Национальный исследовательский ядерный университет «МИФИ»

(Московский Инженерно-Физический Институт)

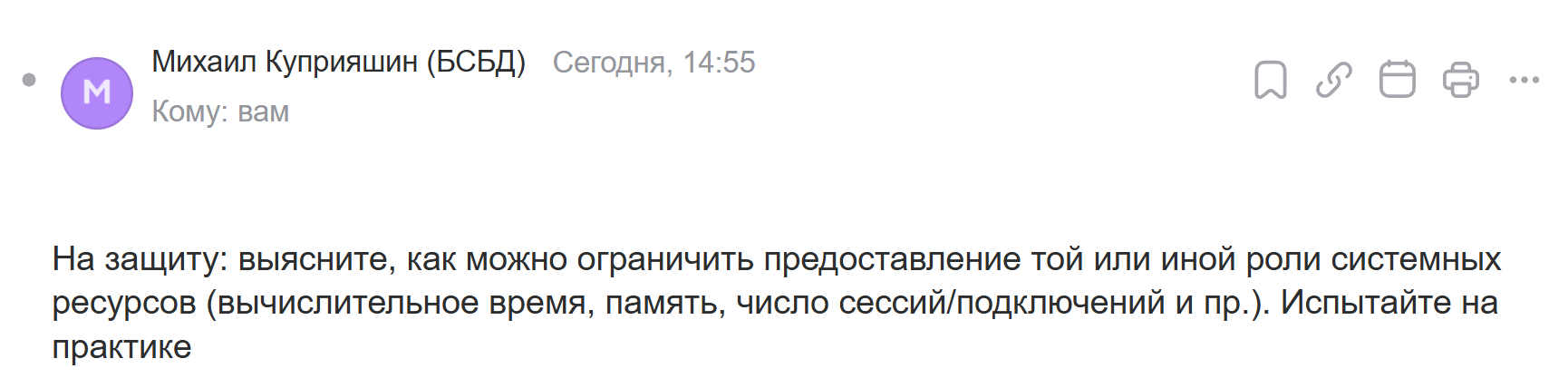
Кафедра №42 «Криптология и кибербезопасность»

**На защиту лабораторной работы №2-1**

**«**Пользователи. Роли. Привилегии**»**

Тимин Александр Б21-515 (2024г.)

**1 Ограничение системных ресурсов для роли**

****

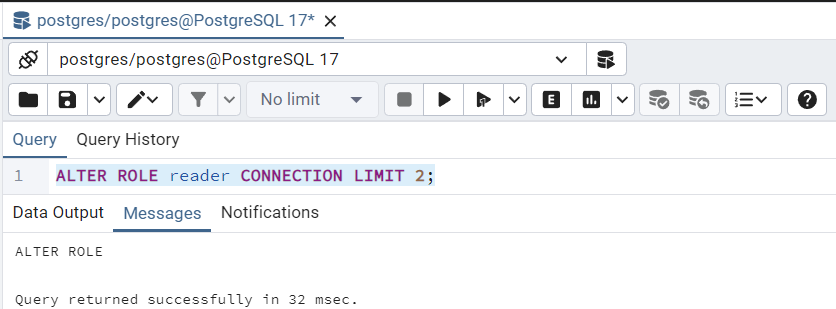
В PostgreSQL можно ограничить использование системных ресурсов для ролей с помощью различных параметров. Это позволяет управлять вычислительными ресурсами, количеством подключений, временем выполнения запросов и другими характеристиками.

