**Федеральное государственное автономное образовательное учреждение высшего образования**

**«Национальный исследовательский университет «Высшая школа экономики» Факультет социальных наук**

**О Т Ч Е Т**

**по дисциплине**

**«Проектирование Баз Данных»**

**Выполнили студенты гр. МСПСЦА**

**Кузнецова Полина МСПСЦ232**

**Маркова Елизавета МСПСЦ232**

**Пестерева Дарина МСПСЦ232**

**Чаленко Варвара МСПСЦ232**

***(подпись)***

**Проверил:**

*(должность, ФИО руководителя практики)*

*28.12.2024*

*(дата)*

**2024 год**

Содержание

[Цели и содержание проекта 2](#_Toc186252470)

[Концептуальное проектирование 3](#_Toc186252471)

[*Краткое описание предметной области* 3](#_Toc186252472)

[*Построение инфологической модели* 3](#_Toc186252473)

[*ER-диаграмма с комментариями* 4](#_Toc186252474)

[Проектирование реляционной модели 5](#_Toc186252475)

[*Процесс перехода к реляционной модели* 5](#_Toc186252476)

[*Диаграмма реляционной модели* 10](#_Toc186252477)

[*Механизмы обеспечения целостности данных* 10](#_Toc186252478)

[*Триггеры* 13](#_Toc186252479)

[Развертывание базы данных 13](#_Toc186252480)

[*Создание структуры базы данных (DDL-скрипт)* 13](#_Toc186252481)

[*Заполнение базы данных (DML-скрипт)* 14](#_Toc186252482)

[*Контроль работы ограничений целостности* 15](#_Toc186252483)

[Разработка клиентского приложения 17](#_Toc186252484)

[*Тестовое приложение и реализованные сценарии использования* 17](#_Toc186252485)

[*Примеры ввода и редактирования данных в БД* 17](#_Toc186252486)

[*Запросы и отчёты* 18](#_Toc186252487)

[*Примеры комплексной инфографики* 22](#_Toc186252488)

[Заключение 25](#_Toc186252489)

# **Цели и содержание проекта**

Лига чемпионов УЕФА является одним из самых престижных и популярных футбольных соревнований в мире. Это турнир, в котором принимают участие крупнейшие клубы и лучшие игроки со всей Европы.

Лига чемпионов УЕФА – это еще и турнир, богатый данными: здесь можно найти массу информации о матчах, командах, игроках и других аспектах соревнований.

Эти данные можно использовать для получения ценных сведений о Лиге чемпионов УЕФА и футболе в целом. Например, данные можно использовать для выявления тенденций и закономерностей в соревновании, для оценки эффективности команд и игроков, а также для разработки новых стратегий и тактик.

Мы считаем, что база данных Лиги чемпионов на языке SQL будет являться ценным инструмент для пользователей, которые хотят выявить инсайты и информацию о соревнованиях, командах, игроках основанную на данных. SQL сейчас один из основных языков программирования, который используется для взаимодействия с реляционными базами данных и манипулирования данными в них. Такая база данных позволит пользователям хранить и запрашивать данные о соревнованиях, например, результаты матчей, статистику команд и игры игроков.

Также эти данные можно использовать для создания отчетов, визуализаций и т.д. Например, пользователь может использовать базу данных для создания отчета о наиболее результативных командах в соревновании или для определения наиболее эффективных стилей игры, наиболее эффективных игроках (с финансовой и физической точки зрения) на основе данных. SQL-база данных Лиги чемпионов стала бы ценным ресурсом для различных заинтересованных сторон, включая:

* футбольных болельщиков, которые могли бы использовать базу данных, чтобы узнать больше о своих любимых командах и игроках;
* журналистов и комментаторов, которые могли бы использовать базу данных для подготовки материалов и аналитических материалов, основанных на данных;
* тренеров и менеджеров клубов, которые могут использовать базу данных для анализа игр своих команд и соперников;
* футбольных аналитиков, которые могут использовать базу данных для выявления талантливых игроков и разработки новых стратегий и тактик.

Таким образом, *целью* *проекта* является создание базы данных Лиги чемпионов УЕФА и предоставления мощного инструмента для анализа и хранения данных, связанных с одним из самых популярных футбольных турниров в мире. *Технической целью* является разработка базы данных, которая будет полезна различным заинтересованным сторонам: болельщикам, журналистам, тренерам, менеджерам клубов и аналитикам.

# **Концептуальное проектирование**

## ***Краткое описание предметной области***

Лига чемпионов УЕФА — один из крупнейших и престижных турниров в мире футбола, где участвуют лучшие команды и игроки Европы. Данные о соревновании включают информацию о матчах, командах, игроках, тренерах, судьях, спонсорах и стадионах, поскольку эти данные важны для анализа результатов, стратегий, выступлений команд и игроков.

*Основные задачи*:

* хранение структурированных данных о турнире;
* поддержка аналитических запросов;
* возможность создания отчетов и визуализаций для различных стейкхолдеров.

*Данные проекта***:** База данных построена на основе таблиц, содержащих сведения о турнирах Лиги Чемпионов УЕФА с 2016 по 2022 годы. Первоначально данные были представлены в нескольких таблицах Excel, которые затем были загружены в базу данных SQLite с использованием Python.

Основное внимание уделено возможности выполнения аналитических запросов, например, подсчета голов, анализа трансферов игроков, поиска победителей турниров.

## ***Построение инфологической модели***

Выделено 9 основных сущностей:

1. **Команды (Teams):** включают информацию о названиях, владельцах, бюджетах, тренерах и стадионах.
2. **Игроки (Players):** содержат демографическую информацию, статистику, позиции, трансферы.
3. **Матчи (Matches):** включают данные о времени, стадионе, командах, счете, картах.
4. **Турниры (Tournaments):** информация о сезонах, победителях, участниках и спонсорах.
5. **Голы (Goals):** описание забитых голов, время, авторы и ассистенты.
6. **Стадионы (Stadiums):** характеристика мест проведения матчей.
7. **Тренеры (Coaches):** сведения о тренерах, их национальности и командах.
8. **Арбитры (Referees):** информация о судьях и их опыте.
9. **Спонсоры (Sponsors):** данные о компаниях, поддерживающих турниры и команды.

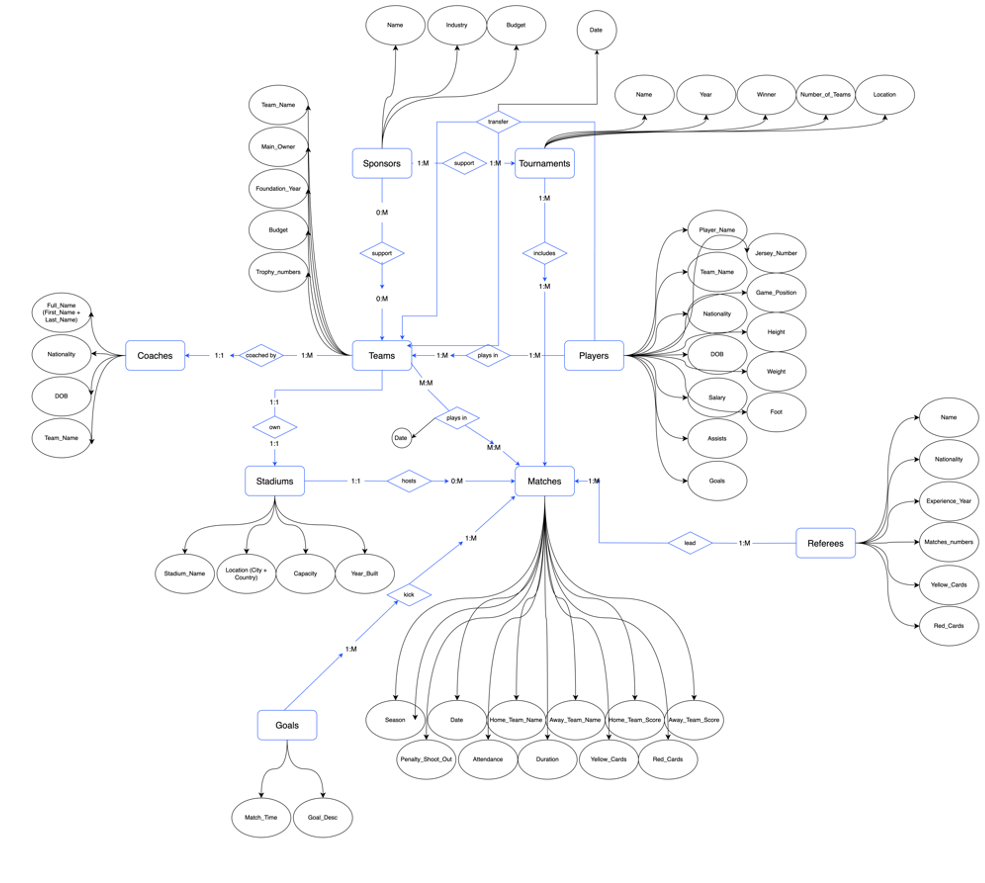
Для построения инфологической модели:

1. Были выделены ключевые сущности;
2. Определены связи между сущностями:

* Связь «один ко многим» между командами и игроками: одна команда может включать множество игроков.
* Связь «один ко многим» между турнирами и матчами: один турнир включает множество матчей.
* Связь «многие ко многим» между матчами и командами: один матч включает две команды (домашнюю и гостевую).

1. Выделены ключевые атрибуты для каждой сущности, включая первичные и внешние ключи.

## ***ER-диаграмма с комментариями***



[ER-диаграмма](https://app.diagrams.net/#G1CmbxPvvWPspex04c64AZfJ2xXRph5HL8)

Диаграмма отражает структуру базы данных, предназначенной для анализа данных Лиги чемпионов УЕФА, и включает основные сущности, их атрибуты и взаимосвязи. В центре модели находятся **матчи (Matches)**, которые объединяют команды, игроков, судей и стадионы. Каждая **команда (Teams)** может быть связана с одним или несколькими **тренерами (Coaches)**, которые руководят её действиями, и **игроками (Players)**, участвующими в матчах. Команды также поддерживаются **спонсорами (Sponsors)**, которые предоставляют финансовую поддержку не только самим командам, но и **турнирам (Tournaments)**.

Каждый матч проводится на **стадионе (Stadiums)**, который может принимать множество матчей, но один матч может быть привязан только к одному стадиону. **Судьи (Referees)** играют ключевую роль, обслуживая матчи и фиксируя события игры, такие как количество желтых и красных карточек, **голы (Goals)**, а также результативные передачи.

**Турниры (Tournaments)** состоят из множества матчей, объединяя команды из разных стран. Спонсоры играют важную роль в поддержке турниров, команд и иногда отдельных игроков. **Игроки (Players)** могут переходить между командами через трансферы, которые описывают переходы игроков, включая дату трансфера и сумму сделки.

Диаграмма отражает экосистему Лиги чемпионов УЕФА, обеспечивая анализ данных о турнирах, матчах, командах, игроках и их статистике. Взаимосвязи между сущностями продуманы, чтобы обеспечить согласованность и поддерживать детальный анализ данных.

# **Проектирование реляционной модели**

## ***Процесс перехода к реляционной модели***

1. **Преобразование сущностей в таблицы:**

Каждая сущность из ER-диаграммы преобразована в таблицу, где атрибуты сущностей стали столбцами таблиц.

1. **Идентификация первичных ключей (PK):**

Для каждой таблицы был выбран первичный ключ, обеспечивающий уникальную идентификацию записей. Например:

* В таблице Players используется составной первичный ключ (Player\_ID, Transfer\_Year), который обеспечивает уникальность записи для игрока в конкретный год трансфера.
* В таблице Matches используется поле Match\_ID как уникальный идентификатор матча.
* В таблице Goals используется поле Goal\_ID как уникальный идентификатор гола.

1. **Реализация связей:**

Связи между сущностями в ER-диаграмме преобразованы в отношения между таблицами с использованием внешних ключей. Например:

* **Связи "один ко многим" (1:N):**

Одна команда (Teams) может быть связана с несколькими игроками (Players), реализовано через внешний ключ Team\_ID в таблице Players.

Один стадион (Stadiums) может быть связан с несколькими матчами (Matches), реализовано через внешний ключ Stadium\_ID в таблице Matches.

Один турнир (Tournaments) может быть связан с несколькими матчами, реализовано через внешний ключ Tournament\_ID в таблице Matches.

* + - **Связь "многие ко многим" (M:N):**

Одна команда может участвовать в нескольких матчах, а каждый матч включает две команды (домашнюю и гостевую). Эта связь реализована через поля Home\_Team\_ID и Away\_Team\_ID в таблице Matches.

* + - **Связи "один к одному" (1:1):**

Каждая команда (Teams) может быть связана только с одним тренером (Coaches), и один тренер может быть привязан только к одной команде. Эта связь реализована через внешний ключ Coach\_ID в таблице Teams, который ссылается на Coach\_ID в таблице Coaches.

Каждая команда также может быть связана только с одним стадионом (Stadiums) через внешний ключ Stadium\_ID.

1. **Установление связей через внешние ключи:**

Для каждой связи между таблицами определены внешние ключи. Например:

* Поле Team\_ID в таблице Players связано с полем Team\_ID в таблице Teams.
* Поле Stadium\_ID в таблице Matches связано с полем Stadium\_ID в таблице Stadiums.
* Поле Match\_ID в таблице Goals связано с полем Match\_ID в таблице Matches.
* Поле Scored\_ID в таблице Goals связано с полем Player\_ID в таблице Players.

Переход от концептуальной модели к реляционной базе данных требует преобразования всех сущностей и связей в таблицы с чётко определёнными атрибутами, первичными и внешними ключами. Особое внимание уделено сложным связям типа «многие ко многим» и управлению общими атрибутами.

#### В нашей модели не использовалось **прямое наследование**, однако некоторые сущности содержат общие атрибуты, которые можно рассматривать как своего рода обобщение. Например, сущности Player и Manager оба обладают общими атрибутами, такими как имя и фамилия. Однако, вместо явного наследования, эти атрибуты просто дублируются в обеих таблицах, так как в контексте данной предметной области они несут разные смысловые нагрузки.

#### Некоторые сущности, такие как Transfer, были разбиты на **отдельные таблицы** для упрощения обработки данных. Например, сущность Transfer разбивается на две таблицы: одна для хранения информации о самом трансфере, другая — для хранения истории трансфера игрока.

#### 1. **Players**

* **Таблица и атрибуты**:
  + Player\_ID (PK) - уникальный идентификатор игрока.
  + Player\_Name - имя игрока.
  + Team\_Name - имя команды, за которую играет игрок.
  + Nationality - национальность.
  + DOB - дата рождения.
  + Salary - зарплата игрока.
  + Assists - количество ассистов.
  + Goals - количество голов.
  + Jersey\_Number - номер на футболке.
  + Game\_Position - игровая позиция.
  + Height - рост игрока.
  + Weight - вес игрока.
  + Foot - ведущая нога.
  + **Связь**:
    - FK Team\_ID связан с PK Team\_ID в таблице Teams.

#### 2. **Teams**

* **Таблица и атрибуты**:
  + Team\_ID (PK) - уникальный идентификатор команды.
  + Team\_Name - имя команды.
  + Main\_Owner - владелец команды.
  + Foundation\_Year - год основания команды.
  + Budget - бюджет команды.
  + Trophy\_numbers - количество трофеев.
  + **Связи**:
    - FK Coach\_ID связан с PK Coach\_ID в таблице Coaches.
    - FK Sponsor\_ID связан с PK Sponsor\_ID в таблице Sponsors.
    - FK Stadium\_ID связан с PK Stadium\_ID в таблице Stadiums.

