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

ОТЧЁТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №8

По дисциплине: «Системы хранения и обработки данных»

Тема: «Разработка физической структуры базы данных»

Выполнил работу студент группы мИИВТ-231: Никулин В.С.

подпись, дата

Принял: Короленко В.В.

подпись, дата

Воронеж 2023

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

**Задачи работы:** необходимо выполнить все пункты учебной задачи:

1) Изучить основы работы с СУБД PostgreSQL.

2) Изучить инструментарий для работы с СУБД (DBeaver или аналог).

3) Сформировать SQL-запросы для создания таблиц в СУБД Postgres, соответствующих логической структуре базы данных, разработанной в предыдущей лабораторной работе.

4) Сохранить SQL-запросы в соответствующем файле со скриптами SQL.

5) Описать каждый запрос (что делает запрос, из каких элементов (составляющих) состоит запрос, роль этих составляющих).

6) Обернуть созданную базу данных в docker-контейнер (файл docker-compose.yml).

7) Построить графическую диаграмму, отображающую физическую структуру базы данных. Сохранить диаграмму в графическом файле.

8) Описать диаграмму. Пояснить выбор сущностей, атрибутов, связей, ключей.

9) Подготовить отчёт о проделанной работе.

10) Отчётные материалы загрузить в репозиторий Git и отправить ссылку на ваш репозиторий на платформе github на почту преподавателю. Репозиторий должен быть публичным.

**Ход работы:**

Лабораторная работа выполнялась в операционной системе Fedora Linux. Для начала был создан файл docker-compose.yml (рисунок 1). Также, был создан файл script.sql, где будет записан SQL-скрипт для создания таблиц базы данных в соответствии с логической схемой из прошлой лабораторной работы. Данный скрипт занесён в графу «volumes», чтобы при выполнении данного docker compose-файла все таблицы и связи между ними были созданы автоматически.



Рисунок - Содержимое файла «docker-compose.yml»

Далее, скрипт был выполнен и запущен соответствующий контейнер в Docker. После чего, было выполнено подключение к базе данных с помощью DBeaver, где впоследствии были написаны SQL-запросы для создания таблиц базы данных  
(рисунок 2).

