

***Федеральное агентство по рыболовству***

***Федеральное государственное бюджетное образовательное***

***учреждение высшего образования***

***«Астраханский государственный технический университет»***

**Система менеджмента качества в области образования, воспитания, науки и инноваций сертифицирована**

**ООО «ДКС РУС» по международному стандарту ISO 9001:2015**

|  |  |
| --- | --- |
| Институт | Информационных технологий и коммуникаций |
| Направление | 09.03.01 Информатика и вычислительная техника |
| Профиль | «Автоматизированные системы обработки информации и управления» |
| Кафедра | «Автоматизированные системы обработки информации и управления» |

**Лабораторная работа № 4**

**«Типы данных в СУБД PostgreSQL**»

по дисциплине «СУБД PostgreSQL»

|  |  |  |
| --- | --- | --- |
|  |  | Работа выполнена студентом группы ДИНРБ-31  Кузургалиев Р.А. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (Фамилия И.О.) подпись |
|  |  | Проверил работу:  ст. преподаватель Мамлеева А.Р,\_\_\_\_\_\_\_\_\_\_\_\_  (ученая степень, ученое звание, Фамилия И.О.) |

Работа защищена

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 2024 г.

**АСТРАХАНЬ – 2024**

**СОДЕРЖАНИЕ ОТЧЕТА**

**Цель:** Получить практические навыки в использовании различных типов данных в СУБД PostgreSQL.

Выполнение задания:

**Задание 1**. Создайте таблицу, содержащую атрибут типа numeric(precision, scale).

Например, таблица, содержащая результаты измерений.

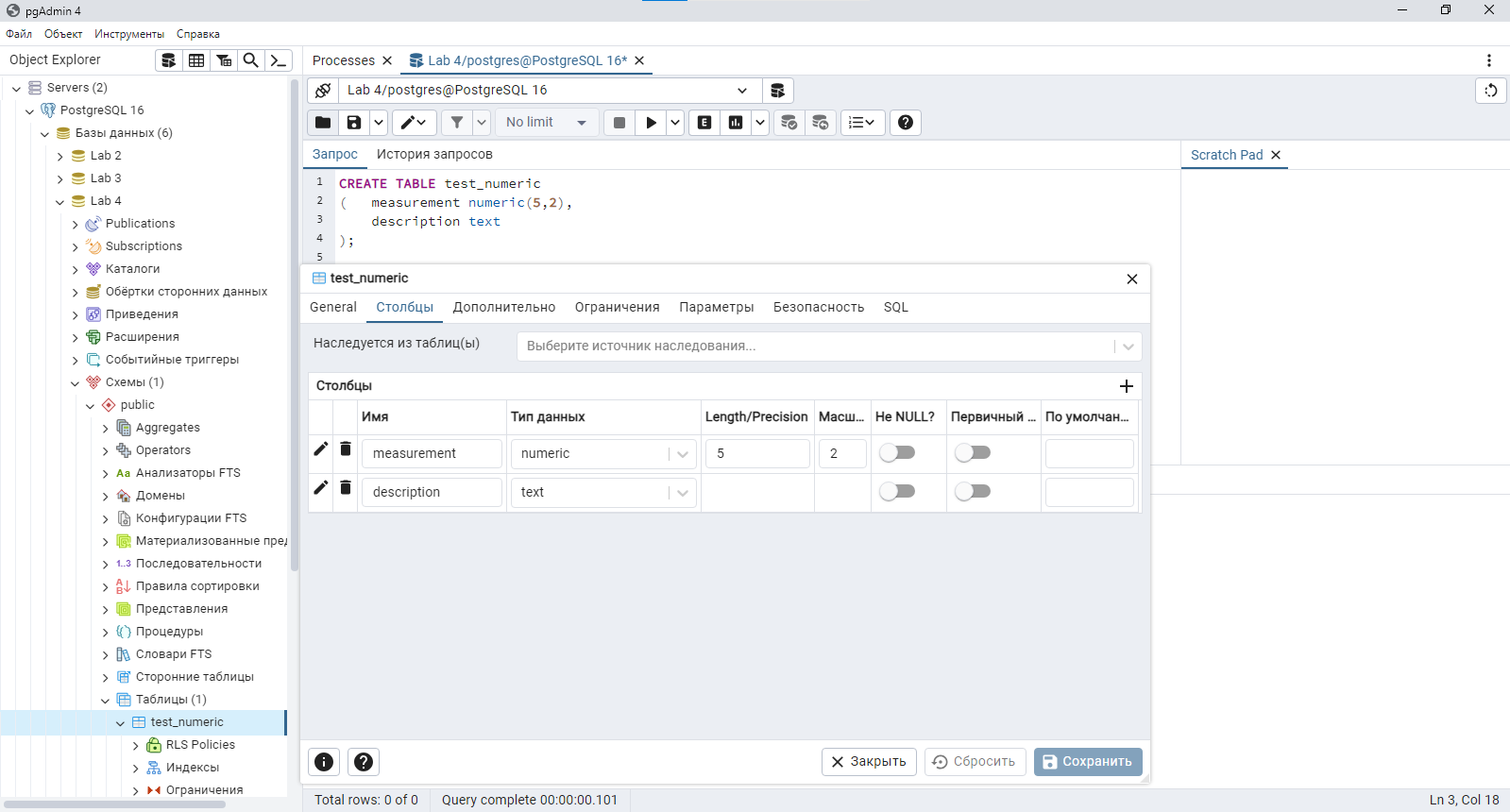
* 1. Создайте таблицу с помощью команды:

CREATE TABLE test\_ numeric

( measurement numeric(5,2),

description text

);



* 1. С помощью команды INSERT продемонстрируйте округление вводимого числа до то точности, которая задана при создании таблицы.

