Laboratorium 3: Entity Framework

Mateusz Surjak

Tydzień A W
t $9{:}35$

1 Wprowadznie

Stworzyłem projekt type Console Application zgodnie z poleceniem.

2 Podpunkt b

Dodaj klasę Product z polami int ProductID, string Name, int UnitsInStock.

```
using System. Collections. Generic;
 3
   using System. Text;
 5
  namespace ISurjakProductEF
 6
 7
       class Product
 8
 9
           public int ProductID { get; set; }
10
           public string Name { get; set; }
11
           public int UnitsInStock { get; set; }
12
13
```

3 Podpunkt c i d

Stwórz klasę ProdContext dziedziczącą po DbContext. Dodaj do klasy kontekstowej zbiór (DbSet) produktów i nazwij go Products.

```
using System;
using System.Collections.Generic;
using Microsoft.EntityFrameworkCore;
using System.Text;

namespace ISurjakProductEF
{
    class ProdContext : DbContext
    {
        public DbSet<Product> Products { get; set; }
}
```

4 Podpunkt e

Dodanie niezbędnej konfiguracji do ProdContext.

Podpunkt e wykonałem zgodnie z poleceniami, dostałem oczekiwane błędy w trakcie wykonywania kolejnych kroków. Kod i screeny zamieszczone w tym podpunkcie są z juz działającej aplikacji po wykonaniu wszystkich kroków.

```
using System;
  using System. Collections. Generic;
 3 using Microsoft.EntityFrameworkCore;
  using System.Text;
  using Microsoft. EntityFrameworkCore. Sqlite;
 6
  namespace ISurjakProductEF
 7
 8
       class ProdContext : DbContext
 9
           public DbSet<Product> Products { get; set; }
10
11
           protected override void OnConfiguring(DbContextOptionsBuilder options)
      => options.UseSqlite("DataSource=Product.db");
12
13
```

```
PM> dotnet ef migrations add InitialProductCreation
Build started...
Build succeeded.
Done. To undo this action, use 'ef migrations remove'
PM> dotnet ef database update
Build started...
Build succeeded.
Applying migration '20200406180039_InitialProductCreation'.
Done.
```

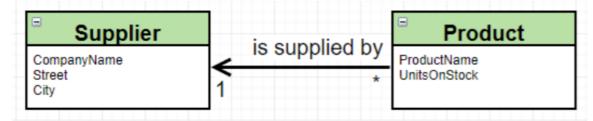
```
using System;
   using System. Linq;
 3
  namespace ISurjakProductEF
 4
 5
       class Program
 6
 7
           static void Main(string[] args)
 8
 9
                string name = Console.ReadLine();
10
                Product product = new Product { Name = name };
11
                ProdContext prodContext = new ProdContext();
12
                prodContext.Products.Add(product);
13
                prodContext.SaveChanges();
14
                var data = (from p in prodContext.Products select p).ToList();
15
                Console. WriteLine("Lista produktow: ");
16
                foreach (var p in data)
17
18
                    Console. WriteLine (p. Name);
19
20
                Console. WriteLine();
21
           }
22
       }
23|}
```

Milk
Lista produktow:
Food
Milk
C:\Users\surja\Documents\Programowanie\NET\ISurjakProductEF\ISurjakProduct
e (proces 18904) zakończono z kodem 0.
Naciśnij dowolny klawisz, aby zamknąć to okno...

Po tych zmianach mogłem już insertować do bazy kolejne produkty.

5 Zadanie II

Zmodyfikuj model wprowadzając pojęcie Dostawcy jak poniżej



```
using System;
   using System. Collections. Generic;
 3
   using System. Component Model. Data Annotations;
4
  using System.Text;
 5
 6
  namespace ISurjakProductEF
 7
8
       class Supplier
9
           [Key]
10
           public int SupplierId { get; set; }
11
           public string CompanyName { get; set; }
12
           public string Street { get; set; }
13
           public string City { get; set; }
14
       }
15
```

```
using System;
   using System. Collections. Generic;
   {\color{red} \mathbf{using}} \ \ System. \ Component Model. \ Data Annotations. Schema;
 4
   using System.Text;
 5
 6
   namespace ISurjakProductEF
 7
 8
        class Product
 9
10
            public int ProductID { get; set; }
11
            public string Name { get; set; }
12
            public int UnitsInStock { get; set; }
13
14
            [ForeignKey("Supplier")]
15
            public int SupplierId { get; set; }
16
            public Supplier supplier { get; set; }
17
        }
18|}
```

Schemat w bazie danych wyglądaą zgodnie z oczekiwaniami.

W celu możliwości dodania Dostawcó do Bazy danych zmodyfikowałem klasę ProdContext dodając kolejny DbSet.

```
using System;
  using System. Collections. Generic;
 3
  using Microsoft. EntityFrameworkCore;
  using System.Text;
 5
  using Microsoft. EntityFrameworkCore. Sqlite;
 6
  namespace ISurjakProductEF
 7
 8
       class ProdContext : DbContext
9
10
           public DbSet<Product> Products { get; set; }
           protected override void OnConfiguring(DbContextOptionsBuilder options)
11
      => options. UseSqlite("Data Source=Product.db");
12
13
           public DbSet<Supplier> Suppliers { get; set; }
14
       }
15
```

```
using System;
  using System. Linq;
 3
  namespace ISurjakProductEF
 4
 5
       class Program
6
 7
           static void Main(string[] args)
8
9
               string name = Console.ReadLine();
10
               Product product = new Product { Name = name };
11
               ProdContext prodContext = new ProdContext();
12
               Supplier supplier = new Supplier { CompanyName = "Google"};
13
               prodContext.Suppliers.Add(supplier);
14
               prodContext.SaveChanges();
15
16
               var sup = (from p in prodContext.Suppliers select p).First();
17
18
               product.SupplierId = sup.SupplierId;
19
               prodContext.Products.Add(product);
20
               prodContext.SaveChanges();
21
               var data = (from p in prodContext.Products select p).ToList();
22
               Console. WriteLine ("Lista produktow: ");
23
               foreach (var p in data)
24
25
                    Console. WriteLine (p. Name);
26
27
               Console. WriteLine();
28
           }
29
       }
30|}
```

```
C:\Users\surja\Documents\Programowanie\NET\ISurjakProductEF\ISurjakProductEF\bin\De
SQLite version 3.28.0 2019-04-16 19:49:53
Enter ".help" for usage hints.
sqlite> select * from Products;
1|Mleko|0|1
sqlite> select * from Supplioers;
Error: no such table: Supplioers
sqlite> select * from Suppliers;
1|Google||
sqlite> _
```

