

Министерство образования Республики Беларусь

Учреждение образования
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей

Кафедра электронных вычислительных машин

ОТЧЕТ
по лабораторной работе № 1
на тему
РАЗРАБОТКА ТРЕБОВАНИЙ К СХЕМЕ ДАННЫХ И
ПОЛЬЗОВАТЕЛЬСКОМУ ИНТЕРФЕЙСУ ПРИКЛАДНОЙ ПРОГРАММЫ.
ПРАКТИЧЕСКОЕ ЗНАКОМСТВО С ИНТЕРФЕЙСОМ POSTGRESQL.
ВАРИАНТ №11 (ШКОЛА)

Студент:

А.Н. Климович

Преподаватель:

Ю.Ю. Желтко

МИНСК 2024

1 ЦЕЛЬ РАБОТЫ

1. Научиться разрабатывать схему базы данных (БД) и требования к пользовательскому интерфейсу (UI).

2. Ознакомиться с основными командами и интерфейсом работы в PostgreSQL.

2 ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

2.1 Разработка схемы данных

1. Спроектировать схему базы данных, включающую основную таблицу, содержащую данные, достаточные для работы пользовательского интерфейса.

2. Добавить 3-4 справочных таблиц (LUT) для заполнения и верификации полей основной таблицы.

3. Не менее двух справочных таблиц должны быть связаны с основной таблицей отношением «многие ко многим».

2.2 Разработка пользовательского интерфейса

1. Описать, как пользователи будут взаимодействовать с приложением для работы с базой данных.

2. Интерфейс должен поддерживать следующие функции: добавление, изменение и удаление данных в интерактивном и пакетном режимах.

Интерактивный режим – это режим, в котором пользователь работает с программой в реальном времени, вводя команды вручную и немедленно получая на них отклик.

Пакетный режим – режим, в котором команды или запросы собираются в одном файле или пакете и выполняются все вместе, одним запуском, без участия пользователя в реальном времени.

2.3 Технические требования

1. Разработать спецификацию (техническое задание) для базы данных и пользовательского интерфейса:

- описание структуры базы данных (какие таблицы нужны, какие поля они содержат);

- описание связей между таблицами (какие поля связаны и как);

- описание операций, доступных пользователю (что можно добавлять, изменять, удалять);

- описание взаимодействия пользователя с интерфейсом программы.

2.4 Работа с PostgreSQL

1. Практически освоить команды работы с PostgreSQL для выполнения операций добавления, изменения и удаления данных в базе данных (БД).

3 ВЫПОЛНЕНИЕ РАБОТЫ

3.1 Разработка схемы данных

В таблице 3.1 представлено описание сущностей, выделенных в модели «Школа».

Таблица 3.1 – Сущности

Имя сущности	Имя таблицы, отображающей сущность	Названия атрибутов	Названия полей таблицы
Ученик	student	Имя	first_name
		Фамилия	last_name
		Пол	gender_type
Учитель	teacher	Имя	first_name
		Фамилия	last_name
		Пол	gender_type
		Возраст	age
		Номер телефона	phone_no
Класс	class	Название класса	class_name
Предмет	subject	Название предмета	subject_name

В таблице 3.2 представлено описание связей для сущностей в модели «Школа».

Таблица 3.2 – Связи

Связь	Промежуточная таблица	Описание
Предметы учащегося	student_subject	Описывает предметы, которые изучает ученик
Учитель для класса	class_teacher	Описывает учителей, которые ведут предметы в определенных классах
Класс ученика	—	Описывает принадлежность ученика какому-либо классу
Предмет учителя	—	Описывает предмет как специализацию учителя
Классный руководитель	—	Описывает учителя для каждого класса

По описанным сущностям и связям была разработана реляционная модель «Школа», представленная на рисунке 3.1. Для отображения связей использовалась нотация «Crow's Foot».

