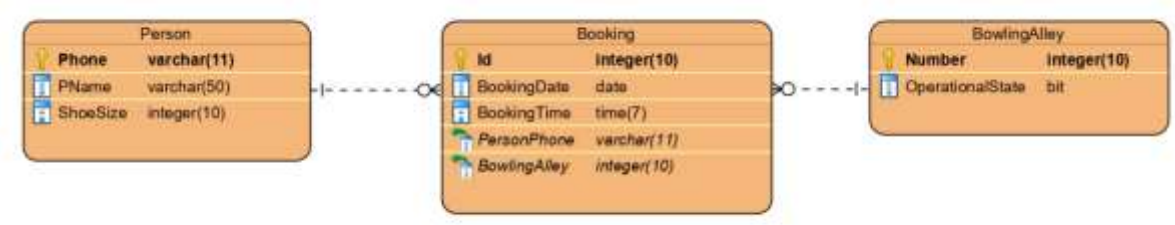
Opgave: Database & ConsoleApplikation

Du skal lave en ny applikation til opgaven med BowlingBanen (se DatabaseOgConsoleApp), dvs du kan benytte de samme tre tabeller, du lavede der.



Du skal lave en prototype på et system til en bowlingbane. Dette indebærer; Et lille program, der kan vise værdierne.

Lave et lille program, hvor du benytter EntityFramework

Til det skal du have model-klasser, samt selve ConsoleApp'en.

Trin 1: Model-klasse(r)

Du skal lave et projekt **'Class Library (.Net 5.0)'**

Du skal **IKKE** selv kode de 3 model klasser. De bliver genereret af EntityFramworket.

1.A Installer værktøjet: EF Core Power Tool

For at få lavet klasserne ud fra dine tabeller skal du benytte et tredje part værktøj.
Du skal downloade værktøjet '**EF Core Power Tool**' fra
https://marketplace.visualstudio.com/items?itemName=ErikEJ.EFCorePowerTools

Du bliver vist nok nødt til at lukke for Visual Studio for at installere vsix-filen.

1.B Installer NuGet pakke: Microsoft.EntityFrameWorkCore.SqlServer

Installer pakken '**Microsoft.EntityFrameWorkCore.SqlServer**'

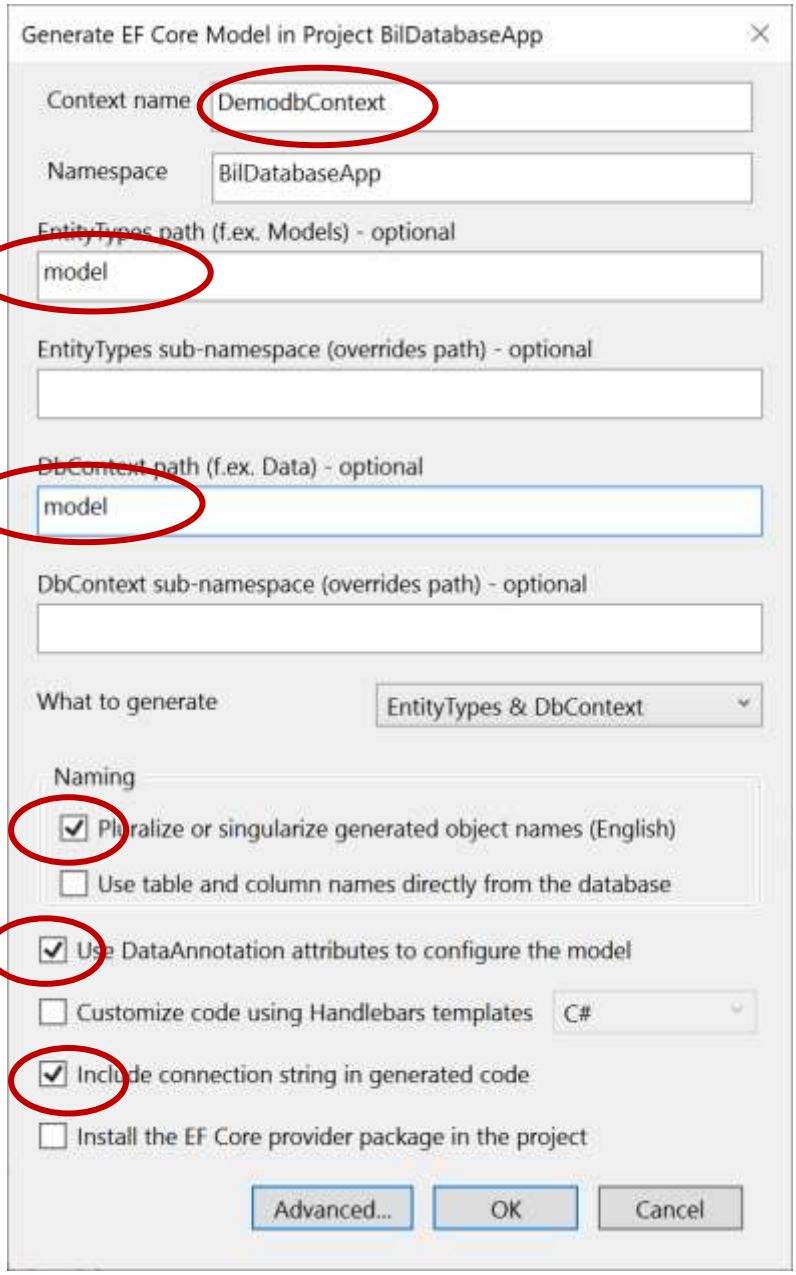
MEN benyt version 5.xxx, hvilket ikke er den seneste men version 6.xxx, som kun virker sammen med .Net 6.0.