Какая из следующих команд вызовет ошибку и почему?

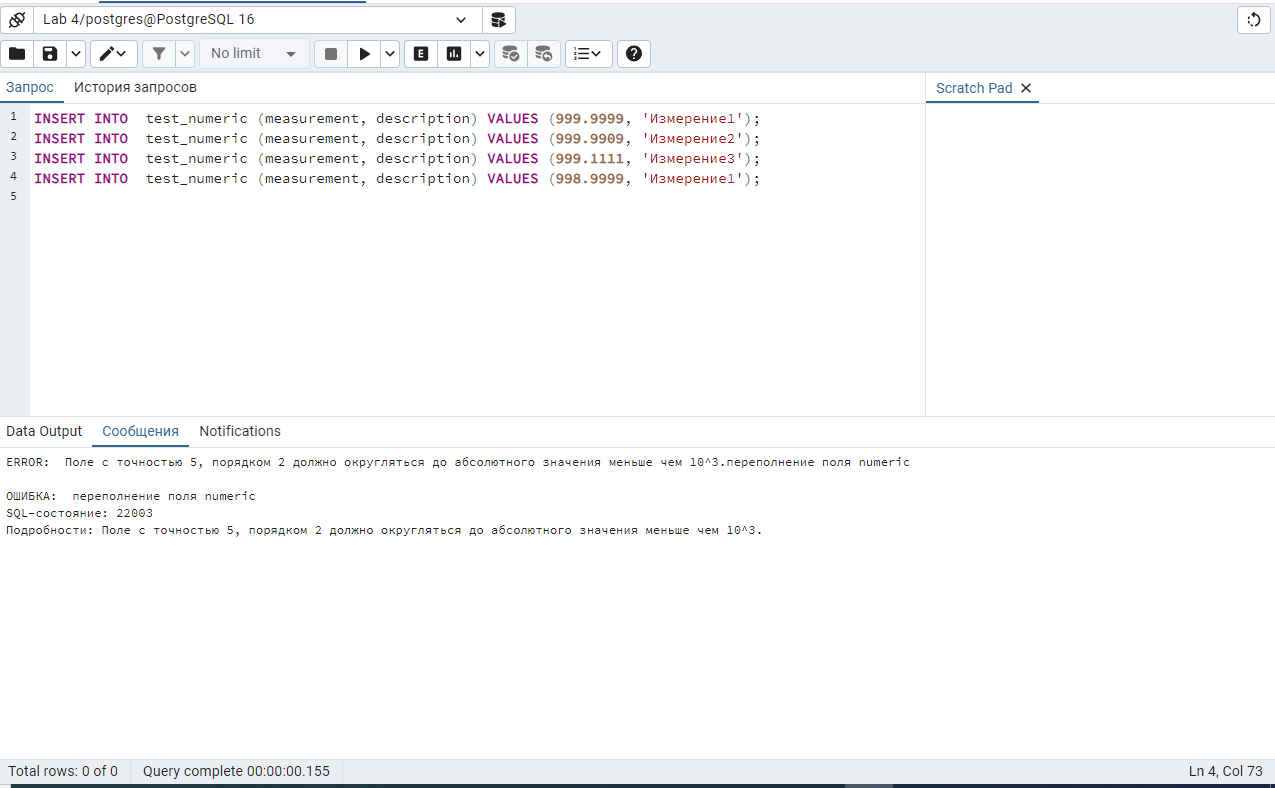
INSERT INTO test\_numeric VALUES (999.9999, ‘Измерение 1’);

INSERT INTO test\_numeric VALUES (999.9909, ‘Измерение 2’);

INSERT INTO test\_numeric VALUES (999.1111, ‘Измерение 3’);

INSERT INTO test\_numeric VALUES (998.9999, ‘Измерение 1’);