Jak widać baza danych została zamodelowana poprawnie, W tabeli Products mamy klucz obcy który wskazuje na odpowiedniego Dostawcę.

6 Podpunkt g

Wyświetl wszystkie produkty wraz z nazwą dostawcy

```
using System;
 2
   using System. Linq;
 3
  namespace ISurjakProductEF
 4
 5
       class Program
 6
 7
           static void Main(string[] args)
 8
9
                ProdContext prodContext = new ProdContext();
10
11
                var data = (from p in prodContext. Products
12
                             join s in prodContext. Suppliers on p. SupplierId equals s
13
                             SupplierId select new {prod=p.Name, dost=s.CompanyName })
       .ToList();
14
                foreach (var d in data)
15
16
                    Console. WriteLine(d.prod);
17
                    Console. WriteLine(d.dost);
18
                    Console. WriteLine ("
19
20
                }
21
           }
22
       }
23
```

```
Mleko
Google
C:\Users\surja\Documents\Programowanie\NET\ISurjakProductEF\ISurjakProductEF\bin\Debug\netcore
e (proces 18536) zakończono z kodem 0.
Naciśnij dowolny klawisz, aby zamknąć to okno...
```

Dodałem więcej danych w celu zobrazowania że aplikacja napewno działa.

🚳 Konsola debugowania programu Microsoft Visual Studio	_		\times
Marchewka			
Lista produktow: Mleko			
Marchewka			
nal Cliewna			
Mleko			
Google			
Marchewka			
Google			
C:\Users\surja\Documents\Programowanie\NET\ISurjakProductEF\ISurjakProductEF\bin\Debug\netcoreapp3.1\ISu	riakPr	oductEE	ex
e (proces 3588) zakończono z kodem 0.	Jaki i	OddC CLI	
Naciśnij dowolny klawisz, aby zamknąć to okno			
sqlite> select * from Suppliers;			
1 Google			
2 Facebook			
sqlite> select * from Products;			
1 Mleko 0 1			
2 Marchewka 0 1			
sqlite> _			

7 Zadanie III

Odwróć relacje zgodnie z poniższym schematem



Dodałem do klasy Supplier listę produktów ktore dany dostawca dostarcza. Jednocześnie usunąłem polę Supplier z klasy Product.

```
using System;
  using System.Collections.Generic;
   using System.Collections.ObjectModel;
   using System. Component Model. Data Annotations;
   using System. ComponentModel. DataAnnotations. Schema;
   using System.Text;
 6
 7
 8
  namespace ISurjakProductEF
 9
10
       class Supplier
11
12
           public Supplier()
13
14
                Products = new Collection < Product > ();
15
16
17
           [Key]
18
           public int SupplierId { get; set; }
19
           public string CompanyName { get; set; }
20
           public string Street { get; set; }
21
           public string City { get; set; }
22
           public ICollection < Product > Products { get; set; }
23
24
       }
25
```

```
using System;
   using System.Collections.Generic;
 3
   using System. Component Model. Data Annotations. Schema;
   using System.Text;
 4
 5
6
  namespace ISurjakProductEF
 7
8
       class Product
9
10
           public int ProductID { get; set; }
11
           public string Name { get; set; }
12
           public int UnitsInStock { get; set; }
13
14
       }
15
```

```
using Microsoft.EntityFrameworkCore;
  using System;
 3
  using System.Collections.ObjectModel;
  using System. Linq;
 |5|
  namespace ISurjakProductEF
 6
 7
       class Program
 8
 9
           static void Main(string[] args)
10
11
                ProdContext prodContext = new ProdContext();
12
                string name = Console.ReadLine();
13
                Product product = new Product { Name = name };
14
                prodContext.Products.Add(product);
15
                Supplier supplier = prodContext.Suppliers.FirstOrDefault(b => b.
      CompanyName == "Amazon");
16
                if (supplier = null)
17
18
                    supplier = new Supplier { CompanyName = "Amazon" };
19
20
                    prodContext.Suppliers.Add(supplier);
21
22
                supplier . Products . Add(product);
23
                prodContext.SaveChanges();
24
25
26
                var data1 = prodContext.Suppliers.Include(s \Rightarrow s.Products).ToList();
27
                Console. WriteLine ("Lista supplierow: ");
28
                foreach (var s in data1)
29
30
                    Console. WriteLine (s. CompanyName);
31
                    foreach (var p in s. Products)
32
33
                        Console. WriteLine (p. Name);
34
                    Console . WriteLine ( "————" ) ;
35
36
                }
37
38
39
           }
40
       }
41|
```

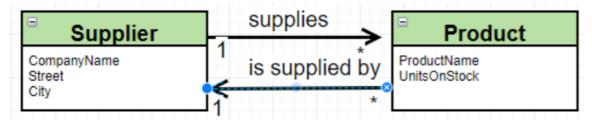
Aplikacja działa zgodnie z oczekiwaniami, Produkty dostawcy "Amazon" wyświetliły się poprawnie.

