Федеральное государственное автономное образовательное учреждение
высшего образования
"Национальный исследовательский университет
"Высшая школа экономики"

Московский институт электроники и математики им. А.Н Тихонова

Департамент компьютерной инженерии

ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ ВОЛЕЙБОЛЬНОГО КЛУБА

Преподаватель:
Карпова Ирина Петровна

Выполнила:
Кноль Мария Алексеевна, БИВ221

Москва 2024

Оглавление

1. Инфологическое проектирование

1.1 Анализ предметной области

База данных создаётся для повышения эффективность работы за счет систематизации и быстрого поиска необходимой информации. БД должна содержать данные о тренерах, игроках, залах, командах и матчах.

Цель создания БД – осуществить удобное ведение учета и быстрый поиск нужной информации.

В соответствии с предметной областью система строится с учётом следующих особенностей:

1. Каждую команду могут тренировать несколько тренеров.
2. Каждый тренер может тренировать только одну команду.
3. Тренировка проходит под руководством всех тренеров, закрепленных за одной командой.
4. Каждая команда однозначно определяется своим названием.
5. Каждый игрок - член какой-то команды.
6. Каждый матч проходит в одном зале.
7. В каждом матче может участвовать только две команды.
8. В одном зале могут тренироваться несколько команд.
9. Одна команда в один промежуток времени может тренироваться только в одном зале.
10. Каждый тренер может иметь три состояния помимо рабочего: уволен, в отпуске, на больничном.
11. Оклад каждого тренера зависит от его специализации.

Для создания ER-модели необходимо выделить сущности предметной области и указать их атрибуты. **Идентифицирующие** (уникальные) атрибуты мы выделим полужирным шрифтом, *многозначные* – курсивом, составные подчеркнем:

1) **Игроки**. Атрибуты: ФИО, **паспортные данные**, пол, дата рождения, *номер телефона*, рост, вес, *позиция на поле*.
2) **Тренеры**. Атрибуты: ФИО, **паспортные данные**, **ИНН**, дата рождения, *номер телефона*, пол, дата начала работы, специализация, оклад, статус (работает, в отпуске, болеет, уволен).
3) **Команды**. Атрибуты: **название команды,** количество игроков.
4) **Матчи**. Атрибуты: дата проведения, время проведения, счет.
5) **Залы**. Атрибуты: **номер зала**, адрес, вместимость, *тип покрытия*.

Исходя из выявленных сущностей, построим ER–диаграмму (Рис. 1).

Пометки у линий отражают кардинальность связи: 1:1, 1:N и N:M.