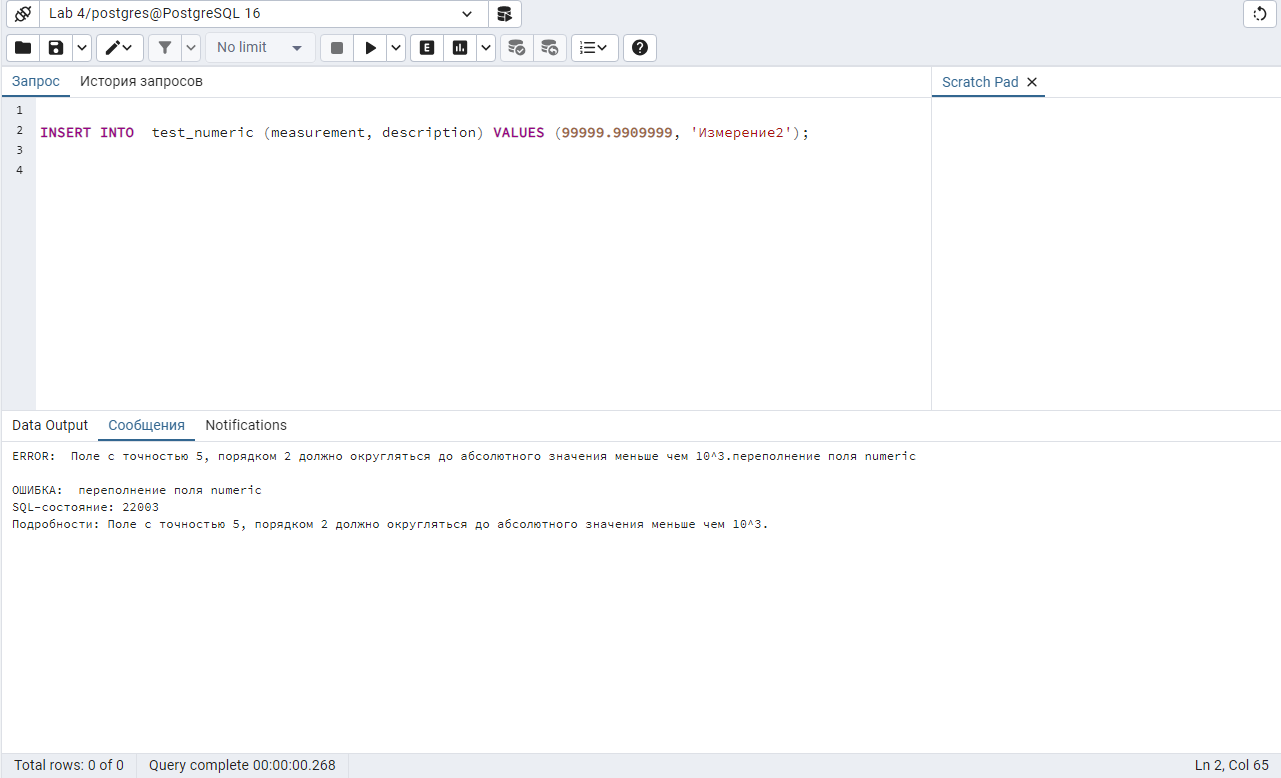
**Ошибка в первом запросе!!!**



* 1. Продемонстрируйте генерирование ошибки при попытке ввода числа, количество цифр в котором слева от десятичной точки (запятой) превышает допустимое.

***Примечание***: Если Вам уже не нужна таблица test\_ numeric, то ее можно удалить командой:

DROP TABLE test\_ numeric;

****

**Задание 2**. Предположим, что возникла необходимость хранить в одной столбце таблицы данные, представленные с различной точностью

Это могут быть, например, результаты физических измерений разнородных показателей или различные медицинские показатели здоровья пациентов (результаты анализов). В таком случае можно использовать тип numeric без указания масштаба и точности.

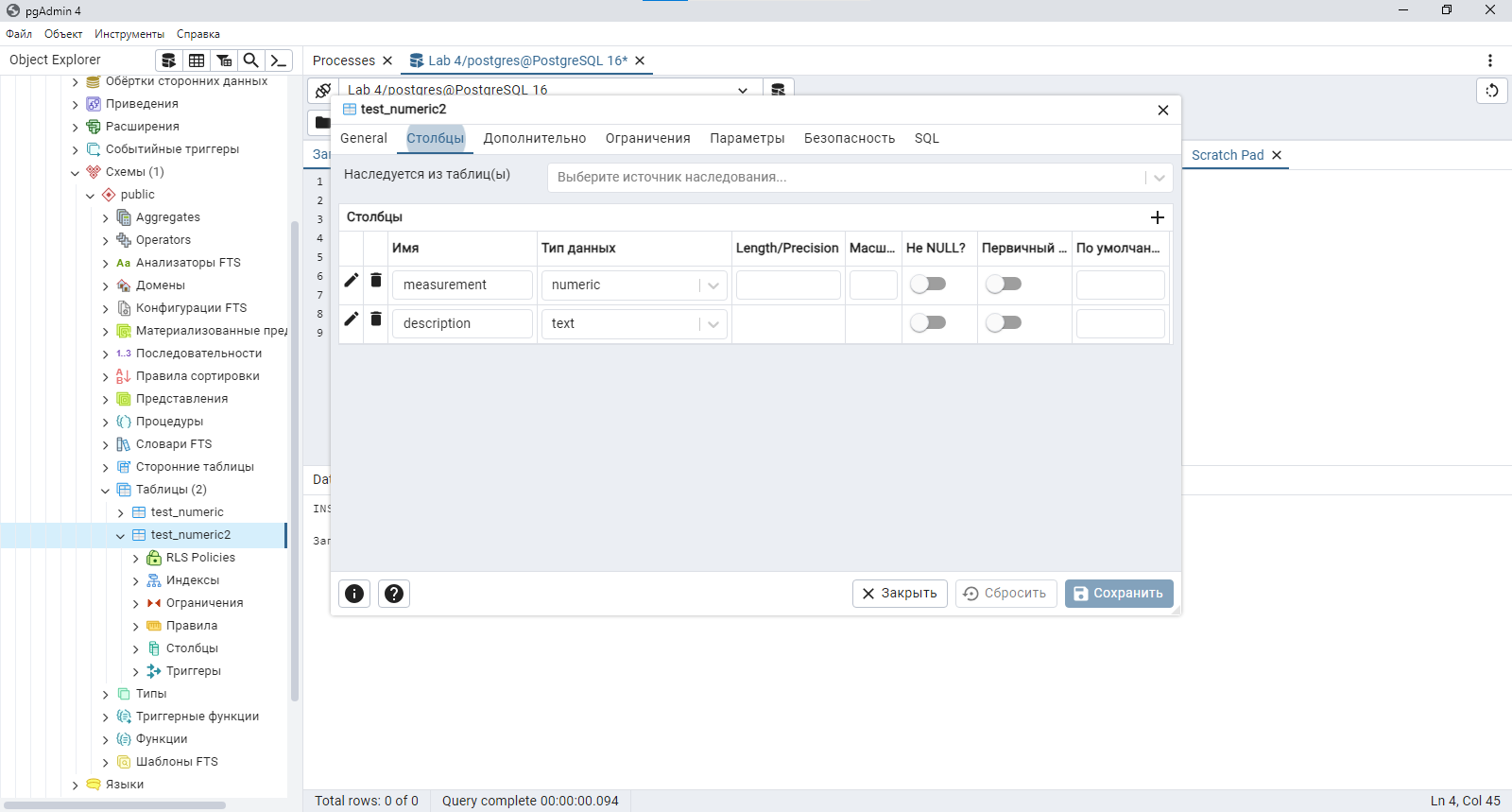
2.1. Создайте таблицу с помощью команды:

CREATE TABLE test\_ numeric2

( measurement numeric,

description text

);



2.2. Вставьте в таблицу test\_ numeric несколько строк:

INSERT INTO test\_numeric2

VALUES (1234567890.0987654321, ‘Точность 20 знаков, масштаб 10 знаков’);

INSERT INTO test\_numeric2

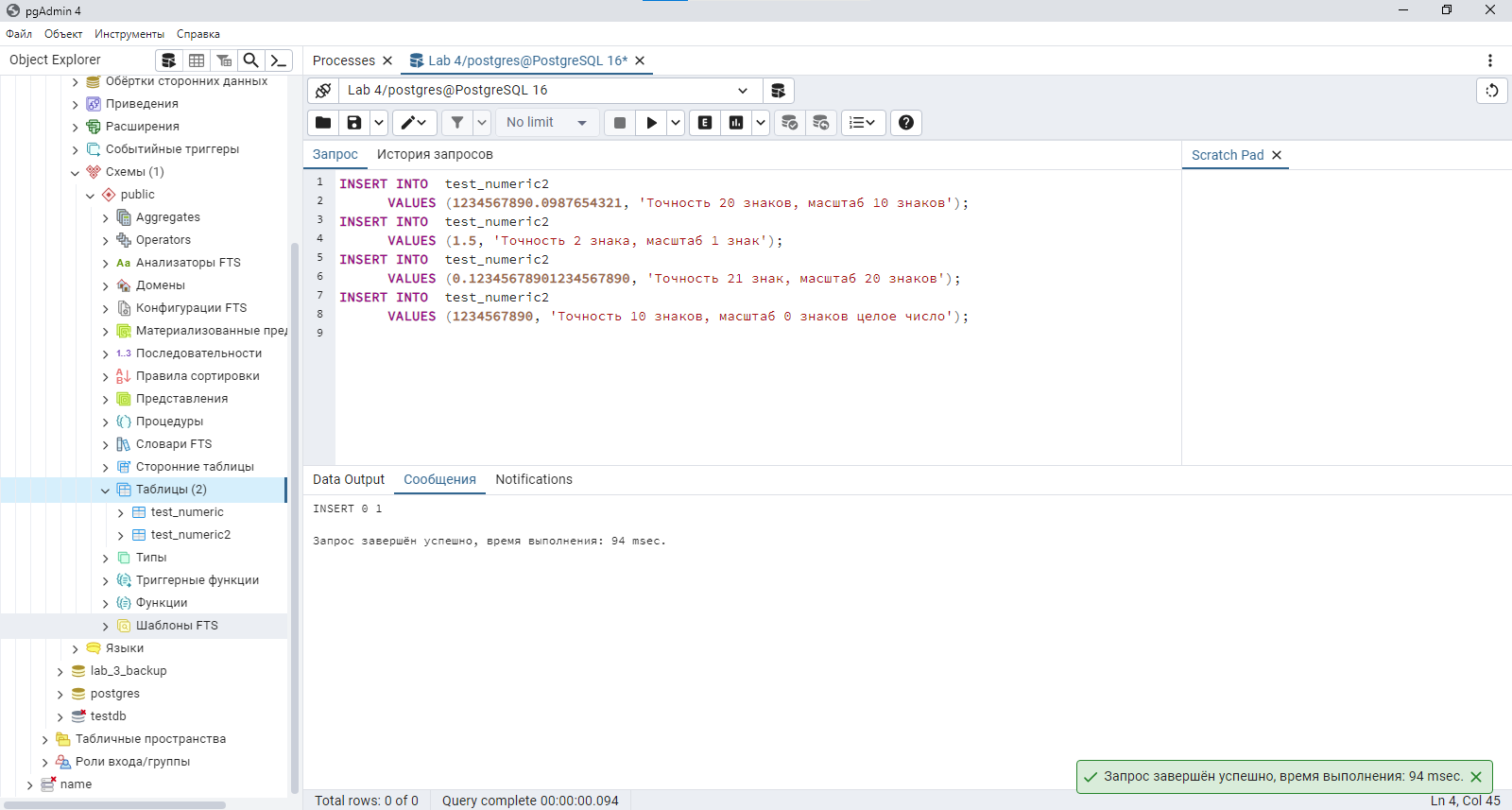
VALUES (1.5, ‘Точность 2 знака, масштаб 1 знак’);