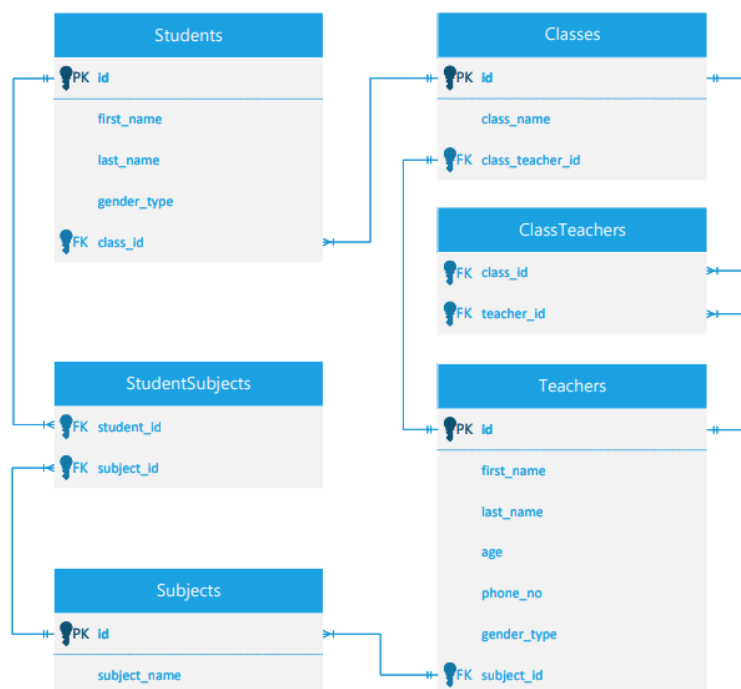


Рисунок 3.1 – Реляционная модель «Школа»

3.2. Аналогично была построена модель данных, изображенная на рисунке

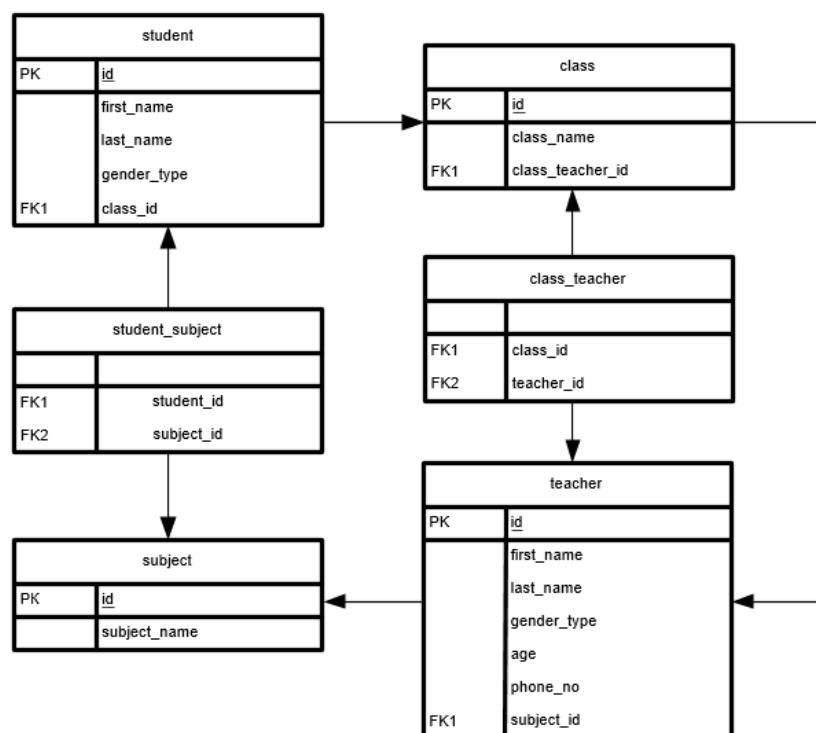


Рисунок 3.2 – Модель данных «Школа»

3.2 Разработка пользовательского интерфейса

3.2.1 Техническое требования

Для работы с базой данных «Школа» был разработан пользовательский интерфейс (UI) с возможностью взаимодействия исключительно через клавиатуру. Это подразумевает минимизацию необходимости использования графических элементов и внедрение удобной навигации через горячие клавиши и текстовые команды.

Интерфейс должен поддерживать основные функции взаимодействия с базой данных:

- добавление данных;
- просмотр данных;
- изменение данных;
- удаление данных.

Пользователи будут взаимодействовать с программой исключительно с помощью клавиатуры, что требует внедрения системы горячих клавиш и команд. Пример основных взаимодействий:

1. «Tab» – переключение между элементами управления (например, между формами ввода).
2. «Enter» – подтверждение выбора или выполнение команды.
3. «Alt + буква» – открытие соответствующего раздела меню.
4. «Ctrl + буква» – выполнение специфических действий (например, «Ctrl + A» для добавления новой записи, «Ctrl + E» для редактирования, «Ctrl + D» для удаления).

Пример пользовательского интерфейса показан на рисунке 2.2.