Poniżej przedstawiam jak wygląda baza danych w tym momencie.

```
sqlite> select * from Products;
10|Jajka|0|2
11|Kasza|0|2
12|Pizza|0|2
13|Pierogi|0|2
14|Brokul|0|2
15|Zaberka|0|2
16|Ziemniaki|0|2
17|Kaszanka|0|2
sqlite> select * from Suppliers;
2|Amazon||
sqlite>
```

8 Zadanie IV

Zamodeluj relacje dwustronną jak poniżej:



Do klasy Product dodałem Dostawce aby można było się do niego bezpośrednio odwoływać.

```
using System;
 2
   using System. Collections. Generic;
 3
   using System. Collections. ObjectModel;
  using System. Component Model. Data Annotations;
  using System. Component Model. Data Annotations. Schema;
6
  using System. Text;
8
  namespace ISurjakProductEF
9
10
       class Supplier
11
12
           public Supplier()
13
14
                Products = new Collection < Product > ();
15
16
17
           public int SupplierId { get; set; }
18
           public string CompanyName { get; set; }
19
           public string Street { get; set; }
20
           public string City { get; set; }
21
           public virtual ICollection < Product > Products { get; set; }
22
23
       }
24
```

```
1
   using System;
   using System. Collections. Generic;
   using System. Collections. ObjectModel;
  using System. Component Model. Data Annotations;
 5
   using System. Component Model. Data Annotations. Schema;
6
   using System.Text;
 7
8
   namespace ISurjakProductEF
9
10
       class Product
11
12
13
           public int ProductId { get; set; }
14
           public string Name { get; set; }
15
           public int UnitsInStock { get; set; }
16
           public Supplier Supplier { get; set; }
17
       }
18|}
```

```
{\color{red} \textbf{using}} \quad \textbf{Microsoft} . \\ \textbf{EntityFrameworkCore} \ ; \\
   using System;
 3
   using System. Collections. ObjectModel;
  using System. Linq;
 |5|
  namespace ISurjakProductEF
 6
 7
       class Program
 8
 9
            static void Main(string[] args)
10
11
                ProdContext prodContext = new ProdContext();
12
                string name = Console.ReadLine();
13
                Product product = new Product { Name = name };
14
                prodContext.Products.Add(product);
15
                Supplier supplier = prodContext.Suppliers.FirstOrDefault(b => b.
      CompanyName == "Amazon");
16
                if (supplier = null)
17
18
                     supplier = new Supplier { CompanyName = "Amazon" };
19
20
                     prodContext.Suppliers.Add(supplier);
21
22
                product. Supplier = supplier;
23
                supplier . Products . Add(product);
24
                prodContext.SaveChanges();
25
26
27
                var data1 = prodContext.Suppliers.Include(s \Rightarrow s.Products).ToList();
28
                Console. WriteLine ("Lista supplierow: ");
29
                foreach (var s in data1)
30
                {
31
                     Console. WriteLine (s. CompanyName);
32
                     foreach (var p in s. Products)
33
34
                         Console. WriteLine (p. Name);
35
                     Console. WriteLine("———");
36
37
38
                Console.WriteLine("——PRODUCTS——");
39
                var data2 = prodContext.Products.Include(p => p.Supplier).ToList();
40
                foreach(var p in data2)
41
                     Console. WriteLine (p. Supplier. CompanyName);
42
43
44
45
46
            }
47
       }
48| \}
```

```
nonsola acpagowania programa iviicrosoft visuai studio
Lista supplierow:
Amazon
Schabowy
Jajka
(asza
Pizza
Pierogi
Brokul
Zaberka
Ziemniaki
(aszanka
Salami
Mozarella
---PRODUCTS---
Amazon
 mazon
```

Można zauważyć że dostawca produktu wypisuje się dobrze. Dla każdego produktu wypisał się Amazon co jest zgodne z zawartością bazy danych którą przedstawiam poniżej.

```
sqlite> select * from Products;

10|Jajka|0|2

11|Kasza|0|2

12|Pizza|0|2

13|Pierogi|0|2

14|Brokul|0|2

15|Zaberka|0|2

16|Ziemniaki|0|2

17|Kaszanka|0|2

18|Salami|0|2

19|Mozarella|0|2
```

9 Zadanie V

Dodaj klase Category z property int Category ID, String Name oraz listą produktow

```
using System;
   using System.Collections.Generic;
 2
 3
   using System. Text;
 4
5
  namespace ISurjakProductEF
 6
 7
       class Category
8
9
           public int CategoryId { get; set; }
10
           public string Name { get; set; }
           public List<Product> Products { get; set; }
11
12
13|}
```

```
1
  using System;
  using System. Collections. Generic;
  using System. Collections. ObjectModel;
  using System. Component Model. Data Annotations;
 5
  using System. Component Model. Data Annotations. Schema;
6
  using System. Text;
8
  namespace ISurjakProductEF
9
10
       class Product
11
12
           public int ProductId { get; set; }
13
           public string Name { get; set; }
           public int UnitsInStock { get; set; }
14
15
           public Supplier Supplier { get; set; }
16
           public Category Category { get; set; }
17
       }
18|}
```

