Министерство науки и высшего образования РФ

**НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

Кафедра Вычислительной техники

ЛАБОРАТОРНАЯ РАБОТА №3

ПО ДИСЦИПЛИНЕ «Базы данных»

Факультет: АВТФ

Группа: АВТ-819

Студент: Ванин К.Е.

Преподаватель:

Бычков М.И.

**Цель работы**

Изучить правила создания функций. Приобрести практические навыки создания функций в среде PostgreSQL. Научиться проектировать БД в целом на основе поставленного задания с использованием всех полученных ранее базовых знаний и навыков.

**Задание**

Ознакомиться с теоретическими сведениями о возможностях создания пользовательских функций в PostgreSQL. Разработать БД в соответствии с индивидуальным заданием. Создать функции, реализующие интерфейс для работы с базой данных. Проверить работоспособность функций путем выполнения этих функций с параметрами, обеспечивающими как успешное выполнение функции, так и невыполнение функции.

Обязательные требования к БД:

1) Наличие таблиц-справочников и таблиц, использующих справочники. Предусмотреть сохранение ссылочной целостности для таблиц, использующих таблицы-справочники.

2) Предусмотреть следующие роли:

а) оператор БД (пополнение справочников)

б) пользователь БД (основная работа с БД, с ограничениями для некоторого вида операций)

в) аналитик (разрешено выполнение запросов и функций, не изменяющих данные в БД)

г) администратор БД (просмотр протокола операций, любые изменения БД)

3) Действия, изменяющие БД пользователем с любой ролью протоколируются в таблице-журнале операций.

4) Для всех запросов необходимо создать индексы (для гарантированного использования индексов можно использовать отключение параметра enable\_seqscan в текущей сессии)

Создать функции, реализующие интерфейс для работы с базой данных. Проверить работоспособность функций путем выполнения этих функций с параметрами, обеспечивающими как успешное выполнение функции, так и невыполнение функции.

*Вариант 4:* База данных сети книжных магазинов. Должна содержать следующие данные: текущие складские запасы печатной продукции, информацию о заказах и продажах.

Предусмотреть анализ следующих показателей:

* наиболее часто заказываемые книги
* средний чек по разным группам товаров
* рейтинг популярности книг для разных возрастных групп

**Ход работы**

Создадим новую базу данных.

create database bookshops;

\c bookshops

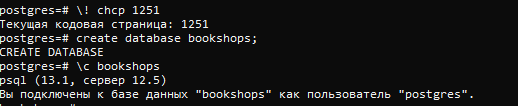


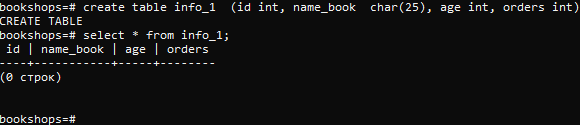
Рис. 1 – Создание новой базы данных

Создадим следующую таблицу:

1. info\_1 (id, name\_book, age, orders) - Создаем таблицу товаров, которая будет содержать информацию о идентификаторе товара (id), о названии книги (name\_book), о возрасте (age), о частоте заказа книги (orders).
2. info\_2 (id, product, orders, cost) Создаем таблицу товаров, которая будет содержать информацию о идентификаторе товара (id), о названии товара (product), о количестве заказов (orders), о стоимости (cost).

Создам таблицу 1:

create table info\_1 (id int, name\_book char(25), age int, orders int);

Рис. Рис. 2 – Создание таблицы info\_1

Заполняем таблицу:

insert into info\_1 values (1,'Harry Potter', 6, 8);

insert into info\_1 values (2,'Airport', 12, 5);

insert into info\_1 values (3,'Martin Eden', 16, 6);

insert into info\_1 values (4,'White Fang', 10, 3);

insert into info\_1 values (5,'Jane Eyre', 13, 11);

insert into info\_1 values (6,'Bridget Jones', 16, 10);

insert into info\_1 values (7,'To Kill a Mockingbird', 4, 12);

insert into info\_1 values (8,'Dead Souls', 11, 14);

insert into info\_1 values (9,'Three Men in a Boat', 14, 7);

insert into info\_1 values (10,'Fathers and Sons', 7, 6);

insert into info\_1 values (11,' Alice in Wonderland', 4, 5);

insert into info\_1 values (12,' Prince and the Pauper', 14, 8);

insert into info\_1 values (13,' Murder on Orient Express', 17, 4);

insert into info\_1 values (14,' Bridget Jones Diary', 8, 7);

insert into info\_1 values (15,' Interview with a Vampire', 12, 5);

