АРХАНГЕЛЬСКИЙ КОЛЛЕДЖ ТЕЛЕКОММУНИКАЦИЙ
ИМ. Б.Л. РОЗИНГА (ФИЛИАЛ) СПбГУТ
(АКТ (ф) СПбГУТ)

Отчеты по лабораторным и практическим работам МДК 11.01

Студент ИСПП-35 *А.Е. Туйкова*

(Группа) (Подпись) (Дата) (И.О. Фамилия)

Преподаватель *Ю. С. Маломан*

(Подпись) (Дата) (И.О. Фамилия)

Архангельск 2024

Лабораторная работа №1
Сбор и анализ требований методом use-case

1. Цель работы

1.1. Изучить процесс описания требований к системе методом use-case.

1.2. Изучить процесс создания диаграммы вариантов использования.

2. Контрольные вопросы

2.1.Для чего используется диаграмма вариантов использования?

Отражает отношения между “акторами” и “прецедентами” и является составной частью модели прецедентов, позволяющей описать систему на концептуальном уровне.

2.2.Что такое «актор» и как он обозначается на диаграмме вариантов использования?

Актор – внешняя сущность, взаимодействующая с системой (пользователь, устройство, другая система).

На диаграмме обозначается палочкой.

2.3.Что такое «прецедент» и как он обозначается на диаграмме вариантов использования?

Прецедент – конкретный вариант использования системы, описывающий её функциональность. На диаграмме обозначается овалом.

2.4. Что обозначает «отношение ассоциации»?

Указывает на связь между актором и прецедентом.

2.5. Что обозначает «отношение обобщения»?

Означает наследование свойства и поведение одного элемента другим.

2.6. Что обозначает «отношение включения»?

Означает, что один прецедент всегда включает в себя выполнение другого прецедента.

2.7. Что обозначает «отношение расширения»?

Означает, что дополнительный прецедент может быть выполнен в рамках основного, если выполняются определенные условия.

3. Вывод

3.1.В ходе лабораторной работы был изучен процесс описания требований к системе методом use-case, а также процесс создания диаграммы вариантов использования.

Лабораторная работа №2
Проектирование реляционной схемы базы данных в среде СУБД

1. Цель работы

1.1. Научиться применять MySQL Workbench в процессе создания схем моделей БД;

1.2. Научиться представлять логическую модель данных согласно нотациям ERD и IDEF1X.

2. Контрольные вопросы

2.1. Что такое «сущность»?

Это объект системы или предметной области, категория, представляющая данные, которые необходимо хранить в системе.

2.2. Что такое «атрибут»?

Атрибут – характеристика или свойство сущности, которое описывает ее состояние или данные.

2.3. Что такое «ключевое поле»?

Атрибут – атрибут или набор атрибутов, который уникально идентифицирует запись в таблице базы данных.

2.4. Каково назначение первичных и внешних ключей?

Первичный ключ – уникальный идентифицирует каждую запись в базе данных таблицы. Он обеспечивает целостность данных, гарантируя, что в таблице нет дублирующихся записей. Внешний ключ – используется для просмотра связи между таблицами. Он присылает на первый ключ другую таблицу, обеспечивая связь с данными и удержание ссылочной почты.

2.5. Что такое «связь»?

Связь – отношение между сущностями (таблицами) в базе данных, определяющее, как данные одной таблицы с другой.

2.6. Какие виды связей между сущностями существуют?

Один к одному (1:1), один ко многим (1:M), многие ко многим (M:M).

2.7. Какие элементы входят в ER-диаграммы?

Сущности, Атрибуты, Связи, Ключи, Кардинальность.

2.8 Для чего применяются ER-диаграммы?

ER-диаграммы применяются для моделирования структур баз данных.

3. Вывод

3.1.В ходе лабораторной работы научились применять MySQL Workbench в процессе создания схем моделей БД, а также представлять логическую модель данных согласно нотациям ERD и IDEF1X.

Лабораторная работа №3
Приведение БД к нормальной форме

1. **Цель работы**

1.1. Изучить процесс приведения отношений от ненормализованного вида к четвертой нормальной форме;

1.2. Изучить процесс декомпозиции отношений.

2. Контрольные вопросы

2.1. Что называется первичным ключом отношения?

Это атрибут в таблице который уникально идентифицирует каждую запись в этой таблице.

2.2. Что называется внешним ключом отношения?

Это атрибут, который ссылается на первичный ключ другой таблицы.

2.3. В чем заключается процесс нормализации отношений?

Это процесс упорядочивания данных на основе повторяемости и предотвращения аномалий.

2.4. В каком случае атрибут A функционально зависит от атрибута B?

Если для каждой уникальной величины B существует только одна соответствующая величина A.

2.5. В каком случае атрибут A транзитивно зависит от атрибута B?

Если существует атрибут C, который зависит от B и при этом атрибут A зависит от C.

2.6. Каким требованиям должно отвечать отношение, находящееся в 1НФ?

Отношение в 1НФ должно сохранять атомарные значения (неделимые) и не иметь повторяющихся групп данных или множественных результатов в одной ячейке

2.7. Каким требованиям должно отвечать отношение, находящееся во 2НФ?

Отношения во 2НФ должны быть в 1НФ и не иметь частичных зависимостей, все неключевые атрибуты должны зависеть от всего первичного ключа.

2.8. Каким требованиям должно отвечать отношение, находящееся в 3НФ?

Отношение 3НФ должно быть в 2НФ и не иметь транзитивных зависимостей между не ключевыми атрибутами

3. Вывод

3.1. В ходе лабораторной работы изучили процесс приведения отношений от ненормализованного вида к четвертой нормальной форме, а также процесс декомпозиции отношений.

Лабораторная работа №4
Установка SQL сервера

1. Цель работы

1.1. Научиться устанавливать разные СУБД, используя docker.

2. Контрольные вопросы

2.1. Какие стандартные порты для подключения к СУБД MSSQL, MySQL, Postgres?

MSSQL: 1433, MySQL: 3306, PostgreSQL: 5432

2.2. Какие стандартные папки для хранения данных СУБД MSSQL, MySQL, Postgres?