INSERT INTO test\_numeric2

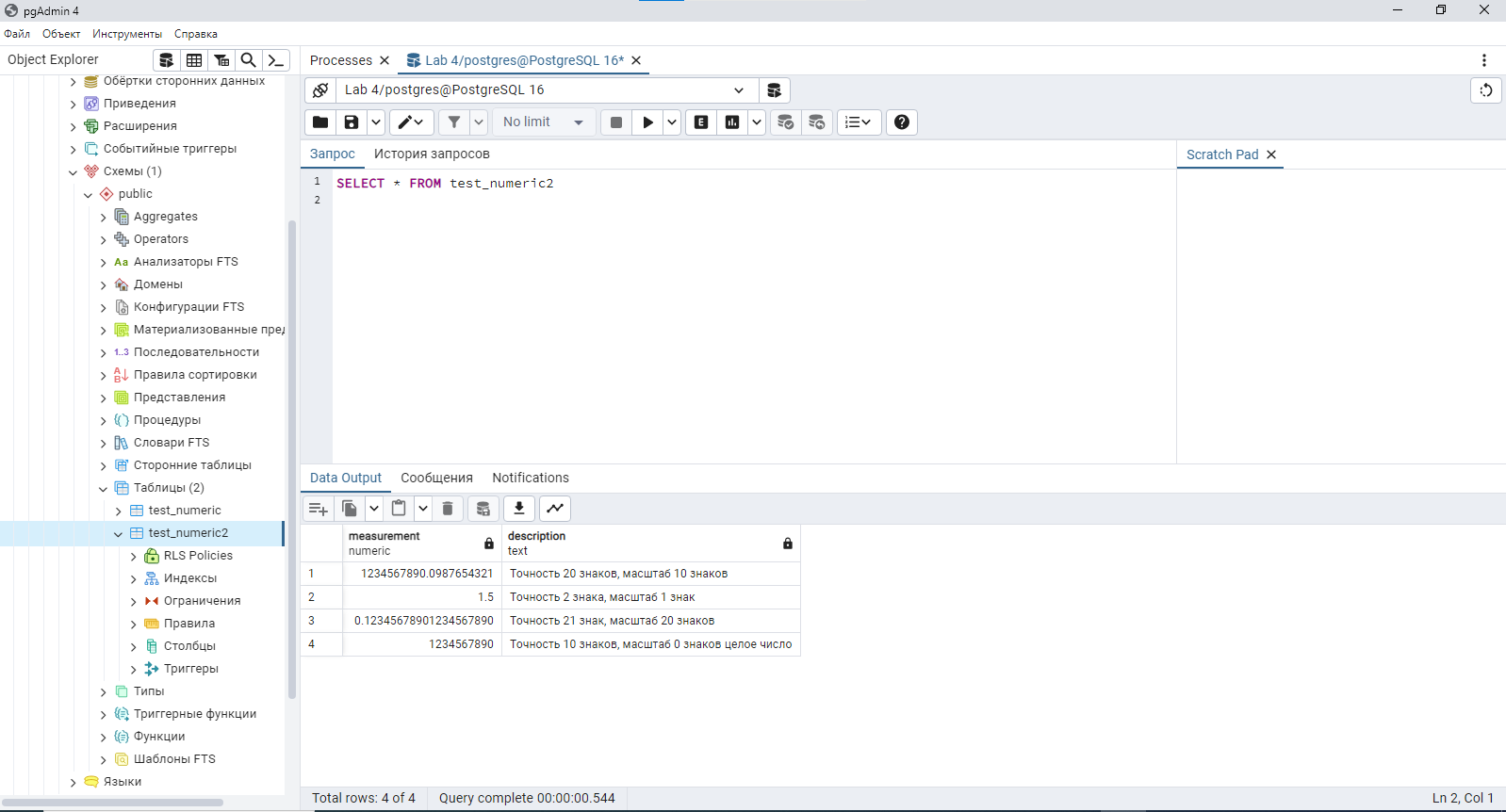
VALUES (0.12345678901234567890, ‘Точность 21 знак, масштаб 20 знаков’);

INSERT INTO test\_numeric2

VALUES (1234567890, ‘Точность 10 знаков, масштаб 0 знаков (целое число’);



2.3. Выполните выборку строк из таблицы командой SELECT и убедитесь в том, что все эти разнообразные значения сохранены в таблице в том виде, как Вы вводили.