Dodałem Category do Contextu w celu insertu do bazy danych.

```
using System;
   using System. Collections. Generic;
 3
  using Microsoft. EntityFrameworkCore;
  using System. Text;
 5
  using Microsoft. EntityFrameworkCore. Sqlite;
 6
  namespace ISurjakProductEF
 7
 8
       class ProdContext : DbContext
 9
10
           public DbSet<Product> Products { get; set; }
11
           protected override void OnConfiguring (DbContextOptionsBuilder options)
      => options. UseSqlite("Data Source=Product.db");
12
13
           public DbSet<Supplier> Suppliers { get; set; }
14
15
           public DbSet<Category> Categories { get; set; }
16
       }
17
```

```
1
  namespace ISurjakProductEF
 2
  {
 3
       class Program
 4
 5
           static void Main(string[] args)
 6
 7
                Category category1 = new Category { Name = "Food" };
 8
                Category category = new Category { Name = "Drinks" };
 9
10
                ProdContext prodContext = new ProdContext();
11
12
                prodContext. Categories . Add(category1);
13
                prodContext. Categories . Add(category2);
14
15
                string companyName = "Facebook";
16
                string productName = "Kasza";
17
18
                Product product = new Product { Name = productName };
19
                product.Category = category1;
20
                Product product1 = new Product { Name = "Marchew" };
21
                product1. Category = category2;
22
23
                prodContext . Products . Add( product ) ;
24
25
                Supplier supplier = prodContext.Suppliers.FirstOrDefault(b => b.
      CompanyName = companyName);
26
                if (supplier = null)
27
28
                    supplier = new Supplier { CompanyName = companyName };
29
                    prodContext.Suppliers.Add(supplier);
30
31
                product.Supplier = supplier;
32
                product1.Supplier = supplier;
33
                supplier . Products . Add(product);
34
                supplier . Products . Add(product1);
35
                prodContext.SaveChanges();
36
37
                var data1 = prodContext.Suppliers.Include(s \Rightarrow s.Products).ToList();
38
                Console.WriteLine("Lista supplierow: ");
39
                foreach (var s in data1)
40
41
                    Console. WriteLine (s. CompanyName);
42
                    Console. WriteLine ("Produkty dla firmy: ");
43
                    foreach (var p in s. Products)
44
45
                        Console. WriteLine (p. Name);
46
                    Console. WriteLine("———");
47
                }
48
49
50
                Console. WriteLine ("——PRODUCTS——");
51
                var data2 = prodContext.Products.Include(p => p.Supplier).ToList();
52
                foreach (var p in data2)
53
54
                    Console. WriteLine (p. Supplier. CompanyName);
55
56
                var productsFromCatergory = prodContext.Categories.Include(c => c.
      Products). Where (c \Rightarrow c.Name = "Warzywa");
57
                foreach (var c in productsFromCatergory)
58
                {
```

```
59
                    foreach (var p in c. Products)
60
61
                         Console. WriteLine (p. Name);
62
63
                }
64
65
                var categoryFromProd = prodContext.Products.Where(p => p.Name="Kiwi
      "). Include (c => c. Category). FirstOrDefault();
66
                Console. WriteLine (categoryFromProd. Category.Name);
67
68
       }
69
```

Przedstawiam wygląd bazy danych po wprowadzonych modyfikacjach.

```
SQLite version 3.28.0 2019-04-16 19:49:59
Enter ".help" for usage hints.
sqlite> .tables
Categories
Products
sqlite
                                                  Suppliers
___EFMigrationsHistory
  sqlite> select * from Categories;
 1|Food
2|Drinks
2|Drinks
sqlite> select * from Suppliers;
1|Facebook||
sqlite> select * from Products;
1|Kasza|0|1|1
2|Marchew|0|1|2
sqlite> .schema Products;
sqlite> .schema Products
CREATE TABLE IF NOT EXISTS "Products" (
    "ProductId" INTEGER NOT NULL CONSTRAINT "PK_Products" PRIMARY KEY AUTOINCREMENT,
    "Name" TEXT NULL,
    "UnitsInStock" INTEGER NOT NULL,
    "SupplierId" INTEGER NULL,
         "UnitsInstock" INTEGER NOT NULL,
"SupplierId" INTEGER NULL,
"CategoryId" INTEGER NULL,
CONSTRAINT "FK_Products_CategoryId" FOREIGN KEY ("CategoryId") REFERENCES "Categories" ("CategoryId") ON DELETE RESTRICT,
CONSTRAINT "FK_Products_Suppliers_SupplierId" FOREIGN KEY ("SupplierId") REFERENCES "Suppliers" ("SupplierId") ON DELETE RESTRICT,
"City" TEXT NULL
 );
sqlite> .schema Categories
CREATE TABLE IF NOT EXISTS "Categories" (
"CategoryId" INTEGER NOT NULL CONSTRAINT "PK_Categories" PRIMARY KEY AUTOINCREMENT,
"Name" TEXT NULL
   ,
qlite>
```

Wydobądź produkty z wybranej kategorii 9.1

Kod do tego podpunktu znajduje się mpowyżej w klasie Program

```
Konsola debugowania programu Microsoft Visual Studio
                                                                                                                   X
 \Users\surja\Documents\Programowanie\NET\ISurjakProductEF\ISurjakProductEF\bin\Debug\netcoreapp3.1\ISurjakProductEF.ex
 (proces 22796) zakończono z kodem 0.
laciśnij dowolny klawisz, aby zamknąć to okno...
```

9.2 Wydobądź kategorię do której należy wybrany produkt

Kod do tego podpunktu znajduje się mpowyżej w klasie Program

Konsola debugowania programu Microsoft Visual Studio

```
Kiwi

Melon

Kasza

Burak

C:\Users\surja\Documents\Programowanie\NET\ISurjakProductEF\ISurjakProductEF\bin\Debug\netcoreapp3.1\ISurjakProductEF.exe

e (proces 16784) zakończono z kodem 0.

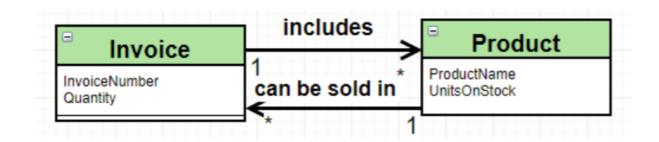
Naciśnij dowolny klawisz, aby zamknąć to okno...

