**АРХАНГЕЛЬСКИЙ КОЛЛЕДЖ ТЕЛЕКОММУНИКАЦИЙ (ФИЛИАЛ) СПбГУТ**

**(АКТ (ф) СПбГУТ)**

**Отчеты по лабораторным и практическим работам**

**по МДК.11.01**

Студент: Группа: ИСПП-21 И.С. Погребной

Преподаватель: Ю.С. Маломан

**Архангельск 2024**

**Лабораторная работа №1**

**Сбор и анализ требований методом use-case**

* **Цель работы**
* Изучить процесс описания требований к системе методом use-case.
* Изучить процесс создания диаграммы вариантов использования.
* **Ответы на контрольные вопросы**
* Для чего используется диаграмма вариантов использования?

Ответ: диаграмма вариантов использования необходима для:

* Моделирования функциональности системы с точки зрения её пользователей (акторов).
* Отражения взаимодействий между пользователями (актерами) и системой.
* Представления основных функций системы в виде прецедентов (вариантов использования).
* Помощи в анализе требований, планировании и проектировании системы.
* Что такое «актор» и как он обозначается на диаграмме вариантов использования?

Ответ: Актор (Actor) — это внешняя сущность, которая взаимодействует с системой, чтобы достичь определённой цели. Актор может быть:

* Человеком.
* Другой системой.
* Устройством.

Обозначение: актор изображается в виде человечка (стилизованной палочки) с подписью его роли или названия.

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

Ответ: Прецедент (Use Case) — это конкретный вариант использования системы, описывающий, как актор взаимодействует с системой для выполнения задачи.

Обозначение: прецедент изображается в виде овала с названием действия (глагол + существительное, например, «Создать заказ»).

* Что обозначает «отношение ассоциации»?

Ответ: Отношение ассоциации показывает взаимодействие между актором и прецедентом. Оно обозначает, что актор может использовать данный вариант использования.

Обозначение: линия, соединяющая актера и прецедент;

* Что обозначает «отношение обобщения»?

Ответ: Отношение обобщения показывает наследование между:

* Акторами (например, специализированный актор наследует поведение от базового актора).
* Прецедентами (один вариант использования наследует поведение другого).

Обозначение: стрелка с пустым треугольником на конце, указывающим на более общий элемент.

* Что обозначает «отношение включения»?

Ответ: Отношение включения (<<include>>) указывает, что один прецедент обязательно включает выполнение другого. Это используется для выделения общего функционала.

Обозначение: пунктирная стрелка с меткой <<include>>, указывающая на включаемый прецедент.

* Что обозначает «отношение расширения»?

Ответ: Отношение расширения (<<extend>>) указывает, что один прецедент может быть дополнен другим при выполнении определённых условий. Используется для описания необязательного или дополнительного поведения.

Обозначение: пунктирная стрелка с меткой <<extend>>, указывающая на прецедент, который расширяется.

* **Вывод**
* В ходе лабораторной работы изучен процесс описания требований к системе методом use-case и создания диаграммы вариантов использования.

**Лабораторная работа №2**

**Проектирование реляционной схемы базы данных в среде СУБД**

* **Цель работы**
* Научиться применять MySQL Workbench в процессе создания схем моделей БД;
* Научиться представлять логическую модель данных согласно нотациям ERD и IDEF1X.
* **Ответы на контрольные вопросы**
* Что такое «сущность»?

Ответ: Сущность (Entity) — это объект или предмет реального мира, информация о котором должна быть сохранена в базе данных. Сущности представляют основные элементы предметной области, например, «Сотрудник», «Товар», «Заказ».

* Что такое «атрибут»?

Ответ: Атрибут (Attribute) — это характеристика или свойство сущности. Например:

* У сущности «Сотрудник» атрибутами могут быть «Имя», «Должность», «Возраст».
* У сущности «Товар» — «Название», «Цена», «Количество на складе».
* Что такое «ключевое поле»?

Ответ: Ключевое поле — это атрибут или набор атрибутов сущности, который однозначно идентифицирует каждую запись в таблице. Оно используется для обеспечения уникальности данных.

* Каково назначение первичных и внешних ключей?

Ответ:

Первичный ключ (Primary Key):

* Уникально идентифицирует каждую запись в таблице.
* Не может быть пустым (NULL).
* Пример: поле ID в таблице «Сотрудник».

Внешний ключ (Foreign Key):

* Связывает таблицу с другой таблицей через первичный ключ.
* Обеспечивает целостность данных.
* Пример: поле EmployeeID в таблице «Заказ» может быть внешним ключом, связанным с таблицей «Сотрудник».
* Что такое «связь»?

Ответ:

Связь (Relationship) — это логическая ассоциация между двумя или более сущностями. Связи показывают, как сущности взаимодействуют друг с другом. Примеры:

* «Сотрудник работает в Отделе».
* «Клиент делает Заказ».
* Какие виды связей между сущностями существуют?

Ответ:

"Один-к-одному" (1:1): Каждой записи в одной сущности соответствует одна запись в другой.

* Пример: «Пользователь» и его «Паспорт».

"Один-ко-многим" (1:N): Каждой записи в одной сущности соответствует несколько записей в другой.

* Пример: «Отдел» и «Сотрудники».

"Многие-ко-многим" (M:N): Несколько записей одной сущности могут быть связаны с несколькими записями другой.

* Пример: «Студенты» и «Курсы» (один студент может учиться на нескольких курсах, и курс может посещать несколько студентов).
* Какие элементы входят в ER-диаграммы?

Ответ:

Элементы ER-диаграммы (диаграммы "сущность-связь") включают:

* Сущности — прямоугольники.
* Атрибуты — овалы, связанные с сущностями или связями.
* Ключевые атрибуты — подчёркиваются или отмечаются.
* Связи — ромбы, соединяющие сущности.
* Мощности связей (кардинальность): отображают тип связи (1:1, 1:N, M:N).
* Для чего применяются ER-диаграммы?

Ответ:

ER-диаграммы используются для:

* Анализа и проектирования структуры базы данных.
* Визуализации данных и их связей.
* Выявления ключевых сущностей и атрибутов.
* Обеспечения целостности данных.
* Улучшения взаимодействия между разработчиками и бизнес-аналитиками при разработке систем.
* **Вывод**
* В ходе лабораторной работы были приобретены навыки применять MySQL Workbench в процессе создания схем моделей БД и представлять логическую модель данных согласно нотациям ERD и IDEF1X.

**Лабораторная работа №3**

**Приведение БД к нормальной форме**

* **Цель работы**
* Изучить процесс приведения отношений от ненормализованного вида к четвертой нормальной форме;
* Изучить процесс декомпозии отношений.
* **Ответы на контрольные вопросы**
* Что называется первичным ключом отношения?

Ответ: Первичный ключ (Primary Key) — это атрибут (или набор атрибутов) таблицы, который однозначно идентифицирует каждую запись в отношении (таблице).  
Требования к первичному ключу:

* Значение ключа должно быть уникальным для каждой строки.
* Значение ключа не может быть NULL.
* Что называется внешним ключом отношения?

Ответ: Внешний ключ (Foreign Key) — это атрибут (или набор атрибутов) в одной таблице, который ссылается на первичный ключ в другой таблице.  
Назначение: устанавливает связь между таблицами и обеспечивает целостность данных.

* В чем заключается процесс нормализации отношений?

Ответ: Нормализация отношений — это процесс организации структуры базы данных для минимизации избыточности и устранения аномалий (вставки, обновления и удаления).  
Процесс включает преобразование отношений в более строгие нормальные формы (1НФ, 2НФ, 3НФ и далее). Основные цели:

* Уменьшение дублирования данных.
* Обеспечение целостности и консистентности данных.
* В каком случае атрибут А функционально зависит от атрибута В?

Ответ: Атрибут A функционально зависит от атрибута B (записывается как B→A), если значение B однозначно определяет значение A.

* В каком случае атрибут А транзитивно зависит от атрибута В?

