Делаем по гайду <https://info-comp.ru/install-postgresql-13-on-ubuntu-server>  
тест  
ubuntusql@ubuntusql-VirtualBox:~/sql$ docker run -d --name pg -p 5432:5432 -v /home/ubuntusql/sql/pg:/var/lib/postgresql/data -e POSTGRES\_PASSWORD=mysecretpassword postgres:12

test@ubuntu:~/Docker$ docker exec -ti 48967449986d /bin/bash

root@cd6bf3a66786:/# su - postgres # Давайте переключимся на пользователя postgres (данная учетная запись была создана автоматически во время установки PostgreSQL).

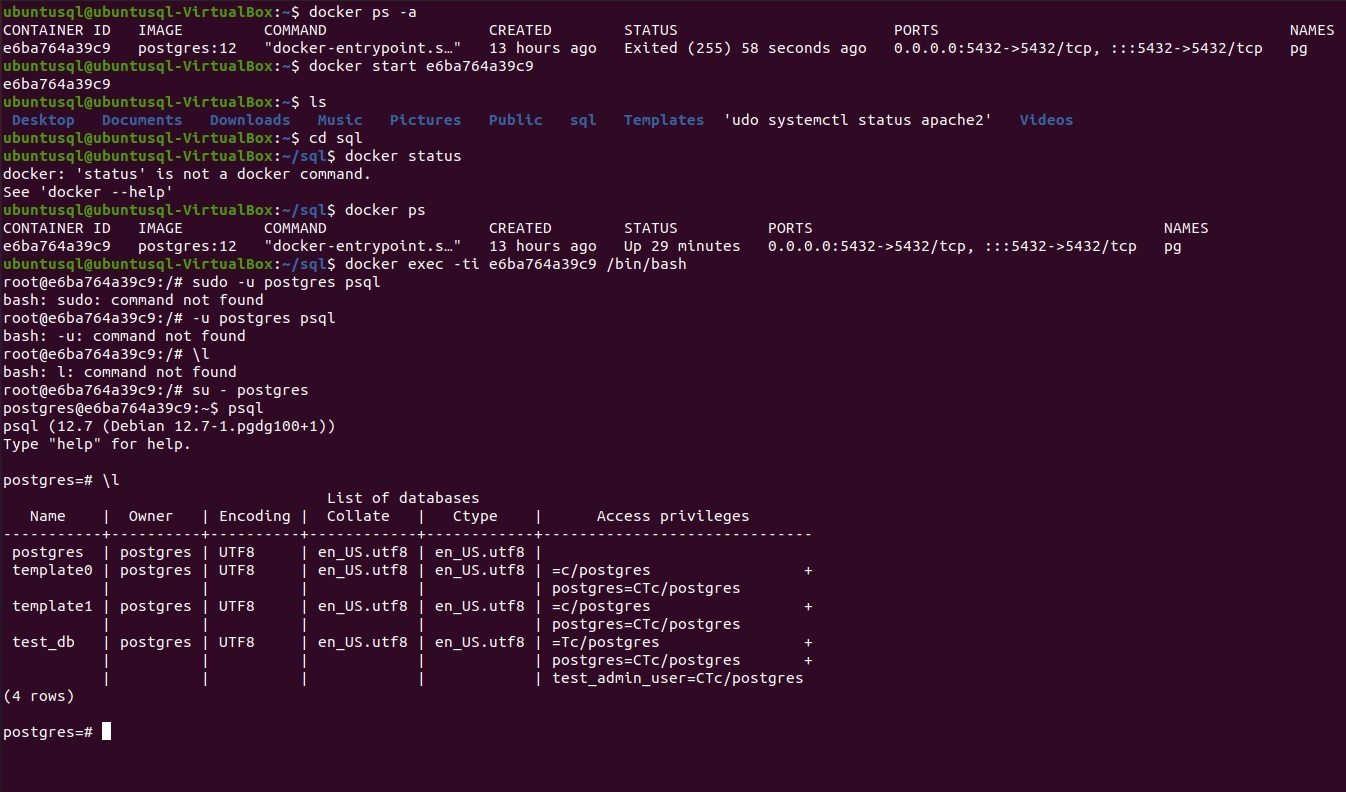
postgres@cd6bf3a66786:~$ psql # Затем запускаем утилиту psql – это консоль для PostgreSQL.

postgres=# \password postgres #Первым делом нам нужно задать пароль для пользователя postgres.

1234

postgres=# \l # вывести список БД

см. скриншот “Список бд Домашнее задание к занятию 6.2. SQL.jpg”



postgres=# create database test\_db # Далее давайте создадим базу данных.

1.2 создаем таблички:

**create** **table** orders(

id SERIAL,

**name** **VARCHAR**(255),

price **INT**

);

**create table clients (**

**id SERIAL,**

**Surname VARCHAR(100),**

**"contry of residence" VARCHAR(100),**

**"order" VARCHAR(100)**

**);**

create user test\_admin\_user; # Затем создаем нового пользователя на сервере PostgreSQL, так как работать от имени postgres крайне не рекомендуется.  
  
  
1.3 postgres=# grant all privileges on database test\_db to test\_admin\_user; # выдать привилегии на БД test\_db юзеру test\_admin\_user и в конце обязательно ;

ALTER ROLE test\_admin\_user SUPERUSER CREATEDB CREATEROLE INHERIT LOGIN;

grant all privileges on database test\_db to test\_admin\_user;

GRANT ALL ON TABLE public.clients TO test\_admin\_user;

GRANT ALL ON TABLE public.orders TO test\_admin\_user;

создать пользователя # test\_simple\_user

# и выдать права **GRANT SELECT INSERT UPDATE DELETE на табличку** clients и orders

**CREATE** **ROLE** test\_simple\_user NOSUPERUSER NOCREATEDB NOCREATEROLE NOINHERIT LOGIN;

**GRANT SELECT ON TABLE public.clients TO test\_simple\_user;**

**GRANT INSERT ON TABLE public.clients TO test\_simple\_user;**

**GRANT UPDATE ON TABLE public.clients TO test\_simple\_user;**

**GRANT DELETE ON TABLE public.clients TO test\_simple\_user;**

**GRANT SELECT ON TABLE public.orders TO test\_simple\_user;**

**GRANT INSERT ON TABLE public.orders TO test\_simple\_user;**

**GRANT UPDATE ON TABLE public.orders TO test\_simple\_user;**

## GRANT DELETE ON TABLE public.orders TO test\_simple\_user; Задача 3

Используя SQL синтаксис - наполните таблицы следующими тестовыми данными:

аблица orders

| **Наименование** | **цена** |
| --- | --- |
| Шоколад | 10 |
| Принтер | 3000 |
| Книга | 500 |
| Монитор | 7000 |
| Гитара | 4000 |

Таблица clients

| **ФИО** | **Страна проживания** |
| --- | --- |
| Иванов Иван Иванович | USA |
| Петров Петр Петрович | Canada |
| Иоганн Себастьян Бах | Japan |
| Ронни Джеймс Дио | Russia |
| Ritchie Blackmore | Russia |

Используя SQL синтаксис:

* вычислите количество записей для каждой таблицы
* приведите в ответе:
  + запросы
  + результаты их выполнения.

