

Laboratorium 3
Entity Framework
Paweł Kocimski

Zadanie I

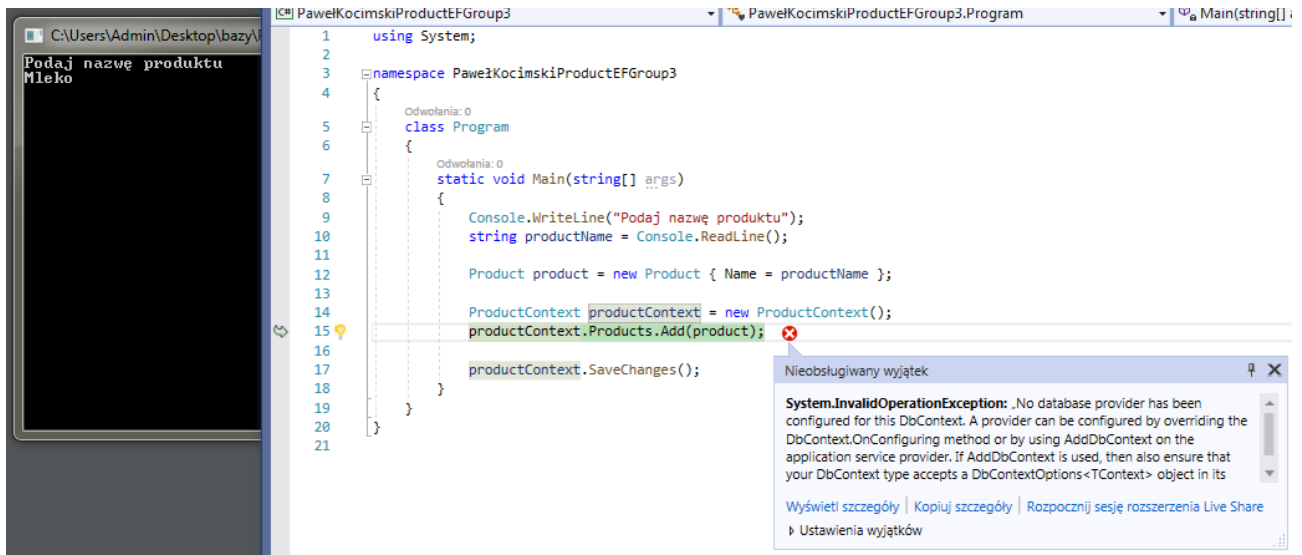
1. Stwórz projekt typu ConsoleApplication .Net Core
2. Dodaj klasę Product z polami int ProductID, string Name, int UnitsInStock

```
4
5 namespace PawełKocińskiProductEFGroup3
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 }
14
15
```

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

```
5
6 namespace PawełKocińskiProductEFGroup3
7 {
8     class ProductContext : DbContext
9     {
10        public DbSet<Product> Products { get; set; }
11    }
12 }
13
14
```

4. W Mainie :
 - poproś użytkownika o podanie nazwy produktu i czytaj podaną przez użytkownika nazwę
 - zainstancjonuj obiekt produktu ustawiając mu nazwę na tą czytaną od użytkownika
 - Stwórz instancję ProdContext'u
 - dodaj zainstancjonowany obiekt do kontekstowej kolekcji Produktów
 - zapisz zmiany na kontekście
 - Zbuduj i uruchom aplikację



5.No to skonfigurujmy nasz kontekst, żeby wiedział do jakiej bazy chcemy się łączyć

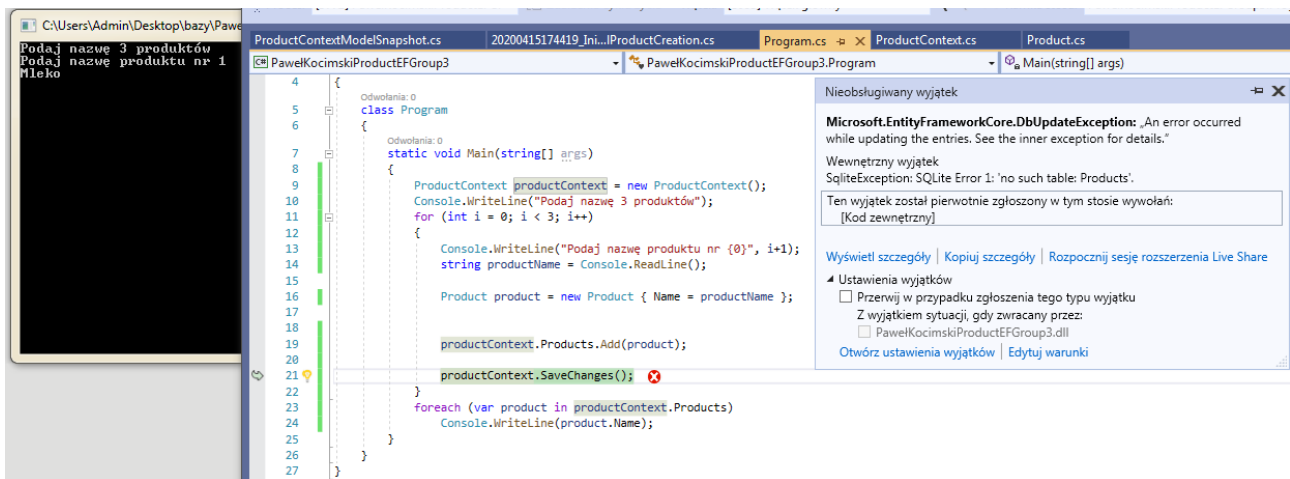
```

5
6 namespace PawełKocimskiProductEFGGroup3
7 {
8     class ProductContext : DbContext
9     {
10         protected override void OnConfiguring(DbContextOptionsBuilder options) => options.UseSqlite("DataSource=Product.db");
11         public DbSet<Product> Products { get; set; }
12     }
13 }
14
15
16

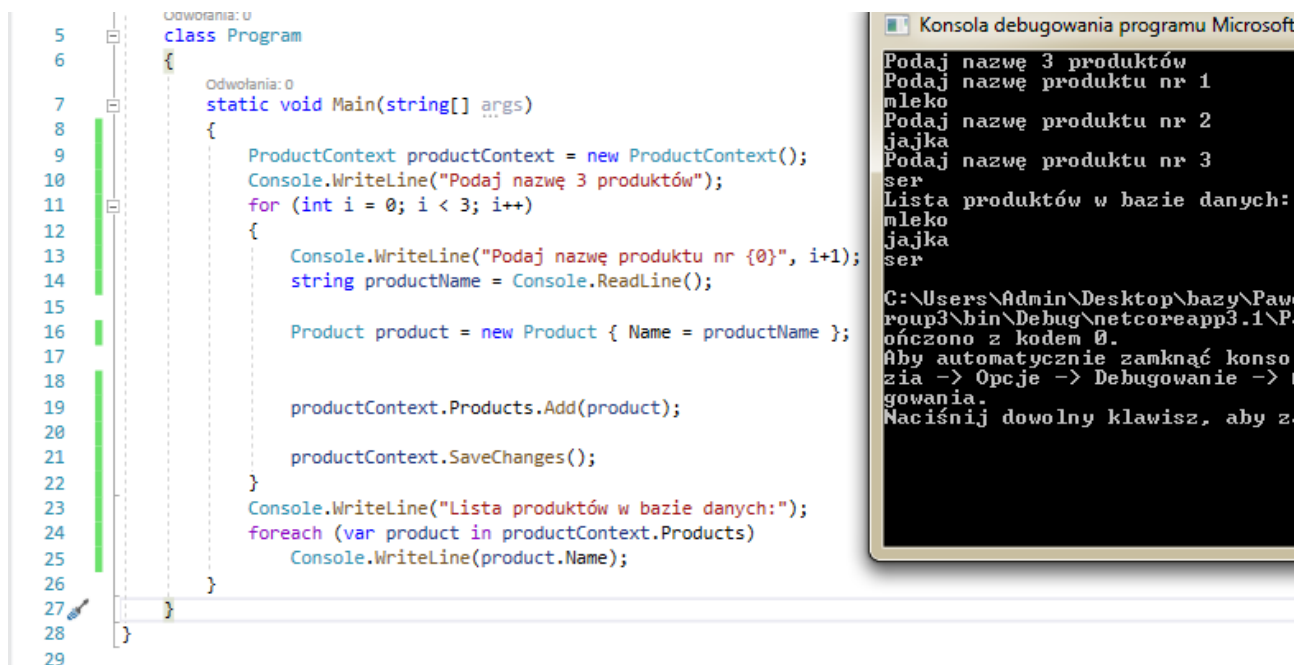
```



