**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КОМИ**

**Государственное профессиональное образовательное учреждение**

**«Воркутинский арктический горно-политехнический колледж»**

**КУРСОВОЙ ПРОЕКТ**

По дисциплине МДК.07.01 Управление и автоматизация баз данных

Разработка базы данных «Спортивный клуб»

Выполнил студент гр. ИСП-21 /\_\_\_\_\_\_\_\_\_\_\_\_\_/Волчков Александр Владиславович/

(подпись) (Ф.И.О.)

**ОЦЕНКА: \_\_\_\_\_\_\_\_\_\_\_\_\_**

Дата: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**ПРОВЕРИЛ**

Научный руководитель\_\_\_\_\_\_\_\_\_\_\_\_\_/ Егоров Данил Павлович/

(подпись) (Ф.И.О.)

Воркута

2024

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 3](#_Toc187886690)

[ГЛАВА 1. ВЫБОР ИНСТРУМЕНТАРИЯ 4](#_Toc187886691)

[1.1 СУБД SQL server 4](#_Toc187886692)

[1.2 Microsoft SQL Server Management Studio 6](#_Toc187886693)

[ГЛАВА 2. ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ 8](#_Toc187886694)

[2.1 Разработка диаграммы ERD 8](#_Toc187886695)

[2.2 Словарь данных 14](#_Toc187886696)

[ГЛАВА 3. РАЗРАБОТКА БАЗЫ ДАННЫХ 17](#_Toc187886697)

[3.1 Разработка базы данных 17](#_Toc187886698)

[3.2 Введение в безопасность SQL Server 25](#_Toc187886699)

[3.3 Управление безопасностью уровня сервера 28](#_Toc187886700)

[3.4 Управление участниками уровня базы данных 31](#_Toc187886701)

[3.5 Управление разрешениями уровня базы данных 32](#_Toc187886702)

[3.6 Резервное копирование баз данных 33](#_Toc187886703)

[3.7 Шифрование данных баз данных 35](#_Toc187886704)

[ЗАКЛЮЧЕНИЕ 39](#_Toc187886705)

[СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ 41](#_Toc187886706)

# ВВЕДЕНИЕ

Современный ритм жизни требует от человека активного подхода к своему здоровью и физической форме. Спортивные клубы играют ключевую роль в формировании здорового образа жизни, предоставляя качественные услуги для тренировок, развития и восстановления.

Целью курсового проекта является разработка базы данных для спортивного клуба, которая упростит управление данными о посетителях, тренерах и услугах. Внедрение такой системы не только повышает эффективность работы клуба, но и создает комфортные условия для клиентов, помогая им достигать своих спортивных и оздоровительных целей.

Создание базы данных для спортивного клуба подчеркнет его значимость как центра, который объединяет людей вокруг здоровья, активности и личных достижений.

**Объект:** база данных “Спортивный клуб”.

**Предмет:** автоматизация бизнес-процесса работы спортивного клуба.

**Цель работы –** разработать базу данных для информационной системы “Спортивный клуб”.

**Задачи:**

* выбрать инструментарий**;**
* спроектировать базу данных;
* сделать шифрование данных;
* сделать Back Up базы данных;
* сделать восстановление базы данных;
* сделать разграничение прав доступа базы данных.

# ГЛАВА 1. ВЫБОР ИНСТРУМЕНТАРИЯ

# 1.1 СУБД SQL server

Microsoft SQL Server — это реляционная система управления базами данных (RDBMS). Приложения и средства подключаются к экземпляру ИЛИ базе данных SQL Server и взаимодействуют с помощью Transact-SQL (T-SQL).

Таблица 1. Компоненты

|  |  |
| --- | --- |
| Компонент | Description |
| Ядро СУБД | Компонент ядра СУБД представляет собой основную службу для хранения, обработки и обеспечения безопасности данных. Ядро СУБД обеспечивает контролируемый доступ и обработку транзакций в соответствии с требованиями наиболее требовательных приложений, использующих данные в вашей организации. Ядро СУБД также обеспечивает поддержку обеспечения непрерывности бизнес-процессов с помощью непрерывности бизнес-процессов и восстановления базы данных — SQL Server. |
| Службы Машинное обучение (MLS) | Службы SQL Server Машинное обучение поддерживают интеграцию машинного обучения с использованием популярных языков R и Python в корпоративные рабочие процессы. Службы машинного обучения (в базе данных) интегрируют R и Python с SQL Server, что позволяет легко создавать, повторно обучать и оценивать модели, вызывая хранимые процедуры. Машинное обучение Server предоставляет корпоративную поддержку R и Python, не требуя SQL Server. |
| Службы Integration Services (SSIS) | Службы SQL Server Integration Services — это платформа для создания решений для интеграции с данными высокой производительности, включая пакеты, обеспечивающие обработку извлечения, преобразования и загрузки (ETL) для хранения данных. |
| Службы Analysis Services (SSAS) | SQL Server Analysis Services — это платформа аналитических данных и набор инструментов для личной, командной и корпоративной бизнес-аналитики. Серверы и клиентские конструкторы поддерживают традиционные решения OLAP, новые решения табличного моделирования, а также самостоятельную аналитику и совместную работу с помощью Power Pivot, Excel и среды SharePoint Server. Службы Analysis Services также включают интеллектуальный анализ данных, чтобы выявить шаблоны и связи, скрытые внутри больших объемов данных. |
| Службы Reporting Services | Службы SQL Server Reporting Services предоставляют корпоративные функции отчетов с поддержкой Веб-приложений. Вы можете создавать отчеты, которые извлекает содержимое из различных источников данных, публиковать отчеты в различных форматах и централизованно управлять безопасностью и подписками. |
| Репликация | Репликация SQL Server — это набор технологий для копирования и распространения данных и объектов базы данных из одной базы данных в другую, а затем синхронизации между базами данных для обеспечения согласованности. Используя, реплика tion, вы можете распространять данные в разные расположения, а также удаленным или мобильным пользователям с локальными и широкими сетями, подключениями по телефону, беспроводными подключениями и Интернетом. |
| Data Quality Services (DQS) | Службы качества данных предоставляют решение очистки данных на основе знаний. Службы DQS позволяют создать базу знаний, а затем выполнить в ней исправление данных и удаление дубликатов с помощью как автоматизированных, так и интерактивных средств. Можно использовать службы справочных данных на основе облачных вычислений, а также создавать решения по управлению данными, где службы DQS будут интегрированы со службами SQL Server Integration Services и Master Data Services. |
| Master Data Services (MDS) | Службы Master Data Services — это решение SQL Server для управления главными данными. Решение, созданное на основе служб Master Data Services, помогает гарантировать, что отчеты и анализ основаны на правильной информации. Используя службы Master Data Services, вы создаете центральный репозиторий для основных данных и сохраняете проверяемую, защищаемую запись этих данных по мере изменения с течением времени. |

# 1.2 Microsoft SQL Server Management Studio

SQL Server Management Studio (SSMS) — это интегрированная среда для управления любой инфраструктурой SQL, от SQL Server до баз данных SQL Azure. SSMS предоставляет средства для настройки, наблюдения и администрирования экземпляров SQL Server и баз данных. Используется SSMS для развертывания, мониторинга и обновления компонентов уровня данных, используемых приложениями, и создания запросов и скриптов.

SSMS используется для запроса, проектирования и управления базами данных и хранилищами данных, где бы они ни находились на локальном компьютере или в облаке.

Для клиентов, требующих кроссплатформенного компаньона sSMS для управления SQL и другими базами данных Azure, используйте Azure Data Studio.

Таблица 2. Компоненты

|  |  |
| --- | --- |
| Компонент | Description |
| Обозреватель объектов | **Обозреватель объектов** используется для просмотра всех объектов и управления ими в одном экземпляре SQL Server (или более). |
| Обозреватель шаблонов | **Template Explorer** используется для создания файлов со стандартным текстом, которые можно использовать для ускорения разработки запросов и скриптов, и управления ими. |
| Обозреватель решений | Устаревший **обозреватель решений** используется для создания проектов, применяемых для управления такими элементами администрирования, как скрипты и запросы. |
| Visual Database Tools | Используйте средства визуального проектирования, включенные в Management Studio, для создания запросов, таблиц и баз данных схем. |
| Редакторы запросов и текста | Используйте редакторы языка Management Studio для интерактивной сборки и отладки запросов и скриптов. |

# ГЛАВА 2. ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ

# 2.1 Разработка диаграммы ERD

Характеристики диаграммы ERD

Диаграмма ERD (Entity-Relationship Diagram) — это инструмент для проектирования баз данных, который описывает сущности, их атрибуты и взаимосвязи между ними. Основные характеристики диаграммы ERD включают:

Сущности в ERD представляют объекты или понятия, которые требуют хранения данных. Они являются основными компонентами базы данных и отображаются в виде прямоугольников.

