Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение

высшего образования

ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)

Кафедра компьютерных систем в управлении и проектировании (КСУП)

ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ POSTGRESQL

Отчет по лабораторной работе по дисциплине «Основы разработки баз данных»

Студент гр. 573-3

\_\_\_\_\_\_\_\_\_\_\_\_ Е. А. Макаров

дата

\_\_\_\_\_\_\_\_\_\_\_\_

подпись

Руководитель:

Преподаватель:

\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_ Р. О. Остапенко

подпись

оценка

\_\_\_\_\_\_\_\_\_\_\_\_\_

дата

Томск, 2025

**Введение**

В современном мире базы данных играют ключевую роль в системах хранения и обработки информации. Они используются в самых разных областях — от небольших веб-приложений до крупных корпоративных систем. Эффективная организация данных позволяет оптимизировать работу программных продуктов, обеспечить целостность и безопасность информации, а также упростить её поиск и анализ.

Цель данной лабораторной работы — познакомиться с основными принципами проектирования реляционных баз данных, освоить синтаксис языка структурированных запросов SQL и научиться создавать таблицы, определять связи между ними, а также накладывать ограничения для обеспечения корректности данных. В процессе выполнения работы будет спроектирована и реализована структура базы данных на языке SQL в среде PostgreSQL, отражающая реальные сущности и их взаимосвязи.

**Задачи:**

Реализовать БД согласно требованиям:

1. БД должна быть реляционной;
2. БД должна быть нормализована по 3НФ (иди БКНФ);
3. БД должна состоять минимум из 3-4 связанных таблиц (на каждого студента); в таблицах БД должны быть наложены ограничения на поля таблиц (по усмотрению разработчиков решается, на какие поля будут наложены ограничения). Должно быть не менее 3 ограничений на разные поля таблиц (без учета ограничений PRIMARY KEY, FOREIGN KEY). Ограничения проверки, значения по умолчанию обязательно;
4. должна быть обеспечена ссылочная целостность;
5. В БД должны быть обязательно поля разных типов данных, в том числе поля:
   1. Позволяющие хранить рисунки;
   2. Позволяющие хранить длинный текст;
   3. Значения, в которых должны соответствовать списку значений, например Пол мужской или женский;
   4. Значения логического типа.
6. **ОСНОВНАЯ ЧАСТЬ**

**1.1 Описание данных для реализации БД**

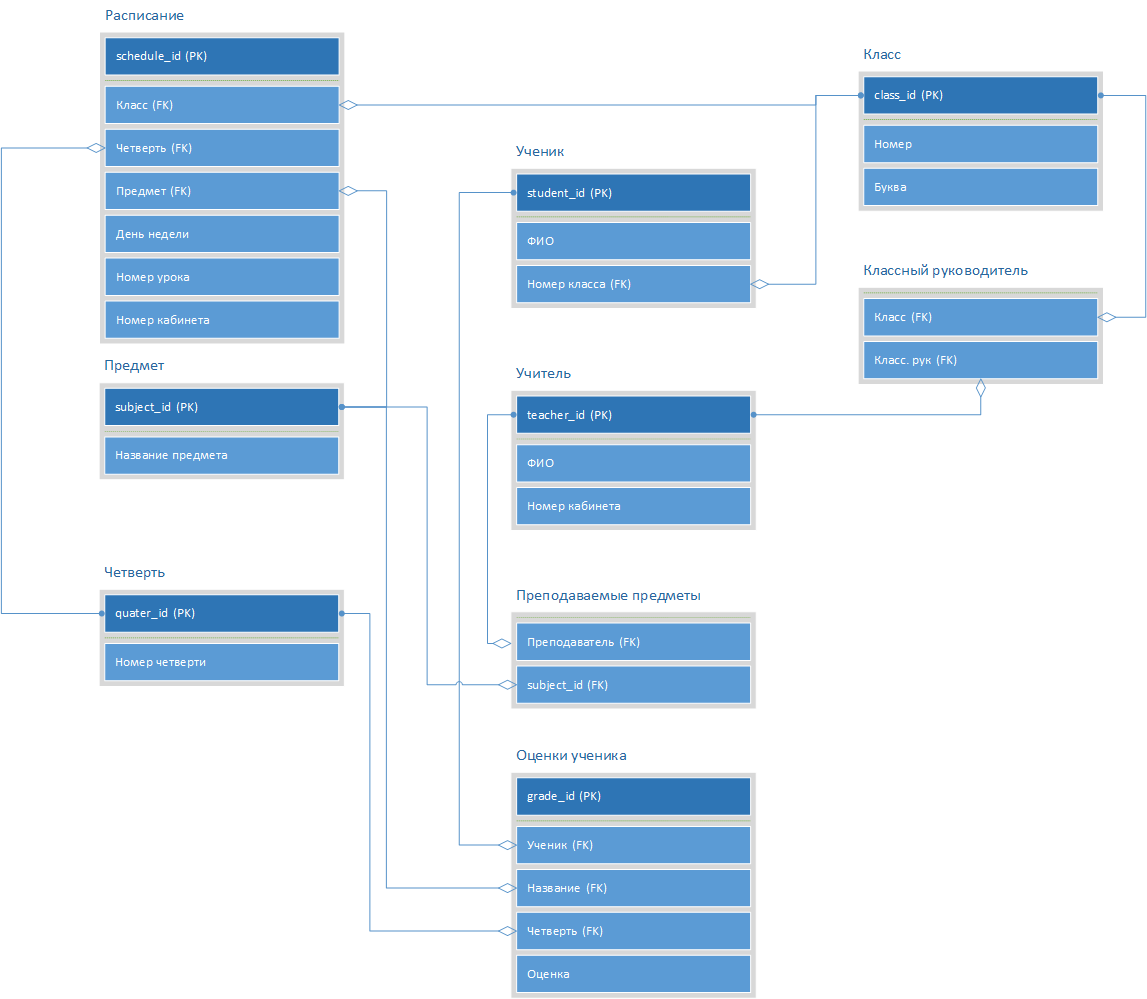
В ходе индивидуальных заданий была разработана реляционная модель данных (рисунок 1.1), которую необходимо реализовать в виде БД.