Ответ: Атрибут A транзитивно зависит от атрибута BBB, если:

* B→C (атрибут B определяет C).
* C→A (атрибут C определяет A).
* Следовательно, B→A через C.
* Каким требованиям должно отвечать отношение, находящееся в 1НФ?

Ответ: Для выполнения первой нормальной формы (1НФ) отношение должно:

* Иметь уникальные строки (каждая строка идентифицируется первичным ключом).
* Содержать только атомарные значения (каждое поле должно содержать одно значение, а не списки или массивы).
* Не иметь повторяющихся групп данных.
* Каким требованиям должно отвечать отношение, находящееся во 2НФ?

Ответ: Для выполнения второй нормальной формы (2НФ) отношение должно:

* Соответствовать требованиям 1НФ.
* Устранить частичные функциональные зависимости, то есть:

Все неключевые атрибуты должны быть полностью функционально зависимы от всего первичного ключа, а не от его части (в случае составного ключа).

* Каким требованиям должно отвечать отношение, находящееся в 3НФ?

Ответ: Для выполнения третьей нормальной формы (3НФ) отношение должно:

* Соответствовать требованиям 2НФ.
* Исключить транзитивные зависимости, то есть:

Неключевые атрибуты не должны зависеть друг от друга транзитивно через другие неключевые атрибуты.

* **Вывод**
* В ходе лабораторной работы изучен процесс приведения отношений от ненормализованного вида к четвертой нормальной форме и декомпозии отношений.

**Лабораторная работа №4**

**Установка SQL сервера**

* **Цель работы**
* Научиться устанавливать разные СУБД, используя docker.
* **Ответы на контрольные вопросы**
* Какие стандартные порты для подключения к СУБД MSSQL, MySQL, Postgres?

Ответ:

* MSSQL (Microsoft SQL Server): порт 1433.
* MySQL: порт 3306.
* PostgreSQL: порт 5432.
* Какие стандартные папки для хранения данных СУБД MSSQL, MySQL, Postgres?

Ответ:

* MSSQL:
* Папка для хранения данных по умолчанию:  
  C:\Program Files\Microsoft SQL Server\MSSQL{Version}.MSSQLSERVER\MSSQL\DATA\.
* MySQL:
* Папка для хранения данных по умолчанию:
* Windows: C:\ProgramData\MySQL\MySQL Server {Version}\Data\.
* Linux: /var/lib/mysql/.
* PostgreSQL:
* Папка для хранения данных по умолчанию:
* Windows: C:\Program Files\PostgreSQL\Data\.
* Linux: /var/lib/postgresql/{Version}/main.
* Для чего используется Docker?

Ответ: Docker — это инструмент для контейнеризации приложений, который позволяет:

* Создавать, распространять и запускать контейнеры: лёгкие и изолированные окружения для запуска приложений.
* Изолировать приложения: каждое приложение работает в собственном контейнере с заданными зависимостями.
* Упрощать развертывание: приложение вместе с его окружением можно запустить на любом сервере с установленным Docker.
* Ускорять разработку и тестирование: благодаря быстрому развертыванию окружения.

Применение в СУБД:

* Развёртывание СУБД (MSSQL, MySQL, PostgreSQL) для тестирования и разработки без сложной настройки серверов.
* Лёгкое управление версиями баз данных и переносимость.
* **Вывод**
* В ходе лабораторной работы приобретены навыки устанавливать разные СУБД, используя docker.

**Лабораторная работа №5**

**Создание базы данных в среде разработки**

* **Цель работы**
* Изучить способы обеспечения целостности данных в MS SQL Server;
* Научиться работать в среде SQL Server Management Studio (SSMS).
* **Ответы на контрольные вопросы**
* Что такое SQL Server Management Studio?

Ответ: SQL Server Management Studio (SSMS) — это интегрированная среда разработки (IDE) от Microsoft для управления, настройки и разработки баз данных в Microsoft SQL Server.

* Какие виды авторизации поддерживаются в MS SQL Server?

Ответ: MS SQL Server поддерживает два вида авторизации:

* Windows Authentication (авторизация через учетные записи Windows);
* SQL Server Authentication (авторизация с помощью логина и пароля SQL Server).
* Что такое первичный ключ?

Ответ: Первичный ключ (Primary Key) — это атрибут или комбинация атрибутов, которые однозначно идентифицируют каждую запись в таблице.

* Как указать заполнение столбца автоинкрементными значениями?

Ответ: При создании столбца на нижней панели указать нужный параметр в поле спецификация идентификатора, или же при создании таблицы скриптом указать это ключевым словом.

* Как указать значение по умолчанию?

Ответ: Также как и автоинкремент, только в поле «значение по умолчанию», либо в скрипте после слова DEFAULT.

* Как задать проверочное ограничение?

Ответ: В ручную – ПКМ по столбцу и задаем проверочные ограничения, либо в скрипте после слова CHECK.

* Как обеспечить уникальность значений в столбце или наборе столбцов?

Ответ: В ручную, также как и проверочное ограничение/в скрипте при помощи слова UNIQUE.

* Что такое внешний ключ?

Ответ: Внешний ключ (foreign key) — это столбец или набор столбцов в одной таблице, который ссылается на первичный ключ другой таблицы. Он используется для обеспечения ссылочной целостности между таблицами.

* Какие значения может принимать внешний ключ?

Ответ: Внешний ключ может принимать значения, которые существуют в столбце, на который он ссылается (то есть первичном ключе другой таблицы). Если внешний ключ не имеет соответствующего значения в родительской таблице, то вставка или обновление записи с этим внешним ключом будут запрещены. Кроме того, внешний ключ может принимать значение NULL, если это допускается, что означает отсутствие ссылки на другую запись.

* **Вывод**
* В ходе лабораторной работы изучены способы обеспечения целостности данных в MS SQL Server и приобретены навыки работать в среде SQL Server Management Studio (SSMS).

**Лабораторная работа №6**

**Создание представлений в СУБД**

* **Цель работы**
* Научиться создавать и использовать представления в MS SQL Server;
* Закрепить навык создания запросов на выборку в MS SQL Server.
* **Ответы на контрольные вопросы**
* Что такое представления и чем они отличаются от таблиц?

Ответ: Представления (views) — это виртуальные таблицы, которые представляют собой результат выполнения SQL-запроса. Они не хранят данные физически, а формируются динамически на основе данных из одной или нескольких таблиц. Основные отличия от таблиц:

* Физическое хранение: Таблицы хранят данные на диске, тогда как представления не хранят данные, а лишь определяют способ их отображения.
* Обновляемость: Некоторые представления могут быть обновляемыми, но не все, тогда как таблицы всегда могут содержать данные.
* Для чего применяются представления?

Ответ: Представления используются для:

* Упрощения сложных запросов: Они позволяют скрыть сложные SQL-запросы и предоставить пользователям более простой интерфейс для работы с данными.
* Безопасности: Позволяют ограничить доступ к определённым столбцам или строкам таблиц, предоставляя пользователям только необходимые данные.
* Агрегации и сводной информации: Позволяют создавать сводные данные и представлять их в удобном формате, например, для отчетов.
* Возможно ли создание представления, включающего информацию из нескольких таблиц одновременно?

Ответ: Да, создание представления, включающего информацию из нескольких таблиц, возможно. Это достигается с помощью операций соединения (JOIN) в SQL-запросе.

* Какие требование предъявляются к обновляемым представлениям?

Ответ: Для того чтобы представление было обновляемым, оно должно соответствовать следующим требованиям:

* Оно должно ссылаться только на одну базовую таблицу (в большинстве случаев).
* Не должно содержать агрегатных функций (таких как SUM, AVG и т. д.).
* Не должно содержать DISTINCT, GROUP BY или HAVING.
* Не должно содержать подзапросов в списке SELECT.
* Не должно содержать JOIN (хотя некоторые СУБД поддерживают обновление представлений с JOIN).
* Как создать обновляемое представление?

Ответ: Чтобы создать обновляемое представление, нужно использовать простой SELECT-запрос, который соответствует требованиям, описанным в прошлом ответе.