Посмотрим содержимое таблицы 1:

select \* from info\_1;

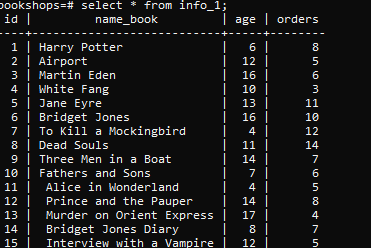


Рис. 3 – содержимое таблицы info\_1

Создам таблицу 2:

create table info\_2 (id int, product char(25), orders int, cost int);

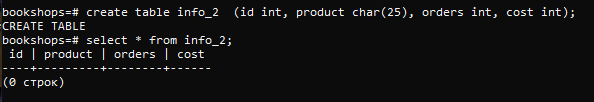


Рис. 4 – Создание новой базы данных info\_2

insert into info\_2 values (1,'book', 6, 8);

insert into info\_2 values (2,'journal', 12, 5);

insert into info\_2 values (3,'day book', 16, 6);

insert into info\_2 values (4,'bimonthly', 10, 3);

insert into info\_2 values (5,'biweekly', 13, 11);

insert into info\_2 values (6,'mag', 16, 10);

insert into info\_2 values (7,'ruler', 4, 12);

insert into info\_2 values (8,'pencil', 11, 14);

insert into info\_2 values (9,'pen', 14, 7);

insert into info\_2 values (10,'album', 7, 6);

insert into info\_2 values (11,'scrapbook', 4, 5);

insert into info\_2 values (12,'cover', 14, 8);

insert into info\_2 values (13,'envelope', 17, 4);

insert into info\_2 values (14,'album sheet', 8, 7);

insert into info\_2 values (15,'planer', 12, 5);

Посмотрим содержимое таблицы 2:

select \* from info\_2;

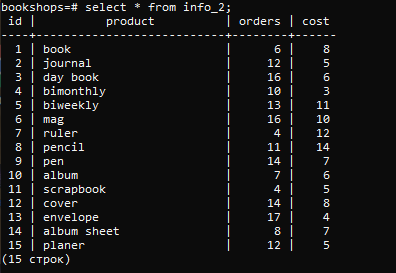


Рис. 5 – содержимое таблицы info\_2

* наиболее часто заказываемые книги

(1 задание)

create or replace function rating()returns

table (name\_book char(25), orders integer) as $$ begin

return query

select info\_1.name\_book, info\_1.orders

from info\_1 order

by info\_1.orders desc;

return; end; $$ language 'plpgsql';

select \*from rating();

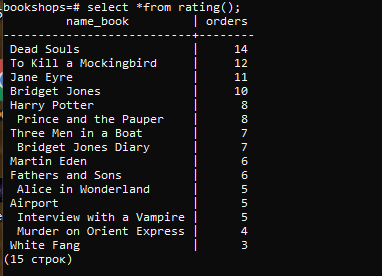


Рис. 6 Результат выполнения функции

* средний чек по разным группам товаров

(2 задание)

create or replace function avg\_receipt(p\_name char(25)) returns table (product char(25), orders int, cost int, avg\_r int)  
AS $$ begin return query select info\_2. product, info\_2.orders, info\_2.cost, (info\_2.orders \* info\_2.cost) avg\_rec from info\_2 where info\_2. product = p\_name; return; end; $$ language 'plpgsql';

Результат выполнения функции

select avg\_receipt('pen');