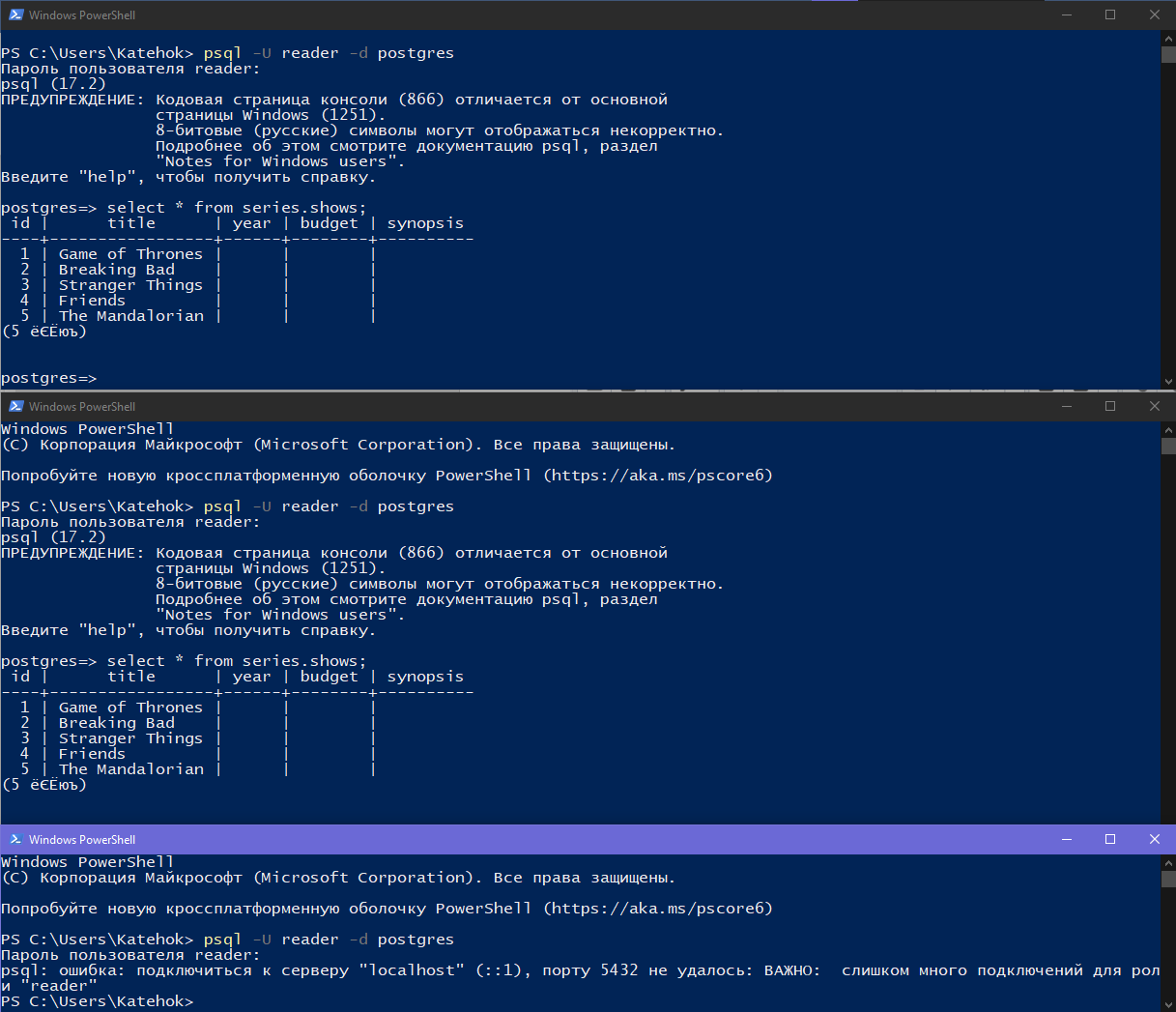
**1.1 Ограничение числа подключений**

Параметр CONNECTION LIMIT ограничивает количество одновременных подключений для роли.

Установим лимит в 2 сессии для роли reader и попробуем превысить его.

ALTER ROLE reader CONNECTION LIMIT 2;