#### 3. **Coaches**

* **Таблица и атрибуты**:
  + Coach\_ID (PK) - уникальный идентификатор тренера.
  + Full\_Name - полное имя тренера.
  + Nationality - национальность.
  + DOB - дата рождения.
  + **Связь**:
    - FK Team\_Name связан с PK Team\_Name в таблице Teams.

#### 4. **Sponsors**

* **Таблица и атрибуты**:
  + Sponsor\_ID (PK) - уникальный идентификатор спонсора.
  + Name - имя спонсора.
  + Industry - отрасль спонсора.
  + Budget - бюджет спонсора.
  + **Связь**:
    - FK Sponsor\_ID связан с Teams и Tournaments.

#### 5. **Stadiums**

* **Таблица и атрибуты**:
  + Stadium\_ID (PK) - уникальный идентификатор стадиона.
  + Stadium\_Name - имя стадиона.
  + Location - местоположение (город + страна).
  + Capacity - вместимость стадиона.
  + Year\_Built - год постройки.
  + **Связь**:
    - FK Team\_ID связан с PK Team\_ID в таблице Teams.

#### 6. **Matches**

* **Таблица и атрибуты**:
  + Match\_ID (PK) - уникальный идентификатор матча.
  + Season - сезон матча.
  + Date - дата матча.
  + Home\_Team\_Score - счет домашней команды.
  + Away\_Team\_Score - счет гостевой команды.
  + Penalty\_Shoot\_Out - пенальти.
  + Attendance - посещаемость.
  + Duration - длительность матча.
  + Yellow\_Cards - количество желтых карточек.
  + Red\_Cards - количество красных карточек.
  + **Связи**:
    - FK Home\_Team\_ID и Away\_Team\_ID связаны с PK Team\_ID в таблице Teams.
    - FK Stadium\_ID связан с PK Stadium\_ID в таблице Stadiums.
    - FK Tournament\_ID связан с PK Tournament\_ID в таблице Tournaments.
    - FK Referee\_ID связан с PK Referee\_ID в таблице Referees.

#### 7. **Goals**

* **Таблица и атрибуты**:
  + Goal\_ID (PK) - уникальный идентификатор гола.
  + Match\_Time - время гола.
  + Scored - идентификатор игрока, забившего гол.
  + Assist - идентификатор игрока, сделавшего ассист.
  + Goal\_Desc - описание гола.
  + **Связи**:
    - FK Match\_ID связан с PK Match\_ID в таблице Matches.
    - FK Scored\_ID и Assist\_ID связаны с PK Player\_ID в таблице Players.

#### 8. **Tournaments**

* **Таблица и атрибуты**:
  + Tournament\_ID (PK) - уникальный идентификатор турнира.
  + Name - имя турнира.
  + Year - год проведения турнира.
  + Winner - победитель турнира.
  + Number\_of\_Teams - количество команд.
  + Location - место проведения.
  + **Связь**:
    - FK Sponsor\_ID связан с PK Sponsor\_ID в таблице Sponsors.

#### 9. **Referees**

* **Таблица и атрибуты**:
  + Referee\_ID (PK) - уникальный идентификатор судьи.
  + Name - имя судьи.
  + Nationality - национальность.
  + Experience\_Years - опыт работы (в годах).
  + Matches\_numbers - количество обслуженных матчей.
  + Yellow\_Cards - количество выданных желтых карточек.
  + Red\_Cards - количество выданных красных карточек.
  + **Связь**:
    - FK Referee\_ID связан с таблицей Matches.

## ***Диаграмма реляционной модели***

#### На рисунке представлена диаграмма реляционной модели базы данных Лиги чемпионов УЕФА. В ней показаны все таблицы, их поля, а также связи между таблицами, выраженные через внешние ключи. Каждая таблица имеет уникальный первичный ключ, обеспечивающий уникальность записей, а также внешние ключи, поддерживающие целостность данных при удалении или изменении записей.

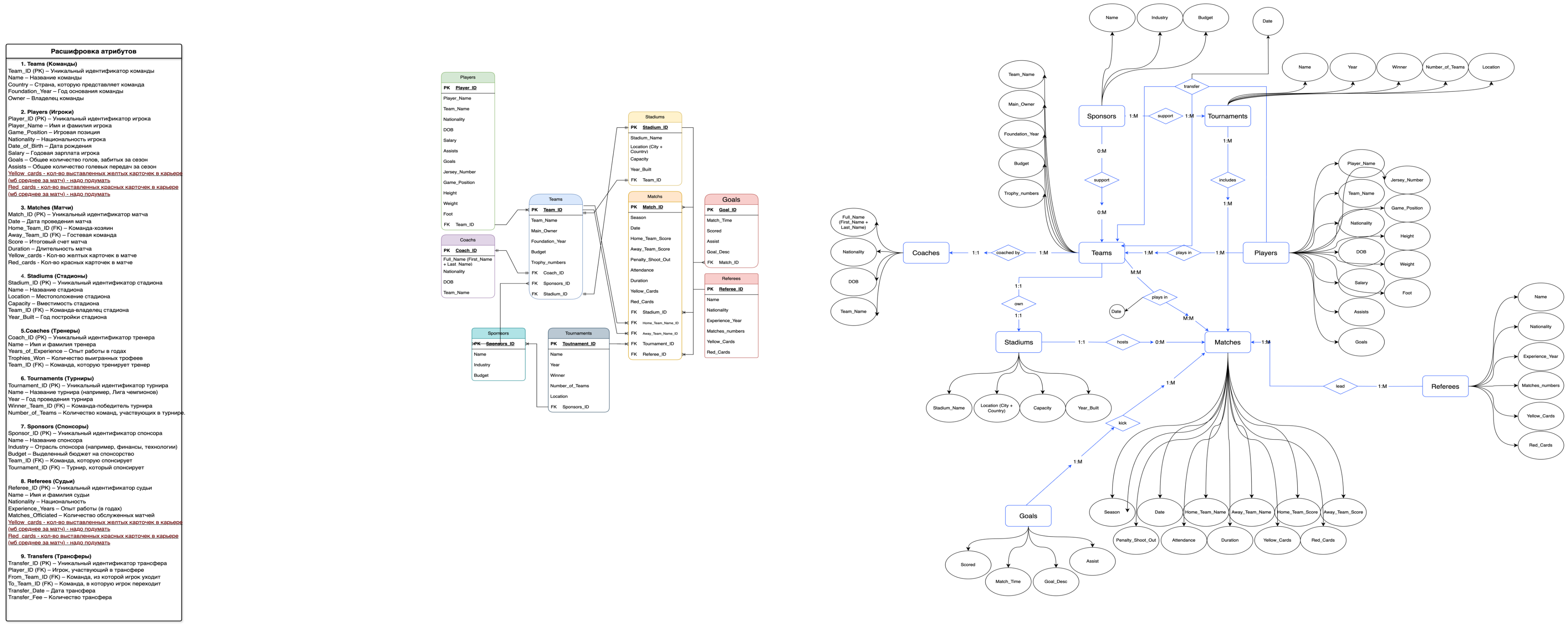


Диаграмма реляционной модели

## ***Механизмы обеспечения целостности данных***

Для поддержания целостности данных были установлены следующие ограничения:

* + - 1. **Общие ограничения**
* PRIMARY KEY: Гарантирует уникальность записей в таблице. Для каждой таблицы первичный ключ используется для уникальной идентификации строки.
* FOREIGN KEY: Обеспечивает ссылочную целостность, связывая записи между таблицами. Например, внешний ключ Sponsor\_ID в таблице Tournaments ссылается на Sponsors.
* CHECK: Устанавливает условные ограничения на значения полей (например, числа должны быть больше 0 или даты — не из будущего).
* UNIQUE: Гарантирует уникальность значений в одном или нескольких столбцах, что предотвращает дублирование.