Рисунок 1.1 – Реляционная модель данных

Опишем сущности в виде таблиц, чтобы потом перенести их в БД. Также при необходимости добавим поля и ограничения, чтобы они соответствовали формулировке задания.

Для Ученика (таблица 1.1) было добавлено отдельное поле, которое будет хранить путь к файлу с его фотографией.

Таблица 1.1 – Описание Ученика

|  |  |  |  |
| --- | --- | --- | --- |
| Имя поля | Тип данных | Ограничения | Описание |
| student\_id | SERIAL | Первичный ключ. | Идентификатор ученика. |
| full\_name | VARCHAR(100) | Размер поля 100. Обязательное поле. | Полное ФИО. |
| class\_id | INT | Внешний ключ. Обязательное поле. | Класс ученика. |
| photo\_path | TEXT | Обязательное поле. Уникальное. | Путь к файлу с фото. |

Для Учителя (таблица 1.2) было добавлено поле для хранения пути к фотографии и его статуса (1 – работает, 0 – уволился). Это было сделано, чтобы уволившееся учителя не пропадали из БД со всеми своими предметами.

Таблица 1.2 – Описание Учителя

|  |  |  |  |
| --- | --- | --- | --- |
| Имя поля | Тип данных | Ограничения | Описание |
| teacher\_id | SERIAL | Первичный ключ. | Идентификатор учителя. |
| full\_name | VARCHAR(100) | Размер поля 100. Обязательное поле. | Полное ФИО. |
| cabinet\_number | INT | По умолчанию пусто. | Номер закреплённого кабинета. |
| is\_active | BOOL | Обязательное поле. True или False. | Статус работы учителя. |
| photo\_path | TEXT | Обязательное поле. Уникальное. | Путь к файлу с фото. |

Для Класса (таблица 1.3) были добавлены следующие характеристики.

Таблица 1.3 – Описание Класса

|  |  |  |  |
| --- | --- | --- | --- |
| Имя поля | Тип данных | Ограничения | Описание |
| class\_id | SERIAL | Первичный ключ. | Идентификатор класса. |
| number | INT | Обязательное поле. | Номер класса. |
| letter | VARCHAR(1) | Обязательное поле. | Буква класса. |

Для Расписания (таблица 1.4) было добавлено ограничение на день недели, чтобы он принадлежал промежутку от Понедельника до Субботы.