6.Dopisz w mainie fragment kodu pobierający oraz wyświetlający dostępne Produkty.

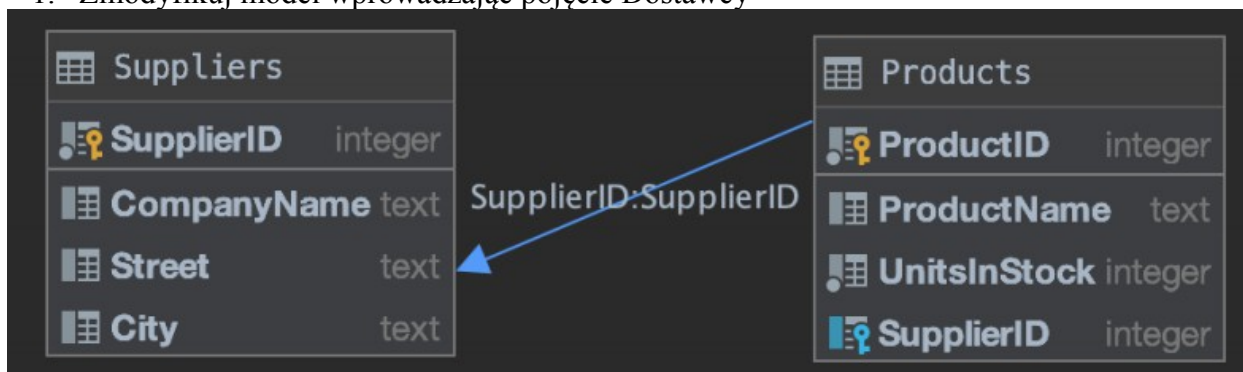


7. Po skopiowaniu pliku Product.db do miejsca gdzie są wrzucane pliki wykonywalne aplikacji w momencie jej uruchamiania.



Zadanie II

1. Zmodyfikuj model wprowadzając pojęcie Dostawcy



a) Zmodyfikowalna klasę Product, z wprowadzoną asocjacją z tabelą Suppliers

```

1  using System;
2  using System.Collections.Generic;
3  using System.Text;
4  using System.ComponentModel.DataAnnotations;
5  using System.ComponentModel.DataAnnotations.Schema;
6
7  namespace PawełKocimskiProductEFGroup3
8  {
9      Odwołania: 3
      class Product
10     {
11         Odwołania: 0
        public int ProductId { get; set; }
12         Odwołania: 2
        public string Name { get; set; }
13         Odwołania: 0
        public int UnitsInStock { get; set; }
14
15         [ForeignKey("Supplier")]
16         Odwołania: 0
        public int SupplierID { get; set; }
17         Odwołania: 0
        public Supplier supplier { get; set; }
18     }
19 }
20
21

```

b) Nowa tablica Supplier wraz z kluczem obcym odwołującym się do pola w tablicy Product

```

using System;
using System.Collections.Generic;
using System.Text;
using System.ComponentModel.DataAnnotations;

namespace PawełKocimskiProductEFGroup3
{
    Odwołania: 2
    class Supplier
    {
        [Key]
        Odwołania: 0
        public int SupplierId { get; set; }
        Odwołania: 0
        public string CompanyName { get; set; }
        Odwołania: 0
        public string Street { get; set; }
        Odwołania: 0
        public string City { get; set; }
    }
}

```

c) Nowy zbiór supplierów w kontekście

```

5 | using Microsoft.EntityFrameworkCore;
6 |
7 | namespace PawełKocińskiProductEFGGroup3
8 | {
9 |     class ProductContext : DbContext
10 |    {
11 |        protected override void OnConfiguring(DbContextOptionsBuilder options) => options.UseSqlite("DataSource=Product.db");
12 |        public DbSet<Product> Products{ get; set; }
13 |        public DbSet<Supplier> Suppliers { get; set; }
14 |    }
15 | }
16 |

```

```

C:\Users\Admin\Desktop\bazy\PawełKocińskiProductEFGGroup3\PawełKocińskiProductEFGGroup3>dotnet ef migrations add InitialSupplierCreation
Build started...
Build succeeded.
Done. To undo this action, use 'ef migrations remove'

C:\Users\Admin\Desktop\bazy\PawełKocińskiProductEFGGroup3\PawełKocińskiProductEFGGroup3>dotnet ef database update
Build started...
Build succeeded.
Applying migration '20200415195909_InitialSupplierCreation'.
Done.

