Министерство науки и высшего образования Российской Федерации
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«Национальный исследовательский университет ИТМО»
(Университет ИТМО)

Министерство образования и науки

федеральное государственное автономное образовательное учреждение высшего образования

«Национальный исследовательский университет ИТМО»

Факультет инфокоммуникационных технологий

Отчет по лабораторной работе

по дисциплине: **«Проектирование и реализация баз данных»**

Выполнили: Титов Георгий
Закоурцев Андрей
Зюзин Владислав
Орлова Алёна
Маркозубова Анастасия

Группы: К3223, К3222, К3220

Проверила: Осетрова Ирина Станиславовна

Санкт-Петербург
2024

Введение

Создание базы данных для управления и учета спортивных мероприятий среди студентов внутри университета ИТМО. База данных будет включать функционал для регистрации команд, подачи заявок, фиксации результатов каждых проведенных игр, результатов команды на конкретном мероприятии и индивидуальных результатов участников, формирования отчетов. Она охватывает исключительно командные соревнования.

Соревнования охватывают следующие спортивные дисциплины: футбол, баскетбол, волейбол, теннис, регби, киберспорт, бег на длинные дистанции и другие виды спорта, где участвуют команды.

Организация спортивных мероприятий в вузе включает множество задач, таких как регистрация участников, учет заявок, распределение ресурсов, фиксация результатов и создание отчетности. Отсутствие автоматизированной системы усложняет эти процессы, повышает вероятность ошибок и снижает эффективность управления. Разработка базы данных решает эти проблемы, упрощая управление мероприятиями, минимизируя ошибки и позволяя оперативно получать данные по заявкам, результатам и достижениям. Это особенно актуально для университета ИТМО, где спортивные мероприятия занимают важное место в студенческой жизни.

1. Целеполагание

Цель: проектирование и реализация базы данных для для управления автоматизацией внутриуниверситетских спортивных мероприятий

Задачи:

1. **Создание углубленного сценария базы данных**

   Создана система для автоматизации управления спортивными мероприятиями, включающая функционал для регистрации, учета заявок, управления командами и участниками, фиксации результатов, учета ресурсов и формирования отчетов.

2. **Определение ключевых сущностей**
   Определены сущности: мероприятие, дисциплина, команда, участник, результат игры, заявка, формат, статус заявки.

3. **Создать логическое проектирование**
   - **Нормализация**: проведена до 3NF для устранения избыточности данных.
   - **Ключевые ограничения**: установлены первичные и внешние ключи, обязательные атрибуты.
   - **Денормализация**: добавлена избыточность для повышения производительности.
4. **Создать физическую модель**
   Разработаны таблицы, связи, типы данных и индексы, реализующие структуру базы данных.

5. **Написать основные запросы**
   Разработаны SQL-запросы для добавления, обновления, поиска данных и формирования отчетов о мероприятиях, заявках и результатах.

1.1 Создание углубленного сценария базы данных

На основе работы, выполненной в прошлом учебном семестре по дисциплине «Проектирование и реализация баз данных», предлагается расширить существующий сценарий базы данных, добавив новые функциональные возможности. Университету необходимо автоматизировать процесс организации и проведения спортивных мероприятий, связанных с определёнными спортивными дисциплинами и форматами проведения. Для каждого мероприятия фиксируются ключевые данные: сроки, дисциплина, формат, а также команды-участники.

Команды, состоящие из участников, подают заявки на участие в мероприятиях. Каждая заявка имеет статус, который может быть «Принята», «Отклонена» или «В работе». По завершении мероприятия система фиксирует результаты для команд. Для команды сохраняются такие данные, как занятое место и общий результат, а для каждого участника — его личные достижения, включая набранные очки, время выполнения или другие показатели, зависящие от дисциплины.

Вдобавок к автоматизации управления дисциплинами, заявками и результатами, система позволяет формировать отчёты о проведённых мероприятиях, а также отслеживать личные и командные достижения.

Этот сценарий охватывает все ключевые этапы организации спортивных мероприятий, от подачи заявок и распределения участников до учёта результатов и анализа использования ресурсов, что способствует эффективному управлению данными и оптимальному использованию ресурсов университета.

2. Определение ключевых объектов системы

2.1 Определение сущностей

Проанализировав сценарий, определим потенциальные объекты, которые должны быть представлены в реляционной базе данных.

Потенциальные объекты:

- Мероприятие
- Дисциплина
- Команда
- Результат игры
- Статус заявки
- Участник
- Команда-участник
- Заявка
- Формат

2.2 Интервью с сотрудником организации

Было проведено интервью с менеджером университета, курирующим организацию спортивных мероприятий. В ходе интервью были уточнены характеристики системы, а также определены атрибуты для хранения информации о ключевых объектах, таких как мероприятия, дисциплины, команды, участники и их результаты.

Интервьюер: Какие данные о каждом мероприятии следует хранить дополнительно к уже указанным (дисциплина, формат, сроки проведения)?
Менеджер: Помимо этого, важно фиксировать количество участников, список команд-участников, а также дополнительные требования, такие как необходимое оборудование или техническая поддержка.

Интервьюер: Какие параметры должны быть учтены для дисциплин?
Менеджер: Помимо названия дисциплины, необходимо хранить её описание, формат (командный или индивидуальный), а также перечень необходимого оборудования.

Интервьюер: Какие данные нужно фиксировать для заявок?
Менеджер: Важно хранить дату подачи заявки, её статус (например, «Принята», «Отклонена» или «В работе»), а также причину отклонения заявки, если она была отклонена.

Интервьюер: Какие параметры следует учитывать для команд?
Менеджер: Для каждой команды важно фиксировать её название, состав участников, а также количество очков, набранных командой в предыдущих соревнованиях.

Интервьюер: Какие данные необходимо сохранять для участников?
Менеджер: Для участников нужно хранить ФИО, дату рождения, пол и спортивные достижения.

Интервьюер: Какую информацию следует учитывать для результатов?
Менеджер: Для команды необходимо фиксировать итоговое место и общий результат (например, количество очков), а для участников — индивидуальные достижения.

Интервьюер: Какие дополнительные параметры следует учитывать для формата проведения мероприятий?
Менеджер: Формат должен учитывать, является ли формат мероприятия очным или онлайн, а также особенности распределения участников по командам.

Интервьюер: Нужно ли учитывать дополнительные параметры для команды-участника?
Менеджер: Да, необходимо фиксировать связь между командой и участником, а также учитывать вклад каждого участника в общий результат команды.

2.3 Определение обязательности атрибутов и уникальных идентификаторов

Далее определим для каждой сущности, какие атрибуты являются обязательными, а какие нет, а также назначим первичные и вторичные уникальные идентификаторы.

Таблица 2.1 — Сущность «Мероприятие»

Наименование атрибута	Обязательный/ необязательный (*/o)	Уникальный идентификатор (#)
ID_мероприятия	*	#
Название	*	
Место проведения	*	
Начало	*	
Конец	*	
Форматы_id_Формата	*	
Дисциплина_id_дисциплины	*	
Комментарий	o	

Таблица 2.2 — Сущность «Дисциплина»

Наименование атрибута	Обязательный/ необязательный (*/o)	Уникальный идентификатор (#)
id_Дисциплины	*	#
Название	*	
Описание	o	

Таблица 2.3 — Сущность «Команда»

Наименование атрибута	Обязательный/ необязательный (*/o)	Уникальный идентификатор (#)
id_Команды	*	#
Название	*	
Дата основания	*	
Тренер	o	
Комментарий	o	

Таблица 2.4 — Сущность «Участник»

Наименование атрибута	Обязательный/необязательный (*/o)	Уникальный идентификатор (#)
id_Участника	*	#
Имя	*	
Фамилия	*	
Отчество	o	
Пол	*	
Дата_Рождения	*	

Таблица 2.5 — Сущность «Команда_Участник»

Наименование атрибута	Обязательный/ необязательный (*/o)	Уникальный идентификатор (#)
id_Команда_Участник	*	#
Команда_id_Команды	*	
Участник_id_Участника	*	
Дата_вступления	*	
Дата_ухода	o	
Причина_ухода	o	
Достижения	o	

Таблица 2.6 — Сущность «Заявка»