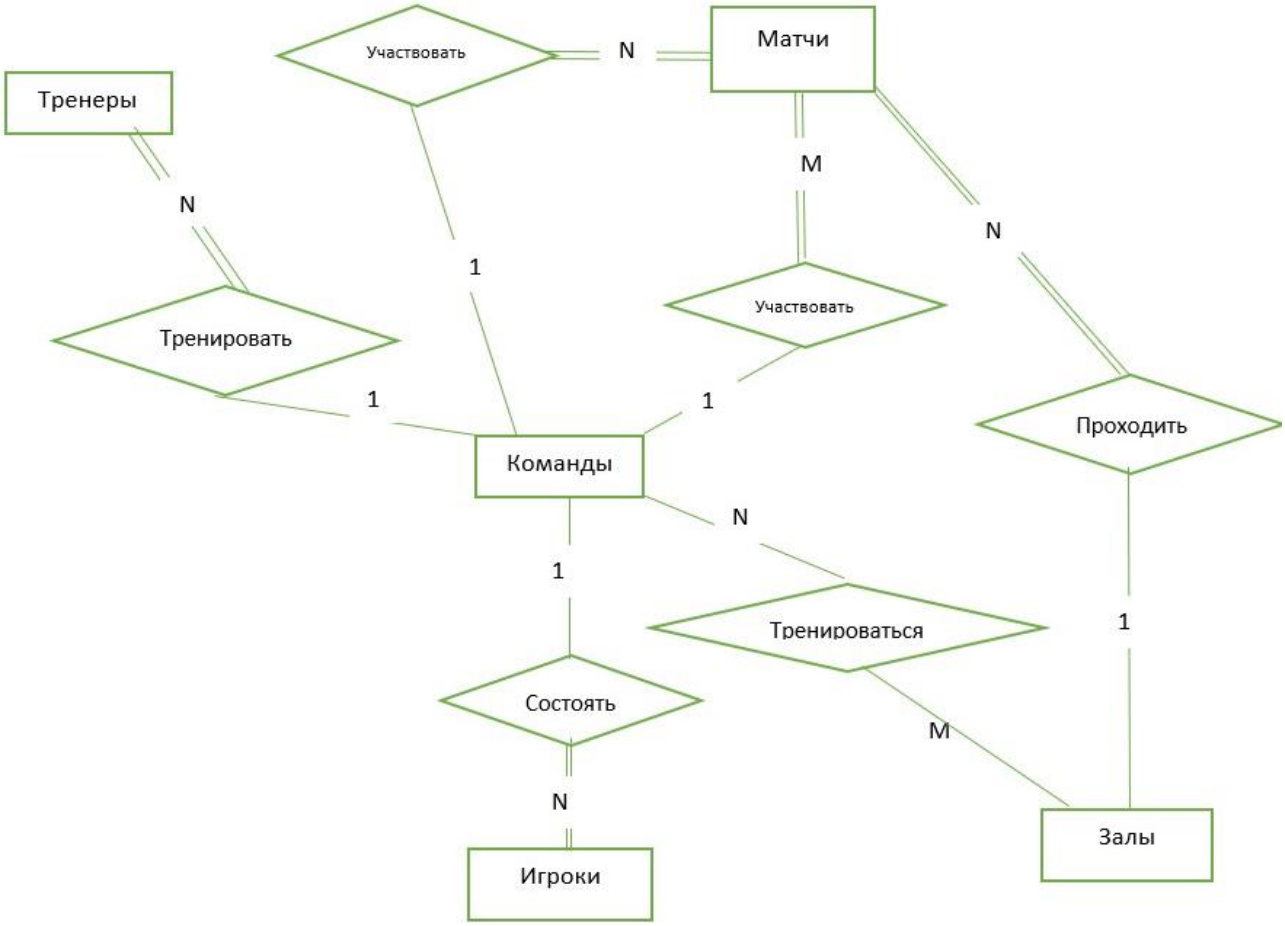

Рис. 1. ER–диаграмма «Волейбольный клуб»

1.2 1.2 Анализ информационных задач и круга пользователей системы

Определим группы пользователей, их основные задачи и запросы к БД:

1) Администратор клуба:
   - Добавление, редактирование и удаление информации о тренерах, игроках, командах, матчах и залах.
   - Управление состоянием тренеров (работает, в отпуске, болеет, уволен).
   - Отслеживание статистики матчей, включая даты, времена проведения и счет.

2) Тренеры:

- Просмотр информации о своем графике тренировок, зарплате, составе команды.
- Внесение изменений в график тренировок.

3) Игроки:
   - Просмотр информации о тренировках и матчах.
   - Внесение изменений в личные данные (например, номер телефона).
4) Зрители и болельщики:
   - Просмотр информации о расписании матчей и их результатов.

2. Определение требований к операционной обстановке.

Объём внешней памяти, необходимый для функционирования системы, складывается из двух составляющих: память, занимаемая модулями СУБД (ядро, утилиты, вспомогательные программы), и память, отводимая под данные ($М_Д$). Для реальных баз данных обычно наиболее существенным является $М_д$.

На основе результатов анализа ПрО можно приблизительно оценить объём памяти, требуемой для хранения данных. Примем ориентировочно:

- каждый день проходит примерно 50 тренировок приблизительным весом по 0,5К
- около 150 участников по 0,5К
- каждый день в среднем проходит по 5 матчей (по 0,1К)
- в каждом мачте участвуют примерно 20 игроков (по 0,2К)

Тогда объём памяти для хранения данных за день примерно составит:

$$М_д = 2(50*50+150 * 50+5*10 + 20*20)) \approx 20,9 \text{ Мб},$$

Значит за год потребуется примерно $20,9 * 365 \approx 7628$ Мб. Коэффициент 2 необходим для того, чтобы учесть необходимость выделения памяти под дополнительные структуры (например, индексы). Объём памяти будет увеличиваться ежегодно на столько же при сохранении объёма работы.

Требуемый объём оперативной памяти определяется на основании анализа интенсивности запросов и объёма результирующих данных. Для нашей БД требуемый объём памяти мал, поэтому никаких специальных требований к объёму внешней и оперативной памяти компьютера не предъявляется.

3. Выбор СУБД и других программных средств

При анализе информационных задач становится очевидным, что практически все системы управления базами данных (СУБД) для персональных электронно-вычислительных машин (ПЭВМ), такие как PostgreSQL, MS Access, Firebird, MySQL и другие, подходят для реализации необходимых функций. Все эти СУБД основаны на реляционной модели данных и предоставляют разнообразные инструменты для работы с данными.

Однако, учитывая созданный с применением объектно-реляционной модели PostgreSQL, следует отметить его способность к обработке сложных структур и широкий набор встроенных и определяемых пользователем типов данных. Исходя из этого, принято решение остановить выбор на PostgreSQL.

4. Логическое проектирование реляционной БД

4.1. Преобразование ER-диаграммы в схему базы данных

База данных создаётся на основании схемы базы данных. Преобразование ER–диаграммы (Рис. 1) в схему БД выполняется путем сопоставления каждой сущности и каждой связи, имеющей атрибуты, отношения (таблицы) БД. Связь типа 1:n (один-ко-многим) между отношениями реализуется через внешний ключ. Ключ вводится для дочернего отношения. Внешнему ключу должен соответствовать первичный или уникальный ключ основного (родительского) отношения. Связь тренироваться между залами и командами относится к типу N:M и реализуется через вспомогательное отношение Тренировки, которое содержит комбинации первичных ключей соответствующих исходных отношений.

