СОЗДАНИЕ ТЕСТОВОЙ БАЗЫ ДАННЫХ HOSPITAL С ПРИМЕНЕНИЕМ ТЕХНОЛОГИИ ENTITY FRAMEWORK CORE

Габдуллин Д.Р., 32 гр. ПИ, 3 к., студент

Тазетдинов Б.И., к.ф-м.н., доцент,

Бирский филиал УУНиТ, г. Бирск, Россия

Аннотация. В данной работе создана тестовая база данных Hospital реализованной с помощью технологии доступа к базе данных Entity FrameWork Core, которая позволила изучить основы создания и поддержки в дальнейшем готовой базы данных в проекте с применением шаблона CodeFirst.

Ключевые слова: База данных, Entity FrameWork Core, C#.

В настоящее время очень распространенным инструментом разработки и доступа к базам данных (БД) является технология EntityFrameWorkCore. Была поставлена задача разработать тестовую базу данных Hospital для изучения основ создания и поддержки в дальнейшем готовой базы данных в проекте.

В базе данных определены три основные сущности: **Patient** (Пациент), **MedCard** (Медицинская карта) и **Specialist** (Специалист).

Рассмотрим каждую сущность более подробно.

Сущность Patient (Пациент):

Сущность представляет собой информацию о пациенте больницы. Она содержит следующие поля:

- **GuidId**: Уникальный идентификатор пациента.
- **stringName**: Имя пациента.

- **stringSurname**: Фамилия пациента.
- **string? Patronymic**: Отчество пациента (может быть пустым).
- **intAge**: Возраст пациента.
- **MedCard? MedCard**: Связь с медицинской картой пациента. Каждый пациент может иметь только одну медицинскую карту.

Сущность MedCard (Медицинская карта):

Сущность описывает медицинскую карту пациента. Она включает следующие поля:

- **GuidId**: Уникальный идентификатор карты.
- **GuidPatientId**: Идентификатор пациента, которому принадлежит медицинская карта.
- **Patient? Patient**: Навигационное свойство для связи с объектом пациента.
- **List<Specialist>Specialists**: Список специалистов, которые работали с этой медицинской картой. Каждая медицинская карта может содержать несколько записей о специалистах.

Сущность Specialist (Специалист):

Сущность хранит данные о специалистах, работающих в больнице. Поля включают:

- **Guid Id**: Уникальный идентификатор специалиста.
- **string Name**: Имя специалиста.
- **string Surname**: Фамилия специалиста.
- **string? Patronymic**: Отчество специалиста (может быть пустым).
- **string Speciality**: Специальность специалиста.
- **int Age**: Возраст специалиста.

- **List<MedCard> MedCards**: Список медицинских карт, с которыми работал данный специалист. Один специалист может работать с несколькими медицинскими картами.

Контекст базы данных HospitalDbContext:

Контекст базы данных определяет три набора данных, соответствующие каждой из сущностей:

- **DbSet<Specialist> Specialists**
- **DbSet<MedCard> MedCards**
- **DbSet<Patient> Patients**

Эти наборы позволяют выполнять операции CRUD (создание, чтение, обновление, удаление) над данными через Entity Framework Core.

На рисунке 1 показана диаграмма сущность связь, разрабатываемой базы данных.

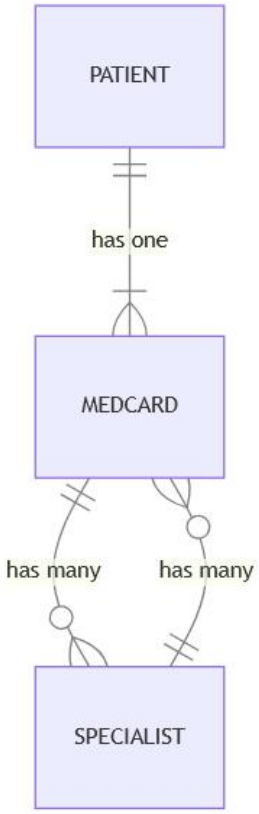

Рис. 1. Диаграмма сущность-связь, разрабатываемой базы данных

Взаимосвязи:

1. **Один к одному (One-to-one)**:

   Пациент имеет ровно одну медицинскую карту. Это связь реализуется через поле MedCard в классе Patient, а также через поле PatientId и навигационное свойство Patient в классе MedCard.

2. **Многие ко многим (Many-to-many)**:

   Медицинские карты могут включать записи о нескольких специалистах, и каждый специалист может работать с несколькими медицинскими картами. Эта связь реализована через списки Specialists в классе MedCard и MedCards в классе Specialist.

Реализация привязки данных показана на листинге кода ниже.

C#’’’

```
// к пациенту добавили мед карту
MedCard medcard1 = new MedCard { Patient = patients[0]
};

context.AddRange(medcard1, medcard2);

//привязываем специалистов к медицинским картам
medcard1.Specialists.AddRange(new List<Specialist>() {
specialists[1], specialists[2] });
medcard2.Specialists.AddRange(new List<Specialist>() {
specialists[0], specialists[2] });

context.SaveChanges();
```

’’’

При работе над проектом в качестве справочных материалов использовались литературные источники [1-3].

Заключение

В работе была создана база данных Hospital с применением технологии EntityFrameWorkCore. Были изучены основы работы с данной технологией для создания, чтения, обновления и удаления из базы данных. Получен опыт работы над базой данных по шаблону CoderFirst.

Литература

1. Пособие по EFCore– URL: https://metanit.com/sharp/efcore/(Дата обращения: 28.11.24)
2. Построение mermaid диаграмм – URL: https://mermaid.live/ (Дата обращения: 28.11.24)
3. Внешние ключи и связи –URL: https://metanit.com/sql/tutorial/1.3.php (Дата обращения: 28.11.24)