1. **Таблица Sponsors**

* PRIMARY KEY: Sponsor\_ID — уникальный идентификатор спонсора.
* NOT NULL: Sponsor\_Name не может быть пустым.

1. **Таблица Coaches**

* PRIMARY KEY: Coach\_ID — уникальный идентификатор тренера.
* NOT NULL: Coach\_Full\_Name — обязательное поле.
* CHECK: DOB < DATE('now') — дата рождения должна быть в прошлом.
* FOREIGN KEY: Team\_Name ссылается на таблицу Teams, при удалении команды устанавливается NULL.

1. **Таблица Referees**
   * PRIMARY KEY: Referee\_ID — уникальный идентификатор судьи.
   * NOT NULL: Поля Referee\_Name и Nationality обязательны.
   * CHECK:

Experience\_Years > 0 — опыт должен быть положительным числом.

Matches\_numbers >= 0, Yellow\_Cards >= 0, Red\_Cards >= 0 — числа не могут быть отрицательными.

1. **Таблица Tournaments**
   * + PRIMARY KEY: Tournament\_ID — уникальный идентификатор турнира.
     + NOT NULL: Поле Tournament\_Name обязательно.
     + CHECK:

Year <= CAST(strftime('%Y', 'now') AS INTEGER) — год турнира не может быть в будущем.

Number\_of\_Teams > 0 — число команд должно быть больше 0.

* + - FOREIGN KEY: Sponsor\_ID ссылается на таблицу Sponsors, при удалении спонсора устанавливается NULL.

1. **Таблица Teams**
   * PRIMARY KEY: Team\_ID — уникальный идентификатор команды.
   * UNIQUE: Team\_Name должен быть уникальным.
   * NOT NULL: Поле Team\_Name обязательно.
   * CHECK: Foundation\_year <= CAST(strftime('%Y', 'now') AS INTEGER) — год основания не может быть в будущем.
   * FOREIGN KEY: Coach\_ID ссылается на Coaches; Sponsor\_ID ссылается на Sponsors; Stadium\_ID ссылается на Stadiums. Внешние ключи настроены на установку NULL при удалении связанных записей.
2. **Таблица Stadiums**
   * PRIMARY KEY: Stadium\_ID — уникальный идентификатор стадиона.
   * NOT NULL: Поле Stadium\_Name и Country обязательны.
   * CHECK:

Capacity > 0 — вместимость стадиона должна быть больше 0.

Year\_Built <= CAST(strftime('%Y', 'now') AS INTEGER) — год постройки стадиона не может быть в будущем.

* + FOREIGN KEY: Team\_ID ссылается на таблицу Teams.

1. **Таблица Players**

* PRIMARY KEY: Player\_ID — уникальный идентификатор игрока.
* NOT NULL: Поля Player\_Name и Nationality обязательны.
* CHECK:

DOB < DATE('now') — дата рождения должна быть в прошлом.

Jersey\_Number >= 0 — номер на футболке должен быть неотрицательным.

Height > 0 и Weight > 0 — рост и вес игрока должны быть больше 0.

Foot IN ('L', 'R') — поле "ведущая нога" принимает значения только 'L' или 'R'.

Salary >= 0 — зарплата должна быть неотрицательной.

Goals >= 0 и Assists >= 0 — количество голов и передач должно быть неотрицательным.

Transfer\_Year <= CAST(strftime('%Y', 'now') AS INTEGER) — год трансфера не может быть в будущем.

* UNIQUE: Team\_ID и Jersey\_Number вместе должны быть уникальными (один и тот же номер футболки в одной команде невозможен).
* FOREIGN KEY: Team\_ID ссылается на таблицу Teams.

1. **Таблица Matches**
   * + PRIMARY KEY: Match\_ID — уникальный идентификатор матча.
     + CHECK: Penalty\_Shoot\_Out IN (0, 1) — допускаются только значения 0 (не было серии пенальти) или 1 (была).
     + FOREIGN KEY:

Tournament\_ID ссылается на Tournaments.

Referee\_ID ссылается на Referees.

## ***Триггеры***

#### Введена хранимая процедура для обновления статистики матчей - автоматически обновляется статистика матчей после внесения изменений в таблицы, обеспечивая актуальность данных.

1. Триггеры аудита для таблицы players:
   * INSERT: После добавления нового игрока, триггер trg\_players\_audit\_insert записывает все детали нового игрока с типом операции INSERT в таблицу players\_audit.
   * UPDATE: После изменения данных существующего игрока, триггер trg\_players\_audit\_update фиксирует обновленные данные с типом операции UPDATE.
   * DELETE: После удаления игрока, триггер trg\_players\_audit\_delete сохраняет данные удаленного игрока с типом операции DELETE.

Таблица Аудита (players\_audit) содержит поля для хранения информации о типе операции, временной метке, и всех данных игрока на момент изменения. Не имеет внешних ключей к основной таблице players, обеспечивая независимость и сохранение истории даже при удалении записей. В SQLite нет встроенной функции для отслеживания текущего пользователя, поэтому поле user\_name заполняется фиксированным значением или может быть опущено.

1. Триггер для автоматического обновления числа матчей арбитра:
   * INSERT: После добавления нового матча, триггер trg\_update\_referee\_matches автоматически увеличивает значение поля Matches\_numbers у соответствующего арбитра в таблице referee.

#### В процессе проектирования реляционной модели базы данных Лиги чемпионов УЕФА были учтены все необходимые связи между сущностями, включая сложные связи типа "многие ко многим" и наследование. Для реализации этих связей использованы дополнительные таблицы-связки, что обеспечивает корректное хранение и обработку данных. Все таблицы снабжены уникальными первичными ключами и внешними ключами для поддержания целостности данных. Дополнительно были введены триггеры и хранимые процедуры для автоматического обновления и предотвращения удаления несуществующих записей, что гарантирует логическую согласованность базы данных.

# **Развертывание базы данных**

База данных была развёрнута с использованием СУБД SQLite. Для автоматизации создания структуры базы данных, заполнения данными, а также проверки и анализа данных использовался язык программирования Python с привлечением библиотеки sqlite3. Процесс включал следующие шаги:

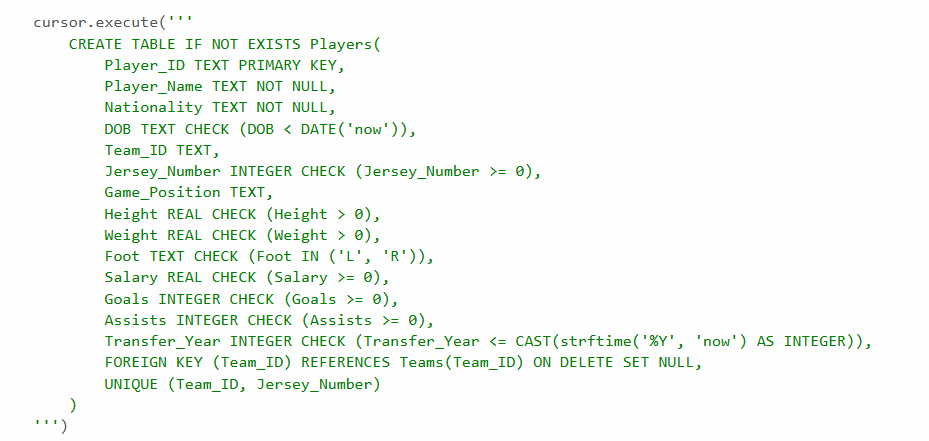
## ***Создание структуры базы данных (DDL-скрипт)***

Создание базы данных и её таблиц осуществлялось в скрипте load\_data. Включённые таблицы и их атрибуты соответствуют схеме базы данных, представленной в ER-диаграмме. Для каждой таблицы применялись следующие подходы:

* Условие IF NOT EXISTS: Это позволяет избежать ошибок при повторном выполнении скрипта, если таблица уже существует.
* Описанные выше ограничения целостности.
* Структура таблиц: Например:

Таблица Players включает атрибуты: Player\_ID, Player\_Name, Team\_ID, DOB, и другие, с ограничением уникальности на сочетание Team\_ID и Jersey\_Number.

