Exercises: LINQ

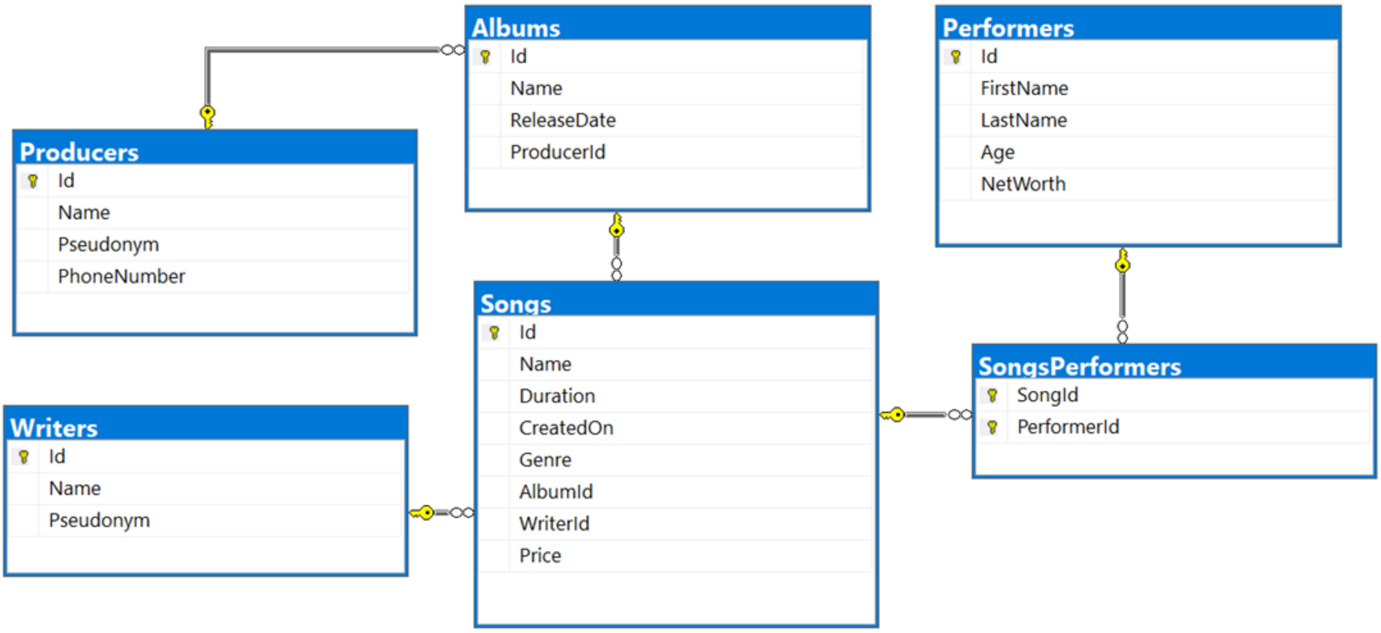
You can check your solutions here: https://judge.softuni.bg/Contests/3201/LINQ.

MusicHub

People love listening to music, but they see that YouTube is getting older and older. You want to make people happy and you've decided to make a better version of YouTube – **MusicHub**. It's time for you to start coding. Good luck and impress us.

1. MusicHub Database

You must create a **database** for a **MusicHub**. It should look like this:



Constraints

Your **namespaces** should be:

- `MusicHub` – for your **StartUp** class, if you have one
- `MusicHub.Data` – for your **DbContext**
- `MusicHub.Data.Models` – for your **Models**

Your **models** should be:

Song

- `Id` – **Integer**, **Primary Key**
- `Name` – **Text** with **max length 20** (**required**)
- `Duration` – **TimeSpan** (**required**)
- `CreatedOn` – **Date** (**required**)
- `Genre` – **Genre enumeration with possible values: "Blues, Rap, PopMusic, Rock, Jazz" (required)**
- `AlbumId` – **Integer**, **Foreign key**
- `Album` – **The song's album**
- `WriterId` – `Integer, Foreign key (required)`