MSSQL: C:\Program Files\Microsoft SQL Server\MSSQLXX.MSSQLSERVER\MSSQL\DATA\

MySQL: C:\ProgramData\MySQL\MySQL Server X.X\data\

PostgresSQL: C:\Program Files\PostgreSQL\XX\data\

2.3. Для чего используется Docker?

Docker используется для упаковки приложений и их зависимостей в контейнерах, обеспечения изоляции, переносимости и упрощения развертывания в разных средах.

3. Вывод

3.1. В ходе лабораторной работы установили различные СУБД, используя docker.

Лабораторная работа №5
Создание базы данных в среде разработки

1. Цель работы

1.1. Изучить способы обеспечения целостности данных в MS SQL Server;

1.2. Научиться работать в среде SQL Server Management Studio (SSMS).

2. Контрольные вопросы

2.1. Что такое SQL Server Management Studio?

утилита из Microsoft SQL Server 2005 и более поздних версий для конфигурирования, управления и администрирования всех компонентов Microsoft SQL Server.

2.2. Какие виды авторизации поддерживаются в MS SQL Server?

Windows Authentication (Аутентификация Windows) или SQL Server Authentication (Аутентификация SQL Server).

2.3. Что такое первичный ключ?

Это атрибут который уникально идентифицирует каждую запись в этой таблице.

2.4. Как указать заполнение столбца автоинкрементными значениями?

IDENTITY.

2.5. Как указать значение по умолчанию?

DEFAULT.

2.6. Как задать проверочное ограничение?

CHECK

2.7. Как обеспечить уникальность значений в столбце или наборе столбцов?

UNIQUE.

2.8. Что такое внешний ключ?

Это атрибут в одной таблице, который ссылается на первичный ключ другой таблице.

2.9. Какие значения может принимать внешний ключ?

Может учитывать любые значения из связанной таблицы или значения NULL, если это разрешено.

3.Вывод

3.1. В ходе лабораторной работы был изучен способ обеспечения целостности данных в MS SQL Server, а также изучила как работать в среде SQL Server Management Studio (SSMS).

Лабораторная работа №6
Создание представлений в СУБД

1. Цель работы

1.1. Научиться создавать и использовать представления в MS SQL Server;

1.2. Закрепить навык создания запросов на выборку в MS SQL Server.

2. Контрольные вопросы

2.1. Что такое представления и чем они отличаются от таблиц?

Это виртуальная таблица, которая отображает данные на основе запроса.

2.2. Для чего применяются представления?

Используется для упрощения запросов, фильтрации данных, повышения безопасности и логики инкапсуляции

2.3. Возможно ли создание представления, включающего информацию из нескольких таблиц одновременно?

Да, для этого используется соединения (JOIN)

2.4. Какие требование предъявляются к обновляемым представлениям?

Они должны основываться на одной таблице, без агрегатов, группировок и возвращаемых полей, с указанием соответствия обновляемых полей таблицы

2.5. Как создать обновляемое представление?

```
CREATE VIEW view_name AS
SELECT column1, column2
FROM table_name
```

3. Вывод

3.1. В ходе лабораторной работы мы научились создавать и использовать представления в MS SQL Server, а также закрепили навык создания запросов на выборку в MS SQL Server.

Лабораторная работа №7
Создание функций пользователя в СУБД

1.Цель работы

1.1. Научиться создавать и использовать скалярные и табличные функции пользователя в MS SQL Server;

1.2. Закрепить навык создания запросов на выборку в MS SQL Server.

2. Контрольные вопросы

2.1.Что такое функции пользователя и для чего они применяются?

Функции, которые вы используете для организации кода в основной части политики

2.2.Чем отличается скалярная функция от табличной?

Скалярные функции - функции, которые возвращают число или текст, то есть одно или несколько значений; Табличные функции - функции, которые выводят результат в виде таблицы.

2.3.Как объявить переменную и присвоить ей значение?

DECLARE @variable_name datatype [= initial_value]

2.4.Как указать в функции параметр по умолчанию?

@param1 INT = 10

2.5.Как вызвать скалярную функцию?

SELECT dbo.function_name(param1, param2);

2.6.Как вызвать табличную функцию?

SELECT * FROM dbo.function_name(param1, param2);

3. Вывод

3.1.В ходе лабораторной работы создали и использовали скалярные и табличные функции пользователя в MS SQL Server.

Лабораторная работа №8
Создание хранимых процедур в СУБД

1. Цель работы

1.1. Научиться создавать и использовать хранимые процедуры в MS SQL Server;

1.2. Закрепить навык создания запросов на выборку и модификацию данных в MS SQL Server.

2. Контрольные вопросы

2.1. Что такое хранимые процедуры и для чего они применяются?

Это заранее написанные и защищенные в базе данных программы, которые выполняют одну или несколько операций (например, запрос, вставку, обновление

2.2. Чем отличается функция пользователя от хранимой процедуры?

Пользовательские функции можно использовать как и любое другое выражение в SQL запросе, в то время как хранимые процедуры должны быть вызваны с помощью функции CALL

2.3. Как задать входные параметры хранимой процедуры?

Объявляются параметры в скобках после имени процедуры: @имя_параметра1 тип_данных1, @имя_параметра2 тип_данных2, …

2.4. Как задать выходные параметры хранимой процедуры?

Ответ:

Объявляются параметры в скобках после имени процедуры, помечая их как OUTPUT: @имя_параметра1 тип_данных1 OUTPUT, @имя_параметра2 тип_данных2 OUTPUT

Внутри процедуры присваиваются значения выходным параметрам.

При вызове процедуры передаются ссылки на переменные для получения выходных значений: EXEC имя_процедуры @имя_параметра1 = @переменная1 OUTPUT, @имя_параметра2 = @переменная2 OUTPUT

2.5. Как вызвать выполнение хранимой процедуры?

Вызов хранимой процедуры:

EXEC имя_процедуры [параметры]

3. Вывод

3.1. В ходе лабораторной работы создали и использовали хранимые процедуры в MS SQL Server.

Лабораторная работа №9
Создание триггеров в СУБД

1. Цель работы

1.1. Научиться создавать и использовать триггеры в MS SQL Server;

1.2. Закрепить навык создания запросов на выборку и модификацию данных в MS SQLServer.