Типы сущностей:

* Сильные сущности (Strong Entities): Сущности, которые могут существовать самостоятельно. У них есть собственный первичный ключ, который не зависит от других сущностей.
* Слабые сущности (Weak Entities): Сущности, которые зависят от другой сущности для идентификации. Они не имеют самостоятельного первичного ключа и используют внешний ключ, связанный с родительской сущностью.

Каждая сущность должна быть однозначно определена и описывать конкретный объект, а ее данные не должны повторяться в других сущностях.

Атрибуты описывают свойства или характеристики сущностей и отображаются в виде овалов, соединенных с прямоугольниками (сущностями). Каждый атрибут определяет данные, которые будут храниться в таблице, соответствующей сущности.

Классификация атрибутов:

* Простые атрибуты (Simple Attributes): имеют атомарные значения, которые не могут быть разделены. Например, имя, возраст.
* Составные атрибуты (Composite Attributes): состоят из нескольких частей. Например, атрибут "Адрес" может включать "Город", "Улицу", "Индекс".
* Однозначные атрибуты (Single-Valued Attributes): имеют одно значение для каждой записи. Например, номер телефона.
* Многозначные атрибуты (Multivalued Attributes): могут содержать несколько значений для одной записи. Например, список навыков сотрудника.
* Деривативные атрибуты (Derived Attributes): рассчитываются на основе других атрибутов. Например, возраст может быть вычислен из даты рождения.

Атрибуты могут быть обязательными или необязательными в зависимости от их роли в хранении данных.

Первичный ключ — это атрибут или комбинация атрибутов, который используется для уникальной идентификации каждой записи, в сущности. Он гарантирует, что каждая запись будет уникальной, что важно для обеспечения целостности данных.

Характеристики первичных ключей:

* Значение первичного ключа должно быть уникальным для каждой записи.
* Первичный ключ не может содержать NULL-значения.
* Первичный ключ выбирается из атрибутов сущности и должен быть минимально возможным набором данных.

Вторичный или внешний ключ — это атрибут (или комбинация атрибутов), который используется для создания связи между двумя сущностями. Внешний ключ в одной сущности ссылается на первичный ключ другой сущности.

Роли внешнего ключа:

* Устанавливает логическую связь между таблицами.
* Обеспечивает целостность данных, гарантируя, что значение внешнего ключа существует в связанной таблице.

Связи в ERD описывают, как сущности взаимодействуют друг с другом. Они отображаются линиями, соединяющими сущности. Связи также включают информацию о типе взаимодействия и количестве объектов, участвующих в связи.

Типы связей:

* Один-к-одному (1:1): Каждая запись одной сущности связана с одной записью другой сущности. Пример: один человек — один паспорт.
* Один-ко-многим (1: N): Одна запись в одной сущности может быть связана с несколькими записями другой сущности. Пример: один преподаватель ведет несколько курсов.
* Многие-ко-многим (M: N): Несколько записей одной сущности могут быть связаны с несколькими записями другой сущности. Пример: студенты записываются на разные курсы, а курсы посещают разные студенты.

Кардинальность связей:

* Кардинальность определяет количество записей, которые могут быть связаны:
* Минимальная кардинальность: указывает минимальное количество объектов, необходимых для участия в связи.
* Максимальная кардинальность: указывает максимальное количество объектов, которые могут быть связаны.

Промежуточные таблицы:

Для реализации связи "многие-ко-многим" создаются промежуточные таблицы, которые включают внешние ключи обеих связанных сущностей.

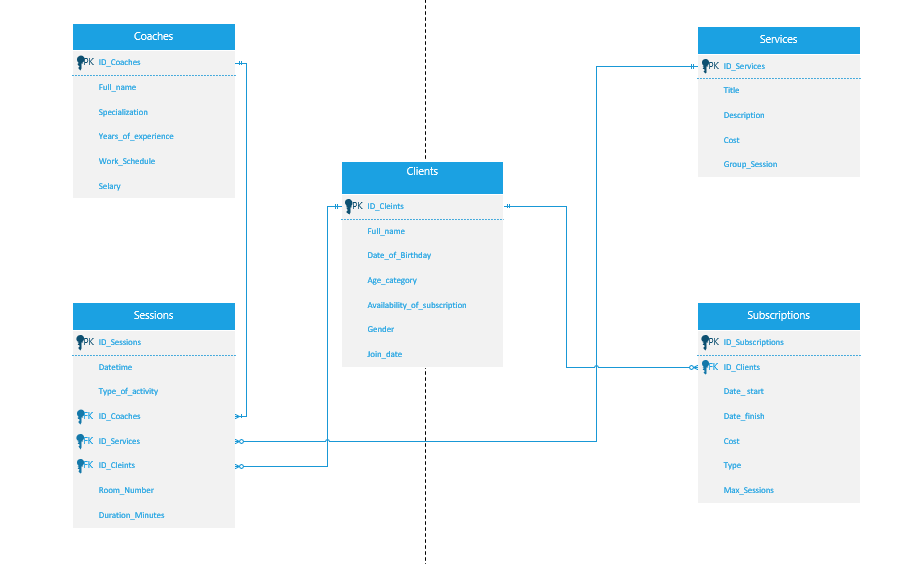


Рисунок 2.1. Диаграмма ERD

Сущности и их атрибуты

1. Clients (Клиенты)

* ID\_Clients (PK): Уникальный идентификатор клиента.
* Full\_name: Полное имя клиента.
* Date\_of\_Birthday: Дата рождения клиента.
* Age\_category: Возрастная категория клиента.
* Availability\_of\_subscription: Доступность абонемента (активен/неактивен).
* Gender: Пол клиента.
* Join\_date: Дата вступления в клуб.

1. Coaches (Тренеры)

* ID\_Coaches (PK): Уникальный идентификатор тренера.
* Full\_name: Полное имя тренера.
* Specialization: Специализация тренера.
* Years\_of\_experience: Количество лет опыта.
* Work\_Schedule: График работы.
* Salary: Зарплата тренера.

1. Services (Услуги)

* ID\_Services (PK): Уникальный идентификатор услуги.
* Title: Название услуги.
* Description: Описание услуги.
* Cost: Стоимость услуги.
* Group\_Session: Флаг или индикатор, определяющий, является ли услуга групповой.

1. Sessions (Занятия)

* ID\_Sessions (PK): Уникальный идентификатор занятия.
* Datetime: Дата и время занятия.
* Type\_of\_activity: Тип активности.
* ID\_Coaches (FK): Связь с тренером, проводящим занятие.
* ID\_Services (FK): Связь с предоставляемой услугой.
* ID\_Clients (FK): Связь с клиентом, участвующим в занятии.
* Room\_Number: Номер комнаты или зала, где проводится занятие.
* Duration\_Minutes: Продолжительность занятия (в минутах).

1. Subscriptions (Абонементы)

* ID\_Subscriptions (PK): Уникальный идентификатор абонемента.
* ID\_Clients (FK): Связь с клиентом, которому принадлежит абонемент.
* Date\_start: Дата начала действия абонемента.
* Date\_finish: Дата окончания действия абонемента.
* Cost: Стоимость абонемента.
* Type: Тип абонемента (например, месячный, годовой).
* Max\_Sessions: Максимальное количество занятий по абонементу.

Связи между сущностями

1. Clients ↔ Subscriptions

* Тип связи: Один-ко-многим (1: N).
* Описание: Один клиент может иметь несколько абонементов. Внешний ключ ID\_Clients, в сущности, Subscriptions связывает абонемент с клиентом.

1. Clients ↔ Sessions

* Тип связи: Один-ко-многим (1: N).
* Описание: Один клиент может участвовать в нескольких занятиях. Внешний ключ ID\_Clients, в сущности, Sessions связывает занятие с клиентом.

1. Coaches ↔ Sessions

* Тип связи: Один-ко-многим (1: N).
* Описание: Один тренер может проводить несколько занятий. Внешний ключ ID\_Coaches, в сущности, *Sessions* связывает занятие с тренером.

