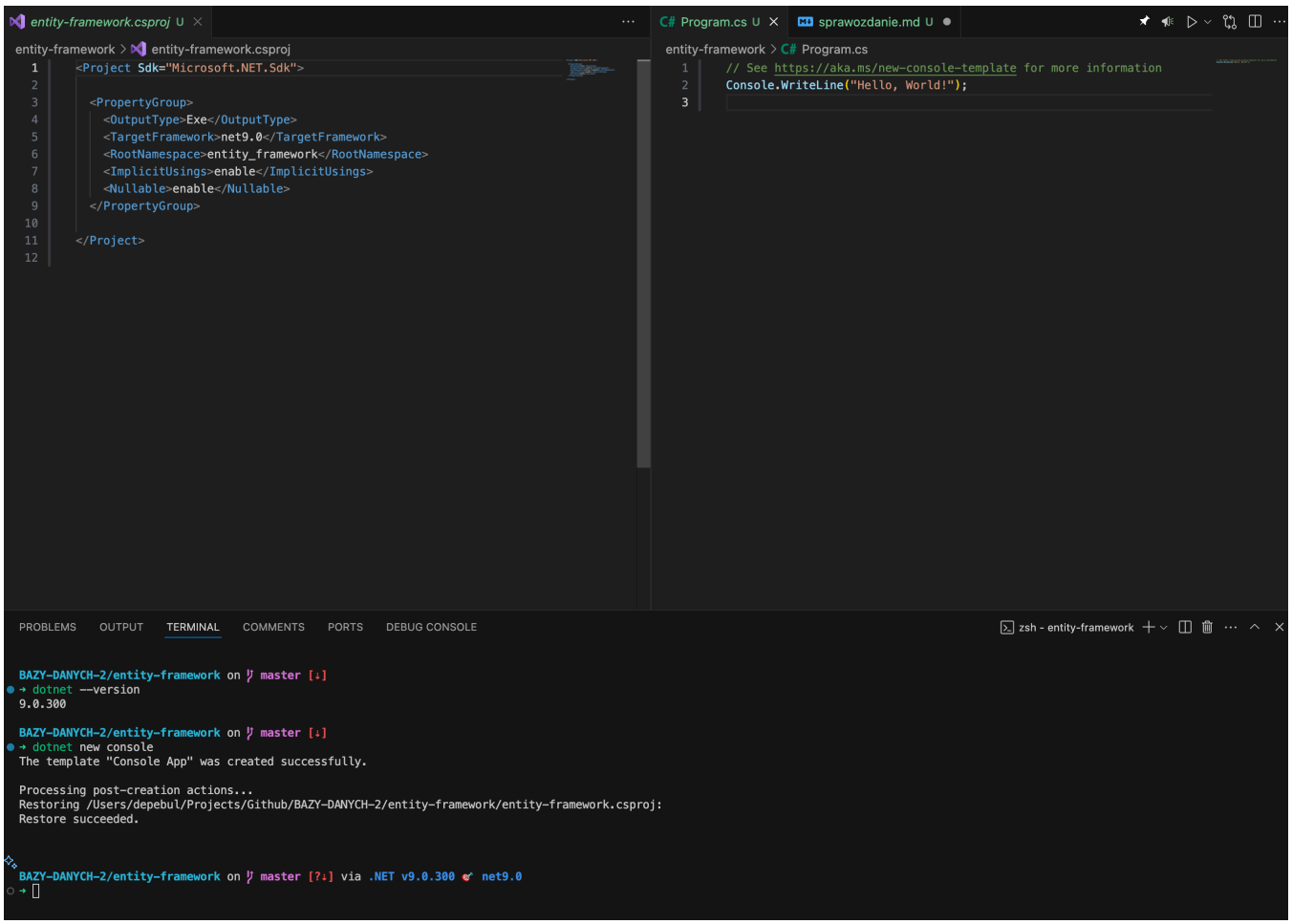
Entity Framework

Autorzy: Szymon Migas, Dawid Żak

Część I

Weryfikacja wersji dotnet framework oraz utworzenie nowej aplikacji konsolowej w edytorze Visual Studio Code.



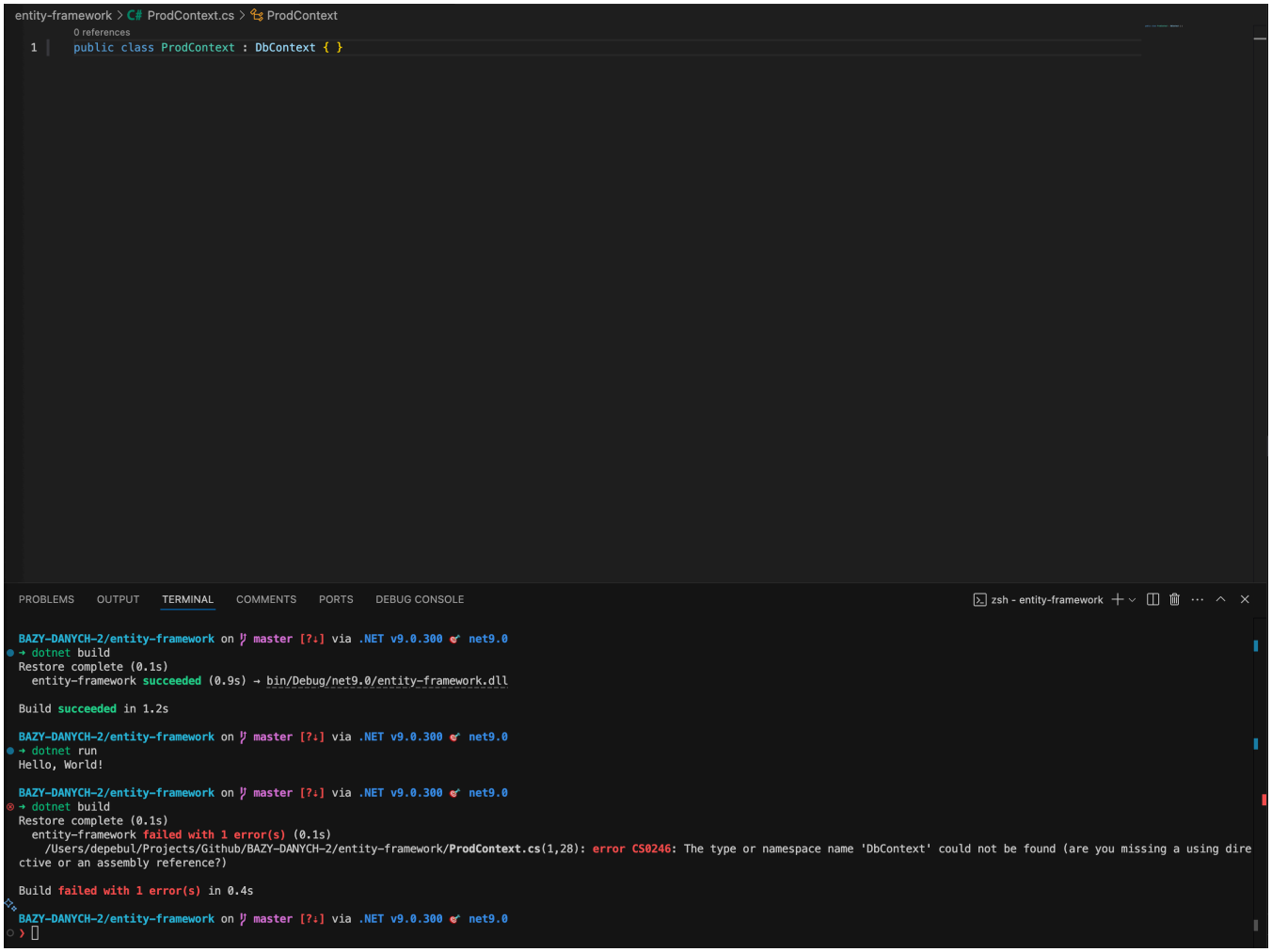
Próba zbudowania oraz uruchomienia aplikacji:

```
BAZY-DANYCH-2/entity-framework on  master [?⇣] via .NET v9.0.300 🎯 net9.0
➜ dotnet build
Restore complete (0.1s)
  entity-framework succeeded (0.9s) → bin/Debug/net9.0/entity-framework.dll

Build succeeded in 1.2s

BAZY-DANYCH-2/entity-framework on  master [?⇣] via .NET v9.0.300 🎯 net9.0
➜ dotnet run
Hello, World!
```

Stworzona klasa `Product`

```
entity-framework > C# Product.cs > Product
        0 references
1       public class Product
2       {
            0 references
3           public int ProductId { get; set; }
            0 references
4           public string? ProductName { get; set; }
            0 references
5           public int UnitsInStock { get; set; }
6       }
```

W pkiku `prodContext.cs` stworzyliśmy klase `ProdContext` dziedziczącą po `DbContext`, która będzie odpowiedzialna za komunikację z bazą danych.



Po otrzymaniu komunikatu o błędzie, dodaliśmy do pliku

```
using Microsoft.EntityFrameworkCore;
```

Nie rozwiązało to jednak problemu.

```
entity-framework > ProdContext.cs > ProdContext
1   using Microsoft.EntityFrameworkCore;
    0 references
2   public class ProdContext : DbContext { }
```