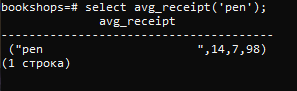


Рис. 7 Результат выполнения функции

* рейтинг популярности книг для разных возрастных групп

(3 задание)

Create or replace function age\_sort(int) returns table (name\_book char(20), orders int, age int) as $$ begin return query select info\_1.name\_book, info\_1.orders, info\_1.age from info\_1 where info\_1.age<=$1 order by info\_1.orders desc; return;end; $$ language 'plpgsql';

Результат выполнения функции:

select \*from age\_sort(7);

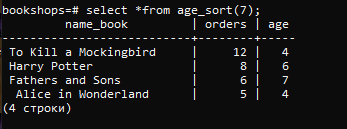


Рис. 8 Результат выполнения функции

Создадим таблицу-журнал для протоколирования любой операции. Таблица будет содержать следующие поля: операция, время совершения операции; пользователь, совершивший операцию; строка, над которой была совершена операция.

create table logs(operation char(6) not null, change\_time timestamp, users text, table\_name name, strings text not null);

Создадим функцию, добавляющую записи в журнал логирования:

create or replace function add\_to\_logs() returns trigger as $$

begin

if(tg\_op = 'DELETE') then insert into logs select 'delete', now(), user, TG\_RELNAME, OLD;

return OLD;

elseif(tg\_op = 'UPDATE') then insert into logs select 'update', now(), user, TG\_RELNAME, NEW;

return NEW;

elseif(tg\_op = 'INSERT') then insert into logs select 'insert', now(), user, TG\_RELNAME, NEW;return NEW;end if;return null;end;$$ language 'plpgsql';

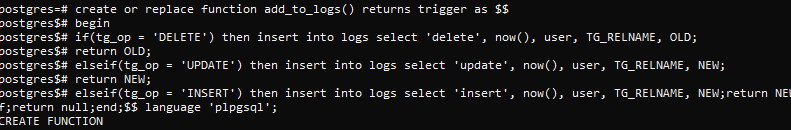


Рис. 9 – Создание таблицы-журнала и функции добавления записей в журнал логирования

Создадим триггеры для вызова функции add\_to\_logs() после вставки, удаления и обновления для каждой из таблиц, кроме таблицы logs.

create trigger log\_trigger after insert or update or delete on info\_1

for each row execute procedure add\_to\_logs();

create trigger log\_trigger after insert or update or delete on info\_2

for each row execute procedure add\_to\_logs();

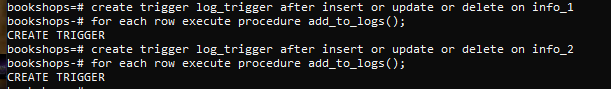


Рис. 10 – Создание триггеров

Создадим следующие роли:

а) оператор БД (пополнение справочников)

б) пользователь БД (основная работа с БД, с ограничениями для некоторого вида операций)

в) аналитик (разрешено выполнение запросов и функций, не изменяющих данные в БД)

г) администратор БД (просмотр протокола операций, любые изменения БД)

create role operatordb with password '123' nosuperuser nocreatedb nocreaterole login;

create role userdb with password '123'nosuperuser nocreatedb nocreaterole login;

create role analystdb with password '123' nosuperuser nocreatedb nocreaterole login;

create role admindb with password '123' superuser createdb createrole login;