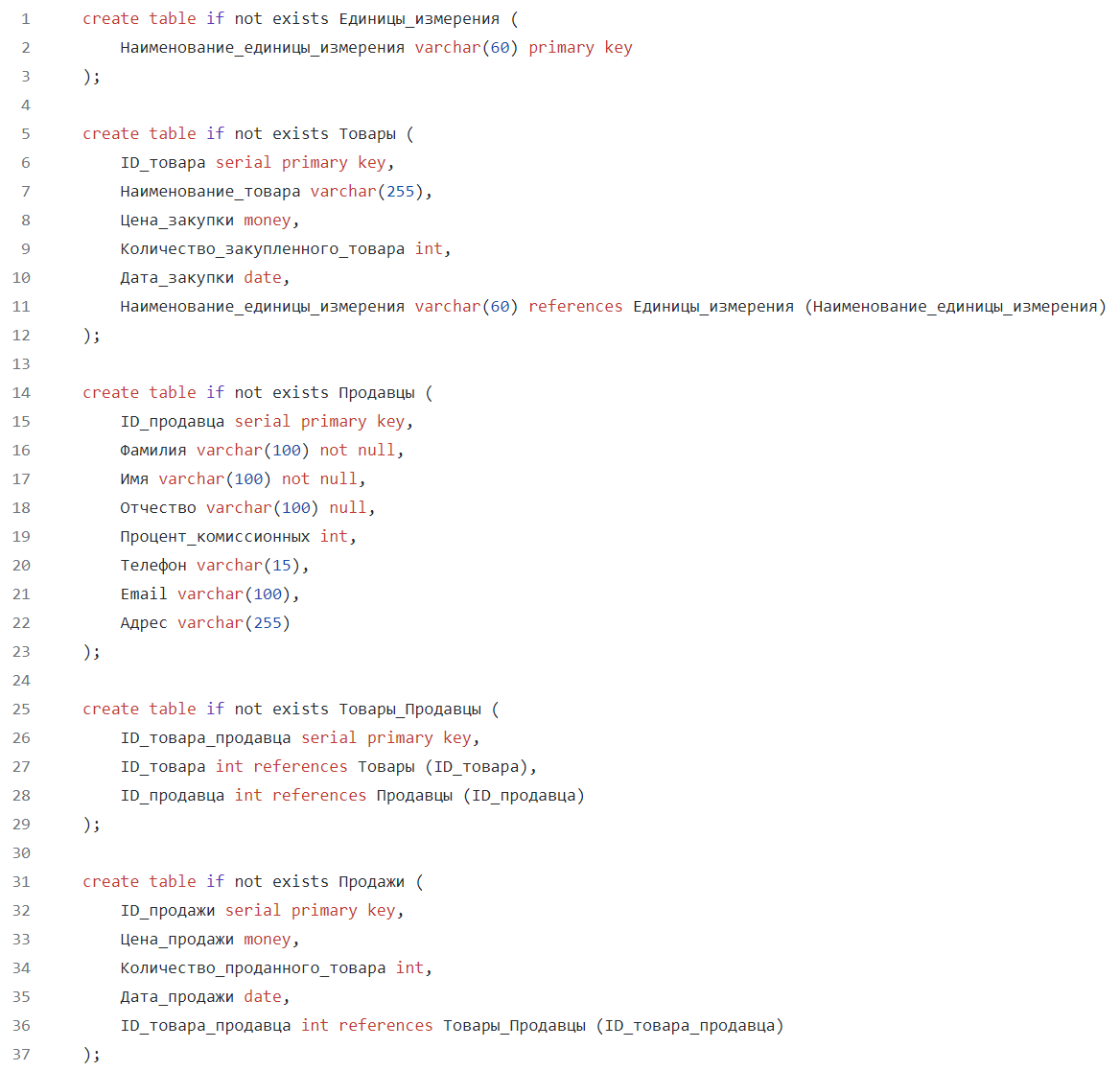


Рисунок - Содержимое файла script.sql с SQL-запросами для создания таблиц БД

Каждый скрипт для создания таблицы записан в порядке иерархии связей, т.е. сначала идут скрипты для таблиц без связей, а только потом скрипты для таблиц, со связями. Таким образом, вначале был записан SQL-скрипт для таблицы «Единицы измерения». Ключевой слово «create table if not exists» позволяет создать новую таблицу при условии, что она ещё не существует в БД. Далее идёт название таблицы, после этого в круглых скобках перечисляются поля таблицы через запятую. У таблицы «Единицы измерения» только одно поле – «Наименование единицы измерения», которое обладает типом «varchar» с максимальным количеством символов в 60 единиц. Это поле по совместительству является также первичным ключом таблицы, на что указывает ключевое слово «primary key».

Далее, был создан скрипт для таблицы «Товары». Создание таблицы происходит по аналогии с таблицей «Единицы измерения», ключевые слова одни и те же. Отличия только в количестве полей и типах данных. Поле «ID товара» является первичным ключом с типом данных «serial», который является разновидностью типа «int», специально созданным для того, чтобы использоваться в первичных ключах. Поле «Наименование товара» является типом данных «varchar» с максимальной длинной  
в 255 символов. Поле «Цена закупки» обладает типом «money», так как там будут записаны денежные единицы, поле «Количество закупленного товара» и «Дата закупки» обладают, соответственно, типами «int» и «date» (для записи даты). Поле «Наименование единицы измерения» является внешним ключом по отношению к таблице «Единицы измерения», поэтому данное поле копирует тип первичного ключа таблицы «Единицы измерение» (varchar в 60 символов). Ключевое слово «references» как раз-таки и указывает связь данного поля с таблицей «Единицы измерения», где после названия таблицы ещё указывается поле, на которое нужно сослаться в данной таблице. Таким образом создаётся связь между таблицами типа «Один ко многим». Остальные таблицы были созданы по аналогии с первыми двумя таблицами. Также, стоит отметить, что в таблице «Продавцы» использовались ключевые слова «not null» и «null» в полях фамилии, имени и отчества. Фамилия и имя всегда есть у человека, поэтому это поле «not null», а вот отчество есть не у всех людей, поэтому это поле может быть пустым «null» в некоторых случаях.