* **Вывод**
* В ходе лабораторной работы приобретены навыки создания и использовать представления в MS SQL Server, а также закреплен навык создания запросов на выборку в MS SQL Server.

**Лабораторная работа №7**

**Создание функций пользователя в СУБД**

* **Цель работы**
* Научиться создавать и использовать скалярные и табличные функции

пользователя в MS SQL Server;

* Закрепить навык создания запросов на выборку в MS SQL Server.
* **Ответы на контрольные вопросы**
* Что такое функции пользователя и для чего они применяются?

Ответ: Функции пользователя (user-defined functions, UDF) — это функции, которые создаются пользователем в базе данных для выполнения определённых операций или вычислений. Они могут принимать параметры и возвращать значения. Функции пользователя применяются для:

* Повторного использования кода: Позволяют избежать дублирования логики в различных местах SQL-запросов.
* Упрощения сложных вычислений: Позволяют инкапсулировать сложные логические операции или преобразования данных в одну функцию.
* Улучшения читаемости кода: Делают SQL-запросы более понятными и легкими для сопровождения.
* Чем отличается скалярная функция от табличной?

Ответ: Скалярная функция: Возвращает одно значение (скаляр) и может использоваться в выражениях, например, в SELECT, WHERE и т. д. Например, функция, которая возвращает квадрат числа.

Табличная функция: Возвращает таблицу (набор строк и столбцов) и может использоваться в FROM-клаузе SQL-запроса, как если бы это была обычная таблица. Например, функция, которая возвращает список всех заказов для определённого клиента.

* Как объявить переменную и присвоить ей значение?

Ответ: В SQL для объявления переменной используется ключевое слово DECLARE. Присвоение значения выполняется с помощью оператора SET

* Как указать в функции параметр по умолчанию?

Ответ: Чтобы указать параметр по умолчанию в функции, необходимо задать значение для параметра в момент его объявления. Например:

CREATE FUNCTION example\_function (@param INT = 5).

* Как вызвать скалярную функцию?

Ответ: Скалярную функцию можно вызвать в SQL-запросе, как если бы это было встроенное выражение. Например:

SELECT dbo.example\_function(10) AS Result;

* Как вызвать табличную функцию?

Ответ: Табличную функцию можно вызвать в FROM-клаузе SQL-запроса. Например:

SELECT \* FROM dbo.example\_table\_function('some\_parameter');

* **Вывод**
* В ходе лабораторной работы приобретены навыки создания и использовать представления в MS SQL Server, а также закреплен навык создания запросов на выборку в MS SQL Server.

**Лабораторная работа №8**

**Создание хранимых процедур в СУБД**

* **Цель работы**
* Научиться создавать и использовать хранимые процедуры в MS SQL Server;
* Закрепить навык создания запросов на выборку и модификацию данных в MS SQL Server.
* **Ответы на контрольные вопросы**
* Что такое хранимые процедуры и для чего они применяются?

Ответ: Хранимые процедуры (stored procedures) — это предварительно скомпилированные наборы SQL-запросов и логики, которые хранятся в базе данных и могут быть выполнены по запросу. Они используются для:

* Упрощения выполнения сложных операций: Хранимые процедуры позволяют объединять несколько SQL-запросов в одну единицу, что упрощает выполнение сложных операций.
* Улучшения производительности: Так как хранимые процедуры компилируются и хранятся на сервере, их выполнение может быть быстрее, чем выполнение отдельных запросов.
* Безопасности: Позволяют ограничить доступ к базовым таблицам, предоставляя пользователям возможность выполнять только определённые операции через хранимые процедуры.
* Логики на стороне сервера: Позволяют реализовать бизнес-логику на стороне сервера, уменьшая нагрузку на клиентские приложения.

* Чем отличается функция пользователя от хранимой процедуры?

Ответ:

* Возвращаемое значение: Функции пользователя возвращают значение (скалярное или табличное), тогда как хранимые процедуры не возвращают значения, но могут возвращать результаты через выходные параметры или результаты выполнения запросов.
* Использование в SQL-запросах: Функции могут быть использованы в SQL-запросах, например, в SELECT, WHERE и т. д. Хранимые процедуры не могут быть использованы непосредственно в запросах, их нужно вызывать с помощью специального оператора EXECUTE.
* Количество параметров: Хранимые процедуры могут иметь входные и выходные параметры, тогда как функции обычно имеют только входные параметры.
* Как задать входные параметры хранимой процедуры?

Ответ: Входные параметры хранимой процедуры задаются в момент её объявления. Пример:

CREATE PROCEDURE example\_procedure

@param1 INT,

@param2 VARCHAR(50)

AS

BEGIN

…

END;

* Как задать выходные параметры хранимой процедуры?

Ответ:

CREATE PROCEDURE example\_procedure

@inputParam INT,

@outputParam INT OUTPUT

AS

BEGIN

…

END;

* Как вызвать выполнение хранимой процедуры?

Ответ: Хранимую процедуру можно вызвать с помощью оператора EXECUTE (или сокращённо EXEC)

* **Вывод**
* В ходе лабораторной работы приобретены навыки создания и использовать представления в MS SQL Server, а также закреплен навык создания запросов на выборку и модификацию в MS SQL Server.

**Лабораторная работа №9**

**Создание триггеров в СУБД**

* **Цель работы**
* Научиться создавать и использовать триггеры в MS SQL Server;
* Закрепить навык создания запросов на выборку и модификацию данных в MS SQL Server.
* **Ответы на контрольные вопросы**
* Что такое триггер?

Ответ: Триггер — это специальный вид хранимой процедуры, который автоматически выполняется (или "срабатывает") в ответ на определённые события в базе данных, такие как вставка (INSERT), обновление (UPDATE) или удаление (DELETE) записей в таблице. Триггеры позволяют автоматически выполнять определённые действия, такие как проверка данных, ведение журнала изменений или поддержка целостности данных

* Чем триггер отличается от хранимой процедуры?

Ответ:

* Автоматическое выполнение: Триггеры срабатывают автоматически в ответ на определённые события (например, операции вставки, обновления или удаления), тогда как хранимые процедуры вызываются явно пользователем или приложением.
* Контекст выполнения: Триггеры выполняются в контексте операции, которая их вызвала, и могут доступать данные, связанные с этой операцией, в то время как хранимые процедуры могут принимать параметры и выполнять более общие задачи.
* Возвращаемое значение: Триггеры не могут возвращать значения, как хранимые процедуры, и не могут быть вызваны напрямую в SQL-запросах.
* Как запустить триггер на выполнение?

Ответ: Триггер автоматически срабатывает после выполнения соответствующего события (INSERT, UPDATE, DELETE) на таблице, к которой он привязан. Например, если триггер настроен на выполнение после операции INSERT, он будет автоматически вызван после каждой вставки записи в таблицу.

* Каково назначение триггеров?

Ответ: Назначение триггеров включает в себя:

* Поддержка целостности данных: Проверка данных перед их вставкой или обновлением.
* Автоматизация действий: Выполнение автоматических действий, таких как ведение журнала изменений или обновление связанных таблиц.
* Аудит: Сохранение истории изменений в таблицах для последующего анализа.
* Реакция на события: Реагирование на изменения в данных без необходимости ручного вмешательства.
* Чем отличаются триггеры INSTEAD OF и AFTER?

Ответ:

* AFTER триггеры: Срабатывают после выполнения операции (INSERT, UPDATE, DELETE). Они могут использоваться для выполнения дополнительных действий после изменения данных. Например, триггер может записывать данные в журнал после того, как запись была добавлена.
* INSTEAD OF триггеры: Срабатывают вместо выполнения операции. Они позволяют перехватывать стандартные действия и заменять их на другую логику. Например, INSTEAD OF триггер может использоваться для обновления нескольких таблиц вместо одной при выполнении операции UPDATE на представлении.
* **Вывод**
* В ходе лабораторной работы приобретены навыки создания и использовать представления в MS SQL Server, а также закреплен навык создания запросов на выборку и модификацию в MS SQL Server.