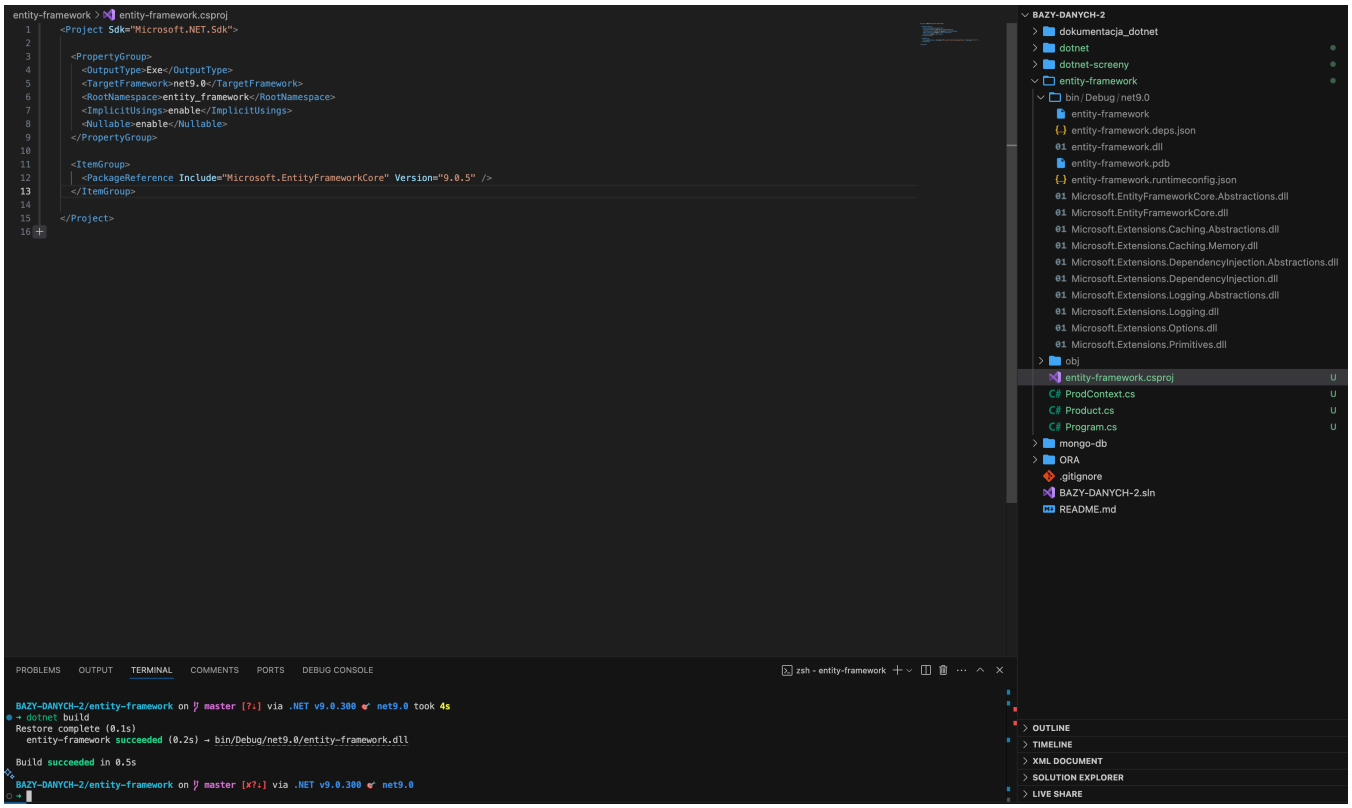
```
PROBLEMS  OUTPUT  TERMINAL  COMMENTS  PORTS  DEBUG CONSOLE                zsh - entity-framework

 BAZY-DANYCH-2/entity-framework on  master [?↓] via .NET v9.0.300  net9.0
❯ dotnet build
Restore complete (0.1s)
  entity-framework failed with 2 error(s) (0.1s)
    /Users/depebul/Projects/Github/BAZY-DANYCH-2/entity-framework/ProdContext.cs(1,17): error CS0234: The type or namespace name 'EntityFrameworkCore' does not exist in the namespace 'Micro
soft' (are you missing an assembly reference?)
    /Users/depebul/Projects/Github/BAZY-DANYCH-2/entity-framework/ProdContext.cs(2,28): error CS0246: The type or namespace name 'DbContext' could not be found (are you missing a using dire
ctive or an assembly reference?)

Build failed with 2 error(s) in 0.4s

 BAZY-DANYCH-2/entity-framework on  master [?↓] via .NET v9.0.300  net9.0
❯
```

Po dodaniu dod projektu pakietu komendą

```
dotnet add package Microsoft.EntityFrameworkCore
```

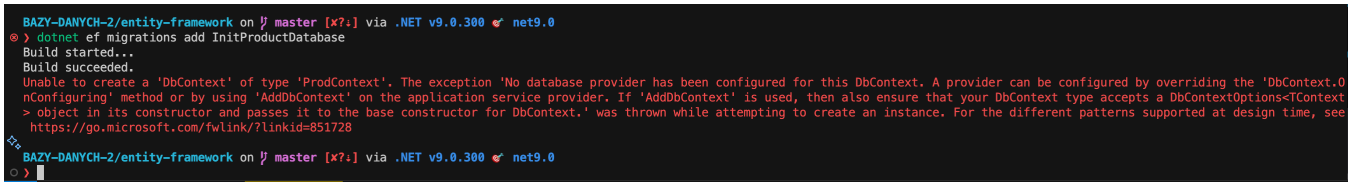
```
 BAZY-DANYCH-2/entity-framework on  master [?↓] via .NET v9.0.300  net9.0
❯ dotnet add package Microsoft.EntityFrameworkCore

Build succeeded in 0.2s
info : X.509 certificate chain validation will use the fallback certificate bundle at '/usr/local/share/dotnet/sdk/9.0.300/trustedroots/codesignctl.pem'.
info : X.509 certificate chain validation will use the fallback certificate bundle at '/usr/local/share/dotnet/sdk/9.0.300/trustedroots/timestampctl.pem'.
info : Adding PackageReference for package 'Microsoft.EntityFrameworkCore' into project '/Users/depebul/Projects/Github/BAZY-DANYCH-2/entity-framework/entity-framework.csproj'.
info :   GET https://api.nuget.org/v3/registration5-gz-semver2/microsoft.entityframeworkcore/index.json
info :   OK https://api.nuget.org/v3/registration5-gz-semver2/microsoft.entityframeworkcore/index.json 190ms
info :   GET https://api.nuget.org/v3/registration5-gz-semver2/microsoft.entityframeworkcore/page/0.0.1-alpha/3.1.2.json
info :   OK https://api.nuget.org/v3/registration5-gz-semver2/microsoft.entityframeworkcore/page/0.0.1-alpha/3.1.2.json 220ms
info :   GET https://api.nuget.org/v3/registration5-gz-semver2/microsoft.entityframeworkcore/page/3.1.3/6.0.0-preview.6.21352.1.json
info :   OK https://api.nuget.org/v3/registration5-gz-semver2/microsoft.entityframeworkcore/page/3.1.3/6.0.0-preview.6.21352.1.json 234ms
info :   GET https://api.nuget.org/v3/registration5-gz-semver2/microsoft.entityframeworkcore/page/6.0.0-preview.7.21378.4/7.0.17.json
info :   OK https://api.nuget.org/v3/registration5-gz-semver2/microsoft.entityframeworkcore/page/6.0.0-preview.7.21378.4/7.0.17.json 219ms
info :   GET https://api.nuget.org/v3/registration5-gz-semver2/microsoft.entityframeworkcore/page/7.0.18/10.0.0-preview.4.25258.110.json
info :   OK https://api.nuget.org/v3/registration5-gz-semver2/microsoft.entityframeworkcore/page/7.0.18/10.0.0-preview.4.25258.110.json 220ms
info : Restoring packages for /Users/depebul/Projects/Github/BAZY-DANYCH-2/entity-framework/entity-framework.csproj...
info :   GET https://api.nuget.org/v3/vulnerabilities/index.json
info :   OK https://api.nuget.org/v3/vulnerabilities/index.json 795ms
info :   GET https://api.nuget.org/v3-vulnerabilities/2025.05.28.05.39.59/vulnerability.base.json
info :   GET https://api.nuget.org/v3-vulnerabilities/2025.05.28.05.39.59/2025.06.01.11.40.17/vulnerability.update.json
info :   OK https://api.nuget.org/v3-vulnerabilities/2025.05.28.05.39.59/vulnerability.base.json 154ms
info :   OK https://api.nuget.org/v3-vulnerabilities/2025.05.28.05.39.59/2025.06.01.11.40.17/vulnerability.update.json 329ms
info : Package 'Microsoft.EntityFrameworkCore' is compatible with all the specified frameworks in project '/Users/depebul/Projects/Github/BAZY-DANYCH-2/entity-framework/entity-framework.csp
roj'.
info : PackageReference for package 'Microsoft.EntityFrameworkCore' version '9.0.5' added to file '/Users/depebul/Projects/Github/BAZY-DANYCH-2/entity-framework/entity-framework.csproj'.
info : Generating MSBuild file /Users/depebul/Projects/Github/BAZY-DANYCH-2/entity-framework/obj/entity-framework.csproj.nuget.g.props.
info : Generating MSBuild file /Users/depebul/Projects/Github/BAZY-DANYCH-2/entity-framework/obj/entity-framework.csproj.nuget.g.targets.
info : Writing assets file to disk. Path: /Users/depebul/Projects/Github/BAZY-DANYCH-2/entity-framework/obj/project.assets.json
log  : Restored /Users/depebul/Projects/Github/BAZY-DANYCH-2/entity-framework/entity-framework.csproj (in 1.21 sec).

 BAZY-DANYCH-2/entity-framework on  master [?↓] via .NET v9.0.300  net9.0 took 4s
➜
```

Spróbowaliśmy ponownie zbudować projekt, wszystko przebiegło pomyślnie.



Następnie utworzyliśmy kolekcję obiektów, którymi EF będzie zarządzać:

Próda przygotowania kodu odpowiedzialnego za migracje bazy danych:



Ponieważ otrzymaliśmy komunikat o błędzie, instalujemy Tools do Entity

Framework Core:

```
BAZY-DANYCH-2/entity-framework on master [✘?⇣] via .NET v9.0.300 🎯 net9.0
❯ dotnet tool install --global dotnet-ef
Tool 'dotnet-ef' is already installed.

BAZY-DANYCH-2/entity-framework on master [✘?⇣] via .NET v9.0.300 🎯 net9.0
➜
```

Kolejna próba wygenerowania migracji również kończy się błędem, więc dodajemy pakiet `Microsoft.EntityFrameworkCore.Design`:

```
BAZY-DANYCH-2/entity-framework on master [✘?⇣] via .NET v9.0.300 🎯 net9.0
❯ dotnet add package Microsoft.EntityFrameworkCore.Design

Build succeeded in 0.2s
info : X.509 certificate chain validation will use the fallback certificate bundle at '/usr/local/share/dotnet/sdk/9.0.300/trustedroots/codesignctl.pem'.
info : X.509 certificate chain validation will use the fallback certificate bundle at '/usr/local/share/dotnet/sdk/9.0.300/trustedroots/timestampctl.pem'.
info : Adding PackageReference for package 'Microsoft.EntityFrameworkCore.Design' into project '/Users/depebul/Projects/Github/BAZY-DANYCH-2/entity-framework/entity-framework.csproj'.
info :   GET https://api.nuget.org/v3/registration5-gz-semver2/microsoft.entityframeworkcore.design/index.json
info :   OK https://api.nuget.org/v3/registration5-gz-semver2/microsoft.entityframeworkcore.design/index.json 579ms
info :   GET https://api.nuget.org/v3/registration5-gz-semver2/microsoft.entityframeworkcore.design/page/0.0.1-alpha/3.1.3.json
info :   OK https://api.nuget.org/v3/registration5-gz-semver2/microsoft.entityframeworkcore.design/page/0.0.1-alpha/3.1.3.json 260ms
info :   GET https://api.nuget.org/v3/registration5-gz-semver2/microsoft.entityframeworkcore.design/page/3.1.4/6.0.0-preview.7.21378.4.json
info :   OK https://api.nuget.org/v3/registration5-gz-semver2/microsoft.entityframeworkcore.design/page/3.1.4/6.0.0-preview.7.21378.4.json 208ms
info :   GET https://api.nuget.org/v3/registration5-gz-semver2/microsoft.entityframeworkcore.design/page/6.0.0-rc.1.21452.10/7.0.18.json
info :   OK https://api.nuget.org/v3/registration5-gz-semver2/microsoft.entityframeworkcore.design/page/6.0.0-rc.1.21452.10/7.0.18.json 182ms
info :   GET https://api.nuget.org/v3/registration5-gz-semver2/microsoft.entityframeworkcore.design/page/7.0.19/10.0.0-preview.4.25258.110.json
info :   OK htt  Follow link (cmd + click)  egistration5-gz-semver2/microsoft.entityframeworkcore.design/page/7.0.19/10.0.0-preview.4.25258.110.json 314ms
info : Restorin                            epebul/Projects/Github/BAZY-DANYCH-2/entity-framework/entity-framework.csproj...
info :   CACHE https://api.nuget.org/v3/vulnerabilities/index.json
info :   CACHE https://api.nuget.org/v3-vulnerabilities/2025.05.28.05.39.59/vulnerability.base.json
info :   CACHE https://api.nuget.org/v3-vulnerabilities/2025.05.28.05.39.59/2025.06.01.11.40.17/vulnerability.update.json
info : Package 'Microsoft.EntityFrameworkCore.Design' is compatible with all the specified frameworks in project '/Users/depebul/Projects/Github/BAZY-DANYCH-2/entity-framework/entity-framew
ork.csproj'.
info : PackageReference for package 'Microsoft.EntityFrameworkCore.Design' version '9.0.5' added to file '/Users/depebul/Projects/Github/BAZY-DANYCH-2/entity-framework/entity-framework.cspr
oj'.
info : Generating MSBuild file /Users/depebul/Projects/Github/BAZY-DANYCH-2/entity-framework/obj/entity-framework.csproj.nuget.g.props.
info : Generating MSBuild file /Users/depebul/Projects/Github/BAZY-DANYCH-2/entity-framework/obj/entity-framework.csproj.nuget.g.targets.
info : Writing assets file to disk. Path: /Users/depebul/Projects/Github/BAZY-DANYCH-2/entity-framework/obj/project.assets.json
log  : Restored /Users/depebul/Projects/Github/BAZY-DANYCH-2/entity-framework/entity-framework.csproj (in 106 ms).

BAZY-DANYCH-2/entity-framework on master [✘?⇣] via .NET v9.0.300 🎯 net9.0 took 2s
➜
```