Наименование атрибута	Обязательный/ необязательный (*/o)	Уникальный идентификатор (#)
id_Заявки	*	#
Мероприятие_id	*	
Команда_id	*	
Статус_заявки_id	*	
Место	o	
Количество_очков	o	

Таблица 2.7 — Сущность «Статус заявки»

Наименование атрибута	Обязательный/ необязательный (*/o)	Уникальный идентификатор (#)
id_Статуса	*	#
Название	*	
Описание	*	

Таблица 2.8 — Сущность «Результат игры»

Наименование атрибута	Обязательный/ необязательный (*/o)	Уникальный идентификатор (#)
id_Результата	*	#
Результат	*	
Дата_игры	*	
id_Заявки 1	*	
id_Заявки 2	*	
Очки команды 1	o	
Очки команды 2	o	

Таблица 2.9— Сущность «Форматы»

Наименование атрибута	Обязательный/ необязательный (*/o)	Уникальный идентификатор (#)
id_Формата	*	#
Название	*	
Описание	*	

2.4 Определение связей

Мы сформулировали бизнес-правила проекта:

- Каждое мероприятие должно быть организовано в одном формате.
- Каждое мероприятие относится к одной дисциплине.
- Каждое мероприятие может принимать несколько заявок.
- Каждая заявка должна быть подана одной командой.
- У каждого участника фиксируется уникальная информация.
- Каждая заявка должна иметь один статус, указывающий её состояние.
- Команда должна подать заявку на участие в мероприятии.
- Команда может состоять из одного или нескольких участников.
- Один участник может покидать одну команду и переходить в другую.
- Каждый результат игры относится к одному мероприятию.
- Каждый результат игры фиксирует участие двух команд (через заявки).
- Участник может покинуть команду с указанием даты ухода.

Таблица 2.10 — Матрица связей

Объект строки	Формат	Дисциплина	Мероприятие	Заявка	Команда	Статус заявки	Участник	Команда_участник	Результат игры
Формат	-		используется						
Дисциплина		-	относится						
Мероприятие	имеет	имеет	-	принимает					
Заявка			относится	-	подаётся	имеет			участвует
Команда				подаёт	-			содержит	
Статус заявки				присваивается		-			
Участник							-	состоит	
Команда_участник					включает		включает	-	
Результат игры				фиксирует					-

3. Логическое проектирование

С помощью программы MySQL WorkBench была составлена ER диаграмма, визуализирующая нашу базу данных, сущности, их атрибуты и связи между сущностями. Еще в прошлом семестре у нас была составлена диаграмма уже в третьей нормальной форме:

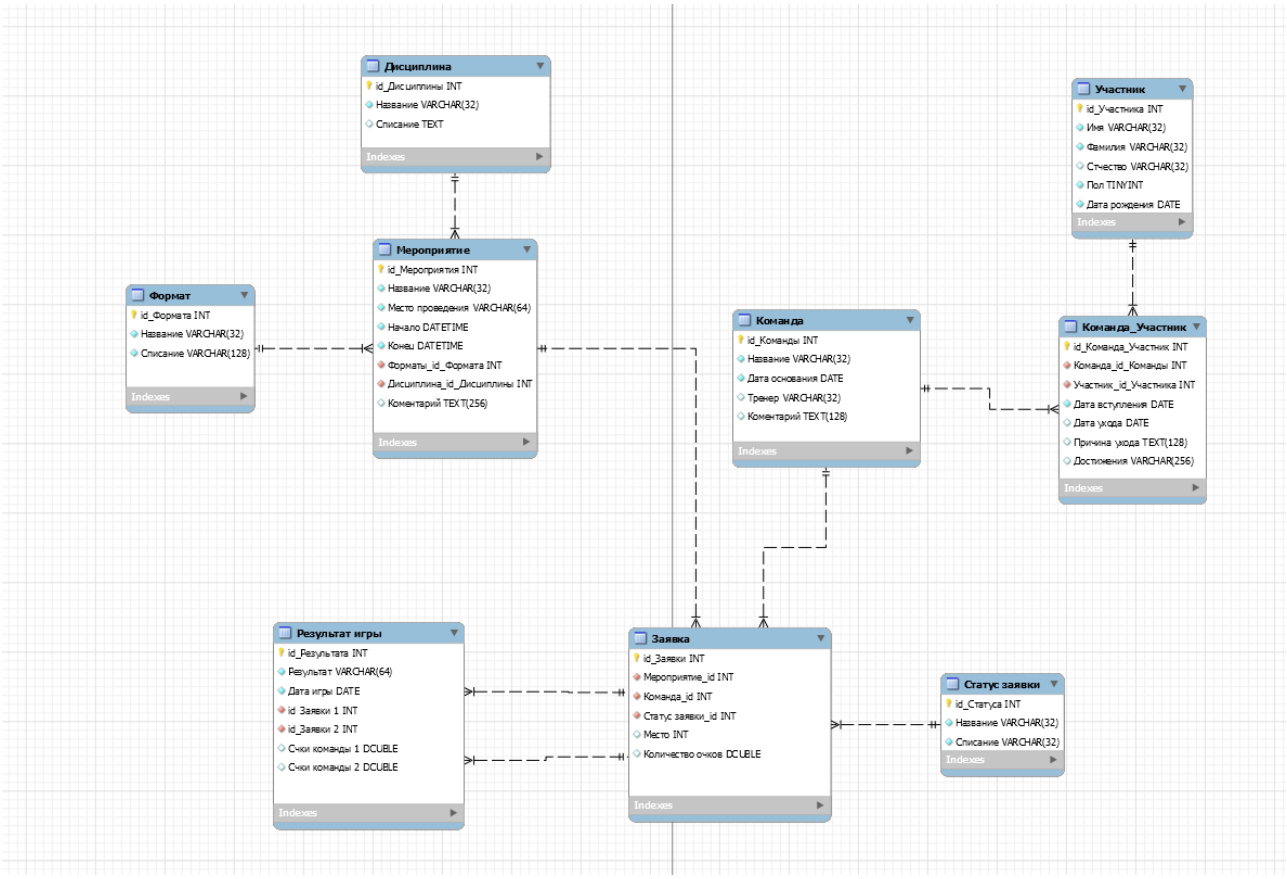


Рисунок 3.1 — Нормализация в третьей нормальной форме

Также была проведена денормализация. Был внесен ряд изменений:

1. Сокращение числа связей

В таблицу Мероприятия были добавлены атрибуты ПЕРВОЕ МЕСТО, ВТОРОЕ МЕСТО, ТРЕТЬЕ МЕСТО, которые указывают на команды, занявшие первое, второе и третье места в данном мероприятии. Это позволяет быстро получать информацию о призерах через один запрос к таблице, вместо выполнения соединений с таблицей результатов или выполнения агрегирующих запросов. Эти атрибуты формирует избыточные данные, так как информация о призерах может быть рассчитана на основе данных в таблице Результаты, но они добавлены для повышения удобства и производительности при анализе данных.

2. Устранение необходимости сложных вычислений

За счёт добавления атрибутов ПЕРВОЕ МЕСТО, ВТОРОЕ МЕСТО, ТРЕТЬЕ МЕСТО система избавляется от необходимости каждый раз вычислять победителей через фильтрацию и сортировку данных таблицы Результаты. Аналогично, атрибут Причина ухода устраняет необходимость дополнительного анализа данных для объяснения ухода участника.

3. Улучшение производительности запросов

Денормализация применяется для ускорения выполнения часто используемых запросов, таких как запросы на определение победителя или анализ состава команд. Несмотря на избыточность данных, эта модель позволяет избежать затратных операций соединения таблиц и сложных вычислений.

В результате данных изменений достигается баланс между производительностью запросов и удобством работы с данными, несмотря на небольшое увеличение объёма хранимой информации.