**Лабораторная работа №10**

**Реализация доступа пользователей к базе данных**

* **Цель работы**
* Научиться использовать системные хранимые процедуры и DDL-команды для управления именами входа и пользователями БД в СУБД;
* Научиться назначать привилегии пользователю БД;
* Закрепить навык создания объектов БД.
* **Ответы на контрольные вопросы**
* В чем отличие между именами входа и пользователями БД?

Ответ:

* Имя входа (Login): Это учетная запись, которая используется для аутентификации на уровне сервера. Имя входа позволяет пользователю подключаться к серверу SQL Server. Оно может быть связано с Windows-учетной записью или с учетной записью SQL Server.
* Пользователь БД (Database User): Это учетная запись, которая существует на уровне конкретной базы данных и связана с именем входа. Пользователь БД определяет, какие права и доступ к объектам базы данных имеются у данного пользователя. Один и тот же вход может быть связан с несколькими пользователями в разных базах данных.
* Как идентифицируются пользователи в MS SQL Server?

Ответ: В MS SQL Server пользователи идентифицируются через:

* Имена входа: Учетные записи, используемые для подключения к серверу.
* Пользователи БД: Учетные записи, которые определяют доступ и права на уровне конкретной базы данных. Пользователь БД связывается с именем входа, что позволяет ему получать доступ к ресурсам базы данных.
* На какие уровни разделяется система безопасности MS SQL Server?

Ответ: Система безопасности MS SQL Server разделяется на следующие уровни:

* Уровень сервера: Управляет аутентификацией и авторизацией на уровне SQL Server. Здесь создаются имена входа и роли сервера.
* Уровень базы данных: Управляет доступом и правами на уровне конкретной базы данных. Здесь создаются пользователи БД и роли БД.
* Уровень объектов: Управляет правами доступа к конкретным объектам базы данных, таким как таблицы, представления, процедуры и т. д. Здесь определяются права на уровне объектов.
* Каково назначение ролей сервера?

Ответ: Роли сервера в MS SQL Server служат для управления правами доступа на уровне сервера. Они позволяют группировать пользователей с одинаковыми правами, упрощая управление безопасностью. Основные роли сервера включают:

* sysadmin: Полный доступ ко всем функциям сервера.
* serveradmin: Управление конфигурацией сервера.
* securityadmin: Управление пользователями и ролями.
* processadmin: Управление процессами на сервере.

Назначение ролей сервера заключается в упрощении управления безопасностью и предоставлении определенных прав группам пользователей.

* Каково назначение ролей БД?

Ответ: Роли БД в MS SQL Server служат для управления правами доступа на уровне конкретной базы данных. Они позволяют группировать пользователей с одинаковыми правами доступа к объектам базы данных. Основные роли БД включают:

* db\_owner: Полный доступ к базе данных, включая возможность управления ее структурой и данными.
* db\_datareader: Доступ к чтению всех данных в базе данных.
* db\_datawriter: Доступ к записи данных в таблицы базы данных.
* db\_securityadmin: Управление правами доступа и пользователями базы данных.

Назначение ролей БД заключается в упрощении управления безопасностью и предоставлении определенных прав группам пользователей в рамках конкретной базы данных.

* **Вывод**
* В ходе лабораторной работы приобретены навыки использования системных хранимые процедур и DDL-команд для управления именами входа и пользователями БД в СУБД и назначения привилегий пользователю БД, а также закреплен навык создания объектов БД.

**Лабораторная работа №11**

**Выполнение резервного копирования и восстановления БД**

* **Цель работы**
* Научиться выполнять резервное копирование БД;
* Научиться сохранять во внешних файлах описание структуры и данные БД;
* Научиться выполнять восстановление БД.
* **Ответы на контрольные вопросы**
* Для чего создаются резервные копии БД?

Ответ: Резервные копии баз данных создаются для защиты данных от потери, повреждения или сбоев. Основные цели резервного копирования:

* Восстановление данных после сбоев системы, аппаратных неисправностей или ошибок пользователя.
* Защита от случайного удаления или изменения данных.
* Возможность восстановления базы данных до определенного состояния (например, на момент последнего резервного копирования).
* Обеспечение непрерывности бизнеса и минимизация времени простоя.
* В чем отличие между полным и разностным резервным копированием?

Ответ:

* Полное резервное копирование: Создает полную копию всей базы данных. Это наиболее надежный способ резервного копирования, но занимает больше времени и места на диске.
* Разностное резервное копирование: Создает резервную копию только тех данных, которые изменились с момента последнего полного резервного копирования. Это быстрее и требует меньше места, но для восстановления данных необходимо наличие полного резервного копирования и последнего разностного резервного копирования.
* Как часто должен выполняться каждый из видов резервного копирования БД (привести пример расписания)?

Ответ:

* Полное резервное копирование: Рекомендуется выполнять раз в неделю или раз в месяц, в зависимости от объема данных и их изменений. Например, каждую субботу в 02:00.
* Разностное резервное копирование: Рекомендуется выполнять ежедневно между полными резервными копированиями. Например, каждый день в 02:00, кроме субботы.
* Какие скрипты можно сформировать для объектов БД?

Ответ: Для объектов базы данных можно сформировать следующие скрипты:

* Скрипты для создания таблиц, представлений, хранимых процедур, функций и триггеров.
* Скрипты для создания индексов и ограничений.
* Скрипты для вставки, обновления или удаления данных.
* Скрипты для создания и изменения ролей и пользователей.
* Скрипты для настройки прав доступа к объектам базы данных.

* Как выполнить восстановление БД из резервной копии?

Ответ: Восстановление базы данных из резервной копии можно выполнить с помощью SQL Server Management Studio (SSMS) или с помощью T-SQL команды. В общем, процесс включает:

* Открытие SSMS.
* Подключение к экземпляру SQL Server.
* Выбор базы данных, которую нужно восстановить.
* Выполнение команды восстановления с указанием пути к резервной копии.
* Какая команда выполняет восстановление БД из резервной копии?

Ответ: Команда для восстановления базы данных из резервной копии в T-SQL выглядит следующим образом:

RESTORE DATABASE [имя\_базы\_данных] FROM DISK = 'путь\_к\_резервной\_копии.bak';

* В каком порядке надо восстанавливать резервные копии?

Ответ: При восстановлении резервных копий необходимо следовать следующему порядку:

* Сначала восстановите полное резервное копирование.
* Затем восстановите все разностные резервные копии в порядке их создания.
* Если используются журналы транзакций, восстановите их в том порядке, в котором они были созданы, для завершения восстановления до последнего состояния.

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

Ответ: Параметр WITH RECOVERY используется для завершения процесса восстановления и перевода базы данных в состояние доступности. Параметр WITH NORECOVERY позволяет продолжить восстановление, если есть дополнительные резервные копии для восстановления.

* Для чего используется мастер импорта и экспорта?

Ответ: Мастер импорта и экспорта в SQL Server используется для:

* Переноса данных между различными источниками данных (например, из Excel, CSV, других баз данных).
* Импорта данных в таблицы базы данных из внешних источников.
* Экспорта данных из таблиц базы данных в другие форматы или источники.
* Создания заданий для автоматизации процессов импорта и экспорта данных.
* **Вывод**
* В ходе лабораторной работы приобретены навыки выполнения резервного копирования БД, сохранения во внешних файлах описания структуры и данных БД и выполнения восстановления БД;

**Лабораторная работа №12**

**Экспорт данных базы в документы пользователя**

* **Цель работы**
* Научиться выполнять экспорт данных из БД.
* **Ответы на контрольные вопросы**
* Как выполнить экспорт csv-файла в Management Studio?

Ответ: Для экспорта данных в CSV-файл из SQL Server Management Studio выполните следующие шаги:

* Откройте SQL Server Management Studio и подключитесь к вашему серверу.
* Выполните запрос, данные которого вы хотите экспортировать.
* В окне результатов щелкните правой кнопкой мыши и выберите "Сохранить результаты как...".
* В диалоговом окне выберите место для сохранения файла и укажите имя файла с расширением .csv.
* Убедитесь, что в поле "Тип" выбран "CSV-файл (\*.csv)".
* Нажмите "Сохранить".
* Как выполнить экспорт xlsx-файла в Management Studio?