После записи всех SQL-скриптов, они были выполнены в DBeaver (рисунок 3). После чего, все таблицы появились в списке «Tables» (рисунок 3).

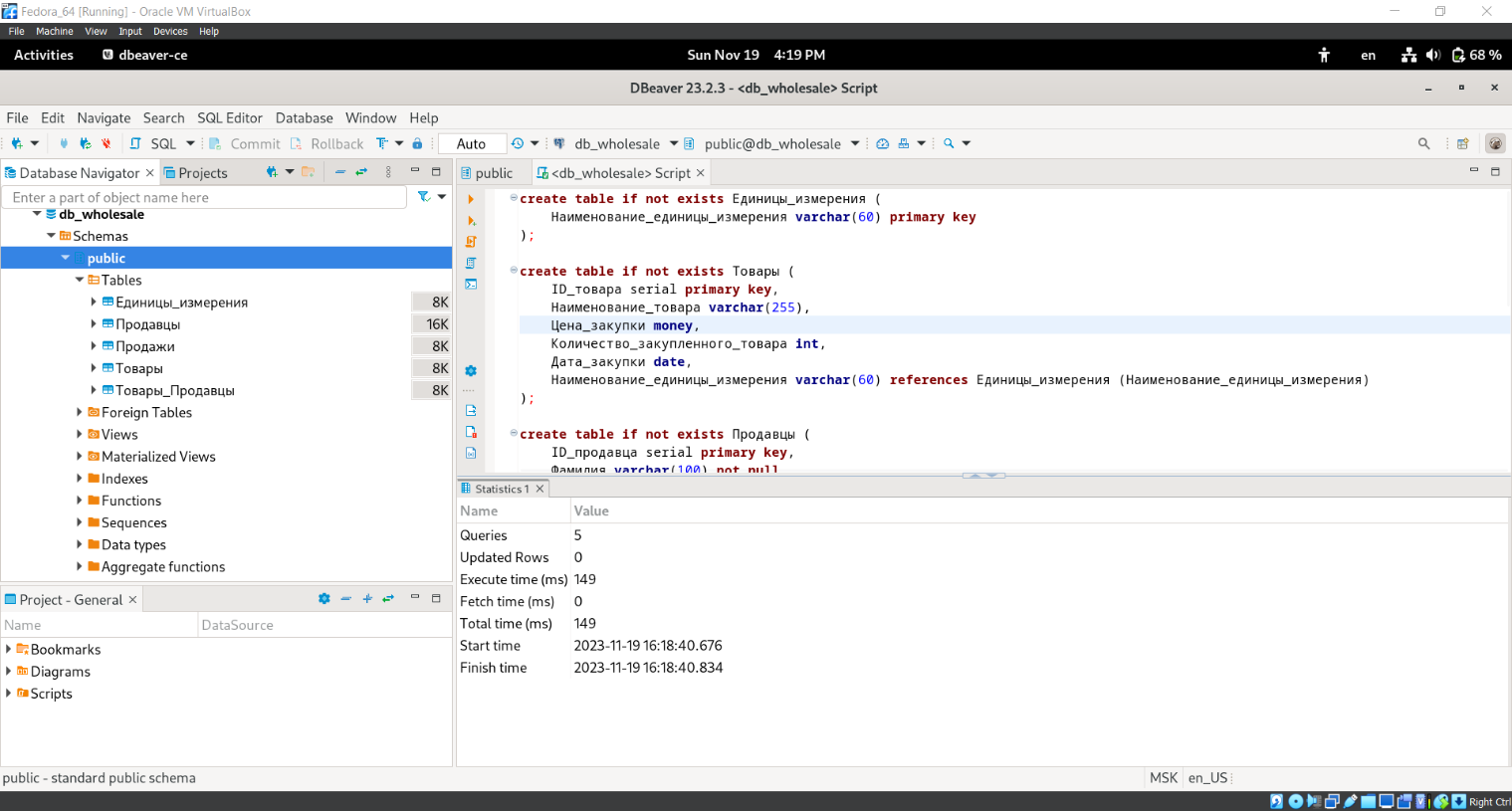


Рисунок - Файл «script.sql» со SQL-запросами был выполнен