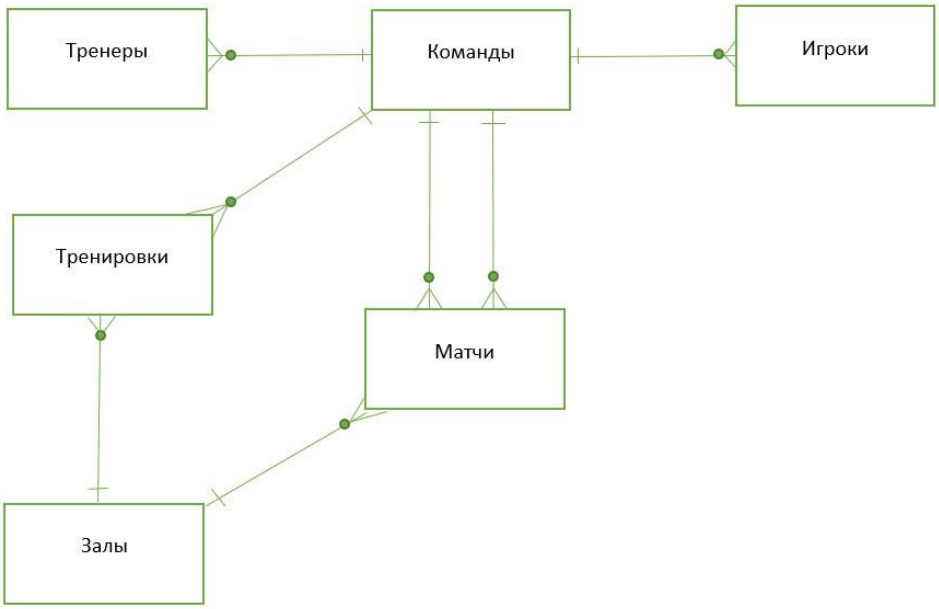

Рис. 2. Схема РБД, полученная из ER–диаграммы

4.2 Составление реляционных отношений

Отношения приведены в Таблицах 1-6. Для каждого отношения указаны атрибуты с их внутренним названием, типом и длиной. Типы данных обозначаются так: N – числовой, C – символьный тип фиксированной длины, V – символьный тип переменной длины, D – дата (этот тип имеет стандартную длину, зависящую от СУБД, поэтому она не указывается), T – timestamp (частный случай даты, дата и время), имеет стандартную длину, зависящую от СУБД, поэтому она не указывается.

Потенциальными ключами отношения ИГРОКИ являются поля паспортные данные и ИНН. Паспортные данные и ИНН занимают достаточно много места, к тому же паспортные данные могут меняться. Поэтому мы введем id игрока в качестве суррогатного первичного ключа.

Таблица 1. Схема отношения ИГРОКИ (Players)

Содержание поля	*Имя поля*	*Тип, длина*	*Примечания*
Id игрока	player_id	N(6)	**суррогатный первичный ключ**
Фамилия	p_surname	V(30)	обязательное поле
Имя	p_name	V(30)	обязательное поле
Отчество	p_patronymic	V(50)	необязательное поле
Дата рождения	p_birth	D	обязательное поле
Пол	p_gender	C(1)	Обязательное поле, ‘м’ или ’ж’
Паспортные данные	p_passport	V(50)	обязательное поле
Телефон	p_phone	V(60)	многозначное поле
ИНН	p_inn	C(12)	обязательное, уникальное поле
Рост	height	N(3)	обязательное поле
Вес	weight	N(3)	обязательное поле
Позиция на поле	position	V(30)	обязательное поле
Номер команды	team_num	N(3)	Обязательное поле, внешний ключ (к teams)

Потенциальными ключами отношения ТРЕНЕРЫ являются поля паспортные данные и ИНН. Паспортные данные и ИНН занимают достаточно много места, к тому же паспортные данные могут меняться. Поэтому мы вводим id тренера в качестве суррогатного первичного ключа.

Таблица 2. Схема отношения ТРЕНЕРЫ (Coaches)

Содержание поля	*Имя поля*	*Тип, длина*	*Примечания*
Id тренера	coach_id	N(6)	**суррогатный первичный ключ**
Фамилия	c_surname	V(30)	обязательное поле
Имя	c_name	V(30)	обязательное поле
Отчество	c_patronymic	V(50)	
Дата рождения	c_birth	D	обязательное поле
Пол	c_gender	C(1)	обязательное поле, ‘м’ или ’ж’
Паспортные данные	c_passport	V(50)	обязательное поле
ИНН	c_inn	C(12)	обязательное, уникальное поле
Телефон	c_phone	V(60)	многозначное поле
Дата начала работы	start_date	D	обязательное поле
Специализация	post	V(30)	обязательное поле
Оклад	salary	N(8,2)	обязательное поле, >14000
Статус	status	V(15)	обязательное поле, проверка на соответствие: или «работает», или «в отпуске», или «на больничном», или «уволен»
Номер команды	tc_num	N(3)	Обязательное поле, внешний ключ (к teams)

Потенциальным ключом отношения КОМАНДЫ является атрибут название команды. Он может занимать много места и также может измениться, поэтому мы введем суррогатный первичный ключ – номер команды.

Таблица 3. Схема отношения КОМАНДЫ (Teams)

Содержание поля	*Имя поля*	*Тип, длина*	*Примечания*
Номер команды	t_num	N(3)	**суррогатный первичный ключ**
Название команды	t_name	V(30)	обязательное поле
Количество игроков	p_count	N(2)	обязательное поле

Введем суррогатный первичный ключ отношения МАТЧИ - номер матча. Поскольку другие атрибуты не являются уникальными и не могут выступать в роли первичных ключей.

Таблица 4. Схема отношения МАТЧИ (Matches)

Содержание поля	*Имя поля*	*Тип, длина*	*Примечания*
Номер матча	n_match	N(6)	**суррогатный первичный ключ**
Дата и время проведения	m_date	T(timestap)	обязательное поле
Счет	score	V(20)	обязательное поле
Команда 1	to_num	N(3)	Обязательное поле, внешний ключ (к teams)
Команда 2	tt_num	N(3)	Обязательное поле, внешний ключ (к teams)
Зал	t_gym	N(3)	Обязательное поле, внешний ключ (к gyms)

Потенциальным ключом отношения ЗАЛЫ является номер зала, так как он является уникальным и не занимает много места.