INSERT into orders (name, price) values ('chokolate', '10'); #INSERT INTO добавить в табличку orders в столбцы (name, price) значения ('chokolate', '10');

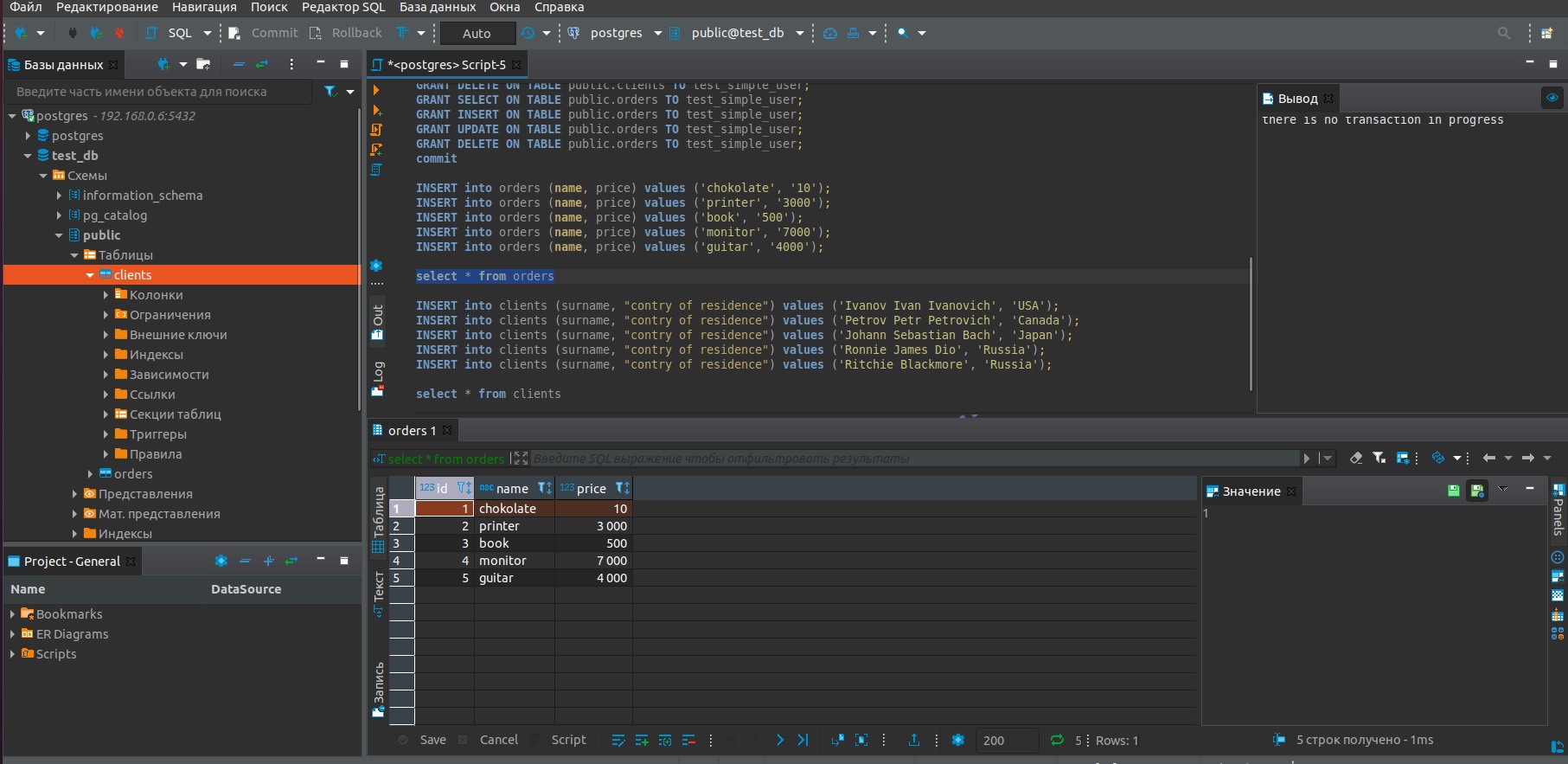
INSERT into orders (name, price) values ('printer', '3000');

INSERT into orders (name, price) values ('book', '500');

INSERT into orders (name, price) values ('monitor', '7000');

INSERT into orders (name, price) values ('guitar', '4000');

См. скрин “select - from orders Домашнее задание к занятию 6.2. SQL.jpg”