2. Контрольные вопросы

2.1. Что такое триггер?

Триггер — это хранимая процедура, которая автоматически выполняется при наступлении определенного события в базе данных (INSERT, UPDATE, DELETE).

2.2. Чем триггер отличается от хранимой процедуры?

триггер срабатывает автоматически при возникновении событий (INSERT, UPDATE, DELETE), а хранимая процедура возникает вручную и по запросу.

2.3. Как запустить триггер на выполнение?

Триггер запускается автоматически при наступлении событий (INSERT, UPDATE, DELETE).

2.4. Каково назначение триггеров?

Ограничение выполнения действий при наступлении определенного события (например, вставка, обновление, удаление) в базе данных.

2.5. Чем отличаются триггеры INSTEAD OF и AFTER?

INSTEAD OF заменяет действие, выполняясь вместо него. AFTER выполняется после завершения действия.

3. Вывод

3.1. В ходе лабораторной работы научились создавать и использовать триггеры в MS SQL Server;

Лабораторная работа №10
Реализация доступа пользователей к базе данных

1. Цель работы

1.1. Научиться использовать системные хранимые процедуры и DDL-команды для управления именами входа и пользователями БД в СУБД;

1.2. Научиться назначать привилегии пользователю БД;

1.3. Закрепить навык создания объектов БД.

2. Контрольные вопросы

2.1. В чем отличие между именами входа и пользователями БД?

Имена входа используются для аутентификации на уровне сервера СУБД. Пользователи БД – для авторизации и управления доступом внутри конкретных баз данных.

2.2. Как идентифицируются пользователи в MS SQL Server?

С помощью логинов.

2.3. На какие уровни разделяется система безопасности MS SQL Server?

Уровень сервера, уровень базы данных.

2.4. Каково назначение ролей сервера?

Используются для управления правами и предоставления доступа пользователей к административным функциям на уровне сервера.

2.5. Каково назначение ролей БД?

Роли БД обеспечивают управление правами пользователей внутри конкретной БД, обеспечивая доступ к ее объектам и операциям

3. Вывод

3.1. В ходе лабораторной работы научились использовать системные хранимые процедуры и DDL-команды для управления именами входа и пользователями БД в СУБД, а также научились назначать привилегии пользователю БД

Лабораторная работа №11
Выполнение резервного копирования и восстановления БД

1. Цель работы

1.1. Научиться выполнять резервное копирование БД;

1.2. Научиться сохранять во внешних файлах описание структуры и данные БД;

1.3. Научиться выполнять восстановление БД

2. Контрольные вопросы

2.1. Для чего создаются резервные копии БД?

Для восстановления данных при сбоях или повреждениях, защиты от потери данных, миграции данных.

2.2. В чем отличие между полным и разностным резервным копированием?

Полное резервное копирование: создаёт полную копию всех данных в БД,

2.3. Как часто должен выполняться каждый из видов резервного копирования БД (привести пример расписания)?

Полное резервное копирование – еженедельно (например, по субботам), разностное резервное копирование – Ежедневно (например, по будням).

2.4. Какие скрипты можно сформировать для объектов БД?

Скрипт создания объекта (CREATE)

Скрипт изменения объекта (ALTER)

Скрипт удаления объекта (DROP)

Скрипт резервного копирования объекта (BACKUP)

Скрипт восстановления объекта из резервной копии (RESTORE)

2.5. Как выполнить восстановление БД из резервной копии?

Остановить работу с базой данных > Выполнить команду RESTORE DATABASE из резервной копии > Указать путь к файлам резервной копии > Задать параметры восстановления (например, время восстановления) > Дождаться завершения процесса восстановления.

2.6. Какая команда выполняет восстановление БД из резервной копии?

RESTORE DATABASE

2.7. В каком порядке надо восстанавливать резервные копии?

1 Полная резервная копия

2 Разностные (инкрементные) резервные копии

2.8. Какой параметр у команды восстановления данных отключает/запускает восстановление БД?

Параметр RECOVERY

2.9. Для чего используется мастер импорта и экспорта?

Для перемещения данных между различными источниками и назначениями, экспорта данных из базы данных в файлы, Импорта данных из файлов в базу данных.

3. Вывод

3.1.В ходе лабораторной работы выполнили резервное копирование БД, сохранили во внешних файлах описание структуры и данные БД, а также выполнили восстановление БД

Лабораторная работа №12
Экспорт данных базы в документы пользователя

1. Цель работы

1.1. Научиться выполнять экспорт данных из БД.

2. Контрольные вопросы

2.1. Как выполнить экспорт csv-файла в Management Studio?

Запустить Мастер импорта и экспорта > Выбрать "Экспорт в плоский файл" > Указать CSV-файл в качестве назначения > Настроить параметры экспорта (разделители, кодировку и т.д.) > Выполнить экспорт.

2.2. Как выполнить экспорт xlsx-файла в Management Studio?

Запустить Мастер импорта и экспорта > Выбрать "Экспорт в Microsoft Excel" > Указать XLSX-файл в качестве назначения > Настроить параметры экспорта > Выполнить экспорт.

2.3. Как в Excel открыть файл формата txt/csv и xml?

TXT/CSV: В Excel выбрать "Данные" > "Из текста/CSV" > Указать файл и настроить параметры импорта > Открыть файл в Excel.

XML: В Excel выбрать "Данные" > "Из XML" > Указать файл XML > Excel импортирует данные из XML-файла.

2.4. Данные из скольких таблиц могут храниться в файле csv?

Данные из 1 таблицы

2.5. Как экспортировать данные в файл формата JSON?

Использовать Мастер импорта/экспорта в MSSQL > Выбрать "Экспорт в плоский файл" и указать формат JSON > Настроить параметры экспорта > Выполнить экспорт.

2.6. Как экспортировать данные в файл формата XML?

Использовать Мастер импорта/экспорта в MSSQL > Выбрать "Экспорт в плоский файл" и указать формат XML > Настроить параметры экспорта > Выполнить экспорт.

3. Вывод

3.1. В ходе лабораторной работы выполнили экспорт данных из БД.