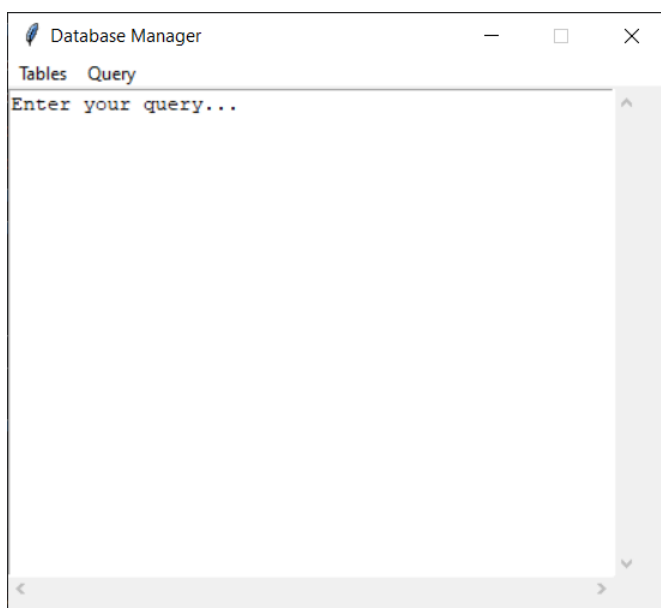


Рисунок 3.3 – Пример пользовательского интерфейса

В меню «Tables» можно выбрать таблицу, с которой дальше можно взаимодействовать (см. рисунок 3.4).

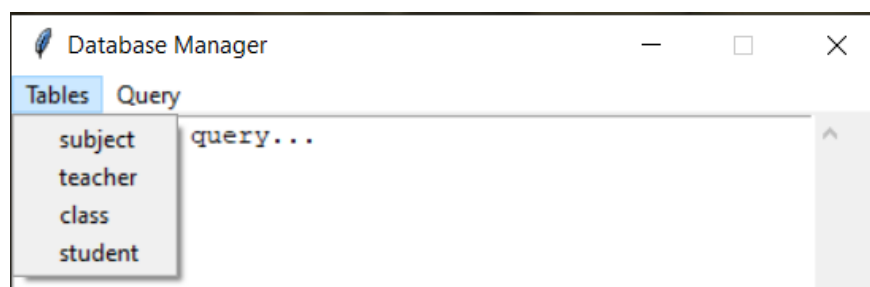


Рисунок 3.4 – Подменю «Tables»

В меню «Query» можно выполнить SQL-запрос либо из текстового редактора приложения, либо из локального файла (см. рисунок 3.5).



Рисунок 3.5 – Подменю «Query»

3.2.1 Интерактивный режим

В интерактивном режиме пользователь будет вводить команды вручную и немедленно получать отклик. Это позволяет администратору работать с базой данных в реальном времени. На рисунке 3.6 в красной рамке выделен текстовый редактор, где можно писать SQL-запросы.

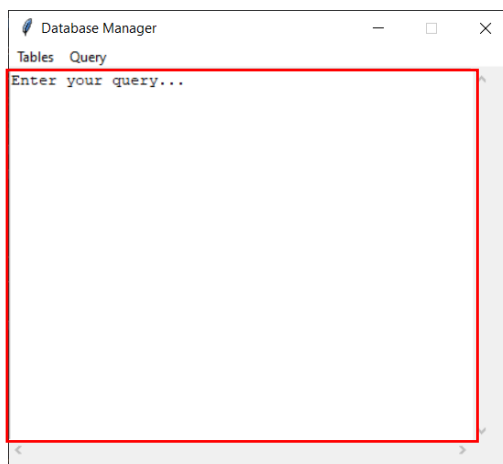
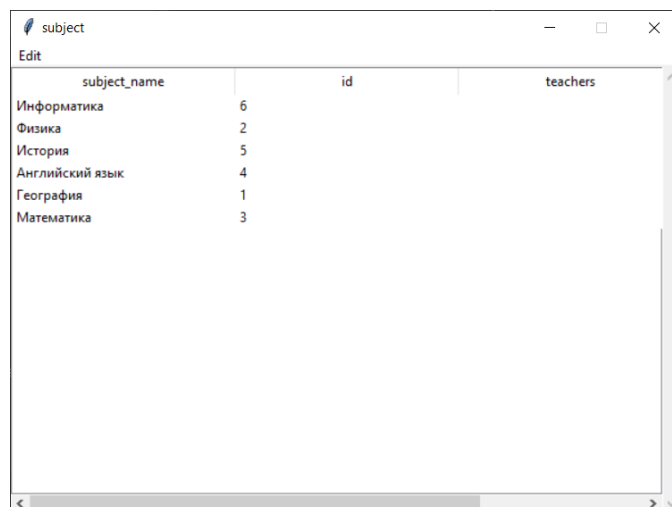


Рисунок 3.6 – Текстовый редактор для SQL-запросов для интерактивного режима

При выборе таблицы открывается новое окно для работы с выбранной таблицей. Например, на рисунке 3.7 показано окно, которое открывается после выбора таблицы «subject».