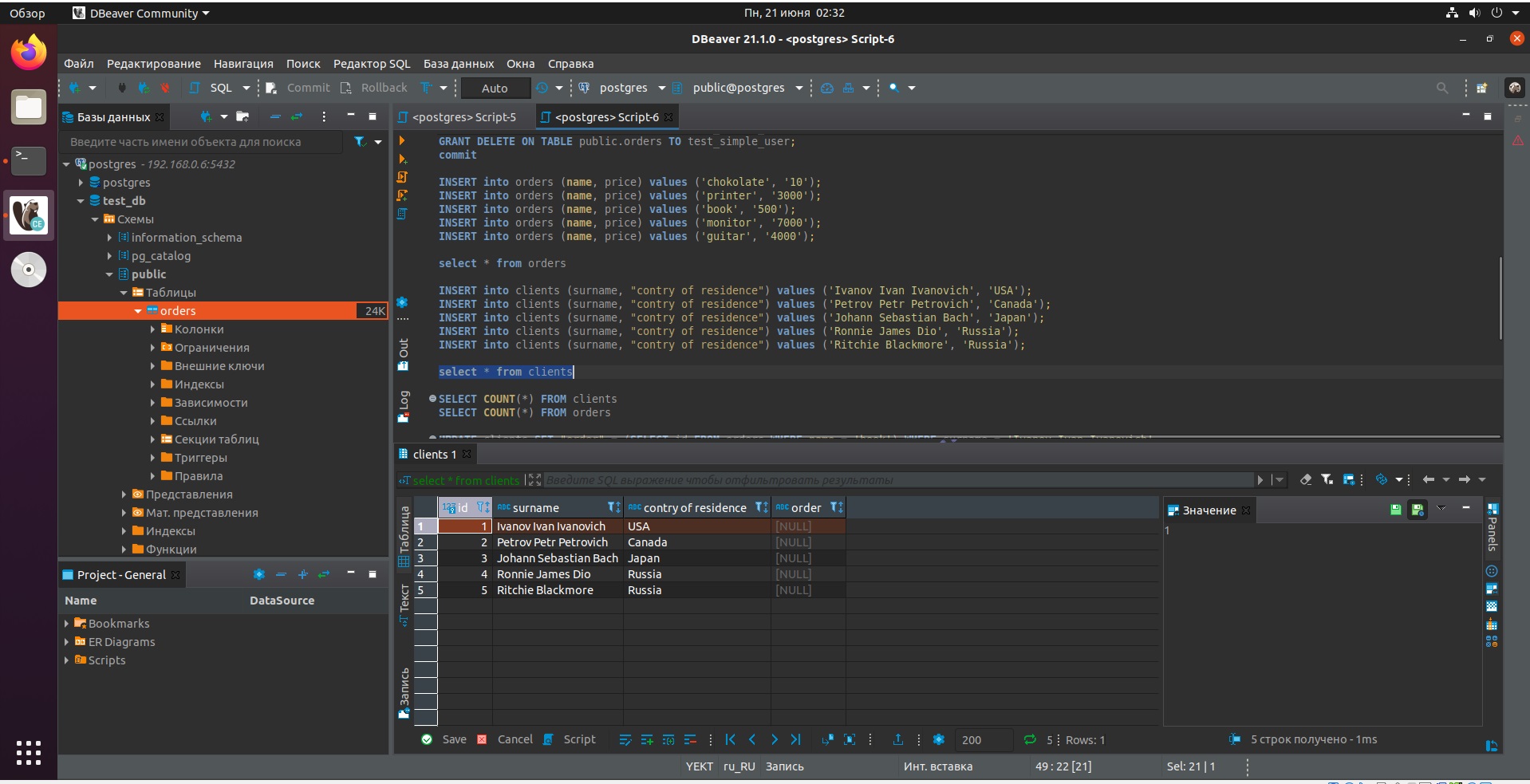


select \* from orders # вывести табличку orders

SELECT COUNT(\*) FROM clients # вывести количество записей в табличку через where можно написать условие

SELECT COUNT(\*) FROM orders

См. скрин “select - from clients Домашнее задание к занятию 6.2. SQL.jpg”



INSERT into clients (surname, "contry of residence") values ('Ivanov Ivan Ivanovich', 'USA');

INSERT into clients (surname, "contry of residence") values ('Petrov Petr Petrovich', 'Canada');

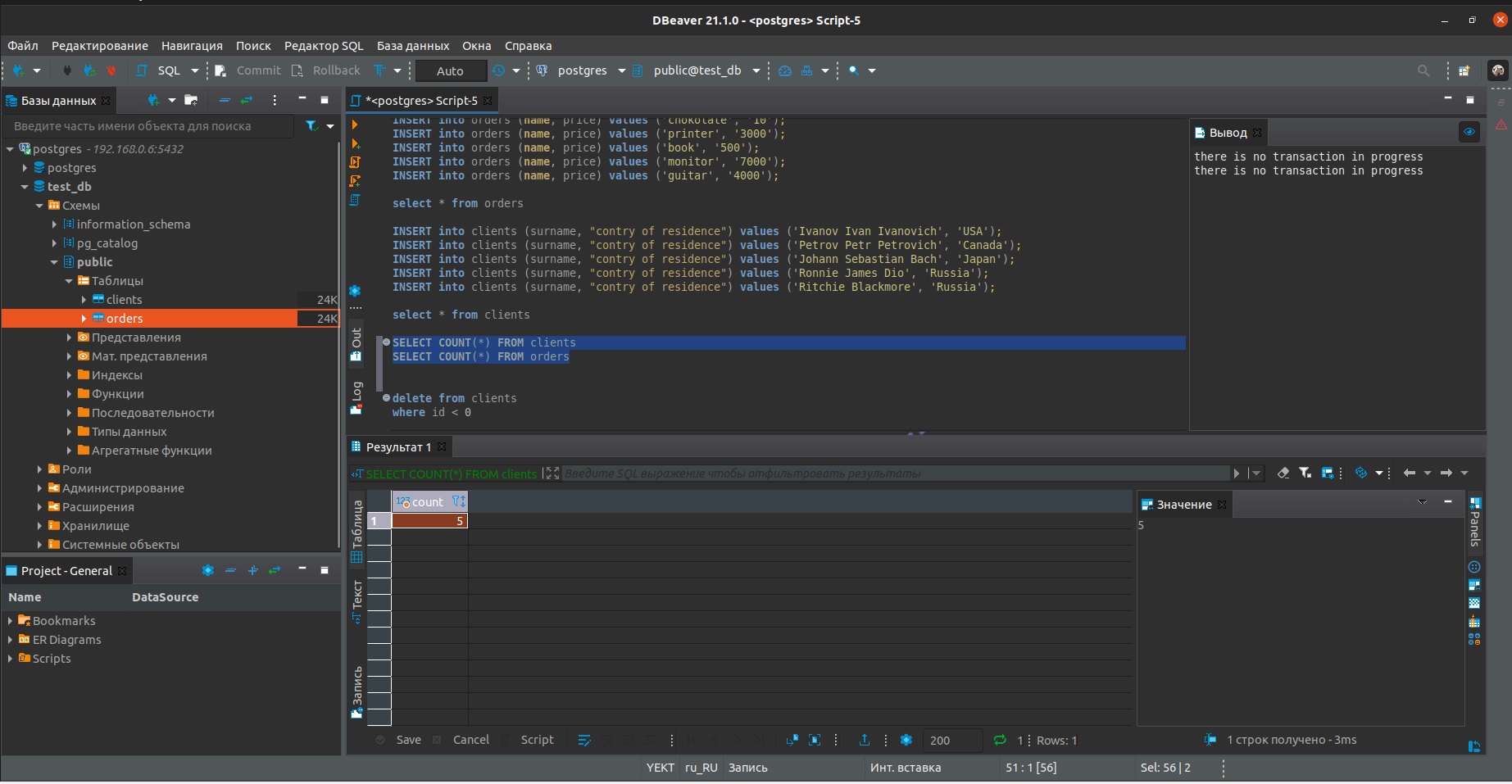
INSERT into clients (surname, "contry of residence") values ('Johann Sebastian Bach', 'Japan');

INSERT into clients (surname, "contry of residence") values ('Ronnie James Dio', 'Russia');

INSERT into clients (surname, "contry of residence") values ('Ritchie Blackmore', 'Russia');

select \* from clients

См. скрин “count количество записей Домашнее задание к занятию 6.2. SQL.jpg”



**Задача 4**

асть пользователей из таблицы clients решили оформить заказы из таблицы orders.