```

d)przykładowe wywołanie

```

1 | using System;
2 | using System.Linq;
3 | namespace PawełKocińskiProductEFGGroup3
4 | {
5 |     class Program
6 |     {
7 |         static void Main(string[] args)
8 |         {
9 |             ProductContext productContext = new ProductContext();
10 |             Console.WriteLine("Podaj nazwę 3 produktów");
11 |             for (int i = 0; i < 3; i++)
12 |             {
13 |                 Console.WriteLine("Podaj nazwę produktu nr {0}", i+1);
14 |                 string productName = Console.ReadLine();
15 |                 Product product = new Product { Name = productName };
16 |                 Console.WriteLine("Podaj jego dostawcę", i + 1);
17 |                 string companyName = Console.ReadLine();
18 |                 Supplier supplier = new Supplier { CompanyName = companyName };
19 |                 productContext.Suppliers.Add(supplier);
20 |                 productContext.SaveChanges();
21 |                 var sup = (from s in productContext.Suppliers where s.CompanyName == companyName select s).First();
22 |                 product.SupplierID = sup.SupplierID;
23 |                 productContext.Products.Add(product);
24 |                 productContext.SaveChanges();
25 |             }
26 |             Console.WriteLine("Lista produktów w bazie danych:");
27 |             foreach (var product in productContext.Products)
28 |             {
29 |                 Console.WriteLine("ProductId={0}\t ProductName={1}\t Product.UnitsInStock={2}\t Product.SupplierID={3}",
30 |                     product.ProductId,product.Name, product.UnitsInStock,product.SupplierID);
31 |             }
32 |         }
33 |     }
34 | }
35 |

```

Konsola debugowania programu Microsoft Visual Studio

```

Podaj nazwę produktu nr 1
mleko
Podaj jego dostawcę
spółdzielnia
Podaj nazwę produktu nr 2
marchew
Podaj jego dostawcę
rolnik
Podaj nazwę produktu nr 3
telefon
Podaj jego dostawcę
eldzi
Lista produktów w bazie danych:
ProductId=1      ProductName=mleko      Product.UnitsInStock=0      Product.SupplierID=1
ProductId=2      ProductName=marchew    Product.UnitsInStock=0      Product.SupplierID=2
ProductId=3      ProductName=telefon    Product.UnitsInStock=0      Product.SupplierID=3

```

C:\Users\Admin\Desktop\bazy\PawełKocińskiProductEFGGroup3\PawełKocińskiProductEFGGroup3>dotnet ef database update

Aby automatycznie zamknąć konsolę po zatrzymaniu debugowania, włącz opcję Narzędzia -> Opcje -> Debugowanie -> Automatycznie zamknij konsolę po zatrzymaniu debugowania

e)Podgląd tabeli w bazie danych, która jest zgodna z wymaganiami zadania

```

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 NOT NULL,
  CONSTRAINT "FK_Products_Suppliers_SupplierId" FOREIGN KEY ("SupplierId") REFERENCES "Suppliers" ("SupplierId") ON DELETE CASCADE
);
CREATE INDEX "IX_Products_SupplierId" ON "Products" ("SupplierId");
sqlite>

```

2) Wyświetl wszystkie produkty wraz z nazwą dostawcy

```

1  using System;
2  using System.Linq;
3  namespace PawełKocimskiProductEFGGroup3
4  {
5      class Program
6      {
7          static void Main(string[] args)
8          {
9              ProductContext productContext = new ProductContext();
10
11              var data = (from prod in productContext.Products
12                          join sup in productContext.Suppliers
13                          on prod.SupplierID equals sup.SupplierId
14                          select new {product=prod.Name, supplier = sup.CompanyName})
15                  .ToList();
16
17              foreach(var row in data)
18              {
19                  Console.WriteLine(row.product);
20                  Console.WriteLine(row.CompanyName);
21                  Console.WriteLine("");
22              }
23          }
24      }
25  }
26  }
27  }
28

```

```

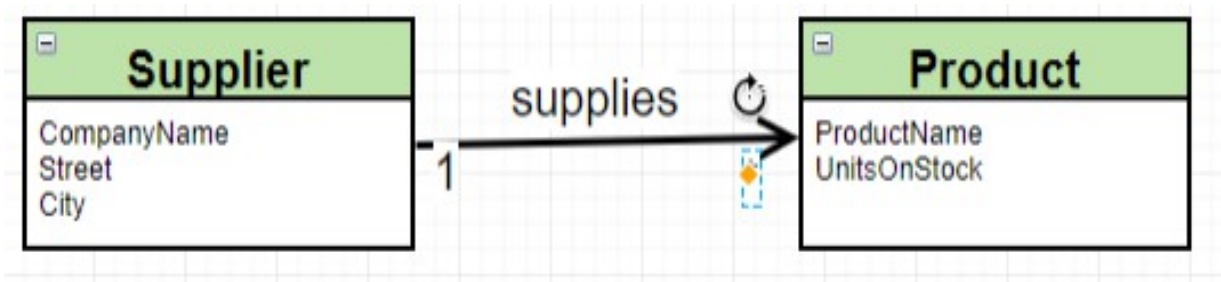
Konsola debugowania programu Microsoft Visual Studio

mleko
spoldzielnia
marchew
rolnik
telefon
eldzi

C:\Users\Admin\Desktop\bazy\PawełKocimskiProductEFGGroup3\PawełKocimskiProductEFGGroup3\bin\Debug\netcoreapp3.1\PawełKocimskiProductEFGGroup3.exe (proces 3732) zakończono z kodem 0.
Aby automatycznie zamknąć konsolę po zatrzymaniu debugowania, włącz opcję Narzędzia -> Opcje -> Debugowanie -> Automatycznie zamknij konsolę po zatrzymaniu debugowania

```

Zadanie III



1) W klasie jest teraz Supplier listę produktów dostarczanych przez dostawcę

```
1  using System;
2  using System.Collections.Generic;
3  using System.Text;
4  using System.ComponentModel.DataAnnotations;
5
6  namespace PawełKocimskiProductEFGroup3
7  {
8      class Supplier
9      {
10         public Supplier()
11         {
12             Products = new CollectionExtensions<Product>();
13         }
14
15         [Key]
16         public int SupplierId { get; set; }
17         public string CompanyName { get; set; }
18         public string Street { get; set; }
19         public string City { get; set; }
20         public ICollection<Product> Products { get; set; }
21     }
22 }
23
24
```

2) Pole Supplier z klasy Product zostało usunięte

```
1  using System;
2  using System.Collections.Generic;
3  using System.Text;
4  using System.ComponentModel.DataAnnotations;
5  using System.ComponentModel.DataAnnotations.Schema;
6
7  namespace PawełKocimskiProductEFGroup3
8  {
9      class Product
10     {
11         public int ProductId { get; set; }
12         public String Name { get; set; }
13         public int UnitsInStock { get; set; }
14     }
15 }
16
17
```