- **Writer** – **The song's writer**
- **Price** – **Decimal** (**required**)
- **SongPerformers** – Collection of type **SongPerformer**

Album

- **Id** – **Integer**, **Primary Key**
- **Name** – **Text** with **max length 40** (**required**)
- **ReleaseDate** – **Date** (**required**)
- **Price** – **calculated property** (**the sum of all song prices in the album**)
- **ProducerId** – **integer, Foreign key**
- **Producer** – **the album's producer**
- **Songs** – collection of all **Songs** in the **Album**

Performer

- **Id** – **Integer**, **Primary Key**
- **FirstName** – **text** with **max length 20** (**required**)
- **LastName** – **text** with **max length 20** (**required**)
- **Age** – **Integer** (**required**)
- **NetWorth – decimal** (**required**)
- **PerformerSongs** – collection of type **SongPerformer**

Producer

- **Id** – **Integer**, **Primary Key**
- **Name** – **text** with **max length 30 (required)**
- **Pseudonym** – **text**
- **PhoneNumber** – **text**
- **Albums** – collection of type **Album**

Writer

- **Id** – **Integer**, **Primary Key**
- **Name** – **text** with **max length 20** (**required**)
- **Pseudonym** – **text**
- **Songs** – collection of type **Song**

SongPerformer

- **SongId** – **Integer**, **Primary Key**
- **Song** – the performer's **Song** (**required**)
- **PerformerId** – **Integer, Primary Key**
- **Performer** – the song's **Performer (required)**

Table relations

- **One Song** can have **many Performers**
- **One Permormer** can have **many Songs**
- **One Writer** can have **many Songs**
- **One Album** can have **many Songs**
- **One Producer** can have **many Albums**

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

2. All Albums Produced by Given Producer

You need to write method string ExportAlbumsInfo(MusicHubDbContext context, int producerId) in the **StartUp** class that receives a **Producer Id**. Export **all albums** which are **produced by** the provided **Producer Id**. For each **Album**, get the **Name**, **Release date** in format "**MM/dd/yyyy**", **Producer Name**, the **Album Songs** with each **Song Name**, **Price** (**formatted to the second digit**) and the **Song Writer Name**. **Sort** the **Songs** by **Song Name** (**descending**) and by **Writer** (**ascending**). At the end export **the Total Album Price** with exactly **two digits after the decimal place**. **Sort** the **Albums** by their **Total Price** (**descending**).

Example

Output(producerId = 9)

```
-AlbumName: Devil's advocate
-ReleaseDate: 07/21/2018
-ProducerName: Evgeni Dimitrov
-Songs:
---#1
---SongName: Numb
---Price: 13.99
---Writer: Kara-lynn Sharpous
---#2
---SongName: Ibuprofen
---Price: 26.50
---Writer: Stanford Daykin
-AlbumPrice: 40.49
…
```

3. Songs Above Given Duration

You need to write method string ExportSongsAboveDuration(MusicHubDbContext context, int duration) in the **StartUp** class that receives **Song** duration (**integer, in seconds**). Export the songs which are **above** the given duration. For each **Song**, export its **Name**, **Performer Full Name**, **Writer Name**, **Album Producer** and **Duration** (**in format**("**c**")). **Sort** the **Songs** by their **Name** (**ascending**), by **Writer** (**ascending**) and by **Performer** (**ascending**).

Example

Output(duration = 4)

```
-Song #1
---SongName: Away
---Writer: Norina Renihan
---Performer: Lula Zuan
---AlbumProducer: Georgi Milkov
---Duration: 00:05:35
-Song #2
---SongName: Bentasil
---Writer: Mik Jonathan
---Performer: Zabrina Amor
```

```
---AlbumProducer: Dobromir Slavchev
---Duration: 00:04:03
…
```