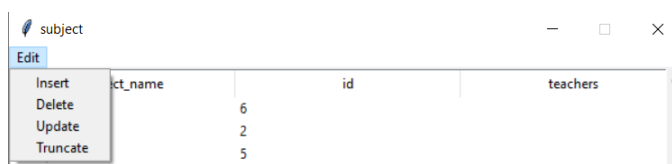


The screenshot shows a window titled 'subject' with a tab labeled 'Edit'. Inside, there is a table with three columns: 'subject_name', 'id', and 'teachers'. The 'teachers' column is currently empty. The data rows are as follows:

subject_name	id	teachers
Информатика	6	
Физика	2	
История	5	
Английский язык	4	
География	1	
Математика	3	

Рисунок 3.7 – Окно таблицы «subject»

В отрывшемся окне операции добавления, обновления и удаления можно выполнять через подменю «Edit» (рисунок 3.8).

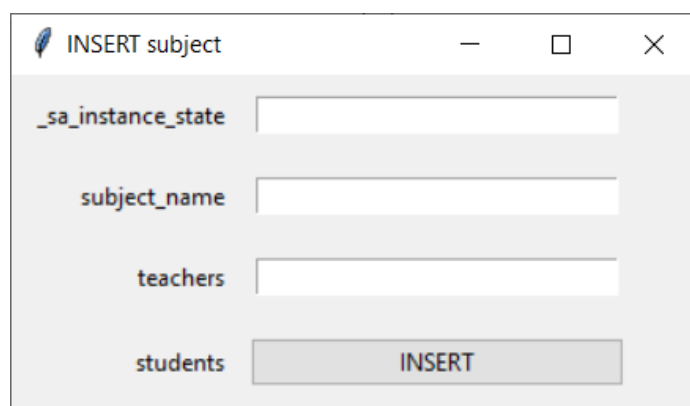


The screenshot shows the same 'subject' window, but with the 'Edit' menu open. The menu options are: Insert, Delete, Update, and Truncate. The table data is partially visible, showing the first three rows:

subject_name	id	teachers
	6	
	2	
	5	

Рисунок 3.8 – Подменю «Edit»

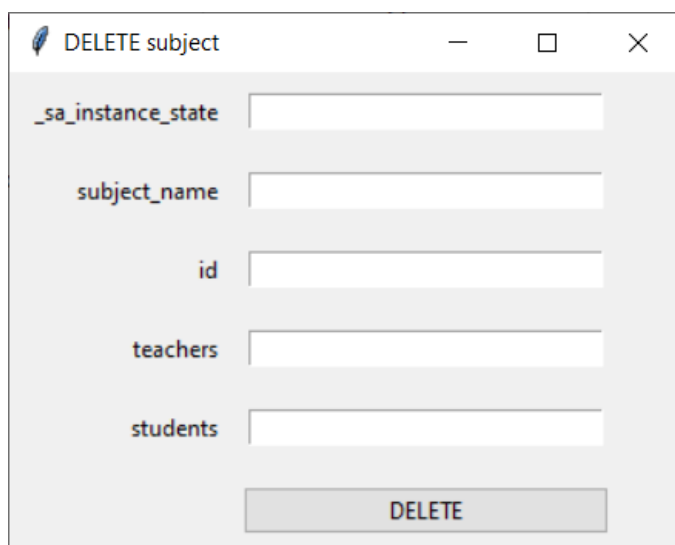
При нажатии на «Insert» создается окно для заполнения полей сущности, которую нужно добавить в таблицу (рисунок 3.9).



The screenshot shows a window titled 'INSERT subject'. It contains four input fields with labels: '_sa_instance_state', 'subject_name', 'teachers', and 'students'. The 'students' field is a button labeled 'INSERT'.