Таблица 1.4 – Описание Расписания

|  |  |  |  |
| --- | --- | --- | --- |
| Имя поля | Тип данных | Ограничения | Описание |
| schedule\_id | SERIAL | Первичный ключ. | Идентификатор расписания. |
| class\_id | INT | Внешний ключ. Обязательное поле. | Идентификатор класса для расписания. |
| quater\_id | INT | Внешний ключ. Обязательное поле. | Идентификатор четверти для расписания. |
| subject\_id | INT | Внешний ключ. Обязательное поле. | Идентификатор предмета для расписания. |
| day\_of\_week | TEXT | Обязательное поле. Принадлежит (Понедельник-Суббота) | День недели для расписания. |
| lesson\_number | INT | Обязательное поле. | Номер урока. |

Для Оценки (таблица 1.5) ученика было добавлено ограничение на значение оценки, чтобы оно принадлежало от 1 до 5.

Таблица 1.5 – Описание Оценки ученика

|  |  |  |  |
| --- | --- | --- | --- |
| Имя поля | Тип данных | Ограничения | Описание |
| grade\_id | SERIAL | Первичный ключ. | Идентификатор оценки. |
| student\_id | INT | Внешний ключ. Обязательное поле. | Идентификатор ученика для оценки. |
| subject\_id | INT | Внешний ключ. Обязательное поле. | Идентификатор предмета для оценки. |
| quater\_id | INT | Внешний ключ. Обязательное поле. | Идентификатор четверти для оценки. |
| grade\_value | INT | Обязательное поле. От 1 до 5 | Оценка. |

Для Четверти (таблица 1.6) был добавлен Академический год, чтобы однозначно определять её.

Таблица 1.6 – Описание Четверти

|  |  |  |  |
| --- | --- | --- | --- |
| Имя поля | Тип данных | Ограничения | Описание |
| quater\_id | SERIAL | Первичный ключ. | Идентификатор четверти. |
| quater\_number | INT | Обязательное поле. От 1 до 4. | Номер четверти. |
| academic\_year | VARCHAR(9) | Обязательное поле. | Академический год. Например «2024/2025». |

Для Предмета (таблица 1.7) были добавлены следующие характеристики.

Таблица 1.7 – Описание Предмета

|  |  |  |  |
| --- | --- | --- | --- |
| Имя поля | Тип данных | Ограничения | Описание |
| subject\_id | SERIAL | Первичный ключ. | Идентификатор предмета. |
| subject\_name | VARCHAR(100) | Обязательное поле. | Название предмета. |

Также необходимо описать таблицы Учитель-Предмет (таблица 1.8) и Класс-Учитель (таблица 1.9), для хранения связи М:М.в

Таблица 1.8 – Описание Учитель-Предмет

|  |  |  |  |
| --- | --- | --- | --- |
| Имя поля | Тип данных | Ограничения | Описание |
| teacher\_id | INT | Внешний ключ. Обязательное поле. | Идентификатор учителя. |
| subject\_id | INT | Внешний ключ. Обязательное поле. | Идентификатор предмета. |

Таблица 1.9 – Описание Класс-Учитель

|  |  |  |  |
| --- | --- | --- | --- |
| Имя поля | Тип данных | Ограничения | Описание |
| class\_id | INT | Внешний ключ. Обязательное поле. | Идентификатор класса. |
| teacher\_id | INT | Внешний ключ. Обязательное поле. | Идентификатор учителя. |

**1.2 Создание БД**

Для создания БД мы будем использовать систему управления БД (СУБД) PostgreSQL и pgAdmin. Она позволяет создавать таблицы как с помощью пользовательского кода, так и SQL кода. Для нашего случая будем использовать второе.

Начнём создание БД с её объявления (рисунок 1.2).

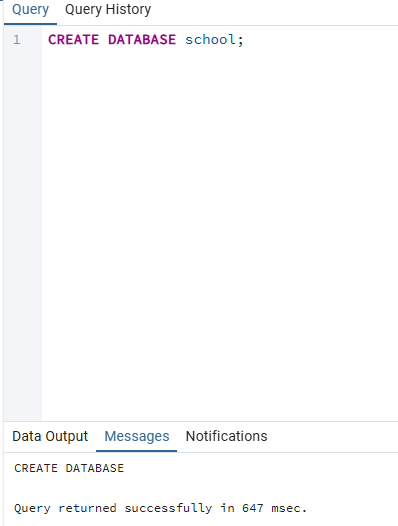


Рисунок 1.2 – Создание БД для школы

Как мы видим, наша БД отобразилась в интерфейсе pgAdmin (рисунок 1.3), и мы можем начинать работу с ней.