Таблица 5. Схема отношения ЗАЛЫ (Gyms)

Содержание поля	*Имя поля*	*Тип, длина*	*Примечания*
Номер зала	n_gym	N(3)	**первичный ключ**
Адрес	address	V(60)	обязательное поле
Вместимость	capacity	N(4)	обязательное поле
Тип покрытия	g_type	V(20)	обязательное поле

Таблица 6. Схема отношения ТРЕНИРОВКИ (Training)

Содержание поля	*Имя поля*	*Тип, длина*	*Примечания*
Дата и время проведения	tr_date	T(timestap)	обязательное поле
Команда	tr_team	N(3)	обязательное поле, внешний ключ (к teams)
Тренировочный зал	t_train	N(3)	обязательное поле, внешний ключ (к gyms)

4.3 Нормализация полученных отношений (до 4 НФ)

Игроки. Разделим составной атрибут *Паспортные данные* на *Серия и номер паспорта* (уникальный), *Дата выдачи* и *Кем выдан*. Серию и номер паспорта не имеет смысла хранить в разных полях, т.к. оба они числовые, и система тратит меньше времени на поддержание уникальности одного поля, а не комбинации двух полей.

Также мы создадим справочную таблицу **ВИДЫ ПОЗИЦИЙ**, эта таблица обеспечит одинаковое написание значений соответствующего поля таблицы ИГРОКИ, и возможность добавлять и изменять их при необходимости. Отношения, получившиеся при вынесении многозначных полей из отношения ИГРОКИ, приведены в Таблицах 7-8.

Таблица 7. Схема отношения ИГРОКИ (Players)

Содержание поля	*Имя поля*	*Тип, длина*	*Примечания*
Id игрока	player_id	N(6)	**суррогатный первичный ключ**
Фамилия	p_surname	V(30)	обязательное поле
Имя	p_name	V(30)	обязательное поле
Отчество	p_patronymic	V(50)	необязательное поле
Дата рождения	p_birth	D	обязательное поле
Пол	p_gender	C(1)	Обязательное поле, ‘м’ или ’ж’
Серия и номер паспорта	p_passport	C(10)*	обязательное поле, уникальное
Когда выдан паспорт	p_date	D	обязательное поле
Кем выдан паспорт	p_given	V(50)	обязательное поле
Телефон	p_phone	V(60)	многозначное поле
ИНН	p_inn	C(12)	обязательное, уникальное поле
Рост	height	N(3)	обязательное поле
Вес	weight	N(3)	обязательное поле

Позиция на поле	position	V(30)	обязательное поле, внешний ключ (к Types)
Номер команды	team_num	N(3)	Обязательное поле, внешний ключ (к teams)

* – серию и номер паспорта храним в символьном поле, т.к. если это будет числовое поле, то мы потеряем ведущие нули, если они будут.

Таблица 8. Схема отношения Виды позиций (Types)

Содержание поля	*Имя поля*	*Тип, длина*	*Примечания*
Название вида позиции	t_name	V(30)	**первичный ключ,** (Изначально: связующий, доигровщик, центральный блокирующий, диагональный, либеро)

Тренеры. Так же, как и в отношении Игроки, разделим составной атрибут *Паспортные данные* на *Серия и номер паспорта* (уникальный), *Дата выдачи* и *Кем выдан*.

Также мы создадим справочную таблицу **ВИДЫ СТАТУСОВ**, эта таблица обеспечит одинаковое написание значений соответствующего поля таблицы ТРЕНЕРЫ, и возможность добавлять и изменять их при необходимости.

Для приведения отношения **Тренеры** к 3НФ найдем транзитивные зависимости. Атрибут *Оклад* зависит от атрибута *Специализация*, при этом *Оклад* еще зависит от атрибута *Номер команды*, поэтому создадим отношение **Оклады**, первичным ключом которой будут атрибуты *Номер команды* и *Название Специализации*, и справочную таблицу **Специализации**, чтобы соблюдалась правильность написания названия специализации в отношении **Оклады** (Таблицы 9, 10).

Таблица 9. Схема отношения Специализации (Posts)

Содержание поля	*Имя поля*	*Тип, длина*	*Примечания*
Название специализации	post	V(30)	**первичный ключ,** (Изначально: «главный тренер», «второй тренер», «тренер по физической подготовке», «тренер по анализу игры»)

Таблица 10. Схема отношения Оклады (Salaries)

Содержание поля	*Имя поля*	*Тип, длина*	*Примечания*	
Номер команды	team_num	N(6)	обязательное поле, внешний ключ (к Teams)	**составной первичный ключ**
Название специализации	post	V(30)	обязательное поле, внешний ключ (к Posts)	
Оклад	salary	N(8,2)	обязательное поле, >14000	

Таблица 11. Схема отношения Виды статусов (StatusTypes)

Содержание поля	*Имя поля*	*Тип, длина*	*Примечания*
Статус	status_name	V(15)	**первичный ключ,** (Изначально: «работает», «в отпуске», «на больничном», «уволен»)

Таблица 12. Схема отношения ТРЕНЕРЫ (Coaches)