-
```

10 Zadanie VI

Zamodeluj relacje wiele-do-wielu, jak poniżej:

W celu wykonania tego podpunktu konieczne było stworzenie nowego obiektu InvoiceProduct który przechowuje relacje pomiędzy Invoice a Product. W klasach Product i Invoice stworzyłem kolekcję obiektów InvoiceProducts w celu wyrażenia relacji many-to-many.



```
using System;
   using System. Collections. Generic;
 3
   using System. Collections. ObjectModel;
   {\color{red} \mathbf{using}} \ \ System. \ Component Model. \ Data Annotations;
 5
   using System. Component Model. Data Annotations. Schema;
 6
   using System.Text;
 7
 8
   namespace ISurjakProductEF
 9
10
       class Supplier
11
12
            public Supplier()
13
14
                 Products = new Collection < Product > ();
15
16
            public int SupplierId { get; set; }
            public string CompanyName { get; set; }
17
18
            public string Street { get; set;
19
            public string City { get; set; }
20
            public virtual ICollection < Product > Products { get; set; }
21
22
       }
23
```

```
using System;
   using System. Collections. Generic;
 3
   using System. Collections. ObjectModel;
  using System. Text;
 5
6
  namespace ISurjakProductEF
 7
8
       class Category
9
10
           public Category()
11
12
                Products = new Collection < Product > ();
13
14
           public int CategoryId { get; set; }
15
           public string Name { get; set; }
16
           public ICollection < Product > Products { get; set; }
17
       }
18|}
```

```
using System;
  using System. Collections. Generic;
 3
  using System. Component Model. Data Annotations;
4
  using System. Text;
6
  namespace ISurjakProductEF
 7
8
       class InvoiceProduct
9
10
11
           public int ProductId { get; set; }
12
           public Product Product { get; set; }
13
           public int InvoiceId { get; set; }
14
           public Invoice Invoice { get; set; }
15
       }
|16|
```

```
using System;
  using System. Collections. Generic;
 3
   using System. Collections. ObjectModel;
4
   using System. Text;
5
  namespace ISurjakProductEF
6
 7
8
       class Invoice
9
10
           public Invoice()
11
12
                InvoiceProducts = new Collection < InvoiceProduct > ();
13
14
           public int InvoiceId { get; set; }
15
16
           public int InvoiceNumber { get; set; }
17
18
           public int Quantity { get; set; }
19
20
           public ICollection < InvoiceProduct > InvoiceProducts { get; set; }
21
       }
22|}
```

```
using System;
   using System. Collections. Generic;
 3
   using System. Collections. ObjectModel;
   using System. Component Model. Data Annotations;
 5
   using System. Component Model. Data Annotations. Schema;
 6
   using System. Text;
 7
8
  namespace ISurjakProductEF
9
10
       class Product
11
12
           public Product()
13
14
                InvoiceProducts = new Collection < InvoiceProduct > ();
15
16
17
           public int ProductId { get; set; }
18
           public string Name { get; set; }
19
           public int UnitsInStock { get; set; }
20
           public Supplier Supplier { get; set; }
21
22
           public Category Category { get; set; }
23
           public ICollection < InvoiceProduct > InvoiceProducts { get; set; }
24
       }
25|}
```

