AppleStore + БД

Введение

В данном документе представлена информация о базе данных для проекта AppleStore. Мы рассмотрим структуру базы данных, таблицы и их взаимосвязи, а также примеры SQL-запросов для работы с данными.

Убедитесь, что у вас есть SQLite и инструменты для работы с ним, так как в рамках проекта мы будем использовать именно эту СУБД.

Инструменты для работы с базой данных

SQLite

SQLite - это встраиваемая кроссплатформенная база данных, которая не требует отдельного сервера и настройки. Она идеально подходит для разработки и тестирования приложений.

Скачать SQLite: https://www.sqlite.org/download.html

DB Browser for SQLite

DB Browser for SQLite - это визуальный инструмент для работы с базами данных SQLite, который позволяет создавать, редактировать и выполнять запросы без необходимости писать SQL-код вручную.

Скачать DB Browser for SQLite: https://sqlitebrowser.org/dl/

Структура базы данных AppleStore

База данных AppleStore содержит следующие таблицы:

- Categories - категории товаров
- Products - товары
- ProductDetails - детали товаров
- Customers - клиенты
- Employees - сотрудники
- Orders - заказы

- OrderItems - элементы заказов

Таблица Categories

```
CREATE TABLE Categories (
    CategoryId INT PRIMARY KEY,
    Name NVARCHAR(100) NOT NULL,
    Description NVARCHAR(500)
);
```

Данные в таблице Categories:

```
1|iPhone|Смартфоны Apple iPhone различных моделей
2|iPad|Планшеты Apple iPad различных моделей
3|Mac|Компьютеры и ноутбуки Apple Mac
4|Watch|Умные часы Apple Watch
5|AirPods|Беспроводные наушники Apple
```

Таблица Products

```
CREATE TABLE Products (
    ProductId INT PRIMARY KEY,
    CategoryId INT NOT NULL,
    Name NVARCHAR(100) NOT NULL,
    Description NVARCHAR(500),
    Price DECIMAL(10, 2) NOT NULL,
    StockQuantity INT NOT NULL DEFAULT 0,
    FOREIGN KEY (CategoryId) REFERENCES Categories(CategoryId)
);
```

Данные в таблице Products (первые 5 записей):

```
1|1|iPhone 13 Pro|6.1-дюймовый дисплей, A15 Bionic, 128GB|
999.99|50
2|1|iPhone 13|6.1-дюймовый дисплей, A15 Bionic, 128GB|799.99|75
3|2|iPad Pro 12.9|12.9-дюймовый дисплей, M1 чип, 256GB|1099.99|
30
4|2|iPad Air|10.9-дюймовый дисплей, M1 чип, 64GB|599.99|45
5|3|MacBook Pro 14|14-дюймовый дисплей, M1 Pro, 512GB SSD|
1999.99|25
```

Таблица Orders

```
CREATE TABLE Orders (
    OrderId INT PRIMARY KEY,
    CustomerId INT NOT NULL,
    EmployeeId INT,
    OrderDate DATETIME NOT NULL,
    TotalAmount DECIMAL(10, 2) NOT NULL DEFAULT 0,
    Status NVARCHAR(50) NOT NULL,
    FOREIGN KEY (CustomerId) REFERENCES Customers(CustomerId),
    FOREIGN KEY (EmployeeId) REFERENCES Employees(EmployeeId)
);
```

Данные в таблице Orders:

```
1|1|1|2023-01-15 10:30:00|999.99|Выполнен
2|2|2|2023-01-20 14:45:00|1699.98|Выполнен
3|3|1|2023-01-25 16:20:00|649.98|В обработке
```

Примеры SQL-запросов

Запрос на объединение таблиц Products и Categories

```
SELECT p.Name, p.Price, c.Name as Category
FROM Products p
JOIN Categories c ON p.CategoryId = c.CategoryId
LIMIT 5;
```

Результат запроса:

```
iPhone 13 Pro|999.99|iPhone
iPhone 13|799.99|iPhone
iPad Pro 12.9|1099.99|iPad
iPad Air|599.99|iPad
MacBook Pro 14|1999.99|Mac
```

Интеграция с .NET

В Visual Studio убедитесь, что у вас есть менеджер пакетов NuGet и установите необходимый пакет для работы с SQLite:

```
Install-Package Microsoft.EntityFrameworkCore.Sqlite
```

Модели данных

Скопируйте себе код, исправьте под себя, запустите и изучите:

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

namespace AppleStore
{
    public class AppleStoreContext : DbContext
    {
        public DbSet<Category> Categories { get; set; }
        public DbSet<Product> Products { get; set; }
        public DbSet<ProductDetail> ProductDetails { get; set; }
        public DbSet<Customer> Customers { get; set; }
        public DbSet<Employee> Employees { get; set; }
        public DbSet<Order> Orders { get; set; }
        public DbSet<OrderItem> OrderItems { get; set; }

        protected override void
OnConfiguring(DbContextOptionsBuilder optionsBuilder)
        {
            optionsBuilder.UseSqlite("Data
Source=AppleStore.db");
        }
    }

    public class Category
    {
        public int CategoryId { get; set; }
        public string Name { get; set; }
        public string Description { get; set; }

        public List<Product> Products { get; set; } = new
List<Product>();
    }

    public class Product
    {
        public int ProductId { get; set; }
        public int CategoryId { get; set; }
        public string Name { get; set; }
        public string Description { get; set; }
        public double Price { get; set; }
        public int StockQuantity { get; set; }
```

```
        public Category Category { get; set; }
        public ProductDetail Details { get; set; }
        public List<OrderItem> OrderItems { get; set; } = new
List<OrderItem>();
    }

    public class ProductDetail
    {
        public int ProductId { get; set; }
        public string Color { get; set; }
        public string ScreenSize { get; set; }

        public Product Product { get; set; }
    }

    public class Customer
    {
        public int CustomerId { get; set; }
        public string FullName { get; set; }
        public string Email { get; set; }
        public string Phone { get; set; }
        public string Address { get; set; }

        public List<Order> Orders { get; set; } = new
List<Order>();
    }

    public class Employee
    {
        public int EmployeeId { get; set; }
        public string FullName { get; set; }
        public string Position { get; set; }
        public string Email { get; set; }
        public string Phone { get; set; }

        public List<Order> Orders { get; set; } = new
List<Order>();
    }

    public class Order
    {
        public int OrderId { get; set; }
        public int CustomerId { get; set; }
        public int EmployeeId { get; set; }
        public DateTime OrderDate { get; set; }
        public double TotalAmount { get; set; }
        public string Status { get; set; }

        public Customer Customer { get; set; }
        public Employee Employee { get; set; }
        public List<OrderItem> OrderItems { get; set; } = new
```

```
List<OrderItem>();
    }

    public class OrderItem
    {
        public int OrderItemId { get; set; }
        public int OrderId { get; set; }
        public int ProductId { get; set; }
        public int Quantity { get; set; }
        public double UnitPrice { get; set; }

        public Order Order { get; set; }
        public Product Product { get; set; }
    }
}
```

Пример использования базы данных в коде

```
using System;
using System.Linq;

namespace AppleStore
{
    class Program
    {
        static void Main(string[] args)
        {
            using (var context = new AppleStoreContext())
            {
                // Получение всех категорий
                var categories = context.Categories.ToList();
                Console.WriteLine("Категории товаров:");
                foreach (var category in categories)
                {
                    Console.WriteLine($"{category.CategoryId}.
{category.Name} - {category.Description}");
                }

                // Получение товаров определенной категории
                var iPhones = context.Products
                    .Where(p => p.CategoryId == 1)
                    .ToList();

                Console.WriteLine("\nТовары категории iPhone:");
                foreach (var iPhone in iPhones)
                {
                    Console.WriteLine($"{iPhone.Name} -
{iPhone.Price:C}");
                }
```

```
                // Получение заказов с информацией о клиентах
                var orders = context.Orders
                    .Join(context.Customers,
                        o => o.CustomerId,
                        c => c.CustomerId,
                        (o, c) => new { Order = o, Customer =
c })
                    .ToList();

                Console.WriteLine("\nЗаказы клиентов:");
                foreach (var order in orders)
                {
                    Console.WriteLine($"Заказ №
{order.Order.OrderId} от {order.Order.OrderDate:dd.MM.yyyy
HH:mm}");
                    Console.WriteLine($"Клиент:
{order.Customer.FullName}");
                    Console.WriteLine($"Сумма:
{order.Order.TotalAmount:C}");
                    Console.WriteLine($"Статус:
{order.Order.Status}");
                    Console.WriteLine();
                }
            }
        }
    }
}
```

Заключение

В этом документе мы рассмотрели структуру базы данных для проекта AppleStore, включая таблицы, их взаимосвязи и примеры SQL-запросов. Также мы показали, как интегрировать базу данных с .NET приложением с использованием Entity Framework Core.

База данных содержит все необходимые таблицы для хранения информации о товарах, категориях, клиентах, сотрудниках и заказах, что позволяет создать полноценное приложение для управления магазином Apple техники.