Содержание поля	*Имя поля*	*Тип, длина*	*Примечания*
Id тренера	coach_id	N(6)	**первичный ключ**
Фамилия	c_surname	V(30)	обязательное поле
Имя	c_name	V(30)	обязательное поле
Отчество	c_patronymic	V(50)	
Дата рождения	c_birth	D	обязательное поле
Пол	c_gender	C(1)	обязательное поле, ‘м’ или ’ж’
Серия и номер паспорта	c_passport	C(10)	обязательное поле, уникальное
Когда выдан паспорт	c_date	D	обязательное поле
Кем выдан паспорт	c_given	V(50)	обязательное поле
ИНН	c_inn	C(12)	обязательное, уникальное поле
Телефон	c_phone	V(60)	многозначное поле
Стаж работы	expwork	N(5,2)	обязательное поле
Номер команды	tc_num	N(3)	Обязательные поля, составной внешний ключ (к Salaries)
Специализация	post	V(30)	
Статус	status	V(15)	обязательное поле, внешний ключ (к StatusTypes)

Матчи. Разделим составной атрибут *Счет* на *Количество очков в 1 сете у 1 команды*, *Количество очков в 1 сете у 2 команды*, *Количество очков во 2 сете у 1 команды*, *Количество очков во 2 сете у 2 команды*, *Количество очков в 3 сете у 1 команды*, *Количество очков в 3 сете у 2 команды*, *Количество очков в 4 сете у 1 команды*, *Количество очков в 4 сете у 2 команды*, *Количество очков в 5 сете у 1 команды*, *Количество очков в 5 сете у 2 команды*.

Таблица 13. Схема отношения МАТЧИ (Matches)

Содержание поля	*Имя поля*	*Тип, длина*	*Примечания*
Номер матча	n_match	N(6)	**суррогатный первичный ключ**
Дата и время проведения	m_date	T(timestap)	обязательное поле
Количество очков в 1 сете у 1 команды	points1_1s	N(2)	обязательное поле
Количество очков в 1 сете у 2 команды	points2_1s	N(2)	обязательное поле
Количество очков во 2 сете у 1 команды	points1_2s	N(2)	обязательное поле
Количество очков во 2 сете у 2 команды	points2_2s	N(2)	обязательное поле
Количество очков в 3 сете у 1 команды	points1_3s	N(2)	обязательное поле
Количество очков в 3 сете у 2 команды	points2_3s	N(2)	обязательное поле
Количество очков в 4 сете у 1 команды	points1_4s	N(2)	необязательное поле
Количество очков в 4 сете у 2 команды	points2_4s	N(2)	необязательное поле
Количество очков в 5 сете у 1 команды	points1_5s	N(2)	необязательное поле
Количество очков в 5 сете у 2 команды	points2_5s	N(2)	необязательное поле
Команда 1	to_num	N(6)	Обязательное поле, внешний ключ (к teams)

Команда 2	tt_num	N(6)	Обязательное поле, внешний ключ (к teams)
Зал	t_gym	N(3)	Обязательное поле, внешний ключ (к gyms)

Схема базы данных после нормализации приведена на Рис. 3

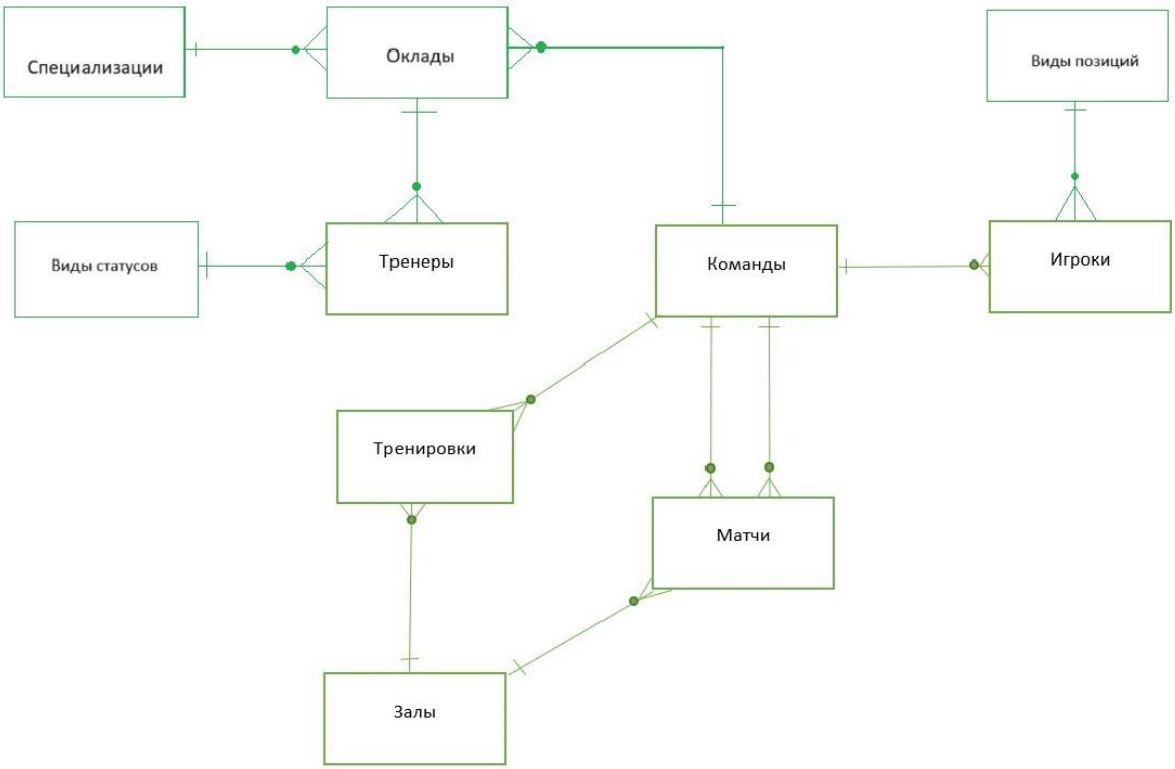

Рис 3. Окончательная схема БД волейбольного клуба

4.4 Определение дополнительных ограничений целостности

Перечислим ограничения целостности, которые не указаны в табл. 1–12.