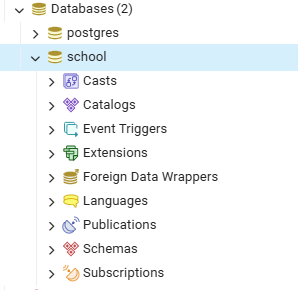


Рисунок 1.3 – Отображение БД в интерфейсе

Теперь напишем код для создания таблиц в нашей БД и связей между ними.

Листинг 1.1 – Создание таблиц

-- Таблица классов

CREATE TABLE class (

class\_id SERIAL PRIMARY KEY,

number INT NOT NULL,

letter VARCHAR(1) NOT NULL

);

-- Таблица четвертей

CREATE TABLE quater (

quater\_id SERIAL PRIMARY KEY,

quater\_number INT NOT NULL CHECK (quater\_number BETWEEN 1 AND 4),

academic\_year VARCHAR(9) NOT NULL

);

-- Таблица предметов

CREATE TABLE subject (

subject\_id SERIAL PRIMARY KEY,

subject\_name VARCHAR(100) NOT NULL

);

-- Таблица учителей

CREATE TABLE teacher (

teacher\_id SERIAL PRIMARY KEY,

full\_name VARCHAR(100) NOT NULL,

cabinet\_number INT,

is\_active BOOLEAN NOT NULL,

photo\_path TEXT UNIQUE NOT NULL

);

-- Таблица учеников

CREATE TABLE student (

student\_id SERIAL PRIMARY KEY,

full\_name VARCHAR(100) NOT NULL,

class\_id INT NOT NULL,

photo\_path TEXT UNIQUE NOT NULL,

FOREIGN KEY (class\_id) REFERENCES class(class\_id)

);

-- Таблица расписания

CREATE TABLE schedule (

schedule\_id SERIAL PRIMARY KEY,

class\_id INT NOT NULL,

quater\_id INT NOT NULL,

subject\_id INT NOT NULL,

day\_of\_week TEXT NOT NULL CHECK (

day\_of\_week IN ('Понедельник', 'Вторник', 'Среда', 'Четверг', 'Пятница', 'Суббота')

),

lesson\_number INT NOT NULL,

FOREIGN KEY (class\_id) REFERENCES class(class\_id),

FOREIGN KEY (quater\_id) REFERENCES quater(quater\_id),

FOREIGN KEY (subject\_id) REFERENCES subject(subject\_id)

);

-- Таблица оценок

CREATE TABLE grade (

grade\_id SERIAL PRIMARY KEY,

student\_id INT NOT NULL,

subject\_id INT NOT NULL,

quater\_id INT NOT NULL,

grade\_value INT NOT NULL CHECK (grade\_value BETWEEN 1 AND 5),

FOREIGN KEY (student\_id) REFERENCES student(student\_id),

FOREIGN KEY (subject\_id) REFERENCES subject(subject\_id),

FOREIGN KEY (quater\_id) REFERENCES quater(quater\_id)

);

-- Связь Учитель–Предмет

CREATE TABLE teacher\_subject (

teacher\_id INT NOT NULL,

subject\_id INT NOT NULL,

PRIMARY KEY (teacher\_id, subject\_id),

FOREIGN KEY (teacher\_id) REFERENCES teacher(teacher\_id),

FOREIGN KEY (subject\_id) REFERENCES subject(subject\_id)

);

-- Связь Класс–Учитель

CREATE TABLE class\_teacher (

class\_id INT NOT NULL,

teacher\_id INT NOT NULL,

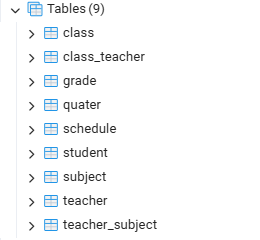
PRIMARY KEY (class\_id, teacher\_id),

FOREIGN KEY (class\_id) REFERENCES class(class\_id),

FOREIGN KEY (teacher\_id) REFERENCES teacher(teacher\_id)

);

Теперь мы можем увидеть отображение всех таблиц в интерфейсе (рисунок 1.4) и на ERD схеме данных (рисунок 1.5).