1. Services ↔ Sessions

* Тип связи: Один-ко-многим (1: N).
* Описание: Каждая услуга может быть предоставлена в рамках нескольких занятий. Внешний ключ ID\_Services, в сущности, Sessions связывает занятие с услугой.

Описание связей

1. Clients (Клиенты):  
   Клиенты являются основной сущностью, так как они связаны как с абонементами, так и с занятиями. Это позволяет отслеживать, какие услуги и занятия клиент посещает, а также их статус.
2. Coaches (Тренеры):  
   Тренеры проводят занятия, что связывает их с сущностью Sessions. Каждый тренер имеет график работы и специализацию, что упрощает управление их занятостью.
3. Services (Услуги):  
   Услуги определяют, какие типы активностей могут быть предложены клиентам. Они связаны с конкретными занятиями и позволяют оценивать спрос и стоимость.
4. Sessions (Занятия):  
   Занятия являются точкой пересечения между клиентами, тренерами и услугами. Они содержат информацию о времени, месте и типе активности.
5. Subscriptions (Абонементы):  
   Абонементы предоставляют возможность клиентам посещать занятия в определенные периоды времени. Они привязаны к клиентам и позволяют отслеживать их активность.

# 2.2 Словарь данных

**Словарь данных (Data Dictionary)** — это структурированный набор информации, описывающий данные, используемые в системе управления базами данных (СУБД). Он содержит метаданные, которые предоставляют информацию о структуре, типах данных, ограничениях и других характеристиках базы данных. Описывает таблицы, поля (столбцы), их типы данных и связи между таблицами. Содержит информацию о первичных ключах, внешних ключах, индексах и ограничениях. Помогает разработчикам, администраторам баз данных и пользователям понять, как устроены данные и как они используются. Словарь данных фиксирует правила и ограничения, чтобы предотвратить ошибки (например, уникальность ключей, значения NULL).