Po ponownej próbie wygenerowania migracji, otrzymujemy kolejny komunikat o błędzie:

```
BAZY-DANYCH-2/entity-framework on master [✘?⇣] via .NET v9.0.300 🎯 net9.0
❯ dotnet ef migrations add InitProductDatabase
Build started...
Build succeeded.
Unable to create a 'DbContext' of type 'ProdContext'. The exception 'No database provider has been configured for this DbContext. A provider can be configured by overriding the 'DbContext.O
nConfiguring' method or by using 'AddDbContext' on the application service provider. If 'AddDbContext' is used, then also ensure that your DbContext type accepts a DbContextOptions<TContext
> object in its constructor and passes it to the base constructor for DbContext.' was thrown while attempting to create an instance. For the different patterns supported at design time, see
 https://go.microsoft.com/fwlink/?linkid=851728

BAZY-DANYCH-2/entity-framework on master [✘?⇣] via .NET v9.0.300 🎯 net9.0
❯
```

Ponieważ nigdzie nie zdefiniowaliśmy bazy z której EF ma korzystać,

dodajemy do klasy `ProdContext` metodę `OnConfiguring`, która ustawia bazę danych na SQLite:

```
entity-framework > C# ProdContext.cs > ProdContext
1   using Microsoft.EntityFrameworkCore;
    0 references
2   public class ProdContext : DbContext
3   {
        0 references
4       public DbSet<Product> Products { get; set; }
5
        0 references
6       protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)
7       {
8           base.OnConfiguring(optionsBuilder);
9           optionsBuilder.UseSqlite("Datasource=MyProductDatabase");
10      }
11  }
```

Otrzymujemy kolejny błąd:

```
BAZY-DANYCH-2/entity-framework on  master [✘?↓] via .NET v9.0.300  net9.0
❯ dotnet ef migrations add InitProductDatabase
Build started...
Build failed. Use dotnet build to see the errors.

BAZY-DANYCH-2/entity-framework on  master [✘?↓] via .NET v9.0.300  net9.0
❯ dotnet build
Restore complete (0.1s)
  entity-framework failed with 1 error(s) (0.1s)
    /Users/depebul/Projects/Github/BAZY-DANYCH-2/entity-framework/ProdContext.cs(9,24): error CS1061: 'DbContextOptionsBuilder' does not contain a definition for 'UseSqlite' and no accessib
le extension method 'UseSqlite' accepting a first argument of type 'DbContextOptionsBuilder' could be found (are you missing a using directive or an assembly reference?)

Build failed with 1 error(s) in 0.4s

BAZY-DANYCH-2/entity-framework on  master [✘?↓] via .NET v9.0.300  net9.0
❯
```

Dodajemy więc do projektu pakiet `Microsoft.EntityFrameworkCore.Sqlite`:

```
BAZY-DANYCH-2/entity-framework on  master [✘?↓] via .NET v9.0.300  net9.0
❯ dotnet add package Microsoft.EntityFrameworkCore.Sqlite

Build succeeded in 0.2s
info : X.509 certificate chain validation will use the fallback certificate bundle at '/usr/local/share/dotnet/sdk/9.0.300/trustedroots/codesignctl.pem'.
info : X.509 certificate chain validation will use the fallback certificate bundle at '/usr/local/share/dotnet/sdk/9.0.300/trustedroots/timestampctl.pem'.
info : Adding PackageReference for package 'Microsoft.EntityFrameworkCore.Sqlite' into project '/Users/depebul/Projects/Github/BAZY-DANYCH-2/entity-framework/entity-framework.csproj'.
info :   GET https://api.nuget.org/v3/registration5-gz-semver2/microsoft.entityframeworkcore.sqlite/index.json
info :   OK https://api.nuget.org/v3/registration5-gz-semver2/microsoft.entityframeworkcore.sqlite/index.json 1186ms
info :   GET https://api.nuget.org/v3/registration5-gz-semver2/microsoft.entityframeworkcore.sqlite/page/0.0.1-alpha/3.1.2.json
info :   OK https://api.nuget.org/v3/registration5-gz-semver2/microsoft.entityframeworkcore.sqlite/page/0.0.1-alpha/3.1.2.json 242ms
info :   GET https://api.nuget.org/v3/registration5-gz-semver2/microsoft.entityframeworkcore.sqlite/page/3.1.3/6.0.0-preview.6.21352.1.json
info :   OK https://api.nuget.org/v3/registration5-gz-semver2/microsoft.entityframeworkcore.sqlite/page/3.1.3/6.0.0-preview.6.21352.1.json 199ms
info :   GET https://api.nuget.org/v3/registration5-gz-semver2/microsoft.entityframeworkcore.sqlite/page/6.0.0-preview.7.21378.4/7.0.17.json
info :   OK https://api.nuget.org/v3/registration5-gz-semver2/microsoft.entityframeworkcore.sqlite/page/6.0.0-preview.7.21378.4/7.0.17.json 470ms
info :   GET https://api.nuget.org/v3/registration5-gz-semver2/microsoft.entityframeworkcore.sqlite/page/7.0.18/10.0.0-preview.4.25258.110.json
info :   OK https://api.nuget.org/v3/registration5-gz-semver2/microsoft.entityframeworkcore.sqlite/page/7.0.18/10.0.0-preview.4.25258.110.json 204ms
info : Restoring packages for /Users/depebul/Projects/Github/BAZY-DANYCH-2/entity-framework/entity-framework.csproj...
info :   CACHE https://api.nuget.org/v3/vulnerabilities/index.json
info :   CACHE https://api.nuget.org/v3-vulnerabilities/2025.05.28.05.39.59/vulnerability.base.json
info :   CACHE https://api.nuget.org/v3-vulnerabilities/2025.05.28.05.39.59/2025.06.01.11.40.17/vulnerability.update.json
info : Package 'Microsoft.EntityFrameworkCore.Sqlite' is compatible with all the specified frameworks in project '/Users/depebul/Projects/Github/BAZY-DANYCH-2/entity-framework/entity-framew
ork.csproj'.
info : PackageReference for package 'Microsoft.EntityFrameworkCore.Sqlite' version '9.0.5' added to file '/Users/depebul/Projects/Github/BAZY-DANYCH-2/entity-framework/entity-framework.cspr
oj'.
info : Generating MSBuild file /Users/depebul/Projects/Github/BAZY-DANYCH-2/entity-framework/obj/entity-framework.csproj.nuget.g.targets.
info : Writing assets file to disk. Path: /Users/depebul/Projects/Github/BAZY-DANYCH-2/entity-framework/obj/project.assets.json
log  : Restored /Users/depebul/Projects/Github/BAZY-DANYCH-2/entity-framework/entity-framework.csproj (in 90 ms).

BAZY-DANYCH-2/entity-framework on  master [✘?↓] via .NET v9.0.300  net9.0 took 3s
➜
```

Tym razem próba wygenerowania migracji kończy się sukcesem:

```
BAZY-DANYCH-2/entity-framework on  master [✘?⇣] via .NET v9.0.300 🎯 net9.0 took 3s
➜ dotnet ef migrations add InitProductDatabase
Build started...
Build succeeded.
Done. To undo this action, use 'ef migrations remove'

BAZY-DANYCH-2/entity-framework on  master [✘?⇣] via .NET v9.0.300 🎯 net9.0 took 2s
➜ ls -la Migrations
.rw-r--r--@ 1.1k depebul  1 Jun 19:39 C# 20250601173958_InitProductDatabase.cs
.rw-r--r--@ 1.2k depebul  1 Jun 19:39 C# 20250601173958_InitProductDatabase.Designer.cs
.rw-r--r--@ 1.1k depebul  1 Jun 19:39 C# ProdContextModelSnapshot.cs

BAZY-DANYCH-2/entity-framework on  master [✘?⇣] via .NET v9.0.300 🎯 net9.0
➜
```

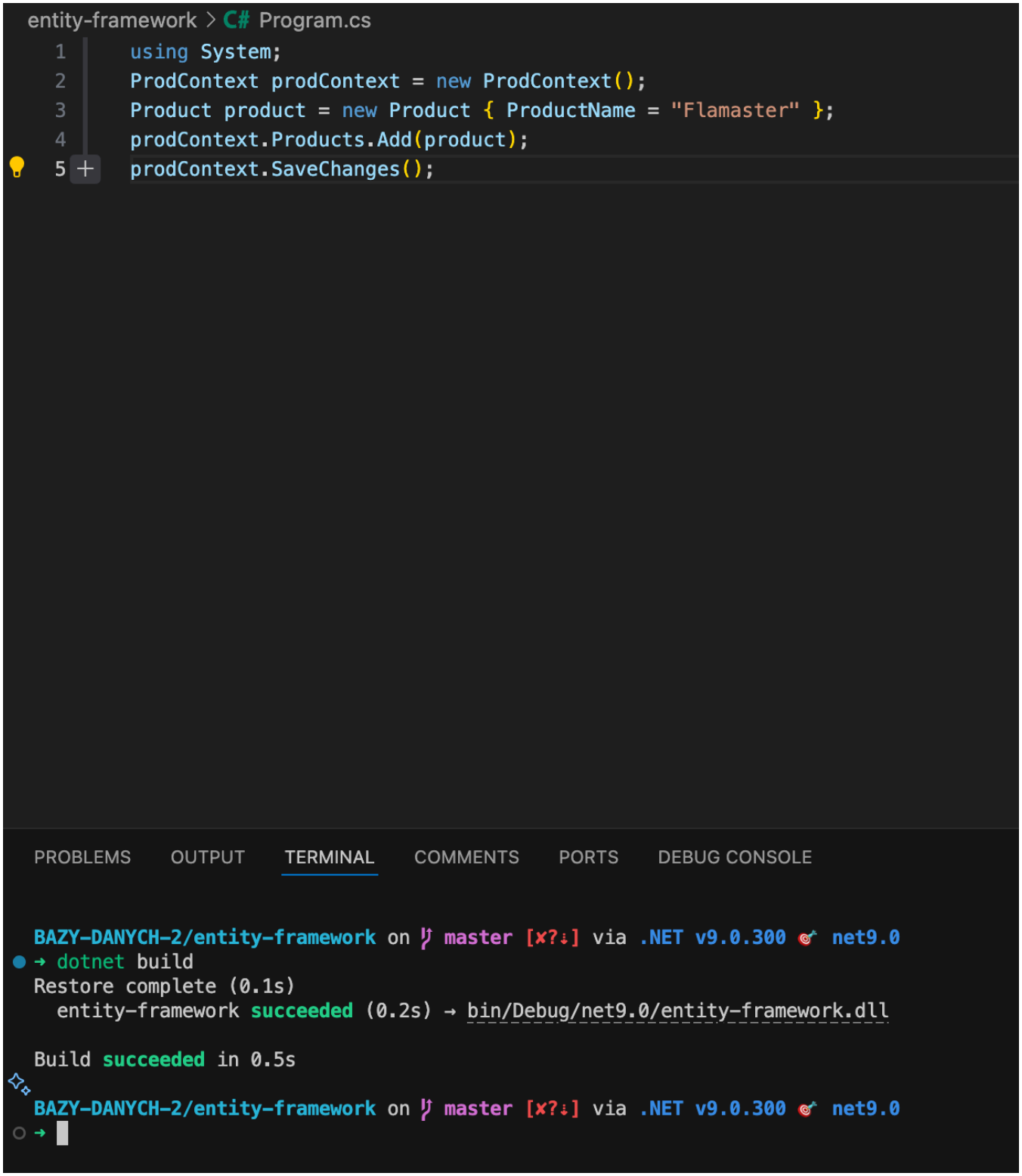
Aktualizujemy bazę danych poleceniem

```
dotnet ef database update
```