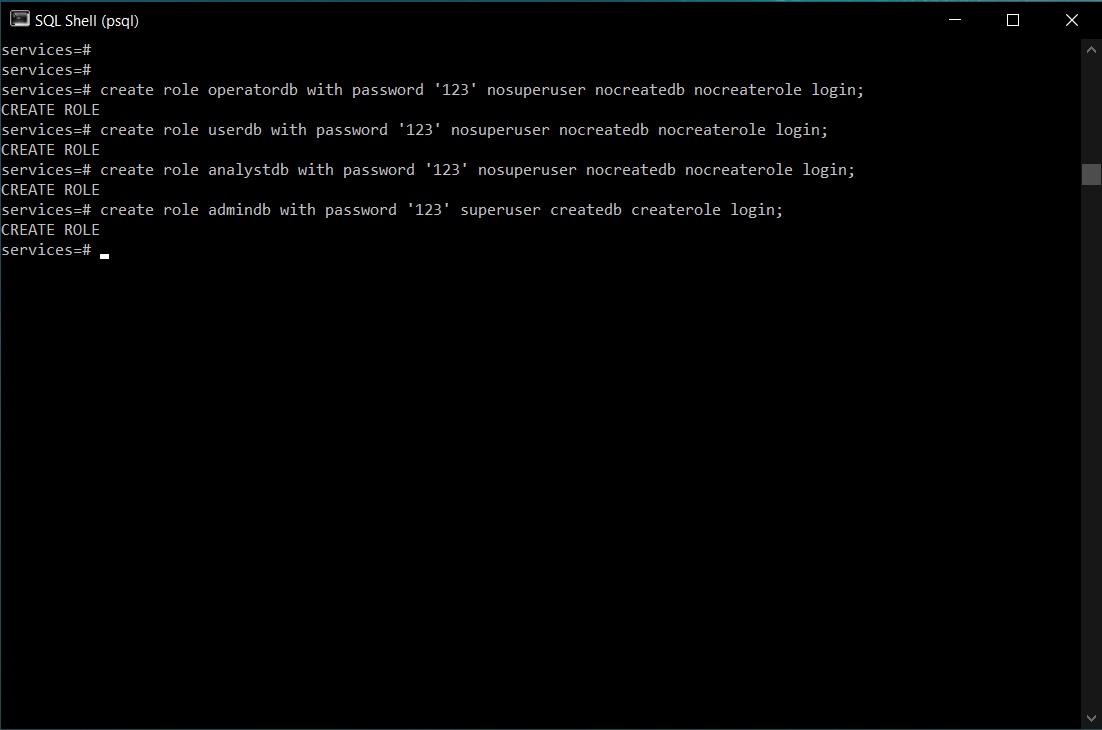


Рис. 11 – Создание ролей

Разрешим доступ на добавление новых записей всем пользователям в журнал логирования:

grant insert

on table logs to operatordb, userdb, analystdb, admindb;

Разрешим доступ оператору на изменение и просмотр таблиц-справочников:

grant update, select

on table info\_1, info\_2

to operatordb;

Разрешим доступ пользователю БД и выполнение основных операций (некоторые ограничения все же остаются):

grant insert, delete, update, select

on table info\_1, info\_2

to userdb;

Разрешим аналитику выполнять запросы и функции, не изменяющие данные в БД:

grant select

on table info\_1, info\_2

to analystdb;

Разрешим использовать все привилегии администратору:

grant all privileges

on table info\_1, info\_2

to admindb;

Для гарантированного использования индексов отключим в текущей сессии параметр enable\_scan.

create index idx\_info\_1 on info\_1(id);

create index idx\_info\_2 on info\_2(id);