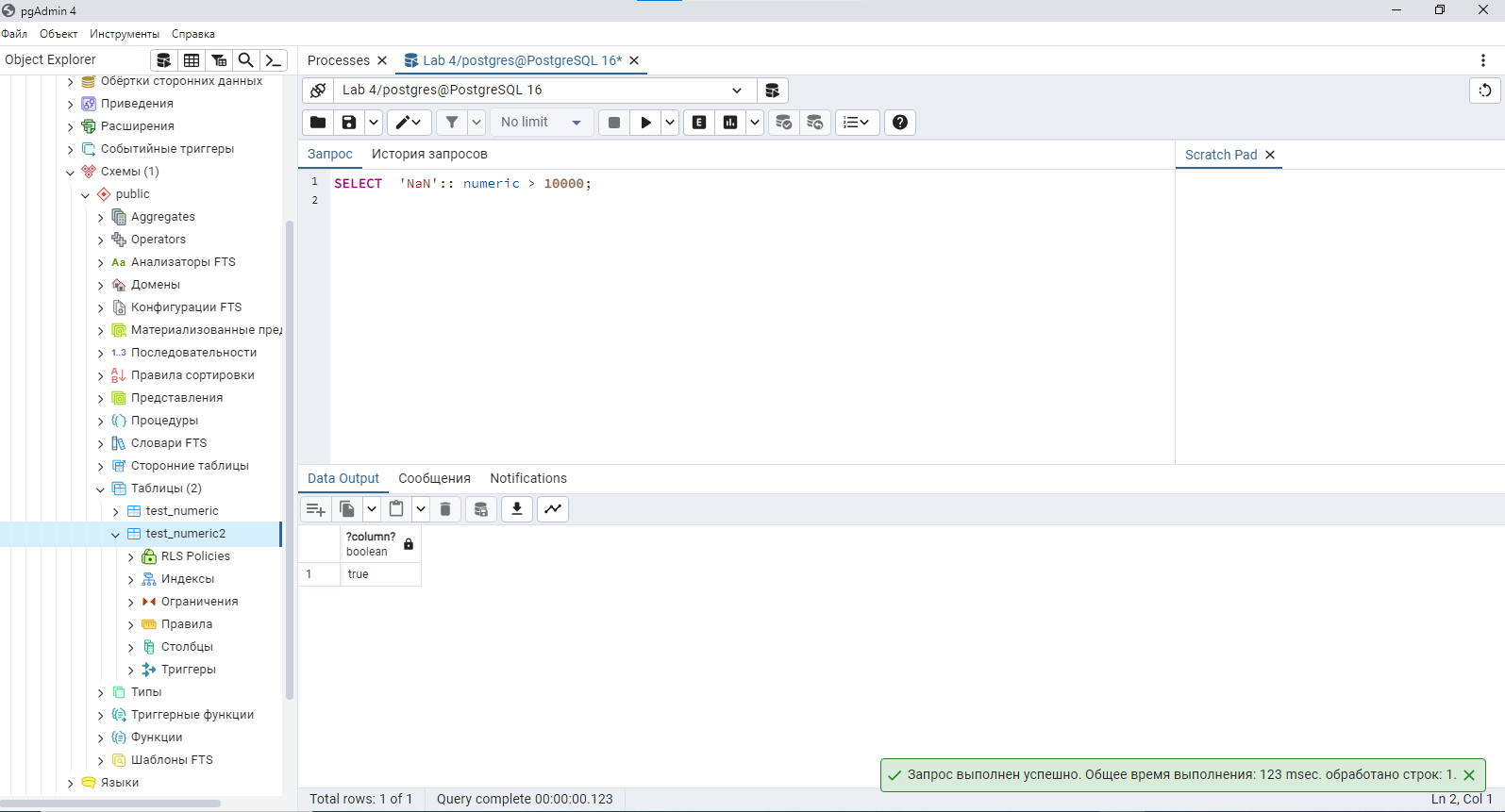
**Задание 3**. Тип данных numeric поддерживает специальное значение NaN (not a number, «не число»).

Значение NaN считается равным другому значению NaN.

Значение NaN считается большим любого другого «нормального» значения.

Проверьте это утверждение с помощью команды:

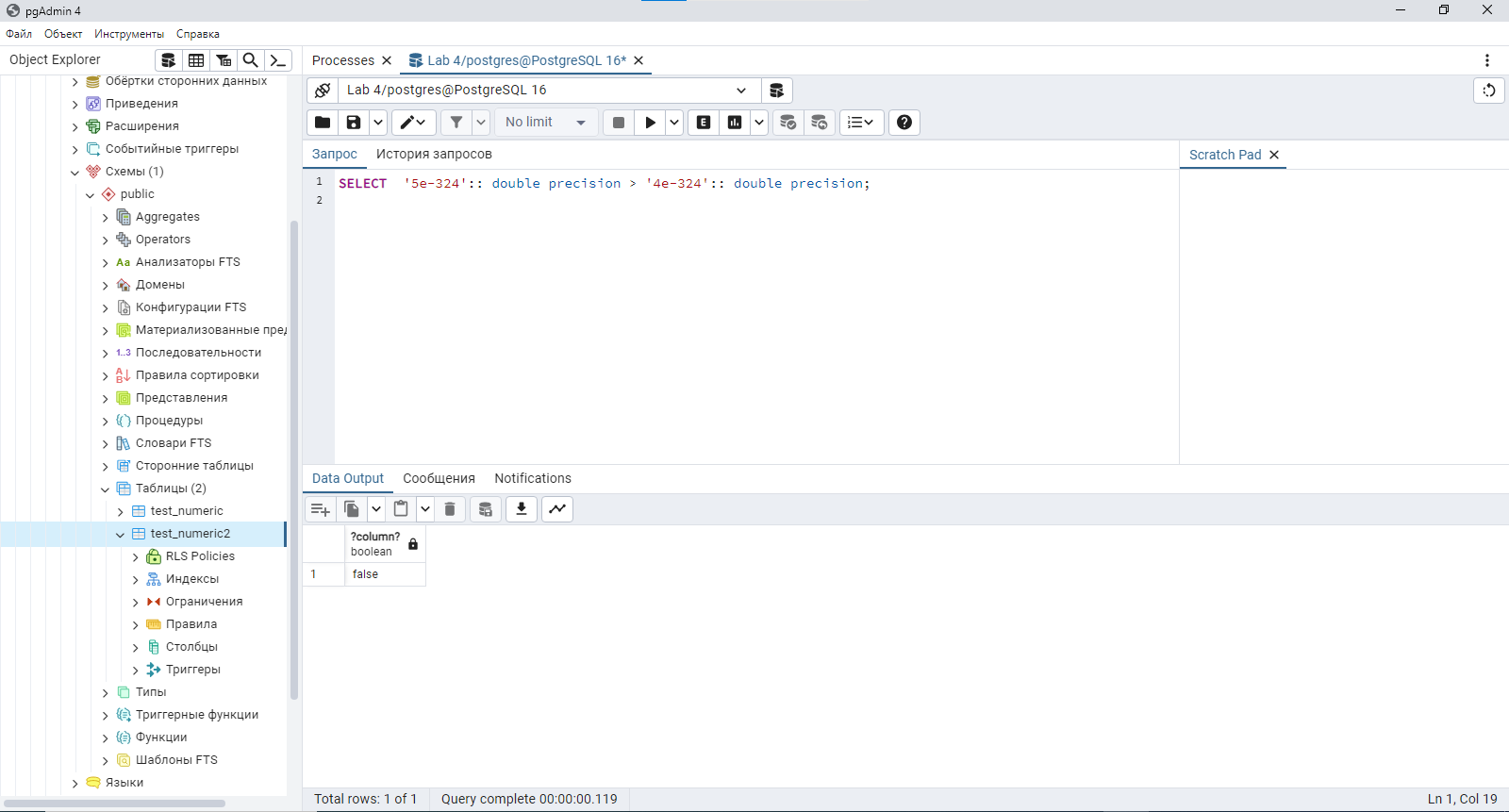
SELECT ‘NaN’:: numeric > 10000;



