Bazy danych – Entity Framework – zadanie domowe

Białka Dawid

Grupa czwartek B, godz.14;40

I.

Product.cs

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

namespace IBiałkaProdutcEF
{
	class Product
	{
		public int ProductId { get; set; }
		public string ProductName { get; set; }
		public int UnitsInStock { get; set; }
	}
}
```

Program.cs

```
using System;
using System.Linq;


namespace IBiałkaProdutcEF
{
	class Program
	{
		static void Main(string[] args)
		{
			System.Console.WriteLine("Podaj nazwe produktu: ");
			String prodName = System.Console.ReadLine();
			Product prod = new Product { ProductName = prodName };

			ProdContext context = new ProdContext();
			context.Products.Add(prod);
			context.SaveChanges();

			var Products = context.Products;
			System.Console.WriteLine("Produkty w bazie : ");
			foreach (Product p in Products)
				System.Console.WriteLine("- " + p.ProductName);

		}
	}
}
```

ProductContext.cs

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


namespace IBiałkaProdutcEF
{
	class ProdContext : DbContext
	{
		protected override void OnConfiguring(DbContextOptionsBuilder options)
			=> options.UseSqlite("DataSource=Product.db");
		public DbSet<Product> Products { get; set; }
	}
}
```

Wyniki z pierwszego podpunktu są w raporcie z laboratorium.

II.

Product.cs

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

namespace IBiałkaProdutcEF
{

	class Product
	{
		public int ProductId { get; set; }
		public string ProductName { get; set; }
		public int UnitsInStock { get; set; }

		public int? SupplierId { get; set; }
		[ForeignKey("SupplierId")]
		public Supplier Supplier { get; set; }

	}
}
```

Supplier.cs

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

namespace IBiałkaProdutcEF
{
    class Supplier
    {
        public int SupplierId { get; set; }
        public string CompanyName { get; set; }
        public string Street { get; set; }
        public string City { get; set; }
    }
}
```

DataBaseContext.cs

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


namespace IBiałkaProdutcEF
{
	class DataBaseContext : DbContext
	{
		protected override void OnConfiguring(DbContextOptionsBuilder options)
			=> options.UseSqlite("DataSource=Entity.db");


		public DbSet<Product> Products { get; set; }
		public DbSet<Supplier> Suppliers { get; set; }
	}
}
```

Program.cs

```
using System;
using System.Linq;
using Microsoft.EntityFrameworkCore;

namespace IBiałkaProdutcEF
{
	class Program
	{

		static void Main(string[] args)
		{
			DataBaseContext context = new DataBaseContext();

			System.Console.WriteLine("Podaj nazwe produktu: ");
			String prodName = System.Console.ReadLine();
			Product prod = new Product
			{
				ProductName = prodName,
				SupplierId = null
			};

			context.Products.Add(prod);

			System.Console.WriteLine("Podaj nazwe dostawcy: ");
			String suppName = System.Console.ReadLine();
			Supplier supp = new Supplier
			{
				CompanyName = suppName,
			};

			context.Suppliers.Add(supp);
			context.SaveChanges();

			var Products = context.Products;
			System.Console.WriteLine("Produkty w bazie : ");
			foreach (Product p in Products)
				System.Console.WriteLine("- " + p.ProductName + " " + p.ProductId);

			var Suppliers = context.Suppliers;
			System.Console.WriteLine("Dostawcy w bazie : ");
			foreach (Supplier s in Suppliers)
				System.Console.WriteLine("- " + s.CompanyName + " " + s.SupplierId);
```

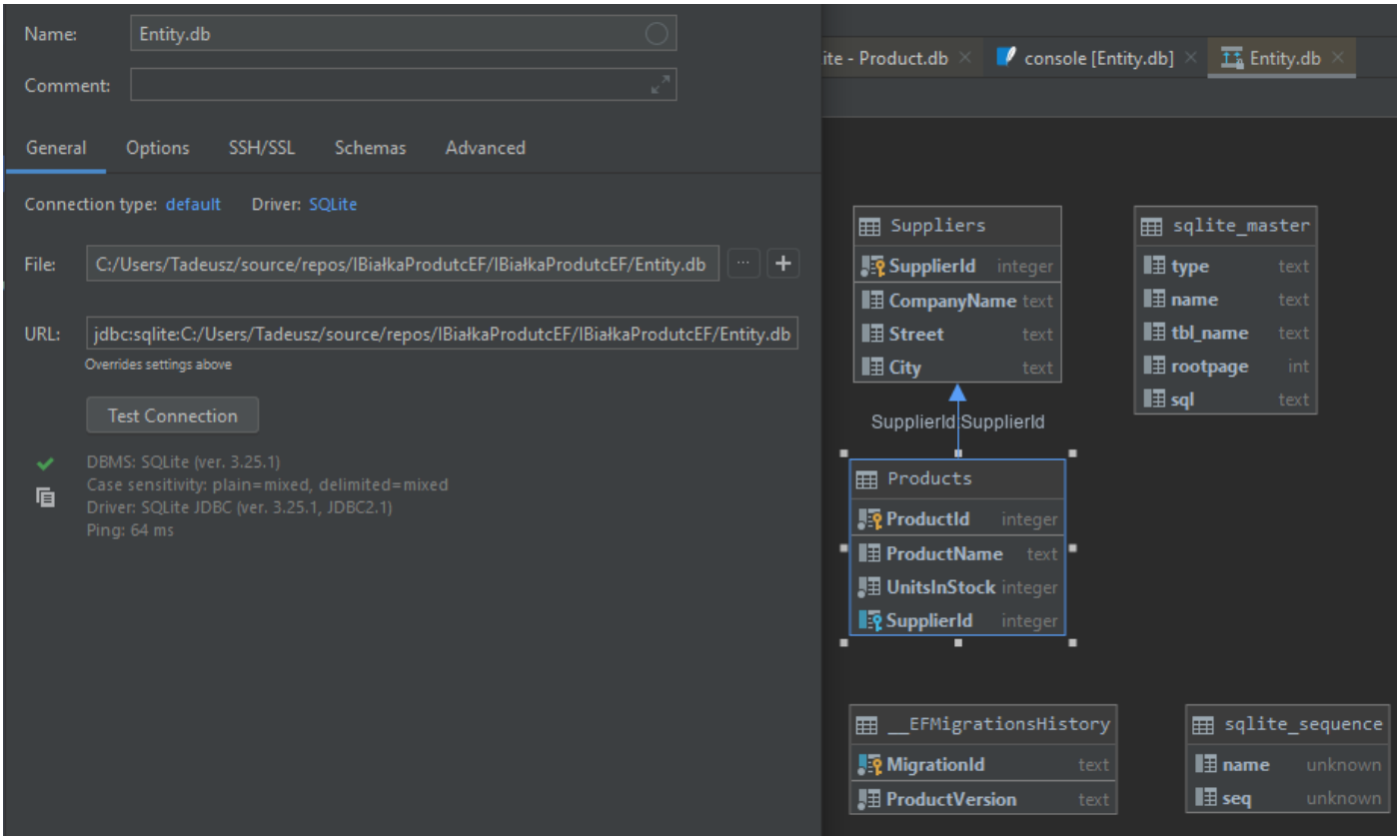
```
                var Item1 = context.Products.Where(a => a.ProductName == "Baklazan").Single();
                var Item2 = context.Products.Where(a => a.ProductName == "Marchewka").Single();
                var NorthWind = context.Suppliers.Where(a => a.CompanyName ==
                                                        "Northwind").Single();