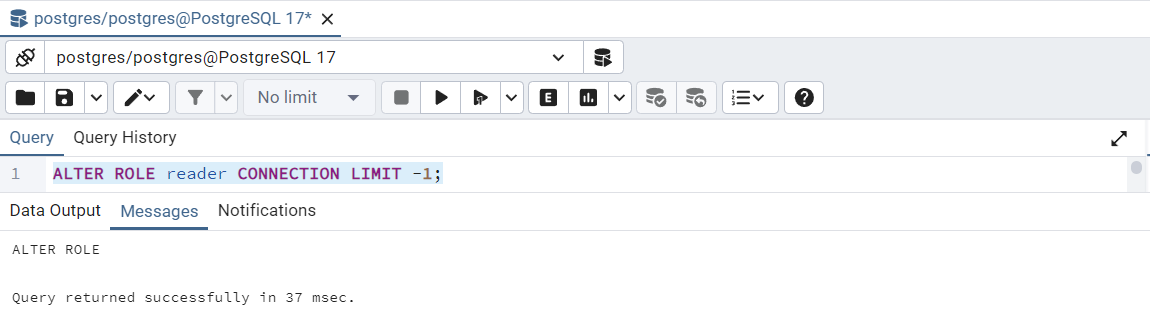


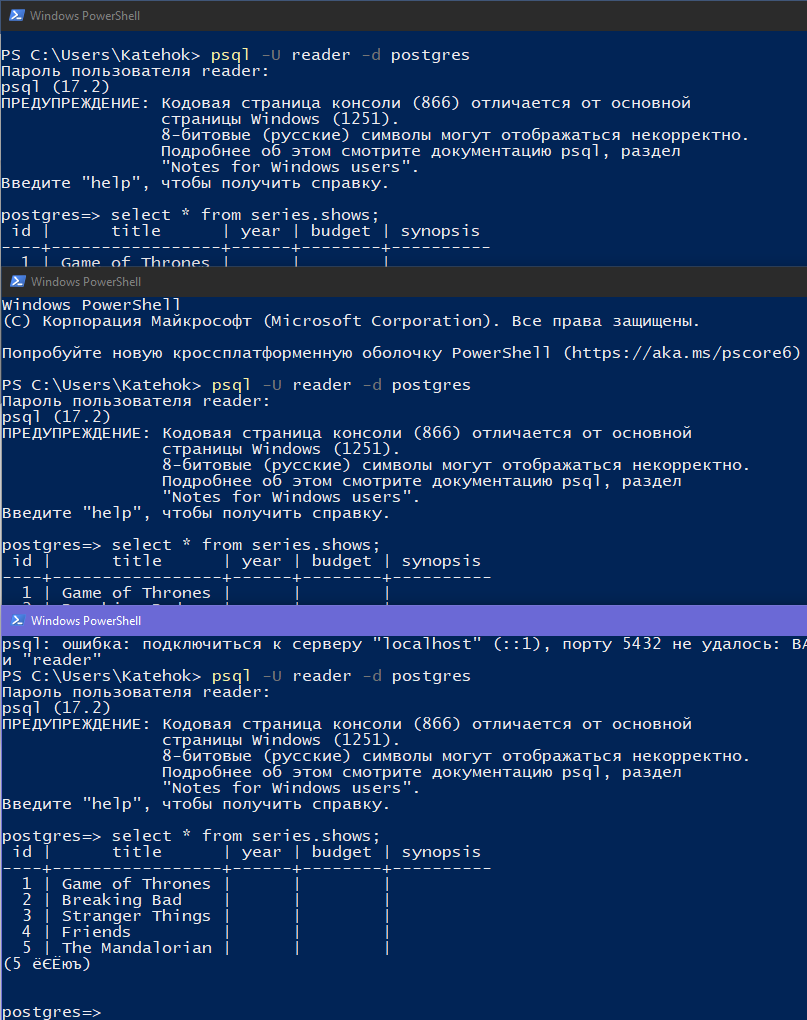


Как видно из скриншота, открыть 3 сессию не удалось.

Сбросим лимит.

ALTER ROLE reader CONNECTION LIMIT -1;