Ответ: Для экспорта данных в XLSX-файл из SQL Server Management Studio:

* Откройте SQL Server Management Studio и подключитесь к вашему серверу.
* Выполните запрос, данные которого вы хотите экспортировать.
* В окне результатов щелкните правой кнопкой мыши и выберите "Сохранить результаты как...".
* В диалоговом окне выберите место для сохранения файла и укажите имя файла с расширением .xlsx.
* Однако, SSMS не поддерживает прямой экспорт в XLSX. Вместо этого, вы можете использовать Microsoft Excel для импорта данных из SQL Server, а затем сохранить в формате XLSX.
* Как в Excel открыть файл формата txt/csv и xml?

Ответ:

* TXT/CSV:
* Откройте Microsoft Excel.
* Перейдите в меню "Файл" и выберите "Открыть".
* Найдите ваш файл формата TXT или CSV.
* Выберите файл и нажмите "Открыть".
* Если это CSV, Excel автоматически откроет его и разделит данные по столбцам. Для TXT может понадобиться использовать "Мастер импорта текстов", чтобы указать разделители.
* XML:
* Откройте Microsoft Excel.
* Перейдите в меню "Файл" и выберите "Открыть".
* Найдите ваш файл формата XML.
* Выберите файл и нажмите "Открыть".
* Excel предложит вам выбрать, как открыть XML-файл (например, как таблицу или в виде схемы).
* Данные из скольких таблиц могут храниться в файле csv?

Ответ: Файл CSV представляет собой текстовый файл, в котором данные организованы в виде строк и столбцов. Он может хранить данные только из одной таблицы. Если вы хотите экспортировать данные из нескольких таблиц, вам нужно будет объединить их в один набор данных (например, с помощью SQL-запроса с JOIN) перед экспортом.

* Как экспортировать данные в файл формата JSON?

Ответ: Для экспорта данных в файл формата JSON в SQL Server можно использовать следующий подход:

* Выполните запрос, который возвращает данные, которые вы хотите экспортировать.
* Используйте функцию FOR JSON в вашем запросе. Например:
* SELECT \* FROM [ваша\_таблица]
* FOR JSON AUTO;
* Сохраните результаты в текстовый файл с расширением .json:
* Щелкните правой кнопкой мыши в окне результатов и выберите "Сохранить результаты как...".
* Укажите имя файла с расширением .json.
* Как экспортировать данные в файл формата XML?

Ответ: Для экспорта данных в файл формата XML в SQL Server выполните следующие шаги:

* Выполните запрос, который возвращает данные, которые вы хотите экспортировать.
* Используйте функцию FOR XML в вашем запросе. Например:
* SELECT \* FROM [ваша\_таблица]
* FOR XML AUTO;
* Сохраните результаты в текстовый файл с расширением .xml:
* Щелкните правой кнопкой мыши в окне результатов и выберите "Сохранить результаты как...".
* Укажите имя файла с расширением .xml.
* **Вывод**
* В ходе лабораторной работы приобретены навыки выполнения экспорта данных из БД.

**Лабораторная работа №13**

**Импорт данных пользователя в базу данных**

* **Цель работы**
* Научиться выполнять импорт данных в БД.
* **Ответы на контрольные вопросы**
* Как выполнить импорт xlsx-файла в Management Studio?

Ответ:

* Подготовка Excel файла: Убедитесь, что ваш Excel файл (xlsx) имеет правильный формат. Убедитесь, что данные находятся в первой таблице и имеют заголовки.
* Использование SQL Server Import and Export Wizard:
* Откройте SQL Server Management Studio (SSMS).
* Подключитесь к вашему серверу базы данных.
* Щелкните правой кнопкой мыши на базе данных, в которую вы хотите импортировать данные.
* Выберите Tasks -> Import Data....
* В мастере импорта данных выберите Microsoft Excel в качестве источника данных и укажите путь к вашему файлу.
* Выберите версию Excel, если это необходимо.
* Укажите целевую базу данных и таблицу, в которую вы хотите импортировать данные.
* Завершите мастер и запустите импорт.

* Как выполнить импорт csv-файла в Management Studio?

Ответ:

Использование SQL Server Import and Export Wizard:

* Откройте SSMS.
* Подключитесь к вашему серверу базы данных.
* Щелкните правой кнопкой мыши на базе данных, в которую вы хотите импортировать данные.
* Выберите Tasks -> Import Data....
* В мастере импорта данных выберите Flat File Source в качестве источника данных и укажите путь к вашему CSV файлу.
* Настройте параметры (например, разделитель, текстовые ограничители и т.д.).
* Укажите целевую базу данных и таблицу, в которую вы хотите импортировать данные.
* Завершите мастер и запустите импорт.
* Как выполнить импорт файла XML?

Ответ:

Использование SQL Server Import and Export Wizard:

* Откройте SSMS.
* Щелкните правой кнопкой мыши на базе данных и выберите Tasks -> Import Data....
* В мастере выберите XML Source в качестве источника данных и укажите путь к вашему XML файлу.
* Настройте соответствие между элементами XML и таблицами базы данных.
* Завершите мастер и запустите импорт.
* Как выполнить импорт файла JSON?

Ответ:

Использование T-SQL:  
SQL Server 2016 и более поздние версии поддерживают работу с JSON. Вы можете использовать OPENJSON для импорта данных.

DECLARE @json NVARCHAR(MAX)

SELECT @json = BulkColumn

FROM OPENROWSET(BULK 'C:\Path\To\Your\File.json', SINGLE\_CLOB) AS x;

SELECT \*

FROM OPENJSON(@json)

WITH (

Property1 NVARCHAR(100) '$.property1',

Property2 NVARCHAR(100) '$.property2'

);

* **Вывод**
* В ходе лабораторной работы приобретены навыки выполнения импорта данных в БД.

**Лабораторная работа №14**

**Создание слоя доступа к данным БД**

* **Цель работы**
* Научиться создавать приложение C# для организации доступа к БД.
* **Ответы на контрольные вопросы**
* Какое назначение у элемента SqlConnection?

Ответ: SqlConnection — это класс в .NET, который используется для установления соединения с базой данных Microsoft SQL Server. Он предоставляет методы и свойства для открытия и закрытия соединений, управления транзакциями и выполнения команд SQL. Основные функции SqlConnection включают:

* Установка строки подключения к базе данных.
* Открытие и закрытие соединений с базой данных.
* Управление состоянием соединения.
* Какое назначение у элемента SqlCommand?

Ответ: SqlCommand — это класс, который представляет SQL-запрос или хранимую процедуру, которую вы хотите выполнить на SQL Server. Он позволяет выполнять команды SQL, такие как SELECT, INSERT, UPDATE, DELETE, а также управлять параметрами команд. Основные функции SqlCommand включают:

* Выполнение SQL-запросов или хранимых процедур.
* Передача параметров в запросы.
* Получение результатов выполнения команды.

* Какое назначение у элемента SqlDataReader?

Ответ: SqlDataReader — это класс, который используется для чтения данных из базы данных SQL Server в потоковом режиме. Он позволяет извлекать данные из результата выполнения SQL-запроса, предоставляя доступ к строкам и столбцам данных. Основные функции SqlDataReader включают:

* Чтение данных по одной строке за раз, что делает его эффективным для работы с большими объемами данных.
* Обеспечение быстрого доступа к данным, так как он работает в режиме только для чтения и не требует загрузки всех данных в память.
* Какие пространства имен требуется подключить для реализации подключения к СУБД MS SQL Server, MySQL соответственно?

Ответ: using System.Data.SqlClient для MS SQL Server, MySql.Data.MySqlClient для My Sql.

* Какие методы класса SqlCommand позволяют выполнить SQL-запрос?

