

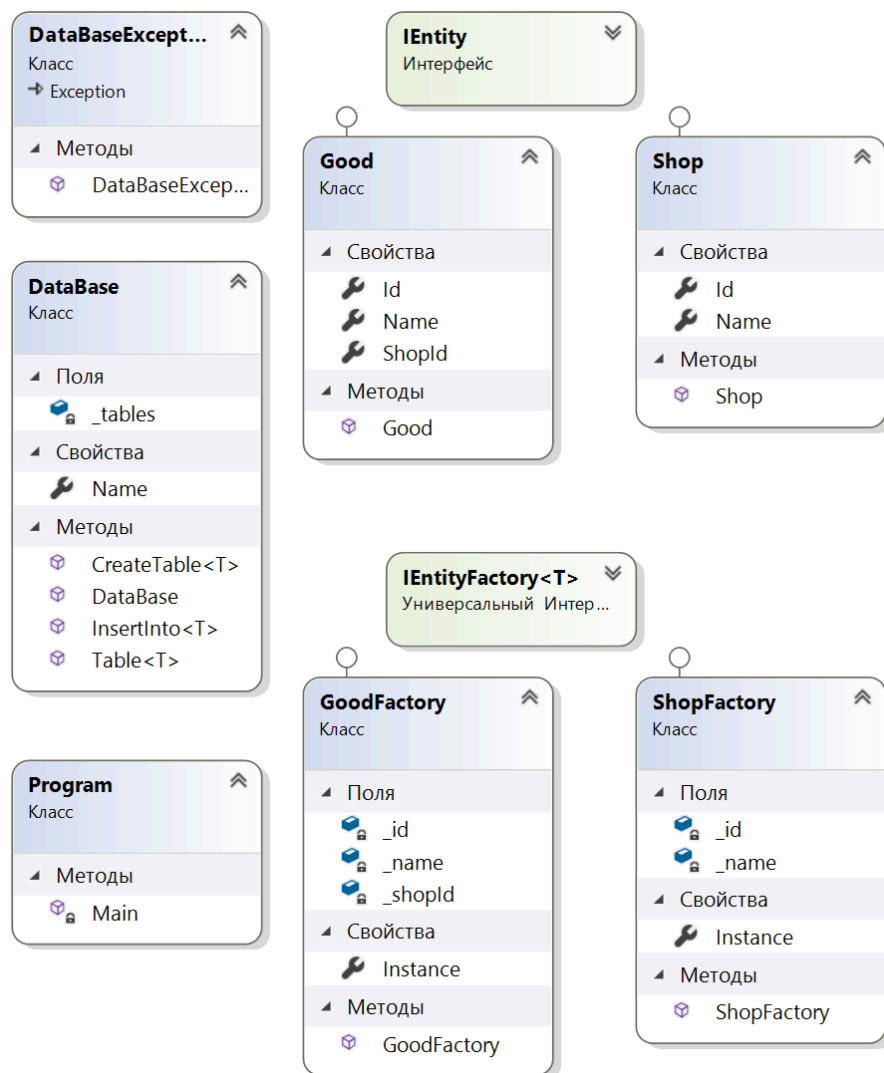
Тема: LINQ-запросы, интерфейсы, обобщения, работа с файлами, коллекции

Прежде чем приступить к выполнению задания необходимо ознакомиться с:

- особенностями реализации шаблона проектирования «Фабрика» – <https://refactoring.guru/ru/design-patterns/factory-method>
- коллекцией Dictionary <https://docs.microsoft.com/ru-ru/dotnet/csharp/programming-guide/classes-and-structs/how-to-initialize-a-dictionary-with-a-collection-initializer>
- схема «звезда» для базы данных <http://market-pages.ru/ias/22.html>

Ознакомьтесь с решением DataBaseTask2 (<https://github.com/hse-programming-CSharp2018-2019/CSharp2018-2019/tree/master/4module/DataBaseTask2>). В решении смоделирована простая база данных, состоящая из таблиц сущностей двух типов Good и Shop. Good – товар, который можно купить (каждый товар имеет уникальный номер). Shop – магазин, в котором можно купить товары (каждый магазин имеет уникальный номер).