                Item1.Supplier = NorthWind;
                Item1.SupplierId = NorthWind.SupplierId;

                Item2.Supplier = NorthWind;
                Item2.SupplierId = NorthWind.SupplierId;

                var query = context.Products.Include(a => a.Supplier).ToList();

                System.Console.WriteLine("Join:");
                foreach (var q in query)
                {
                        System.Console.WriteLine(q.ProductName + "-" + q.Supplier.CompanyName);
                }

            }
        }
}
```

Konsola debugowania programu Microsoft Visual Studio

```
Podaj nazwe produktu:
Baklazan
Podaj nazwe dostawcy:
Northwind
Produkty w bazie :
- Marchewka 1
- Baklazan 2
Dostawcy w bazie :
- Northwind 1

C:\Users\Tadeusz\source\repos\IBiałkaProdutcEF\IBiałkaProdutcEF\bin\Debug\netcoreapp
24) zakończono z kodem 0.
Aby automatycznie zamknąć konsolę po zatrzymaniu debugowania, włącz opcję Narzędzia
znie zamknij konsolę po zatrzymaniu debugowania.
Naciśnij dowolny klawisz, aby zamknąć to okno...
```

Konsola debugowania programu Microsoft Visual Studio

```
Produkty w bazie :
- Marchewka 1
- Baklazan 2
Dostawcy w bazie :
- Northwind 1
Join:
Marchewka-Northwind
Baklazan-Northwind

C:\Users\Tadeusz\source\repos\IBiałkaProdutcEF\IBiał
64) zakończono z kodem 0.
Aby automatycznie zamknąć konsolę po zatrzymaniu deb
znie zamknij konsolę po zatrzymaniu debugowania.
Naciśnij dowolny klawisz, aby zamknąć to okno...
```



III.

Product.cs

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


namespace IBiałkaProdutcEF
{

        class Product
        {
                public int ProductId { get; set; }
                public string ProductName { get; set; }
                public int UnitsInStock { get; set; }
        }
}
```

Supplier.cs

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

namespace IBiałkaProdutcEF
{
    class Supplier
    {
```

```
        public Supplier()
        {
            this.ProductSet = new List<Product>();
        }
        public int SupplierId { get; set; }
        public string CompanyName { get; set; }
        public string Street { get; set; }
        public string City { get; set; }