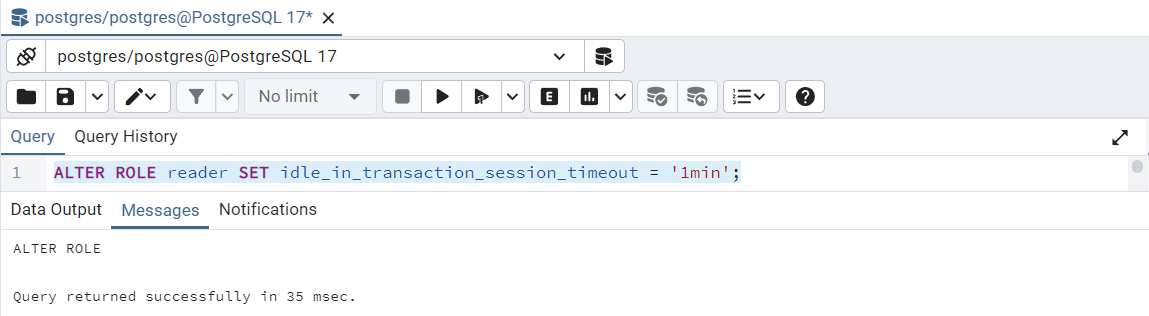


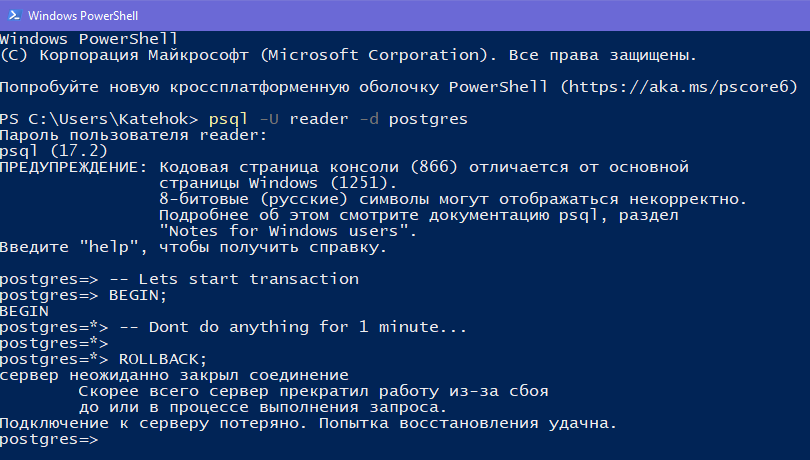
Теперь 3 сессия успешно создалась.

**1.2 Ограничение времени выполнения запроса**

Параметр idle\_in\_transaction\_session\_timeout ограничивает время бездействия внутри транзакции.

ALTER ROLE reader SET idle\_in\_transaction\_session\_timeout = '1min';





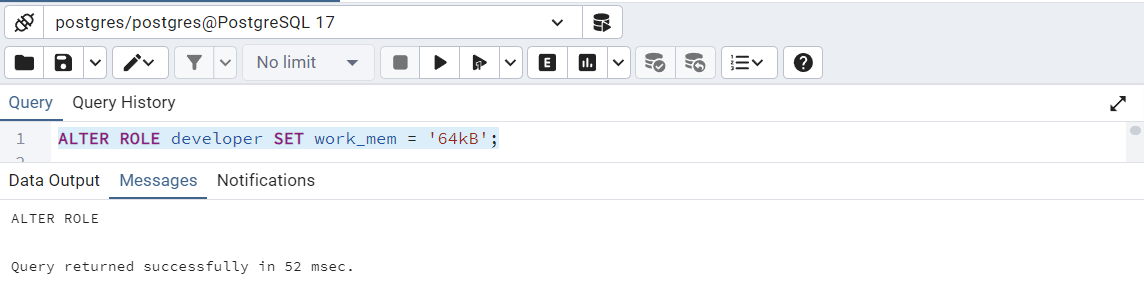
Как видно из скриншота, при попытке выполнить запрос спустя минуту бездействия, мы получили ошибку, связанную с тем, что сервер просто закрыл соединение.

**1.3 Ограничение использования памяти и процессора**

Через pg\_settings можно управлять параметрами work\_mem, maintenance\_work\_mem, cpu\_operator\_cost и др.

Установим (минимально допустимый) лимит на использование оперативной памяти.

ALTER ROLE developer SET work\_mem = '64kB';



Добавим данных.

DO $$

DECLARE

i INTEGER;

BEGIN

FOR i IN 1..2000 LOOP

INSERT INTO series.shows (title, year, budget, synopsis)

VALUES (

'Show ' || i, -- Уникальное название

2000 + (i % 20), -- Год (случайное значение в диапазоне)