**Задание 4**. При работе с числами типа real и double precision нужно помнить, что сравнение двух чисел с плавающей точкой на предмет равенства их значений может привести к неожиданным результатам.

4.1. Сравните два очень маленьких числа (они представлены в экспоненциальной форме):

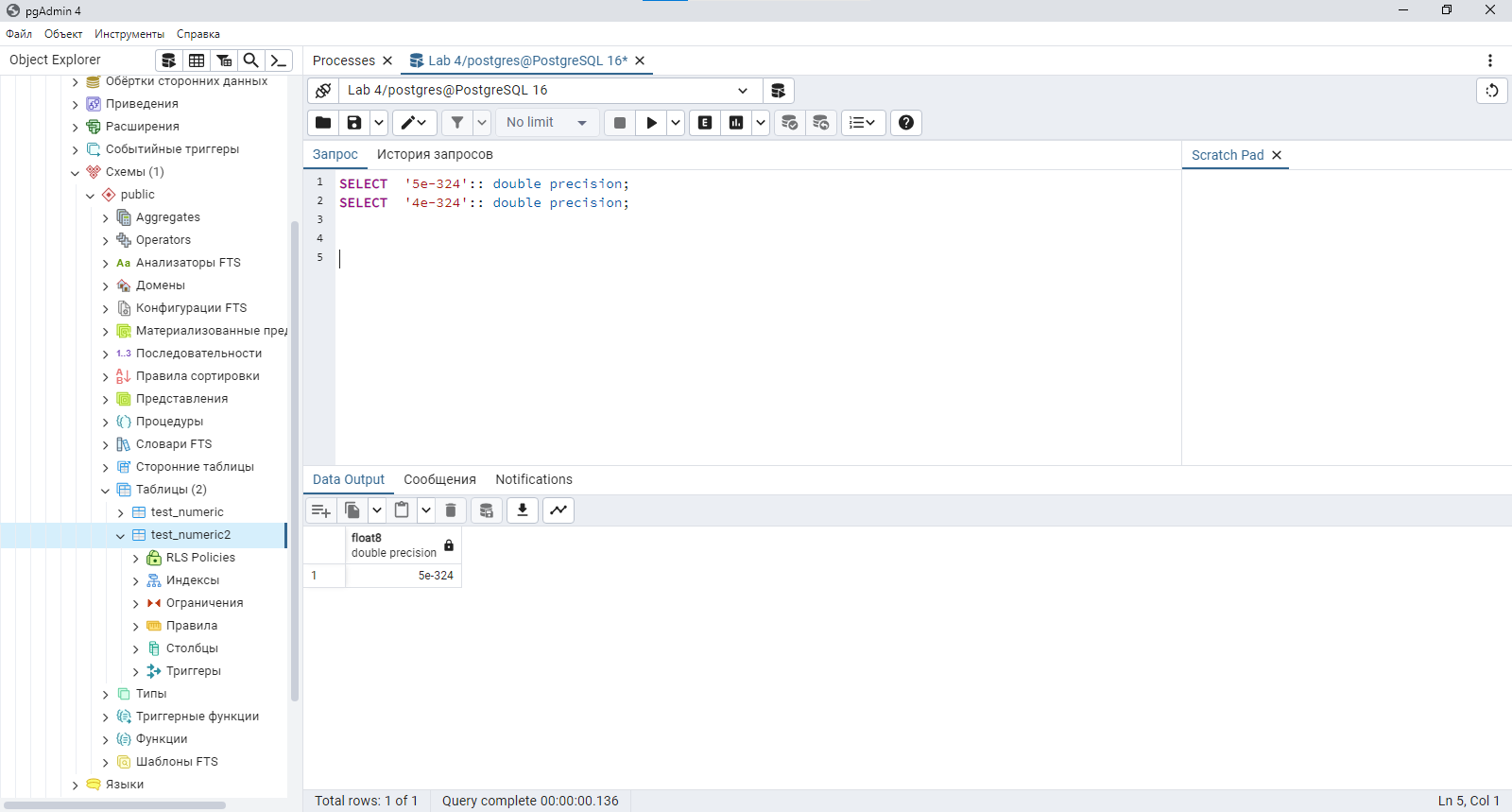
SELECT ‘5e-324’:; double precision > ‘4e-324’:: double precision;



4.2. Выполните еще два запроса и прокомментируйте результаты:

SELECT '5e-324':: double precision;

SELECT '4e-324':: double precision;



1. SELECT '5e-324'::double precision;  
Результат: 5e-324  
Комментарий: В данном запросе мы пытаемся преобразовать строку '5e-324' в тип double precision. Это очень маленькое число, близкое к нулю, представленное в экспоненциальной форме. Такие маленькие числа могут быть округлены до нуля при работе с типом double precision из-за ограничений точности.  
  