3)Przykładowe wywołanie

```
1  using System;
2  using System.Linq;
3  namespace PawełKocimskiProductEFGGroup3
4  {
5      class Program
6      {
7          static void Main(string[] args)
8          {
9              ProductContext productContext = new ProductContext();
10             string productName = Console.ReadLine();
11             Product product = new Product {Name = productName};
12             productContext.Products.Add(product);
13
14             Supplier supplier = productContext.Suppliers.FirstOrDefault(
15                 comp => comp.CompanyName = "spoldzielnia");
16
17             if(supplier == null)
18             {
19                 supplier = new Supplier {CompanyName = "spoldzielnia"};
20                 productContext.Suppliers.Add(supplier);
21             }
22             supplier.Products.Add(product);
23             productContext.SaveChanges();
24
25             var sups = productContext.Suppliers.Include(sup => sup.Products).ToList();
26             Console.WriteLine("Lista Dostawców: ");
27             foreach (var sup in sups)
28             {
29                 Console.WriteLine(supplier.CompanyName);
30                 foreach(var prod in sup.Products)
31                 {
32                     Console.WriteLine(prod.Name);
33                 }
34             }
35
36         }
37     }
38 }
39
```

```

Konsola debugowania programu Microsoft Visual Studio

mleko
spoldzielnia
sok jablkowy
marchew
ryż

C:\Users\Admin\Desktop\bazy\PawełKocinskiProductEFGroup3\PawełKocinskiPro
roup3\bin\Debug\netcoreapp3.1\PawełKocinskiProductEFGroup3.exe (proces 37
ończono z kodem 0.
Aby automatycznie zamknąć konsolę po zatrzymaniu debugowania, włącz opcję
zia -> Opcje -> Debugowanie -> Automatycznie zamknij konsolę po zatrzymaniu

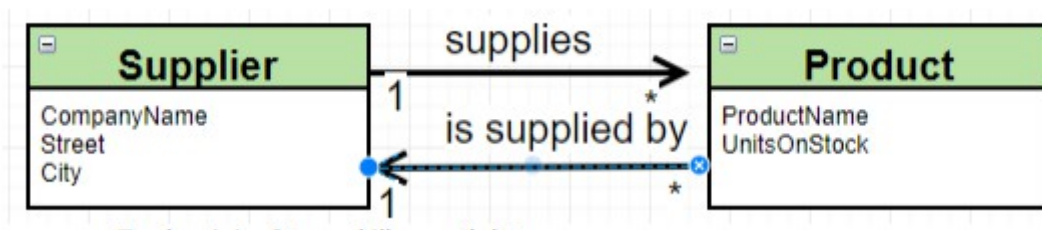
```

```

sqlite> select * from Products;
5|sok jablkowy|0|3
6|marchew|0|3
7|ryż|0|3
8|mleko|0|3
sqlite> select * from Suppliers;
3|spoldzielnia||
C:\Users\Admin\Desktop\bazy\PawełKocinskiProductEFGroup3\PawełKocinskiP
roup3\bin\Debug\netcoreapp3.1\PawełKocinskiProductEFGroup3.exe (proces
ończono z kodem 0.

```

Zadanie IV



1) W klasie Product zostało stworzone pole Supplier

```

1  using System;
2  using System.Collections.Generic;
3  using System.Text;
4  using System.ComponentModel.DataAnnotations;
5
6  namespace PawełKocinskiProductEFGroup3
7  {
8      class Supplier
9      {
10         public Supplier()
11         {
12             Products = new Collection<Product>();
13         }
14
15
16         [Key]
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 ICollection<Product> Products { get; set; }
22     }
23 }
24

```

```
using System;
using System.Collections.Generic;
using System.Text;
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace PawełKocimskiProductEFGroup3
{
    class Product
    {
        public int ProductId { get; set; }
        public String Name { get; set; }
        public int UnitsInStock { get; set; }
        public Supplier Supplier { get; set; }
    }
}
```

```

1  using System;
2  using System.Linq;
3  namespace PawełKocińskiProductEFGGroup3
4  {
5      class Program
6      {
7          static void Main(string[] args)
8          {
9              ProductContext productContext = new ProductContext();
10             string productName = Console.ReadLine();
11             Product product = new Product {Name = productName};
12             productContext.Products.Add(product);
13
14             Supplier supplier = productContext.Suppliers.FirstOrDefault(
15                 comp => comp.CompanyName = "spoldzielnia");
16
17             if(supplier == null)
18             {
19                 supplier = new Supplier {CompanyName = "spoldzielnia"};
20                 productContext.Suppliers.Add(supplier);
21             }
22             product.Supplier = supplier;
23             supplier.Products.Add(product);
24             productContext.SaveChanges();
25
26             var sups = productContext.Suppliers.Include(sup => sup.Products).ToList();
27             Console.WriteLine("Lista Dostawców: ");
28             foreach (var sup in sups)
29             {
30                 Console.WriteLine(supplier.CompanyName);
31                 foreach(var prod in sup.Products)
32                 {
33                     Console.WriteLine(prod.Name);
34                 }
35             }
36
37             var prods = productContext.Products.Include(prod =>prod.Supplier).ToList();
38             foreach(var pr in prods)
39             {
40                 Console.WriteLine(pr.Supplier.CompanyName);
41             }
42         }
43     }
44 }
45
46
47

```

Konsola debugowania programu Microsoft Visual Studio

```

mleko
Lista Dostawców
spoldzielnia
sok jablkowy
marchew
ryż
spoldzielnia
spoldzielnia
spoldzielnia
C:\Users\Admin\Desktop\bazy\PawełKocińskiProductEFGGroup3\PawełKocińskiPro
roup3\bin\Debug\netcoreapp3.1\PawełKocińskiProductEFGGroup3.exe (proces 37
ończono z kodem 0.
Aby automatycznie zamknąć konsolę po zatrzymaniu debugowania, włącz opcję
zia -> Opcje -> Debugowanie -> Automatycznie zamknij konsolę po zatrzymaniu

```

Zadanie V

1. Dodaj klasę Category z property int CategoryID, String Name oraz listą produktów

```
1  using System;
2  using System.Collections.Generic;
3  using System.Text;
4
5  namespace PawełKocimskiProductEFGroup3
6  {
7      class Category
8      {
9
10         public int CategoryId { get; set; }
11         public string Name { get; set; }
12         public List<Product> Products { get; set; }
13     }
14 }
15
```