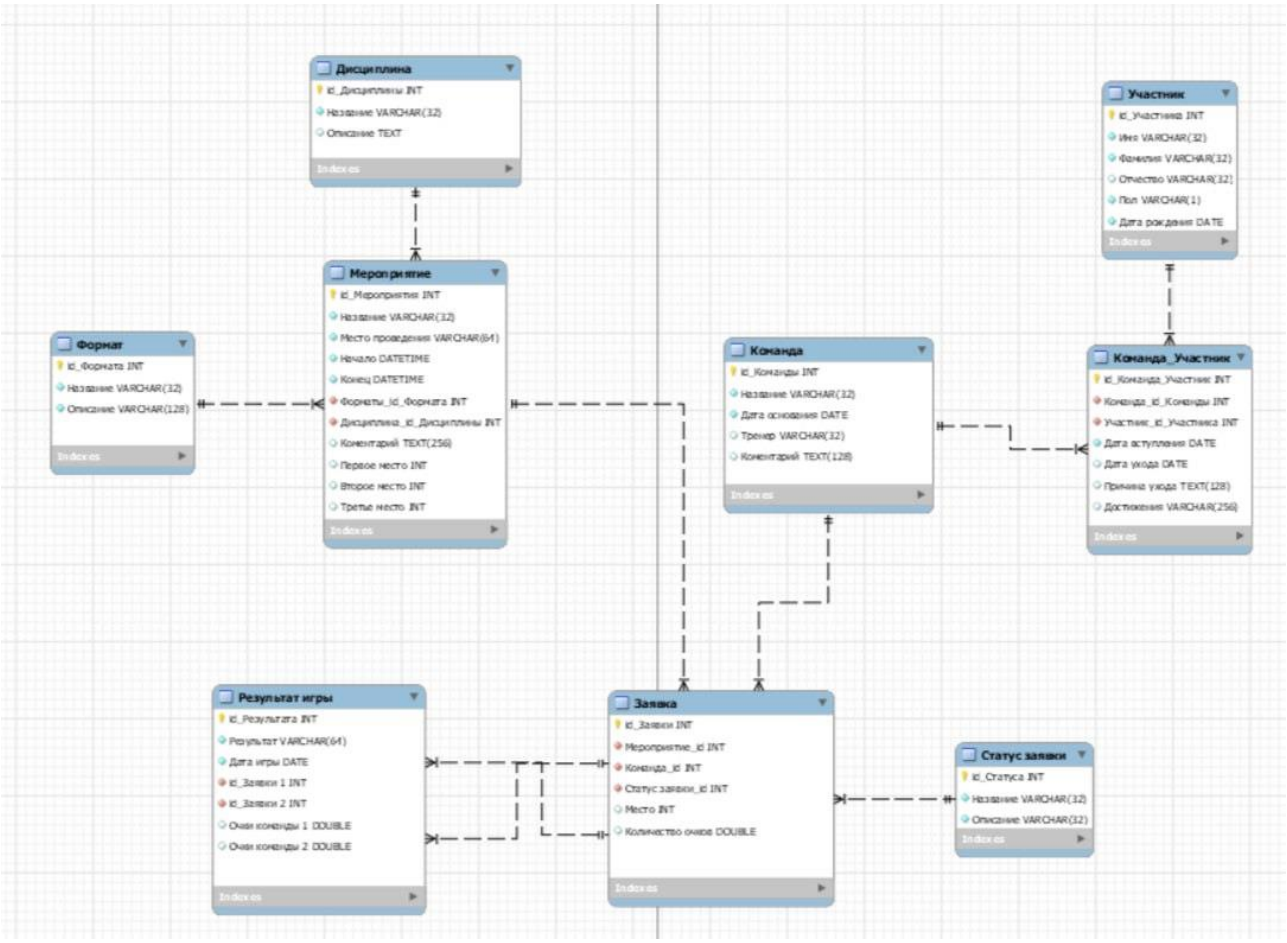


Рисунок 3.2 — Денормализированная диаграмма

4. Физическое проектирование

В этом разделе логическая база данных преобразуется в физическую. Для начала создаются таблицы для каждой сущности, где будет сопоставление обеих - логической и физической - моделей.

Таблица 4.1 — Сущность «Мероприятие»

Имя таблицы	**Краткое имя таблицы**			
Мероприятие	Мер			
Тип ключа	**Обяз.**	**Атрибут**	**Тип**	**Размер**
pk	#	id_Мероприятия	INT	4
	*	Название	VARCHAR(32)	32
	*	Место проведения	VARCHAR(64)	64
	*	Начало	DATETIME	3
	*	Конец	DATETIME	3
	o	Победитель	INT	4
	o	Второе место	INT	4
	o	Третье место	INT	4
fk1	*	Форматы_id_Формата	INT	4
fk2	*	Дисциплина_id_Дисциплины	INT	4

Таблица 4.2 – Сущность «Дисциплина»

Имя таблицы	**Краткое имя таблицы**			
Дисциплина	Дисц			
Тип ключа	**Обяз.**	**Атрибут**	**Тип**	**Размер**
pk	#	id_Дисциплины	INT	4
	*	Название	VARCHAR(32)	32
	o	Описание	TEXT(256)	256

Таблица 4.3 – Сущность «Формат»

Имя таблицы	**Краткое имя таблицы**			
Формат	Фмт			
Тип ключа	**Обяз.**	**Атрибут**	**Тип**	**Размер**
pk	#	id_Формата	INT	4
	*	Название	VARCHAR(32)	32
	*	Описание	VARCHAR(128)	128

Таблица 4.4 – Сущность «Команда»

Имя таблицы	**Краткое имя таблицы**			
Команда	Кмд			
Тип ключа	**Обяз.**	**Атрибут**	**Тип**	**Размер**
pk	#	id_Команды	INT	4
	*	Название	VARCHAR(32)	32
	o	Тренер	VARCHAR(32)	32
fk1	*	Дата основания	DATE	4

Таблица 4.5 – Сущность «Команда_Участник»

Имя таблицы	**Краткое имя таблицы**			
Команда-участник	Кмд-учс			
Тип ключа	**Обяз.**	**Атрибут**	**Тип**	**Размер**
pk	#	id_Команда_Участник	INT	4
fk	#	Команда_id_Команды	INT	4
		Участник_id_Участника	INT	4
	*	Дата вступления	DATE	3
	o	Дата ухода	DATE	3
	o	Причина ухода	TEXT(128)	128
	o	Достижения	VARCHAR(256)	256

Таблица 4.6 - Сущность «Участник»

Имя таблицы	**Краткое имя таблицы**			
Участник	Учс			
Тип ключа	**Обяз.**	**Атрибут**	**Тип**	**Размер**
pk	#	id_Участника	INT	4
	*	Имя	VARCHAR(32)	32
	*	Фамилия	VARCHAR(32)	32
	o	Отчество	VARCHAR(32)	32
	*	Пол	VARCHAR(1)	1
	*	Дата рождения	DATE	3

Таблица 4.7 - Сущность «Результат игры»

Имя таблицы	**Краткое имя таблицы**			
Результат игры	Рез_учс			
Тип ключа	**Обяз.**	**Атрибут**	**Тип**	**Размер**
pk	#	id_Результата	INT	4
	*	Результат	VARCHAR(64)	64
	*	Дата игры	DATE	3
fk1	#	id_Заявки 1	INT	4
fk2	#	id_Заявки 2	INT	4
	o	Очки команды 1	DOUBLE	8
	o	Очки команды 2	DOUBLE	8

Таблица 4.8 – Сущность «Заявка»

Имя таблицы	**Краткое имя таблицы**			
Заявка	Звк			
Тип ключа	**Обяз.**	**Атрибут**	**Тип**	**Размер**
pk	#	id_Заявки_	INT	4
fk1	*	Мероприятие_id	INT	4
fk2	*	Команда_id	INT	4
fk3	*	Статус заявки_id	INT	4
	o	Место	INT	4
	o	Количество очков	DOUBLE	8

Таблица 4.9 – Сущность «Статус заявки»

Имя таблицы	**Краткое имя таблицы**			
Статус заявки	Стс_звк			
Тип ключа	**Обяз.**	**Атрибут**	**Тип**	**Размер**
pk	#	id_Статуса	INT	4
	*	Название	VARCHAR(32)	32
	*	Описание	VARCHAR(32)	32

Далее логическая база данных преобразуется в физическую. Используется скрипт, сделанный на основе имеющихся данных, он копируется и переносится в СУБД. Получается такая картина:

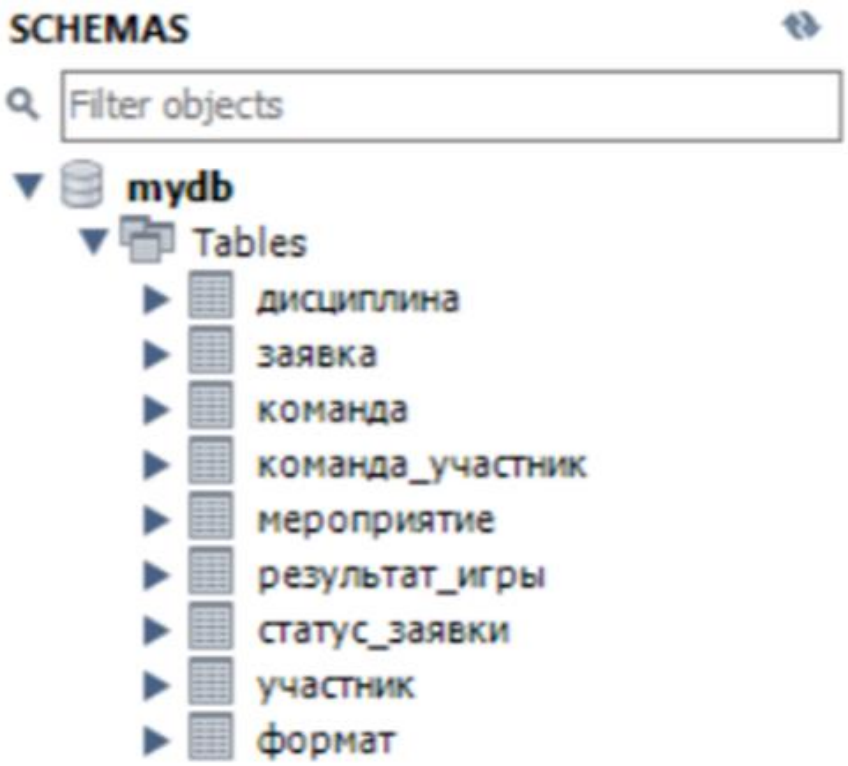


Рисунок 4.10 - Структура физической модели базы данных

Для примера берется сущность «дисциплина», она будет выглядеть таким образом:

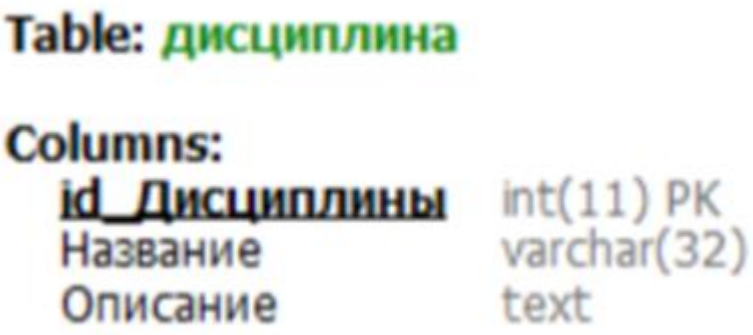


Рисунок 4.11 - Сущность «дисциплина»

5. Разработка запросов к базе данных

В данной лабораторной работе были выполнены различные запросы к базе данных, которые демонстрируют использование основных компонентов языка SQL и его подъязыков, таких как DDL, DML, DQL, TCL.

5.1 Операторы языка DDL

Предположим, что менеджеру необходимо знать, кто проверяет каждую из заявок, для этого столбец «проверяющий» с типом данных varchar (32) вставляется в таблицу ЗАЯВКА с помощью оператора ALTER TABLE и модификатора ADD.

```
201 •    alter table `заявка`
202      add `проверяющий` varchar(32);
```

Рисунок 5.1.1 - Команда добавляет в таблицу ЗАЯВКА столбец “проверяющий”

id_Заявки	Мероприятие_id	Команда_id	Статус_заявки_id	Место	Количество_очков
1	1	1	1	1	11
2	1	2	1	2	9
3	2	1	1	2	14
4	2	2	1	1	17
5	3	1	1	1	25
6	3	2	1	2	22
7	4	1	3	NULL	NULL
8	4	2	3	NULL	NULL
9	5	1	1	1	10
10	5	2	2	NULL	NULL
11	6	1	3	NULL	NULL
12	6	2	1	1	11
13	7	1	1	1	11
14	7	2	1	2	6
15	8	1	3	NULL	NULL
16	8	2	3	NULL	NULL
17	9	1	2	NULL	NULL
18	9	2	1	1	29
19	10	1	1	1	16
20	10	2	1	2	12
NULL	NULL	NULL	NULL	NULL	NULL

Рисунок 5.1.2 - Таблица ЗАЯВКА до добавления столбца “проверяющий”

id_Заявки	Мероприятие_id	Команда_id	Статус_заявки_id	Место	Количество_очков	проверяющий
1	1	1	1	1	11	Иванов
2	1	2	1	2	9	Петров
3	2	1	1	2	14	Сидоров
4	2	2	1	1	17	Кузнецов
5	3	1	1	1	25	Смирнов
6	3	2	1	2	22	Попов
7	4	1	3	NULL	NULL	Васильев
8	4	2	3	NULL	NULL	Михайлов
9	5	1	1	1	10	Фёдоров
10	5	2	2	NULL	NULL	Алексеев
11	6	1	3	NULL	NULL	Андреев
12	6	2	1	1	11	Николаев
13	7	1	1	1	11	Лебедев
14	7	2	1	2	6	Семенов
15	8	1	3	NULL	NULL	Егоров
16	8	2	3	NULL	NULL	Павлов
17	9	1	2	NULL	NULL	Козлов
18	9	2	1	1	29	Степанов
19	10	1	1	1	16	Зайцев
20	10	2	1	2	12	Соловьев
NULL	NULL	NULL	NULL	NULL	NULL	NULL

Рисунок 5.1.3 - Таблица ЗАЯВКА после добавления столбца “проверяющий”

Аналогично будет выглядеть добавление колонки в таблицу СТАТУС_ЗАЯВКИ:

```
405    alter table `статус_заявки`
406    add `рекомендации` varchar(100);
```

Рисунок 5.1.4 - Команда добавляет в таблицу СТАТУС_ЗАЯВКИ столбец “рекомендации”

id_Статуса	Название	Описание
1	Принята	Проверяющий принял заявку
2	В работе	Идёт проверка заявки
3	Отклонена	Проверяющий отклонил Вашу заявку
NULL	NULL	NULL

Рисунок 5.1.5 - Таблица СТАТУС_ЗАЯВКИ до добавления столбца “рекомендации”

id_Статуса	Название	Описание	рекомендации
1	Принята	Проверяющий принял заявку	Озвучте результат заявки Вашему тренеру
2	В работе	Идёт проверка заявки	Пожалуйста, подождите
3	Отклонена	Проверяющий отклонил Вашу заявку	При несогласии с результатами заявки обратитесь в студофис
NULL	NULL	NULL	NULL

Рисунок 5.1.6 - Таблица СТАТУС_ЗАЯВКИ после добавления столбца “рекомендации”

Участник соревнований должен быть 18-летнего возраста или старше, поэтому устанавливается следующий триггер:

```
DELIMITER $$
CREATE TRIGGER check_age_before_insert
BEFORE INSERT ON `участник`
FOR EACH ROW
BEGIN
    IF NEW.`Дата_рождения` > CURDATE() - INTERVAL 18 YEAR THEN
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Возраст участника должен быть 18 лет или старше';
    END IF;
END $$

DELIMITER ;
```