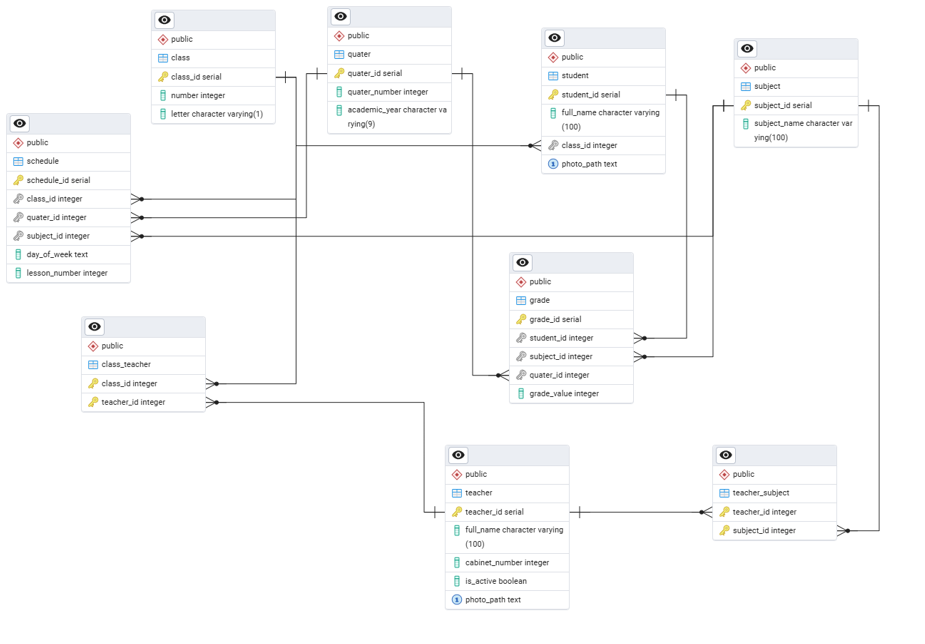
Рисунок 1.4 – Отображение таблиц в интерфейсе

Рисунок 1.5 – ERD схема данных

Теперь мы можем писать запросы (рисунки 1.6 - 1.14) к БД, в которую заранее записали данные.