2. Zmodyfikuj produkty dodając wskazanie na kategorie do której należy.

```
1  using System;
2  using System.Collections.Generic;
3  using System.Text;
4  using System.ComponentModel.DataAnnotations;
5  using System.ComponentModel.DataAnnotations.Schema;
6
7  namespace PawełKocimskiProductEFGroup3
8  {
9      class Product
10     {
11         public int ProductId { get; set; }
12         public string Name { get; set; }
13         public int UnitsInStock { get; set; }
14         public Supplier Supplier { get; set; }
15         public Category Category { get; set; }
16     }
17 }
18
```

3) Dodanie Category do contextu

```
1
2 using System;
3 using System.Collections.Generic;
4 using System.Text;
5 using Microsoft.EntityFrameworkCore;
6
7 namespace PawełKocińskiProductEFGGroup3
8 {
9     class ProductContext : DbContext
10     {
11         protected override void OnConfiguring(DbContextOptionsBuilder options) => options.UseSqlite("DataSource=Product.db");
12         public DbSet<Product> Products { get; set; }
13         public DbSet<Supplier> Suppliers { get; set; }
14         public DbSet<Category> Categories { get; set; }
15     }
16 }
17
18
19
```

4) Schemat z bazy danych

```
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,
  "CategoryId" INTEGER NULL,
  CONSTRAINT "FK_Products_Categories_CategoryId" FOREIGN KEY ("CategoryId") REFERENCES "Categories" ("CategoryId") ON DELETE RESTRICT,
  CONSTRAINT "FK_Products_Suppliers_SupplierId" FOREIGN KEY ("SupplierId") REFERENCES "Suppliers" ("SupplierId") ON DELETE RESTRICT
);
CREATE INDEX "IX_Products_CategoryId" ON "Products" ("CategoryId");
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> .schema Categories
CREATE TABLE IF NOT EXISTS "Categories" (
  "CategoryId" INTEGER NOT NULL CONSTRAINT "PK_Categories" PRIMARY KEY AUTOINCREMENT,
  "Name" TEXT NULL
);
sqlite>
```

5)Wydobądź produkty z wybranej kategorii oraz kategorię do której należy wybrany produkt

```
1 using Microsoft.EntityFrameworkCore;
2 using System;
3 using System.Linq;
4 namespace PawełKocimskiProductEFGGroup3
5 {
6     Odwrotnia: 0
7     class Program
8     {
9         Odwrotnia: 0
10        static void Main(string[] args)
11        {
12            ProductContext productContext = new ProductContext();
13
14            Category cat1 = new Category { Name = "nabial" };
15            Category cat2 = new Category { Name = "wedliny" };
16
17            productContext.Categories.Add(cat1);
18            productContext.Categories.Add(cat2);
19
20            Product product1 = new Product { Name = "mleko" };
21            product1.Category = cat1;
22
23            Product product2 = new Product { Name = "szynka" };
24            product2.Category = cat2;
25
26            Supplier supplier = productContext.Suppliers.FirstOrDefault(
27                comp => comp.CompanyName == "spoldzielnia");
28
29            if (supplier == null)
30            {
31                supplier = new Supplier { CompanyName = "spoldzielnia" };
32                productContext.Suppliers.Add(supplier);
33            }
34
35            product1.Supplier = supplier;
36            product2.Supplier = supplier;
37            supplier.Products.Add(product1);
38            supplier.Products.Add(product2);
39            productContext.SaveChanges();
40
41            var sups = productContext.Suppliers.Include(sup => sup.Products).ToList();
42            Console.WriteLine("lista dostawców: ");
43            foreach (var sup in sups)
44            {
45                Console.WriteLine(supplier.CompanyName);
46                foreach (var prod in sup.Products)
47                {
48                    Console.WriteLine(prod.Name);
49                }
50            }
51
52            var prods = productContext.Products.Include(prod => prod.Supplier).ToList();
53            foreach (var pr in prods)
54            {
55                Console.WriteLine(pr.Supplier.CompanyName);
56            }
57
58            var prodByCat = productContext.Categories.Include(cat => cat.Products).Where(cat => cat.Name == "nabial");
59            foreach(var cat in prodByCat)
60            {
61                foreach (var prod in cat.Products)
62                {
63                    Console.WriteLine(prod.Name);
64                }
65            }
66
67            var catByProd = productContext.Products.Where(prod => prod.Name == "twarog")
68                .Include(cat => cat.Category).FirstOrDefault();
69            Console.WriteLine(catByProd.Category.Name);
70        }
71    }
72 }
```

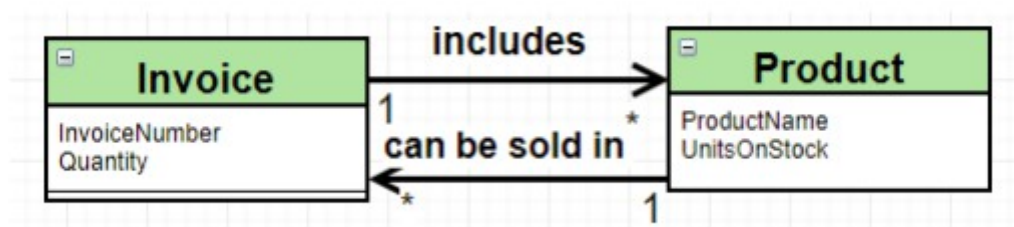
```
Konsola debugowania programu Microsoft Visual Studio

Lista Dostawców
spółdzielnia
szynka
mleko
mleko

C:\Users\Admin\Desktop\bazy\PawełKocińskiProductEFGroup3\PawełKocińskiPro
group3\bin\Debug\netcoreapp3.1\PawełKocińskiProductEFGroup3.exe (proces 37
ończono z kodem 0.
Aby automatycznie zamknąć konsolę po zatrzymaniu debugowania, włącz opcję
zia -> Opcje -> Debugowanie -> Automatycznie zamknij konsolę po zatrzyman
```

Zadanie VI

1. Zamodeluj relacje wiele-do-wielu, jak poniżej:



- a) klasa Supplier


```

1  using System;
2  using System.Collections.Generic;
3  using System.Text;
4  using System.ComponentModel.DataAnnotations;
5  using System.Collections.ObjectModel;
6
7  namespace PawełKocimskiProductEFGroup3
8  {
9      Odwołania: 5
10     class Supplier
11     {
12         1 odwołanie
13         public Supplier()
14         {
15             Products = new Collection<Product>();
16         }
17
18         Odwołania: 0
19         public int SupplierId { get; set; }
20         Odwołania: 4
21         public string CompanyName { get; set; }
22         Odwołania: 0
23         public string Street { get; set; }
24         Odwołania: 0
25         public string City { get; set; }
26         Odwołania: 5
27         public ICollection<Product> Products { get; set; }
28     }
29 }