W ten sposób otrzymujemy plik bazy danych `MyProductDatabase`

```
BAZY-DANYCH-2/entity-framework on  master [✘?⇣] via .NET v9.0.300 🎯 net9.0
➜ dotnet ef database update
Build started...
Build succeeded.
Acquiring an exclusive lock for migration application. See https://aka.ms/efcore-docs-migrations-lock for more information if this takes too long.
Applying migration '20250601173958_InitProductDatabase'.
Done.

BAZY-DANYCH-2/entity-framework on  master [✘?⇣] via .NET v9.0.300 🎯 net9.0
➜ ls -la
drwxr-xr-x@   - depebul  1 Jun 19:15  bin
.rw-r--r--@ 746 depebul  1 Jun 19:39 C# entity-framework.csproj
drwxr-xr-x@   - depebul  1 Jun 19:39  Migrations
.rw-r--r--@ 25k depebul  1 Jun 19:41  MyProductDatabase
drwxr-xr-x@   - depebul  1 Jun 19:39  obj
.rw-r--r--@ 331 depebul  1 Jun 19:37 C# ProdContext.cs
.rw-r--r--@ 150 depebul  1 Jun 19:17 C# Product.cs
.rw-r--r--@ 103 depebul  1 Jun 19:09 C# Program.cs

BAZY-DANYCH-2/entity-framework on  master [✘?⇣] via .NET v9.0.300 🎯 net9.0
➜
```

W pliku `Program.cs` dodajemy kod odpowiedzialny za dodawanie produktu do bazy danych:

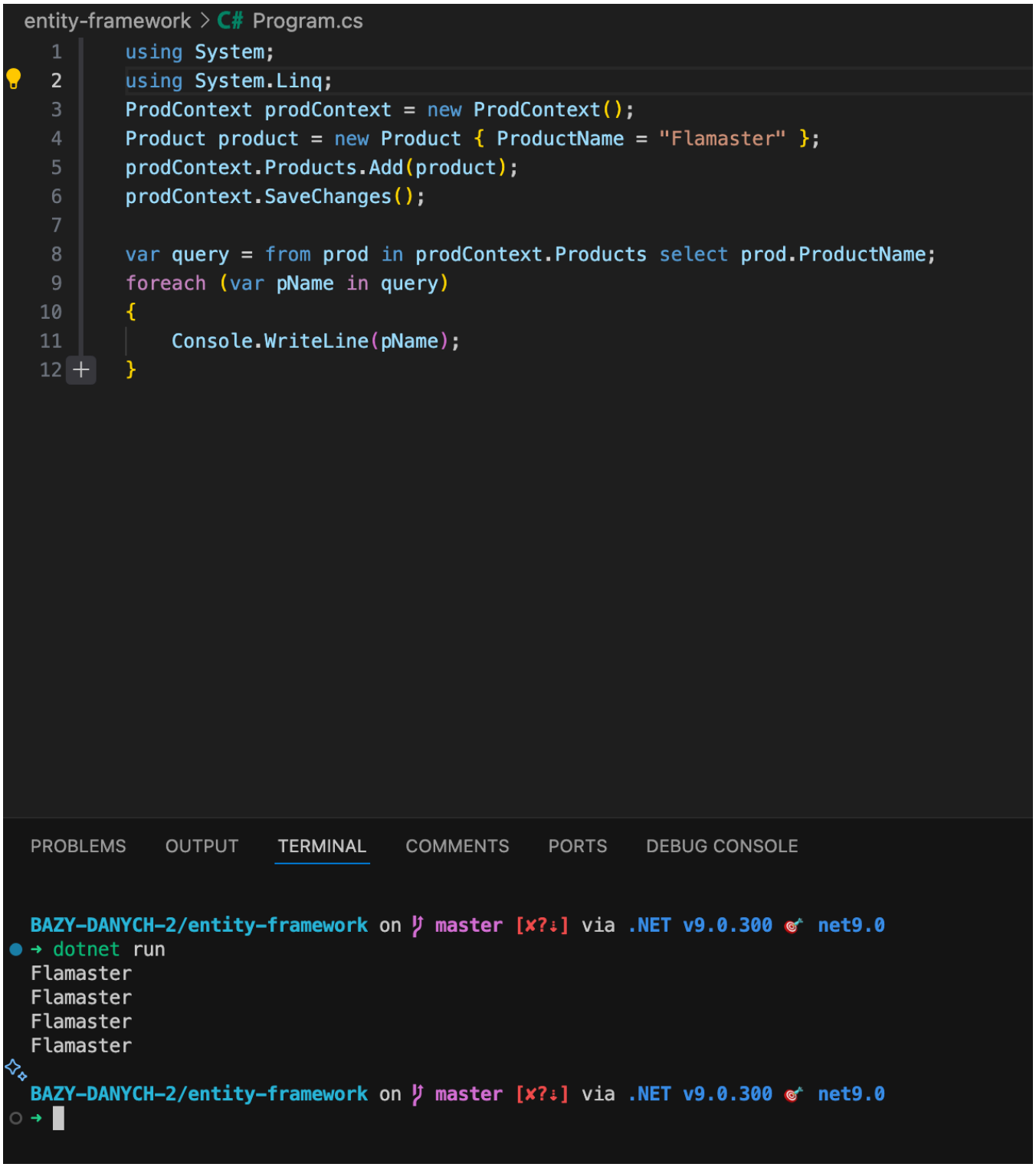


Poleceniem `sqlite3` otwieramy bazę danych i sprawdzamy czy produkt został dodany:

```
BAZY-DANYCH-2/entity-framework on  master [✘?⇣] via .NET v9.0.300 🎯 net9.0
➜ sqlite3 MyProductDatabase
SQLite version 3.45.3 2024-04-15 13:34:05
Enter ".help" for usage hints.
sqlite> .tables
Products                __EFMigrationsHistory  __EFMigrationsLock
sqlite> .schema Products
CREATE TABLE IF NOT EXISTS "Products" (
    "ProductId" INTEGER NOT NULL CONSTRAINT "PK_Products" PRIMARY KEY AUTOINCREMENT,
    "ProductName" TEXT NULL,
    "UnitsInStock" INTEGER NOT NULL
);
sqlite>
```

```
BAZY-DANYCH-2/entity-framework on  master [✘?⇣] via .NET v9.0.300 🎯 net9.0
➜ dotnet run

BAZY-DANYCH-2/entity-framework on  master [✘?⇣] via .NET v9.0.300 🎯 net9.0
➜ sqlite3 MyProductDatabase
SQLite version 3.45.3 2024-04-15 13:34:05
Enter ".help" for usage hints.
sqlite> select * from Products
   ...> ;
1|Flamaster|0
sqlite> .exit

BAZY-DANYCH-2/entity-framework on  master [✘?⇣] via .NET v9.0.300 🎯 net9.0 took 19s
➜ dotnet run

BAZY-DANYCH-2/entity-framework on  master [✘?⇣] via .NET v9.0.300 🎯 net9.0
➜ sqlite3 MyProductDatabase
SQLite version 3.45.3 2024-04-15 13:34:05
Enter ".help" for usage hints.
sqlite> select * from Products;
1|Flamaster|0
2|Flamaster|0
sqlite>
```

Do programu dopisujemy kod odpowiedzialny za pobieranie wszystkich produktów z bazy danych oraz ich wyświetlanie:



Teraz dodamy kod odpowiedzialny za dodanie nowego produktu, podczas uruchomienia programu:

entity-framework > C# Program.cs

```
1   using System;
2   using System.Linq;
3   Console.WriteLine("Podaj nazwę produktu: ");
4   String? prodName = Console.ReadLine();
5   ProdContext prodContext = new ProdContext();
6   Product product = new Product { ProductName = prodName };
7   prodContext.Products.Add(product);
8   prodContext.SaveChanges();
9
10  var query = from prod in prodContext.Products select prod.ProductName;
11  foreach (var pName in query)
12  {
13      Console.WriteLine(pName);
14  }
```

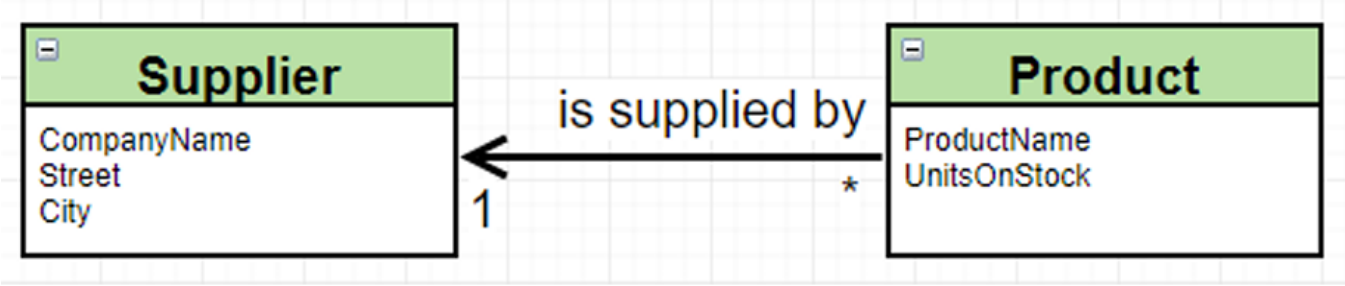
PROBLEMS OUTPUT TERMINAL COMMENTS PORTS DEBUG CONSOLE

```
BAZY-DANYCH-2/entity-framework on  master [✘?⇣] via .NET v9.0.300 🎯 net9.0
➜ dotnet run
Podaj nazwę produktu:
Kredki
Flamaster
Flamaster
Flamaster
Flamaster
Krzeslo
Kredki

BAZY-DANYCH-2/entity-framework on  master [✘?⇣] via .NET v9.0.300 🎯 net9.0 took 2s
➜
```

Część II

Zadanie A.