        public ICollection<Product> ProductSet { get; set; }
    }
}
```

DataBaseContext.cs bez zmian

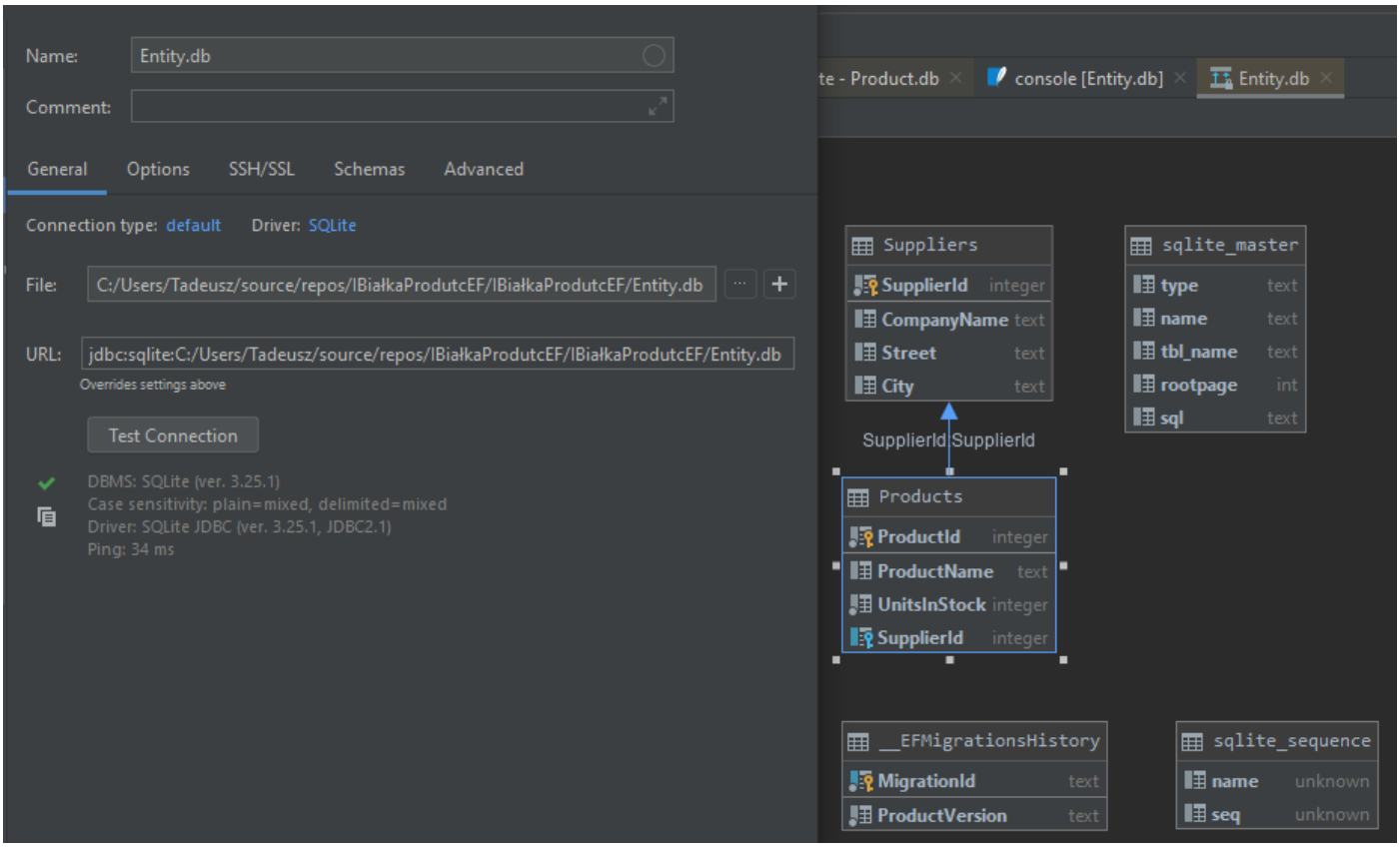
Program.cs

```
using System;
using System.Linq;
using Microsoft.EntityFrameworkCore;

namespace IBiałkaProdutcEF
{
       class Program
       {

              static void Main(string[] args)
              {
                     DataBaseContext context = new DataBaseContext();

                     System.Console.WriteLine("Podaj nazwe produktu: ");
                     String prodName = System.Console.ReadLine();
                     Product prod = new Product
                     {
                            ProductName = prodName,
                     };

                     context.Products.Add(prod);

                     System.Console.WriteLine("Podaj nazwe dostawcy: ");
                     String suppName = System.Console.ReadLine();
                     Supplier supp = new Supplier
                     {
                            CompanyName = suppName
                     };

                     context.Suppliers.Add(supp);
                     context.SaveChanges();

                     var Products = context.Products;
                     System.Console.WriteLine("Produkty w bazie : ");
                     foreach (Product p in Products)
                            System.Console.WriteLine("- " + p.ProductName + " " + p.ProductId);

                     var Suppliers = context.Suppliers;
                     System.Console.WriteLine("Dostawcy w bazie : ");
                     foreach (Supplier s in Suppliers)
                            System.Console.WriteLine("- " + s.CompanyName + " " + s.SupplierId);

                     var Supplier = context.Suppliers.Where(a => a.CompanyName ==
                                                   "Northwind").Single();

                     foreach (Product p in Products)
                            Supplier.ProductSet.Add(p);

                     System.Console.WriteLine("Join:");
                     foreach (var s in Suppliers)
                     {
```

```
                        foreach(var p in s.ProductSet)
                        {
                                System.Console.WriteLine(s.CompanyName + " supplies " +
                                                                                p.ProductName);
                        }
                }

            }
        }
}
```

Konsola debugowania programu Microsoft Visual Studio

```
Produkty w bazie :
- Marchewka 1
- Baklazan 2
- Kalarepa 3
- Cebula 4
Dostawcy w bazie :
- Northwind 1
Join:
Northwind supplies Marchewka
Northwind supplies Baklazan
Northwind supplies Kalarepa
Northwind supplies Cebula

C:\Users\Tadeusz\source\repos\IBiałkaProdutcEF\IB
72) zakończono z kodem 0.
Aby automatycznie zamknąć konsolę po zatrzymaniu
znie zamknij konsolę po zatrzymaniu debugowania.
Naciśnij dowolny klawisz, aby zamknąć to okno...
```



IV

Product.cs

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


namespace IBiałkaProdutcEF
{

	class Product
	{
		public int ProductId { get; set; }
		public string ProductName { get; set; }
		public int UnitsInStock { get; set; }

		public int? SupplierId { get; set; }
		[ForeignKey("SupplierId")]
		public Supplier Supplier { get; set; }

	}
}
```