Oprócz tego dodałem brakujące kolekcje do Contextu.

```
using System;
   using System. Collections. Generic;
 2
 3
  using Microsoft. EntityFrameworkCore;
  using System. Text;
 5
  using Microsoft. EntityFrameworkCore. Sqlite;
 6
  namespace ISurjakProductEF
 7
8
       class ProdContext : DbContext
9
       {
           public DbSet<Product> Products { get; set; }
10
11
           protected override void OnConfiguring(DbContextOptionsBuilder options)
      => options. UseSqlite("Data Source=Product.db");
12
13
           public DbSet<Supplier> Suppliers { get; set; }
14
15
           public DbSet<Category> Categories { get; set; }
16
17
           public DbSet<Invoice> Invoices { get; set; }
18
19
           public DbSet<InvoiceProduct> invoiceProducts { get; set; }
20
21
           protected override void
22
            OnModelCreating (ModelBuilder modelBuilder)
23
24
                modelBuilder. Entity < Invoice Product > ()
25
                    . HasKey(x => new { x. ProductId, x. InvoiceId});
26
           }
27
28
29
       }
30| }
```

```
using Microsoft. EntityFrameworkCore;
  2
      using System;
  3
     using System. Collections. ObjectModel;
     using System. Linq;
  5
     namespace ISurjakProductEF
  6
  7
               class Program
 8
 9
                        static void Main(string[] args)
10
11
12
                                 ProdContext prodContext = new ProdContext();
13
                                 Product product1 = new Product { Name = "Kurczak" };
14
                                 Product product2 = new Product { Name = "Cukinia"
15
                                 Product product3 = new Product { Name = "Maka" };
16
                                 Product product4 = new Product { Name = "Ogorek" };
                                 Category category = new Category { Name = "Owoce" };
17
18
                                 category. Products. Add(product1);
19
                                 category. Products. Add(product2);
20
                                 category. Products. Add(product3);
21
                                 category. Products. Add(product4);
22
                                 product1. Category = category;
23
                                 product2.Category = category;
24
                                 product3.Category = category;
25
                                 product4.Category = category;
26
                                 Supplier supplier = new Supplier { CompanyName = "Facebook" };
27
                                 supplier . Products . Add(product1);
28
                                 supplier.Products.Add(product2);
29
                                 supplier.Products.Add(product3);
30
                                 supplier. Products. Add(product4);
31
                                 product1.Supplier = supplier;
32
                                 product2.Supplier = supplier;
33
                                 product3.Supplier = supplier;
34
                                 product4. Supplier = supplier;
35
                                prodContext.Categories.Add(category);
36
                                 prodContext.Products.Add(product1);
37
                                 prodContext.Products.Add(product2);
38
                                 prodContext . Products . Add( product3 ) ;
39
                                 prodContext . Products . Add(product4);
40
                                 prodContext.Suppliers.Add(supplier);
41
42
                                 Invoice invoice1 = new Invoice { InvoiceNumber = 1, Quantity = 3 };
43
                                 Invoice invoice2 = new Invoice { InvoiceNumber = 2, Quantity = 2 };
44
                                 prodContext.Invoices.Add(invoice1);
45
                                 prodContext . Invoices . Add(invoice2);
46
47
                                InvoiceProduct invoiceProduct1 = new InvoiceProduct { Invoice =
             invoice1, Product = product1 \};
48
                                invoice1. InvoiceProducts. Add(invoiceProduct1);
49
                                InvoiceProduct invoiceProduct2 = new InvoiceProduct { Invoice =
             invoice1 , Product = product2 };
50
                                invoice1 . InvoiceProducts . Add(invoiceProduct2);
51
                                InvoiceProduct invoiceProduct3 = new InvoiceProduct { Invoice =
             invoice2 , Product = product3 };
52
                                invoice2.InvoiceProducts.Add(invoiceProduct3);
53
                                Invoice Product \ invoice Product 4 = {\color{red} new} \ Invoice Product \ \{ \ Invoice = {\color{red} new} \ {\color{red} Invoice} = {\color{red} Invoice} = {\color{red} new} \ {\color{red} Invoice} = {\color{red} new} \ {\color{red} Invoice} = {\color{red} I
             invoice2, Product = product4 \;
54
                                 invoice2 . InvoiceProducts . Add(invoiceProduct4);
55
                                 prodContext.invoiceProducts.Add(invoiceProduct1);
56
                                prodContext.invoiceProducts.Add(invoiceProduct2);
```

```
57
                prodContext.invoiceProducts.Add(invoiceProduct3);
58
                prodContext.invoiceProducts.Add(invoiceProduct4);
59
60
                product1.InvoiceProducts.Add(invoiceProduct1);
61
                product2.InvoiceProducts.Add(invoiceProduct2);
62
                product3.InvoiceProducts.Add(invoiceProduct3);
63
                product4.InvoiceProducts.Add(invoiceProduct4);
64
65
                prodContext.SaveChanges();
66
67
                var products = prodContext.invoiceProducts.Include(d => d.Product).
      Where (d \Rightarrow d. InvoiceId == 3). Select (d \Rightarrow d. Product. Name). ToList();
68
                var invoices = prodContext.invoiceProducts.Include(d => d.Invoice).
      Where (p => p. ProductId == 1). Select (d => d. Invoice. InvoiceNumber);
69
70
                foreach (var p in products)
71
72
                    Console. WriteLine(p);
73
74
75
                foreach (var p in invoices)
76
77
                    Console. WriteLine(p);
78
79
80
                var productsFromCatergory = prodContext.Categories.Include(c => c.
      Products). Where (c \Rightarrow c.Name = "Warzywa");
81
                foreach (var c in productsFromCatergory)
82
83
                    foreach (var p in c.Products)
84
85
                         Console. WriteLine (p. Name);
86
87
                }
88
89
                var categoryFromProd = prodContext.Products.Where(p => p.Name="Kiwi
      "). Include (c => c. Category). FirstOrDefault();
90
                Console. WriteLine (categoryFromProd. Category.Name);
91
           }
92
       }
93|}
```

Prezentuję wygląd bazy danych

```
sqlite> select * from Products;
1|Kiwi|0|1|1
2|Melon|0|1|1
3|Kasza|0|1|1
4|Burak|0|1|1
5 | Kurczak | 0 | 2 | 2
6|Cukinia|0|2|2
7 | Maka | 0 | 2 | 2
8|Ogorek|0|2|2
sqlite> select * from Categories;
1 Warzywa
2 Owoce
sqlite> select * from Suppliers;
1|Google||
2 | Facebook | |
sqlite> select * from Invoices;
1 | 1 | 3
2|2|2
3|1|3
4|2|2
sqlite> select * from InvoiceProducts;
1 | 1
2 | 1
3 | 2
4 | 2
5 | 3
6 | 3
7 | 4
8 4
sqlite>
```

10.1 Pokaż produkty sprzedane w ramach wybranej faktury/transakcji

Kod do tego podpunktu znajduje się pozyżej w klasie Program w linii 67

```
Kurczak
Cukinia
C:\Users\surja\Documents\Programowanie\NET\ISurjakProductEF\ISurjakProductEF\bin\Debug\netcoreapp3.1\ISurjakProductEF.ex
e (proces 24840) zakończono z kodem 0.
Naciśnij dowolny klawisz, aby zamknąć to okno...
-
```

10.2 Pokaż faktury w ramach których był sprzedany wybrany produkt

Kod do tego podpunktu znajduje się pozyżej w klasie Program w linii 68

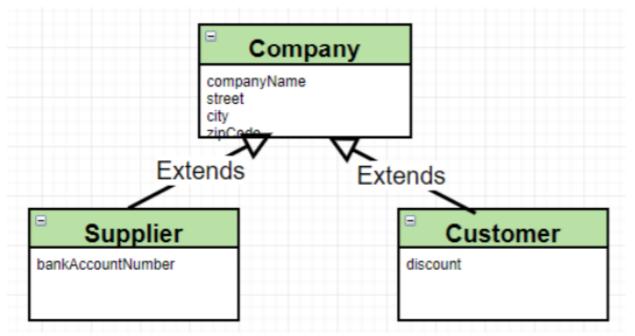
```
Kurczak
Cukinia

1

C:\Users\surja\Documents\Programowanie\NET\ISurjakProductEF\ISurjakProductEF\bin\Debug\netcoreapp3.1\ISurjakProductEF.e
e (proces 21044) zakończono z kodem 0.
Naciśnij dowolny klawisz, aby zamknąć to okno...
```