Таблица 1. Clients

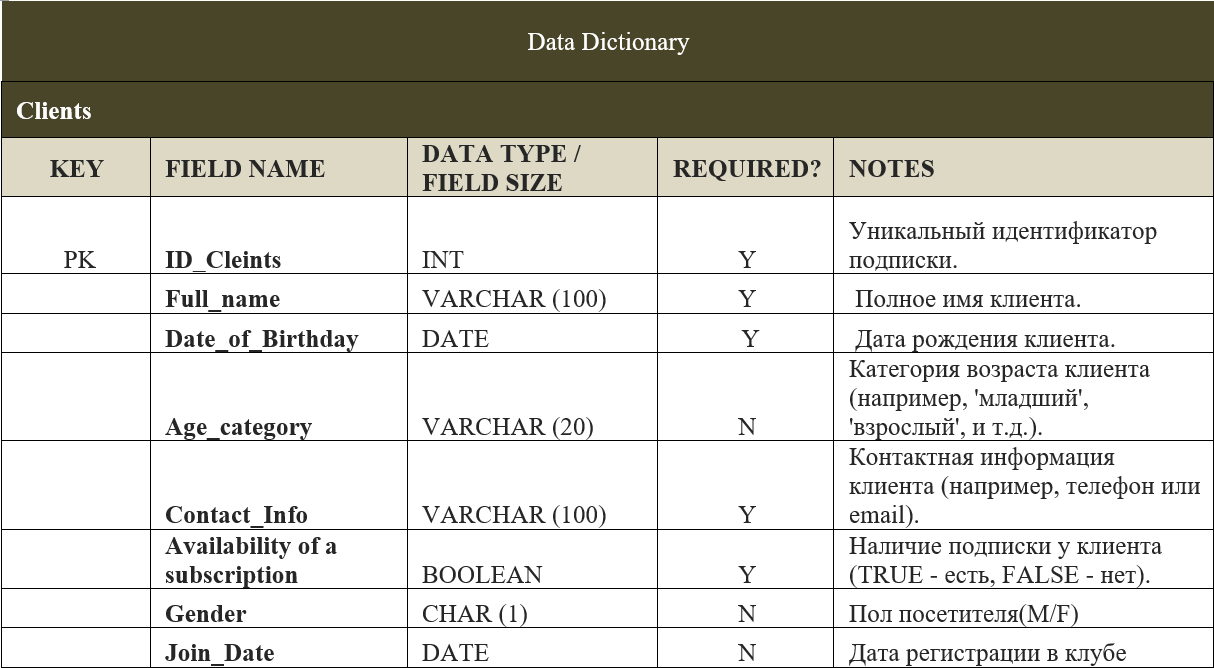


Таблица 2. Coaches

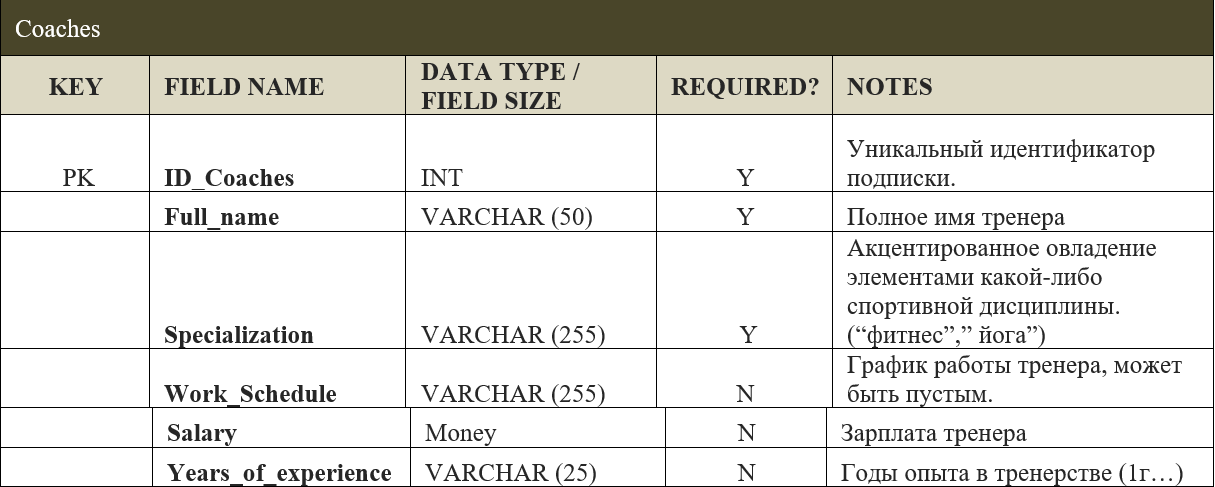


Таблица 3. Subscriptions

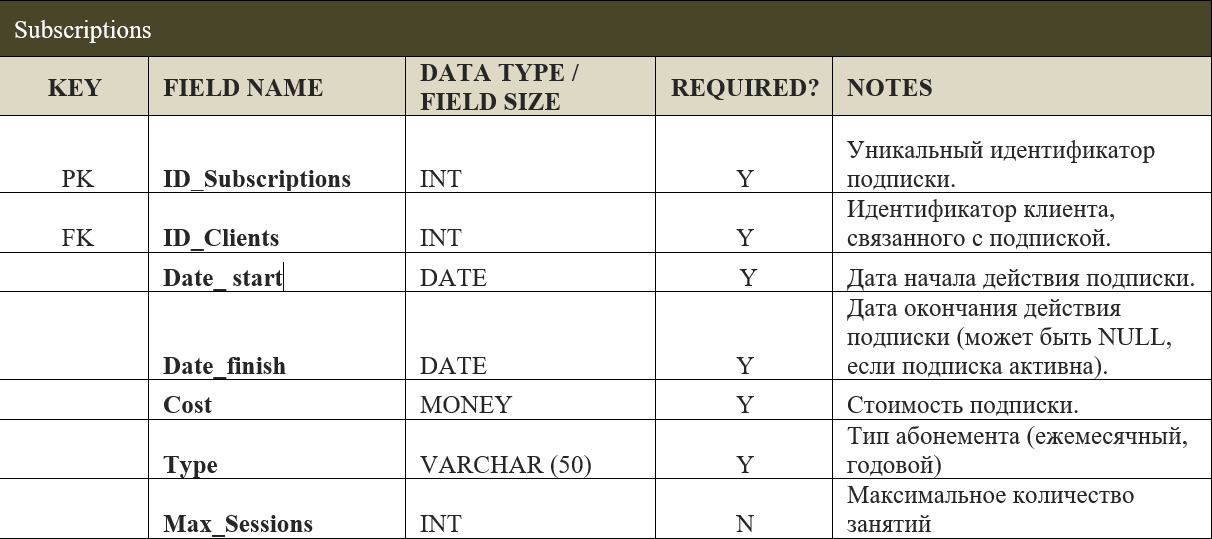


Таблица 4. Sessions

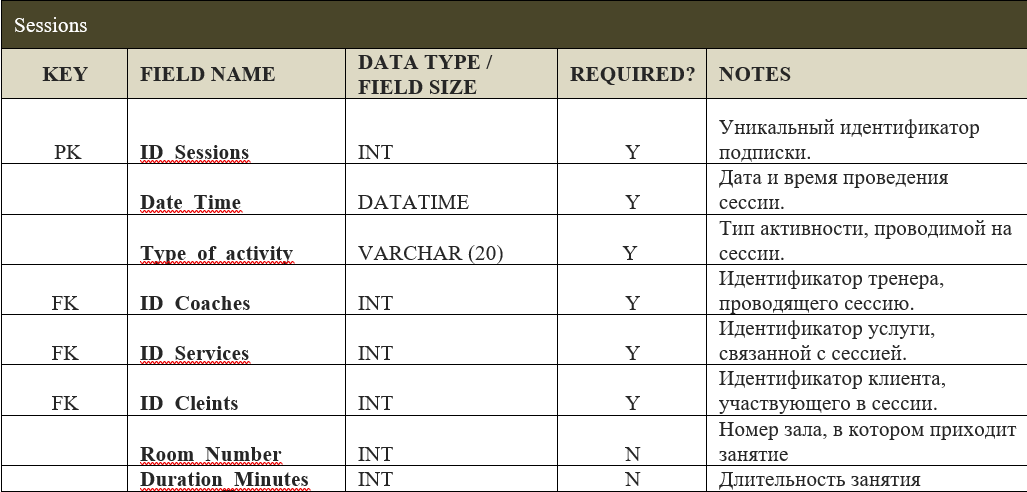
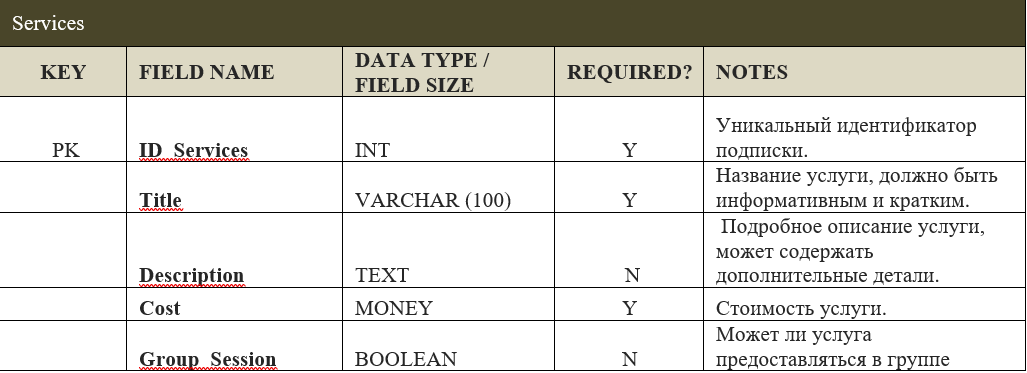


Таблица 5. Services



# ГЛАВА 3. РАЗРАБОТКА БАЗЫ ДАННЫХ

# 3.1 Разработка базы данных

MySQL — это реляционная система управления базами данных (СУБД), которая распространяется как свободное программное обеспечение. Она отличается гибкостью, лёгкостью и удобством в использовании. MySQL работает с языком запросов SQL, который традиционно используется в базах данных.

MySQL имеет клиент-серверную архитектуру, где база данных хранится на сервере, а клиенты — сторонние устройства — связываются с ней для выполнения запросов и получения информации.

MySQL используется бэкенд-разработчиками, разработчиками десктопных и мобильных приложений, администраторами баз данных. Система популярна благодаря своей гибкости, универсальности, высокой производительности и бесплатному доступу.

MySql Workbench — это программное обеспечение для создания и проектирования баз данных с помощью схем и других визуальных средств. Сегодня мы покажем, что это такое, как установить Workbench и подключиться к кластеру, как создавать таблицы и модели, как делать импорт и экспорт данных.

Разработка базы данных начинается с ранее разработанной ERD диаграммой. Откроем MySQL Workbrench подключимся к нашему серверу и вводим пароль.

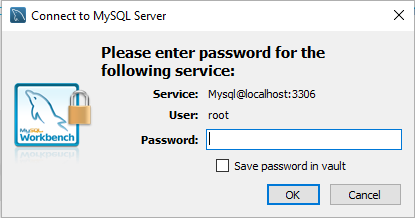


Рисунок 3.1.1 Подключение к серверу

После успешного подключению к локальному серверу переходим к созданию базы данных.

CREATE DATABASE Club;

use club;

CREATE TABLE Clients (

ID\_Client INT AUTO\_INCREMENT PRIMARY KEY,

Full\_Name VARCHAR (50) NOT NULL,

Date\_of\_Birthday DATE NOT NULL,

Age\_Category VARCHAR (20),

Contact\_Info VARCHAR (100),

Availability\_of\_subscription BOOLEAN NOT NULL,

Gender CHAR (1),

Join\_Date DATE

);

CREATE TABLE Coaches (

ID\_Coach INT AUTO\_INCREMENT PRIMARY KEY,

Full\_Name VARCHAR (50) NOT NULL,

Specialization VARCHAR (100) NOT NULL,

Work\_Schedule VARCHAR (255),

Salary DECIMAL (10,2),

Years\_of\_experience VARCHAR (25) NOT NULL

);

CREATE TABLE Services (

ID\_Service INT AUTO\_INCREMENT PRIMARY KEY,

Named VARCHAR (100) NOT NULL,

Descriptions TEXT,

Cost DECIMAL (10,2) NOT NULL,

Group\_Session BOOLEAN

);

CREATE TABLE Sessions (

ID\_Session INT AUTO\_INCREMENT PRIMARY KEY,

ID\_Coach INT NOT NULL,

ID\_Service INT NOT NULL,

ID\_Client INT NOT NULL,

Date\_Time DATETIME NOT NULL,

Session\_Type VARCHAR (20) NOT NULL,

Duration\_Minutes INT,

Room\_Number INT,

FOREIGN KEY (ID\_Coach) REFERENCES Coaches (ID\_Coach) ON DELETE CASCADE,

FOREIGN KEY (ID\_Service) REFERENCES Services (ID\_Service) ON DELETE CASCADE,

FOREIGN KEY (ID\_Client) REFERENCES Clients (ID\_Client) ON DELETE CASCADE

);

CREATE TABLE Subscriptions (

ID\_Subscription INT AUTO\_INCREMENT PRIMARY KEY,

ID\_Client INT NOT NULL,

Start\_Date DATE NOT NULL,

End\_Date DATE NOT NULL,

Cost DECIMAL (10,2) NOT NULL,

Type VARCHAR (50) NOT NULL,

Max\_Sessions INT,

FOREIGN KEY (ID\_Client) REFERENCES Clients (ID\_Client) ON DELETE CASCADE

);

После успешной проверки, нужно заполнить данные в созданных таблицах.

