МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ  
 РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Чувашский государственный университет имени И.Н. Ульянова»

Факультет информатики и вычислительной техники

Кафедра математического и аппаратного обеспечения   
 информационных систем

Лабораторная работа №7 по дисциплине “Безопасность систем баз данных”

Тема: **Представления**

Выполнил: студент гр. ИВТ 21-22

Иванов Н.А. \_\_\_\_\_\_\_\_\_\_\_\_\_\_

Проверил Старший преподаватель

Марков А.В. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Цель работы:

Изучить способы создания представлений в БД PostgreSQL, создать запросы согласно заданию.

Содержание:

Дополнительная информация по созданию представлений доступна на https://postgrespro.ru/docs/postgresql/15/sql-createview, https://postgrespro.ru/docs/postgresql/15/rules-views.

CREATE VIEW создаёт представление запроса. Создаваемое представление лишено физической материализации, поэтому указанный запрос будет выполняться при каждом обращении к представлению.

Команда CREATE OR REPLACE VIEW действует подобным образом, но если представление с этим именем уже существует, оно заменяется. Новый запрос должен выдавать те же столбцы, что выдавал запрос, ранее определённый для этого представления (то есть, столбцы с такими же именами должны иметь те же типы данных и следовать в том же порядке), но может добавить несколько новых столбцов в конце списка. Вычисления, в результате которых формируются столбцы представления, могут быть совершенно другими.

Пример запросов по созданию представления:

список всех счетов со статусом 1

CREATE VIEW active\_accounts AS

SELECT

id, balance, id\_client

FROM

accounts

WHERE id\_account\_status = 1;

список всех счетов с дополнительным полем, в котором содержится количество всех исходящих платежей:

CREATE VIEW accounts\_and\_payments\_from\_cnt AS

SELECT

accounts.id as id,

COUNT(\*) as payments\_count

FROM

accounts

INNER JOIN payments on payments.id\_account\_from = accounts.id

GROUP BY accounts.id

Задание:

1. Создать, запустить и проверить работу скрипта create\_views.sql, который создаёт следующие представления:

- список платежей за последний месяц;

- список счетов, баланс которых больше 1000;

- список всех счетов с дополнительным полем, в котором содержится сумма всех входящих платежей;

- список всех счетов с дополнительным полем, в котором содержится сумма исходящих платежей, в которых сумма больше 100;

- список всех счетов с дополнительным полем, в котором содержится сумма исходящих платежей, в которых сумма больше 100, в итоговом списке должны быть только записи, в которых полученная сумма больше 1000;

- список всех страниц(pages) с дополнительным полем, в котором содержится название родительской страницы;

- список всех платежей из таблиц payments и archive\_payments c дополнительным полем, в котором выводится 1, если запись из payments, и 2, если запись из archive\_payments (использовать UNION).

1. Создать и запустить скрипт check\_views.sql, который выполняет запросы к БД, демонстрирующие работоспособность и правильность представлений из п.1 (достаточно просто запросов, которые выводят все записи представления, например, SELECT \* FROM active\_accounts).
2. Создать и запустить скрипт drop\_views.sql, который удаляет представления из п.1.

Результат:

<https://github.com/bwl-andromeda/db/tree/lab-7>

Выводы:

В ходе данной лабораторной работы были изучены способы создания представлений в базе данных PostgreSQL. Представления являются важным инструментом, позволяющим организовать и упростить доступ к данным, выполняя сложные запросы и объединяя таблицы, без необходимости изменения самой структуры данных.

В рамках работы были созданы несколько представлений, обеспечивающих удобный доступ к информации:

1. Список платежей за последний месяц.
2. Список счетов с балансом выше 1000.
3. Список всех счетов с суммой входящих и исходящих платежей.
4. Представления для анализа платежей, превышающих определённые пороги по суммам.
5. Представления, объединяющие данные из таблиц payments и archive\_payments с использованием команды UNION, что позволило организовать единый интерфейс для работы с историей платежей.

Проверка работы представлений проводилась с помощью соответствующих SQL-запросов, продемонстрировавших корректность и работоспособность созданных представлений. Кроме того, для упрощения выполнения и удаления представлений были созданы скрипты create\_views.sql, check\_views.sql, и drop\_views.sql, что позволило автоматизировать процесс работы с ними.

Результатом выполнения данной лабораторной работы стало понимание того, как представления могут быть использованы для упрощения запросов и оптимизации работы с базой данных. В ходе экспериментов было показано, что представления значительно облегчают доступ к сложным данным, сокращают время написания запросов и делают работу с базой данных более эффективной.