Лабораторная работа №13
Импорт данных пользователя в базу данных

1. Цель работы

1.1. Научиться выполнять импорт данных в БД.

2. Контрольные вопросы

2.1. Как выполнить импорт xlsx-файла в Management Studio?

Использовать Мастер импорта/экспорта в MSSQL > Выбрать "Импорт из плоского файла" > указать XLSX-файл в качестве источника > настроить параметры импорта > выполнить импорт.

2.2. Как выполнить импорт csv-файла в Management Studio?

Использовать Мастер импорта/экспорта в MSSQL > Выбрать "Импорт из плоского файла" > указать csv-файл в качестве источника > настроить параметры импорта > выполнить импорт.

2.3. Как выполнить импорт файла XML?

Использовать Мастер импорта/экспорта в MSSQL > Выбрать "Импорт из плоского файла" и указать XML-файл > Настроить параметры импорта, например, сопоставление XML-структуры с таблицами > выполнить импорт.

2.4. Как выполнить импорт файла JSON?

Использовать Мастер импорта/экспорта в MSSQL > Выбрать "Импорт из плоского файла" и указать JSON-файл > Настроить параметры импорта, например, сопоставление JSON-структуры с таблицами > выполнить импорт.

3. Вывод

3.1. В ходе лабораторной работы выполнили импорт данных в БД.

Лабораторная работа №14
Создание слоя доступа к данным БД

1. Цель работы

1.1. Научиться создавать приложение C# для организации доступа к БД.

1.2. Изучить свойства и методы компонентов SqlConnection, SqlCommand, SqlDataReader, научиться их применять и настраивать;

2. Контрольные вопросы

2.1. Какое назначение у элемента SqlConnection?

Предназначен для установления соединения с БД SQL Server.

2.2. Какое назначение у элемента SqlCommand?

Предназначен для выполнения SQL-запросов и хранимых процедур в БД SQL Server.

2.3. Какое назначение у элемента SqlDataReader?

Предназначен для чтения результатов SQL-запросов из БД.

2.4. Какие пространства имен требуется подключить для реализации подключения к СУБД MS SQL Server, MySQL соответственно?

Для подключения к MS SQL Server: System.Data.SqlClient

Для подключения к MySQL: MySql.Data.MySqlClient

2.5. Какие методы класса SqlCommand позволяют выполнить SQL-запрос?

ExecuteReader() - для выполнения запросов, возвращающих набор данных

ExecuteNonQuery() - для выполнения запросов, не возвращающих данные

ExecuteScalar() - для выполнения запросов, возвращающих одно значение

3. Вывод

3.1. В ходе лабораторной работы создали приложение C# для организации доступа к БД, а также изучили свойства и методы компонентов SqlConnection, SqlCommand, SqlDataReader, научиться их применять и настраивать;

Лабораторная работа №15
Применение ORM для доступа к данным БД

1. Цель работы

1.1. Научиться создавать приложение C# для организации доступа к БД.

1.2. Научиться проектировать приложение, использующее паттерн репозиторий и Dapper.

2. Контрольные вопросы

2.1. Что такое Dapper и для чего используется?

Dapper — это микро-ORM (Object-Relational Mapping) библиотека для .NET, которая используется для упрощения работы с базами данных.

2.2. Какие методы Dapper позволяют извлечь данные из БД?

1) Query() - для выполнения SQL-запросов, возвращающих набор данных

2) QueryFirst() / QueryFirstOrDefault() - для получения первой строки результата

3) QuerySingle() / QuerySingleOrDefault() - для получения единственной строки результата

2.3. Какие методы Dapper позволяют изменить данные в БД?

1) Execute() - для выполнения SQL-запросов, не возвращающих данные (INSERT, UPDATE, DELETE)

ExecuteScalar() - для выполнения запросов, возвращающих одно значение

2.4. Зачем используется паттерн «репозиторий»?

используется для: абстрагирования доступа к данным, Инкапсуляции логики работы с базой данных,Обеспечения единого интерфейса для доступа к данным.

3. Вывод

3.1. В ходе лабораторной работы создали приложение C# для организации доступа к БД, а также спроектировали приложение, использующее паттерн репозиторий и Dapper.

Лабораторная работа №16
Применение ORM EF Core

1. Цель работы

1.1. Научиться создавать приложение C# для организации доступа к БД.
1.2. Научиться разрабатывать приложение, использующее EF Core.

2. Контрольные вопросы

2.1. Что такое «ORM»?

Технология, которая позволяет отображать объекты приложения на таблицы базы данных.

2.2. Что такое «EF Core» и для чего он предназначен?

Объектно-реляционный маппер (ORM) для .NET, предназначенный для работы с базами данных.

2.3. Как получить данные из БД, используя EF Core?

Определить DbContext и DbSet для нужных сущностей, Использовать методы DbSet, такие как Find(), ToList(), FirstOrDefault() и т.д.

2.4. Как выполнить редактирование (вставку, обновление, удаление данных), используя EF Core?

Получить нужную сущность из контекста > Изменить данные сущности > Вызвать метод SaveChanges() для сохранения изменений в базе данных.

3. Вывод

3.1. В ходе лабораторной работы создали приложение C# для организации доступа к БД, а также разработали приложение, использующее EF Core.

Лабораторная работа №17
Разработка приложения для фильтрации, поиска и сортировки данных

1. Цель работы

1.1. Научиться создавать приложение для доступа к БД средствами EF Core.

1.2. Научиться выполнять сортировку, фильтрацию и постраничный вывод данных, используя LINQ-запросы.

2. Контрольные вопросы

2.1. Для чего используются метод OrderBy, OrderByDescending, ThenBy, ThenByDescending и в чем их отличие?

Эти методы используются для сортировки данных в LINQ-запросах

2.2. Для чего используются методы Take и Skip и как они применяются при пагинации?

Методы Take и Skip используются для реализации пагинации в LINQ-запросах

2.3. Для чего используется метод Where?

Метод Where используется для фильтрации данных в LINQ-запросах.

2.4. Какие логические операторы могут применяться при составлении условий?

При составлении условий в методе Where могут использоваться следующие логические операторы:

&& (and)
|| (or)
! (not)

2.5. Как проверить, что значение есть в списке?