Рисунок 5.1.7 - Триггер на проверку возраста

Если участники удовлетворяют всем характеристикам, то будет следующий вывод:

```
INSERT INTO `участник` (`id_Участника`, `Имя`, `Фамилия`, `Отчество`, `Пол`, `Дата_рождения`)
values (15, 'Участников', 'Игрок', 'Победович', 'м', '2006-01-01');

select * from `участник`;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

id_Участника	Имя	Фамилия	Отчество	Пол	Дата_рождения
1	Иван	Иванов	Иванович	м	2010-01-01
2	Петр	Петров	Петрович	м	2004-07-22
3	Мария	Смирнова	Игоревна	ж	2003-03-15
4	Елена	Кузнецова	Анатольевна	ж	2002-09-03
5	Алексей	Захаров	Вячеславович	м	2005-01-25
6	Анна	Федорова	Александровна	ж	2005-12-09
7	Сергей	Михайлов	Александрович	м	2003-05-14
8	Ольга	Васильева	Сергеевна	ж	2004-11-23
9	Дмитрий	Ковалёв	Иванович	м	2006-02-18
10	Татьяна	Ильина	Петровна	ж	2003-08-09
11	Николай	Попов	Михайлович	м	2005-06-30
12	Виктория	Громова	Алексеевна	ж	2002-12-15
13	Евгений	Соколов	Владимирович	м	2004-03-21
14	Елена	Орлова	Сергеевна	ж	2006-09-17
15	Участников	Игрок	Победович	м	2006-01-01

участник 5

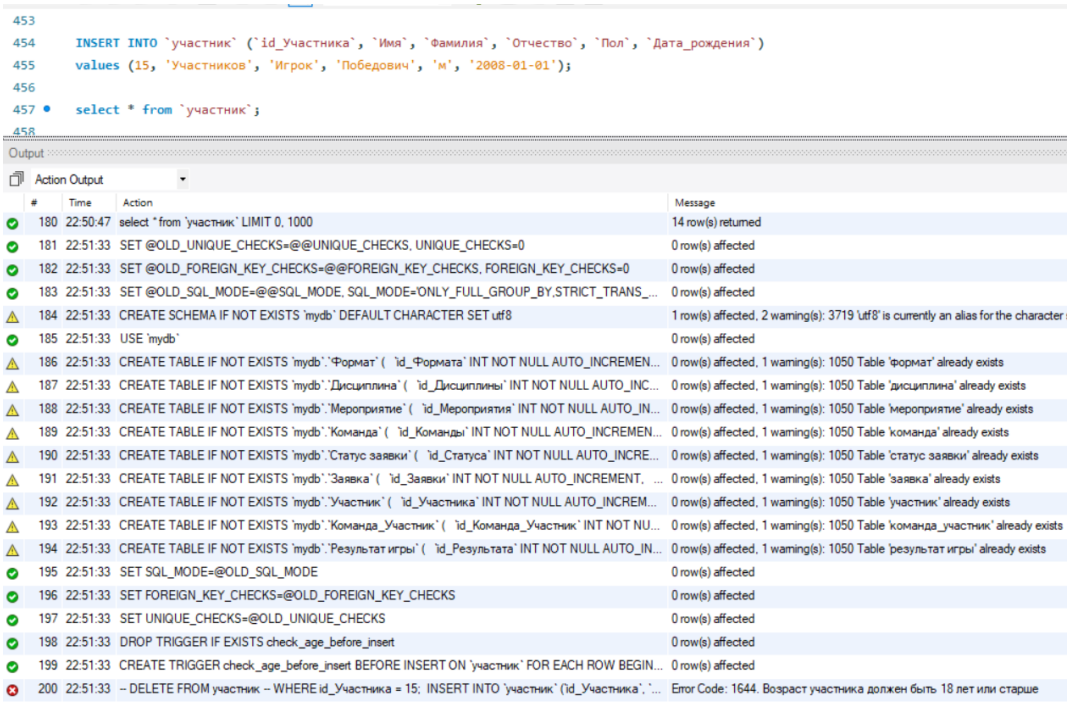
Output

Action Output

#	Time	Action	Message
157	22:49:58	CREATE TRIGGER check_age_before_insert BEFORE INSERT ON `участник` FOR EACH ROW BEGIN...	0 row(s) affected
158	22:49:58	INSERT INTO `участник` (`id_Участника`, `Имя`, `Фамилия`, `Отчество`, `Пол`, `Дата_рождения`) val...	1 row(s) affected
159	22:49:58	select * from `участник` LIMIT 0, 1000	15 row(s) returned

Рисунок 5.1.8 - Вывод при верных характеристиках

Иначе - представленный ниже:



```
453
454     INSERT INTO `участник` (`id_Участника`, `Имя`, `Фамилия`, `Отчество`, `Пол`, `Дата_рождения`)
455     values (15, 'Участников', 'Игрок', 'Победович', 'м', '2008-01-01');
456
457 •   select * from `участник`;
458
```

Output

Action Output

#	Time	Action	Message
180	22:50:47	select * from `участник` LIMIT 0, 1000	14 row(s) returned
181	22:51:33	SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0	0 row(s) affected
182	22:51:33	SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0	0 row(s) affected
183	22:51:33	SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='ONLY_FULL_GROUP_BY,STRICT_TRANS_...	0 row(s) affected
184	22:51:33	CREATE SCHEMA IF NOT EXISTS `mydb` DEFAULT CHARACTER SET utf8	1 row(s) affected, 2 warning(s): 3719 'utf8' is currently an alias for the character
185	22:51:33	USE `mydb`	0 row(s) affected
186	22:51:33	CREATE TABLE IF NOT EXISTS `mydb`.`Формат` (`id_Формата` INT NOT NULL AUTO_INCREMEN...	0 row(s) affected, 1 warning(s): 1050 Table 'формат' already exists
187	22:51:33	CREATE TABLE IF NOT EXISTS `mydb`.`Дисциплина` (`id_Дисциплины` INT NOT NULL AUTO_INC...	0 row(s) affected, 1 warning(s): 1050 Table 'дисциплина' already exists
188	22:51:33	CREATE TABLE IF NOT EXISTS `mydb`.`Мероприятие` (`id_Мероприятия` INT NOT NULL AUTO_IN...	0 row(s) affected, 1 warning(s): 1050 Table 'мероприятие' already exists
189	22:51:33	CREATE TABLE IF NOT EXISTS `mydb`.`Команда` (`id_Команды` INT NOT NULL AUTO_INCREMEN...	0 row(s) affected, 1 warning(s): 1050 Table 'команда' already exists
190	22:51:33	CREATE TABLE IF NOT EXISTS `mydb`.`Статус заявки` (`id_Статуса` INT NOT NULL AUTO_INCRE...	0 row(s) affected, 1 warning(s): 1050 Table 'статус заявки' already exists
191	22:51:33	CREATE TABLE IF NOT EXISTS `mydb`.`Заявка` (`id_Заявки` INT NOT NULL AUTO_INCREMENT, ...	0 row(s) affected, 1 warning(s): 1050 Table 'заявка' already exists
192	22:51:33	CREATE TABLE IF NOT EXISTS `mydb`.`Участник` (`id_Участника` INT NOT NULL AUTO_INCREM...	0 row(s) affected, 1 warning(s): 1050 Table 'участник' already exists
193	22:51:33	CREATE TABLE IF NOT EXISTS `mydb`.`Команда_Участник` (`id_Команда_Участник` INT NOT NU...	0 row(s) affected, 1 warning(s): 1050 Table 'команда_участник' already exists
194	22:51:33	CREATE TABLE IF NOT EXISTS `mydb`.`Результат игры` (`id_Результата` INT NOT NULL AUTO_IN...	0 row(s) affected, 1 warning(s): 1050 Table 'результат игры' already exists
195	22:51:33	SET SQL_MODE=@OLD_SQL_MODE	0 row(s) affected
196	22:51:33	SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS	0 row(s) affected
197	22:51:33	SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS	0 row(s) affected
198	22:51:33	DROP TRIGGER IF EXISTS check_age_before_insert	0 row(s) affected
199	22:51:33	CREATE TRIGGER check_age_before_insert BEFORE INSERT ON `участник` FOR EACH ROW BEGIN...	0 row(s) affected
200	22:51:33	-- DELETE FROM участник -- WHERE id_Участника = 15; INSERT INTO `участник` (`id_Участника`, `...	Error Code: 1644. Возраст участника должен быть 18 лет или старше

Рисунок 5.1.9 - Вывод при неверных характеристиках

5.2 Операторы языка DML и TCL

Необходимо добавить новые заявки в таблицу ЗАЯВКА и обновить записи в таблице МЕРОПРИЯТИЕ, чтобы команды участвовали в новом мероприятии и для этого были необходимые заявки. Все операции выполнены в рамках одной транзакции для обеспечения целостности данных.

В итоге выполнена транзакция, включающая добавление двух новых заявок в таблицу ЗАЯВКА и обновление записи в таблице МЕРОПРИЯТИЕ. Использование транзакции позволило обеспечить атомарность операций, гарантируя, что все изменения были успешно применены.

Здесь создаются строки для команды, далее начинается транзакция и в таблицу МЕРОПРИЯТИЕ вставляется информация. После мероприятие обновляется с указанием всех этих характеристик.

Выполняются следующие шаги:

1. Добавление новых строк в таблицу заявка - добавляются строки для нового мероприятия.
2. Изменение данных в таблице мероприятие - меняется место проведения одного из мероприятий.
3. Использование транзакции - фиксируются изменения с использованием языка TCL (COMMIT).