11 Zadanie VII

Wprowadź do modelu następującą hierarchie:



```
using System;
 2
   using System. Collections. Generic;
 3
   using System. Collections. ObjectModel;
  using System. Component Model. Data Annotations;
 5
  using System. Component Model. Data Annotations. Schema;
6
   using System. Text;
8
  namespace ISurjakProductEF
9
   {
10
       class Supplier: Company
11
12
            public Supplier()
13
14
                Products = new Collection < Product > ();
15
            public int BankAccountNumber { get; set; }
16
17
            public virtual ICollection < Product > Products { get; set; }
18
       }
19
```

```
using System;
 2 using System. Collections. Generic;
 3 using System. Text;
4
5
  namespace ISurjakProductEF
 6
 7
       class Company
 8
9
           public int CompanyId { get; set; }
10
           public string CompanyName { get; set; }
11
           public string Street { get; set; }
           public string City { get; set; }
12
13
14
           public string ZipCode { get; set; }
15
       }
16|}
```

11.1 TablePerHierarchy

Do wykonanania TablePerHierarchy wprowadziłem zmiany w Contexie w metodzie OnModel-Creating.

```
using System;
  using System. Collections. Generic;
  using Microsoft. EntityFrameworkCore;
  using System. Text;
 5
  using Microsoft. EntityFrameworkCore. Sqlite;
 6
  namespace ISurjakProductEF
 7
8
       class ProdContext : DbContext
9
10
           public DbSet<Product> Products { get; set; }
11
           protected override void OnConfiguring(DbContextOptionsBuilder options)
      => options. UseSqlite("Data Source=Product.db");
12
13
           public DbSet<Company> Companies { get; set; }
14
15
16
           public DbSet<Category> Categories { get; set; }
17
18
19
           public DbSet<Invoice> Invoices { get; set; }
20
21
           public DbSet<InvoiceProduct> invoiceProducts { get; set; }
22
23
           protected override void
24
            OnModelCreating (ModelBuilder modelBuilder)
25
26
                modelBuilder.Entity<InvoiceProduct>()
27
                    . HasKey(x \Rightarrow new \{ x.ProductId, x.InvoiceId \});
28
                modelBuilder. Entity < Customer > ();
29
                modelBuilder.Entity<Supplier>();
30
           }
31
32
33
       }
34
```

```
using Microsoft. EntityFrameworkCore;
 3
  using System;
  using System.Collections.ObjectModel;
 4
 5
  using System. Linq;
 6
  namespace ISurjakProductEF
 7
 8
       class Program
9
       {
10
           static void Main(string[] args)
11
           {
12
13
                ProdContext prodContext = new ProdContext();
14
                Customer customer = new Customer
15
16
                    City = "Krakow",
17
                    CompanyName = "PegaSystems",
18
                    Street = "Puszkarska",
19
                    ZipCode = "30-551",
20
                    Discount = 12,
```

```
21
22
23
                 Supplier supplier = new Supplier
24
25
                     City = "Manchester",
26
                     CompanyName = "Sabre".
27
                     Street = "Aleja pokoju",
28
                     ZipCode = "30-333",
29
                     BankAccountNumber = 123123123
30
31
                 };
32
                 Customer\ customer2 = new\ Customer
33
34
                     City = "Warsaw",
35
                     CompanyName = "Qualticks",
                     Street = "Main street",
ZipCode = "20-551",
36
37
38
                     Discount = 55
39
40
41
                 Supplier supplier 2 = new Supplier
42
43
                     City = "London",
                     CompanyName = "SM",
44
                     Street = "LondonStreet",
ZipCode = "30-311",
45
46
47
                     BankAccountNumber = 1231666
48
49
50
                 prodContext.Companies.Add(customer);
51
                 prodContext.Companies.Add(customer2);
52
                 prodContext.Companies.Add(supplier);
53
                 prodContext.Companies.Add(supplier2);
54
55
                 prodContext.SaveChanges();
56
57
            }
58
       }
59|}
```

```
sqlite> select * from Companies;
1|PegaSystems|Puszkarska|Krakow|30-551|Customer||12
sqlite> select * from Companies;
1|PegaSystems|Puszkarska|Krakow|30-551|Customer||12
2|PegaSystems|Puszkarska|Krakow|30-551|Customer||12
3|Qualticks|Main street|Warsaw|20-551|Customer||55
4|Sabre|Aleja pokoju|Manchester|30-333|Supplier|123123123|
5|SM|LondonStreet|London|30-311|Supplier|1231666|
sqlite>
```

11.2 Pobranie danych

Pobrałem wszystkich Customerów w celu zbadania działania metody OfType, metoda wybrałą tylko pola powiązane z Customerem z Dazy danych pomiojając obiekty które są typu Supplier. Wyświetliłem property które posiadają tylko Customerzy.