можно использовать оператор IN

2.6. Как проверить, что строка начинается с определенного текста?

можно использовать оператор LIKE

2.7. Как проверить, что строка содержит определенный текст?

можно использовать оператор LIKE

2.8. Как указать в Select список требуемых данных?

Пример:

```
SELECT column1, column2, column3
 FROM your_table;
```

3. Вывод

3.1. В ходе лабораторной работы создали приложение для доступа к БД средствами EF Core, а также выполнили сортировку, фильтрацию и постраничный вывод данных, используя LINQ-запросы.

Лабораторная работа №18
Разработка приложения для редактирования данных

1. Цель работы

1.1. Научиться выполнять вставку, обновление и удаление записей средствами EF Core;

1.2. Научиться обеспечивать обратную связь при редактировании данных.

2. Контрольные вопросы

2.1. Для чего используются методы Add() и AddRange() в EF Core?

Они используются для добавления новых сущностей в контекст базы данных

2.2. Для чего используются методы Update() в EF Core?

Он используется для обновления существующей сущности в контексте базы данных.

2.3. Для чего используются методы Remove() и RemoveRange() в EF Core?

Они используются для удаления сущностей из контекста базы данных.

2.4. Как сохранить изменения в БД, используя EF Core?

Нужно вызвать метод SaveChanges() на вашем контексте базы данных.

2.5. Как изменить значения полей объекта?

Сначала нужно получить объект из базы данных, затем измените его свойства и сохраните изменения с помощью SaveChanges().

2.6. Какое значение по умолчанию присваивается идентификатору нового объекта?

2.7. Как передать объект с одной формы на другую?

Можно передать объект через конструктор второй формы.

3. Вывод

3.1. В ходе лабораторной работы выполнили вставку, обновление и удаление записей средствами EF Core, а также обеспечили обратную связь при редактировании данных.

Лабораторная работа №19
Выполнение SQL-команд и SQL-подпрограмм

1. Цель работы

1.1. Научиться выполнять SQL-команды и вызывать хранимые процедуры и функции пользователя средствами EF Core.

2. Контрольные вопросы

2.1. Как выполнить команду на выборку данных в EF Core?

Можно использовать методы ToList(), FirstOrDefault(), SingleOrDefault()

2.2. Как выполнить команду на модификацию данных в EF Core?

Получите объект из базы данных > Измените необходимые свойства объекта > Вызовите метод SaveChanges() для сохранения изменений.

2.3. Как объявить и передать параметр в SQL-команду в EF Core?

Ответ:Можно использовать метод FromSqlRaw() или ExecuteSqlRaw() для выполнения необработанных SQL-запросов с параметрами.

2.4. Как вызвать табличную функцию в EF Core?

Пример:

Создайте класс для представления результата функции:

```
public class MyResult
{
   public int Id { get; set; }
   public string Name { get; set; }
}
```

Вызовите табличную функцию:

```
using (var context = new YourDbContext())
{
   var results = context.Set<MyResult > ()
                  .FromSqlRaw("SELECT * FROM dbo.MyTableFunction(@param)", new SqlParameter("@param", paramValue))
                  .ToList();
}
```

2.5. Как вызвать хранимую процедуру в EF Core?

Пример:

Для выборки данных:

```
using (var context = new YourDbContext())
{
   var results = context.MyEntities
                  .FromSqlRaw("EXEC dbo.MyStoredProcedure @param1, @param2",
                              new SqlParameter("@param1", value1),
                              new SqlParameter("@param2", value2))
                  .ToList();
```

```
}
```

Для выполнения хранимой процедуры без возврата данных:

```
using (var context = new YourDbContext())
{
    context.Database.ExecuteSqlRaw("EXEC dbo.MyStoredProcedure @param1, @param2",
                                   new SqlParameter("@param1", value1),
                                   new SqlParameter("@param2", value2));
}
```

2.6. Как объявить и передать выходной параметр из хранимой процедуры в EF Core?

Объявите выходной параметр и вызовите хранимую процедуру:

```
using (var context = new YourDbContext())
{
    // Объявление выходного параметра
    var outputParam = new SqlParameter
    {
        ParameterName = "@outputParam",
        SqlDbType = System.Data.SqlDbType.Int, // Укажите тип данных
        Direction = System.Data.ParameterDirection.Output
    };

    // Вызов хранимой процедуры
    context.Database.ExecuteSqlRaw("EXEC dbo.MyStoredProcedure @inputParam, @outputParam OUT",
                                   new SqlParameter("@inputParam", inputValue),
                                   outputParam);

    // Получение значения выходного параметра
    var result = (int)outputParam.Value;
}
```

3. Вывод

3.1. В ходе лабораторной работы выполнили SQL-команды и вызывать хранимые процедуры и функции пользователя средствами EF Core.

Лабораторная работа №20
Реализация разграничения прав доступа пользователей

1.Цель работы

1.1. Научиться разграничивать права доступа пользователей на уровне интерфейса приложения;