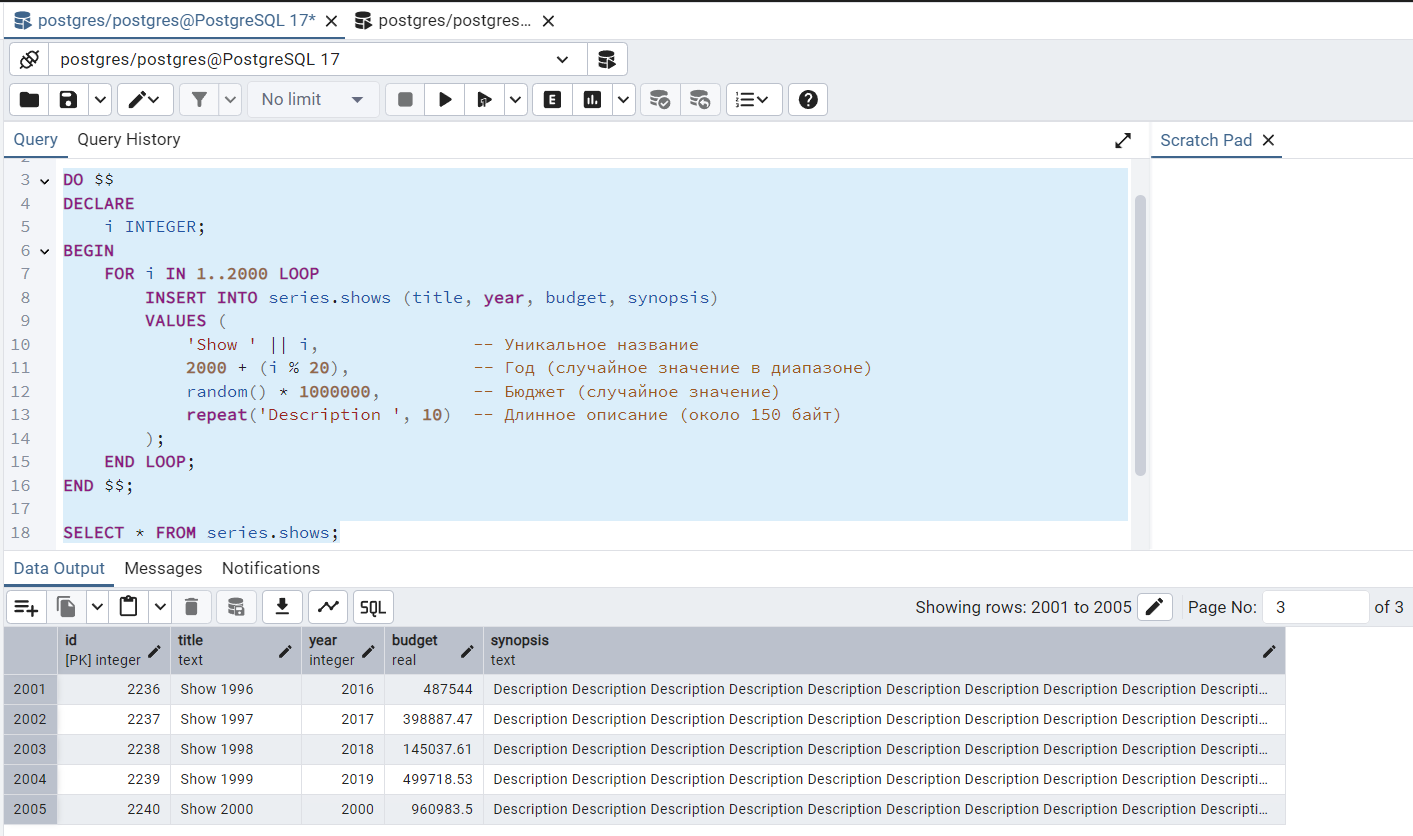
random() \* 1000000, -- Бюджет (случайное значение)

repeat('Description ', 10) -- Длинное описание (около 150 байт)

);

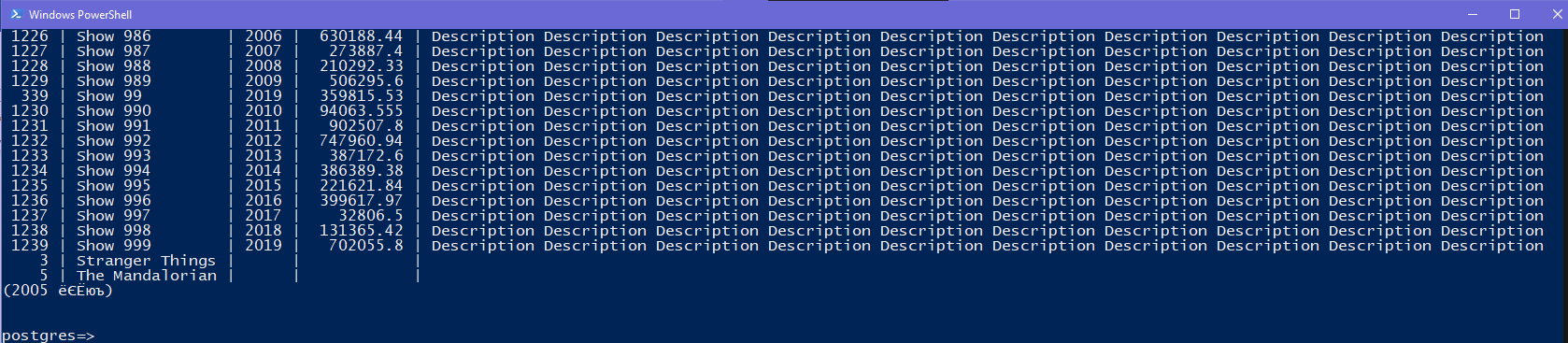
END LOOP;

