МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ

ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

(ФГБОУ ВО «ВГТУ», ВГТУ)

ФАКУЛЬТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И КОМПЬЮТЕРНОЙ БЕЗОПАСНОСТИ

КАФЕДРА КОМПЬЮТЕРНЫХ ИНТЕЛЛЕКТУАЛЬНЫХ ТЕХНОЛОГИЙ ПРОЕКТИРОВАНИЯ

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №8

По дисциплине: «Системы хранения и обработки данных»

Тема: «Разработка физической структуры базы данных»

Выполнил работу студент группы мИИВТ-231 Костюков К.А.

Принял: Короленко В.В.

Защищена \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Воронеж 2023

**Цель работы:**

изучить и освоить процесс разработки физической структуры базы данных с использованием системы управления базами данных Postgres.

**Основные задачи:**

* создание таблиц в СУБД Postgres с помощью SQL-запросов с атрибутами, связями, первичными и внешними ключами в соответствии с разработанной в предыдущей лабораторной работе логической структурой базы данных;
* построение графической диаграммы для отображения физической структуры базы данных.

**Ход работы:**

После изучения основ работы с СУБД PostgreSQL и инструментария для работы с СУБД DBeaver, были сформированы SQL-запросы для создания таблиц в СУБД Postgres, соответствующих логической структуре базы данных, разработанной в предыдущей лабораторной работе.

SQL-запросы:

CREATE TABLE Абитуриент (

Абитуриент\_ID SERIAL PRIMARY KEY,

Имя VARCHAR(50),

Фамилия VARCHAR(50),

Дата\_рождения DATE,

Адрес VARCHAR(100)

);

Данный запрос создает таблицу "Абитуриент" в базе данных, определяет структуру таблицы с заданными полями и их типами данных.

Элементы запроса:

* CREATE TABLE - ключевое слово, указывающее на создание новой таблицы.
* АбитуриентID - это столбец таблицы, который будет автоматически генерировать уникальные значения (с помощью типа данных SERIAL).
* SERIAL - тип данных, который обеспечивает автоинкремент и генерацию последовательных уникальных значений для столбца АбитуриентID.
* PRIMARY KEY - ограничение, указывающее, что столбец АбитуриентID является первичным ключом.
* Имя - столбец таблицы, который будет содержать значения имени абитуриента типа VARCHAR с максимальной длиной в 50 символов.
* Фамилия - аналогично столбцу Имя, но для хранения фамилии абитуриента.
* Датарождения - столбец типа DATE для хранения даты рождения абитуриента.
* Адрес - столбец типа VARCHAR с максимальной длиной в 100 символов для хранения адреса абитуриента.

CREATE TABLE Экзаменатор (

Экзаменатор\_ID SERIAL PRIMARY KEY,

Имя VARCHAR(50),

Фамилия VARCHAR(50),

Специализация VARCHAR(50),

Уровень\_оплаты DECIMAL(10, 2)

);

Этот запрос создает таблицу "Экзаменатор" в БД.

Элементы запроса:

* CREATE TABLE - ключевое слово, указывающее на создание новой таблицы.
* ЭкзаменаторID: столбец типа SERIAL, который будет автоматически генерировать уникальные идентификаторы для каждого экзаменатора.
* Имя: столбец типа VARCHAR(50), который будет хранить имя экзаменатора.
* Фамилия: столбец типа VARCHAR(50), который будет хранить фамилию экзаменатора.
* Специализация: столбец типа VARCHAR(50), который будет хранить специализацию экзаменатора.
* Уровеньоплаты: столбец типа DECIMAL(10, 2), который будет хранить уровень оплаты экзаменатора.

CREATE TABLE Экзамен (

Экзамен\_ID SERIAL PRIMARY KEY,

Название VARCHAR(100),

Дата\_проведения DATE,

Аудитория VARCHAR(50)

);

Этот запрос, как и остальные создает таблицу, но со следующими столбцами:

* ЭкзаменID: столбец типа SERIAL, который будет автоматически генерировать уникальные идентификаторы для каждого экзамена.
* Название: столбец типа VARCHAR(100), который будет хранить название экзамена.
* Датапроведения: столбец типа DATE, который будет хранить дату проведения экзамена.
* Аудитория: столбец типа VARCHAR(50), который будет хранить информацию о аудитории, где будет проводиться экзамен.

CREATE TABLE Результат\_экзамена (

Результат\_ID SERIAL PRIMARY KEY,

Оценка INTEGER,

Дата\_сдачи DATE,

Абитуриент\_ID INTEGER REFERENCES Абитуриент(Абитуриент\_ID),

Экзамен\_ID INTEGER REFERENCES Экзамен(Экзамен\_ID),

Экзаменатор\_ID INTEGER REFERENCES Экзаменатор(Экзаменатор\_ID)

);

Данный запрос создает таблицу "Результат экзамена".