Ниже представлена диаграмма классов:

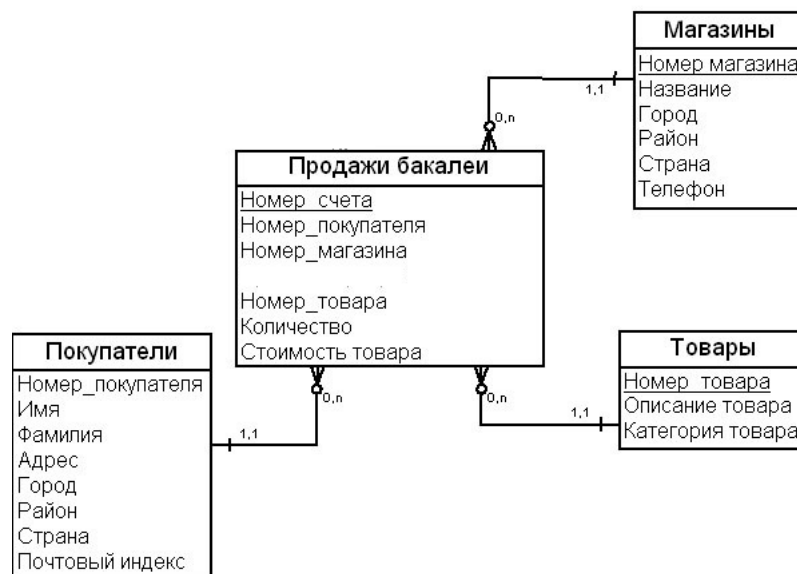


DataBase имитирует базу данных и состоит из набора таблиц различных типов.

Для создания таблицы используется метод `db.CreateTable<T>()`, где `T` – тип сущностей, хранимых в таблице.

Для добавления объекта в таблицу базы данных используется метод `db.InsertInto<T>(new TFactory(...))`.

Задание 1. Необходимо доработать программу, таким образом, чтобы база данных выглядела как на схеме ниже.



Таким образом, необходимо:

- 1) Расширить список типов таблиц. Добавить таблицу сущностей типа Buyer (покупатели) и таблицу сущностей типа Sale (продажи), состоящих из списка покупателей и списка всех продаж, соответственно.
- 2) Добавить тип Buyer.
- 3) Расширить классы Good и Shop (добавить необходимые поля, в соответствии со схемой выше).

Задание 2.

- 1) Добавить механизм сохранения базы данных в виде файлов формата JSON (используйте механизм JSON-сериализации). Каждая таблица должна быть сохранена в отдельном файле с названием "DB<Type>", например, таблица с товарами будет называться "DBGood" и выглядеть следующим образом (поля из задания 1 не добавлены):

```
{
  "Id":1,
  "Name":"Pepsi",
  "ShopId":1
},{
  "Id":29,
  "Name":"Lays",
  "ShopId":2
}
```

- 2) Добавить механизм создания базы данных из файлов формата JSON (см. п. 1)). То есть, необходимо реализовать возможность восстановить базу данных.

Задание 3.

- 1) Создать LINQ-запрос, позволяющий вывести все товары, купленные покупателем с самым длинным именем.
- 2) Создать LINQ-запрос, позволяющий найти категорию самого дорогого товара.

- 3) Создать LINQ-запрос, позволяющий найти город, в котором общая сумма всех продаж наименьшая.
- 4) Создать LINQ-запрос, позволяющий найти фамилии покупателей, купивших самый популярный товар (популярный товар – товар, которого было куплено наибольшее количество среди всех покупок).
- 5) Создать LINQ-запрос, позволяющий найти количество магазинов в стране, где их меньше всего.
- 6) Создать LINQ-запрос, позволяющий найти покупки, которые были сделаны не в своем городе, т. е. если покупатель проживает в городе А, а покупку сделал в городе Б.
- 7) Создать LINQ-запрос, позволяющий найти общую стоимость всех покупок.

Задание 4.

Разработать меню для консольного приложения, которое позволит:

- 1) Добавлять новые сущности (покупки, товары, магазины, покупателей).
- 2) Сохранять отдельные таблицы в виде JSON-файлов.
- 3) Меню может реализовывать и другой функционал, который необходим для работы программы.