END $$;



Подключимся от имени developer и попытаемся выполнить запрос с сортировкой.

SELECT \* FROM shows ORDER BY title;



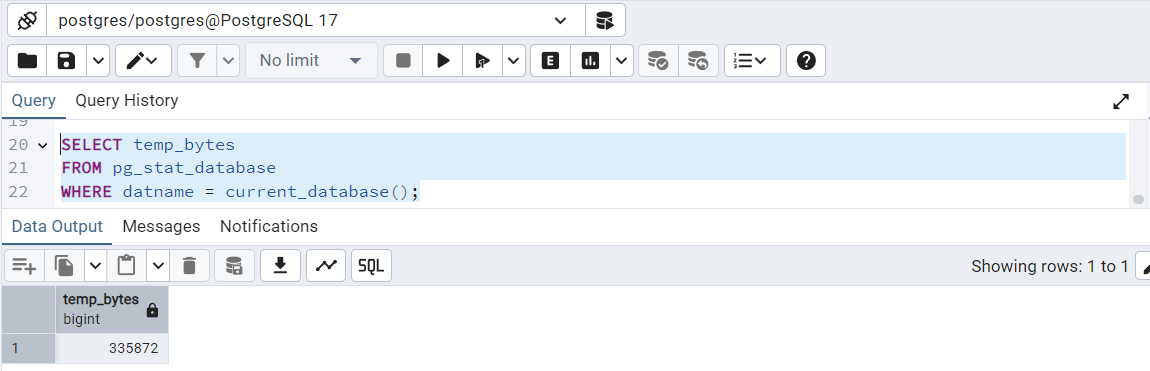
Как видно из рисунка, операция завершилась успешно.

Когда постгрес достигает лимита по оперативке, он создает временные файлы. Проверим их наличие.

SELECT temp\_bytes

FROM pg\_stat\_database

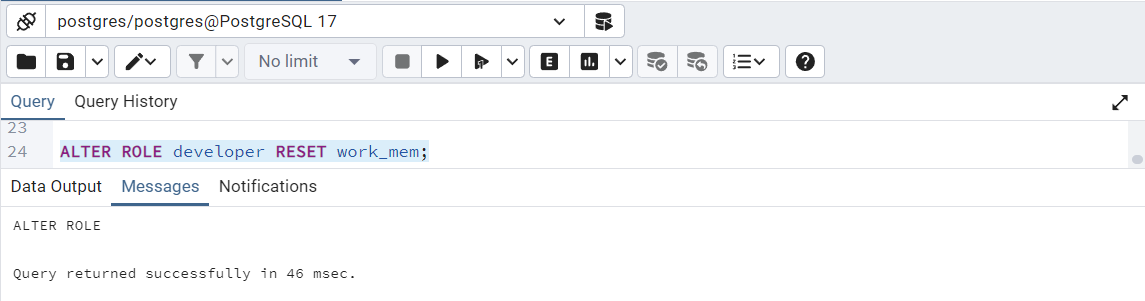
WHERE datname = current\_database();



