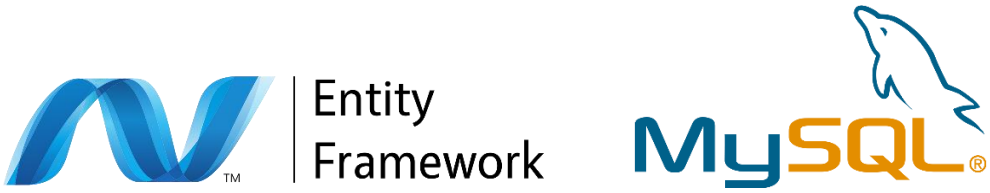
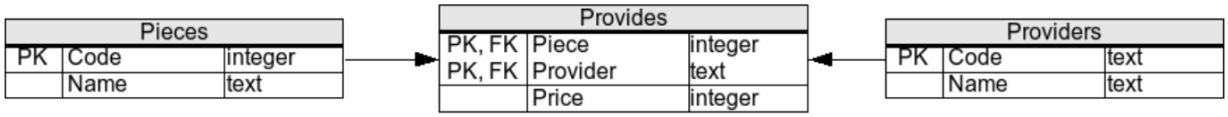
Gegevensbeheer en beveiliging – Les 7

In deze les maken we een connectie met een MySql database via het Entity Framework. MySql (en ook MariaDB) is een uitgebreide database. Voor een groter project heb je misschien de mogelijkheden van deze database nodig in plaats van de beperktere SqLite database. MySql ondersteunt bijvoorbeeld meerdere users met verschillende bevoegdheden.



Database aanmaken

We maken eerst een database aan en laten Entity de nodige C# classes genereren (*scaffolding*). Als voorbeeld nemen we deze database:
https://en.wikibooks.org/wiki/SQL_Exercises/Pieces_and_providers



De SQL-code om de database te maken staat hieronder. Je kan deze ingeven in een programma zoals MySQL Workbench, de MySQL console of via een systeem zoals PhpMyAdmin. Je hebt dan wel een draaiend MySQL proces nodig (dat kan je starten via een packet zoals XAMP, WAMP of MAMP).

```
CREATE DATABASE warehouse;
USE warehouse;

CREATE TABLE Pieces (
   Code INTEGER PRIMARY KEY NOT NULL AUTO_INCREMENT,
   Name TEXT NOT NULL
);

CREATE TABLE Providers (
  Code VARCHAR(255) PRIMARY KEY NOT NULL AUTO_INCREMENT,
  Name TEXT NOT NULL
);
```

```
CREATE TABLE Provides (
   Piece INTEGER,
   Provider VARCHAR(255),
   Price INTEGER NOT NULL,
   PRIMARY KEY(Piece, Provider),
   FOREIGN KEY (Piece) REFERENCES Pieces(Code),
   FOREIGN KEY (Provider) REFERENCES Providers(Code)
);
```

Eens de database gemaakt is kunnen we ze gaan vullen. Onderaan de site https://en.wikibooks.org/wiki/SQL_Exercises/Pieces_and_providers vind je deze handige code:

```
INSERT INTO Providers(Code, Name) VALUES('HAL','Clarke Enterprises');
INSERT INTO Providers(Code, Name) VALUES('RBT','Susan Calvin Corp.');
INSERT INTO Providers(Code, Name) VALUES('TNBC','Skellington Supplies');

INSERT INTO Pieces(Code, Name) VALUES(1,'Sprocket');
INSERT INTO Pieces(Code, Name) VALUES(2,'Screw');
INSERT INTO Pieces(Code, Name) VALUES(3,'Nut');
INSERT INTO Pieces(Code, Name) VALUES(4,'Bolt');

INSERT INTO Provides(Piece, Provider, Price) VALUES(1,'HAL',10);
INSERT INTO Provides(Piece, Provider, Price) VALUES(1,'RBT',15);
INSERT INTO Provides(Piece, Provider, Price) VALUES(2,'HAL',20);
INSERT INTO Provides(Piece, Provider, Price) VALUES(2,'RBT',15);
INSERT INTO Provides(Piece, Provider, Price) VALUES(2,'TNBC',14);
INSERT INTO Provides(Piece, Provider, Price) VALUES(3,'RBT',50);
INSERT INTO Provides(Piece, Provider, Price) VALUES(3,'TNBC',45);
INSERT INTO Provides(Piece, Provider, Price) VALUES(4,'HAL',5);
INSERT INTO Provides(Piece, Provider, Price) VALUES(4,'RBT',7);
```

Scaffolding van de C# code

In plaats van de classes en het DbContext bestand zelf te gaan schrijven, zoals we bij SqLite deden, gaan we de code door het Entity framework laten genereren. Dit proces noemt *scaffolding*, het snel in stelling zetten van de benodigde code (scaffold = ondersteuning of stelling).

Omdat we vertrekken vanuit de database is dit *database first scaffolding*. Een andere optie is om te beginnen met C# classes en van daaruit de database te laten genereren, dat wordt dan *code first scaffolding* genoemd. Code first scaffolding gaan we in deze cursus niet toepassen, maar je vindt ongetwijfeld voorbeelden online van deze techniek.

We installeren eerst de nodige NuGet packages via de NuGet Package Manager Console (Tools > NuGet Package Manager > Package Manager Console):

```
Install-Package MySql.EntityFrameworkCore -Version 5.0.0+m8.0.23
```

Om de database te scaffolden hebben we ook nog de tools nodig, die installeer je zo:

```
Install-Package Microsoft.EntityFrameworkCore.Tools
```

Merk op dat de tools niet specifiek zijn voor MySQL. Je kan hier ook een SqLite database of een andere database mee scaffolden.

