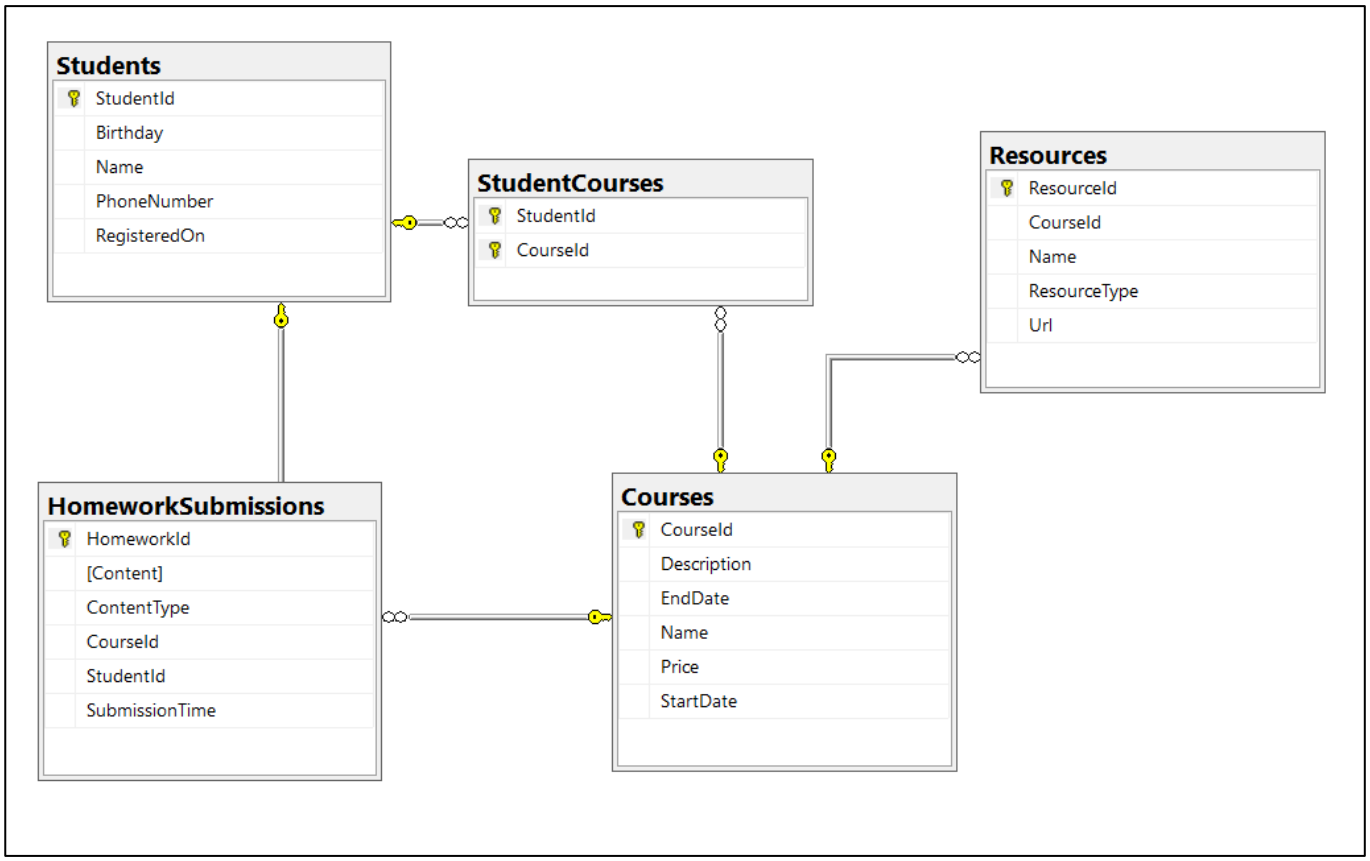
Exercises: Entity Relations

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:



Constraints

Your **namespaces** should be:

- **`P01_StudentSystem`** – for your Startup class, if you have one
- **`P01_StudentSystem.Data`** – for your DbContext
- **`P01_StudentSystem.Data.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`**
  - **`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**