Рисунок 3.9 – Окно для добавления объекта в таблицу

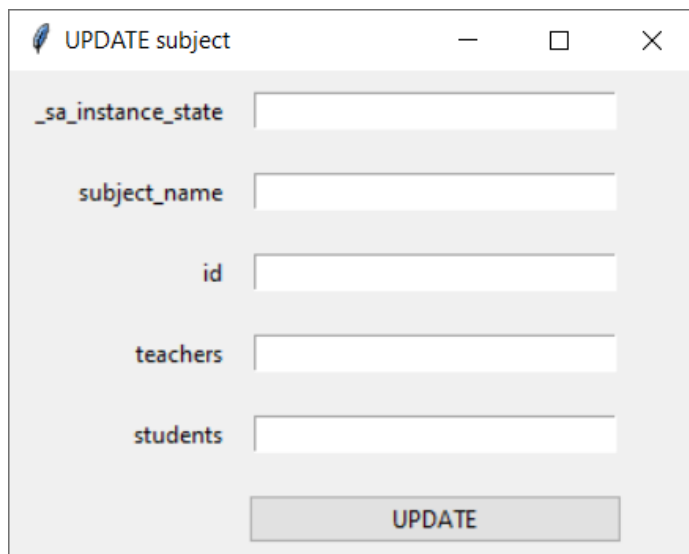
При нажатии на «Delete» создается окно для заполнения полей сущности, которую нужно удалить из таблицы (рисунок 3.10).



The image shows a dialog box titled "DELETE subject". It has a standard window frame with minimize, maximize, and close buttons. Inside, there are five text input fields arranged vertically, each with a label to its left: "_sa_instance_state", "subject_name", "id", "teachers", and "students". At the bottom of the dialog is a button labeled "DELETE".

Рисунок 3.10 – Окно для удаления объекта из таблицы

При нажатии на «Update» создается окно для заполнения полей сущности, которую нужно обновить в таблице (рисунок 3.11).



The image shows a dialog box titled "UPDATE subject". It has a standard window frame with minimize, maximize, and close buttons. Inside, there are five text input fields arranged vertically, each with a label to its left: "_sa_instance_state", "subject_name", "id", "teachers", and "students". At the bottom of the dialog is a button labeled "UPDATE".

Рисунок 3.11 – Окно для обновления объекта в таблице

При нажатии на «Truncate» таблица полностью очищается.

3.2.2 Пакетный режим

Пакетный режим позволяет выполнять набор команд за один запуск программы.

В приложении этот режим осуществляется через подменю «Query» кнопкой «Execute from file» или с помощью комбинации клавиш (рисунок 3.5).

Пакетный режим может использоваться для массового добавления, обновления, чтения или изменения данных.

3.3 Работа с PostgreSQL

3.3.1 Создание таблиц

Далее приведен скрипт для создания таблиц. Связи между таблицами создаются с помощью оператора «ALTER». В самом начале скрипта таблицы предварительно удаляются, если были ранее созданы. Сами таблицы создаются с помощью оператора «CREATE». Также для ускорения работы с данными БД некоторые таблицы были проиндексированы.

```
-- Удаление таблиц, если они существуют
DROP TABLE IF EXISTS public.student_subject CASCADE;
DROP TABLE IF EXISTS public.student CASCADE;
DROP TABLE IF EXISTS public.teacher CASCADE;
DROP TABLE IF EXISTS public.subject CASCADE;
DROP TABLE IF EXISTS public.class CASCADE;

-- Создание таблицы subject
CREATE TABLE IF NOT EXISTS public.subject (
    id SERIAL PRIMARY KEY,
    subject_name VARCHAR(100) NOT NULL
);

-- Создание таблицы student
CREATE TABLE IF NOT EXISTS public.student (
    id SERIAL PRIMARY KEY,
    first_name VARCHAR(100) NOT NULL,
    last_name VARCHAR(100) NOT NULL,
    gender_type VARCHAR(20) NOT NULL CHECK (gender_type IN
('MALE', 'FEMALE', 'OTHER')),
    class_id INTEGER NOT NULL
);

-- Создание таблицы student_subject для отношения многие ко
многим между student и subject
CREATE TABLE IF NOT EXISTS public.student_subject (
    student_id INTEGER NOT NULL,
    subject_id INTEGER NOT NULL,
    PRIMARY KEY (student_id, subject_id)
);

-- Создание таблицы class
CREATE TABLE IF NOT EXISTS public.class (
    id SERIAL PRIMARY KEY,
```

```

        class_name VARCHAR(100) NOT NULL,
        class_teacher_id INTEGER
    );

-- Создание таблицы teacher
CREATE TABLE IF NOT EXISTS public.teacher (
    id SERIAL PRIMARY KEY,
    first_name VARCHAR(100) NOT NULL,
    last_name VARCHAR(100) NOT NULL,
    age INTEGER NOT NULL CHECK (age >= 18),
    phone_no VARCHAR(15) NOT NULL,
    gender_type VARCHAR(20) NOT NULL CHECK (gender_type IN
('MALE', 'FEMALE', 'OTHER')),
    subject_id INTEGER NOT NULL
);

-- Добавление индекса для оптимизации запросов
CREATE INDEX IF NOT EXISTS idx_student_class ON
public.student(class_id);
CREATE INDEX IF NOT EXISTS idx_teacher_subject ON
public.teacher(subject_id);

-- ALTER TABLE для добавления всех внешних ключей

-- Добавление внешнего ключа для связи студента с классом
ALTER TABLE public.student
    ADD CONSTRAINT fk_class FOREIGN KEY (class_id)
    REFERENCES public.class(id)
    ON UPDATE CASCADE
    ON DELETE CASCADE;

-- Добавление внешнего ключа для связи учителя с предметом
ALTER TABLE public.teacher
    ADD CONSTRAINT fk_subject FOREIGN KEY (subject_id)
    REFERENCES public.subject(id)
    ON UPDATE CASCADE
    ON DELETE CASCADE;

-- Добавление внешнего ключа для связи классного руководителя
с классом
ALTER TABLE public.class
    ADD CONSTRAINT fk_teacher FOREIGN KEY (class_teacher_id)
    REFERENCES public.teacher(id)
    ON UPDATE CASCADE
    ON DELETE SET NULL;

-- Добавление внешнего ключа для связи таблицы
student_subject (многие ко многим) с таблицей student
ALTER TABLE public.student_subject
    ADD CONSTRAINT fk_student FOREIGN KEY (student_id)
    REFERENCES public.student(id)
    ON DELETE CASCADE;