Zmodyfikuj model wprowadzając pojęcie Dostawcy jak poniżej



- Stwórz nowego dostawcę.
- Znajdź poprzednio wprowadzony produkt i ustaw jego dostawcę na właśnie dodanego.
- Udokumentuj wykonane kroki oraz uzyskany rezultat (.schema table/diagram z datagrip, select * from….)

Plik `Supplier.cs` zawiera klasę `Supplier`, która będzie reprezentować dostawcę produktów:

```
// Dodano klasę Supplier
public class Supplier
{
    public int SupplierId { get; set; }
    public string? CompanyName { get; set; }
    public string? Street { get; set; }
    public string? City { get; set; }
}
```

Plik `Product.cs`:

```
public class Product
{
    public int ProductId { get; set; }
    public string? ProductName { get; set; }
    public int UnitsInStock { get; set; }
    // Dodano relacje z Supplier
    public Supplier? Supplier { get; set; }
}
```

Plik `ProdContext.cs`:

```
using Microsoft.EntityFrameworkCore;
public class ProdContext : DbContext
{
    public DbSet<Product> Products { get; set; }
    // Dodano DbSet dla Supplier
    public DbSet<Supplier> Suppliers { get; set; }

    protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)
    {
        base.OnConfiguring(optionsBuilder);
        optionsBuilder.UseSqlite("Datasource=MyProductDatabase");
    }
}
```

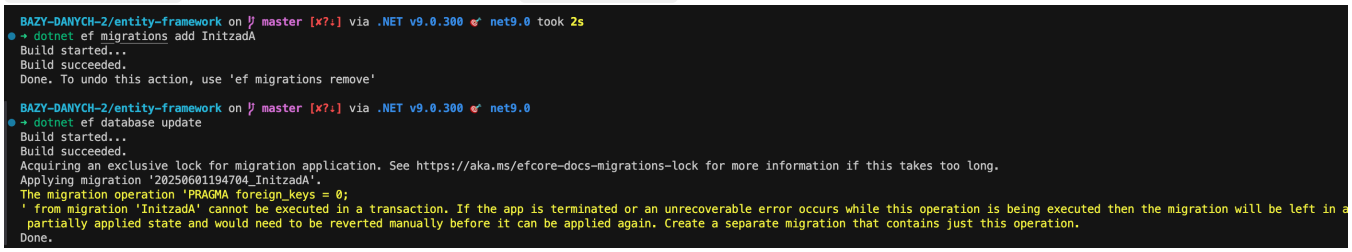
Plik `Program.cs`:

```
using System;
using System.Linq;
var prodContext = new ProdContext();
var supplier = new Supplier
{
    CompanyName = "firma",
    Street = "fajna",
    City = "Krakow"
};
prodContext.Suppliers.Add(supplier);
prodContext.SaveChanges();
var lastProduct = prodContext.Products
    .OrderByDescending(p => p.ProductId)
    .FirstOrDefault();

if (lastProduct != null)
{
    lastProduct.Supplier = supplier;
    prodContext.SaveChanges();
}
```

Prezentacja działania zastosowanych zmian:

Migracja oraz aktualizacja bazy danych:

Migracja bazy danych została zaktualizowana, aby uwzględnić nową klasę `Supplier` oraz relację z klasą `Product`.



Schematy powstałych tabel:

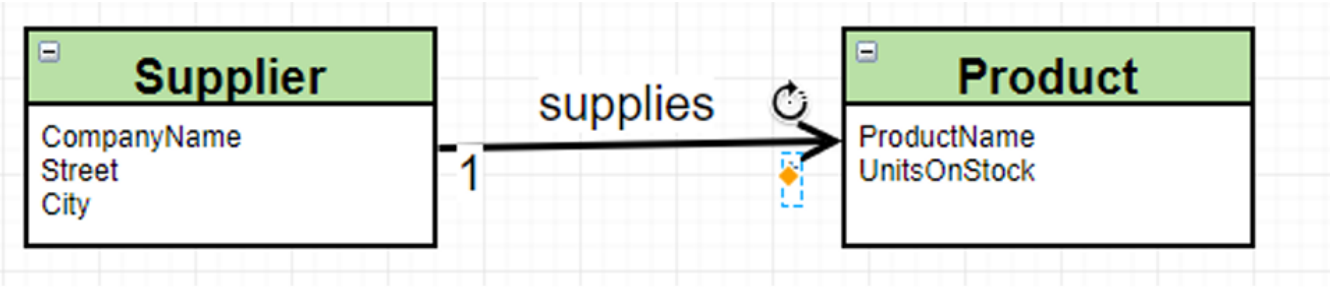
```
BAZY-DANYCH-2/entity-framework on master [✘?⇣] via .NET v9.0.300 net9.0 took 10s
➜ sqlite3 MyProductDatabase
SQLite version 3.45.3 2024-04-15 13:34:05
Enter ".help" for usage hints.
sqlite> .schema Suppliers
CREATE TABLE IF NOT EXISTS "Suppliers" (
    "SupplierId" INTEGER NOT NULL CONSTRAINT "PK_Suppliers" PRIMARY KEY AUTOINCREMENT,
    "CompanyName" TEXT NULL,
    "Street" TEXT NULL,
    "City" TEXT NULL
);
sqlite> .schema Products
CREATE TABLE IF NOT EXISTS "Products" (
    "ProductId" INTEGER NOT NULL CONSTRAINT "PK_Products" PRIMARY KEY AUTOINCREMENT,
    "ProductName" TEXT NULL,
    "SupplierId" INTEGER NULL,
    "UnitsInStock" INTEGER NOT NULL,
    CONSTRAINT "FK_Products_Suppliers_SupplierId" FOREIGN KEY ("SupplierId") REFERENCES "Suppliers" ("SupplierId")
);
CREATE INDEX "IX_Products_SupplierId" ON "Products" ("SupplierId");
sqlite>
```

Wyniki po dodaniu dostawcy do ostatnio dodanego produktu:

```
BAZY-DANYCH-2/entity-framework on master [✘?⇣] via .NET v9.0.300 net9.0
❯ dotnet run

BAZY-DANYCH-2/entity-framework on master [✘?⇣] via .NET v9.0.300 net9.0
➜ sqlite3 MyProductDatabase
SQLite version 3.45.3 2024-04-15 13:34:05
Enter ".help" for usage hints.
sqlite> select * from Products
   ...> ;
1|Flamaster||0
2|Flamaster||0
3|Flamaster||0
4|Krzeslo||0
5|Kredki|1|0
sqlite>
```

Zadanie B.

Odwróć relację zgodnie z poniższym schematem



- Stwórz kilka produktów.

- Dodaj je do produktów dostarczanych przez nowo stworzonego dostawcę.
- Udokumentuj wykonane kroki oraz uzyskany rezultat (.schema table/diagram z datagrip, select * from….)

Plik `Supplier.cs`:

```
public class Supplier
{
    public int SupplierId { get; set; }
    public string? CompanyName { get; set; }
    public string? Street { get; set; }
    public string? City { get; set; }
    // Dodano relację z Product
    public ICollection<Product> Supplies { get; set; } = new List<Product>();
}
```

Plik `Product.cs`:

```
public class Product
{
    public int ProductId { get; set; }
    public string? ProductName { get; set; }
    public int UnitsInStock { get; set; }
    // Usunięto relacje z Supplier
}
```

Plik `ProdContext.cs`:

```
using Microsoft.EntityFrameworkCore;
public class ProdContext : DbContext
{
    public DbSet<Product> Products { get; set; }
    // Dodano DbSet dla Supplier
    public DbSet<Supplier> Suppliers { get; set; }

    protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)
    {
        base.OnConfiguring(optionsBuilder);
        optionsBuilder.UseSqlite("Datasource=MyProductDatabase");
    }
}
```

Plik `Program.cs`:

```
using System;
using System.Linq;
var prodContext = new ProdContext();



var supplier = new Supplier
{
    CompanyName = "firma2",
    Street = "fajna2",
    City = "Krakow2"
};
prodContext.Suppliers.Add(supplier);

var product1 = new Product
{
    ProductName = "Produkt1",
    UnitsInStock = 10
};
var product2 = new Product
{
    ProductName = "Produkt2",
    UnitsInStock = 20
};
var product3 = new Product
{
    ProductName = "Produkt3",
    UnitsInStock = 30
};

prodContext.Products.AddRange(product1, product2, product3);

supplier.Supplies.Add(product1);
supplier.Supplies.Add(product2);
supplier.Supplies.Add(product3);

prodContext.SaveChanges();
```

Wyniki migracji i działania programu:

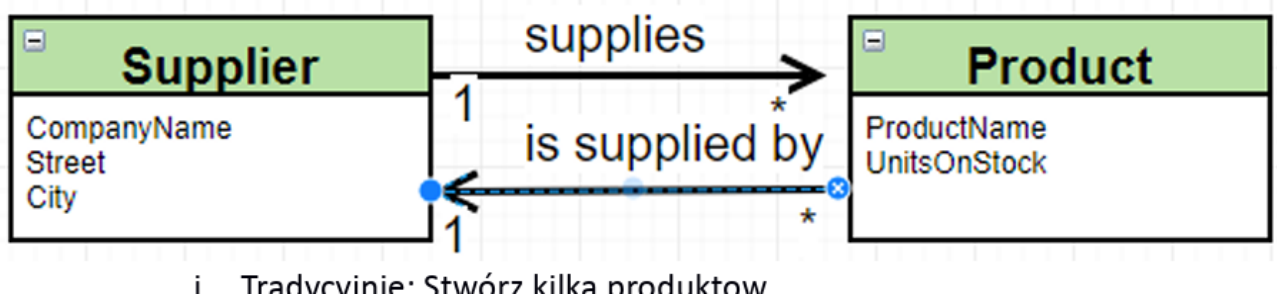
```
BAZY-DANYCH-2/entity-framework on  master [✗?↓] via .NET v9.0.300  net9.0
➜ dotnet ef migrations add InitzadB
Build started...
Build succeeded.
Done. To undo this action, use 'ef migrations remove'

BAZY-DANYCH-2/entity-framework on  master [✗?↓] via .NET v9.0.300  net9.0
➜ dotnet ef database update
Build started...
Build succeeded.
Acquiring an exclusive lock for migration application. See https://aka.ms/efcore-docs-migrations-lock for more information if this takes too long.
Applying migration '20250601200157_InitzadB'.
Done.

BAZY-DANYCH-2/entity-framework on  master [✗?↓] via .NET v9.0.300  net9.0
➜ dotnet run

BAZY-DANYCH-2/entity-framework on  master [✗?↓] via .NET v9.0.300  net9.0
➜ sqlite3 MyProductDatabase
SQLite version 3.45.3 2024-04-15 13:34:05
Enter ".help" for usage hints.
sqlite> select * from Products
   ...> ;
1|Flamaster||0
2|Flamaster||0
3|Flamaster||0
4|Krzeslo||0
5|Kredki|1|0
6|Produkt1|2|10
7|Produkt2|2|20
8|Produkt3|2|30
sqlite> select * from Suppliers
   ...> ;
1|firma|fajna|Krakow
2|firma2|fajna2|Krakow2
sqlite>
```

Schematy powstałych tabel:

```
sqlite> .schema Products
CREATE TABLE IF NOT EXISTS "Products" (
    "ProductId" INTEGER NOT NULL CONSTRAINT "PK_Products" PRIMARY KEY AUTOINCREMENT,
    "ProductName" TEXT NULL,
    "SupplierId" INTEGER NULL,
    "UnitsInStock" INTEGER NOT NULL,
    CONSTRAINT "FK_Products_Suppliers_SupplierId" FOREIGN KEY ("SupplierId") REFERENCES "Suppliers" ("SupplierId")
);
CREATE INDEX "IX_Products_SupplierId" ON "Products" ("SupplierId");
sqlite> .schema Suppliers
CREATE TABLE IF NOT EXISTS "Suppliers" (
    "SupplierId" INTEGER NOT NULL CONSTRAINT "PK_Suppliers" PRIMARY KEY AUTOINCREMENT,
    "CompanyName" TEXT NULL,
    "Street" TEXT NULL,
    "City" TEXT NULL
);
sqlite>
```

Zadanie C.