Далее, был выполнен просмотр физической схемы БД в DBeaver. По нажатию кнопки «ER diagram» была получена физическая модель (схема/структура) для базы данных (рисунок 4).

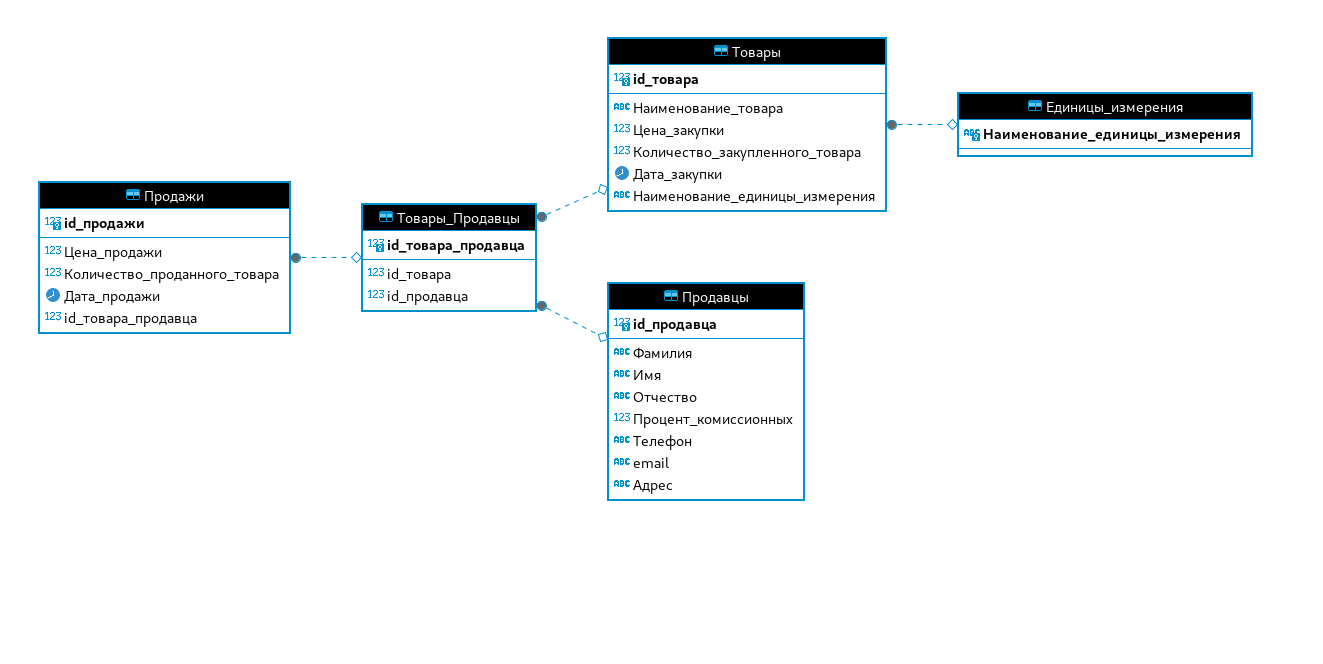


Рисунок – Физическая схема БД

**Ответы на контрольные вопросы:**

Вопрос 1: Особенности PostgreSQL.