```
START TRANSACTION;

INSERT INTO `заявка` (`id_Заявки`, `Мероприятие_id`, `Команда_id`, `Статус_заявки_id`, `Место`, `Количество_очков`, `проверяющий`)
VALUES
    (21, 1, 1, 2, 0, 0, 'Проверяющев'),
    (22, 1, 2, 2, 0, 0, 'Проверяенко');

UPDATE `мероприятие`
SET `Место_проведения` = 'Спортзал', Начало = '2024-04-01 10:00:00', Конец = '2024-04-01 15:00:00'
WHERE `id_Мероприятия` = 1;

SELECT * FROM `заявка`;
SELECT * FROM `мероприятие`;

COMMIT;
```

Рисунок 5.2.1 - Запрос на выполнение транзакции по добавлению и обновлению строки

504	19:08:08	START TRANSACTION	0 row(s) affected		0.000 sec
505	19:08:08	INSERT INTO `заявка` (`id_Заявки`, `Мероприятие_id`, `Команда_id`, `Статус_заявки_id`, `Место`, ...	2 row(s) affected Records: 2 Duplicates: 0 Warnings: 0		0.000 sec
506	19:08:08	UPDATE `мероприятие` SET `Место_проведения` = 'Спортзал', Начало = '2024-04-01 10:00:00', Ко...	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0		0.000 sec
507	19:08:08	SELECT * FROM `заявка` LIMIT 0, 1000	22 row(s) returned		0.000 sec / 0.000 sec
508	19:08:08	SELECT * FROM `мероприятие` LIMIT 0, 1000	10 row(s) returned		0.000 sec / 0.000 sec
509	19:08:08	COMMIT	0 row(s) affected		0.031 sec

Рисунок 5.2.2 - Итог выполнения транзакции по добавлению и обновлению строки

id_Заявки	Мероприятие_id	Команда_id	Статус_заявки_id	Место	Количество_очков	проверяющий
1	1	1	1	1	11	Иванов
2	1	2	1	2	9	Петров
3	2	1	1	2	14	Сидоров
4	2	2	1	1	17	Кузнецов
5	3	1	1	1	25	Смирнов
6	3	2	1	2	22	Попов
7	4	1	3	NULL	NULL	Васильев
8	4	2	3	NULL	NULL	Михайлов
9	5	1	1	1	10	Фёдоров
10	5	2	2	NULL	NULL	Алексеев
11	6	1	3	NULL	NULL	Андреев
12	6	2	1	1	11	Николаев
13	7	1	1	1	11	Лебедев
14	7	2	1	2	6	Семенов
15	8	1	3	NULL	NULL	Егоров
16	8	2	3	NULL	NULL	Павлов
17	9	1	2	NULL	NULL	Козлов
18	9	2	1	1	29	Степанов
19	10	1	1	1	16	Зайцев
20	10	2	1	2	12	Соловьев
NULL	NULL	NULL	NULL	NULL	NULL	NULL

Рисунок 5.2.3 - Таблица ЗАЯВКА до транзакции по добавлению и обновлению строки

id_Заявки	Мероприятие_id	Команда_id	Статус_заявки_id	Место	Количество_очков	проверяющий
1	1	1	1	1	11	Иванов
2	1	2	1	2	9	Петров
3	2	1	1	2	14	Сидоров
4	2	2	1	1	17	Кузнецов
5	3	1	1	1	25	Смирнов
6	3	2	1	2	22	Попов
7	4	1	3	NULL	NULL	Васильев
8	4	2	3	NULL	NULL	Михайлов
9	5	1	1	1	10	Фёдоров
10	5	2	2	NULL	NULL	Алексеев
11	6	1	3	NULL	NULL	Андреев
12	6	2	1	1	11	Николаев
13	7	1	1	1	11	Лебедев
14	7	2	1	2	6	Семенов
15	8	1	3	NULL	NULL	Егоров
16	8	2	3	NULL	NULL	Павлов
17	9	1	2	NULL	NULL	Козлов
18	9	2	1	1	29	Степанов
19	10	1	1	1	16	Зайцев
20	10	2	1	2	12	Соловьев
21	1	1	2	NULL	NULL	Проверяющев
22	1	2	2	NULL	NULL	Проверяенко
NULL	NULL	NULL	NULL	NULL	NULL	NULL

Рисунок 5.2.4 - Таблица ЗАЯВКА после транзакции по добавлению и обновлению строки

id_Мероприятия	Название	Место_проведения	Начало	Конец	Формат_id_Формата	Дисциплина_id_Дисциплины	Комментарий
1	Турнир по футболу	Стадион	2023-04-01 10:00:00	2023-04-01 15:00:00	1	1	Студенческий чемпионат по футболу
2	Баскетбольный чемпионат	Спортзал	2023-04-02 12:00:00	2023-04-02 18:00:00	1	2	Турнир среди команд факультетов
3	Волейбольный турнир	Спортзал	2023-05-10 09:00:00	2023-05-10 14:00:00	1	3	Соревнования среди студентов
4	Соревнования по лёгкой атлетике	Стадион	2023-06-15 08:00:00	2023-06-15 12:00:00	1	4	Студенческое первенство по лёгкой атлетике
5	Теннисный матч	Корт	2023-07-01 10:00:00	2023-07-01 13:00:00	1	5	Студенческий теннисный турнир
6	Турнир по настольному теннису	Игровая зона	2023-07-15 14:00:00	2023-07-15 17:00:00	1	6	Турнир среди студентов университета
7	Шахматный турнир	Онлайн платформа	2023-08-05 09:00:00	2023-08-05 13:00:00	2	7	Интеллектуальные игры среди студентов
8	Бадминтонный чемпионат	Спортзал	2023-09-10 15:00:00	2023-09-10 19:00:00	1	8	Соревнования по бадминтону среди студентов
9	Киберспортивный турнир	Онлайн платформа	2023-10-20 18:00:00	2023-10-20 22:00:00	2	9	Турнир по CSGO и DOTA2 среди студентов
10	Забег на длинные дистанции	Парк	2023-11-01 07:00:00	2023-11-01 11:00:00	1	10	Соревнования по бегу среди студентов

Рисунок 5.2.5 - Таблица МЕРОПРИЯТИЕ до транзакции по добавлению и обновлению строки

id_Мероприятия	Название	Место_проведения	Начало	Конец	Формат_id_Формата	Дисциплина_id_Дисциплины	Комментарий
1	Турнир по футболу	Спортзал	2024-04-01 10:00:00	2024-04-01 15:00:00	1	1	Студенческий чемпионат по футболу
2	Баскетбольный чемпионат	Спортзал	2023-04-02 12:00:00	2023-04-02 18:00:00	1	2	Турнир среди команд факультетов
3	Волейбольный турнир	Спортзал	2023-05-10 09:00:00	2023-05-10 14:00:00	1	3	Соревнования среди студентов
4	Соревнования по лёгкой атлетике	Стадион	2023-06-15 08:00:00	2023-06-15 12:00:00	1	4	Студенческое первенство по лёгкой атлетике
5	Теннисный матч	Корт	2023-07-01 10:00:00	2023-07-01 13:00:00	1	5	Студенческий теннисный турнир
6	Турнир по настольному теннису	Игровая зона	2023-07-15 14:00:00	2023-07-15 17:00:00	1	6	Турнир среди студентов университета
7	Шахматный турнир	Онлайн платформа	2023-08-05 09:00:00	2023-08-05 13:00:00	2	7	Интеллектуальные игры среди студентов
8	Бадминтонный чемпионат	Спортзал	2023-09-10 15:00:00	2023-09-10 19:00:00	1	8	Соревнования по бадминтону среди студентов
9	Киберспортивный турнир	Онлайн платформа	2023-10-20 18:00:00	2023-10-20 22:00:00	2	9	Турнир по CSGO и DOTA2 среди студентов
10	Забег на длинные дистанции	Парк	2023-11-01 07:00:00	2023-11-01 11:00:00	1	10	Соревнования по бегу среди студентов

Рисунок 5.2.6. – Таблица МЕРОПРИЯТИЕ после транзакции по добавлению и обновлению строки

5.3 Операторы языка DQL

Чтобы получить информацию об участниках первой команде, используется оператор SELECT в сочетании с условием WHERE, которое позволяет отобразить только те записи, которые соответствуют первой команде.

На этом этапе выбираются исключительно участники первой команды из общего списка. Сначала фильтруются данные по участнику команды, затем добавляется условие, ограничивающее выбор участников только первой командой. Сортировка данных, если она потребуется, будет выполнена на следующем этапе.

id_Команда_Участник	Команда_id_Команды	Участник_id_Участника	Дата_вступления	Дата_ухода	Причина_ухода	Достижения
1	1	1	2022-02-01	NULL	NULL	Победа в локальном турнире
2	1	2	2022-03-15	2023-05-20	Личные обстоятельства	Участие в межвузовском соревновании
3	1	3	2022-05-20	NULL	NULL	Призёр факультетских соревнований
4	1	4	2022-06-10	NULL	NULL	Лучший игрок месяца
5	1	5	2022-07-05	NULL	NULL	Участие в университетской олимпиаде
6	1	6	2022-09-01	NULL	NULL	Разработка стратегии команды
7	1	7	2022-10-15	NULL	NULL	Участие в региональном турнире
8	2	8	2022-02-12	NULL	NULL	Лучший новичок команды
9	2	9	2022-04-01	NULL	NULL	Серебряный призёр факультетского чемпионата
10	2	10	2022-05-25	2024-01-01	Окончание учёбы	Победа в межфакультетском соревновании
11	2	11	2022-07-15	NULL	NULL	Организация внутреннего чемпионата
12	2	12	2022-08-20	NULL	NULL	NULL
13	2	13	2022-10-05	NULL	NULL	Лидер по количеству забитых очков
14	2	14	2022-11-10	NULL	NULL	Участие в международном соревновании
NULL	NULL	NULL	NULL	NULL	NULL	NULL

Рисунок 5.3.1 - Таблица КОМАНДА_УЧАСТНИК до применения условия