USE club;

INSERT INTO Clients (Full\_Name, Date\_of\_Birthday, Age\_Category, Contact\_Info, Availability\_of\_subscription, Gender, Join\_Date) VALUES

('МухАмедьяров Вячеслав Денисович', '1990-01-15', 'Взрослый', 'myxa@mail.ru', TRUE, 'M', '2023-01-01'),

('Олийнык Юрий Нетотчествович', '1985-05-20', 'Взрослый', 'mrfunny67@ya.ru', TRUE, 'F', '2023-01-02'),

('Матвеев Даниил Александрович', '2000-03-10', 'Молодежь', 'danymat@yaaaho.com', TRUE, 'M', '2023-01-03'),

('Федорова Елизавета Артёмовна', '1995-07-25', 'Взрослый', 'lizkakergyzka@gufun.com', TRUE, 'F', '2023-01-04'),

('Зайцев Константин Максимович', '1988-11-30', 'Взрослый', 'kostya228@fugun.com', false, 'M', '2023-01-05');

INSERT INTO Coaches (Full\_Name, Specialization, Work\_Schedule, Salary, Years\_of\_experience) VALUES

('Алексей Алексеев', 'Тренеры-по-пауэрлифтингу', 'Пн-Пт 9:00-18:00', 50000.00, '15 лет'),

('Ольга Васильева', 'Йога-инструктор', 'Вт-Вс 10:00-19:00', 45000.00, '3 года'),

('Андрей Андреянович', 'Персональный тренер', 'Пн-Сб 8:00-17:00', 60000.00, '7 лет'),

('Анастасия Павловна', 'Пилатес-инструктор', 'Пн-Чт 11:00-20:00', 48000.00, '5 лет'),

('Николай Владимирович', 'Тренер-по-битбоксу', 'Сб-Вс 10:00-15:00', 55000.00, '6 лет');

INSERT INTO Services (Named, Descriptions, Cost, Group\_Session) VALUES

('Тренеровка по паэрлифтингу', 'Занятия в группе под руководством тренера', 1500.00, TRUE),

('Индивидуальная тренировка', 'Персональные занятия с тренером', 3000.00, FALSE),

('Йога для начинающих', 'Основы йоги для новичков', 1200.00, TRUE),

('Пилатес', 'Упражнения для укрепления мышц', 1300.00, TRUE),

('Битбокс', 'Занятие для горловых связок', 2000.00, TRUE);

INSERT INTO Sessions (ID\_Coach, ID\_Service, ID\_Client, Date\_Time, Session\_Type, Duration\_Minutes, Room\_Number) VALUES

('2023-01-10 10:00:00', 'Групповая', 60, 111),

( '2023-01-11 11:00:00', 'Индивидуальная', 45, 52),

('2023-01-12 12:00:00', 'Групповая', 60, 66),

('2023-01-13 13:00:00', 'Групповая', 60, 61),

('2023-01-14 14:00:00', 'Групповая', 60, 106);

INSERT INTO Subscriptions (ID\_Client, Start\_Date, End\_Date, Cost, Type, Max\_Sessions) VALUES

(1, '2023-01-01', '2023-06-01', 10000.00, 'Месячная', 10),

(2, '2023-01-02', '2023-07-01', 12000.00, 'Полугодовая', 20),

(3, '2023-01-03', '2023-08-01', 15000.00, 'Годовая', 30),

(4, '2023-01-04', '2023-09-01', 8000.00, 'Месячная', 10),

(5, '2023-01-05', '2023-10-01', 9000.00, 'Месячная', 10);

После успешного заполнения таблиц. Согласно техническому заданию, процесс заполнения данных должен включать следующие шаги:

1. Выдать список услуг и их стоимость;

SELECT Full\_Name AS ФИО, Specialization AS Специализация, Salary AS Зарплата,

Years\_of\_experience AS Стаж, Work\_Schedule AS График\_работы

FROM Coaches;

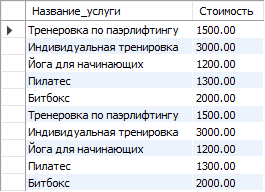


Рисунок 3.1.1 Выдача список услуг и стоимость.

1. Выдать список посетителей с их персональными данными;

SELECT Full\_Name AS ФИО, Specialization AS Специализация, Salary AS Зарплата,

Years\_of\_experience AS Стаж, Work\_Schedule AS График\_работы  
FROM Coaches;

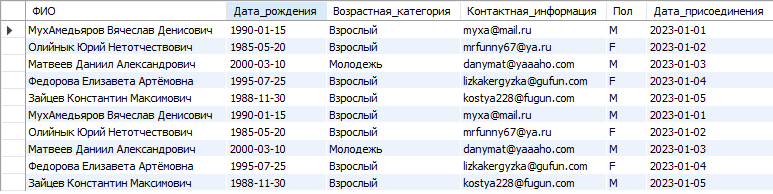


Рисунок 3.1.2 Персональные данные посетителей.

1. Выдать список посетителей по возрастным категориям;

SELECT Full\_Name AS ФИО\_тренера, Work\_Schedule AS График\_работы

FROM Coaches;



Рисунок 3.1.3 Возрастные категории.

1. Рассчитать стоимость предоставленных услуг за определенный промежуток времени;

SELECT

c.Full\_Name AS ФИО\_посетителя,

s.Named AS Название\_услуги,

sess.Date\_Time AS Дата\_и\_время

FROM Sessions sess

JOIN Clients c ON sess.ID\_Client = c.ID\_Client

JOIN Services s ON sess.ID\_Service = s.ID\_Service

WHERE s.Group\_Session = TRUE

ORDER BY sess.Date\_Time;

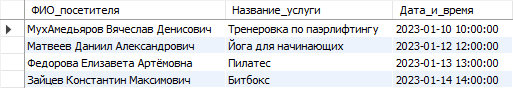


Рисунок 3.1.4 Стоимость услуг за определённый промежуток времени.

1. Вывести список тренеров спортивного клуба с их персональными данными;

SELECT Full\_Name AS ФИО, Specialization AS Специализация, Salary AS Зарплата,

Years\_of\_experience AS Стаж, Work\_Schedule AS График\_работы

FROM Coaches;



Рисунок 3.1.5 Список тренеров с их персональными данными.

1. Сформировать график работы тренеров;

SELECT Full\_Name AS ФИО, Specialization AS Специализация, Salary AS Зарплата,

Years\_of\_experience AS Стаж, Work\_Schedule AS График\_работы

FROM Coaches;

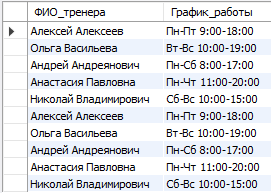


Рисунок 3.1.6 График работ тренеров.

1. Выдать списки на групповые занятия;

SELECT s.Named AS Групповые\_занятия, GROUP\_CONCAT(c.Full\_Name SEPARATOR ', ') AS Участники

FROM Sessions sess

JOIN Services s ON sess.ID\_Service = s.ID\_Service

JOIN Clients c ON sess.ID\_Client = c.ID\_Client

WHERE s.Group\_Session = TRUE

GROUP BY s.Named;

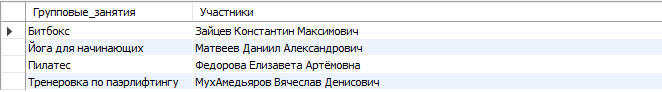


Рисунок 3.1.7 Список групповых занятий.

1. Выдать списки посетителей, имеющих абонемент;

SELECT s.Named AS Название\_услуги, GROUP\_CONCAT(c.Full\_Name SEPARATOR ', ') AS Участники

FROM Sessions sess

JOIN Services s ON sess.ID\_Service = s.ID\_Service

JOIN Clients c ON sess.ID\_Client = c.ID\_Client

WHERE s.Group\_Session = TRUE

GROUP BY s.Named;

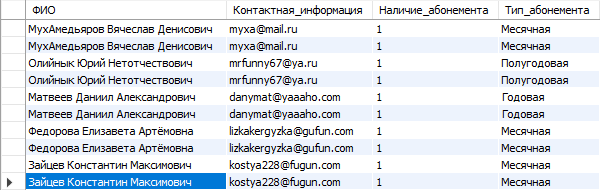


Рисунок 3.1.8 Список посетителей имеющий абонемент.

1. Рассчитать стоимость услуг по каждому тренеру;

SELECT co.Full\_Name AS ФИО\_тренера, SUM(s.Cost) AS Общая\_стоимость\_услуг

FROM Sessions sess

JOIN Coaches co ON sess.ID\_Coach = co.ID\_Coach

JOIN Services s ON sess.ID\_Service = s.ID\_Service

GROUP BY co.Full\_Name;

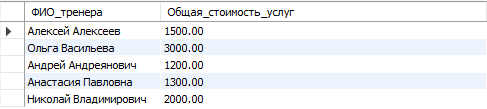


Рисунок 3.1.9 Стоимость услуг по каждому тренеру.

1. Рассчитать стоимость услуг, потребленных каждым посетителем.

SELECT

c.Full\_Name AS ФИО\_посетителя,

SUM (s.Cost) AS Общая\_стоимость\_услуг

FROM Sessions sess

JOIN Clients c ON sess.ID\_Client = c.ID\_Client

JOIN Services s ON sess.ID\_Service = s.ID\_Service

GROUP BY c.Full\_Name

ORDER BY Общая\_стоимость\_услуг DESC;

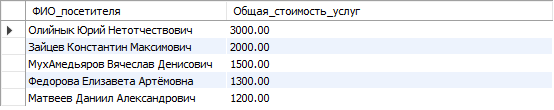


Рисунок 3.1.10 Услуги, которые взял каждых потребитель.