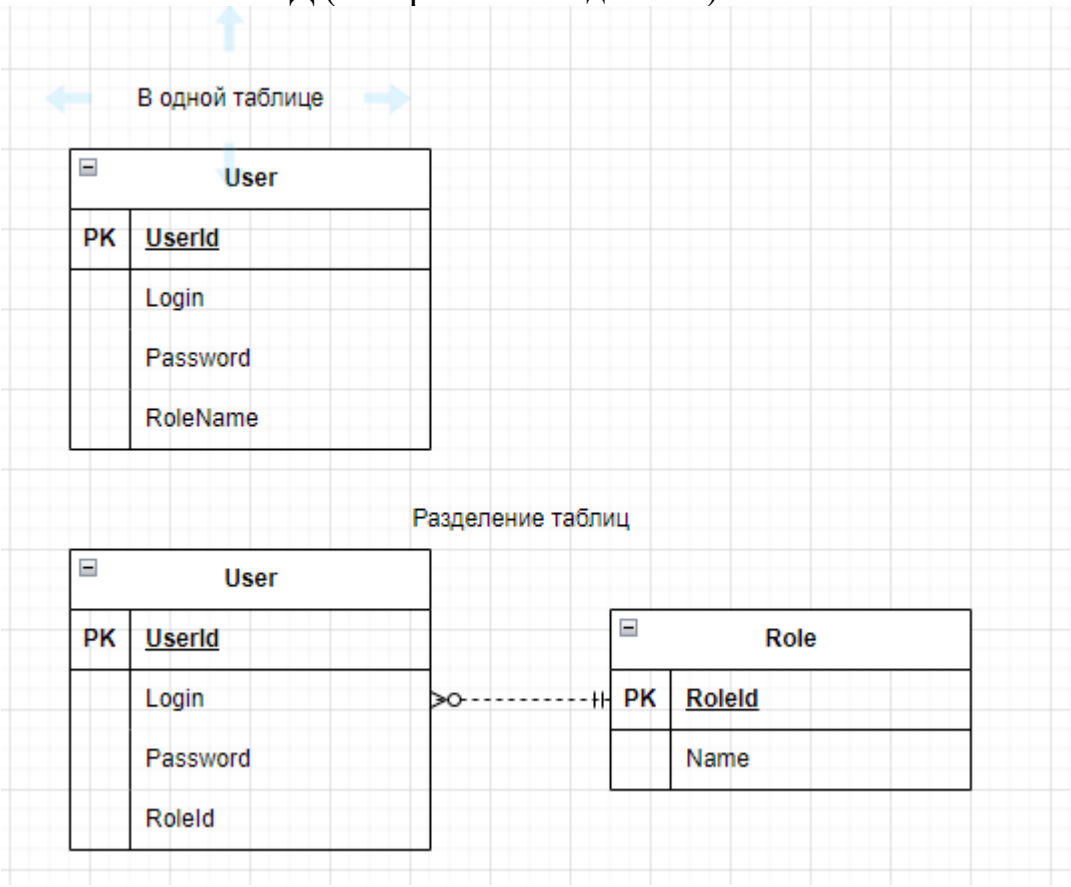
1.2. Научиться изменять настройки подключения к БД средствами Entity Framework Core.

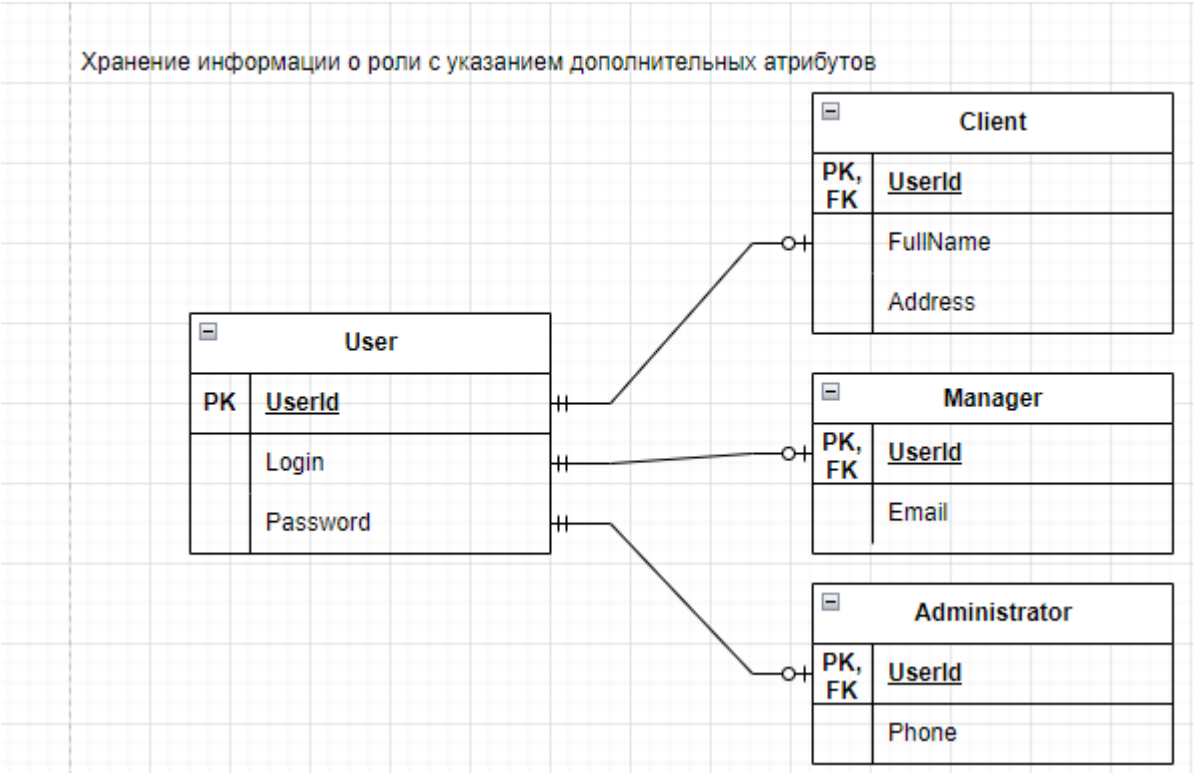
2. Контрольные вопросы

2.1.Как изменить настройки подключения к БД в клиентском приложении?

Изменить строку подключения в файле конфигурации > Обновите контекст базы данных > Настройте сервисы в Startup.cs

2.2.Какими способами можно обеспечить хранение пользователей и ролей пользователей в БД (отобразить в виде ERD)?





2.3. Что такое «авторизация»?

Авторизация — это процесс, который определяет, имеет ли пользователь право на доступ к определенным ресурсам или выполнению определенных действий в системе после того, как его личность была подтверждена (аутентификация).

2.4. Что такое «регистрация»?

Регистрация — это процесс создания учетной записи пользователя в системе или приложении. Во время регистрации пользователи предоставляют свои данные, такие как имя, адрес электронной почты, пароль и другие необходимые сведения.

3. Вывод

3.1. В ходе лабораторной работы разграничили права доступа пользователей на уровне интерфейса приложения, а также изменили настройки подключения к БД средствами Entity Framework Core.

Лабораторная работа №21
Разработка приложения для импорта данных

1. Цель работы

1.1. Импорт данных пользователя в базу данных средствами EF Core.