1. Дата начала работы тренера должна быть хотя бы на 16 лет больше, чем дата его рождения, т.к, по законам РФ, заключение трудового договора разрешается лицам, достигшим возраста шестнадцати лет.
2. Матчи, проходящие в одном зале, не должны пересекаться по времени, также у каждой команды не должно быть пересечений времени матчей.
3. Тренировки, проходящие в одном зале, не должны пересекаться по времени, также у каждой команды не должно быть пересечений времени тренировок.
4. В каждом сыгранном сете 1-4, если у одной из команд количество очков < 24, то у другой команды количество очков не может превышать 25, если у обоих команд количество очков в сете >= 24, то разность между очками команд должна быть равна 2. В сыгранном 5 сете, если у одной из команд количество очков < 14, то у другой команды количество очков не может превышать 15, если у обоих команд количество очков в 5 сете >= 14, то разность между очками команд все так же должна быть равна

Эти ограничения нельзя реализовать в схеме отношения. В реальных БД подобные ограничения целостности реализуются вручную или программно (через внешнее приложение или специальную процедуру контроля данных – триггер).

4.5 Описание групп пользователей и прав доступа

Опишем для каждой группы пользователей права доступа к каждой таблице (Таблица 15). Права доступа должны быть распределены так, чтобы для каждого объекта БД был хотя бы один пользователь, который имеет право добавлять и удалять данные из объекта. Ниже описаны используемые сокращения (Таблица 14).

Таблица 14. Обозначения прав доступа

Обозначение	Расшифровка
I (insert)	добавление данных
S (select)	чтение данных
U (update)	модификация данных
D (delete)	удаление данных

Права администраторов клубов на изменение данных в таблицах ***Игроки***, ***Тренеры***, ***Команды***, ***Матчи***, ***Оклады***, ***Специализации*** и ***Виды статусов*** будут назначены через представления, т.к. администратор клуба может изменять данные, связанные только с его командой, права тренерам на изменение данных в таблице ***Виды позиций***, ***Тренировки*** а также на просмотр данных в таблице ***Игроки***, ***Тренировки***, ***Оклады***, ***Тренеры***, ***Специализации*** тоже будут назначены через представления, права игрокам на просмотр таблицы ***Игроки***, ***Тренеры***, ***Тренировки*** также будут назначены через

представления, и права болельщикам на таблицы ***Игроки***, ***Тренеры*** будут назначены через представления, чтобы скрыть личные данные.

Права назначает администратор БД.

Таблица 15. Права доступа к таблицам для групп пользователей

Таблицы	Группы пользователей (роли)			
	Администратор клуба	Тренеры	Игроки	Болельщики
Игроки	SUID	S	S	S
Виды позиций	S	SUID	S	
Тренеры	SUID	S	S	S
Оклады	SUID	S		
Специализации	SUID	S		
Виды статусов	SUID	S		
Команды	SUID	S	S	S
Матчи	SUID	S	S	S
Залы	SUID	S	S	S
Тренировки	S	SUID	S	

4.6. Реализация проекта базы данных

4.6.1. Создание таблиц

1. Отношение Виды позиций (Types)

```
create table types (
    t_name VARCHAR(30) PRIMARY KEY
);
```

2. Отношение Специализации (Posts)

```
create table posts(
    post varchar(30) primary key
);
```

3. Отношение Виды статусов (StatusTypes)

```
create table status_types (
    status_name varchar(15) primary key
);
```

4. Отношение Команды (Teams)

```
create table teams (
    t_num numeric(3) primary key,
    t_name varchar(30) not null,
    p_count numeric(2) not null
);
```

5. Отношение Оклады (Salaries)

```
create table salaries (
    team_num numeric(6) not null,
    post varchar(30) not null,
    salary numeric(8,2) check (salary > 14000),
    primary key (team_num, post),
    foreign key (team_num) references teams(t_num),
    foreign key (post) references posts(post)
);
```

6. Отношение Игроки (Players)

```
create table players (
    player_id numeric(6) primary key,
    p_surname varchar(30) not null,
    p_name varchar(30) not null,
    p_patronymic varchar(50),
    p_birth date not null,
```

```
    p_gender char(1) not null check (p_gender in
('м', 'ж')),
    p_passport char(10) unique not null,
    p_date date not null,
    p_given varchar(50) not null,
    p_phone varchar(60),
    p_inn char(12) not null unique,
    height numeric(3) not null,
    weight numeric(3) not null,
    position varchar(30) not null,
    team_num numeric(6) not null,
    foreign key (position) references types(t_name),
    foreign key (team_num) references teams(t_num)
);
```