```

b)klasa Category

```

1  using System;
2  using System.Collections.Generic;
3  using System.Collections.ObjectModel;
4  using System.Text;
5
6  namespace PawełKocimskiProductEFGroup3
7  {
8      Odwołania: 7
9      class Category
10     {
11         Odwołania: 2
12         public Category()
13         {
14             Products = new Collection<Product>();
15         }
16
17         Odwołania: 0
18         public int CategoryId { get; set; }
19         Odwołania: 4
20         public string Name { get; set; }
21         Odwołania: 3
22         public ICollection<Product> Products { get; set; }
23     }
24 }

```

c) Stworzona dodatkowa klasa InvoiceProduct w celu realizacji relacji między Product, a Invoice

```
1  using System;
2  using System.Collections.Generic;
3  using System.Text;
4
5  namespace PawełKocimskiProductEFGroup3
6  {
7      class InvoiceProduct
8      {
9          public int ProductId { get; set; }
10         public Product Product { get; set; }
11         public int InvoiceId { get; set; }
12         public Invoice Invoice { get; set; }
13     }
14 }
15
```

d) Nowa klasa Invoice, pole InvoiceProducts służy do realizacji relacji wiele do wielu

```

1  using System;
2  using System.Collections.Generic;
3  using System.Collections.ObjectModel;
4  using System.Text;
5
6  namespace PawełKocimskiProductEFGGroup3
7  {
8      class Invoice
9      {
10         public Invoice()
11         {
12             InvoiceProducts = new Collection<InvoiceProduct>();
13         }
14
15         public int InvoiceId { get; set; }
16         public int InvoiceNumber { get; set; }
17         public int Quantitiy { get; set; }
18
19         public ICollection<InvoiceProduct> InvoiceProducts { get; set; }
20     }
21 }
22

```

c) Zmodyfikowana klasa Product, pole InvoiceProducts służy do realizacji relacji wiele do wielu

```

1  using System;
2  using System.Collections.Generic;
3  using System.Text;
4  using System.ComponentModel.DataAnnotations;
5  using System.ComponentModel.DataAnnotations.Schema;
6  using System.Collections.ObjectModel;
7
8  namespace PawełKocimskiProductEFGGroup3
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         public Category Category { get; set; }
22         public ICollection<InvoiceProduct> InvoiceProducts { get; set; }
23     }
24 }
25

```

d) Dodane odpowiednie zbiory do kontekstu

```
1
2 using System;
3 using System.Collections.Generic;
4 using System.Text;
5 using Microsoft.EntityFrameworkCore;
6
7 namespace PawełKocimskiProductEFGroup3
8 {
9     Odwolania: 4
10    class ProductContext : DbContext
11    {
12        Odwolania: 0
13        protected override void OnConfiguring(DbContextOptionsBuilder options) => options.UseSqlite("DataSource=Product.db");
14        Odwolania: 2
15        public DbSet<Product> Products { get; set; }
16        Odwolania: 3
17        public DbSet<Supplier> Suppliers { get; set; }
18        Odwolania: 3
19        public DbSet<Category> Categories { get; set; }
20        Odwolania: 0
21        public DbSet<Invoice> Invoice { get; set; }
22        Odwolania: 0
23        public DbSet<InvoiceProduct> invoiceProducts { get; set; }
24
25        protected override void
26        Odwolania: 0
27        OnModelCreating(ModelBuilder modelBuilder)
28        {
29            modelBuilder.Entity<InvoiceProduct>()
30                .HasKey(a => new { a.ProductId, a.InvoiceId });
31        }
32    }
33 }
```

2) Stórz kilka produktów i “sprzedaj” je na kilku transakcjach

```

4 namespace PawełKocimskiProductEFGGroup3
5 {
6     class Program
7     {
8         static void Main(string[] args)
9         {
10             ProductContext productContext = new ProductContext();
11
12             Product product1 = new Product { Name = "smietana" };
13             Product product2 = new Product { Name = "jogurt" };
14             Product product3 = new Product { Name = "kefir" };
15             Product product4 = new Product { Name = "pieprz" };
16             Category category1 = new Category { Name = "przyprawy" };
17
18             category1.Products.Add(product1);
19             category1.Products.Add(product2);
20             category1.Products.Add(product3);
21             category1.Products.Add(product4);
22
23             Supplier supplier1 = new Supplier { CompanyName = "kamis" };
24
25             supplier1.Products.Add(product1);
26             supplier1.Products.Add(product2);
27             supplier1.Products.Add(product3);
28             supplier1.Products.Add(product4);
29
30             product1.Supplier = supplier1;
31             product2.Supplier = supplier1;
32             product3.Supplier = supplier1;
33             product4.Supplier = supplier1;
34
35             productContext.Categories.Add(category1);
36             productContext.Products.Add(product1);
37             productContext.Products.Add(product2);
38             productContext.Products.Add(product3);
39             productContext.Products.Add(product4);
40
41             Invoice invoice1 = new Invoice { InvoiceNumber = 1, Quantity = 5 };
42             Invoice invoice2 = new Invoice { InvoiceNumber = 2, Quantity = 8 };
43
44             InvoiceProduct invoiceProduct1 = new InvoiceProduct { Invoice = invoice1, Product = product1 };
45             InvoiceProduct invoiceProduct2 = new InvoiceProduct { Invoice = invoice1, Product = product2 };
46             InvoiceProduct invoiceProduct3 = new InvoiceProduct { Invoice = invoice2, Product = product3 };
47             InvoiceProduct invoiceProduct4 = new InvoiceProduct { Invoice = invoice2, Product = product4 };
48
49             invoice1.InvoiceProducts.Add(invoiceProduct1);
50             invoice1.InvoiceProducts.Add(invoiceProduct2);
51             invoice2.InvoiceProducts.Add(invoiceProduct3);
52             invoice2.InvoiceProducts.Add(invoiceProduct4);
53
54             productContext.invoiceProducts.Add(invoiceProduct1);
55             productContext.invoiceProducts.Add(invoiceProduct2);
56             productContext.invoiceProducts.Add(invoiceProduct3);
57             productContext.invoiceProducts.Add(invoiceProduct4);
58
59             product1.InvoiceProducts.Add(invoiceProduct1);
60             product1.InvoiceProducts.Add(invoiceProduct2);
61             product1.InvoiceProducts.Add(invoiceProduct3);
62             product1.InvoiceProducts.Add(invoiceProduct4);
63
64             productContext.SaveChanges();
65

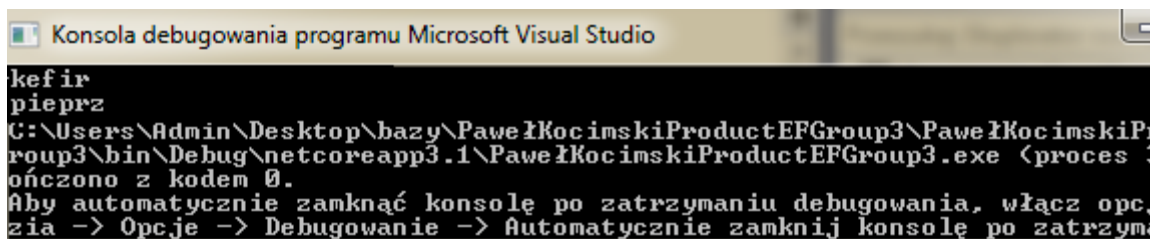
```