Ответ:

1. PostgreSQL — объектно-реляционная СУБД.
2. Поддержка большого количества типов записи информации. Это не только стандартные целочисленные значения, числа с плавающей точкой, строки и булевы значения («да/нет»), но и денежный, геометрический, перечисляемый, бинарный и другие типы.
3. Работа с большими объемами. В большинстве СУБД, рассчитанных на средние и небольшие проекты, есть ограничения по объему базы и количеству записей в ней. В PostgreSQL ограничений нет. Ограничения касаются только конкретных записей. Одна таблица может занимать не больше 32 Тб, а одна запись — 1,6 Тб. В одном поле записи может быть не больше 1 Гб данных, а максимальное количество полей зависит от типа и составляет от 250 до 1600 штук. Максимальных значений хватает, чтобы хранить в БД любые данные.
4. PostgreSQL работает со сложными, составными запросами. Система справляется с задачами разбора и выполнения трудоемких операций, которые подразумевают и чтение, и запись, и валидацию одновременно. Она медленнее аналогов, если речь заходит только о чтении, но в других аспектах превосходит конкурентов.
5. Соответствие ACID. ACID — это набор принципов для обеспечения целостности данных. Аббревиатура расшифровывается как Atomicity, Consistency, Isolation, Durability — атомарность, согласованность, изолированность, прочность.
6. PostgreSQL — ПО с открытым исходным кодом, которое распространяется по свободной лицензии. Это означает, что любой разработчик может посмотреть, как написана система, или предложить для нее свои правки.
7. Чаще всего PostgreSQL используют на серверах с операционными системами семейства Linux, но СУБД поддерживает и другие ОС. Ее можно установить в системы на базе Windows, BSD, macOS и Solaris. Кроме того, у PostgreSQL есть автономный веб-сервер PostgREST, с которым можно работать с помощью REST API. СУБД можно развернуть и в облаке.

Вопрос 2: Что такое DDL?

Ответ: DDL (Data Definition Language) — это подмножество языка SQL, предназначенное для определения структуры базы данных. Оно включает операторы для создания, изменения и удаления объектов базы данных. Команды DDL используются для создания таблиц, индексов, представлений и других структур данных. Несколько примеров команд DDL: CREATE, ALTER, DROP, TRUNCATE.

Вопрос 3: Что такое DML?

Ответ: DML (Data Manipulation Language) — это подмножество языка SQL, которое отвечает за манипуляции данными внутри базы данных. Он включает операторы для добавления, изменения и удаления данных. Вот несколько команд DML: INSERT, UPDATE, DELETE, SELECT.

Вопрос 4: Что такое DCL?

Ответ: DCL (Data Control Language) — это подмножество языка SQL, которое управляет правами доступа и безопасностью в базе данных. Он включает операторы для управления разрешениями пользователей и контролем доступа к данным. Вот две основные команды DCL: GRANT, REVOKE.

Вопрос 5: С какой команды, как правило, начинается запрос?

Ответ: Запросы к базе данных обычно начинаются с команды SELECT. Эта команда используется для извлечения данных из одной или нескольких таблиц.

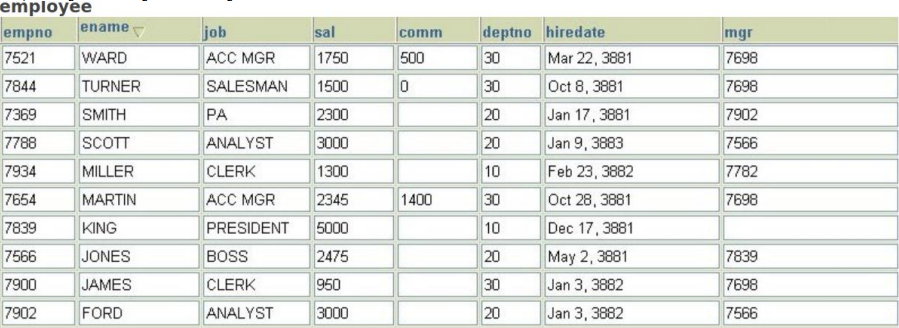
Вопрос 6: Какая команда используется для создания базы данных?