Meer heb je niet nodig om aan de slag te gaan. We gaan nu de modellen en DbContext laten genereren met het commando Scaffold-DbContext. Onderstaande code is een voorbeeld voor een lokale WAMP-omgeving. WAMP heeft geen paswoord ingesteld voor de database-toegang. Mogelijk moet je in jouw omgeving het wachtwoord nog toevoegen. Voeg dan achter user=root; ook nog *pwd=je_wachtwoord;* toe.

```
Scaffold-DbContext "server=localhost;database=warehouse;user=root;" MySql.EntityFrameworkCore
```

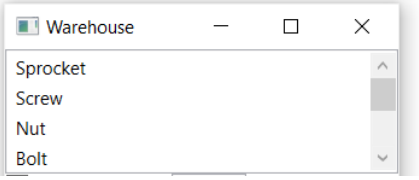
Als alles goed is gegaan zijn er nu 4 bestanden aangemaakt die klaar staan om te communiceren met de database: Piece.cs, Provider.cs, Provide.cs (de namen komen van de tables) en warehouseContext.cs (die naam komt van de database). De code die wordt gegenereerd bevat nog allerlei informatie over de tabellen zoals primary keys, veld-datatype en dergelijke. Je kan vanaf nu using statements gebruiken zoals je dat gewoon bent met SqLite (zie eerste document van dit vak). Onderstaand code-voorbeeld toont al de items uit de table pieces in een ListBox (met naam lboxPieces).

```
using(var db = new WarehouseContext())
{
    var pcs = db.Pieces.ToList();

    foreach(var p in pcs)
    {
        lboxPieces.Items.Add(p);
    }
}
```

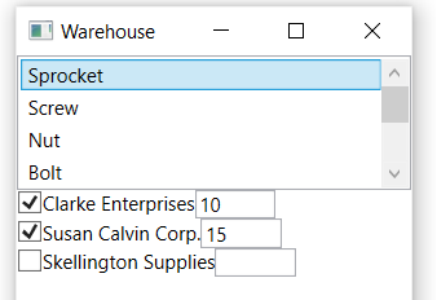
Oefening 1 – Warehouse ListBox Pieces

Toon al de namen van de verschillende items uit de table pieces in een ListBox.



Dynamisch UI elementen toevoegen aan een WPF applicatie

Stel dat je de gegevens van de table providers wil tonen in een StackPanel met een CheckBox (om aa te duiden of het bedrijf dit product levert) en een TextBox voor de prijs, zoals hier:



Dan moeten we dus de tabel providers overlopen en voor elke rij in de tabel een nieuw item toevoegen aan onze StackPanel. Elke "rij" in de StackPanel bestaat uit 3 elementen (CheckBox, TextBlock en TextBox) die naast elkaar staan. Voor elke rij hebben we dus ook een StackPanel nodig om die 3 elementen naast elkaar te zetten.

Hoe we een item toevoegen aan een ListBox hebben we al gezien. Om een element toe te voegen aan een StackPanel gebruiken we Children.Add(). Als we in onze XAML een StackPanel hebben gezet met als naam *spanelProvider*…

```
<StackPanel x:Name="spanelProviders"></StackPanel>
```

… kunnen we hier een StackPanel aan toevoegen met deze code:

```
var panelRow = new StackPanel(); // panelRow is een zelfgekozen naam
panelRow.Orientation = Orientation.Horizontal;
spanelProviders.Children.Add(panelRow);
```

We hebben nu een horizontale StackPanel *dynamisch* toegevoegd aan de reeds bestaande StackPanel, de toegevoegde StackPanel is voorlopig leeg. We kunnen op dezelfde manier verder gaan en de CheckBox toevoegen aan de net toegevoegde *panelRow* StackPanel:

```
var cbox = new CheckBox();
panelRow.Children.Add(cbox);
```

Je kan de instellingen van de elementen wijzigen met de punt notatie, de instellingen zijn immers properties van de elementen. We deden dit reeds met bovenstaande StackPanel (.Orientation = …).

In XAML kan je elementen een naam geven (x:Name="…"). Dat kan ook voor een dynamisch gemaakt element. Je stelt daarvoor de Name property in. Om het helemaal goed te laten gaan

moet je ook nog RegisterName aanroepen, pas dan kan je de naam ergens anders gaan gebruiken in je programma om het element te bereiken (je kan natuurlijk ook elementen in variabelen opslaan in je programma, maar soms werkt een naam handiger).

```
var cbox = new CheckBox();
cbox.Name = "MijnCheckBox";
RegisterName(cbox.Name, cbox);
```

Om je CheckBox terug te vinden kan je de FindName functie gebruiken. Om te zoeken naar het element met naam *MijnCheckBox* in de StackPanel met naam *spanelProviders* gebruik je deze code:

```
CheckBox cbox = spanelProviders.FindName("MijnCheckBox") as CheckBox;
```

Oefening 2 – Warehouse met pieces, providers en prijs

Maak de oefening verder af zodat de providers worden getoond in de StackPanel met een CheckBox, TextBlock en TextBox. Zorg ervoor dat de CheckBoxes checked staan als het product wordt geleverd door het bedrijf (de “provider”) en toon ook de juiste prijs in dat geval. Als het product niet wordt aangeboden door de provider moet de CheckBox unchecked zijn.

Mogelijk komt onderstaande code van pas, deze zet al de CheckBoxes in spanelProviders uit en verwijdert al de prijzen uit de TextBoxes.

```
foreach(StackPanel row in spanelProviders.Children)
{
    row.Children.OfType<CheckBox>().FirstOrDefault().IsChecked = false;
    row.Children.OfType<TextBox>().FirstOrDefault().Text = "";
}
```