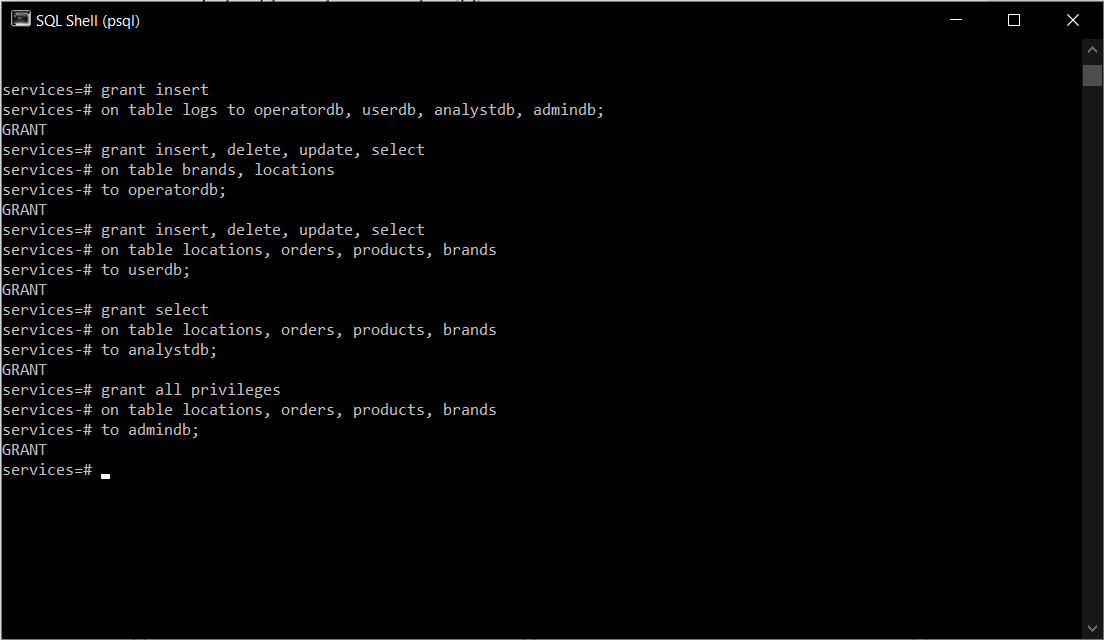


Рис. 12 – Установка привилегий для пользователей БД

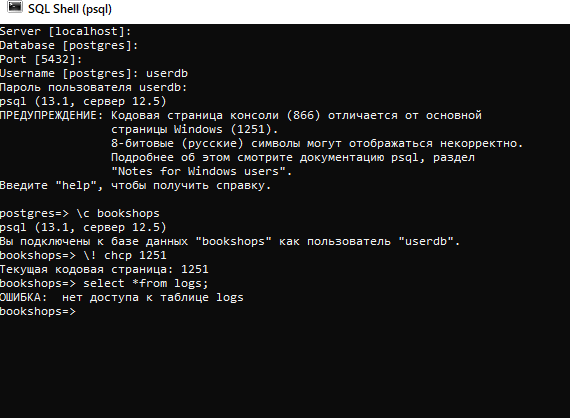


Рис. 13 – Демонстрация ограничений привилегий для пользователя userdb

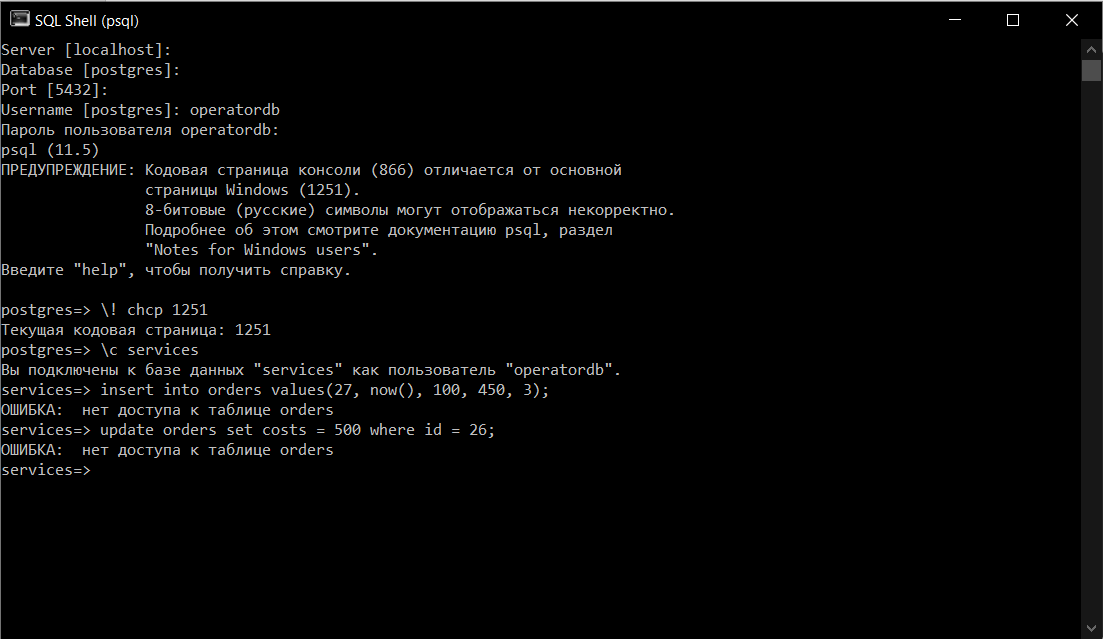


Рис. 14 – Демонстрация ограничений привилегий для пользователя operatordb