Ответ: Для создания базы данных используется команда CREATE DATABASE mydatabase.

Вопрос 7: Какая команда используется для создания таблицы в базе данных?

Ответ: Для создания таблицы в PostgreSQL используется команда CREATE TABLE. Также, возможно использовать ключевое слово «IF NOT EXISTS», чтобы создать таблицу только в том случае, если она отсутствует в БД.

Вопрос 8: Существует следующая таблица:

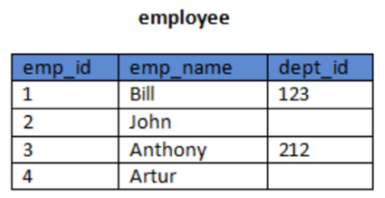


Из таблицы employee необходимо показать сотрудников с empno 7844, 7788 и 7902 одним запросом. Какие из приведенных ниже запросов соответствуют данному условию?

* SELECT \* FROM employee WHERE mod(mod(sal, 23), 5) = 0;
* SELECT \* FROM employee WHERE mod(sal, 23) = 0;
* SELECT \* FROM employee WHERE sal = 3000;
* SELECT \* FROM employee WHERE sal = 1500;
* SELECT \* FROM employee WHERE sal IN (1500, 3000);

Ответ: «SELECT \* FROM employee WHERE mod(mod(sal, 23), 5) = 0;» и  
SELECT \* FROM employee WHERE sal IN (1500, 3000);

Вопрос 9: 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 LAST;
* SELECT \* FROM employee ORDER BY dept\_id ASC;
* SELECT \* FROM employee ORDER BY dept\_id NULLS FIRST;
* SELECT \* FROM employee ORDER BY dept\_id;
* SELECT \* FROM employee ORDER BY dept\_id NULLS FIRST ASC;

Ответ: «SELECT \* FROM employee ORDER BY dept\_id NULLS FIRST;» и  
«SELECT \* FROM employee ORDER BY dept\_id NULLS FIRST ASC;». Оба этих запроса справятся с задачей, так как в ней не сказано должны ли работники с заполненным dept\_id быть отсортированы.

Вопрос 10: Какие из следующих значений подходят для типа char в PostgreSQL?

* 12
* '1'
* 1.22
* 'a'
* 'char'

Ответ:

1. '1' - Подходит. Это строка, и тип char может содержать символы.
2. 'a' - Подходит. Это также строка с одним символом.
3. 'char' - Подходит. Это строка фиксированной длины.

Вопрос 11: Какие из перечисленных конструкций запроса НЕ являются верными? Выбрать можно несколько вариантов ответа.

* Select список полей from список таблиц where условия order by поля для сортировки
* Select список полей from список таблиц union Select список полей from список таблиц order by поля для сортировки
* Select список полей from список таблиц having условия группировки group by поля для группировки
* Select список полей from список таблиц where условия group by поля для группировки 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

Ответ:

1. «Select список полей from список таблиц union Select список полей from список таблиц order by поля для сортировки» — эта конструкция неверна, потому что порядок ORDER BY должен указываться только один раз в запросе, и это делается после всех операторов SELECT. В данном случае, он указан дважды, что недопустимо.
2. «Select список полей from список таблиц where условия group by поля для группировки order by поля для сортировки» — эта конструкция тоже неверна, потому что порядок операторов должен быть SELECT, FROM, WHERE, GROUP BY, HAVING (если есть), ORDER BY.

Вопрос 12: 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 where type between ’business’ or  
  ’internet’
* Select avg (advance), sum (sales) from titles
* Select avg (advance), sum (sales) from titles where type between ’business’ and  
  ’internet’
* Select avg (advance), sum (sales) from titles group by type
* Select avg (advance), sum (sales) from titles where type = ’business’ and type =  
  ’internet’

Ответ: «Select avg(advance), sum(sales) from titles».

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