Zamodeluj relację dwustronną jak poniżej:



i Tradycyjnie: Stwórz kilka produktow

- Stwórz kilka produktów.

- Dodaj je do produktów dostarczanych przez nowo stworzonego dostawcę.
- Udokumentuj wykonane kroki oraz uzyskany rezultat (.schema table/diagram z datagrip, select * from….)

Plik `Supplier.cs`:

```
// Bez zmian
public class Supplier
{
    public int SupplierId { get; set; }
    public string? CompanyName { get; set; }
    public string? Street { get; set; }
    public string? City { get; set; }

    public ICollection<Product> Supplies { get; set; } = new List<Product>();
}
```

Plik `Product.cs`:

```
public class Product
{
    public int ProductId { get; set; }
    public string? ProductName { get; set; }
    public int UnitsInStock { get; set; }
    // Ponownie dodano relację z Supplier
    public Supplier? Supplier { get; set; }
}
```

Plik `ProdContext.cs`:

```
using Microsoft.EntityFrameworkCore;
public class ProdContext : DbContext
{
    public DbSet<Product> Products { get; set; }

    public DbSet<Supplier> Suppliers { get; set; }

    protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)
    {
        base.OnConfiguring(optionsBuilder);
        optionsBuilder.UseSqlite("Datasource=MyProductDatabase");
    }
}
```

Plik `Program.cs`:

```
using System;
using System.Linq;
var prodContext = new ProdContext();

var supplier = new Supplier
{
    CompanyName = "firma2",
    Street = "fajna2",
    City = "Krakow2"
};
prodContext.Suppliers.Add(supplier);

var product1 = new Product
{
    ProductName = "Produkt1",
    UnitsInStock = 10,
    Supplier = supplier // Ustawienie relacji z dostawcą
};
var product2 = new Product
{
    ProductName = "Produkt2",
    UnitsInStock = 20
    Supplier = supplier // Ustawienie relacji z dostawcą
};
var product3 = new Product
{
    ProductName = "Produkt3",
    UnitsInStock = 30
    Supplier = supplier // Ustawienie relacji z dostawcą
};

prodContext.Products.AddRange(product1, product2, product3);

supplier.Supplies.Add(product1);
supplier.Supplies.Add(product2);
supplier.Supplies.Add(product3);

prodContext.SaveChanges();
```

Polecenie migracji i aktualizacji bazy danych:

```
BAZY-DANYCH-2/entity-framework on  master [x?↓] via .NET v9.0.300  net9.0
❯ dotnet ef migrations add InitzadC
Build started...
Build succeeded.
Done. To undo this action, use 'ef migrations remove'

BAZY-DANYCH-2/entity-framework on  master [x?↓] via .NET v9.0.300  net9.0
➜ dotnet ef database update
Build started...
Build succeeded.
Acquiring an exclusive lock for migration application. See https://aka.ms/efcore-docs-migrations-lock for more information if this takes too long.
Applying migration '20250601200959_InitzadC'.
Done.

BAZY-DANYCH-2/entity-framework on  master [x?↓] via .NET v9.0.300  net9.0
➜ dotnet run
```

Wyniki działania programu:

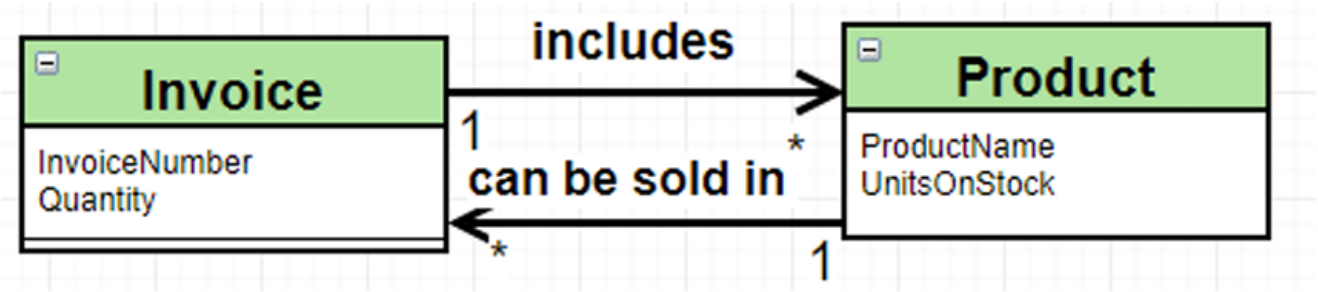
```
BAZY-DANYCH-2/entity-framework on  master [x?↓] via .NET v9.0.300  net9.0
➜ sqlite3 MyProductDatabase
SQLite version 3.45.3 2024-04-15 13:34:05
Enter ".help" for usage hints.
sqlite> select * from Products
   ...> ;
1|Flamaster||0
2|Flamaster||0
3|Flamaster||0
4|Krzeslo||0
5|Kredki|1|0
6|Produkt1|2|10
7|Produkt2|2|20
8|Produkt3|2|30
9|Produkt1|3|10
10|Produkt2|3|20
11|Produkt3|3|30
sqlite> select * from Suppliers
   ...> ;
1|firma|fajna|Krakow
2|firma2|fajna2|Krakow2
3|firma2|fajna2|Krakow2
```

Schematy powstałych tabel:

```
sqlite> .schema Products
CREATE TABLE IF NOT EXISTS "Products" (
    "ProductId" INTEGER NOT NULL CONSTRAINT "PK_Products" PRIMARY KEY AUTOINCREMENT,
    "ProductName" TEXT NULL,
    "SupplierId" INTEGER NULL,
    "UnitsInStock" INTEGER NOT NULL,
    CONSTRAINT "FK_Products_Suppliers_SupplierId" FOREIGN KEY ("SupplierId") REFERENCES "Suppliers" ("SupplierId")
);
CREATE INDEX "IX_Products_SupplierId" ON "Products" ("SupplierId");
sqlite> .schema Suppliers
CREATE TABLE IF NOT EXISTS "Suppliers" (
    "SupplierId" INTEGER NOT NULL CONSTRAINT "PK_Suppliers" PRIMARY KEY AUTOINCREMENT,
    "CompanyName" TEXT NULL,
    "Street" TEXT NULL,
    "City" TEXT NULL
);
sqlite>
```

Zadanie D.

Zamodeluj relację wiele-do-wielu, jak poniżej:



- Stwórzy kilka produktów i "sprzedaj" je na kilku transakcjach.
- Pokaż produkty sprzedane w ramach wybranej faktury/transakcji
- Pokaż faktury, w ramach których sprzedany został wybrany produkt
- Udokumentuj wykonane kroki oraz uzyskany rezultat (.schema table/diagram z datagrip, select * from….)

Plik `Product.cs`:

```
public class Product
{
    public int ProductId { get; set; }
    public string? ProductName { get; set; }
    public int UnitsInStock { get; set; }
    public Supplier? Supplier { get; set; }
    // Dodano relację z Invoice
    public ICollection<Invoice> Invoices { get; set; } = new List<Invoice>();
}
```

Plik `Invoice.cs`:

```
// Dodano klasę Invoice
public class Invoice
{
    public int InvoiceId { get; set; }
    public int InvoiceNumber { get; set; }
    public int Quantity { get; set; }
    public ICollection<Product> Products { get; set; } = new List<Product>();
}
```

Plik `ProdContext.cs`:

```
using Microsoft.EntityFrameworkCore;
public class ProdContext : DbContext
{
    public DbSet<Product> Products { get; set; }

    public DbSet<Supplier> Suppliers { get; set; }
    // Dodano DbSet Invoice
    public DbSet<Invoice> Invoices { get; set; }

    protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)
    {
        base.OnConfiguring(optionsBuilder);
        optionsBuilder.UseSqlite("Datasource=MyProductDatabase");
    }
}
```

Plik `Program.cs`:

```
using System;
using System.Linq;
var prodContext = new ProdContext();

var product1 = new Product
{
    ProductName = "Produktor",
    UnitsInStock = 10,

};
var product2 = new Product
{
    ProductName = "Pizza",
    UnitsInStock = 20,

};
var product3 = new Product
{
    ProductName = "Ogorek",
    UnitsInStock = 30,

};
var product4 = new Product
{
    ProductName = "Smietana",
    UnitsInStock = 30,

};

var invoice1 = new Invoice
{
    InvoiceNumber = 1001,
    Quantity = 4,
    Products = new List<Product> { product1, product2, product3, product4 }
};

product1.Invoices.Add(invoice1);
product2.Invoices.Add(invoice1);
product3.Invoices.Add(invoice1);
product4.Invoices.Add(invoice1);

var invoice2 = new Invoice
{
    InvoiceNumber = 1002,
    Quantity = 2,
    Products = new List<Product> { product1, product2 }
};
```