Используя foreign keys свяжите записи из таблиц, согласно таблице:

| **ФИО** | **Заказ** |
| --- | --- |
| Иванов Иван Иванович | Книга |
| Петров Петр Петрович | Монитор |
| Иоганн Себастьян Бах | Гитара |

Приведите SQL-запросы для выполнения данных операций.

Приведите SQL-запрос для выдачи всех пользователей, которые совершили заказ, а также вывод данного запроса.

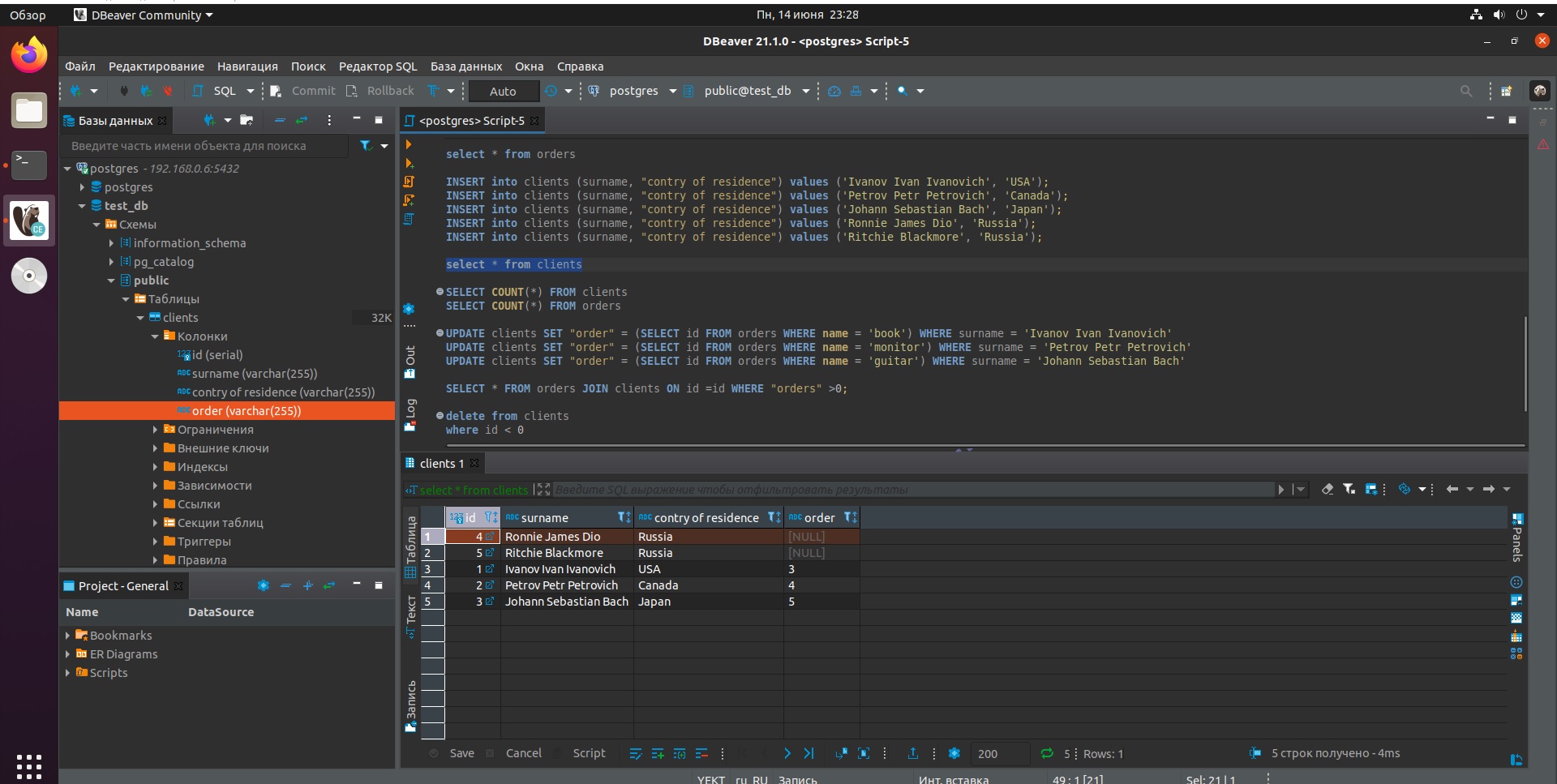
Подсказк - используйте директиву UPDATE.

UPDATE clients SET "order" = (SELECT name FROM orders WHERE name = 'book') WHERE surname = 'Ivanov Ivan Ivanovich' # обнови в табличке clients столбец order, используя значение name из таблички orders и вставь where значение только где используется имя name = ‘book’

UPDATE clients SET "order" = (SELECT name FROM orders WHERE name = 'monitor') WHERE surname = 'Petrov Petr Petrovich'

UPDATE clients SET "order" = (SELECT name FROM orders WHERE name = 'guitar') WHERE surname = 'Johann Sebastian Bach'