3) Pokaż produkty sprzedane w ramach wybranej faktury/transakcji

```

1  using Microsoft.EntityFrameworkCore;
2  using System;
3  using System.Linq;
4  namespace PawełKocinskiProductEFGGroup3
5  {
6      class Program
7      {
8          static void Main(string[] args)
9          {
10             ProductContext productContext = new ProductContext();
11             var products = productContext.invoiceProducts.Include(a => a.Product)
12                 .Where(a => a.InvoiceId == 2).Select(a => a.Product.Name).ToList();
13
14             foreach (var product in products)
15                 Console.WriteLine(product);
16
17
18
19
20
21
22
23         }
24     }
25 }

```



Konsola debugowania programu Microsoft Visual Studio

```

kefir
pieprz
C:\Users\Admin\Desktop\bazy\PawełKocinskiProductEFGGroup3\PawełKocinskiProductEFGGroup3\bin\Debug\netcoreapp3.1\PawełKocinskiProductEFGGroup3.exe <proces zakończono z kodem 0.
Aby automatycznie zamknąć konsolę po zatrzymaniu debugowania, włącz opcję: Opcje -> Debugowanie -> Automatycznie zamknij konsolę po zatrzymaniu

```

1. .Pokaż faktury w ramach których był sprzedany wybrany produkt

```

using Microsoft.EntityFrameworkCore;
using System;
using System.Linq;
namespace PawełKocimskiProductEFGGroup3
{
    class Program
    {
        static void Main(string[] args)
        {
            ProductContext productContext = new ProductContext();

            var invoices = productContext.invoiceProducts.Include(a => a.Invoice)
                .Where(prod => prod.ProductId == 1).Select(a => a.Invoice.InvoiceNumber);

            foreach (var invoice in invoices)
            {
                Console.WriteLine(invoice);
            }
        }
    }
}

```

Konsola debugowania programu Microsoft Visual Studio

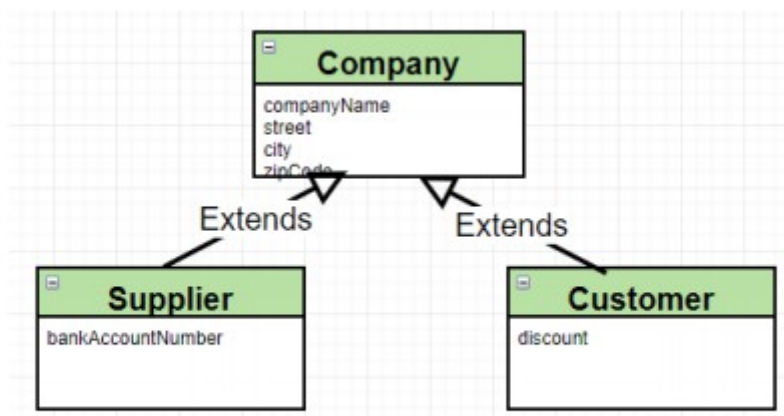
```

1
C:\Users\Admin\Desktop\bazy\PawełKocimskiProductEFGGroup3\PawełKocimskiProductEFGGroup3\bin\Debug\netcoreapp3.1\PawełKocimskiProductEFGGroup3.exe <proces
ończono z kodem 0.
Aby automatycznie zamknąć konsolę po zatrzymaniu debugowania, włącz opcję
zła -> Opcje -> Debugowanie -> Automatycznie zamknij konsolę po zatrzymaniu

```

Zadanie VII

1. Wprowadź do modelu następującą hierarchię:



a) Nowa klasa Company

```

5 namespace PawełKocimskiProductEFGroup3
6 {
7     1 odwołanie
8     class Company
9     {
10         Odwołania: 0
11         public int CompanyId { get; set; }
12         Odwołania: 0
13         public string CompanyName { get; set; }
14         Odwołania: 0
15         public string Street { get; set; }
16         Odwołania: 0
17         public string City { get; set; }
18         Odwołania: 0
19         public string ZipCode { get; set; }
20     }
21 }

```

b) Zmodyfikowana klasa dziedzicząca Supplier

```

1 using System;
2 using System.Collections.Generic;
3 using System.Text;
4 using System.ComponentModel.DataAnnotations;
5 using System.Collections.ObjectModel;
6
7 namespace PawełKocimskiProductEFGroup3
8 {
9     Odwołania: 3
10    class Supplier : Company
11    {
12        Odwołania: 0
13        public Supplier()
14        {
15            Products = new Collection<Product>();
16        }
17
18        Odwołania: 0
19        public int BankAccountNumber { get; set; }
20        1 odwołanie
21        public virtual ICollection<Product> Products { get; set; }
22    }
23 }

```

c) Nowa klasa dziedzicząca Customer


```

1  using System;
2  using System.Collections.Generic;
3  using System.Text;
4
5  namespace PawełKocimskiProductEFGroup3
6  {
7      class Customer : Company
8      {
9          public int Discount { get; set; }
10     }
11 }
12

```

2) Dodaj i pobierz z bazy kilka firm obu rodzajów stosując po kolei trzy różne strategie mapowania dziedziczenia

a) TablePerHierarchy

-zmiany w koncieście

```
2 using System;
3 using System.Collections.Generic;
4 using System.Text;
5 using Microsoft.EntityFrameworkCore;
6
7 namespace PawełKocimskiProductEFGroup3
8 {
9     Odwolania: 4
10    class ProductContext : DbContext
11    {
12        Odwolania: 0
13        protected override void OnConfiguring(DbContextOptionsBuilder options) => options
14            .UseSqlite("DataSource=Product.db");
15        Odwolania: 0
16        public DbSet<Product> Products { get; set; }
17
18        Odwolania: 0
19        public DbSet<Category> Categories { get; set; }
20        Odwolania: 0
21        public DbSet<Invoice> Invoice { get; set; }
22        Odwolania: 0
23        public DbSet<InvoiceProduct> invoiceProducts { get; set; }
24
25        Odwolania: 4
26        public DbSet<Company> Companies { get; set; }
27
28        protected override void
29            Odwolania: 0
30            OnModelCreating(ModelBuilder modelBuilder)
31        {
32            modelBuilder.Entity<InvoiceProduct>()
33                .HasKey(a => new { a.ProductId, a.InvoiceId });
34            modelBuilder.Entity<Customer>();
35            modelBuilder.Entity<Supplier>();
36        }
37    }
38 }
```