Supplier.cs

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

namespace IBiałkaProdutcEF
{
    class Supplier
    {
        public Supplier()
        {
            this.ProductSet = new List<Product>();
        }
        public int SupplierId { get; set; }
        public string CompanyName { get; set; }
        public string Street { get; set; }
        public string City { get; set; }

        public ICollection<Product> ProductSet { get; set; }
    }
}
```

DataBaseContext.cs bez zmian

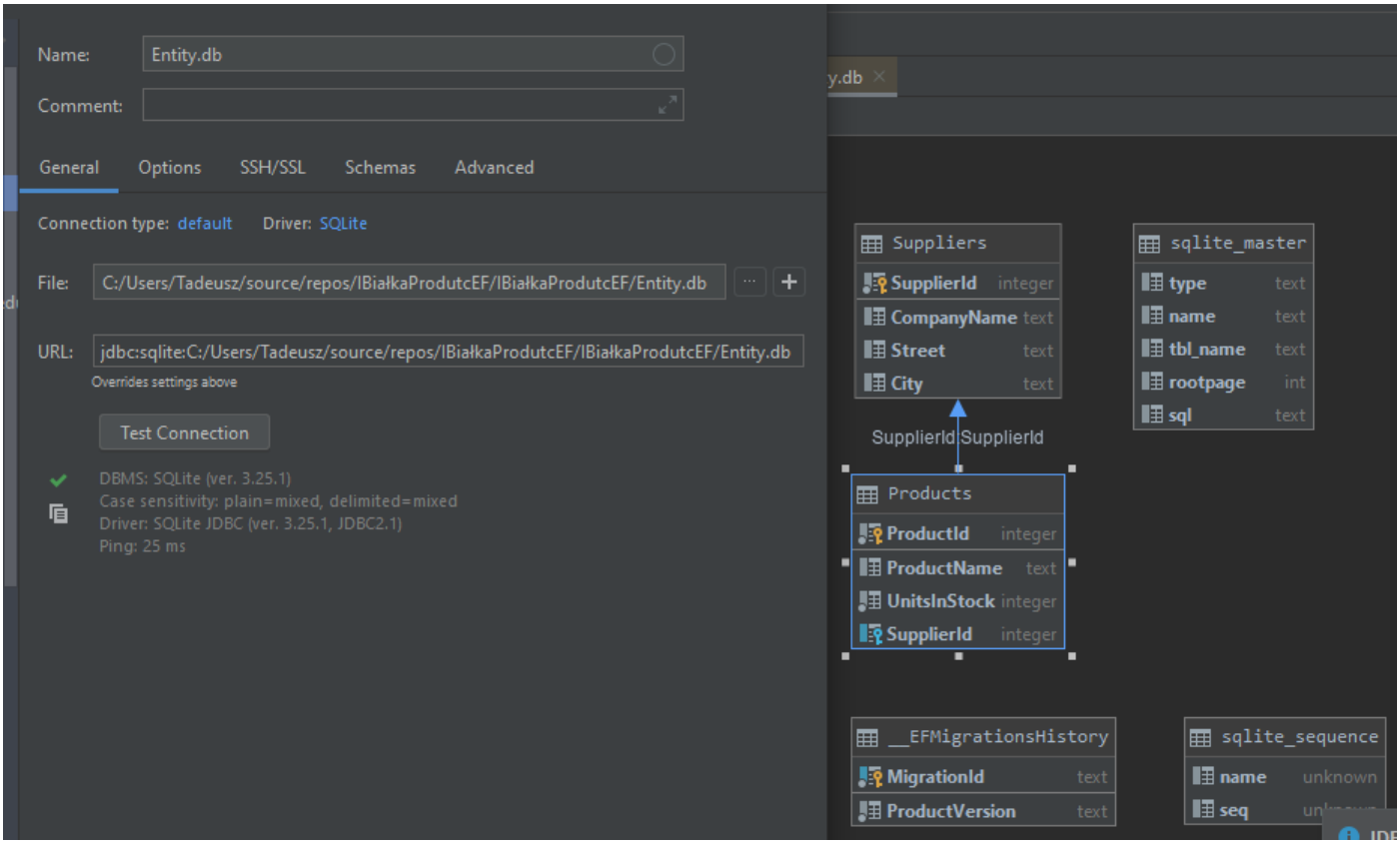
Program.cs

```
using System;
using System.Linq;
using Microsoft.EntityFrameworkCore;

namespace IBiałkaProdutcEF
{
	class Program
	{

		static void Main(string[] args)
		{
			DataBaseContext context = new DataBaseContext();

			System.Console.WriteLine("Podaj nazwe produktu: ");
			String prodName = System.Console.ReadLine();
			Product prod = new Product
			{
				ProductName = prodName,
			};

			context.Products.Add(prod);

			System.Console.WriteLine("Podaj nazwe dostawcy: ");
			String suppName = System.Console.ReadLine();
			Supplier supp = new Supplier
			{
				CompanyName = suppName
			};

			context.Suppliers.Add(supp);
			context.SaveChanges();

			var Products = context.Products;
			System.Console.WriteLine("Produkty w bazie : ");
			foreach (Product p in Products)
				System.Console.WriteLine("- " + p.ProductName + " " + p.ProductId);

			var Suppliers = context.Suppliers;
			System.Console.WriteLine("Dostawcy w bazie : ");
			foreach (Supplier s in Suppliers)
				System.Console.WriteLine("- " + s.CompanyName + " " + s.SupplierId);

			var supp_found = context.Suppliers.Where(a => a.CompanyName ==
                                                 "Northwind").Single();

			foreach (Product p in Products)
			{
				supp_found.ProductSet.Add(p);
				p.Supplier = supp_found;
				p.SupplierId = p.Supplier.SupplierId;
			}



			System.Console.WriteLine("Join, from supplier to product:");
			foreach (var s in Suppliers)
			{
				foreach(var p in s.ProductSet)
				{
					System.Console.WriteLine(s.CompanyName + " supplies " +
                                                 p.ProductName);
				}
			}

			System.Console.WriteLine("Join, from product to supplier:");
```

```
                var query = context.Products.Include(a => a.Supplier).ToList();
                foreach (var q in query)
                {
                        System.Console.WriteLine(q.ProductName + " is supplied by " +
                                                 q.Supplier.CompanyName);
                }
            }
        }
}
```

Konsola debugowania programu Microsoft Visual Studio