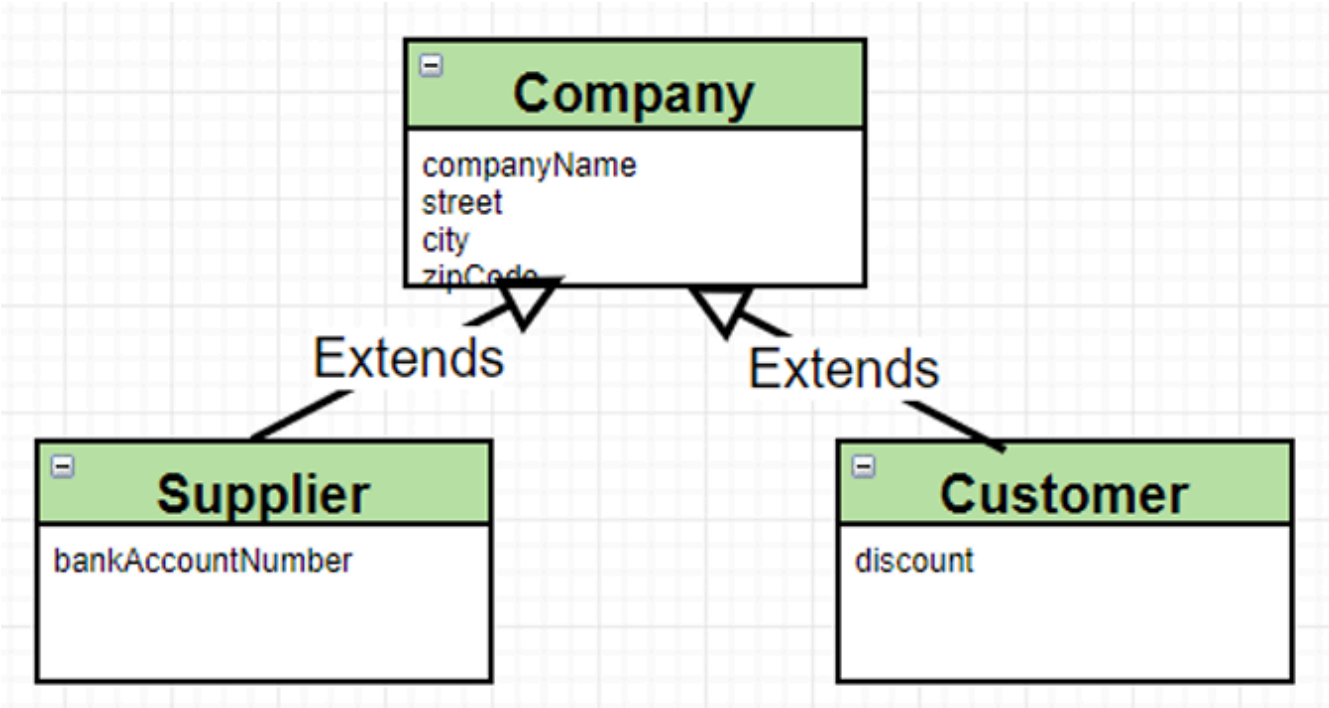
```
product1.Invoices.Add(invoice2);
product2.Invoices.Add(invoice2);

var invoice3 = new Invoice
{
    InvoiceNumber = 1003,
    Quantity = 2,
    Products = new List<Product> { product3, product4 }
};
product3.Invoices.Add(invoice3);
product4.Invoices.Add(invoice3);

prodContext.Products.AddRange(product1, product2, product3, product4);
prodContext.Invoices.AddRange(invoice1, invoice2, invoice3);

prodContext.SaveChanges();

var query1 = from invoice in prodContext.Invoices
             where invoice.InvoiceNumber == 1001
             select invoice.Products;

foreach (var products in query1)
{
    foreach (var product in products)
    {
        Console.WriteLine($"Invoice 1001 contains product: {product?.ProductName}");
    }
}

var query2 = from product in prodContext.Products
             where product.ProductName == "Produktor"
             select product.Invoices;
foreach (var invoices in query2)
{
    foreach (var invoice in invoices)
    {
        Console.WriteLine($"Product 'Produktor' is sold in invoice number: {invoice?.InvoiceNumber}");
    }

}
```

```
BAZY-DANYCH-2/entity-framework on  master [x?↓] via .NET v9.0.300 🎯 net9.0 took 4m13s
➜ dotnet run
Invoice 1001 contains product: Produktor
Invoice 1001 contains product: Pizza
Invoice 1001 contains product: Ogorek
Invoice 1001 contains product: Smietana
Product 'Produktor' is sold in invoice number: 1001

BAZY-DANYCH-2/entity-framework on  master [x?↓] via .NET v9.0.300 🎯 net9.0
➜ dotnet run
Invoice 1001 contains product: Produktor
Invoice 1001 contains product: Pizza
Invoice 1001 contains product: Ogorek
Invoice 1001 contains product: Smietana
Product 'Produktor' is sold in invoice number: 1001
Product 'Produktor' is sold in invoice number: 1002

BAZY-DANYCH-2/entity-framework on  master [x?↓] via .NET v9.0.300 🎯 net9.0
➜
```

```
BAZY-DANYCH-2/entity-framework on  master [x?↓] via .NET v9.0.300 🎯 net9.0
➜ sqlite3 MyProductDatabase
SQLite version 3.45.3 2024-04-15 13:34:05
Enter ".help" for usage hints.
sqlite> .schema Products
CREATE TABLE IF NOT EXISTS "Products" (
    "ProductId" INTEGER NOT NULL CONSTRAINT "PK_Products" PRIMARY KEY AUTOINCREMENT,
    "ProductName" TEXT NULL,
    "SupplierId" INTEGER NULL,
    "UnitsInStock" INTEGER NOT NULL,
    CONSTRAINT "FK_Products_Suppliers_SupplierId" FOREIGN KEY ("SupplierId") REFERENCES "Suppliers" ("SupplierId")
);
CREATE INDEX "IX_Products_SupplierId" ON "Products" ("SupplierId");
sqlite> .schema Invoices
CREATE TABLE IF NOT EXISTS "Invoices" (
    "InvoiceId" INTEGER NOT NULL CONSTRAINT "PK_Invoices" PRIMARY KEY AUTOINCREMENT,
    "InvoiceNumber" INTEGER NOT NULL,
    "Quantity" INTEGER NOT NULL
);
sqlite>
```

Zadanie E.

Wprowadźdomodeluponiższą hierarchie dziedziczenia używając startegii `Table-Per-Hierarchy`:



- Dodaj i pobierz z bazy kilka firm obu rodzajów.
- Udokumentuj wykonane kroki oraz uzyskany rezultat (.schema table/diagram z datagrip, select * from….)

Plik `Company.cs`:

```
// Dodano klase Company
public class Company
{
    public int CompanyId { get; set; }
    public string? CompanyName { get; set; }
    public string? Street { get; set; }
    public string? City { get; set; }
    public string? ZipCode { get; set; }
}
```

Plik `Customer.cs`:

```
// Dodano klase Customer
public class Customer : Company
{
    public double Discount { get; set; }
}
```

Plik `Supplier.cs`:

```
// Zmieniono klase Supplier
public class Supplier : Company
{
    public string? BankAccountNumber { get; set; }
}
```

Plik `ProdContext.cs`:

```
using Microsoft.EntityFrameworkCore;
public class ProdContext : DbContext
{
    public DbSet<Product> Products { get; set; }
    // Dodano DbSet Companies i Customers
    public DbSet<Company> Companies { get; set; }
    public DbSet<Supplier> Suppliers { get; set; }
    public DbSet<Customer> Customers { get; set; }
    public DbSet<Invoice> Invoices { get; set; }

    protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)
    {
        base.OnConfiguring(optionsBuilder);
        optionsBuilder.UseSqlite("Datasource=MyProductDatabase");
    }
}
```

Plik `Program.cs`:

```
using System;
using System.Linq;
var prodContext = new ProdContext();


var supplier1 = new Supplier { CompanyName = "Cuowiek", Street = "Jane", City = "Londym", ZipCode = "12345",
var supplier2 = new Supplier { CompanyName = "Produktor", Street = "Kowalska", City = "Warszawa", ZipCode =

var customer2 = new Customer { CompanyName = "Kowal", Street = "Nowa", City = "Kraków", ZipCode = "67890", D
var customer1 = new Customer { CompanyName = "Nowak", Street = "Stara", City = "Wrocław", ZipCode = "09876",

prodContext.Suppliers.AddRange(supplier1, supplier2);
prodContext.Customers.AddRange(customer1, customer2);

prodContext.SaveChanges();
// Query suppliers from Londym
var londynSuppliers = prodContext.Suppliers
    .Where(s => s.City == "Londym")
    .ToList();
Console.WriteLine("Suppliers from Londym:");
foreach (var supplier in londynSuppliers)
{
    Console.WriteLine($"- {supplier.CompanyName}, {supplier.Street}");
}

// Query suppliers by company name containing specific text
var suppliersWithCo = prodContext.Suppliers
    .Where(s => s.CompanyName.Contains("o"))
    .ToList();
Console.WriteLine("\nSuppliers with 'o' in name:");
foreach (var supplier in suppliersWithCo)
{
    Console.WriteLine($"- {supplier.CompanyName}");
}

// Query customers by zip code
var customersWithZip = prodContext.Customers
    .Where(c => c.ZipCode.StartsWith("0"))
    .ToList();
Console.WriteLine("\nCustomers with zip starting with '0':");
foreach (var customer in customersWithZip)
{
    Console.WriteLine($"- {customer.CompanyName}, Zip: {customer.ZipCode}");
}
```

```
BAZY-DANYCH-2/entity-framework on  master [x?↓] via .NET v9.0.300 🎯 net9.0
➜ sqlite3 MyProductDatabase
SQLite version 3.45.3 2024-04-15 13:34:05
Enter ".help" for usage hints.
sqlite> .schema
CREATE TABLE IF NOT EXISTS "__EFMigrationsLock" (
    "Id" INTEGER NOT NULL CONSTRAINT "PK___EFMigrationsLock" PRIMARY KEY,
    "Timestamp" TEXT NOT NULL
);
CREATE TABLE IF NOT EXISTS "__EFMigrationsHistory" (
    "MigrationId" TEXT NOT NULL CONSTRAINT "PK___EFMigrationsHistory" PRIMARY KEY,
    "ProductVersion" TEXT NOT NULL
);
CREATE TABLE sqlite_sequence(name,seq);
CREATE TABLE IF NOT EXISTS "Invoices" (
    "InvoiceId" INTEGER NOT NULL CONSTRAINT "PK_Invoices" PRIMARY KEY AUTOINCREMENT,
    "InvoiceNumber" INTEGER NOT NULL,
    "Quantity" INTEGER NOT NULL
);
CREATE TABLE IF NOT EXISTS "InvoiceProduct" (
    "InvoicesInvoiceId" INTEGER NOT NULL,
    "ProductsProductId" INTEGER NOT NULL,
    CONSTRAINT "PK_InvoiceProduct" PRIMARY KEY ("InvoicesInvoiceId", "ProductsProductId"),
    CONSTRAINT "FK_InvoiceProduct_Invoices_InvoicesInvoiceId" FOREIGN KEY ("InvoicesInvoiceId") REFERENCES "Invoices" ("InvoiceId") ON DELETE CASCADE,
    CONSTRAINT "FK_InvoiceProduct_Products_ProductsProductId" FOREIGN KEY ("ProductsProductId") REFERENCES "Products" ("ProductId") ON DELETE CASCADE
);
CREATE INDEX "IX_InvoiceProduct_ProductsProductId" ON "InvoiceProduct" ("ProductsProductId");
CREATE TABLE IF NOT EXISTS "Products" (
    "ProductId" INTEGER NOT NULL CONSTRAINT "PK_Products" PRIMARY KEY AUTOINCREMENT,
    "ProductName" TEXT NULL,
    "SupplierCompanyId" INTEGER NULL,
    "UnitsInStock" INTEGER NOT NULL,
    CONSTRAINT "FK_Products_Companies_SupplierCompanyId" FOREIGN KEY ("SupplierCompanyId") REFERENCES "Companies" ("CompanyId")
);
CREATE TABLE IF NOT EXISTS "Companies" (
    "CompanyId" INTEGER NOT NULL CONSTRAINT "PK_Companies" PRIMARY KEY AUTOINCREMENT,
    "BankAccountNumber" TEXT NULL,
    "City" TEXT NULL,
    "CompanyName" TEXT NULL,
    "Discount" REAL NULL,
    "Discriminator" TEXT NOT NULL,
    "Street" TEXT NULL,
    "ZipCode" TEXT NULL
);
CREATE INDEX "IX_Products_SupplierCompanyId" ON "Products" ("SupplierCompanyId");
sqlite>
```

```
❯ dotnet run
/Users/depebul/Projects/Github/BAZY-DANYCH-2/entity-framework/Program.cs(28,17): warning CS8602: Dereference of a possibly null reference.
/Users/depebul/Projects/Github/BAZY-DANYCH-2/entity-framework/Program.cs(38,17): warning CS8602: Dereference of a possibly null reference.
Suppliers from Londym:
- Cuowiek, Jane

Suppliers with 'o' in name:
- Cuowiek
- Produktor

Customers with zip starting with '0':
- Nowak, Zip: 09876

BAZY-DANYCH-2/entity-framework on  master [x?↓] via .NET v9.0.300 🎯 net9.0
➜
```

```
BAZY-DANYCH-2/entity-framework on  master [x?↓] via .NET v9.0.300 🎯 net9.0
➜ sqlite3 MyProductDatabase
SQLite version 3.45.3 2024-04-15 13:34:05
Enter ".help" for usage hints.
sqlite> select * from Suppliers;
Parse error: no such table: Suppliers
sqlite> select * from Customers;
Parse error: no such table: Customers
sqlite> select * from Companies;
1|Nowak|Stara|Wrocław|09876|Customer|10.0|
2|Kowal|Nowa|Kraków|67890|Customer|5.0|
3|Cuowiek|Jane|Londym|12345|Supplier||123456789
4|Produktor|Kowalska|Warszawa|54321|Supplier||987654321
sqlite>
```