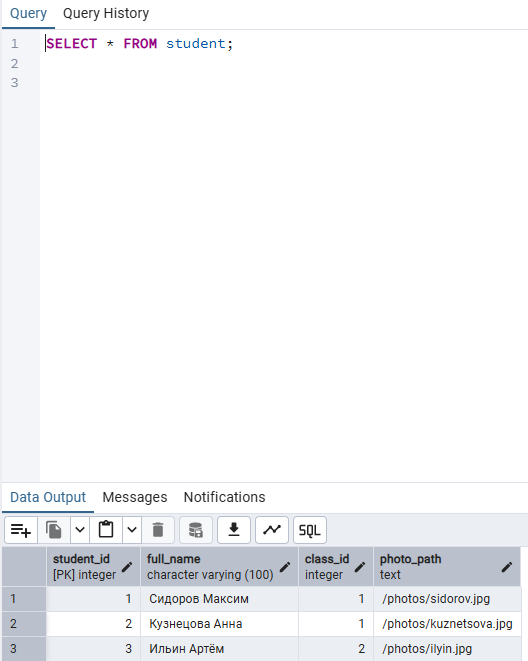


Рисунок 1.6 – Наличие данных в таблице

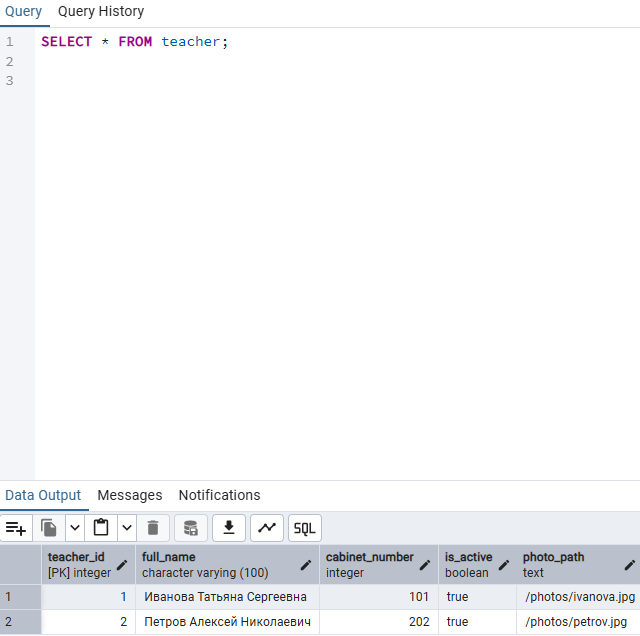


Рисунок 1.7 – Наличие данных в таблице

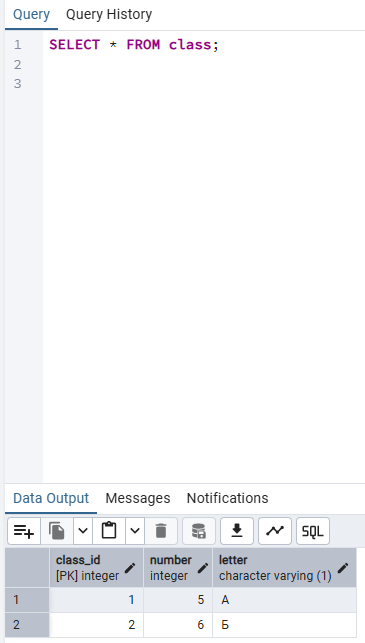


Рисунок 1.8 – Наличие данных в таблице

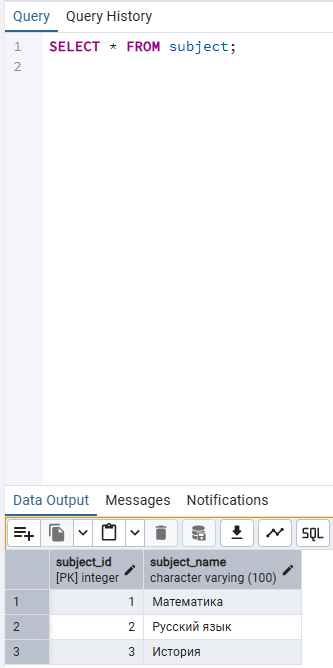


Рисунок 1.9 – Наличие данных в таблице

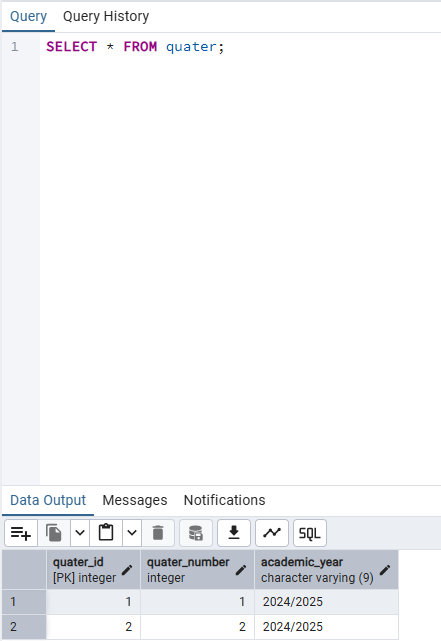


Рисунок 1.10 – Наличие данных в таблице

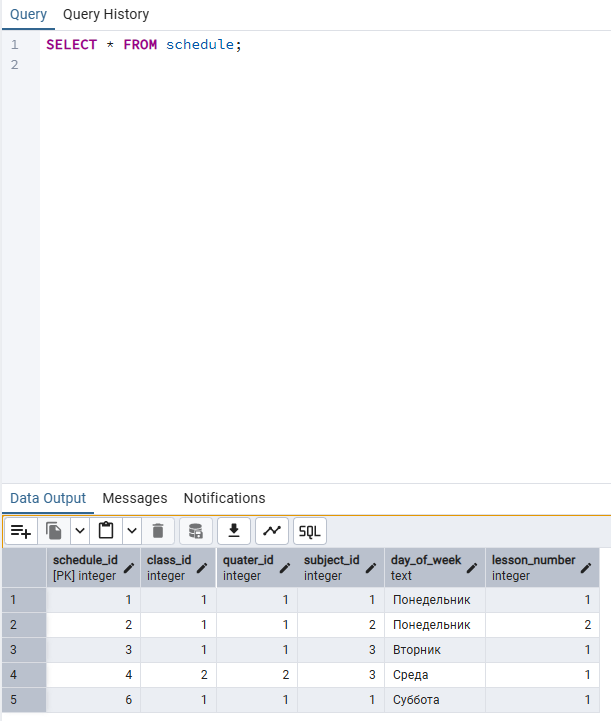


Рисунок 1.11 – Наличие данных в таблице

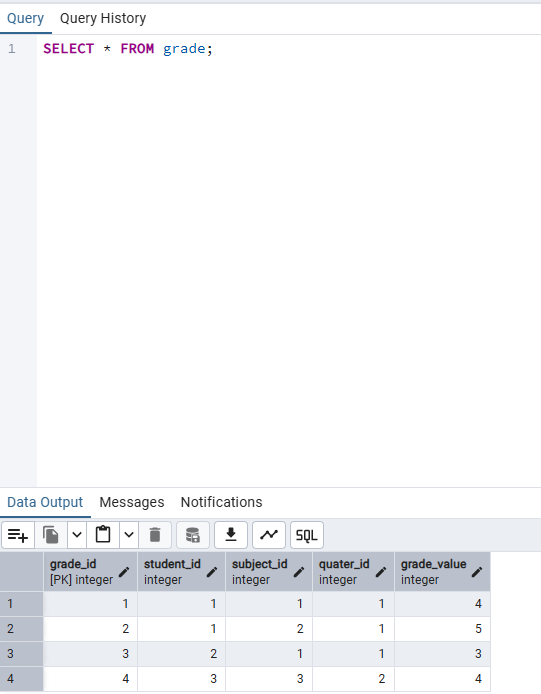


Рисунок 1.12 – Наличие данных в таблице

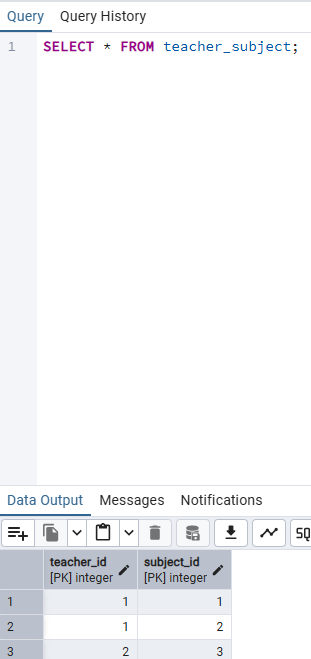


Рисунок 1.13 – Наличие данных в таблице

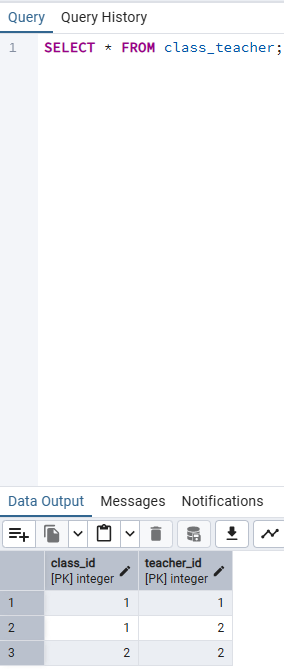


Рисунок 1.14 – Наличие данных в таблице

БД проверят корректность ввод данных при добавлении элементов (рисунок 1.15 – 1.16), согласно выставленным ограничениям.



Рисунок 1.15 – Проверка добавления элементов



Рисунок 1.16 – Проверка добавления элементов

**ЗАКЛЮЧЕНИЕ**

В ходе выполнения лабораторной работы была спроектирована и реализована реляционная база данных, отражающая структуру школьной информационной системы. Были созданы основные таблицы, такие как student, teacher, class, subject, grade и другие, а также определены связи между ними с использованием внешних ключей.

Особое внимание было уделено применению ограничений целостности (PRIMARY KEY, FOREIGN KEY, NOT NULL, CHECK, UNIQUE), что обеспечило корректность и надёжность хранения данных. Для проверки функционирования базы данных были добавлены тестовые и некорректные данные, что позволило убедиться в правильности реализованных ограничений.

Таким образом, цель лабораторной работы была достигнута: приобретены теоретические знания и практические умения по проектированию, созданию и тестированию реляционных баз данных.