```
Podaj nazwe produktu:
Kalarepa
Podaj nazwe dostawcy:
Northwind
Produkty w bazie :
- Marchewka 1
- Baklazan 2
- Cebula 3
- Kalarepa 4
Dostawcy w bazie :
- Northwind 1
Join, from supplier to product:
Northwind supplies Marchewka
Northwind supplies Baklazan
Northwind supplies Cebula
Northwind supplies Kalarepa
Join, from product to supplier:
Marchewka is supplied by Northwind
Baklazan is supplied by Northwind
Cebula is supplied by Northwind
Kalarepa is supplied by Northwind

C:\Users\Tadeusz\source\repos\IBiałkaProdutcEF\IB
08) zakończono z kodem 0.
Aby automatycznie zamknąć konsolę po zatrzymaniu
znie zamknij konsolę po zatrzymaniu debugowania.
Naciśnij dowolny klawisz, aby zamknąć to okno...
```



V.

Product.cs

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


namespace IBiałkaProdutcEF
{

	class Product
	{
		public int ProductId { get; set; }
		public string ProductName { get; set; }
		public int UnitsInStock { get; set; }

		public int? SupplierId { get; set; }
		[ForeignKey("SupplierId")]
		public Supplier Supplier { get; set; }

		public int? CategoryId { get; set; }
		[ForeignKey("CategoryId")]
		public Category Category { get; set; }

	}
}
```

Supplier.cs bez zmian

Category.cs

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

namespace IBiałkaProdutcEF
{
    class Category
    {
        public Category()
        {
            this.ProductSet = new List<Product>();
        }
        public int CategoryId { get; set; }
        public string Name { get; set; }

        public ICollection<Product> ProductSet { get; set; }
    }
}
```

DataBaseContext.cs

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


namespace IBiałkaProdutcEF
{
	class DataBaseContext : DbContext
	{
		protected override void OnConfiguring(DbContextOptionsBuilder options)
			=> options.UseSqlite("DataSource=Entity.db");


		public DbSet<Product> Products { get; set; }
		public DbSet<Supplier> Suppliers { get; set; }
		public DbSet<Category> Categories { get; set; }
	}
}
```

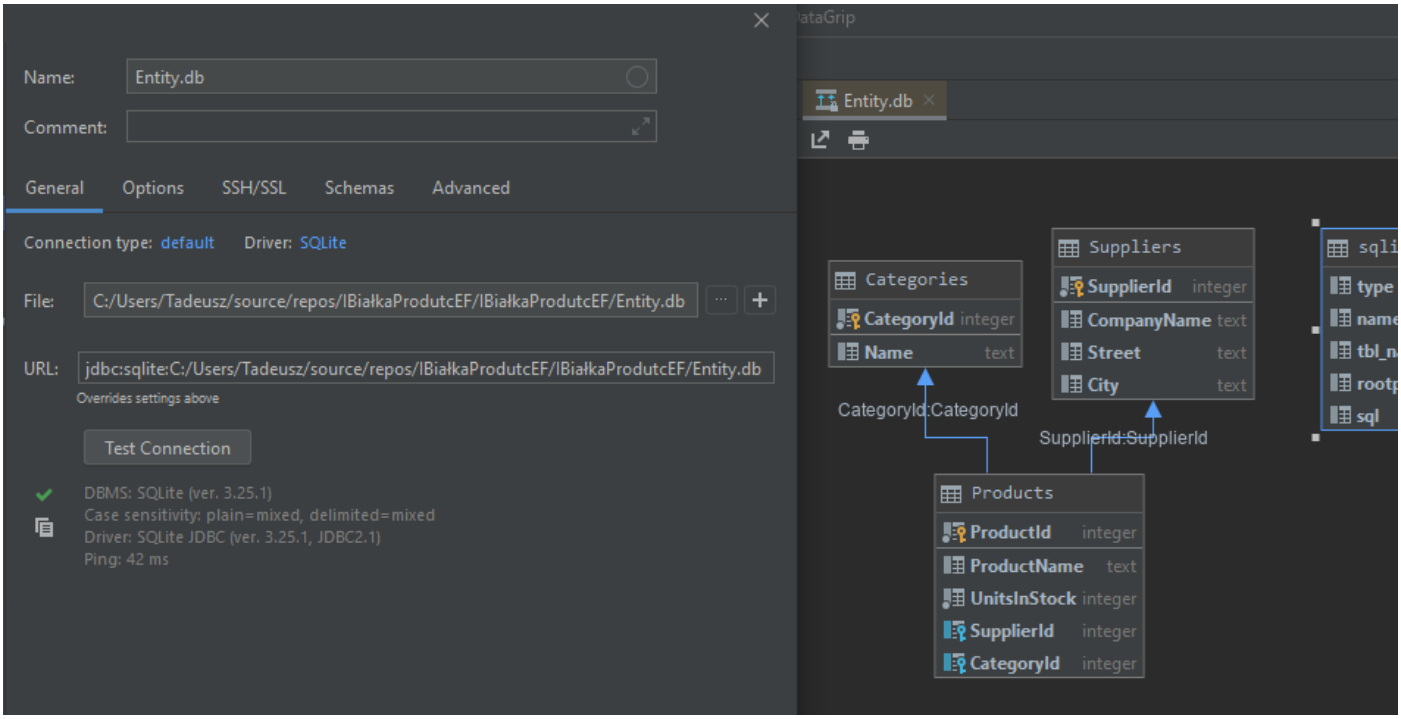
Program.cs

```
using System;
using System.Linq;
using Microsoft.EntityFrameworkCore;

namespace IBiałkaProdutcEF
{
	class Program
	{

		static void Main(string[] args)
		{
			DataBaseContext context = new DataBaseContext();

			System.Console.WriteLine("Podaj nazwe produktu: ");
			String prodName = System.Console.ReadLine();
			Product prod = new Product
			{
				ProductName = prodName,
			};
```

```
                context.Products.Add(prod);
                context.SaveChanges();

                System.Console.WriteLine("Podaj nazwe kategorii: ");
                String catName = System.Console.ReadLine();
                Category cat = new Category
                {
                        Name = catName
                };

                context.Categories.Add(cat);
                context.SaveChanges();

                var Products = context.Products;
                System.Console.WriteLine("Produkty w bazie : ");
                foreach (Product p in Products)
                        System.Console.WriteLine("- " + p.ProductName + " " + p.ProductId);


                var Categories = context.Categories;
                System.Console.WriteLine("Kategorie w bazie : ");
                foreach (Category c in Categories)
                        System.Console.WriteLine("- " + c.Name + " " + c.CategoryId);

                var cat_found = context.Categories.Where(a => a.Name == "Warzywa").Single();

                foreach (Product p in Products)
                {
                        cat_found.ProductSet.Add(p);
                        p.Category = cat_found;
                        p.CategoryId = p.Category.CategoryId;
                }

                System.Console.WriteLine("Join, from category to product:");
                foreach (var c in Categories)
                {
                        foreach(var p in c.ProductSet)
                        {
                                System.Console.WriteLine(c.Name + " has " + p.ProductName);
                        }
                }

                System.Console.WriteLine("Join, from product to category:");

                var query = context.Products.Include(a => a.Category).ToList();
                foreach (var q in query)
                {
                        System.Console.WriteLine(q.ProductName + " is in category: " +
                                                q.Category.Name);
                }
            }
        }
}
```