Zadanie F.

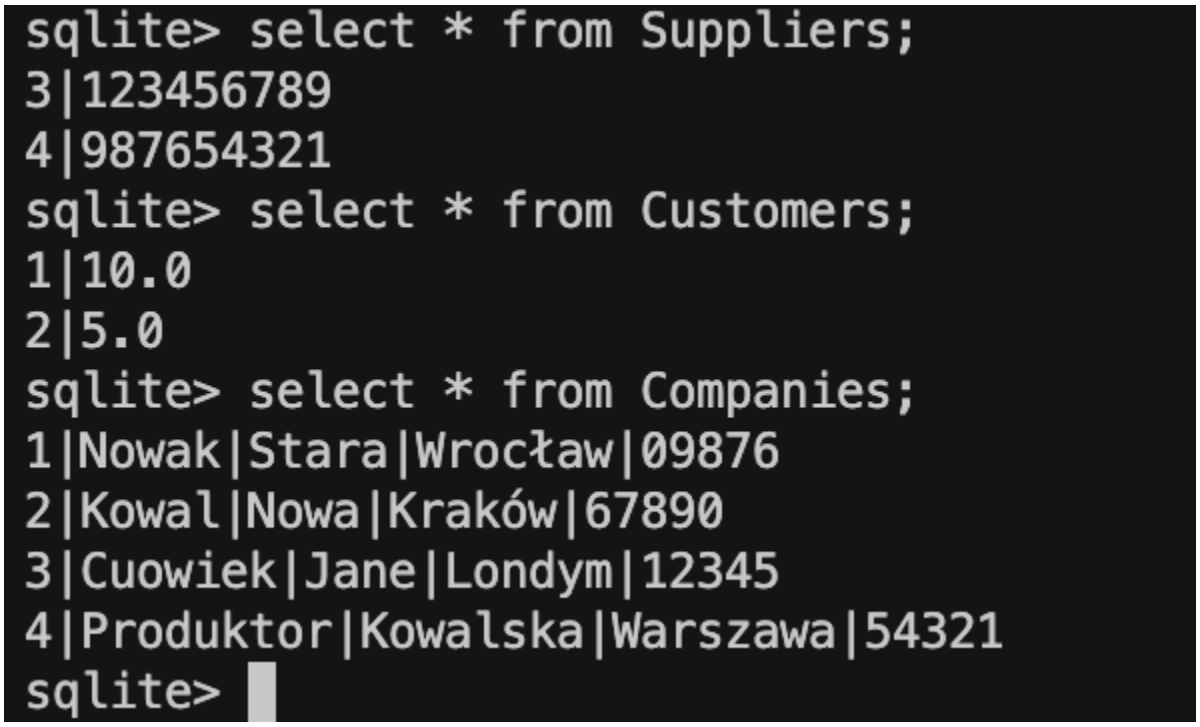
Zamodeluj tę samą hierarchię dziedziczenia, ale tym razem użyj strategii `Table-Per-Type`

- Dodaj i pobierz z bazy kilka firm obu rodzajów.
- Udokumentuj wykonane kroki oraz uzyskany rezultat (.schema table/diagram z datagrip, select * from….)

Plik `ProdContext.cs`:

```
using Microsoft.EntityFrameworkCore;
public class ProdContext : DbContext
{
    public DbSet<Product> Products { get; set; }
    public DbSet<Company> Companies { get; set; }
    public DbSet<Supplier> Suppliers { get; set; }
    public DbSet<Customer> Customers { get; set; }
    public DbSet<Invoice> Invoices { get; set; }

    protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)
    {
        base.OnConfiguring(optionsBuilder);
        optionsBuilder.UseSqlite("Datasource=MyProductDatabase");
    }

    // Dodano sekcję odpowiedzialną za strategię Table Per Type
    protected override void onModelCreating(ModelBuilder modelBuilder)
    {
        base.onModelCreating(modelBuilder);
        modelBuilder.Entity<Company>().ToTable("Companies");
        modelBuilder.Entity<Supplier>().ToTable("Suppliers");
        modelBuilder.Entity<Customers>().ToTable("Customers");
    }
}
```

```
BAZY-DANYCH-2/entity-framework on  master [x?↓] via .NET v9.0.300 🎯 net9.0
➜ dotnet run
Suppliers from Londym:
- Cuowiek, Jane

Suppliers with 'o' in name:
- Cuowiek
- Produktor

Customers with zip starting with '0':
- Nowak, Zip: 09876

BAZY-DANYCH-2/entity-framework on  master [x?↓] via .NET v9.0.300 🎯 net9.0
➜ sqlite3 MyProductDatabase
SQLite version 3.45.3 2024-04-15 13:34:05
Enter ".help" for usage hints.
sqlite> .schema
CREATE TABLE IF NOT EXISTS "__EFMigrationsLock" (
    "Id" INTEGER NOT NULL CONSTRAINT "PK___EFMigrationsLock" PRIMARY KEY,
    "Timestamp" TEXT NOT NULL
);
CREATE TABLE IF NOT EXISTS "__EFMigrationsHistory" (
    "MigrationId" TEXT NOT NULL CONSTRAINT "PK___EFMigrationsHistory" PRIMARY KEY,
    "ProductVersion" TEXT NOT NULL
);
CREATE TABLE IF NOT EXISTS "Companies" (
    "CompanyId" INTEGER NOT NULL CONSTRAINT "PK_Companies" PRIMARY KEY AUTOINCREMENT,
    "CompanyName" TEXT NULL,
    "Street" TEXT NULL,
    "City" TEXT NULL,
    "ZipCode" TEXT NULL
);
CREATE TABLE sqlite_sequence(name,seq);
CREATE TABLE IF NOT EXISTS "Invoices" (
    "InvoiceId" INTEGER NOT NULL CONSTRAINT "PK_Invoices" PRIMARY KEY AUTOINCREMENT,
    "InvoiceNumber" INTEGER NOT NULL,
    "Quantity" INTEGER NOT NULL
);
CREATE TABLE IF NOT EXISTS "Customers" (
    "CompanyId" INTEGER NOT NULL CONSTRAINT "PK_Customers" PRIMARY KEY AUTOINCREMENT,
    "Discount" REAL NOT NULL,
    CONSTRAINT "FK_Customers_Companies_CompanyId" FOREIGN KEY ("CompanyId") REFERENCES "Companies" ("CompanyI
d") ON DELETE CASCADE
);
CREATE TABLE IF NOT EXISTS "Suppliers" (
    "CompanyId" INTEGER NOT NULL CONSTRAINT "PK_Suppliers" PRIMARY KEY AUTOINCREMENT,
    "BankAccountNumber" TEXT NULL,
    CONSTRAINT "FK_Suppliers_Companies_CompanyId" FOREIGN KEY ("CompanyId") REFERENCES "Companies" ("CompanyI
d") ON DELETE CASCADE
);
CREATE TABLE IF NOT EXISTS "Products" (
    "ProductId" INTEGER NOT NULL CONSTRAINT "PK_Products" PRIMARY KEY AUTOINCREMENT,
    "ProductName" TEXT NULL,
    "UnitsInStock" INTEGER NOT NULL,
    "SupplierCompanyId" INTEGER NULL,
    CONSTRAINT "FK_Products_Suppliers_SupplierCompanyId" FOREIGN KEY ("SupplierCompanyId") REFERENCES "Suppli
ers" ("CompanyId")
);
CREATE TABLE IF NOT EXISTS "InvoiceProduct" (
    "InvoicesInvoiceId" INTEGER NOT NULL,
    "ProductsProductId" INTEGER NOT NULL,
    CONSTRAINT "PK_InvoiceProduct" PRIMARY KEY ("InvoicesInvoiceId", "ProductsProductId"),
    CONSTRAINT "FK_InvoiceProduct_Invoices_InvoicesInvoiceId" FOREIGN KEY ("InvoicesInvoiceId") REFERENCES "I
nvoices" ("InvoiceId") ON DELETE CASCADE,
    CONSTRAINT "FK_InvoiceProduct_Products_ProductsProductId" FOREIGN KEY ("ProductsProductId") REFERENCES "P
roducts" ("ProductId") ON DELETE CASCADE
);
CREATE INDEX "IX_InvoiceProduct_ProductsProductId" ON "InvoiceProduct" ("ProductsProductId");
CREATE INDEX "IX_Products_SupplierCompanyId" ON "Products" ("SupplierCompanyId");
sqlite>
```



Zadanie 7.

Porównanie strategii TPH i TPT

TPH (Table Per Hierarchy) - Zadanie E:

- Wszystkie klasy (`Company` , `Supplier` , `Customer`) przechowywane w jednej tabeli `Companies`
- EF dodaje kolumnę dyskryminatora automatycznie do rozróżnienia typów
- Szybsze zapytania (brak JOIN-ów)
- Potencjalnie więcej pustych kolumn dla różnych typów

TPT (Table Per Type) - Zadanie F:

- Każda klasa ma własną tabelę (`Companies` , `Suppliers` , `Customers`)
- Tabele potomne łączą się z tabelą bazową przez klucz obcy
- Bardziej znormalizowana struktura bazy danych

- Wolniejsze zapytania ze względu na konieczność JOIN-ów

W zadaniu ze względu na to, że klasy `Supplier` i `Customer` nie mają dużo unikalnych właściwości, TPH nie jest problemem (nie ma dużo `null`-ów w jednej tabeli). Jednak w przypadku bardziej złożonych hierarchii, TPT może być lepszym rozwiązaniem, ponieważ pozwala na lepszą organizację danych i unikanie dużej liczby kolumn z wartościami `null`.