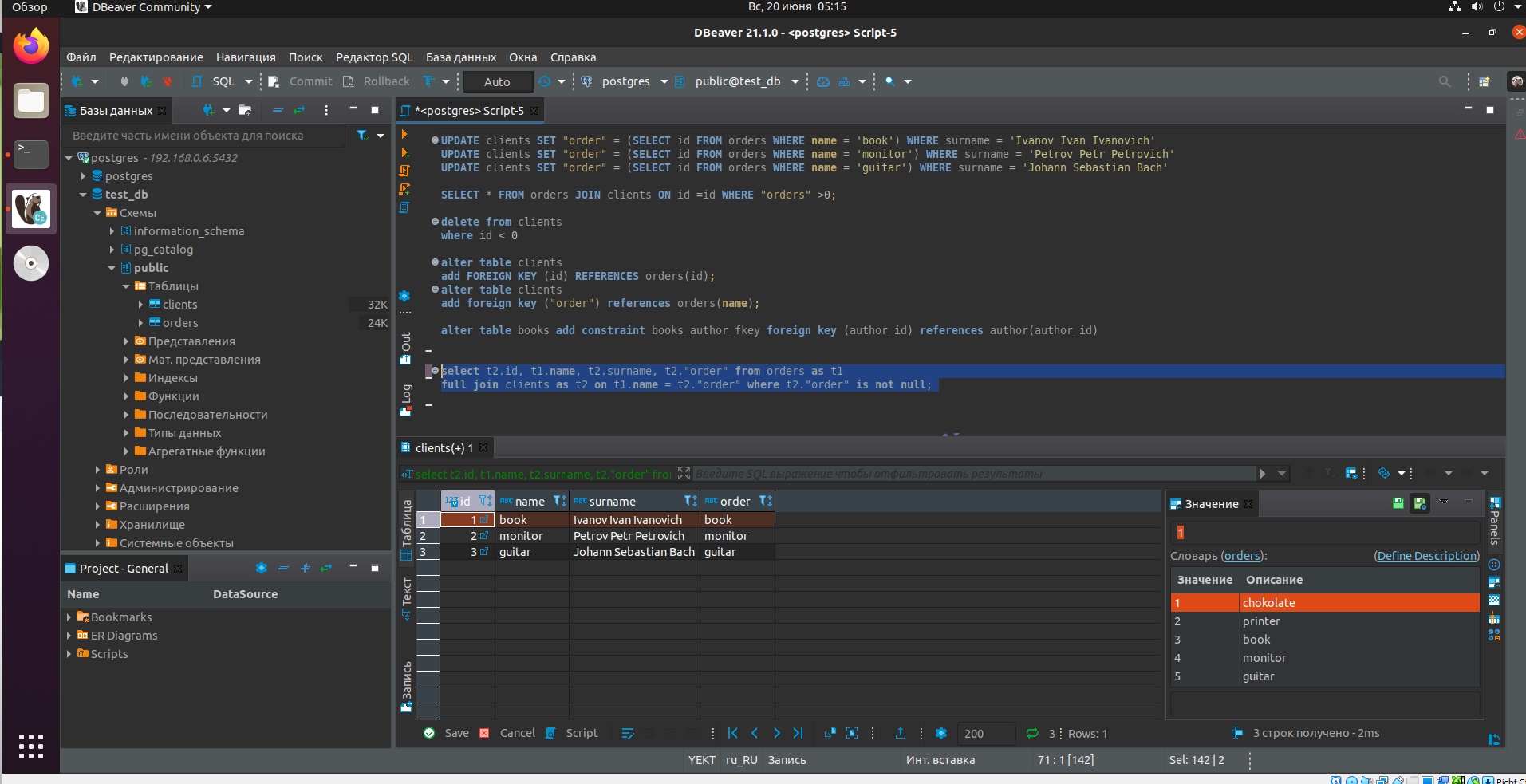
См. скрин “связать таблички Домашнее задание к занятию 6.2. SQL.jpg”



select t2.id, t1.name, t2.surname, t2."order" from orders as t1

full join clients as t2 on t1.name = t2."order" where t2."order" is not null;   
  
 # выведи из таблички clients id, из таблички orders name, surname, “order” столбцы, где from orders считай как алиас as t1, присоедини все зачения full join и выбери значения из столбцов t1.name и t2."order" не равные нулю (пустому значению).

См. скрин “4 вопрос выведи столбцы из табличек где в столбце order не равно нулю Домашнее задание к занятию 6.2. SQL.jpg”



explain select t2.id, t1.name, t2.surname, t2."order" from orders as t1

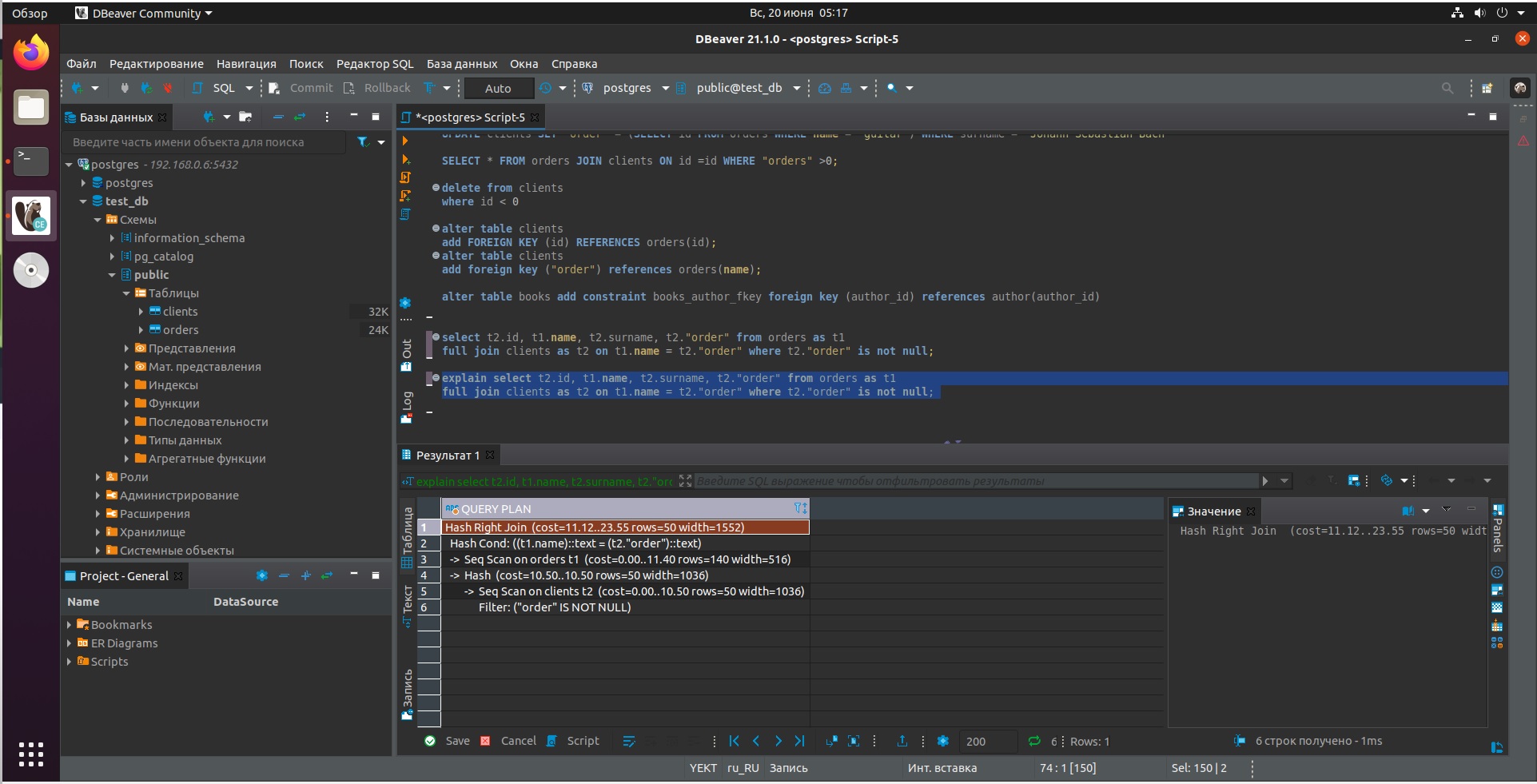
full join clients as t2 on t1.name = t2."order" where t2."order" is not null;

**Задача 5**

Получите полную информацию по выполнению запроса выдачи всех пользователей из задачи 4 (используя директиву EXPLAIN).