Элементы запроса:

* РезультатID: SERIAL PRIMARY KEY. Здесь элемент является первичным ключом. Он уникально идентифицирует каждую запись в таблице.
* Оценка: INTEGER. Это колонка, тип данных которой предназначен для хранения оценки результата экзамена.
* Датасдачи: DATE. Этот столбец, который сохраняет дату сдачи экзамена.
* АбитуриентID: INTEGER REFERENCES Абитуриент(АбитуриентID). Это столбец, который является внешним ключом и ссылается на идентификатор абитуриента из таблицы "Абитуриент".
* ЭкзаменID: INTEGER REFERENCES Экзамен(ЭкзаменID) является внешним ключом и ссылается на идентификатор экзамена из таблицы "Экзамен".
* ЭкзаменаторID: INTEGER REFERENCES Экзаменатор(ЭкзаменаторID) также является внешним ключом и указывает на идентификатор экзаменатора из таблицы "Экзаменатор".

CREATE TABLE Оплата\_экзаменатора (

Оплата\_ID SERIAL PRIMARY KEY,

Сумма\_оплаты DECIMAL(10, 2),

Дата\_оплаты DATE,

Экзаменатор\_ID INTEGER REFERENCES Экзаменатор(Экзаменатор\_ID)

);

Последний запрос создает таблицу "Оплата экзаменатора" со следующими составляющими:

* ОплатаID: столбец типа SERIAL, который будет использоваться в качестве первичного ключа таблицы.
* Сумма\_оплаты: столбец типа DECIMAL (10, 2), предназначенный для хранения суммы оплаты экзаменатора.
* Датаоплаты: столбец типа DATE, который будет хранить дату оплаты.
* ЭкзаменаторID: столбец типа INTEGER, который будет ссылаться на столбец "ЭкзаменаторID" в таблице "Экзаменатор".

После создания запросов необходимо обернуть созданную базу данных в docker-контейнер, содержимое файла docker-compose.yml указано ниже:

version: '3'

services:

postgres:

image: postgres:latest

environment:

POSTGRES\_DB: dbname

POSTGRES\_USER: dbuser

POSTGRES\_PASSWORD: dbpass

ports:

- "5432:5432"

volumes:

- pg\_data:/var/lib/postgresql/data

volumes:

pg\_data:

На рисунке 1 изображена диаграмма, отображающая физическую структуру базы данных.

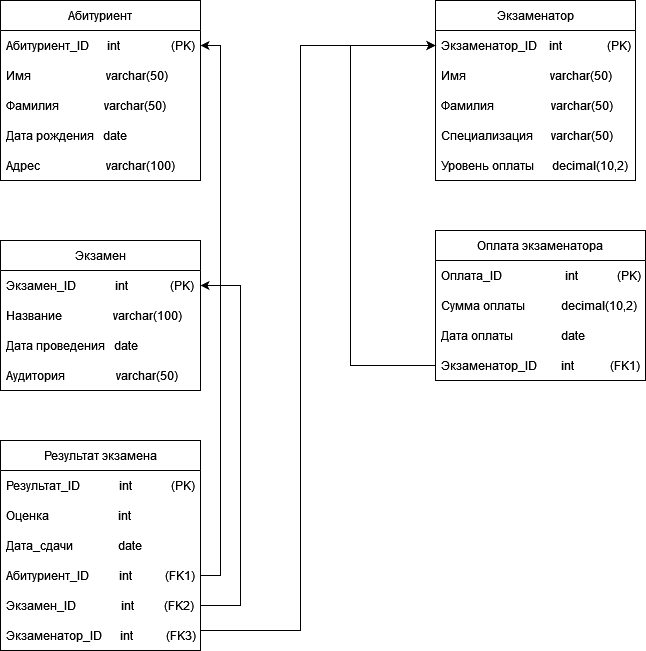


Рисунок 1 – Диаграмма

На диаграмме отображены таблицы, где каждый прямоугольник представляет отдельную таблицу. Внутри прямоугольника указаны атрибуты таблицы, такие как названия столбцов и их тип данных. Также были указаны индексы, которые отображают первичные и внешние ключи таблиц.

Связи между таблицами представляются линиями, указывающими на поля, которые являются внешними ключами. Это отображает связь между таблицами и определяет, как они связаны между собой.

**Вывод**: в данной работе были изучен и освоен процесс разработки физической структуры базы данных с использованием системы управления базами данных Postgres.

**Контрольные вопросы**

1. Особенности PostgreSQL.

Зачастую выделяют следующие особенности PostgreSQL:

- Надежность и целостность данных

- Расширяемость и поддержка пользовательских типов данных

- Гибкость в моделях индексирования и запросов

- Масштабируемость и поддержка больших объемов данных

- Поддержка ACID-транзакций

- Множество расширений и дополнительных функций

- Открытый и бесплатный для использования.