1.C Lav forbindelse og modelklasser

Du skal åbne 'EF Core power tool' dvs. højre klik på dit projekt -> EF Core Power Tools -> Reverse Engineer.

Nu skal du angive din database server (find den i din 'SQL server object explorer'), angive database, samt login (dvs. windows)

Så er det bare at klikke OK, vælg dine tabeller.
Derefter skal du navngive din DbContext (her i eksemplet DemodbContext, men find selv et navn) og angiv hvor dine modelklasser og dbcontext skal ligge – passende i mappen 'model' se nedenunder:



Du skulle nu i model have fire filer en xxxxDBContext samt de tre model klasser Person, Booking og BowlingAlley.

Tag og kig på dem, hvordan er der defineret?

Trin 2: Console Applikation

Du skal lave endnu et projekt '**Console Application (.Net 5.0)**', hvor du i første omgang skal lave en database-forbindelse som lave CRUD på Person (tabellen).

Husk at lav en reference (under dependencies) til dit class-library.

Lav en klasse AWorker med en Start-metode og kald denne fra program-main metoden.

I projektet skal du lave et interface '**IPersonService**' med følgende 5 metoder:

```
private List<Person> GetAllPersons()

private Person GetPersonByPhoneNo(string phone)

private Person AddPerson(Person person)

private Person DeletePerson(string phone)

private Person UpdatePerson(string phone, Person person)
```

I projektet skal du lave en klasse '**PersonService**' der implementerer interfacet IPersonService ved brug af EntityFrameworket som forbindelse til databasen.

I klassen skal du lave et instansfelt af typen xxxxDBContext med f.eks. navnet '_db'. Dette felt initialiserer du i konstruktøren.

Når du skal implementerer metoderne gør du brug af '_db.Persons' til at tilgå tabellen i databasen. Husk at kalde _db.SaveChanges(), når du opretter, sletter eller retter i personer.

Du skal oprette et objekt af din PersonService klasse og kalde de forskellige metoder fra Start-metoden.

Ekstra 1: Lave det samme for Booking-model klassen

Bøvlet Ekstra 2: I Scrumboard projektet implementer en klasse UserStoryServiceEF, som implementer interfacet IUserStoryService ved brug af EntityFrameworket.
Det er bøvlet fordi du kommer til at generere en ny klasse UserStory, hvorved den gamle klasse bliver overflødig og skal erstattes af den nye model-klasse