# 3.2 Введение в безопасность SQL Server

1. Уровни безопасности SQL Server

SQL Server предоставляет несколько уровней защиты:

1. Безопасность на уровне сервера:

* Контролирует доступ к самому серверу SQL.
* Создаются логины (logins), которые разрешают или запрещают подключение.

1. Безопасность на уровне базы данных:

* Управляет доступом к отдельным базам данных.
* Создаются пользователи базы данных (users), связанные с логинами.

1. Безопасность на уровне объектов:

* Определяет доступ к конкретным объектам, например, таблицам, представлениям, хранимым процедурам.

1. Безопасность на уровне строк и столбцов (Row-Level Security, Dynamic Data Masking):

* Позволяет управлять видимостью отдельных строк или маскировать чувствительные данные.

2. Аутентификация в SQL Server

SQL Server поддерживает два основных режима аутентификации:

1. Windows-аутентификация (рекомендуется):

* Использует учетные записи и группы Windows.
* Не требует ручного ввода пароля при подключении, если используется ваша учетная запись Windows.
* Позволяет интегрировать SQL Server с доменами Active Directory.

1. Смешанная аутентификация (SQL Server + Windows):

* Включает поддержку логинов SQL Server (логин и пароль хранятся в базе).
* Используется для пользователей и приложений, которые не находятся в домене Windows.

3. Принципы управления доступом

1. Роли безопасности:

* Серверные роли: например, sysadmin, dbcreator, bulkadmin. Они управляют разрешениями на уровне всего сервера.
* Роли базы данных: например, db\_owner, db\_datareader, db\_datawriter. Управляют доступом внутри конкретной базы данных.
* Пользователи получают доступ через назначение ролей, а не прямую выдачу разрешений.

1. Принцип наименьших привилегий (POLP):

* Пользователи должны получать минимальный набор прав, достаточный для выполнения их задач.

1. GRANT, DENY, REVOKE:

* GRANT: Предоставление разрешений.
* DENY: Запрет на выполнение действий (перебивает все GRANT).
* REVOKE: Удаление ранее выданного GRANT или DENY.

4. Шифрование данных

1. Transparent Data Encryption (TDE):

* Шифрует данные на уровне хранилища, чтобы защитить их от кражи файлов базы данных.

1. Always Encrypted:

* Шифрует данные на стороне клиента. Даже администраторы SQL Server не могут видеть данные в открытом виде.

1. SSL/TLS для шифрования соединений:

* Шифрует данные во время передачи между клиентом и сервером.

5. Защита данных

1. Row-Level Security (RLS):

* Позволяет управлять доступом к отдельным строкам в таблице в зависимости от условий (например, только свои данные).

1. Dynamic Data Masking:

* Маскирует данные, например, скрывает часть информации (например, 1234-5678-\*\*\*\* для номеров кредитных карт).

1. Auditing (аудит):

* Отслеживание всех изменений в базе данных:
* Включение SQL Server Audit для логирования событий, таких как вход пользователей или изменения данных.
* Использование журналов ошибок (Error Logs) и событий Windows.

6. Защита от атак

1. Защита от SQL-инъекций:

* Используйте параметры вместо конкатенации строк в запросах.
* Пример: EXEC sp\_executesql N'SELECT \* FROM Users WHERE Username = @Username', N'@Username nvarchar(50)', @Username = N'admin';

1. Защита от грубой силы:

* Включите защиту паролей: политика сложности, длина пароля, срок действия.
* Используйте защиту от блокировки логинов (Account Lockout).

1. Файрвол SQL Server:

* Открывайте доступ только с определённых IP-адресов.

1. Обновления:

* Регулярно устанавливайте обновления и исправления безопасности.

7. Резервное копирование и восстановление

* Настройте шифрование резервных копий (WITH ENCRYPTION).
* Используйте безопасные каталоги для хранения .bak файлов.

8. Мониторинг и аудит

1. Server Audit:

* Позволяет отслеживать действия пользователей, такие как попытки входа, изменения данных, выполнение запросов.
* Настраивается через CREATE SERVER AUDIT.

1. Extended Events и Triggers:

* Используйте для мониторинга подозрительной активности.

1. Dynamic Management Views (DMVs):

* Просмотр текущих соединений и выполнения запросов:

SELECT \* FROM sys.dm\_exec\_sessions;

9. Полезные T-SQL-команды для безопасности

* Создание логина:

CREATE LOGIN MyUser WITH PASSWORD = 'StrongPassword123';

* Создание пользователя базы данных:

CREATE USER MyUser FOR LOGIN MyUser;

* Назначение роли:

EXEC sp\_addrolemember 'db\_datareader', 'MyUser';

# 3.3 Управление безопасностью уровня сервера

1. Создание логинов и пользователей

Откройте SSMS и подключитесь к серверу.

Перейдите в раздел Security> Logins:

* + - * Щелкните правой кнопкой мыши на Logins и выберите New Login....
      * Укажите имя пользователя (воспользуюсь именование, AdminUser, TrainerUser и ClientUser).
      * В разделе Authentication выберите SQL Server Authentication и установите пароль.
  + Отключите опцию Enforce password policy (если тестовая среда, иначе оставьте включенной для безопасности).

1. Перейдите в раздел User Mapping:

Поставьте галочку напротив вашей базы данных. “CLUB”

Назначьте роли для пользователя:

* + - Для администратора: роль db\_owner.
    - Для тренера: роли db\_datareader и db\_datawriter.
    - Для клиента: роль db\_datareader.
* Нажмите OK для сохранения настроек.

1. Настройка ролей и прав доступа

**Для администратора:**

1. В SSMS создайте нового логина AdminUser:
   * В меню New Login введите имя AdminUser и пароль.
2. В разделе User Mapping:
   * Выберите вашу базу данных.
   * Установите роль db\_owner (полный доступ к базе).
   * Login-AdminUser
   * Pass-AdmIn

**Для тренера:**

1. Создайте нового логина TreinerUser:
   * + Укажите имя ManagerUser и пароль.
2. В разделе User Mapping:
   * Выберите вашу базу данных.
   * Установите роли:
     + db\_datareader (для чтения данных).
     + db\_datawriter (для добавления и изменения данных).
     + Login-ManagerUser
     + Pass-MAnAgEr

**Для клиента:**

1. Создайте нового логина ClientUser:
   * + - Укажите имя ClientUser и пароль.
2. В разделе-User Mapping:
   * Выберите вашу базу данных.
   * Установите только роль db\_datareader (только чтение).
   * Login-ClientUser
   * Pass-123

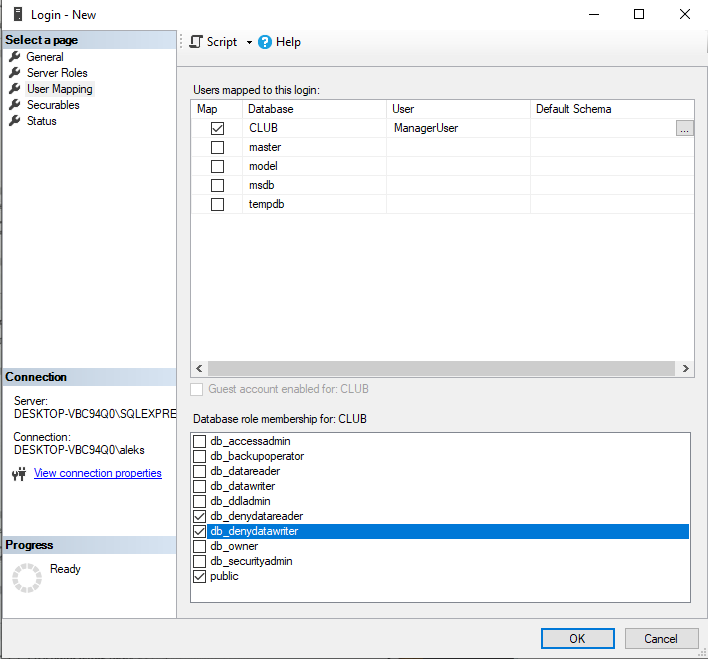


Рисунок 3.2.1 Пример работы с добавлением пользователей (Тренер)