```

```

--      Добавление внешнего ключа для связи таблицы
student_subject (многие ко многим) с таблицей subject
ALTER TABLE public.student_subject
    ADD CONSTRAINT fk_subject FOREIGN KEY (subject_id)
    REFERENCES public.subject(id)
    ON DELETE CASCADE;

```

3.3.2 Добавление данных

Далее приведен скрипт для добавления данных в БД.

```

INSERT INTO public.subject (subject_name) VALUES
('Mathematics'),
('Physics'),
('Chemistry'),
('Biology'),
('History'),
('Geography'),
('English'),
('French'),
('Spanish'),
('Computer Science'),
('Art'),
('Music'),
('Physical Education'),
('Economics'),
('Psychology'),
('Philosophy'),
('Political Science'),
('Sociology'),
('Literature'),
('Drama'),
('Business Studies'),
('Environmental Science'),
('Astronomy'),
('Law'),
('Media Studies'),
('Engineering'),
('Architecture'),
('Design'),
('Statistics'),
('Anthropology');

INSERT INTO public.class (class_name, class_teacher_id)
VALUES
('1A', NULL),
('1B', NULL),
('2A', NULL),
('2B', NULL),
('3A', NULL),

```

```

('3B', NULL),
('4A', NULL),
('4B', NULL),
('5A', NULL),
('5B', NULL),
('6A', NULL),
('6B', NULL),
('7A', NULL),
('7B', NULL),
('8A', NULL),
('8B', NULL),
('9A', NULL),
('9B', NULL),
('10A', NULL),
('10B', NULL),
('11A', NULL),
('11B', NULL),
('12A', NULL),
('12B', NULL),
('13A', NULL),
('13B', NULL),
('14A', NULL),
('14B', NULL),
('15A', NULL),
('15B', NULL);