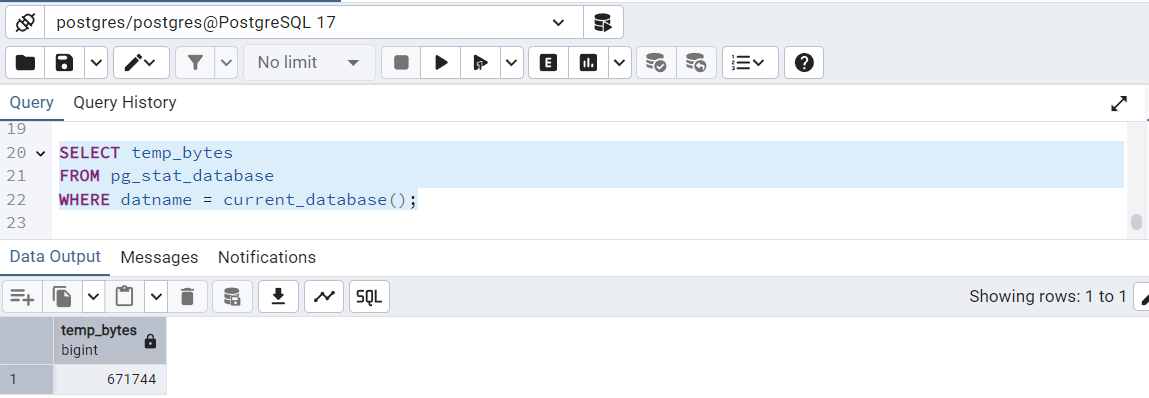
Как видно из рисунка, было создано временных файлов на ~335kB.

Сбросим лимит для developer.

ALTER ROLE developer RESET work\_mem;

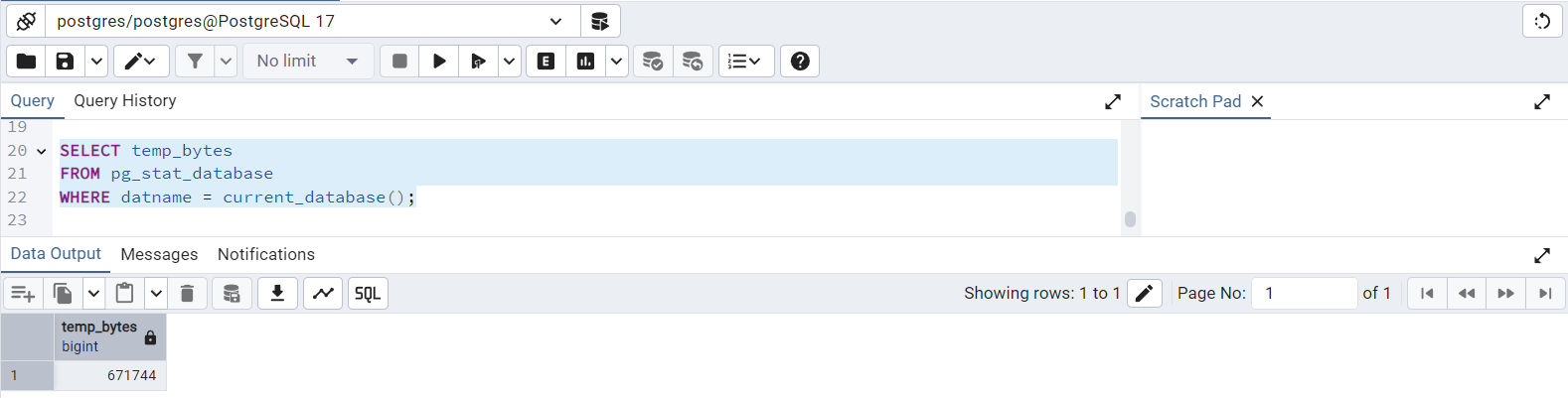


Снова была выполнена сортировка ролью developer, после чего снова проверены временные файлы.



Число, так как я не сбросил сессию для developer.

Я открыл новую сессию для developer, выполнил сортировку и снова вывел инфу о временных файлах.



В сравнении с последним выводом число не изменилось, а значит постгрес не создал дополнительно временных файлов (то есть у него хватило оперативки).

Приложение (из отчета о лр 2-1)

* [директория последней лабы из репозитория предыдущего семестра](https://github.com/KATEHOK/DBS_labs-2024/tree/main/4);
* [ERD структуры таблиц](https://github.com/KATEHOK/DBS-labs-7/blob/main/1/pics/ERD.png);
* [отчет (docx)](https://github.com/KATEHOK/DBS-labs-7/blob/main/1/report.docx);
* [отчет (pdf)](https://github.com/KATEHOK/DBS-labs-7/blob/main/1/report.pdf).