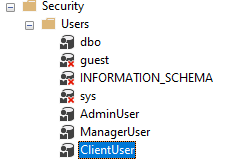


Рисунок 3.2.2 Успешное добавление пользователей

# Управление участниками уровня базы данных

Создание трёх пользователей Admin, Trainer и User на уровне сервера, для дальнейшего подключение их к базе данных

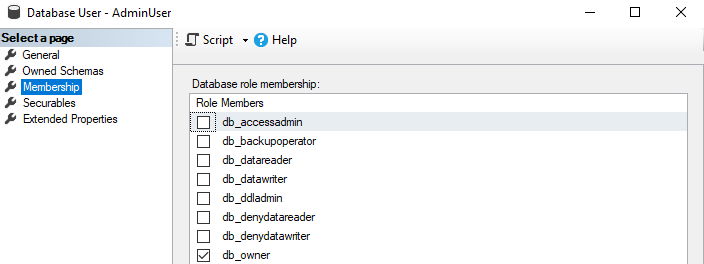


Рисунок 3.4.1 Созданные пользователи и их уровень в базе данных

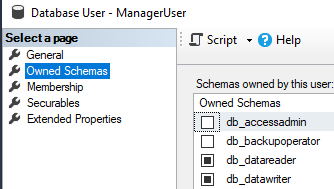


Рисунок 3.4.2 Созданные пользователи и их уровень в базе данных

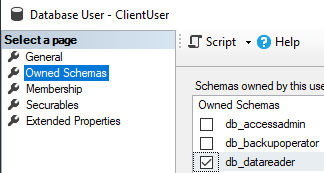


Рисунок 3.4.3 Созданные пользователи и их уровень в базе данных

# Управление разрешениями уровня базы данных

Чтобы легко управлять разрешениями в базах данных, SQL Server предоставляет несколько ролей, которые являются субъектами безопасности, которые группируют другие субъекты. Они похожи на *группы* в операционной системе Windows. Разрешения ролей уровня базы данных распространяются на всю базу данных.

Нам понадобиться нажать правой кнопкой мыши на свою Базу Данных

“Club” далее Properties в “Select a page” выбираем Permissions

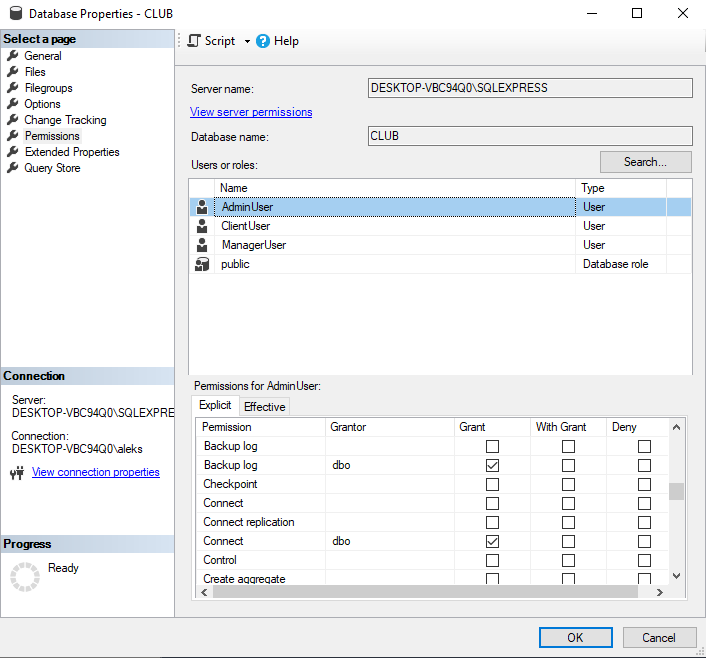


Рисунок 3.5.1 Выдача прав пользователей

Установи разрешения:

Для AdminUser:

* + Установи флажки на SELECT, INSERT, UPDATE, DELETE, EXECUTE.

Для TrainerUser:

* + Установи флажки на SELECT, INSERT, UPDATE.

Для ClientUser:

* + Установи флажок только на SELECT.

# 3.6 Резервное копирование баз данных

Резервное копирование (англ. backup copy) — процесс создания копии данных на носителе (жёстком диске, дискете и т. д.), предназначенном для восстановления данных в оригинальном или новом месте их расположения в случае их повреждения или разрушения.

Создание Backup copy для базы данных Спортивный клуб. В Object Explorer нажав “ПКМ” на свою базу данных появится меню взаимодействия с вашей базой данных там находим “Task” &” Backup”

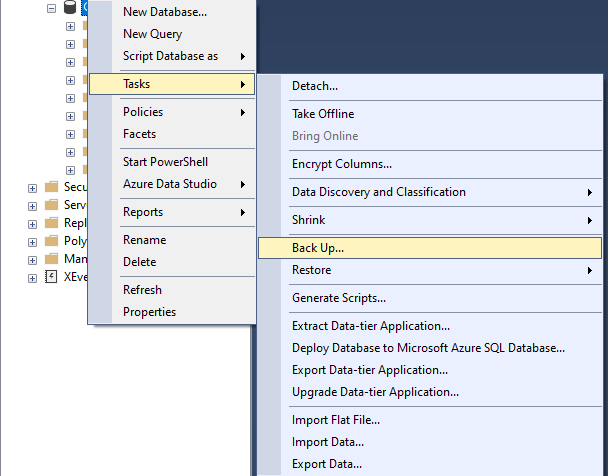


Рисунок 3.6.1 Выполнение резервной копии через использование интерфейс

После открывается окно ничего не меняем и нажимает “ОК”.

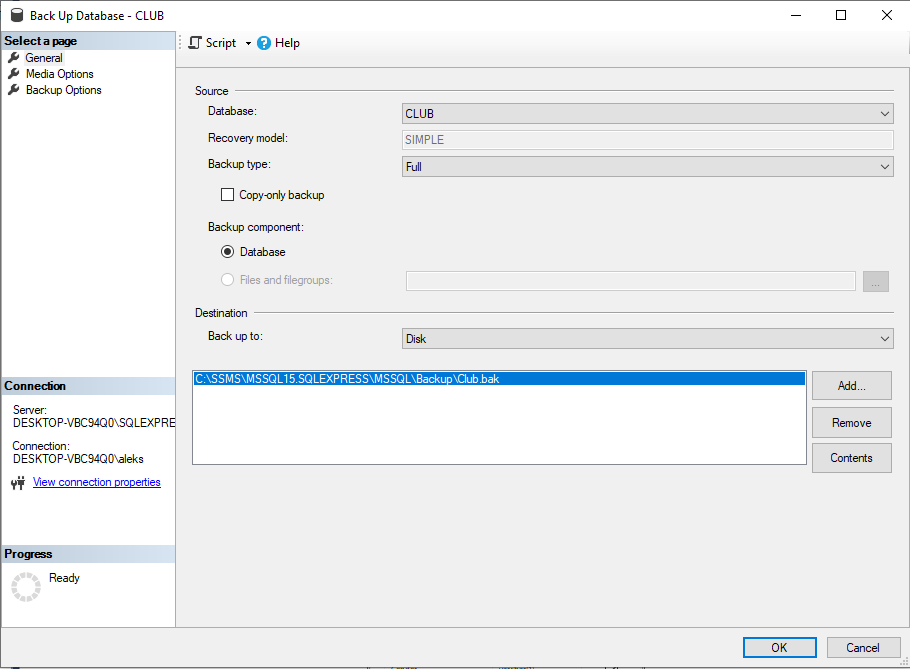


Рисунок 3.6.2 Выполнение резервной копии через использование интерфейс

Далее переходим по пути и копируем в удобное место для хранения.



Рисунок 3.6.2 Файл резервной копии

Создание Backup для базы данных Club.

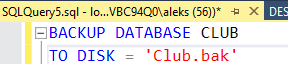


Рисунок 3.2.1 Скрипт для выполнения резервного копирования

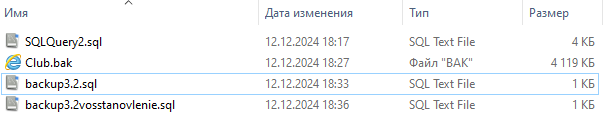


Рисунок 3.2.2 Файлы восстановления

Запрос для восстановления файла с BackUp нашей базы данных.