Приведите получившийся результат и объясните что значат полученные значения.

см. скрин “explain задача 5.jpg”



Cost=11.12..23.55 # стоимость от 11.12 до 23.55 милисекунд, т.е. он потратил 11.12 а мог бы потратить 23.55 если бы пошел по другим сценариям.

Rows # Ожидаемое число строк, которое должен вывести этот узел плана. При этом так же предполагается, что узел выполняется до конца.

width # Ожидаемый средний размер строк, выводимых этим узлом плана (в байтах).

Взято Rows и width с ссылки https://www.postgresql.org/docs/current/using-explain.html

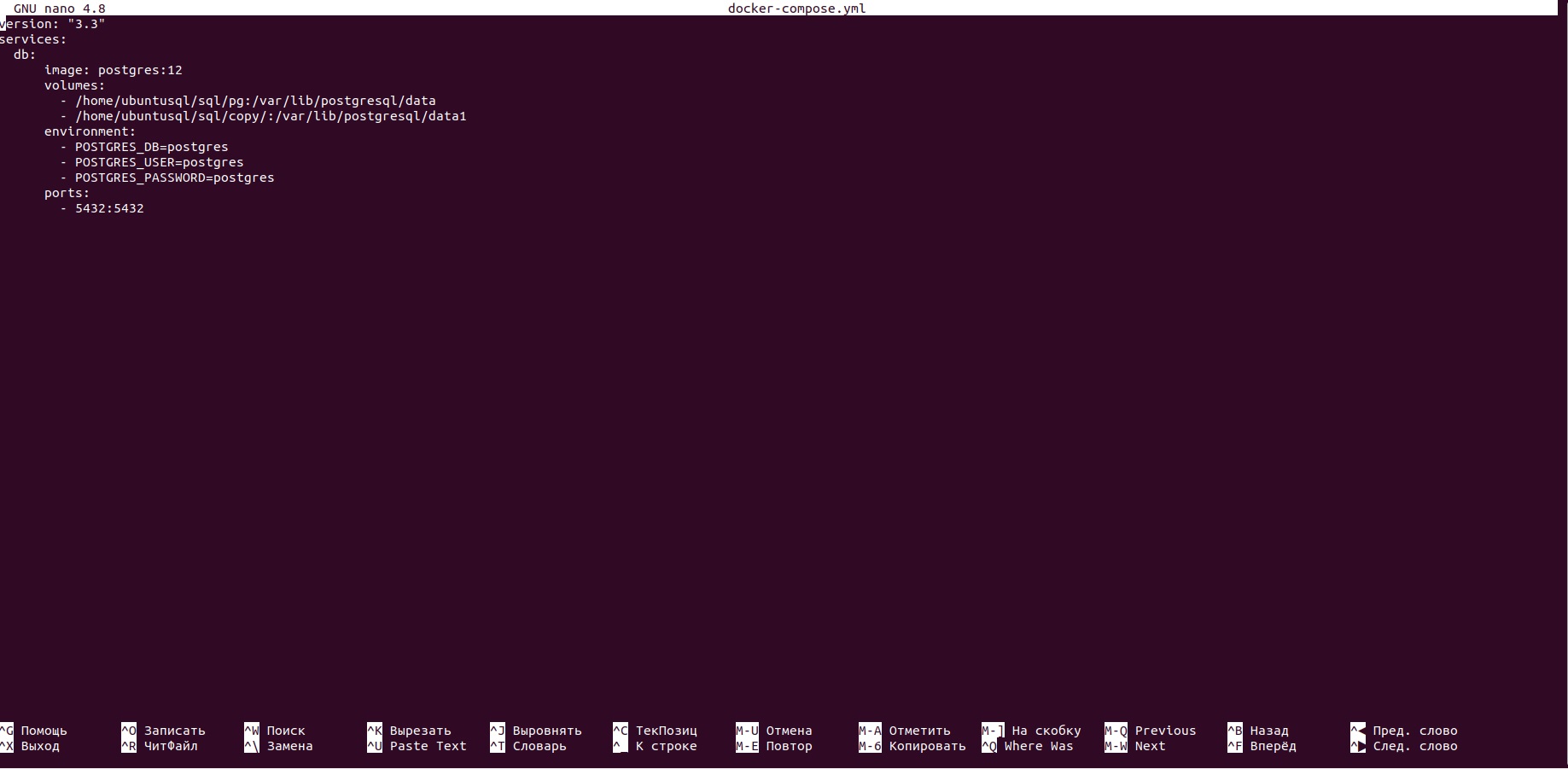
Seq scan on # означает что команда разбита на два селекта, т.е. на табличку orders и clients сделал сканирование и при помощи hash (хешей) он все это объединил

И использует фильтр вывести значения Not null, стоимость запроса из за фильтра немного выросла.

6 задание.

Поднимаем контейнер с docker volume

См. скрин “docker compouse с 2 volume Домашнее задание к занятию 6.2. SQL .jpg”



services:

db:

image: postgres:12

volumes:

- /home/ubuntusql/sql/pg:/var/lib/postgresql/data

- /home/ubuntusql/sql/copy/:/var/lib/postgresql/data1

environment:

- POSTGRES\_DB=postgres

- POSTGRES\_USER=postgres

- POSTGRES\_PASSWORD=postgres

ports:

- 5432:5432

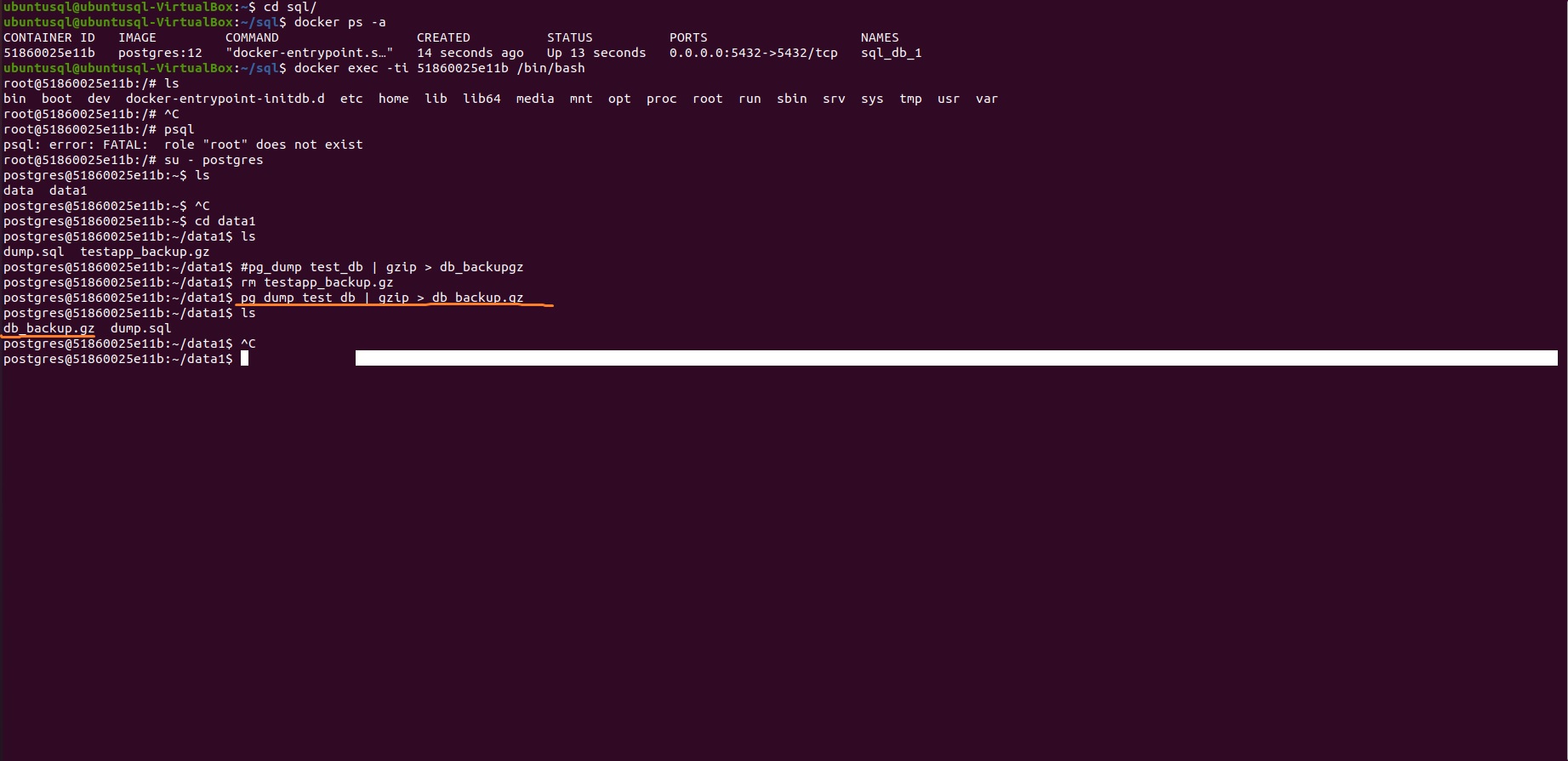
ubuntusql@ubuntusql-VirtualBox:~/sql$ docker ps –a # ищем наш контейнер

docker exec -ti 51860025e11b /bin/bash # залетаем в контейнер

root@51860025e11b:/# su – postgres # заходим по профилем postgres

postgres@51860025e11b:~$ cd data1 # залетаем в volume copy  
  
pg\_dump test\_db | gzip > db\_backup.gz # создаем утилитой pg\_dump копию базы данных test\_db | пакуем наш файл под именем db\_backup.gz

см. скрин “создаем бэкап Домашнее задание к занятию 6.2. SQL.jpg”



Дальше стопаем все контейнеры и удаляем  
ubuntusql@ubuntusql-VirtualBox:~/sql$ docker stop $(docker ps -aq)