```
450 •   select *
451     from `команда_участник`
452     where `Команда_id_Команды` = 1;
```

Рисунок 5.3.2 - Использованный код

id_Команда_Участник	Команда_id_Команды	Участник_id_Участника	Дата_вступления	Дата_ухода	Причина_ухода	Достижения
1	1	1	2022-02-01	NULL	NULL	Победа в локальном турнире
2	1	2	2022-03-15	2023-05-20	Личные обстоятельства	Участие в межвузовском соревновании
3	1	3	2022-05-20	NULL	NULL	Призёр факультетских соревнований
4	1	4	2022-06-10	NULL	NULL	Лучший игрок месяца
5	1	5	2022-07-05	NULL	NULL	Участие в университетской олимпиаде
6	1	6	2022-09-01	NULL	NULL	Разработка стратегии команды
7	1	7	2022-10-15	NULL	NULL	Участие в региональном турнире
NULL	NULL	NULL	NULL	NULL	NULL	NULL

Рисунок 5.3.3 - Таблица КОМАНДА_УЧАСТНИК после применения условия

Для выборки данных о студентках с учетом их возрастной категории и пола используется оператор SELECT. Условие WHERE применяется для фильтрации записей, позволяя выбрать только девушек, соответствующих указанному критерию. После этого с помощью ORDER BY строки упорядочиваются по дате рождения в порядке убывания, начиная с самых старших. Такой подход обеспечивает структурированное представление информации, упрощая анализ и работу с данными.

id_Участника	Имя	Фамилия	Отчество	Пол	Дата_рождения
1	Иван	Иванов	Иванович	м	2010-01-01
2	Петр	Петров	Петрович	м	2004-07-22
3	Мария	Смирнова	Игоревна	ж	2003-03-15
4	Елена	Кузнецова	Анатольевна	ж	2002-09-03
5	Алексей	Захаров	Вячеславович	м	2005-01-25
6	Анна	Федорова	Александровна	ж	2005-12-09
7	Сергей	Михайлов	Александрович	м	2003-05-14
8	Ольга	Васильева	Сергеевна	ж	2004-11-23
9	Дмитрий	Ковалёв	Иванович	м	2006-02-18
10	Татьяна	Ильина	Петровна	ж	2003-08-09
11	Николай	Попов	Михайлович	м	2005-06-30
12	Виктория	Громова	Алексеевна	ж	2002-12-15
13	Евгений	Соколов	Владимирович	м	2004-03-21
14	Елена	Орлова	Сергеевна	ж	2006-09-17
NULL	NULL	NULL	NULL	NULL	NULL

Рисунок 5.3.4 - Таблица УЧАСТНИК до применения условия

```
470 •   select *
471     from `участник`
472     where `Пол` = 'ж'
473     order by `Дата_рождения` ASC;
```

Рисунок 5.3.5 - Использованный код

id_Участника	Имя	Фамилия	Отчество	Пол	Дата_рождения
4	Елена	Кузнецова	Анатольевна	ж	2002-09-03
12	Виктория	Громова	Алексеевна	ж	2002-12-15
3	Мария	Смирнова	Игоревна	ж	2003-03-15
10	Татьяна	Ильина	Петровна	ж	2003-08-09
8	Ольга	Васильева	Сергеевна	ж	2004-11-23
6	Анна	Федорова	Александровна	ж	2005-12-09
14	Елена	Орлова	Сергеевна	ж	2006-09-17
NULL	NULL	NULL	NULL	NULL	NULL

Рисунок 5.3.6 - Таблица УЧАСТНИК после применения условия

Для подсчета количества мероприятий, проводимых в очном и онлайн форматах, используется встроенная агрегатная функция SUM. С её помощью производится суммирование значений, соответствующих каждому типу формата: очным (1) и онлайн (2). Такой подход позволяет точно определить количество мероприятий для каждого формата, предоставляя удобный и наглядный способ анализа данных.

Результат подсчёта выводится в виде отдельной таблицы-запроса, что обеспечивает структурированное отображение информации и упрощает дальнейшую работу с ней.

id_Мероприятия	Название	Место_проведения	Начало	Конец	Формат_id_Формата	Дисциплина_id_Дисциплины
1	Турнир по футболу	Спортзал	2024-04-01 10:00:00	2024-04-01 15:00:00	1	1
2	Баскетбольный чемпионат	Спортзал	2023-04-02 12:00:00	2023-04-02 18:00:00	1	2
3	Волейбольный турнир	Спортзал	2023-05-10 09:00:00	2023-05-10 14:00:00	1	3
4	Соревнования по лёгкой атлетике	Стадион	2023-06-15 08:00:00	2023-06-15 12:00:00	1	4
5	Теннисный матч	Корт	2023-07-01 10:00:00	2023-07-01 13:00:00	1	5
6	Турнир по настольному теннису	Игровая зона	2023-07-15 14:00:00	2023-07-15 17:00:00	1	6
7	Шахматный турнир	Онлайн платформа	2023-08-05 09:00:00	2023-08-05 13:00:00	2	7
8	Бадминтонный чемпионат	Спортзал	2023-09-10 15:00:00	2023-09-10 19:00:00	1	8
9	Киберспортивный турнир	Онлайн платформа	2023-10-20 18:00:00	2023-10-20 22:00:00	2	9
10	Забег на длинные дистанции	Парк	2023-11-01 07:00:00	2023-11-01 11:00:00	1	10

Рисунок 5.3.7 - Таблица МЕРОПРИЯТИЕ

Код выглядит следующим образом:

```
476 •  SELECT `Формат_id_Формата`, COUNT(*) AS `Количество_мероприятий`
477    FROM `мероприятие`
478    WHERE `Формат_id_Формата` BETWEEN 1 AND 2
479    GROUP BY `Формат_id_Формата`;
```

Рисунок 5.3.8 - Код для подсчета количества студентов

Получается следующий результат выполнения встроенной функции:

Формат_id_Формата	Количесвто_мероприятий
1	8
2	2

Рисунок 5.3.9 - Результат выполнения встроенной функции

id_Формата	Название	Описание
1	Очный	Формат, в котором участники соревнуются вживую в кампусе университета, присутствуя на мероприятии лично.
2	Онлайн	Формат, в котором участники соревнуются удаленно через онлайн-платформы, используя интернет.

Рисунок 5.3.10 - Таблица ФОРМАТ в результате выполнения встроенной функции

Для объединения таблиц КОМАНДА, УЧАСТНИК и КОМАНДА_УЧАСТНИК с целью отобразить полный список участников и команд используется оператор INNER JOIN. Этот тип объединения позволяет отобразить пересекающиеся записи из таблицы КОМАНДА и УЧАСТНИК.

INNER JOIN формирует итоговый результат, где каждой команде сопоставляются данные об участниках. Если связь между таблицами отсутствует, выводится только информация о команде.

Ключевым условием объединения является: команда.id_Команды = команда_участник. Команда.id_Команды и комнада_участник.Участник.id_Участника = участник.id_Участника.

Таблицы связываются через их внешние ключи в ассоциативной таблице, что позволяет корректно отобразить данные. Такой подход обеспечивает полное представление информации, включая записи, которые пока не связаны

id_Участника	Имя	Фамилия	Отчество	Пол	Дата_рождения
1	Иван	Иванов	Иванович	м	2010-01-01
2	Петр	Петров	Петрович	м	2004-07-22
3	Мария	Смирнова	Игоревна	ж	2003-03-15
4	Елена	Кузнецова	Анатольевна	ж	2002-09-03
5	Алексей	Захаров	Вячеславович	м	2005-01-25
6	Анна	Федорова	Александровна	ж	2005-12-09
7	Сергей	Михайлов	Александрович	м	2003-05-14
8	Ольга	Васильева	Сергеевна	ж	2004-11-23
9	Дмитрий	Ковалёв	Иванович	м	2006-02-18
10	Татьяна	Ильина	Петровна	ж	2003-08-09
11	Николай	Попов	Михайлович	м	2005-06-30
12	Виктория	Громова	Алексеевна	ж	2002-12-15
13	Евгений	Соколов	Владимирович	м	2004-03-21
14	Елена	Орлова	Сергеевна	ж	2006-09-17
NULL	NULL	NULL	NULL	NULL	NULL

Рисунок 5.3.11 - Таблица УЧАСТНИК

id_Команды	Название	Дата_основания	Тренер
1	ФПИН_и_смешарики	2022-03-15	Иванов Иван
2	СУИИР	2022-07-20	Петров Петр
NULL	NULL	NULL	NULL

Рисунок 5.3.12 - Таблица КОМАНДА

id_Команда_Участник	Команда_id_Команды	Участник_id_Участника	Дата_вступления	Дата_ухода	Причина_ухода	Достижения
1	1	1	2022-02-01	NULL	NULL	Победа в локальном турнире
2	1	2	2022-03-15	2023-05-20	Личные обстоятельства	Участие в межвузовском соревновании
3	1	3	2022-05-20	NULL	NULL	Призёр факультетских соревнований
4	1	4	2022-06-10	NULL	NULL	Лучший игрок месяца
5	1	5	2022-07-05	NULL	NULL	Участие в университетской олимпиаде
6	1	6	2022-09-01	NULL	NULL	Разработка стратегии команды
7	1	7	2022-10-15	NULL	NULL	Участие в региональном турнире
8	2	8	2022-02-12	NULL	NULL	Лучший новичок команды
9	2	9	2022-04-01	NULL	NULL	Серебряный призёр факультетского чемпионата
10	2	10	2022-05-25	2024-01-01	Окончание учёбы	Победа в межфакультетском соревновании
11	2	11	2022-07-15	NULL	NULL	Организация внутреннего чемпионата
12	2	12	2022-08-20	NULL	NULL	NULL
13	2	13	2022-10-05	NULL	NULL	Лидер по количеству забитых очков
14	2	14	2022-11-10	NULL	NULL	Участие в международном соревновании

Рисунок 5.3.13 - Таблица КОМАНДА_УЧАСТНИК

```
481 •  SELECT
482        `команда`.`id_Команды` AS `ID_Команды`,
483        `команда`.`Название` AS `Название_Команды`,
484        `участник`.`id_Участника` AS `ID_Участника`,
485        `участник`.`Имя` AS `Имя_Участника`,
486        `участник`.`Фамилия` AS `Фамилия_Участника`
487    FROM
488        `команда`
489    INNER JOIN
490        `команда_участник`
491    ON
492        `команда`.`id_Команды` = `команда_участник`.`Команда_id_Команды`
493    INNER JOIN
494        `участник`
495    ON
496        `команда_участник`.`Участник_id_Участника` = `участник`.`id_Участника`;
```

Рисунок 5.3.14 - Код для объединения таблиц

В результате все участники будут отображены, включая те, которые пока не имеют связанных команд. Это поможет выявить направления, требующие дополнительного внимания для набора участников или дальнейшего развития.

ID_Команды	Название_Команды	Комментарий_о_команде	ID_Участника	Имя_Участника	Фамилия_Участника
1	ФПИН_и_смешарики	Команда факультета ФПИН (бывш. ФИКТ)	1	Иван	Иванов
1	ФПИН_и_смешарики	Команда факультета ФПИН (бывш. ФИКТ)	2	Петр	Петров
1	ФПИН_и_смешарики	Команда факультета ФПИН (бывш. ФИКТ)	3	Мария	Смирнова
1	ФПИН_и_смешарики	Команда факультета ФПИН (бывш. ФИКТ)	4	Елена	Кузнецова
1	ФПИН_и_смешарики	Команда факультета ФПИН (бывш. ФИКТ)	5	Алексей	Захаров
1	ФПИН_и_смешарики	Команда факультета ФПИН (бывш. ФИКТ)	6	Анна	Федорова
1	ФПИН_и_смешарики	Команда факультета ФПИН (бывш. ФИКТ)	7	Сергей	Михайлов
2	СУИИИР	Команда факультета ФСУИР	8	Ольга	Васильева
2	СУИИИР	Команда факультета ФСУИР	9	Дмитрий	Ковалёв
2	СУИИИР	Команда факультета ФСУИР	10	Татьяна	Ильина
2	СУИИИР	Команда факультета ФСУИР	11	Николай	Попов
2	СУИИИР	Команда факультета ФСУИР	12	Виктория	Громова
2	СУИИИР	Команда факультета ФСУИР	13	Евгений	Соколов
2	СУИИИР	Команда факультета ФСУИР	14	Елена	Орлова

Рисунок 5.3.15 - Результат

Заключение

В ходе выполнения лабораторной работы был разработан и усовершенствован сценарий базы данных для автоматизации управления спортивными мероприятиями. Определены ключевые сущности системы и их атрибуты на основе интервью с менеджером университета. Проведена нормализация до третьей нормальной формы и последующая денормализация данных для повышения производительности запросов. В рамках денормализации добавлены атрибуты, что позволило упростить обработку данных и ускорить доступ к ключевой информации.

На этапе физического проектирования логическая модель была преобразована в физическую с использованием SQL-скриптов, созданных для каждой сущности. Это включало создание таблиц, индексов, первичных и внешних ключей. Были проработаны основные SQL-запросы с использованием операторов DDL, DML, DQL и TCL, позволяющие выполнять операции добавления, обновления, удаления и анализа данных.

В результате работы удалось автоматизировать ключевые процессы: управление мероприятиями, учет заявок, ведение состава команд, фиксацию результатов. Также были проработаны запросы для анализа данных, включая агрегатные функции и объединение таблиц. Лабораторная работа позволила углубить знания в области проектирования баз данных и применения языка SQL на практике, а также закрепить навыки работы с логическими и физическими моделями данных.

Список использованных источников

1. Курс «Проектирование и реализация баз данных» на OpenEDU: URL: https://apps.openedu.ru/learning/course/coursev1:ITMOUniversity+DBDESIMP+fall_2024_ITMO/home (Дата обращения: 13.10.2024)
2. Инструменты MySQL WorkBench: URL: https://www.mysql.com/products/workbench (Дата обращения: 18.11.2024)
3. Создание базы данных MySQL WorkBench: URL: clck.ru/3F9KcS (Дата обращения: 25.11.2024)
4. Нормальные формы БД: URL: https://youtu.be/zqQxWdTpSIA?si=PyhLMki76InR7U3n (Дата обращения: 27.11.2024)