Таблица Matches содержит атрибуты: Match\_ID, Season, Date\_Time, Home\_Team\_ID, Away\_Team\_ID, и другие.



## ***Заполнение базы данных (DML-скрипт)***

Для заполнения таблиц данными использовались исходные данные из Excel-файла. Основные этапы:

1. **Очистка и обработка данных**

* Замена пустых значений (NaN) на None с помощью pandas.
* Удаление лишних пробелов в строковых данных.
* Преобразование формата дат в YYYY-MM-DD для полей DOB и Date\_Time.

1. **Последовательность загрузки данных**

Для соблюдения зависимостей внешних ключей данные загружались в следующем порядке:

Sponsors → Coaches → Referees → Tournaments → Teams → Stadiums → Players → Matches → Goals

1. **Валидация внешних ключей**

Перед вставкой данных в таблицы выполнялась проверка, чтобы убедиться, что ссылки на связанные таблицы являются корректными. Если значения не соответствовали ограничениям, они заменялись на NULL.

1. **Проверка работы ограничений целостности**

Для проверки правильной работы ограничений целостности был реализован код, пытающийся вставить некорректные данные. В случае нарушения ограничений SQLite выдавала соответствующую ошибку.

try:

cursor.execute(f"INSERT INTO Players (DOB) VALUES (?)", ('2100-01-01',))

conn.commit()

print("Data inserted successfully.")

except Exception as e:

print(f"Integrity constraint violation: {e}")

1. **Генерация дополнительных данных**

Поскольку исходные данные не содержали всех необходимых атрибутов (например, описания спонсоров или национальности тренеров), данные были сгенерированы с использованием:

* Синтетических данных, подгруженных с помощью python.
* Модуля random для распределения значений по полям.
* Языковой модели для создания текстовых описаний.

Полный код скрипта доступен в [репозитории проекта](https://github.com/darlorean/db_project/tree/main).

## ***Контроль работы ограничений целостности***

Для контроля работы ограничений целостности в базе данных был реализован ряд тестов, которые проверяют правильность выполнения наложенных ограничений. Эти тесты включают попытки вставки данных, нарушающих ограничения, и обработку ожидаемых ошибок.

1. Проверка работы NOT NULL

try:

cursor.execute('''

INSERT INTO Players (Player\_ID, Nationality, DOB, Team\_ID, Jersey\_Number)

VALUES (?, ?, ?, ?, ?)

''', ('P1001', 'Spanish', '1990-06-15', 'T101', 10))

conn.commit()

print("Test failed: Data inserted without Player\_Name")

except sqlite3.IntegrityError as e:

print(f"Test passed: Integrity constraint violation - {e}")

2. Проверка работы CHECK

try:

cursor.execute('''

INSERT INTO Players (Player\_ID, Player\_Name, Nationality, DOB, Team\_ID, Jersey\_Number, Foot)

VALUES (?, ?, ?, ?, ?, ?, ?)

''', ('P1002', 'John Doe', 'English', '1992-04-10', 'T102', 9, 'B'))

conn.commit()

print("Test failed: Data inserted with invalid Foot value")

except sqlite3.IntegrityError as e:

print(f"Test passed: Integrity constraint violation - {e}")

3. Проверка работы внешних ключей

try:

cursor.execute('''

INSERT INTO Players (Player\_ID, Player\_Name, Nationality, DOB, Team\_ID, Jersey\_Number)

VALUES (?, ?, ?, ?, ?, ?)

''', ('P1003', 'Alex Smith', 'French', '1995-07-20', 'T999', 11))

conn.commit()

print("Test failed: Data inserted with invalid Team\_ID")

except sqlite3.IntegrityError as e:

print(f"Test passed: Integrity constraint violation - {e}")

4. Проверка работы UNIQUE

try:

cursor.execute('''

INSERT INTO Players (Player\_ID, Player\_Name, Nationality, DOB, Team\_ID, Jersey\_Number)

VALUES (?, ?, ?, ?, ?, ?)

''', ('P1004', 'Robert Green', 'German', '1994-03-12', 'T101', 10))

conn.commit()

print("Test failed: Duplicate Jersey\_Number allowed for the same Team\_ID")

except sqlite3.IntegrityError as e:

print(f"Test passed: Integrity constraint violation - {e}")

5. Проверка ссылочной целостности при удалении

try:

# Вставка данных с валидными ссылками

cursor.execute('''

INSERT INTO Matches (Match\_ID, Home\_Team\_ID, Away\_Team\_ID)

VALUES (?, ?, ?)

''', ('M1001', 'T101', 'T102'))

cursor.execute('DELETE FROM Teams WHERE Team\_ID = ?', ('T101',))

conn.commit()

print("Test failed: Referential integrity not enforced")

except sqlite3.IntegrityError as e:

print(f"Test passed: Referential integrity constraint violation - {e}")

# **Разработка клиентского приложения**

## ***Тестовое приложение и реализованные сценарии использования***

Тестовое клиентское приложение было разработано с целью предоставления пользователю удобного интерфейса для работы с данными Лиги чемпионов УЕФА. Оно ориентировано как на любителей футбола, так и на профессионалов, таких как аналитики, журналисты и администраторы команд. Основная цель приложения — предоставить простой и интуитивно понятный доступ к актуальной информации о турнире, а именно о матчах, командах, игроках, тренерах и статистике.

**Основные функциональные возможности приложения включают:**

**Просмотр матчей:** Пользователи могут просматривать расписания и результаты матчей Лиги чемпионов УЕФА, а также анализировать статистику команд и игроков, что позволяет следить за ходом турнира в реальном времени и глубже изучать статистику.

**Добавление и редактирование данных:** Зарегистрированные пользователи могут добавлять новые данные о матчах, результатах и игроках, а также редактировать существующую информацию. Это открывает возможность для взаимодействия с данными и поддержания актуальности информации.

**Интерактивные запросы и отчеты:** Приложение поддерживает выполнение различных SQL-запросов для получения отчетов и специфической статистики, например, топовых бомбардиров, успешных тренеров или анализа матчей по сезонам.

**Социальные сети и контент:** Приложение интегрировано с социальными сетями, позволяя пользователям делиться результатами, новостями и аналитикой. Также пользователи могут создавать и публиковать собственные статьи, комментарии и обзоры матчей.

**Инфографика и визуализация данных:** В приложении реализован функционал для создания и отображения графиков и диаграмм, что позволяет наглядно представлять изменения в динамике голов, побед, передач и других показателей.

Реализованные сценарии использования включают в себя не только стандартные операции (просмотр данных, фильтрация результатов), но и более сложные запросы для анализа и сравнения различных параметров турнира, таких как эффективность тренеров, динамика команды по сезонам и т.д.

## ***Примеры ввода и редактирования данных в БД***

Приложение позволяет пользователям добавлять и редактировать данные, что является важным элементом взаимодействия с информацией. Рассмотрим несколько примеров:

* Регистрация и авторизация пользователей**:** чтобы начать использовать приложение, пользователи должны зарегистрироваться, создав аккаунт. Регистрация позволяет сохранить персональные настройки, такие как выбор любимых команд, предпочтений по уведомлениям и других параметров.
* Обновление профиля**:** зарегистрированные пользователи могут обновлять свои профили, например, изменять информацию о своих любимых командах, тренерах, настраивать уведомления для получения информации о матчах.
* Ввод данных**:** приложение позволяет пользователям добавлять информацию о матчах, таких как даты, результаты, составы команд и статистику игроков. Например, после завершения матча пользователь может вручную ввести результат, статистику голов и передач игроков.
* Редактирование данных**:** пользователи могут редактировать уже добавленные данные, например, если была допущена ошибка в статистике или произошли изменения в составе команды. Это позволяет поддерживать актуальность базы данных.
* Создание контента**:** пользователи могут создавать и редактировать статьи, комментарии и обзоры матчей. Это дает возможность публиковать аналитические материалы, обмениваться мнениями с другими пользователями, а также создавать форумы для обсуждения турнира.