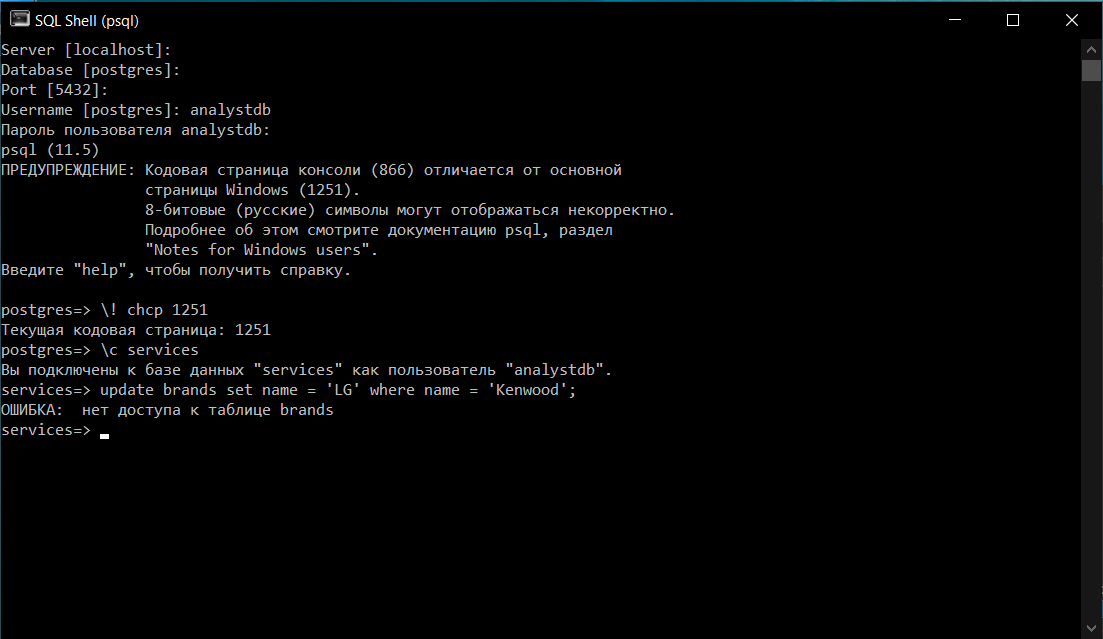


Рис. 15 – Демонстрация ограничений привилегий пользователя analystdb

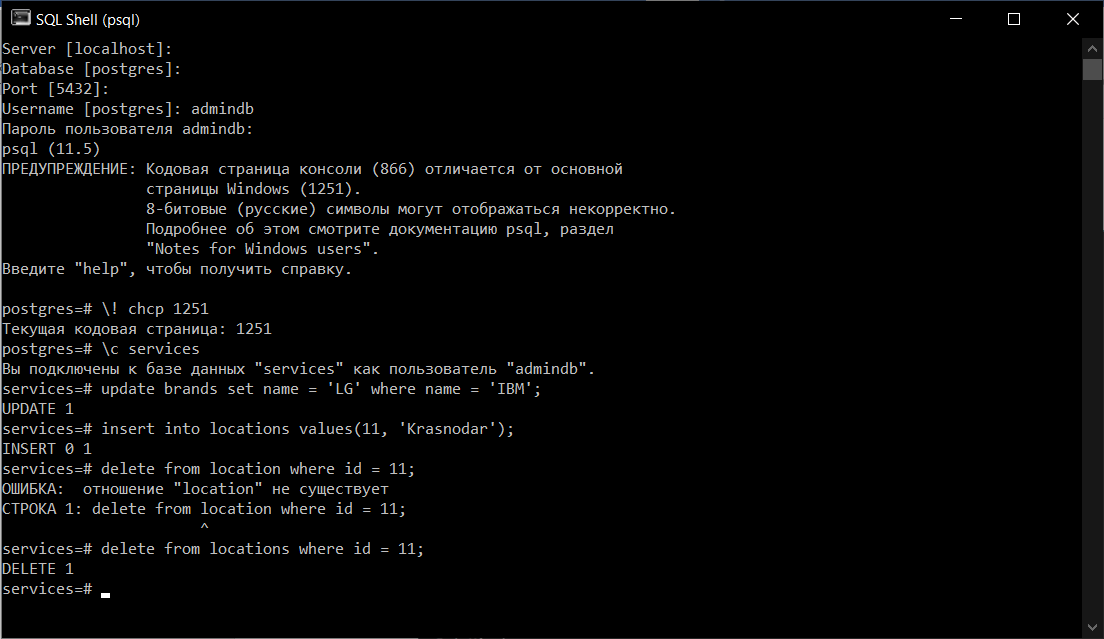


Рис. 16 – Демонстрация всех привилегий пользователя admindb

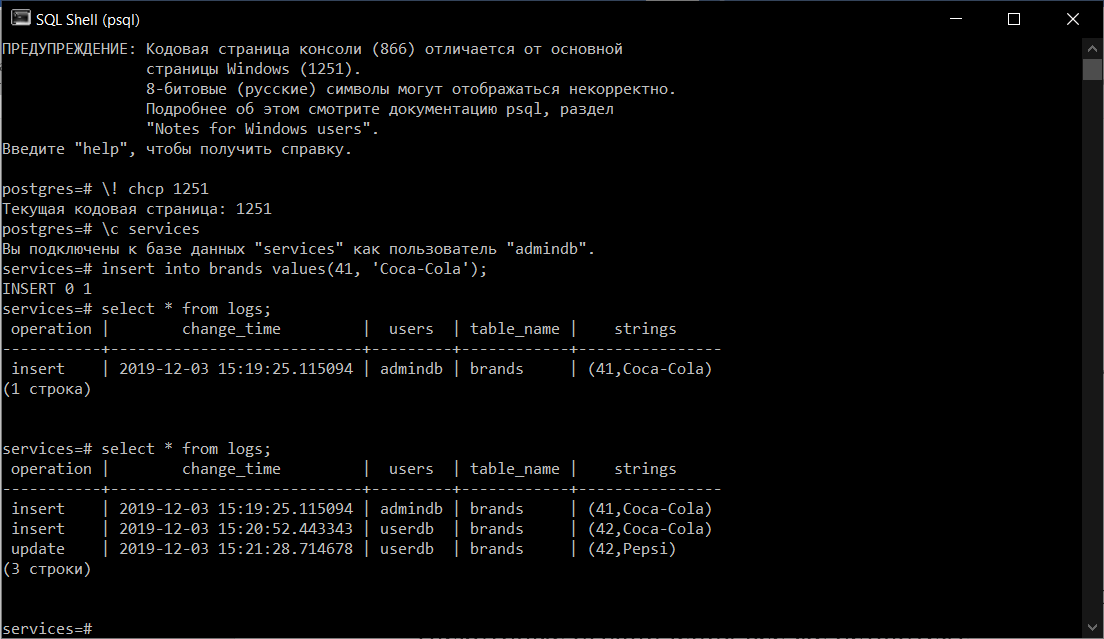


Рис. 17 – Выборка таблицы logs

**Выводы**

В ходе выполнения лабораторной работы были изучены правила создания функций, приобретены практические навыки создания функций в среде PostgreSQL, получены навыки проектирования БД в целом на основе поставленного задания с использованием всех полученных ранее базовых знаний и навыков.