Konsola debugowania programu Microsoft Visual Studio

```
Podaj nazwe produktu:
Kalarepa
Podaj nazwe kategorii:
Warzywa
Produkty w bazie :
- Marchewka 1
- Baklazan 2
- Cebula 3
- Kalarepa 4
Kategorie w bazie :
- Owoce 1
- Warzywa 2
Join, from category to product:
Warzywa has Marchewka
Warzywa has Baklazan
Warzywa has Cebula
Warzywa has Kalarepa
Join, from product to category:
Marchewka is in category: Warzywa
Baklazan is in category: Warzywa
Cebula is in category: Warzywa
Kalarepa is in category: Warzywa

C:\Users\Tadeusz\source\repos\IBiałkaProdutcEF\IBi
08) zakończono z kodem 0.
Aby automatycznie zamknąć konsolę po zatrzymaniu d
znie zamknij konsolę po zatrzymaniu debugowania.
Naciśnij dowolny klawisz, aby zamknąć to okno...
```



VI.

Product.cs

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


namespace IBiałkaProdutcEF
{

	class Product
	{
		public Product()
		{
			this.ProductInvoiceSet = new List<ProductInvoice>();
		}
		public int ProductId { get; set; }
		public string ProductName { get; set; }
		public int UnitsInStock { get; set; }

		public int? SupplierId { get; set; }
		[ForeignKey("SupplierId")]
		public Supplier Supplier { get; set; }

		public int? CategoryId { get; set; }
		[ForeignKey("CategoryId")]
		public Category Category { get; set; }

		public ICollection<ProductInvoice> ProductInvoiceSet { get; set; }

	}
}
```

Supplier.cs bez zmian

Category.cs bez zmian

Invoice.cs

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

namespace IBiałkaProdutcEF
{
    class Invoice
    {
        public Invoice()
        {
            this.ProductInvoiceSet = new List<ProductInvoice>();
        }
        public int InvoiceId { get; set; }
        public int InvoiceNumber { get; set; }
        public int Quantity { get; set; }

        public ICollection<ProductInvoice> ProductInvoiceSet { get; set; }
    }
}
```

ProductInvoice.cs

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

namespace IBiałkaProdutcEF
{
    class ProductInvoice
    {
        public int ProductId { get; set; }
        [ForeignKey("ProductId")]
        public Product Product { get; set; }

        public int InvoiceId { get; set; }
        [ForeignKey("InvoiceId")]
        public Invoice Invoice { get; set; }
    }
}
```

DataBaseContext.cs

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


namespace IBiałkaProdutcEF
{
	class DataBaseContext : DbContext
	{
		protected override void OnConfiguring(DbContextOptionsBuilder options)
			=> options.UseSqlite("DataSource=Entity.db");

		protected override void OnModelCreating(ModelBuilder modelBuilder)
		{
			modelBuilder.Entity<ProductInvoice>().HasKey(pi => new { pi.ProductId,
								pi.InvoiceId });
		}


		public DbSet<Product> Products { get; set; }
		public DbSet<Supplier> Suppliers { get; set; }
		public DbSet<Category> Categories { get; set; }
		public DbSet<Invoice> Invoices { get; set; }
		public DbSet<ProductInvoice> ProductInvoices { get; set; }
	}
}
```