7. Отношение Тренеры (Coaches)

```
create table coaches (
    coach_id numeric(6) primary key,
    c_surname varchar(30) not null,
    c_name varchar(30) not null,
    c_patronymic varchar(50) not null,
    c_birth date not null,
    c_gender char(1) not null check (c_gender in
('м', 'ж')),
    c_passport char(10) not null unique,
    c_date date not null,
    c_given varchar(50) not null,
    c_inn char(12) not null unique,
    c_phone varchar(15),
    expwork numeric(5,2) not null,
    tc_num char(3) not null,
    post varchar(30) not null,
    status varchar(15) not null,
    foreign key (tc_num, post) references
salaries(team_num, post),
    foreign key (status) references
status_types(status_name)
);
```

8. Отношение Залы (Gyms)

```
create table gyms (
    n_gym numeric(3) primary key,
    address varchar(60) not null,
    capacity numeric(4) not null,
    g_type varchar(20) not null
);
```

9. Отношение Матчи (Matches)

```
create table matches (
    n_match numeric(6) primary key,
    m_date timestamp not null,
    points1_1s numeric(2) not null,
    points2_1s numeric(2) not null,
    points1_2s numeric(2) not null,
    points2_2s numeric(2) not null,
    points1_3s numeric(2) not null,
    points2_3s numeric(2) not null,
    points1_4s numeric(2),
    points2_4s numeric(2),
    points1_5s numeric(2),
    points2_5s numeric(2),
    t1_num numeric(3) not null,
    t2_num numeric(3) not null,
    n_gym numeric(3) not null,
    foreign key (t1_num) references teams(t_num),
    foreign key (t2_num) references teams(t_num),
    foreign key (n_gym) references gyms(n_gym)
);
```

10. Отношение Тренировки (Training)

```
create table training (
    tr_date timestamp not null,
    tr_team numeric(3) not null,
    t_train numeric(3) not null,
    foreign key (tr_team) references teams(t_num),
    foreign key (t_train) references gyms(n_gym)
);
```

Начальный состав данных для справочных таблиц. Содержание других таблиц не входит в проект БД и будет вводиться по мере поступления данных:

1) Таблица Виды позиций

```
insert into types (t_name) values
('связующий'),
('доигровщик'),
('центральный блокирующий'),
('диагональный'),
('либеро');
```

2) Таблица Виды статусов

```
insert into status_types (status_name) values
('работает'),
```

```
('в отпуске'),
('на больничном'),
('уволен');
```

3) Таблица Специализации

```
insert into posts (post) values
('главный тренер'),
('второй тренер'),
('тренер по физической подготовке'),
('тренер по анализу игры');
```

4.6.2. Создание представлений (готовых запросов)

Приведём примеры нескольких готовых запросов (представлений):

1) Список всех игроков с указанием их команды:

```
CREATE VIEW PlayerTeamView AS
SELECT p.player_id, p.p_surname, p.p_name,
p.p_patronymic, p.p_birth, p.p_gender, p.p_passport,
p.p_date, p.p_given, p.p_phone, p.p_inn, p.height,
p.weight, p.position, t.t_name AS team_name
FROM players p
INNER JOIN teams t ON p.team_num = t.t_num;
```

2) Список тренеров с их текущим статусом:

```
CREATE VIEW CoachStatus AS
SELECT c.coach_id, c.c_surname, c.c_name,
c.c_patronymic, c.c_birth, c.c_gender, c.c_passport,
c.c_date, c.c_given, c.c_phone, c.c_inn, c.start_date,
c.post, t.t_name AS team_name, st.status_name AS
current_status
FROM coaches c
INNER JOIN teams t ON c.tc_num = t.t_num
INNER JOIN status_types st ON c.status =
st.status_name;
```

3) Список матчей с датой и адресом зала и названиями команд:

```
CREATE VIEW MatchVenue AS
SELECT m.n_match, m.m_date, g.address AS venue_address,
t1.t_name AS team1_name,t2.t_name AS team2_name
FROM matches m
INNER JOIN gyms g ON m.n_gym = g.n_gym
```

```
INNER JOIN teams t1 ON m.t1_num = t1.t_num
INNER JOIN teams t2 ON m.t2_num = t2.t_num;
```

4) Список команд и количества игроков в каждой:

```
CREATE VIEW TeamPlayerCount AS
SELECT t.t_num, t.t_name, t.p_count, COUNT(p.player_id)
AS player_count
FROM teams t
LEFT JOIN players p ON t.t_num = p.team_num
GROUP BY t.t_num, t.t_name, t.p_count;
```

5) Список игроков с их ростом и весом, отсортированных по командам:

```
CREATE VIEW PlayerHeightWeight AS
SELECT p.player_id, p.p_surname, p.p_name,
p.p_patronymic, p.height, p.weight, t.t_name AS
team_name
FROM players p
INNER JOIN teams t ON p.team_num = t.t_num
ORDER BY t.t_name;
```

6) Список тренировок с указанием команды и адреса зала:

```
CREATE VIEW TrainingDetails AS
SELECT t.tr_date, tm.t_name AS team_name, g.address AS
gym_address
FROM training t
INNER JOIN teams tm ON t.tr_team = tm.t_num
INNER JOIN gyms g ON t.t_train = g.n_gym;
```

7) Список тренеров с их зарплатой, специализацией и команде, к которой они принадлежат.

```
CREATE VIEW CoachSpecialization AS
SELECT c.coach_id, c.c_surname, c.c_name,
c.c_patronymic, s.salary, c.post AS
specializationt.t_name AS team_name
FROM coaches c
JOIN salaries s ON c.tc_num::numeric = s.team_num AND
c.post = s.post
JOIN teams t ON s.team_num = t.t_num;
```

Для работы с этими представлениями соответствующим пользователям нужно определить права доступа к представлениям (Таблица 16).