Ответ: ExecuteNonQuery(), ExecuteScalar().

* **Вывод**
* В ходе лабораторной работы были приобретены навыки создания приложения C# для организации доступа к БД.

**Лабораторная работа №15**

**Применение ORM для доступа к данным БД**

* **Цель работы**
* Научиться создавать приложение C# для организации доступа к БД.
* Научиться проектировать приложение, использующее паттерн репозиторий и Dapper.
* **Ответы на контрольные вопросы**
* Что такое Dapper и для чего используется?

Ответ: Dapper — это микро-ORM (Object-Relational Mapper) для .NET, разработанный для упрощения работы с базами данных. Он обеспечивает более легкий и быстрый способ выполнения SQL-запросов и отображения результатов на объекты .NET. Dapper используется для:

* Упрощения работы с ADO.NET, предоставляя более удобный интерфейс для выполнения SQL-запросов.
* Ускорения выполнения запросов за счет минимизации накладных расходов по сравнению с более крупными ORM, такими как Entity Framework.
* Быстрого маппинга данных из таблиц базы данных в объекты C# и обратно.
* Какие методы Dapper позволяют извлечь данные из БД?

Ответ: Query<T>(); QueryFirst<T>(); QueryFirstOrDefault<T>(); QuerySingle<T>(); QuerySingleOrDefault<T>().

* Какие методы Dapper позволяют изменить данные в БД?

Ответ: Execute(); ExecuteScalar();

* Зачем используетсяя паттерн «репозиторий»?

Ответ: Паттерн «репозиторий» используется для абстрагирования доступа к данным и управления взаимодействием с источником данных (например, базой данных). Основные цели и преимущества использования паттерна репозиторий:

* Изоляция логики доступа к данным: Репозитории отделяют бизнес-логику от логики доступа к данным, что делает код более чистым и поддерживаемым.
* Упрощение тестирования: Репозитории позволяют легко подменять зависимости в тестах, что упрощает написание юнит-тестов, так как можно использовать мок-объекты вместо реального доступа к базе данных.
* Упрощение изменения источника данных: Если в будущем потребуется изменить способ хранения данных (например, перейти на другую СУБД), то изменения будут минимальными, так как вся логика доступа к данным сосредоточена в репозиториях.
* Упрощение работы с коллекциями объектов: Репозитории могут предоставлять методы для работы с коллекциями объектов, что упрощает выполнение операций с группами данных.
* Повышение читаемости и структурированности кода: Репозитории помогают организовать код, делая его более понятным и структурированным, что облегчает его сопровождение.

* **Вывод**
* В ходе лабораторной работы были приобретены навыки создавания приложения C# для организации доступа к БД и проектирования приложения, использующего паттерн репозиторий и Dapper.

**Лабораторная работа №16**

**Применение ORM EF Core**

* **Цель работы**
* Научиться создавать приложение C# для организации доступа к БД;
* Научиться разрабатывать приложение, использующее EF Core.
* **Ответы на контрольные вопросы**
* Что такое «ORM»?

Ответ: ORM (Object-Relational Mapping) — это технология, которая позволяет разработчикам работать с базами данных, используя объектно-ориентированные подходы. ORM автоматически преобразует данные между реляционными базами данных и объектами в коде, что позволяет разработчикам манипулировать данными как с объектами, а не с таблицами и строками. Это упрощает взаимодействие с базами данных и уменьшает количество кода, необходимого для выполнения операций CRUD (создание, чтение, обновление, удаление).

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

Ответ: Entity Framework Core (EF Core) — это кроссплатформенный, легковесный и расширяемый ORM для .NET. Он предназначен для упрощения работы с базами данных, предоставляя разработчикам возможность:

* Использовать объектно-ориентированный подход для работы с данными, что делает код более читаемым и поддерживаемым.
* Выполнять операции CRUD с помощью LINQ (Language Integrated Query), что позволяет писать запросы к базе данных на C#.
* Поддерживать различные базы данных, включая SQL Server, SQLite, PostgreSQL и другие через провайдеры.
* Использовать миграции для управления изменениями в структуре базы данных.
* Как получить данные из БД, используя EF Core?

Ответ: Для получения данных из базы данных с использованием EF Core, вы обычно выполняете следующие шаги:

* Подключение библиотек Microsoft.EFCore.SqlServer, M.EFCore.Tools, загрузка контекста и моделей данных из бд с помощью scaffold
* Создание контекста базы данных: Контекст базы данных (наследник DbContext) представляет собой сессию с базой данных и предоставляет методы для работы с ней.

public class MyDbContext : DbContext

{

public DbSet<User> Users { get; set; }

}

* Запрос данных: Используйте LINQ для выполнения запросов к таблицам, представленным DbSet или асинхронные запросы.

var users = await context.Users.ToListAsync()

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

Ответ:

Вставка:

var newUser = new User { Name = "Ваня" };

context.Users.Add(newUser );

context.SaveChangesAsync();

Обновление:

var user = await context.Users.FirstOrDefaultAsync(u => u.Id == 1);

if (user != null)

{

user.Name = "Updated Name";

await context.SaveChangesAsync();

}

Удаление:

var user = await context.Users.FirstOrDefaultAsync(u => u.Id == 1);

if (user != null)

{

context.Users.Remove(user);

await context.SaveChangesAsync();

}

* **Вывод**
* В ходе лабораторной работы приобретены навыки создавания приложений C# для организации доступа к БД и разработки приложений, использующих EF Core.

**Лабораторная работа №17**

**Разработка приложения для фильтрации, поиска и сортировки данных**

* **Цель работы**
* Научиться создавать приложение C# для организации доступа к БД средствами EF Core.
* Научиться выполнять сортировку, фильтрацию и постраничный вывод данных, используя LINQ-запросы.
* **Ответы на контрольные вопросы**
* Для чего используются метод OrderBy, OrderByDescending, ThenBy, ThenByDescending и в чем их отличие?

Ответ:

* OrderBy: Используется для сортировки коллекции по указанному полю в порядке возрастания. Например, если у вас есть список пользователей, вы можете отсортировать их по имени.
* OrderByDescending: Используется для сортировки коллекции по указанному полю в порядке убывания.
* ThenBy: Используется для вторичной сортировки уже отсортированной коллекции. Например, если вы сначала отсортировали по имени, а затем хотите отсортировать по возрасту.
* ThenByDescending: Используется для вторичной сортировки в порядке убывания.

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

Ответ:

Take: Используется для выбора определенного количества элементов из начала коллекции. Например, если вы хотите получить первые 10 пользователей.

var topUsers = users.Take(10);

Skip: Используется для пропуска определенного количества элементов в коллекции. Например, если вы хотите пропустить первые 10 пользователей и получить следующих 10.

var nextUsers = users.Skip(10).Take(10);

* Для чего используется метод Where?

Ответ: Метод Where используется для фильтрации коллекции по определенному условию. Он возвращает только те элементы, которые соответствуют заданному предикату.

* Какие логические операторы могут применяться при составлении условий?

Ответ:

* &&: Используется для объединения нескольких условий, все из которых должны быть истинными.
* ||: Используется для объединения условий, при которых достаточно, чтобы одно из них было истинным.
* !: Используется для отрицания условия.
* Как проверить, что значение есть в списке?

Ответ: Для проверки, содержится ли значение в списке, можно использовать метод Contains().

* Как проверить, что строка начинается с определенного текста?

Ответ: Для проверки, начинается ли строка с определенного текста, можно использовать метод StartsWith().

* Как проверить, что строка содержит определенный текст?

Ответ: Для проверки, содержит ли строка определенный текст, можно использовать метод Contains().

* Как указать в Select список требуемых данных?

Ответ: var userNames = users.Select(u => new { u.Name, u.Age });

* **Вывод**
* В ходе лабораторной работы приобретен навык создавать приложение C# для организации доступа к БД средствами EF Core.