2. Контрольные вопросы

2.1. Как выполнить импорт из csv-файла в БД через приложение?

Чтение CSV-файла:

```
using (var reader = new StreamReader("path/to/file.csv"))
using (var csv = new CsvReader(reader, CultureInfo.InvariantCulture))
{
   var records = csv.GetRecords<YourEntity > ().ToList();
}
```

Подключение к базе данных:

Используйте Entity Framework Core или ADO.NET для подключения к базе данных

Сохранение данных в БД:

```
using (var context = new YourDbContext())
{
   context.YourEntities.AddRange(records);
   context.SaveChanges();
}
```

2.2. Как импортировать изображение в БД через приложение?

Получите изображение:

Используйте диалоговое окно для выбора файла или загрузите изображение через интерфейс приложения.

Чтение изображения в массив байтов:

Прочитайте файл изображения в массив байтов.

```
byte[] imageData;
using (var fs = new FileStream("path/to/image.jpg", FileMode.Open, FileAccess.Read))
{
   using (var br = new BinaryReader(fs))
   {
      imageData = br.ReadBytes((int)fs.Length);
   }
}
```

Сохранение изображения в БД:

```
using (var context = new YourDbContext())
{
```

```
    var entity = new YourEntity
    {
        ImageData = imageData // Поле для хранения изображения в БД
    };
    context.YourEntities.Add(entity);
    context.SaveChanges();
}
```

2.3. Как выполнить импорт файла JSON в БД через приложение?

Чтение JSON-файла:

Используйте библиотеку System.Text.Json или Newtonsoft.Json для чтения и десериализации JSON.

```
var jsonString = File.ReadAllText("path/to/file.json");
var records = JsonSerializer.Deserialize<List<YourEntity >  > (jsonString);
```

Подключение к базе данных:

Используйте Entity Framework Core или ADO.NET для подключения к базе данных.

Сохранение данных в БД:

Добавьте десериализованные записи в контекст базы данных и сохраните изменения.

```
using (var context = new YourDbContext())
{
    context.YourEntities.AddRange(records);
    context.SaveChanges();
}
```

3.Вывод

3.1. В ходе лабораторной работы импортировали данные пользователя в базу данных средствами EF Core.

Лабораторная работа №22
Разработка приложения для экспорта данных в текстовые файлы

1. Цель работы

1.1. Научиться выполнять экспорт данных из БД в формате текстового файла.

1.2. Научиться выполнять экспорт данных из БД с применением Microsoft Office Word.

2. Контрольные вопросы

2.1. Какое пространство имен требуется подключить для работы с Word?

Необходимо подключить пространство имен Microsoft.Office.Interop.Word

2.2. Какие библиотеки позволяют сохранять данные в формате docx?

DocX, Open XML SDK, NPOI

2.3. Как выполнить экспорт данных в формате txt?

Получите данные: Извлеките данные, которые вы хотите экспортировать, например, из базы данных.

Запишите данные в файл TXT: Используйте StreamWriter для записи данных в текстовый файл.

```
using System.IO;

// Получение данных (например, из базы данных)
var data = new List<string >  { "Line 1", "Line 2", "Line 3" };

// Запись данных в файл TXT
using (var writer = new StreamWriter("path/to/file.txt"))
{
   foreach (var line in data)
   {
      writer.WriteLine(line);
   }
}
```

3. Вывод

3.1. В ходе лабораторной работы выполнили экспорт данных из БД в формате текстового файла, а также выполнили экспорт данных из БД с применением Microsoft Office Word.

Лабораторная работа №23
Разработка приложения для экспорта данных в табличные файлы

1. Цель работы

1.1. Научиться выполнять экспорт данных из БД в формате табличного файла.

1.2. Научиться выполнять экспорт данных из БД с применением Microsoft Office Excel.

2. Контрольные вопросы

2.1. Какое пространство имен требуется подключить для работы с Word?

Необходимо подключить пространство имен Microsoft.Office.Interop.Word

2.2. Какие библиотеки позволяют сохранять данные в формате docx?

DocX, Open XML SDK, NPOI

2.3. Как выполнить экспорт данных в формате txt?

Получите данные: Извлеките данные, которые вы хотите экспортировать, например, из базы данных.

Запишите данные в файл TXT: Используйте StreamWriter для записи данных в текстовый файл.

```
using System.IO;

// Получение данных (например, из базы данных)
var data = new List<string >  { "Line 1", "Line 2", "Line 3" };

// Запись данных в файл TXT
using (var writer = new StreamWriter("path/to/file.txt"))
{
   foreach (var line in data)
   {
      writer.WriteLine(line);
   }
}
```

3. Вывод

3.1. В ходе лабораторной работы выполнили экспорт данных из БД в формате табличного файла, а также выполнили экспорт данных из БД с применением Microsoft Office Excel.

Практическая работа №1
Разработка web-API для доступа к данным

1. Цель работы

1.1. Научиться выполнять разработку web-API для доступа к БД.

2. Контрольные вопросы

2.1. Что такое REST-запрос?

запрос, отправляемый клиентом на сервер в рамках архитектурного стиля REST (Representational State Transfer).

2.2. Что такое RESTful?

это термин, описывающий веб-сервисы, которые следуют принципам архитектурного стиля REST (Representational State Transfer).

2.3. Для чего используется метод GET?

Метод GET используется в HTTP-запросах для получения данных с сервера.

2.4. Для чего используется метод POST?

Метод POST используется в HTTP-запросах для отправки данных на сервер.

2.5. Для чего используется метод PUT?

Метод PUT используется в HTTP-запросах для обновления существующего ресурса на сервере или для создания нового ресурса, если он не существует.

2.6. Для чего используется метод DELETE?

Метод DELETE используется в HTTP-запросах для удаления существующего ресурса на сервере.

3. Вывод

3.1. В ходе практической работы выполнять разработку web-API для доступа к БД.

Практическая работа №2
Вызов методов REST API

1. Цель работы

1.1. Научиться проверять работоспособность RESTful API в клиентском приложении.

1.2. Научиться выполнять тестирование RESTful API методом черного ящика.

2. Контрольные вопросы

2.1. Как указать у объекта HttpClient базовый адрес?