```

```

INSERT INTO public.teacher (first_name, last_name, age,
phone_no, gender_type, subject_id) VALUES
('John', 'Doe', 40, '123-456-7890', 'MALE', 1),
('Jane', 'Smith', 35, '123-456-7891', 'FEMALE', 2),
('Alice', 'Johnson', 29, '123-456-7892', 'FEMALE', 3),
('Bob', 'Brown', 50, '123-456-7893', 'MALE', 4),
('Charlie', 'Davis', 45, '123-456-7894', 'MALE', 5),
('Emily', 'Miller', 33, '123-456-7895', 'FEMALE', 6),
('Grace', 'Wilson', 38, '123-456-7896', 'FEMALE', 7),
('Henry', 'Moore', 42, '123-456-7897', 'MALE', 8),
('Ivy', 'Taylor', 30, '123-456-7898', 'FEMALE', 9),
('James', 'Anderson', 36, '123-456-7899', 'MALE', 10),
('Katherine', 'Thomas', 41, '123-456-7800', 'FEMALE', 11),
('Liam', 'Jackson', 39, '123-456-7801', 'MALE', 12),
('Mia', 'White', 31, '123-456-7802', 'FEMALE', 13),
('Noah', 'Harris', 43, '123-456-7803', 'MALE', 14),
('Olivia', 'Martin', 37, '123-456-7804', 'FEMALE', 15),
('Paul', 'Garcia', 44, '123-456-7805', 'MALE', 16),
('Quincy', 'Martinez', 46, '123-456-7806', 'MALE', 17),
('Rachel', 'Robinson', 28, '123-456-7807', 'FEMALE', 18),
('Sam', 'Clark', 49, '123-456-7808', 'MALE', 19),
('Tina', 'Rodriguez', 34, '123-456-7809', 'FEMALE', 20),
('Ulysses', 'Lewis', 32, '123-456-7810', 'MALE', 21),
('Victoria', 'Lee', 47, '123-456-7811', 'FEMALE', 22),
('Will', 'Walker', 35, '123-456-7812', 'MALE', 23),
('Xander', 'Hall', 29, '123-456-7813', 'MALE', 24),

```

```

('Yvonne', 'Allen', 48, '123-456-7814', 'FEMALE', 25),
('Zoe', 'Young', 27, '123-456-7815', 'FEMALE', 26),
('Arthur', 'King', 52, '123-456-7816', 'MALE', 27),
('Betty', 'Wright', 53, '123-456-7817', 'FEMALE', 28),
('Carl', 'Scott', 54, '123-456-7818', 'MALE', 29),
('Diana', 'Green', 55, '123-456-7819', 'FEMALE', 30);

INSERT INTO public.student (first_name, last_name,
gender_type, class_id) VALUES
('Ethan', 'Reed', 'MALE', 1),
('Sophia', 'Bailey', 'FEMALE', 2),
('Jacob', 'Cook', 'MALE', 3),
('Ava', 'Bell', 'FEMALE', 4),
('Michael', 'Parker', 'MALE', 5),
('Isabella', 'Murphy', 'FEMALE', 6),
('William', 'Price', 'MALE', 7),
('Mia', 'Gray', 'FEMALE', 8),
('James', 'Ramirez', 'MALE', 9),
('Olivia', 'Torres', 'FEMALE', 10),
('Benjamin', 'Peterson', 'MALE', 11),
('Amelia', 'Cooper', 'FEMALE', 12),
('Lucas', 'Brooks', 'MALE', 13),
('Emma', 'Foster', 'FEMALE', 14),
('Henry', 'Sanders', 'MALE', 15),
('Emily', 'Jenkins', 'FEMALE', 16),
('Alexander', 'Perez', 'MALE', 17),
('Ella', 'Sullivan', 'FEMALE', 18),
('Daniel', 'James', 'MALE', 19),
('Avery', 'Barnes', 'FEMALE', 20),
('Jackson', 'Ross', 'MALE', 21),
('Harper', 'Ward', 'FEMALE', 22),
('Sebastian', 'Butler', 'MALE', 23),
('Grace', 'Russell', 'FEMALE', 24),
('David', 'Stewart', 'MALE', 25),
('Chloe', 'Young', 'FEMALE', 26),
('Joseph', 'Hughes', 'MALE', 27),
('Scarlett', 'Bryant', 'FEMALE', 28),
('Matthew', 'Williams', 'MALE', 29),
('Abigail', 'Hernandez', 'FEMALE', 30);

INSERT INTO public.student_subject (student_id, subject_id)
VALUES
(1, 1), (1, 2), (1, 3),
(2, 4), (2, 5), (2, 6),
(3, 7), (3, 8), (3, 9),
(4, 10), (4, 11), (4, 12),
(5, 13), (5, 14), (5, 15),
(6, 16), (6, 17), (6, 18),
(7, 19), (7, 20), (7, 21),
(8, 22), (8, 23), (8, 24),
(9, 25), (9, 26), (9, 27),
(10, 28), (10, 29), (10, 30);

```

3.3.3 Выполнение запросов на выборку

Пример 1. Получить всех студентов и их классы.

```
SELECT
    student.first_name AS student_first_name,
    student.last_name AS student_last_name,
    public.class.class_name
FROM
    public.student
JOIN
    public.class      ON      public.student.class_id      =
public.class.id;
```

Результат выполнения скрипта из примера 1 показан на рисунке 3.12.