1. Что такое DDL?

DDL (Data Definition Language) - это язык определения данных, который используется в системах управления базами данных для создания, изменения и удаления структур данных, таких как таблицы, индексы, представления и т. д. DDL включает в себя команды, такие как CREATE, ALTER и DROP, которые позволяют определить схему базы данных и ее объекты.

1. Что такое DML?

DML (Data Manipulation Language) - это язык манипулирования данными, который используется в СУБД для выполнения операций над данными. Он позволяет добавлять, изменять, удалять и извлекать данные из таблиц. Основные команды DML включают INSERT (добавление данных), UPDATE (изменение данных), DELETE (удаление данных) и SELECT (извлечение данных). DML позволяет пользователям работать с данными внутри базы данных, обеспечивая изменение и получение нужной информации.

1. Что такое DCL?

**DCL (Data Control Language)** - группа операторов определения доступа к данным. Иными словами, это операторы для управления разрешениями, с помощью них мы можем разрешать или запрещать выполнение определенных операций над объектами базы данных.

1. С какой команды, как правило, начинается запрос?

Запросы обычно начинаются с команды SELECT, которая используется для получения данных из таблицы.

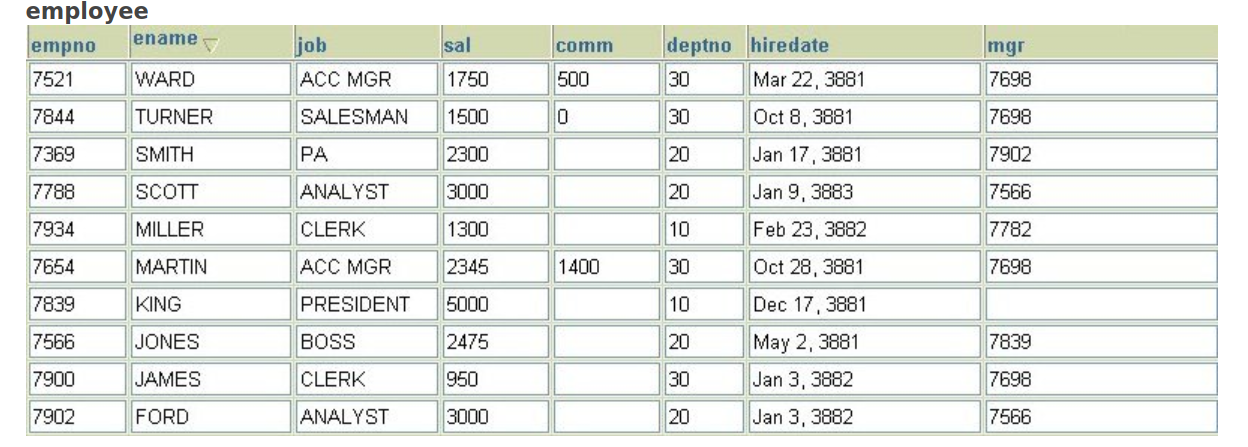
1. Какая команда используется для создания базы данных?

В PostgreSQL для создания новой базы данных используется команда CREATE DATABASE.

1. Какая команда используется для создания таблицы в базе данных?

Для создания новой таблицы используется команда CREATE TABLE.

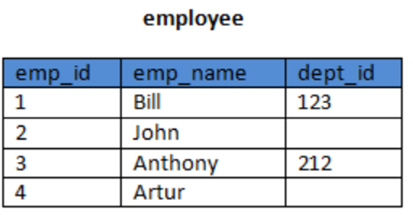
1. Существует следующая таблица:



Из таблицы employee необходимо показать сотрудников с empno 7844, 7788 и 7902 одним запросом. Какие из приведенных ниже запросов соответствуют данному условию?

* SELECT \* FROM employee WHERE sal IN (1500, 3000);

1. В базе данных компании хранятся данные о работниках и подразделениях, к которым они относятся.



Ваша задача: достать весь список работников, при этом все работники без подразделения должны быть в начале списка. Какой из представленных запросов выполнит эту задачу?

* SELECT \* FROM employee ORDER BY dept\_id NULLS FIRST;

1. Какие из следующих значений подходят для типа char в PostgreSQL?

* '1'
* 'a'
* 'char'

1. Какие из перечисленных конструкций запроса НЕ являются верными? Выбрать можно несколько вариантов ответа.

* Select список полей from список таблиц union Select список полей from список таблиц order by поля для сортировки
* Select список полей from список таблиц order by поля для сортировки union Select список полей from список таблиц order by поля для сортировки

1. Необходимо вычислить средний объем продаж (advance) книг и годовой доход (sales) от продажи книг по всем категориям (type) из таблицы titles. Какой из перечисленных фрагментов кода позволит решить поставленную задачу?

* Select avg (advance), sum (sales) from titles