Таблица 16. Права доступа к таблицам для групп пользователей

Таблицы	Группы пользователей (роли)			
	Администратор клуба	Тренеры	Игроки	Болельщики
Список всех игроков с указанием их команды	S	S		
Список тренеров с их текущим статусом	S	S		
Список матчей с датой и адресом зала	S	S	S	S
Список команд и количества игроков в каждой	S	S		
Список игроков с их ростом и весом, отсортированных по командам	S	S	S	S
Список тренировок с указанием команды и адреса зала	S	S	S	
Список тренеров с их зарплатой, специализацией и команде, к которой они принадлежат.	S	S		

4.6.3. Назначение прав доступа

Создадим роли:

```
create role club_administrator;
create role coach;
create role player;
create role fan;
```

Права доступа пользователей предоставляются с помощью команды GRANT. Определим их согласно таблицам 14 и 15.

Администратор клуба:

```
GRANT SELECT, INSERT, UPDATE, DELETE ON
    players,
    coaches,
    posts,
    status_types,
    teams,
    matches,
    gyms,
    salaries
TO club_administrator;
GRANT SELECT ON
    types,
    training
TO club_administrator;
```

Остальные права предоставляются по аналогии.

Представления:

```
-- Для администратора
GRANT SELECT ON PlayerTeam TO club_administrator
GRANT SELECT ON CoachStatus TO club_administrator;
GRANT SELECT ON MatchVenue TO club_administrator;
GRANT SELECT ON TeamPlayerCoun TO club_administrator
GRANT SELECT ON PlayerHeightWeight TO club_administrator;
GRANT SELECT ON MatchScoreSummary TO club_administrator;
GRANT SELECT ON CoachSpecialization TO club_administrator;
```

4.6.4. Создание триггеров

1) Проверка, что тренеру больше 18 лет.

```
CREATE OR REPLACE FUNCTION check_coach_age() RETURNS
TRIGGER AS $$
DECLARE
    birth_date DATE;
    age INT;
BEGIN
    birth_date := NEW.c_birth;
    age := DATE_PART('year', AGE(birth_date));
    IF age < 18 THEN
        RAISE EXCEPTION 'Возраст тренера % меньше 18
лет', NEW.coach_id;
    END IF;

    RETURN NEW;
```

```
  END;
  $$ LANGUAGE plpgsql;

  CREATE TRIGGER check_coach_age_trigger
  BEFORE INSERT OR UPDATE ON coaches
  FOR EACH ROW EXECUTE FUNCTION check_coach_age();
```

2) Проверка, что если у тренера стоит статус «в отпуске», «уволен» или «на больничном», то у него не может быть закрепленной команды, пока не будет стоять статус «работает»:

```
CREATE OR REPLACE FUNCTION check_coach_status()
RETURNS TRIGGER AS $$
BEGIN
    IF NEW.status IN ('в отпуске', 'уволен', 'на
больничном') THEN
        RAISE EXCEPTION 'Тренер со статусом % не может
быть закреплен за командой', NEW.status;
    END IF;
    RETURN NEW;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER check_coach_status_trigger
BEFORE INSERT OR UPDATE ON coaches
FOR EACH ROW
EXECUTE FUNCTION check_coach_status();
```

4.6.5. Создание индексов

Система автоматически создаёт индексы по первичным ключам и уникальным полям.

Анализ готовых запросов показывает, что для повышения эффективности работы с данными необходимо создать индексы для всех внешних ключей.

В данной БД случаи запроса и фильтра игроков по их фамилиям или именам будет встречаться часто, поэтому создадим соответствующие индексы.

Создадим эти индексы:

```
CREATE INDEX idx_players_position ON players
(position);
CREATE INDEX idx_players_team_num ON players
(team_num);
CREATE INDEX idx_coaches_post_tc_num ON coaches (post,
tc_num);
```

```
CREATE INDEX idx_coaches_status ON coaches (status);
CREATE INDEX idx_matches_t1_num ON matches (t1_num);
CREATE INDEX idx_matches_t2_num ON matches (t2_num);
CREATE INDEX idx_matches_n_gym ON matches (n_gym);
CREATE INDEX idx_training_tr_team_t_train ON training
(tr_team, t_train);

CREATE INDEX idx_players_p_surname_p_name ON players
(p_surname, p_name);
```

4.6.6. Разработка стратегии резервного копирования

Интенсивность обновления разработанной базы данных относительно низкая, поэтому для обеспечения сохранности вполне достаточно проводить полное резервное копирование БД раз в день. Такой график резервного копирования обеспечит достаточную защиту от потери данных в случае возникновения непредвиденных сбоев или ситуаций, таких как сбои оборудования, ошибки операторов или программного обеспечения. Нагрузка примерно одинакова во все периоды. Планируется хранение 30 резервных копий, каждый день создается одна полная резервная копия, и все резервные копии сохраняются в течение 30 дней. Таким образом, всегда будет доступно 30 последних резервных копий, что позволяет восстановить данные на любой день в течение последнего месяца.