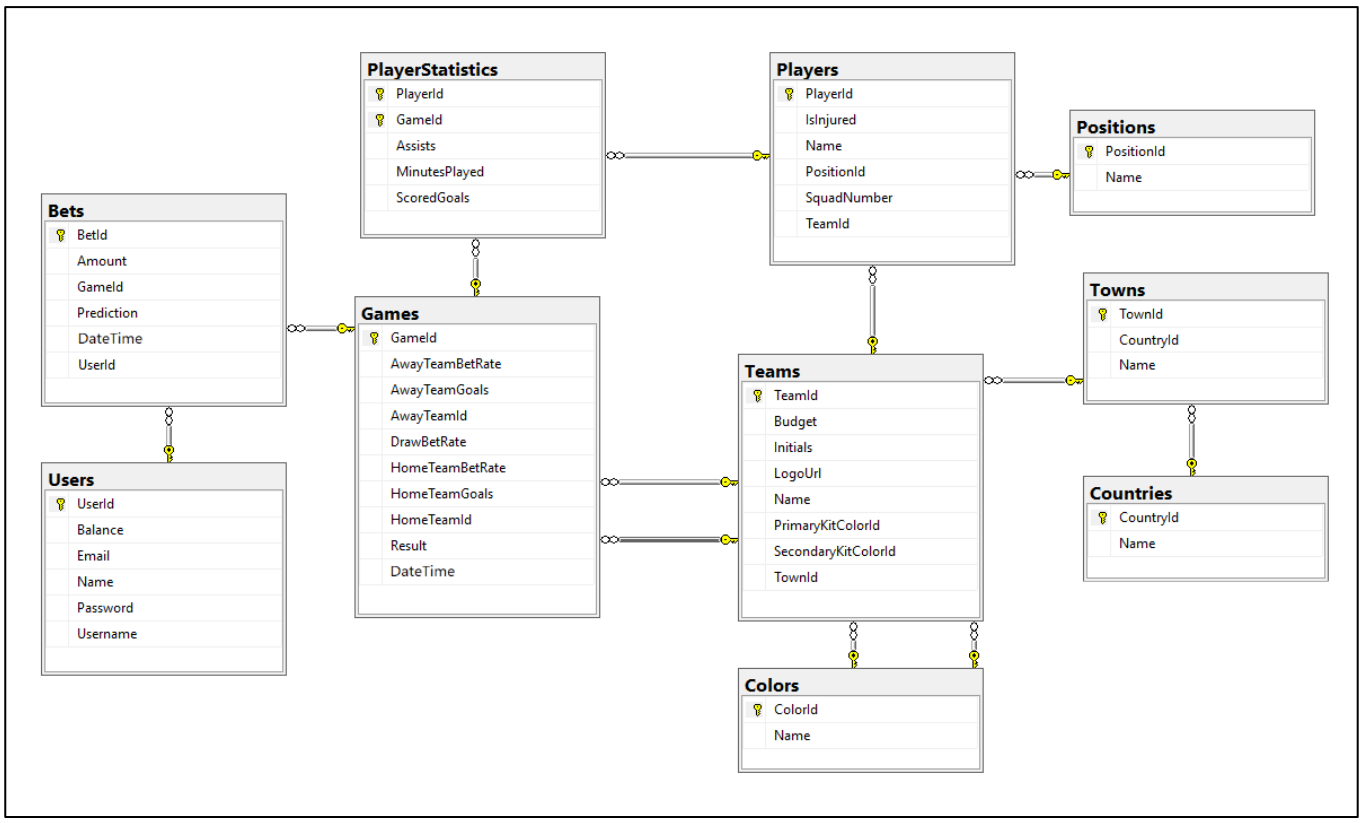
Table relations:

- **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**

You will need a constructor, accepting **DbContextOptions** to test your solution in **Judge**!

2. Football Betting

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



Constraints

Your **namespaces** should be:

- **P03_FootballBetting** – for your Startup class, if you have one
- **P03_FootballBetting.Data** – for your DbContext
- **P03_FootballBetting.Data.Models** – for your models

Your models should be:

- **FootballBettingContext** – your DbContext
- **Team** – **TeamId**, **Name**, **LogoUrl**, **Initials** (JUV, LIV, ARS…), **Budget**, **PrimaryKitColorId**, **SecondaryKitColorId**, **TownId**
- **Color** – **ColorId**, **Name**
- **Town** – **TownId**, **Name**, **CountryId**
- **Country** – **CountryId**, **Name**
- **Player** – **PlayerId**, **Name**, **SquadNumber**, **TeamId**, **PositionId**, **IsInjured**
- **Position** – **PositionId**, **Name**
- **PlayerStatistic** – **GameId**, **PlayerId**, **ScoredGoals**, **Assists**, **MinutesPlayed**
- **Game** – **GameId**, **HomeTeamId**, **AwayTeamId**, **HomeTeamGoals**, **AwayTeamGoals**, **DateTime**, **HomeTeamBetRate**, **AwayTeamBetRate**, **DrawBetRate**, **Result**)
- **Bet** – **BetId**, **Amount**, **Prediction**, **DateTime**, **UserId**, **GameId**
- **User** – **UserId**, **Username**, **Password**, **Email**, **Name**, **Balance**

Table relationships:

- **A Team** has one **PrimaryKitColor** and one **SecondaryKitColor**
- **A Color** has **many PrimaryKitTeams** and **many SecondaryKitTeams**
- **A Team** residents in one **Town**

- **A `Town`** can host **several `Teams`**
- **A `Game`** has one **`HomeTeam`** and one **`AwayTeam`** and a **`Team`** can have **many `HomeGames`** and **many `AwayGames`**
- **A `Town`** can be placed in **one `Country`** and a **`Country`** can have many **`Towns`**
- **A `Player`** can play for **one `Team`** and **one `Team`** can have many **`Players`**
- **A `Player`** can play at one **`Position`** and one **`Position`** can be played by **many `Players`**
- **One `Player`** can play in **many `Games`** and in each **`Game`**, **many `Players`** take part (both collections must be named PlayerStatistics)
- **Many `Bets`** can be placed on **one `Game`**, but **a `Bet`** can be only on **one `Game`**
- Each bet for given game must have **`Prediction`** result
- **A `Bet`** can be placed by only **one `User`** and one **`User`** can place many **`Bets`**

Separate the **models**, **data** and **client** into **different layers** (projects).