Пример 2. Получить всех учителей и их предметы

```
SELECT
    teacher.first_name AS teacher_first_name,
    teacher.last_name AS teacher_last_name,
    subject.subject_name
FROM
    public.teacher
JOIN
    public.subject ON teacher.subject_id = subject.id;
```

Результат выполнения скрипта из примера 2 показан на рисунке 3.13.

Пример 3. Выбрать всех студентов, которые изучают определенный предмет (например, «Mathematics»)

```
SELECT
    student.first_name AS student_first_name,
    student.last_name AS student_last_name
FROM
    public.student
JOIN
    public.student_subject      ON      student.student_id      =
student_subject.student_id
JOIN
    public.subject      ON      student_subject.subject_id      =
subject.subject_id
WHERE
    subject.subject_name = 'Mathematics';
```

Результат выполнения скрипта из примера 3 показан на рисунке 3.14.

Query		Query History	
1	SELECT		
2	student.first_name AS student_first_name,		
3	student.last_name AS student_last_name,		
4	public.class.class_name		
5	FROM		
6	public.student		
7	JOIN		
8	public.class ON public.student.class_id = public.class.id;		
Data Output		Messages	Notifications
	student_first_name character varying (100)	student_last_name character varying (100)	class_name character varying (100)
1	Ethan	Reed	1A
2	Sophia	Bailey	1B
3	Jacob	Cook	2A
4	Ava	Bell	2B
5	Michael	Parker	3A
6	Isabella	Murphy	3B
7	William	Price	4A
8	Mia	Gray	4B
9	James	Ramirez	5A
10	Olivia	Torres	5B
11	Benjamin	Peterson	6A
12	Amelia	Cooper	6B
13	Lucas	Brooks	7A

Рисунок 3.12 – Результаты выполнения SQL-скрипта из примера 1

Query		Query History	
1	SELECT		
2	teacher.first_name AS teacher_first_name,		
3	teacher.last_name AS teacher_last_name,		
4	subject.subject_name		
5	FROM		
6	public.teacher		
7	JOIN		
8	public.subject ON teacher.subject_id = subject.id;		
Data Output		Messages	Notifications
	teacher_first_name character varying (100)	teacher_last_name character varying (100)	subject_name character varying (100)
1	John	Doe	Mathematics
2	Jane	Smith	Physics
3	Alice	Johnson	Chemistry
4	Bob	Brown	Biology
5	Charlie	Davis	History
6	Emily	Miller	Geography
7	Grace	Wilson	English
8	Henry	Moore	French
9	Ivy	Taylor	Spanish
10	James	Anderson	Computer Science
11	Katherine	Thomas	Art
12	Liam	Jackson	Music
13	Mia	White	Physical Education

Рисунок 3.13 – Результаты выполнения SQL-скрипта из примера 2

```
Query      Query History
1  SELECT
2      student.first_name AS student_first_name,
3      student.last_name AS student_last_name
4  FROM
5      public.student
6  JOIN
7      public.student_subject ON student.id = student_subject.student_id
8  JOIN
9      public.subject ON student_subject.subject_id = subject.id
10 WHERE |
11      subject.subject_name = 'Mathematics';
```

Data Output Messages Notifications

	student_first_name character varying (100) 🔒	student_last_name character varying (100) 🔒
1	Ethan	Reed

Рисунок 3.14 – Результат выполнения скрипта из примера 3

4 ВЫВОД

В рамках данной лабораторной работы была спроектирована схема данных и определены требования к пользовательскому интерфейсу.

В процессе разработки схемы были созданы сущности, такие как «subject», «student», «class» и «teacher», а также реализованы связи между таблицами, включая взаимосвязи типа «многие ко многим».

Кроме того, было составлено техническое задание на создание интерфейса, поддерживающего как интерактивный, так и пакетный режимы работы. Пользователи могут добавлять, редактировать и удалять данные, взаимодействуя с программой как с помощью клавиатуры, так и мыши. Это обеспечивает более быструю и удобную работу для опытных пользователей. Также была предложена система горячих клавиш для быстрого выполнения ключевых операций и навигации по разделам программы.

Во время работы были изучены основные принципы использования «PostgreSQL», что помогло закрепить навыки создания, изменения таблиц и выполнения основных операций с данными в реальной базе данных. Практическое применение команд «PostgreSQL» способствовало углублению знаний о взаимодействии с реляционными базами данных и их структурированием. Также были определены связи между различными классами и их мощности.