-przykładowe dane

```
1 using Microsoft.EntityFrameworkCore;
2 using System;
3 using System.Linq;
4 namespace PawełKocińskiProductEFGGroup3
5 {
6     class Program
7     {
8         static void Main(string[] args)
9         {
10             ProductContext productContext = new ProductContext();
11             Customer customer1 = new Customer
12             {
13                 City = "Warszawa",
14                 CompanyName = "Mlepol",
15                 Street = "Polna",
16                 ZipCode = "11-111",
17                 Discount = 10
18             };
19
20             Supplier supplier1 = new Supplier
21             {
22                 City = "Kraków",
23                 CompanyName = "Mlekowita",
24                 Street = "Topolowa",
25                 ZipCode = "22-222",
26                 BankAccountNumber = 987654321
27             };
28
29             Customer customer2 = new Customer
30             {
31                 City = "Poznań",
32                 CompanyName = "Gouda",
33                 Street = "serowa",
34                 ZipCode = "33-333",
35                 Discount = 20
36             };
37
38             Supplier supplier2 = new Supplier
39             {
40                 City = "Łódź",
41                 CompanyName = "Cambert",
42                 Street = "kasztanowa",
43                 ZipCode = "44-444",
44                 BankAccountNumber = 123456789
45             };
46             productContext.Companies.Add(customer1);
47             productContext.Companies.Add(customer2);
48             productContext.Companies.Add(supplier1);
49             productContext.Companies.Add(supplier2);
50         }
51     }
52 }
```

```

sqlte> select * from Companies;
1|Mlekpól|Pólna|Warszawa|11-111|Customer|10
2|Mlekwita|Topolowa|Kraków|22-222|Supplier|987654321
3|Gouda|serowa|Poznań|Customer|133-333|120
3|Cambert|kasztanowa|Łódź|Supplier|44-4444|123456789

```

b) TablePerType

W EntityFrameWork 3.0 zostały prowadzone zmiany w mapowaniu, co powoduje inne wyniki niż wcześniej otrzymano

-zmieniona klasa Customer

```

1  using System;
2  using System.Collections.Generic;
3  using System.ComponentModel.DataAnnotations.Schema;
4  using System.Text;
5
6  namespace PawełKocimskiProductEFGroup3
7  {
8      [Table("Customers")]
9      class Customer : Company
10     {
11         public int Discount { get; set; }
12     }
13 }
14

```

-zmieniona klasa Supplier

```

1  using System;
2  using System.Collections.Generic;
3  using System.Text;
4  using System.ComponentModel.DataAnnotations;
5  using System.Collections.ObjectModel;
6  using System.ComponentModel.DataAnnotations.Schema;
7
8  namespace PawełKocimskiProductEFGroup3
9  {
10     [Table("Suppliers")]
11     class Supplier : Company
12     {
13         public Supplier()
14         {
15             Products = new Collection<Product>();
16         }
17
18         public int BankAccountNumber { get; set; }
19         public virtual ICollection<Product> Products { get; set; }
20     }
21 }
22

```

-zmieniony kontekst

```
1
2  using System;
3  using System.Collections.Generic;
4  using System.Text;
5  using Microsoft.EntityFrameworkCore;
6
7  namespace PawełKocimskiProductEFGrou3
8  {
9      class ProductContext : DbContext
10     {
11         protected override void OnConfiguring(DbContextOptionsBuilder options) => options
12             .UseSqlite("DataSource=Product.db");
13         public DbSet<Product> Products { get; set; }
14
15         public DbSet<Category> Categories { get; set; }
16         public DbSet<Invoice> Invoice { get; set; }
17         public DbSet<InvoiceProduct> invoiceProducts { get; set; }
18
19         public DbSet<Company> Companies { get; set; }
20
21         protected override void
22             OnModelCreating(ModelBuilder modelBuilder)
23         {
24             modelBuilder.Entity<InvoiceProduct>()
25                 .HasKey(a => new { a.ProductId, a.InvoiceId });
26         }
27     }
28 }
29
30
31
32
33
```

-przykładowe wywołanie

```

1  using Microsoft.EntityFrameworkCore;
2  using System;
3  using System.Linq;
4  namespace PawełKocińskiProductEFGGroup3
5  {
6      class Program
7      {
8          static void Main(string[] args)
9          {
10             ProductContext productContext = new ProductContext();
11             Customer customer1 = new Customer
12             {
13                 City = "Warszawa",
14                 CompanyName = "Mlekpól",
15                 Street = "Polna",
16                 ZipCode = "11-111",
17                 Discount = 10
18             };
19
20             Supplier supplier1 = new Supplier
21             {
22                 City = "Kraków",
23                 CompanyName = "Mlekowita",
24                 Street = "Topolowa",
25                 ZipCode = "22-222",
26                 BankAccountNumber = 987654321
27             };
28
29             Customer customer2 = new Customer
30             {
31                 City = "Poznań",
32                 CompanyName = "Gouda",
33                 Street = "Serowa",
34                 ZipCode = "33-333",
35                 Discount = 20
36             };
37
38             Supplier supplier2 = new Supplier
39             {
40                 City = "Łódź",
41                 CompanyName = "Cambert",
42                 Street = "Kasztanowa",
43                 ZipCode = "44-444",
44                 BankAccountNumber = 123456789
45             };
46             productContext.Companies.Add(customer1);
47             productContext.Companies.Add(customer2);
48             productContext.Companies.Add(supplier1);
49             productContext.Companies.Add(supplier2);
50         }
51     }
52 }

```

```

sqlte> select * from Companies;
1|Mlekpól|Polna|Warszawa|11-111|Company|10
2|Mlekowita|Topolowa|Kraków|22-222|Supplier|987654321
3|Gouda|Serowa|Poznań|33-333|120
3|Cambert|Kasztanowa|Łódź|Supplier|44-444|123456789

```

Rozbieżność polega na typie przyjmowanym przez Customer. Jest nim typ Company z klasy nadrzędnej.

c)TablePerClass

EntityFramework3.0 uniemożliwił wykonanie tego zadania.