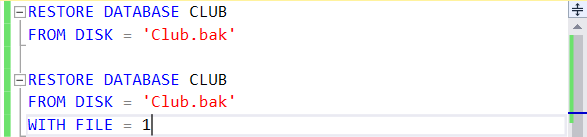


Рисунок 3.2.3 Скрипт для восстановления базы данных

# Шифрование данных баз данных

Шифрование данных в базе данных играет ключевую роль в обеспечении безопасности и конфиденциальности информации. Оно защищает данные от несанкционированного доступа, даже если злоумышленник получает физический доступ к базе данных. Шифрование представляет собой процесс преобразования данных в нечитабельный формат, который можно расшифровать только с использованием определённого ключа или метода.

Защита конфиденциальности данных

Шифрование позволяет скрыть конфиденциальную информацию, такую как пароли пользователей, личные данные, финансовая информация и т. д. Даже если кто-то получит доступ к файлам базы данных или их резервным копиям, данные останутся защищёнными и нечитаемыми без ключа дешифровки.

Пример: В базе данных могут храниться номера кредитных карт, медицинские записи или логины и пароли пользователей. Шифрование этих данных гарантирует, что никто, кроме авторизованных лиц, не сможет их прочитать.

Соответствие нормативным требованиям и стандартам безопасности

Многие отрасли и организации обязаны соблюдать требования безопасности данных, установленные государственными или международными стандартами. Например:

* GDPR (Общий регламент защиты данных) в Европе требует защиты личных данных.
* PCI DSS (Стандарт безопасности данных индустрии платежных карт) регулирует, как должны храниться и обрабатываться данные о кредитных картах.

Шифрование данных помогает организациям соответствовать таким требованиям, гарантируя защиту конфиденциальных данных.

3. Защита данных при утечке или краже

В случае кражи или утечки данных, шифрование снижает риски, так как даже если данные попадут в руки злоумышленников, они не смогут их использовать без соответствующего ключа для расшифровки. Это особенно важно в случае физического доступа к серверам базы данных (например, при краже жесткого диска).

4. Минимизация ущерба при нарушении безопасности

Если система безопасности базы данных будет скомпрометирована (например, атакующий получит доступ к данным на уровне приложения или сервера), зашифрованные данные будут оставаться защищёнными. Даже если атакующий получит доступ к файлам базы данных или резервным копиям, шифрование может предотвратить полный доступ к данным.

5. Защита данных в процессе передачи

Шифрование данных также важно для защиты информации, передаваемой через сети (например, между сервером базы данных и клиентами). Использование SSL/TLS или других методов шифрования сетевого трафика помогает предотвратить утечку данных в процессе их передачи между клиентом и сервером.

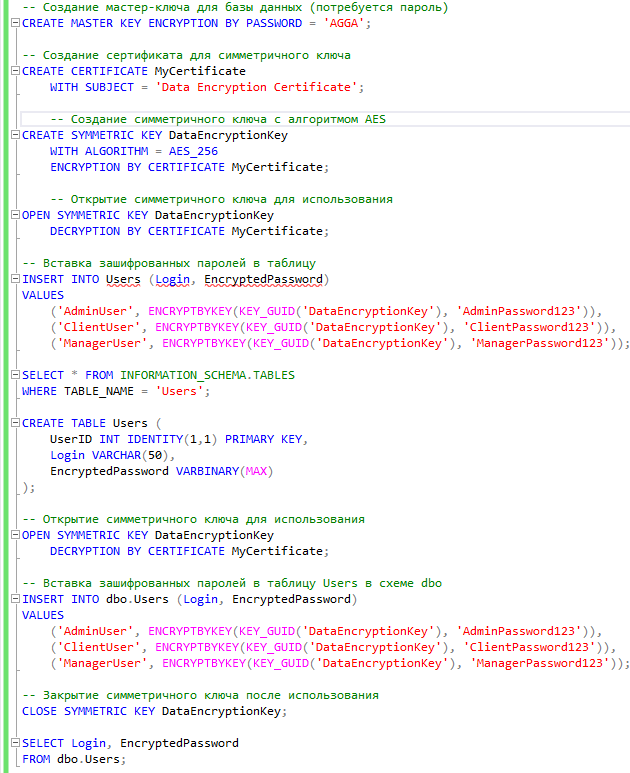


Рисунок 3.7.1 Запрос на шифрование данных

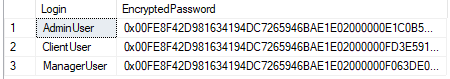


Рисунок 3.7.2 Зашифрованные данные

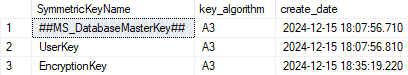


Рисунок 3.7.3 Шифрование зашифрованных данных

# ЗАКЛЮЧЕНИЕ

В ходе выполнения курсового проекта была реализована база данных, обеспечивающая решение поставленных задач и полностью соответствующая требованиям технического задания (ТЗ). Разработанная система оптимизирована для эффективного хранения, обработки и защиты данных, что значительно повышает её практическую ценность и удобство использования.

В процессе работы были внедрены следующие ключевые решения, которые внесли значительный вклад в функциональность и безопасность системы:

Была разработана гибкая система разграничения прав доступа пользователей, что обеспечивает как высокий уровень безопасности, так и удобство работы с базой данных. Каждый пользователь имеет доступ исключительно к тем операциям, которые необходимы ему в рамках функционала, что позволяет минимизировать риски и упростить управление. Данный подход делает систему устойчивой к несанкционированным изменениям и злоупотреблениям.

Для защиты конфиденциальной информации пользователей было внедрено шифрование данных с использованием алгоритма AES\_256. Персональные данные, а также другая критически важная информация надёжно защищены от внешних угроз. Данное решение соответствует современным стандартам безопасности и гарантирует сохранность данных, даже в случае попыток несанкционированного доступа. Этот компонент обеспечивает соответствие базы данных строгим требованиям защиты информации.

В ходе разработки базы данных были применены оптимизированные подходы к организации хранения и обработки данных. Использование индексов на ключевых полях и проработка запросов обеспечили высокую производительность системы. Это особенно важно для работы с большими объёмами данных и многопользовательских систем, где требуется быстрая обработка запросов и высокая надёжность.

Реализованные компоненты системы прошли тщательное тестирование, что позволило устранить возможные уязвимости и ошибки. Система доказала свою устойчивость, надёжность и полное соответствие поставленным требованиям. Были смоделированы различные сценарии работы, чтобы убедиться в её корректности при обработке данных.

Проект успешно выполнен, и созданная база данных соответствует всем требованиям технического задания. Реализованные механизмы шифрования, управления доступом и оптимизации данных гарантируют высокую производительность и надёжность системы. Разработанная база данных демонстрирует возможности современных технологий защиты информации, обеспечивая не только удобство работы, но и соответствие стандартам безопасности.

Проект показывает готовность системы к практическому применению, а также её способность эффективно обрабатывать данные, минимизировать риски и обеспечивать защиту конфиденциальной информации.

Ссылка на репозиторий: [SashaDibiloid/KursovoyProjectDB: "Sportclub"](https://github.com/SashaDibiloid/KursovoyProjectDB)

# СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ

1. Безруков И.А., Сальников А.И., Яковлев В.А., Вылегжанин А.В. Анализ надежности программного отказоустойчивого массива при организации системы долговременного хранения данных радио интерферометрии со сверхдлинными базами // Приборы и техника эксперимента. 2022. № 2. С. 37-42.
2. Божко, В.П. Информатика: данные, технология, маркетинг / В.П. Божко, В.В. Брага, Н.Г. Бубнова. — М.: Финансы и статистика, 2014. — 224 c.

Интернет-ресурсы

1. Базы данных и начало работы с SQL // METANIT - Режим доступа: https://metanit.com/sql/mysql/1.2.php
2. Безопасность в базах данных // Хабр - Режим доступа: https://habr.com/ru/companies/otus/articles/732850/
3. Графический клиент MySQL Workbench // METANIT - Режим доступа: https://metanit.com/sql/mysql/1.3.php
4. Консольный клиент MySQL Command Line Client // METANIT - Режим доступа: https://metanit.com/sql/mysql/1.2.php
5. Типы данных MySQL // METANIT Режим доступа: https://metanit.com/sql/mysql/2.3.php
6. Что такое [SQL Server Management Studio (SSMS) - SQL Server Management Studio (SSMS) | Microsoft Learn](https://learn.microsoft.com/ru-ru/sql/ssms/sql-server-management-studio-ssms?view=sql-server-ver16) – Режим работы: https://learn.microsoft.com/ru-ru/sql/ssms/sql-server-management-studio-ssms?view=sql-server-ver16