**Лабораторная работа №18**

**Разработка приложения для редактирования данных**

* **Цель работы**
* Научиться выполнять вставку, обновление и удаление записей средствами EF Core;
* Научиться обеспечивать обратную связь при редактировании данных.
* **Ответы на контрольные вопросы**
* Для чего используются методы Add() и AddRange() в EF Core?

Ответ:

* Add(): Метод используется для добавления одного объекта в контекст базы данных. После вызова этого метода объект будет помечен как "добавленный" и будет включен в следующую операцию сохранения.
* AddRange(): Метод используется для добавления нескольких объектов в контекст базы данных одновременно. Это более эффективный способ, чем добавление объектов по одному.
* Для чего используются методы Update() в EF Core?

Ответ: Метод Update() используется для изменения существующего объекта в контексте базы данных. Он помечает объект как "измененный", и при следующем вызове метода сохранения изменения будут применены к базе данных.

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

Ответ:

* Remove(): Метод используется для удаления одного объекта из контекста базы данных. Объект будет помечен как "удаленный" и будет удален из базы данных при следующем сохранении.
* RemoveRange(): Метод используется для удаления нескольких объектов одновременно. Это позволяет более эффективно удалять группы объектов.
* Как сохранить изменения в БД, используя EF Core?

Ответ: Для сохранения изменений в базе данных используется метод SaveChanges(). Этот метод применяет все изменения, сделанные в контексте, к базе данных.

* Как изменить значения полей объекта?

Ответ:

var user = context.Users.Find(1);

user.Name = "Updated Name";

context.SaveChanges();

* Какое значение по умолчанию присваивается идентификатору нового объекта?

Ответ: В EF Core, когда вы создаете новый объект, идентификатор (обычно это поле типа int или Guid) не присваивается вручную. По умолчанию, если база данных настроена на автоинкремент (например, столбец с идентификатором в SQL Server), то значение идентификатора будет автоматически присвоено базой данных при вставке нового объекта. В случае Guid обычно используется Guid.NewGuid().

* Как передать объект с одной формы на другую?

Ответ:

private User \_user;

public MainWindow(User user)

{

InitializeComponent();

\_user = user;

}

* **Вывод**
* В ходе лабораторной работы приобретены навыки выполнять вставку, обновление и удаление записей средствами EF Corе и обеспечивать обратную связь при редактировании данных.

**Лабораторная работа №19**

**Выполнение SQL-команд и SQL-подпрограмм**

* **Цель работы**
* Научиться выполнять SQL-команды и вызывать хранимые процедуры и функции пользователя средствами EF Core.
* **Ответы на контрольные вопросы**
* Как выполнить команду на выборку данных в EF Core?

Ответ: Например: context.Users.ToList(), также есть FirstOrDefault()/SingleOrDefault();

* Как выполнить команду на модификацию данных в EF Core?

Ответ:

var user = context.Users.Find(1);

if (user != null)

{

user.Name = "Updated Name";

context.SaveChanges();

}

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

Ответ: var user = context.Users .FromSqlRaw("SELECT \* FROM Users WHERE Id = {0}", userId) .ToList();

* Как вызвать табличную функцию в EF Core?

Ответ: К структурным паттернам относятся: Adapter, Bridge, Composite, Decorator, Facade, Flyweight, Proxy.

* Как вызвать хранимую процедуру в EF Core?

Ответ: var activeUsers = context.Users .FromSqlRaw("SELECT \* FROM GetActiveUsers()") .ToList();

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

Ответ:

var outputParameter = new SqlParameter

{

ParameterName = "@OutputParam",

SqlDbType = SqlDbType.Int,

Direction = ParameterDirection.Output

};

context.Database.ExecuteSqlRaw("EXEC YourStoredProcedure @OutputParam OUT", outputParameter);

* **Вывод**
* В ходе лабораторной работы приобретены навыки выполнять SQL-команды и вызывать хранимые процедуры и функции пользователя средствами EF Core.

**Лабораторная работа №20**

**Реализация разграничения прав доступа пользователей**

* **Цель работы**
* Научиться разграничивать права доступа пользователей на уровне интерфейса приложения;
* Научиться изменять настройки подключения к БД средствами Entity Framework Core.
* **Ответы на контрольные вопросы**
* Как изменить настройки подключения к БД в клиентском приложении?

Ответ: Чтобы изменить настройки подключения к базе данных в клиентском приложении, необходимо обновить строку подключения в конфигурационном файле, таком как appsettings.json в приложениях ASP.NET Core.

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

Ответ:

* User (Пользователь)
* UserId (PK)
* UserName
* Email
* Role (Роль)
* RoleId (PK)
* RoleName
* Description
* UserRole
* UserRoleId (PK)
* UserId (FK)
* RoleId (FK)
* Что такое «авторизация»?

Ответ: Авторизация — это процесс проверки прав доступа пользователя к ресурсам или действиям в приложении.

* Что такое «регистрация»?

Ответ: Регистрация — это процесс создания нового пользователя в системе. Во время регистрации пользователь предоставляет необходимую информацию, такую как имя, адрес электронной почты и пароль.

* **Вывод**
* В ходе лабораторной работы приобретены навыки разграничивать права доступа пользователей на уровне интерфейса приложения и изменять настройки подключения к БД средствами ЕF Core.

**Лабораторная работа №21**

**Разработка приложения для импорта данных**

* **Цель работы**
* Импорт данных пользователя в базу данных средствами EF Core.
* **Ответы на контрольные вопросы**
* Как выполнить импорт из csv-файла в БД через приложение?

Ответ:

string filePath = "data.csv";

// Прочитать все строки из файла

string[] lines = File.ReadAllLines(filePath);

// Пройти по каждой строке и разделить ее на поля

foreach (string line in lines)

{

string[] fields = line.Split(',');

... // работа с полученными полями для преобразования в объекты

}.

* Как импортировать изображение в БД через приложение?

Ответ:

string filePath = "image.jpg"; // Замените на путь к файлу

byte[] fileBytes = File.ReadAllBytes(filePath);

... // для проверки размера файла используется fileBytes.Length (длина в байтах)

…// запись в бд

* Как выполнить импорт файла JSON в БД через приложение?

Ответ:

string filePath = "data.json";

string jsonString = File.ReadAllText(filePath);

List<User> users = JsonSerializer.Deserialize<List<User>>(jsonString);

// Пройти по каждому объекту

foreach (var user in users)

{

... // работа с полученным объектом

}

* **Вывод**
* В ходе лабораторной работы приобретены навыки импорта данных пользователя в БД средствами EF Core.

**Лабораторная работа №22**

**Разработка приложения для экспорта данных в текстовые файлы**

* **Цель работы**
* Научиться выполнять экспорт данных из БД в формате текстового файла.
* Научиться выполнять экспорт данных из БД с применением Microsoft Office Word.
* **Ответы на контрольные вопросы**
* Какое пространство имен требуется подключить для работы с Word?

Ответ: Microsoft.Office.Interop.Word.

* Какие библиотеки позволяют сохранять данные в формате docx?

Ответ: Достаточно System.IO, но порой необходим DocumentFormat.OpenXml/Aspose.Words/DocX

* Как выполнить экспорт данных в формате txt?

Ответ:

using (var writer = new StreamWriter(filePath))

{

foreach (var line in data)

{

await writer.WriteLineAsync(line);

}

}

* **Вывод**
* В ходе лабораторной работы приобретен навыки выполнять экспорт данных из БД в формате текстового файла и экспорт данных из БД с применением Microsoft Office Word.

**Лабораторная работа №23**

**Разработка приложения для экспорта данных в табличные файлы**

* **Цель работы**
* Научиться выполнять экспорт данных из БД в формате табличного файла;
* Научиться выполнять экспорт данных из БД с применением Microsoft Office Excel.
* **Ответы на контрольные вопросы**
* Какое пространство имен требуется подключить для работы с Word?

Ответ: Microsoft.Office.Interop.Word.

* Какие библиотеки позволяют сохранять данные в формате docx?

Ответ: Достаточно System.IO, но порой необходим DocumentFormat.OpenXml/Aspose.Words/DocX

* Как выполнить экспорт данных в формате txt?