## ***Запросы и отчёты***

Приложение поддерживает выполнение SQL-запросов, которые позволяют пользователям получать необходимые данные о турнире, команды и игроков. Примеры таких запросов:

**Общее количество матчей по странам**

SELECT s.country AS country, COUNT(m.match\_id) AS total\_matches

FROM matches m

JOIN stadiums s

ON m.stadium\_id = s.stadium\_id

GROUP BY s.country

ORDER BY total\_matches DESC;



**Самые успешные команды на домашнем стадионе**

SELECT t.team\_name AS team\_name, COUNT(m.match\_id) AS home\_wins

FROM matches m

JOIN teams t

ON m.home\_team\_id = t.team\_id

WHERE m.home\_team\_score > m.away\_team\_score

GROUP BY t.team\_name

HAVING COUNT(m.match\_id) > 3

ORDER BY home\_wins DESC;



**Тренеры, работающие с иностранными командами**

SELECT t.team\_name AS team\_name, c.coach\_full\_name AS coach\_name, s.country AS team\_country, c.nationality AS coach\_country

FROM teams t

JOIN coaches c

ON t.coach\_id = c.coach\_id

JOIN stadiums s

ON t.stadium\_id = s.stadium\_id

WHERE s.country != c.nationality

ORDER BY t.team\_name;



**Матчи на стадионах с вместимостью более 60,000**

SELECT M.MATCH\_ID, M.DATE\_TIME, M.HOME\_TEAM\_ID, M.AWAY\_TEAM\_ID, S.STADIUM\_NAME, S.CAPACITY

FROM MATCHES M

JOIN STADIUMS S

ON M.STADIUM\_ID = S.STADIUM\_ID

WHERE S.CAPACITY > 60000

ORDER BY M.DATE\_TIME;



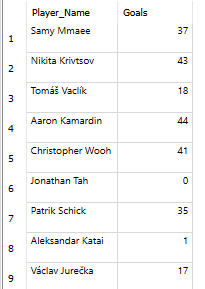
**Игроки с ростом выше 180 см и трансфером в 2020 году**

SELECT player\_name, goals

FROM players

WHERE height > 180

AND transfer\_year = 2020;



**Процент побед домашних команд из Англии**

SELECT t.team\_name AS team\_name, COUNT(m.match\_id) AS total\_home\_matches,

SUM(CASE WHEN m.home\_team\_score > m.away\_team\_score THEN 1 ELSE 0 END) AS home\_wins,

ROUND((SUM(CASE WHEN m.home\_team\_score > m.away\_team\_score THEN 1 ELSE 0 END) \* 100.0) / COUNT(m.match\_id), 2) AS win\_percentage

FROM teams t

JOIN matches m

ON t.team\_id = m.home\_team\_id

JOIN stadiums s

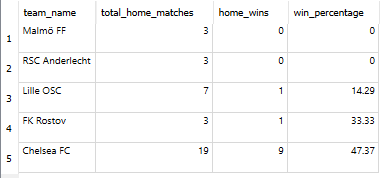
ON t.stadium\_id = s.stadium\_id

WHERE s.country = 'England'

GROUP BY t.team\_name

HAVING win\_percentage < 50

ORDER BY win\_percentage ASC;



**Сезон с наибольшим количеством голов левой ногой**

SELECT m.season AS season, SUM(p.goals) AS left\_foot\_goals

FROM players p

JOIN matches m

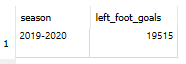
ON p.team\_id = m.home\_team\_id OR p.team\_id = m.away\_team\_id

WHERE p.foot = 'L'

GROUP BY m.season

ORDER BY left\_foot\_goals DESC

LIMIT 1;



**Тренер с наибольшим количеством побед за сезон**

SELECT c.coach\_full\_name AS manager\_name, COUNT(m.match\_id) AS wins

FROM coaches c

JOIN teams t

ON c.coach\_id = t.coach\_id

JOIN matches m

ON t.team\_id = m.home\_team\_id

WHERE m.home\_team\_score > m.away\_team\_score

AND m.season BETWEEN 2016 AND 2022

GROUP BY c.coach\_full\_name

ORDER BY wins DESC

LIMIT 1;



**Победители турниров по годам**

SELECT t.year AS season, t.winner AS winner\_team

FROM tournaments t

WHERE t.year IN ('2016-2017', '2017-2018', '2018-2019', '2019-2020', '2020-2021', '2021-2022')

ORDER BY t.year ASC;



## ***Примеры комплексной инфографики***

Для улучшения восприятия данных и углубленного анализа, приложение включает различные визуализации данных, которые помогают пользователям лучше понять результаты турнира.

На примере данного запроса были построены комплексные инфографики:

SELECT

Matches.Match\_ID AS MatchID,

Matches.Season AS Season,

Matches.Date\_Time AS MatchDate,

HomeTeam.Team\_Name AS HomeTeam,

AwayTeam.Team\_Name AS AwayTeam,

Matches.Home\_Team\_Score AS HomeScore,

Matches.Away\_Team\_Score AS AwayScore,

Tournaments.Tournament\_Name AS Tournament,

Tournaments.Year AS TournamentYear,

Stadiums.Stadium\_Name AS Stadium,

Stadiums.City AS StadiumCity,

Stadiums.Country AS StadiumCountry,

Referees.Referee\_Name AS Referee,

Players.Player\_Name AS Scorer,

Players.Nationality AS ScorerNationality,

Goals.Match\_Time AS GoalTime,

Goals.Goal\_Desc AS GoalDescription

FROM Matches

LEFT JOIN Teams AS HomeTeam ON Matches.Home\_Team\_ID = HomeTeam.Team\_ID

LEFT JOIN Teams AS AwayTeam ON Matches.Away\_Team\_ID = AwayTeam.Team\_ID

LEFT JOIN Tournaments ON Matches.Tournament\_ID = Tournaments.Tournament\_ID

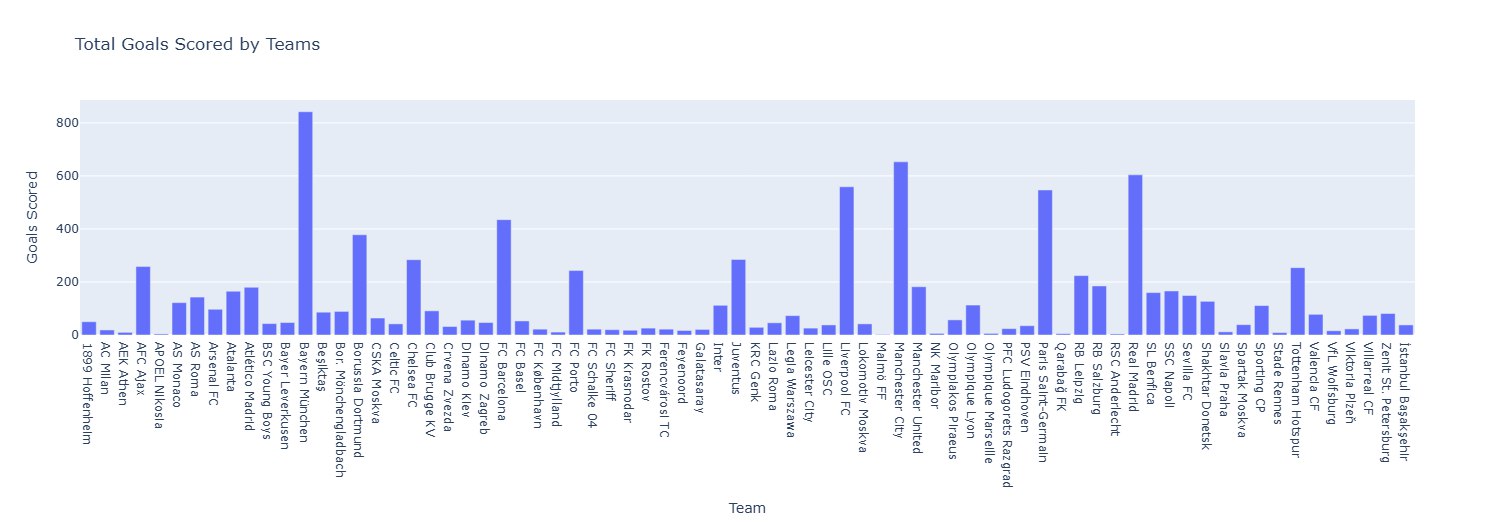
LEFT JOIN Stadiums ON Matches.Stadium\_ID = Stadiums.Stadium\_ID