Program.cs

```
using System;
using System.Linq;
using Microsoft.EntityFrameworkCore;

namespace IBiałkaProdutcEF
{
	class Program
	{

		static void Main(string[] args)
```

```
{
        DataBaseContext context = new DataBaseContext();

        for(int i=0; i<3; i++)
        {
                System.Console.WriteLine("Podaj nazwe produktu: ");
                String prodName = System.Console.ReadLine();
                Product prod = new Product
                {
                        ProductName = prodName,
                };
                context.Products.Add(prod);
        }

        for (int i = 0; i < 3; i++)
        {
                System.Console.WriteLine("Podaj numer tranzakcji: ");
                int invNumber = Int32.Parse(System.Console.ReadLine());
                Invoice inv = new Invoice
                {
                        InvoiceNumber = invNumber
                };
                context.Invoices.Add(inv);
        }

        context.SaveChanges();

        var Products = context.Products;
        System.Console.WriteLine("Produkty w bazie : ");
        foreach (Product p in Products)
                System.Console.WriteLine("- " + p.ProductName + " " + p.ProductId);

        var Invoices = context.Invoices;
        System.Console.WriteLine("Faktury w bazie : ");
        foreach (Invoice i in Invoices)
                System.Console.WriteLine("Numer - " + i.InvoiceNumber + " " +
                                                i.InvoiceId);

        var item_found = context.Products.Where(a => a.ProductName ==
                                                "Marchewka").Single();

        for (int i= 0; i < 2; i++)
        {
                var inv1_found = context.Invoices.Where(a => a.InvoiceNumber == i+1)
                                                        .Single();
                ProductInvoice pinv = new ProductInvoice
                {
                        ProductId = item_found.ProductId,
                        Product = item_found,
                        InvoiceId = inv1_found.InvoiceId,
                        Invoice = inv1_found
                };
                item_found.ProductInvoiceSet.Add(pinv);
                inv1_found.ProductInvoiceSet.Add(pinv);
        }

        var inv_found = context.Invoices.Where(a => a.InvoiceNumber == 3).Single();

        var item1_found = context.Products.Where(a => a.ProductName ==
                                                "Baklazan").Single();
        var item2_found = context.Products.Where(a => a.ProductName ==
                                                "Cebula").Single();

        ProductInvoice pinv1 = new ProductInvoice
        {
                ProductId = item1_found.ProductId,
                Product = item1_found,
                InvoiceId = inv_found.InvoiceId,
                Invoice = inv_found
```

```
                    };
                    ProductInvoice pinv2 = new ProductInvoice
                    {
                           ProductId = item2_found.ProductId,
                           Product = item2_found,
                           InvoiceId = inv_found.InvoiceId,
                           Invoice = inv_found
                    };
                    item1_found.ProductInvoiceSet.Add(pinv1);
                    item2_found.ProductInvoiceSet.Add(pinv2);

                    inv_found.ProductInvoiceSet.Add(pinv1);
                    inv_found.ProductInvoiceSet.Add(pinv2);

                    foreach(var pi in inv_found.ProductInvoiceSet)
                    {
                           System.Console.WriteLine("Invoice number" + inv_found.InvoiceNumber + "
                                                     has " + pi.Product.ProductName);

                    }




                    foreach (var pi in item_found.ProductInvoiceSet)
                    {
                           System.Console.WriteLine("Product "+ item_found.ProductName + " sold on "
                                                     + pi.Invoice.InvoiceNumber);
                    }

             }
       }
}
```

Konsola debugowania programu Microsoft Visual Studio

```
Marchewka
Podaj nazwe produktu:
Baklazan
Podaj nazwe produktu:
Cebula
Podaj numer transkacji:
1
Podaj numer transkacji:
2
Podaj numer transkacji:
3
Produkty w bazie :
- Marchewka 1
- Baklazan 2
- Cebula 3
Faktury w bazie :
Numer - 1 1
Numer - 2 2
Numer - 3 3
Invoice number3 has Baklazan
Invoice number3 has Cebula
Product Marchewka sold on 1
Product Marchewka sold on 2

C:\Users\Tadeusz\source\repos\IBiałkaProdutcEF\IBia
84) zakończono z kodem 0.
Aby automatycznie zamknąć konsolę po zatrzymaniu de
znie zamknij konsolę po zatrzymaniu debugowania.
Naciśnij dowolny klawisz, aby zamknąć to okno...
```

VII.

Product.cs bez zmian

Supplier.cs

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

namespace IBiałkaProdutcEF
{
    public class Supplier: Company
    {
        public int BankAccountNumber { get; set; }
    }
}
```

Customer.cs

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

namespace IBiałkaProdutcEF
{
    public class Customer: Company
    {
        public double Discount { get; set; }
    }
}
```