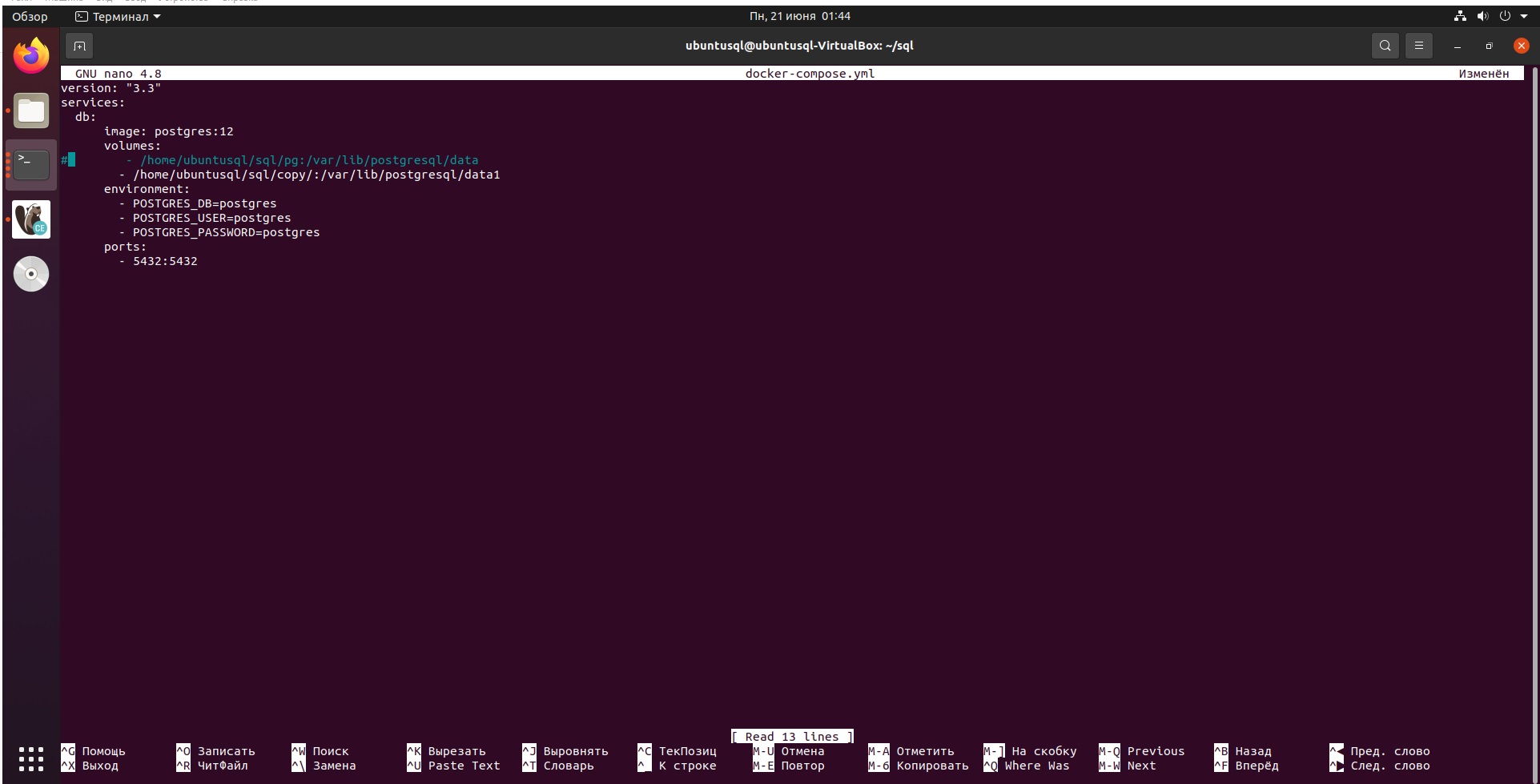
51860025e11b

ubuntusql@ubuntusql-VirtualBox:~/sql$ docker rm $(docker ps -aq)

51860025e11b

Убираем в docker-compose.yml вольюм основной, оставляем только для бэкапов

См. скрин “отрубаем вольюм основной чтобы проверить бэкап Домашнее задание к занятию 6.2. SQL.jpg”



version: "3.3"

services:

db:

image: postgres:12

volumes:

# - /home/ubuntusql/sql/pg:/var/lib/postgresql/data

- /home/ubuntusql/sql/copy/:/var/lib/postgresql/data1

environment:

- POSTGRES\_DB=postgres

- POSTGRES\_USER=postgres

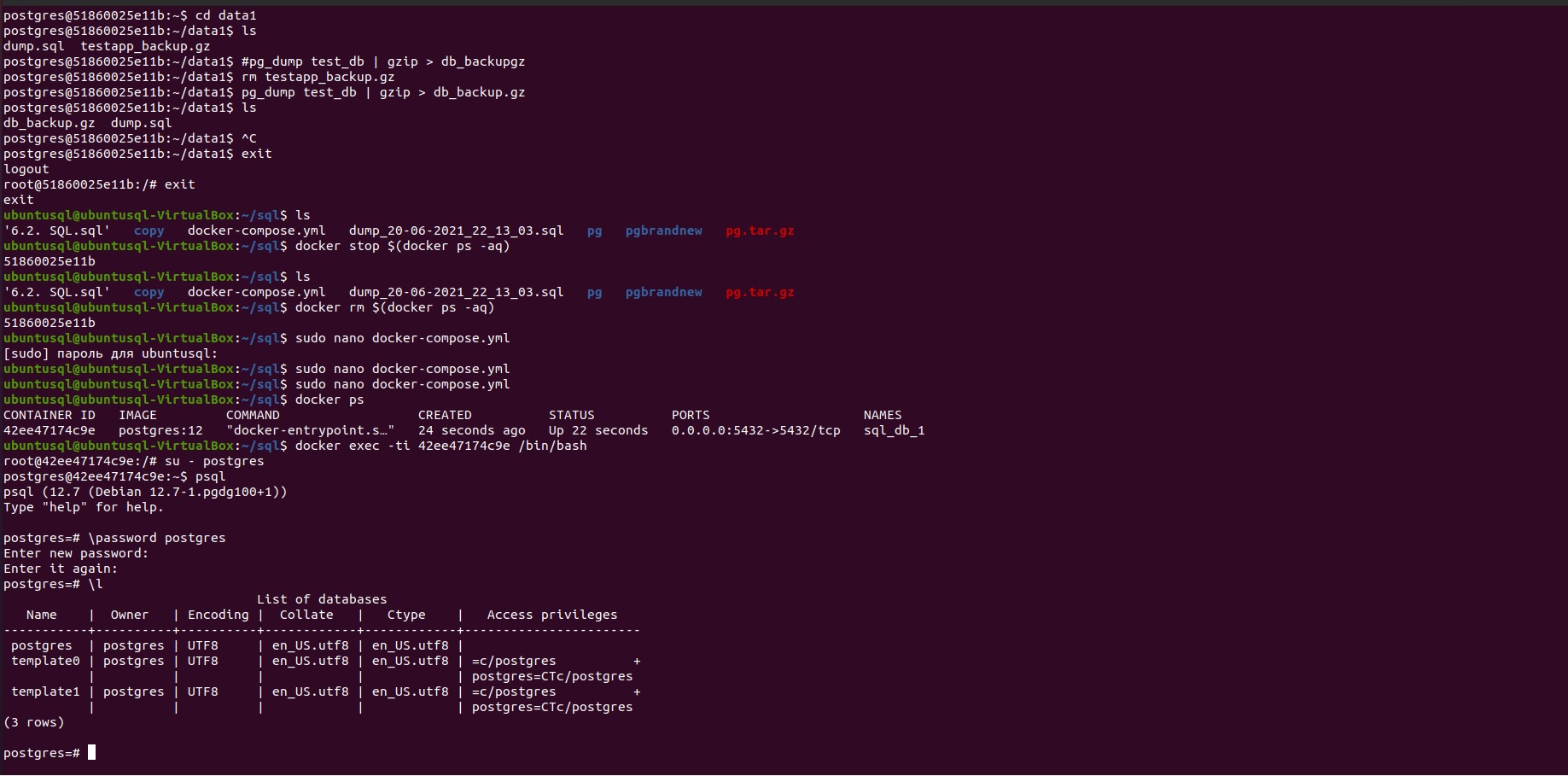
- POSTGRES\_PASSWORD=postgres

ports:

- 5432:5432

Переподнимаем postgre и убеждаемся что лишних табличек нет

См. скрин “переподнимаем postgre и убеждаемся что табличек лишних нет Домашнее задание к занятию 6.2. SQL.jpg”



postgres@42ee47174c9e:~/data1$ psql -c "create database test\_db;" # утилитой psql восстановить –c “create database test\_db;” ## восстанавливаем test\_db;

postgres@42ee47174c9e:~/data1$ gunzip -c db\_backup.gz | psql test\_db; # распаковать с восстановлением файл db\_backup.gz и за тем с помощью утилиты сделать restore БД test\_db данными.

(для себя) Так же показан способ восстановить все данные

https://www.youtube.com/watch?v=RDBia4-kHoM&t=16s&ab\_channel=RubyTreeSoftware%2CInc.