LEFT JOIN Referees ON Matches.Referee\_ID = Referees.Referee\_ID

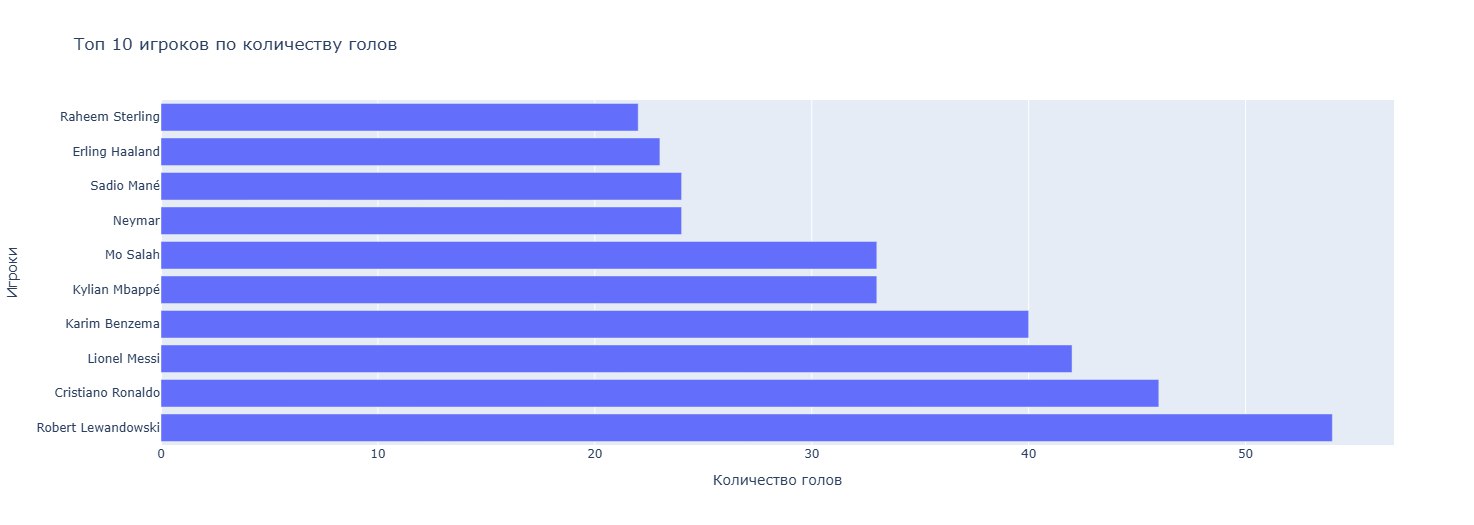
LEFT JOIN Goals ON Matches.Match\_ID = Goals.Match\_ID

LEFT JOIN Players ON Goals.Scored\_ID = Players.Player\_ID

ORDER BY Matches.Date\_Time DESC;



Гистограмма показывает общее количество забитых голов командами в Лиге чемпионов УЕФА. На графике видно, что команды, такие как Bayern Munich, Real Madrid и Manchester City, значительно лидируют по числу забитых голов, в то время как у других клубов показатели значительно ниже.



Горизонтальная столбчатая диаграмма показывает топ-10 игроков по количеству забитых голов. Лидерами являются Robert Lewandowski и Cristiano Ronaldo, значительно опережающие других игроков. За ними следуют Lionel Messi, Karim Benzema и Kylian Mbappé.

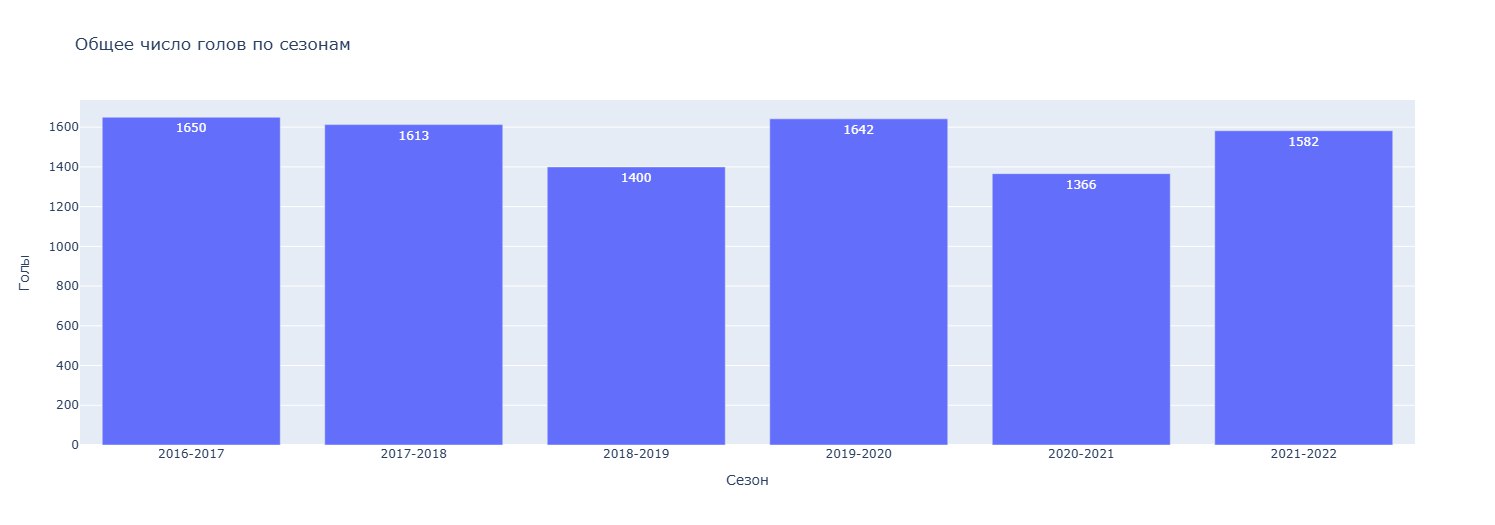


График демонстрирует общее количество забитых голов по сезонам. Видно, что самые результативные сезоны — 2016-2017 (1650 голов) и 2019-2020 (1642 гола), а наименьшее количество голов было в сезоне 2018-2019 (1400 голов).