Company.cs

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

namespace IBiałkaProdutcEF
{
    public class Company
    {
        public int CompanyId { get; set; }
        public string CompanyName { get; set; }
        public string Street { get; set; }
        public string City { get; set; }
        public int ZipCode { get; set; }
    }
}
```

Category.cs bez zmian

Invoice.cs bez zmian

ProductInvoice.cs bez zmian

DataBaseContext.cs

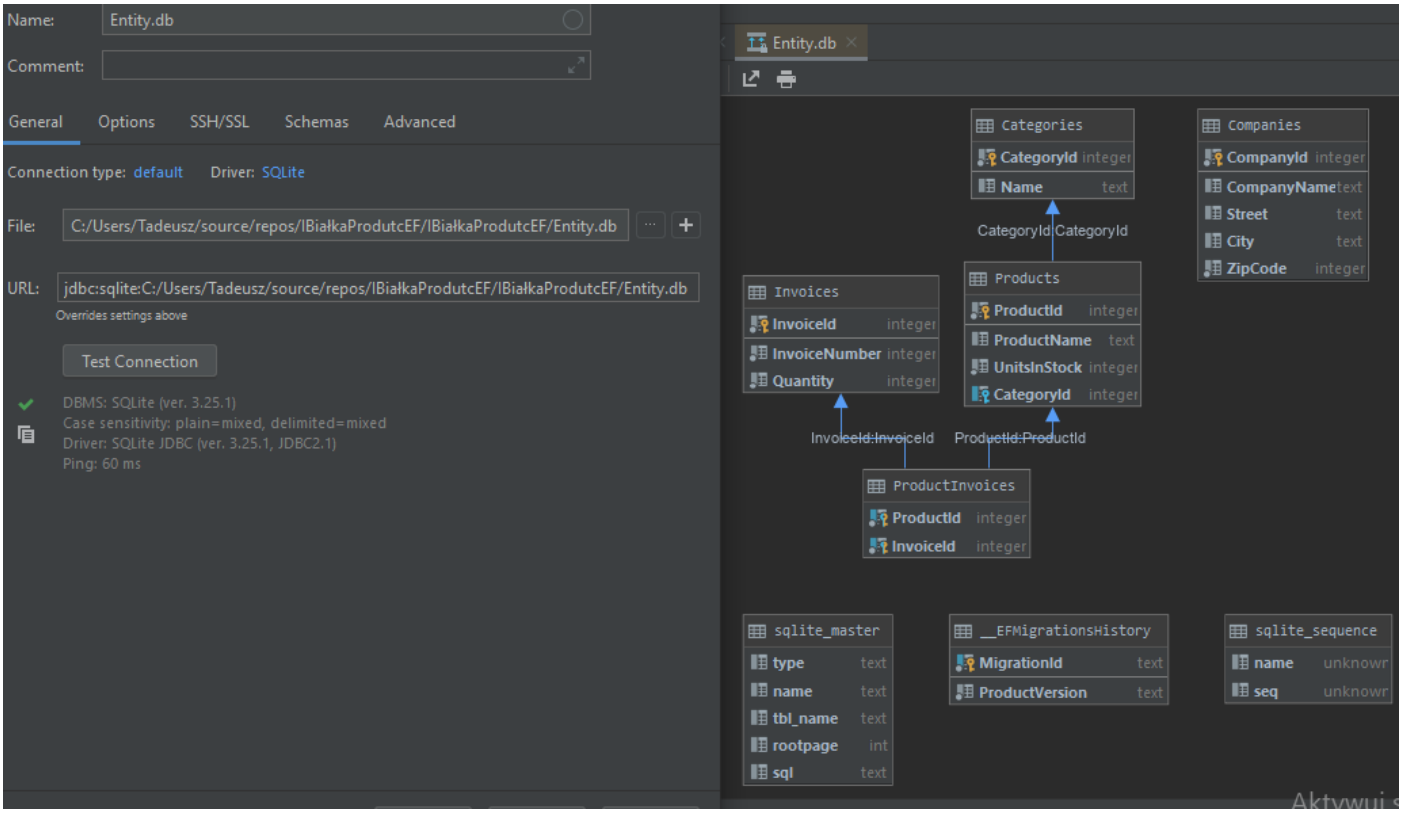
```
using System;
using System.Collections.Generic;
using System.Text;
using Microsoft.EntityFrameworkCore;


namespace IBiałkaProdutcEF
{
       class DataBaseContext : DbContext
       {
              protected override void OnConfiguring(DbContextOptionsBuilder options)
                     => options.UseSqlite("DataSource=Entity.db");

              protected override void OnModelCreating(ModelBuilder modelBuilder)
              {
                     modelBuilder.Entity<ProductInvoice>().HasKey(pi => new { pi.ProductId,
                                                                              pi.InvoiceId });

              }


              public DbSet<Product> Products { get; set; }
              public DbSet<Category> Categories { get; set; }
              public DbSet<Invoice> Invoices { get; set; }
```

```
            public DbSet<ProductInvoice> ProductInvoices { get; set; }
            public DbSet<Company> Companies { get; set; }
        }
}
```

Program.cs

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

namespace IBiałkaProdutcEF
{
        class Program
        {

            static void Main(string[] args)
            {
                DataBaseContext context = new DataBaseContext();

                for (int i=0; i<3; i++)
                {
                    System.Console.WriteLine("Podaj nazwe dostawcy: ");
                    String suppName = System.Console.ReadLine();
                    Supplier supp = new Supplier
                    {
                        CompanyName = suppName,
                    };
                    context.Companies.Add(supp);
                }

                for (int i = 0; i < 3; i++)
                {
                    System.Console.WriteLine("Podaj nazwe klienta: ");
                    String cusName = System.Console.ReadLine();
                    Customer cus = new Customer
                    {
                        CompanyName = cusName
                    };
                    context.Companies.Add(cus);
                }

                context.SaveChanges();

                IQueryable<Supplier> Suppliers = from s in context.Companies.OfType<Supplier>()
                                                 select s;

                System.Console.WriteLine("Dostawcy w bazie : ");
                foreach (Supplier sup in Suppliers)
                    System.Console.WriteLine("- " + sup.CompanyName + " " + sup.CompanyId);

                IQueryable<Customer> Customers = from c in context.Companies.OfType<Customer>()
                                                 select c;

                System.Console.WriteLine("Klienci w bazie : ");
                foreach (Customer cup in Customers)
                    System.Console.WriteLine("- " + cup.CompanyName + " " + cup.CompanyId);

            }
        }
}
```

Strategia TablePerType i TablePerClass nie jest dostępna w obecnej wersji Entity Framework Core.


Name: Entity.db
Comment:
General
Options
SSH/SSL
Schemas
Advanced
Connection type: default
Driver: SQLite
File: C:/Users/Tadeusz/source/repos/IBiałkaProdutcEF/IBiałkaProdutcEF/Entity.db
URL: jdbc:sqlite:C:/Users/Tadeusz/source/repos/IBiałkaProdutcEF/IBiałkaProdutcEF/Entity.db
Overrides settings above
Test Connection
DBMS: SQLite (ver. 3.25.1)
Case sensitivity: plain=mixed, delimited=mixed
Driver: SQLite JDBC (ver. 3.25.1, JDBC2.1)
Ping: 60 ms
Entity.db
Categories
CategoryId integer
Name text
Companies
CompanyId integer
CompanyName text
Street text
City text
ZipCode integer
CategoryId CategoryId
Invoices
InvoiceId integer
InvoiceNumber integer
Quantity integer
Products
ProductId integer
ProductName text
UnitsInStock integer
CategoryId integer
InvoiceId:InvoiceId
ProductId:ProductId
ProductInvoices
ProductId integer
InvoiceId integer
sqlite_master
type text
name text
tbl_name text
rootpage int
sql text
__EFMigrationsHistory
MigrationId text
ProductVersion text
sqlite_sequence
name unknowr
seq unknowr