Ответ:

using (var writer = new StreamWriter(filePath))

{

foreach (var line in data)

{

await writer.WriteLineAsync(line);

}

}

* **Вывод**
* В ходе проделанной лабораторной работы я изучил процесс документирования программного кода.

**Практическая работа №1**

**Разработка web-API для доступа к данным**

* **Цель работы**
* Научиться выполнять разработку web-API для доступа к БД.
* **Ответы на контрольные вопросы**
* Что такое REST-запрос?

Ответ: REST-запрос — это запрос, отправленный клиентом на сервер с использованием архитектурного стиля REST (Representational State Transfer). REST использует стандартные HTTP методы (такие как GET, POST, PUT, DELETE) для выполнения операций над ресурсами, которые представляются в виде URL.

* Что такое RESTful?

Ответ: RESTful — это термин, описывающий веб-сервисы, которые следуют принципам и ограничениям архитектурного стиля REST.

* Для чего используется метод GET?

Ответ: Метод GET используется для запроса данных с сервера. Он извлекает информацию, не изменяя ее.

* Для чего используется метод POST?

Ответ: Метод POST используется для отправки данных на сервер, часто с целью создания нового ресурса.

* Для чего используется метод PUT?

Ответ: Метод PUT используется для обновления существующего ресурса или создания нового ресурса по заданному URI, если он не существует.

* Для чего используется метод DELETE?

Ответ: Метод DELETE используется для удаления указанного ресурса на сервере.

* **Вывод**
* В ходе практической работы приобретены навыки выполнения разработки web-API для доступа к БД.

**Практическая работа №2**

**Работа с табличными файлами**

* **Цель работы**
* Научиться проверять работоспособность RESTful API в клиентском приложении;
* Научиться выполнять тестирование RESTful API методом черного ящика.
* **Ответы на контрольные вопросы**
* Как указать у объекта HttpClient базовый адрес?

Ответ:

HttpClient client = new HttpClient();

client.BaseAddress = new Uri("https://");

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

Ответ:

* 200 OK: Запрос выполнен успешно.
* 201 Created: Запрос выполнен успешно, и ресурс был создан.
* 204 No Content: Запрос выполнен успешно, но нет содержимого для возврата.
* 400 Bad Request: Запрос некорректен (например, неверный синтаксис).
* 401 Unauthorized: Требуется аутентификация для доступа к ресурсу.
* 403 Forbidden: Доступ к ресурсу запрещен.
* 404 Not Found: Запрашиваемый ресурс не найден.
* 500 Internal Server Error: Ошибка на сервере.
* 503 Service Unavailable: Сервер временно недоступен.
* Какой метод класса HttpClient используется для получения данных?

Ответ: GetAsync();

* Какой метод класса HttpClient используется для вставки данных?

Ответ: PostAsync();

* Какой метод класса HttpClient используется для изменения данных?

Ответ: PutAsync.

* Какой метод класса HttpClient используется для удаления данных?

Ответ:DeleteAsync().

* **Вывод**
* В ходе практической работы приобретены навыки проверять работоспособность RESTful API в клиентском приложении и выполнять тестирование RESTful API методом черного ящика.

**Практическая работа №3**

**Разграничение прав доступа на уровне REST API**

* **Цель работы**
* Научиться выполнять разработку web-API для доступа к БД;
* Научиться работать с JWT.
* **Ответы на контрольные вопросы**
* Какие атрибуты можно указать у методов REST для настройки доступа для авторизованных и неавторизованных пользователей?

Ответ:

* [AllowAnonymous]: Позволяет доступ к методу для всех пользователей, независимо от их авторизации.
* [Authorize]: Ограничивает доступ к методу только для авторизованных пользователей. Можно указать дополнительные параметры, такие как роли или политики.
* Для чего используется JWT?

Ответ: JWT (JSON Web Token) используется для передачи информации между клиентом и сервером в безопасном и компактном формате.

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

Ответ:

* **Cookie**:
* Хранятся на стороне клиента и автоматически отправляются с каждым запросом к серверу.
* Уязвимы к атакам CSRF (Cross-Site Request Forgery), если не используются механизмы защиты, такие как CSRF-токены.
* Могут быть настроены с флагами HttpOnly и Secure, что повышает безопасность, но не исключает возможности кражи через XSS (Cross-Site Scripting).
* **JWT**:
* Токены могут храниться в локальном хранилище или в памяти клиента, что делает их более гибкими.
* Не подвержены CSRF, поскольку не отправляются автоматически с запросами (необходима явная отправка токена в заголовках).
* Уязвимы к XSS, если токены хранятся в небезопасном месте.
* Легко масштабируются и могут использоваться в распределенных системах.
* Какие настройки можно указать при создании токена?

Ответ:

При создании JWT можно указать следующие настройки:

* Срок действия (exp): Время, через которое токен станет недействительным.
* Подпись (sign): Алгоритм подписи (например, HMAC, RSA) для обеспечения целостности токена.
* Идентификатор пользователя (sub): Идентификатор пользователя, которому принадлежит токен.
* Роли (roles): Роли или разрешения, связанные с пользователем.
* Дополнительные пользовательские данные: Любая другая информация, необходимая для приложения.

* Почему для авторизации следует использовать метод POST?

Ответ: Метод POST следует использовать для авторизации по следующим причинам:

* Безопасность: Данные передаются в теле запроса, что уменьшает вероятность их перехвата (по сравнению с GET, где данные передаются в URL).
* Изменение состояния: Авторизация может изменять состояние сервера (например, создание сессии), что соответствует принципам REST.
* Долгосрочные операции: POST позволяет передавать большие объемы данных, что может быть полезно для передачи учетных данных и других параметров.
* **Вывод**
* В ходе практической работы приобретены навыки выполнения разработки web-API для доступа к БД и работы с JWT.

**Практическая работа №4**

**Разработка веб-клиента**

* **Цель работы**
* Научиться разрабатывать клиентское веб-приложение для доступа к БД.
* Научиться проверять работоспособность RESTful API в клиентском приложении.
* **Ответы на контрольные вопросы**
* Как выполнить HTTP-запрос к API с помощью Fetch API?

Ответ: Чтобы выполнить HTTP-запрос к API с помощью Fetch API, можно использовать следующий синтаксис:

fetch('https://api.example.com/data', {

method: 'GET', // или 'POST', 'PUT', 'DELETE' и т.д.

headers: {

'Content-Type': 'application/json',

// другие заголовки, если необходимо

},

// body: JSON.stringify(data) // если метод POST или PUT

})

.then(response => {

if (!response.ok) {

throw new Error('Network response was not ok ' + response.statusText);

}

return response.json(); // или response.text() для текстового ответа

})

.then(data => {

console.log(data); // обработка полученных данных

})

.catch(error => {

console.error('There was a problem with the fetch operation:', error);

});

* Как обрабатывать ошибки при вызове API в JavaScript?

Ответ: Ошибки при вызове API можно обрабатывать с помощью конструкции try...catch.

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

Ответ: Асинхронные функции — это функции, которые позволяют выполнять асинхронный код более удобно и читаемо, используя ключевое слово async перед определением функции и await для ожидания завершения промисов.

* Что такое Razor Pages?

Ответ: Razor Pages — это упрощенный способ создания веб-приложений в ASP.NET Core, который позволяет разработчикам строить страницы с использованием Razor-синтаксиса.

* Как создать и настроить новую страницу Razor в ASP.NET Core?

Ответ: Чтобы создать и настроить новую страницу Razor в ASP.NET Core, выполните следующие шаги:

* Создание страницы:
* В проекте ASP.NET Core откройте папку Pages.
* Щелкните правой кнопкой мыши на папке и выберите "Добавить" -> "Новый элемент".
* Выберите "Razor Page" и дайте ей имя (например, MyPage.cshtml).

* **Вывод**
* В ходе практической работы приобретены навыки разрабатывать клиентское веб-приложение для доступа к БД и проверять работоспособность RESTful API в клиентском приложении.