11.3 TablePerType

Starałem się zamodelować bazę zgodnie z dokumentacją. Niestety jak się okazało w wersji ponad 3.0 Entity Framework nie da się skorzystać z TPT. Pokażę co otrzymałem po próbach insertu do bazy.

```
2
   using System;
 3
   using System. Collections. Generic;
   using System. Component Model. Data Annotations. Schema;
 5
   using System. Text;
 6
 7
   namespace ISurjakProductEF
8
9
       [Table ("Customers")]
10
       class Customer: Company
11
12
            public int Discount { get; set; }
13
14
```

```
using System;
   using System. Collections. Generic;
   using System. Collections. ObjectModel;
 4
   using System.ComponentModel.DataAnnotations;
 5
   using System. Component Model. Data Annotations. Schema;
 6
  using System. Text;
 7
8
  namespace ISurjakProductEF
9
10
       [Table ("Suppliers")]
11
       class Supplier : Company
12
13
            public Supplier()
14
15
                Products = new Collection < Product > ();
16
17
            public int BankAccountNumber { get; set; }
18
19
            public virtual ICollection < Product > Products { get; set; }
20
21
       }
22
```

```
using System;
  using System. Collections. Generic;
 3
  using Microsoft. EntityFrameworkCore;
  using System. Text;
 5
  using Microsoft. EntityFrameworkCore. Sqlite;
6
  namespace ISurjakProductEF
 7
8
       class ProdContext : DbContext
9
10
           public DbSet<Product> Products { get; set; }
11
           protected override void OnConfiguring(DbContextOptionsBuilder options)
      => options. UseSqlite("Data Source=Product.db");
12
13
           public DbSet<Company> Companies { get; set; }
14
15
```

```
16
             public DbSet<Category> Categories { get; set; }
17
18
19
             public DbSet<Invoice> Invoices { get; set; }
20
21
             public DbSet<InvoiceProduct> invoiceProducts { get; set; }
22
23
             protected override void
24
               OnModelCreating(ModelBuilder modelBuilder)
25
26
                  modelBuilder. Entity < Invoice Product > ()
27
                        . \operatorname{HasKey}(x \Rightarrow \text{new } \{ x. \operatorname{ProductId}, x. \operatorname{InvoiceId} \});
28
             }
29
30
31
        }
32
```

Spójrzmy na wygląd bazy poniżej. Typ Supplier zmapował się dobrze i jest przechowywany, lecz typ Customer nie jest przechowywany, jego property (Discount) jest ignorowane prze insercie. Sam typ jest przechowywany jako Company (Na co wskazuje 6-ta kolumna).

```
SQLite version 3.28.0 2019-04-16 19:49:53
Enter ".help" for usage hints.
sqlite> select * from Companies;
sqlite> select * from Companies;
sqlite> select * from Companies;
1|PegaSystems|Puszkarska|Krakow|30-551|Company|
2|Qualticks|Main street|Warsaw|20-551|Company|
3|Sabre|Aleja pokoju|Manchester|30-333|Supplier|123123123
4|SM|LondonStreet|London|30-311|Supplier|1231666
sqlite>
```

11.4 Wypisanie danych

Wypisanie danych dla Suppliera jest dobre, lecz dla Customera dostajemy błędy.

```
1 using Microsoft. EntityFrameworkCore;
  using System;
 3
  using System.Collections.ObjectModel;
 4
  using System. Linq;
 5
  namespace ISurjakProductEF
 6
 7
       class Program
8
9
           static void Main(string[] args)
10
11
12
                ProdContext prodContext = new ProdContext();
13
                Customer customer = new Customer
14
15
                    City = "Krakow",
                    CompanyName = "PegaSystems",
16
17
                    Street \ = \ "Puszkarska" \, ,
18
                    ZipCode = "30-551",
19
                    Discount = 12
20
```

```
21
22
                 Supplier supplier = new Supplier
23
24
                     City = "Manchester",
25
                     CompanyName = "Sabre",
26
                     Street = "Aleja pokoju",
                     ZipCode = "30-333",
27
28
                     BankAccountNumber = 123123123
29
30
31
                Customer\ customer2 = new\ Customer
32
33
                     City = "Warsaw",
34
                     CompanyName = "Qualticks",
35
                     Street = "Main street",
                     ZipCode = "20-551",
36
37
                     Discount = 55
38
39
                 };
40
                Supplier supplier 2 = new Supplier
41
42
                     City = "London",
43
                     CompanyName = "SM",
                     Street = "LondonStreet",
ZipCode = "30-311",
44
45
46
                     BankAccountNumber = 1231666
47
48
                 };
49
                 prodContext.Companies.Add(customer);
50
                 prodContext . Companies . Add(customer2);
51
                prodContext.Companies.Add(supplier);
52
                prodContext\,.\,Companies\,.\,Add(\,supplier\,2\,)\,;
53
                 var data = prodContext.Companies.OfType<Supplier >().ToList();
54
                 foreach (var d in data)
55
56
                     Console. WriteLine (d. BankAccountNumber);
57
58
                prodContext.SaveChanges();
59
60
            }
61
62|}
```

Konsola acpagowania programa iviicrosoft visual staalo

```
123123123
1231666
```

11.5 TablePerClass

Niestety nie byłem w stanie wykonać tej części zadania gdyż EntityFramework od wersji 3.0 podczas używania metody ToTable() która jest niezbędna do wykonania tego podpunktu , rzuca błąd zgodnie z dokumentacją.

Poniżej prezentuję screena z dokumentacji który potwierdza takie działanie aplikacji.

Stare zachowanie

Przed EF Core 3,0, ToTable() wywołana dla typu pochodnego zostałaby zignorowana, ponieważ strategia mapowania dziedziczenia była TPH, gdzie jest to nieprawidłowe.

Nowe zachowanie

Począwszy od EF Core 3,0 i przygotowania do dodawania obsługi TPT i TPC w późniejszej wersji, ToTable() wywołana dla typu pochodnego spowoduje teraz zgłoszenie wyjątku, aby uniknąć nieoczekiwanej zmiany mapowania w przyszłości.