использовать свойство BaseAddress:

```
using System;
using System.Net.Http;

class Program
{
   static void Main()
   {
      var client = new HttpClient
      {
         BaseAddress = new Uri("https://api.example.com/")
      };

      // Теперь можно использовать client для отправки запросов
      var response = client.GetAsync("endpoint").Result; // Запрос к https://api.example.com/endpoint
   }
}
```

2.2. Какие коды ответа могут быть получены при Http-запросе (указать коды и значения)?

Информационные:

100 Continue: Сервер получил начальные заголовки запроса и клиент может продолжать отправку тела запроса.

101 Switching Protocols: Сервер согласен переключить протоколы.

Успех

200 OK: Запрос успешно выполнен.

201 Created: Запрос выполнен, и ресурс был создан.

204 No Content: Запрос выполнен, но нет содержимого для возврата.

Перенаправление

301 Moved Permanently: Запрашиваемый ресурс был перемещен на новый URL.

302 Found: Запрашиваемый ресурс временно доступен по другому URL.

304 Not Modified: Ресурс не изменялся с момента последнего запроса.

Ошибки клиента

400 Bad Request: Запрос не может быть обработан из-за неверного синтаксиса.

401 Unauthorized: Необходима аутентификация для доступа к ресурсу.

403 Forbidden: Доступ к ресурсу запрещен.

404 Not Found: Запрашиваемый ресурс не найден.

Ошибки сервера

500 Internal Server Error: Внутренняя ошибка сервера.

502 Bad Gateway: Сервер, действующий как шлюз, получил недопустимый ответ от вышестоящего сервера.

503 Service Unavailable: Сервер временно недоступен, обычно из-за перегрузки или технического обслуживания.

2.3. Какой метод класса HttpClient используется для получения данных?

Метод класса HttpClient, используемый для получения данных, — это GetAsync

2.4. Какой метод класса HttpClient используется для вставки данных?

Метод класса HttpClient, используемый для вставки данных, — это PostAsync.

2.5. Какой метод класса HttpClient используется для изменения данных?

Метод класса HttpClient, используемый для изменения данных, — это PutAsync.

2.6. Какой метод класса HttpClient используется для удаления данных?

Метод класса HttpClient, используемый для удаления данных, — это DeleteAsync

3. Вывод

3.1. В ходе практической работы проверили работоспособность RESTful API в клиентском приложении, а также выполнили тестирование RESTful API методом черного ящика.

Практическая работа №3
Разграничение прав доступа на уровне REST API

1. Цель работы

1.1. Научиться выполнять разработку web-API для доступа к БД.

1.2. Научиться работать с JWT.

2. Контрольные вопросы

2.1. Какие атрибуты можно указать у методов REST для настройки доступа для авторизованных и неавторизованных пользователей?

[Authorize], [AllowAnonymous], [Authorize(Roles = "Admin")]

2.2. Для чего используется JWT?

Используется для безопасной передачи информации между сторонами в виде JSON-объекта.

2.3. В чем отличие между авторизацией с использованием cookie и с использованием JWT с точки зрения безопасности?

Хранение данных:

Cookie: хранит сессионные данные на стороне клиента. Сервер создает сессию и хранит информацию о пользователе на сервере. Cookie содержит идентификатор сессии.

JWT: хранит все необходимые данные (например, идентификатор пользователя, роли) в самом токене, который подписан и может быть проверен на сервере. Токен передается клиентом при каждом запросе.

Статус сессии:

Cookie: Сессия является состоянием, и сервер должен отслеживать активные сессии. Это может привести к проблемам с масштабируемостью.

JWT: без состояния. Сервер не хранит информацию о сессии, что упрощает масштабирование, но может усложнить отзыв токенов.

Уязвимости:

Cookie: Уязвимы к атакам CSRF (Cross-Site Request Forgery), если не используются соответствующие меры защиты (например, SameSite атрибут).

JWT: Уязвимы к атакам XSS (Cross-Site Scripting), если токен хранится в локальном хранилище. Также могут быть уязвимы к атакам на подделку токенов, если не используется надежная подпись.

2.4. Какие настройки можно указать при создании токена?

Заголовок (Header), Полезная нагрузка (Payload), Секретный ключ (Secret Key), Срок действия (Expiration):

2.5. Почему для авторизации следует использовать метод POST?

Безопасность данных: Данные, такие как имя пользователя и пароль, передаются в теле запроса, что делает их менее видимыми в URL, в отличие от метода GET, где параметры передаются в строке запроса.

Изменение состояния: POST предназначен для создания или изменения состояния на сервере. При авторизации сервер проверяет учетные данные и может создавать сессию или токен, что соответствует семантике POST.

3. Вывод

3.1.В ходе практической работы выполнили разработку web-API для доступа к БД, а также работали с JWT.

Практическая работа №4
Разработка веб-клиента

1. Цель работы

1.1. Научиться разрабатывать клиентское веб-приложение для доступа к БД.

1.2. Научиться проверять работоспособность RESTful API в клиентском приложении.

2. Контрольные вопросы

2.1. Как выполнить HTTP-запрос к API с помощью Fetch API?

Чтобы выполнить HTTP-запрос к API с помощью Fetch API, используйте функцию fetch()

2.2. Как обрабатывать ошибки при вызове API в JavaScript?

Для обработки ошибок при вызове API в JavaScript с использованием Fetch API можно использовать блоки then() и catch()

2.3. Что такое асинхронные функции и как они связаны с вызовом API?

Асинхронные функции в JavaScript — это функции, которые позволяют выполнять асинхронные операции, не блокируя основной поток выполнения. Они объявляются с помощью ключевого слова async и могут использовать оператор await для ожидания завершения промисов.

2.4. Что такое Razor Pages?

Razor Pages — это фреймворк для создания веб-приложений на платформе ASP.NET Core, который упрощает разработку страниц с динамическим контентом. Он основан на паттерне "страница как контроллер" и использует синтаксис Razor для генерации HTML.

3. Вывод

3.1. В ходе практической работы разработали клиентское веб-приложение для доступа к БД, а также проверили работоспособность RESTful API в клиентском приложении.