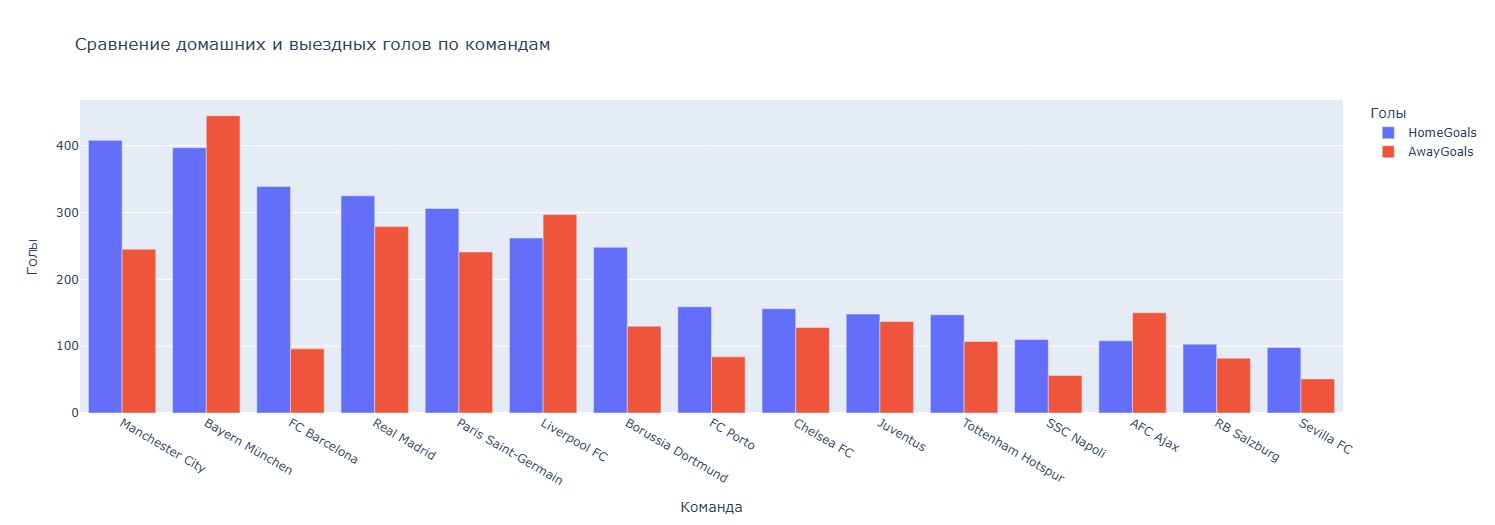


График сравнивает количество забитых голов дома (синие столбцы) и на выезде (красные столбцы) для различных команд. Видно, что Bayern München доминирует как по домашним, так и по выездным голам, а другие команды, такие как Liverpool FC, показывают равномерное распределение голов между домашними и выездными матчами. Некоторые команды, например AFC Ajax, показывают более высокую результативность в выездных матчах.

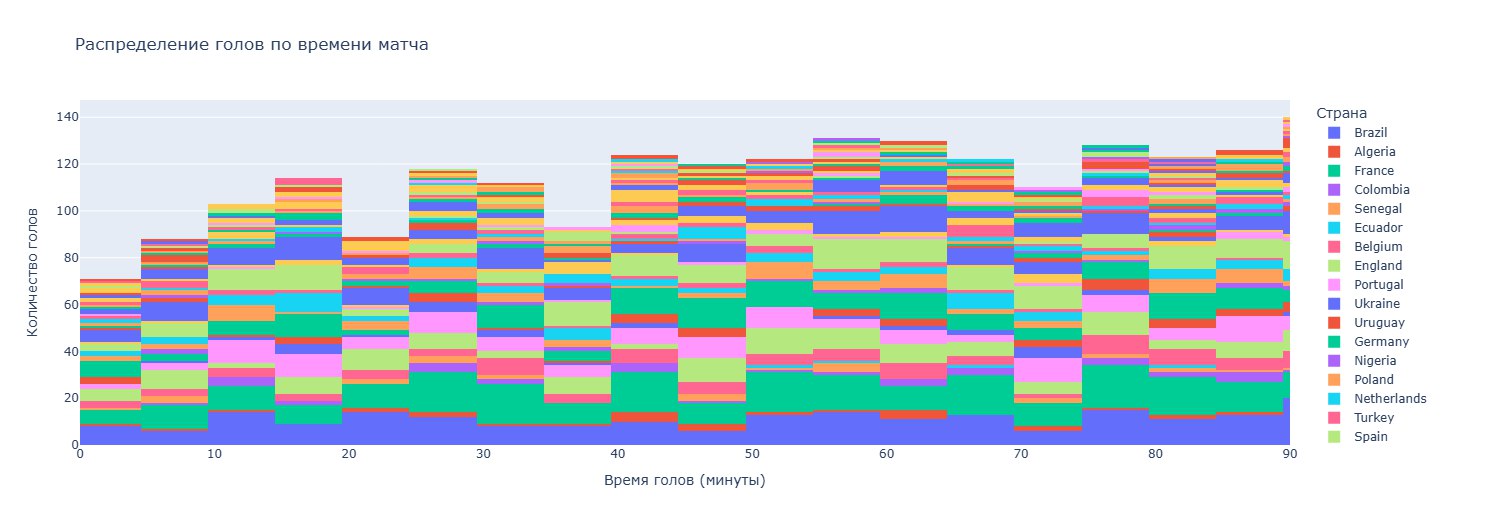


График показывает распределение голов по времени матча, разбитое по странам игроков, которые забили эти голы (диаграмма в оригинале интерактивная). Можно заметить, что количество голов увеличивается ближе к концу каждого тайма, особенно в последние 10 минут (например, между 40–45 и 80–90 минутами), что может быть связано с усилением атакующих действий команд в этих промежутках времени. Также видно, что такие страны, как **Бразилия, Франция** и **Испания**, часто вносят значительный вклад в общее количество забитых голов.

Разработанное тестовое клиентское приложение для работы с данными Лиги чемпионов УЕФА представляет собой мощный инструмент для широкого круга пользователей. Оно сочетает в себе простоту и функциональность, позволяя эффективно работать с футбольной статистикой, получать актуальные данные, создавать персонализированный контент и делиться информацией с другими. Включение интерактивных запросов и отчетов, а также визуализаций данных через инфографику значительно повышает ценность приложения для аналитиков и любителей футбола. Такое приложение может стать незаменимым инструментом для более глубокого погружения в мир футбольных турниров и анализа данных о командах, игроках и тренерах.

# **Заключение**

В рамках данного проекта была разработана реляционная база данных, основанная на данных Лиги чемпионов УЕФА за сезоны 2016–2022 годов. Проект включал полный цикл работы с базой данных, начиная от проектирования модели и заканчивая визуализацией данных. Основные результаты работы:

* + - 1. Проектирование базы данных:
* Разработана концептуальная ER-диаграмма и реляционная модель с учетом всех необходимых сущностей и связей.
* Определены и реализованы ограничения целостности данных, включая первичные ключи, внешние ключи, проверки и правила обработки ссылочной целостности.
  + - 1. Реализация:
* С использованием библиотеки `sqlite3` разработаны DDL- и DML-скрипты для создания таблиц и наполнения их данными.
* Создана структура таблиц, обеспечивающая эффективное хранение данных и поддержание целостности базы.
  + - 1. Анализ данных:
* Реализован ряд SQL-запросов для анализа ключевых показателей: статистика по голам, победам, тренерам и турнирам.
* Сформированы визуализации, отражающие динамику забитых голов, распределение результатов по сезонам, вклад игроков и команд.
  + - 1. Контроль ограничений:
* Проверена корректность работы ограничений целостности через попытки вставки некорректных данных.
* Гарантировано соответствие данных требованиям схемы базы.
  + - 1. Визуализация и интерпретация:
* С использованием `plotly` созданы информативные графики, визуализирующие распределение голов, результаты матчей и вклад различных команд и игроков в успех своих клубов.

В целом данный проект демонстрирует эффективность работы с реляционными базами данных для решения задач анализа и визуализации больших объемов данных. Итоговая структура базы данных может быть использована для дальнейшего анализа, добавления новых данных и расширения функционала. Проект может быть полезен для образовательных целей, а также в качестве основы для более сложных аналитических систем.

#### 