2. SELECT '4e-324'::double precision;  
Результат: 0  
Комментарий: В данном запросе мы пытаемся преобразовать строку '4e-324' в тип double precision. Это также очень маленькое число, еще ближе к нулю, представленное в экспоненциальной форме. Из-за ограничений точности типа double precision, такие маленькие числа могут быть округлены до нуля.

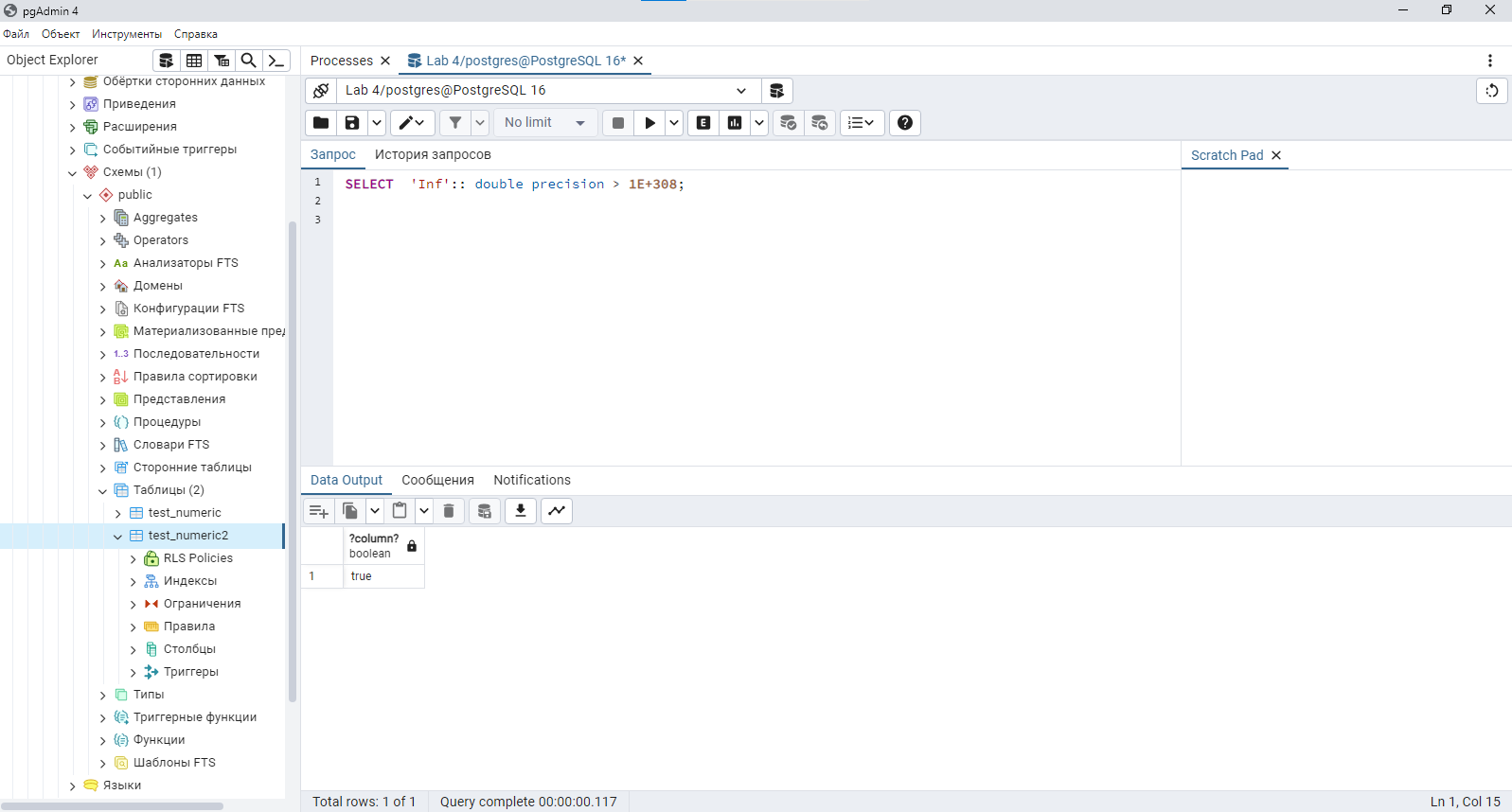
4.3. Проведите аналогичные эксперименты с очень большими числами, находящимися на границе допустимого диапазона для чисел типов real и double precision.

1. SELECT '1e38'::real;  
Результат: 1.00000001e+38  
Комментарий: В данном запросе мы пытаемся преобразовать строку '1e38' в тип real. Это очень большое число, находящееся на границе допустимого диапазона для типа real. В результате преобразования, число было округлено до 1.00000001e+38 из-за ограничений точности типа real.  
  
2. SELECT '1e308'::double precision;  
Результат: 1e+308  
Комментарий: В данном запросе мы пытаемся преобразовать строку '1e308' в тип double precision. Это очень большое число, находящееся на границе допустимого диапазона для типа double precision. В результате преобразования, число было округлено до 1e+308 из-за ограничений точности типа double precision.

**Задание 5**. Типы данных real и double precision поддерживают специальные значения Infinity (бесконечность) и –Infinity (отрицательная бесконечность).

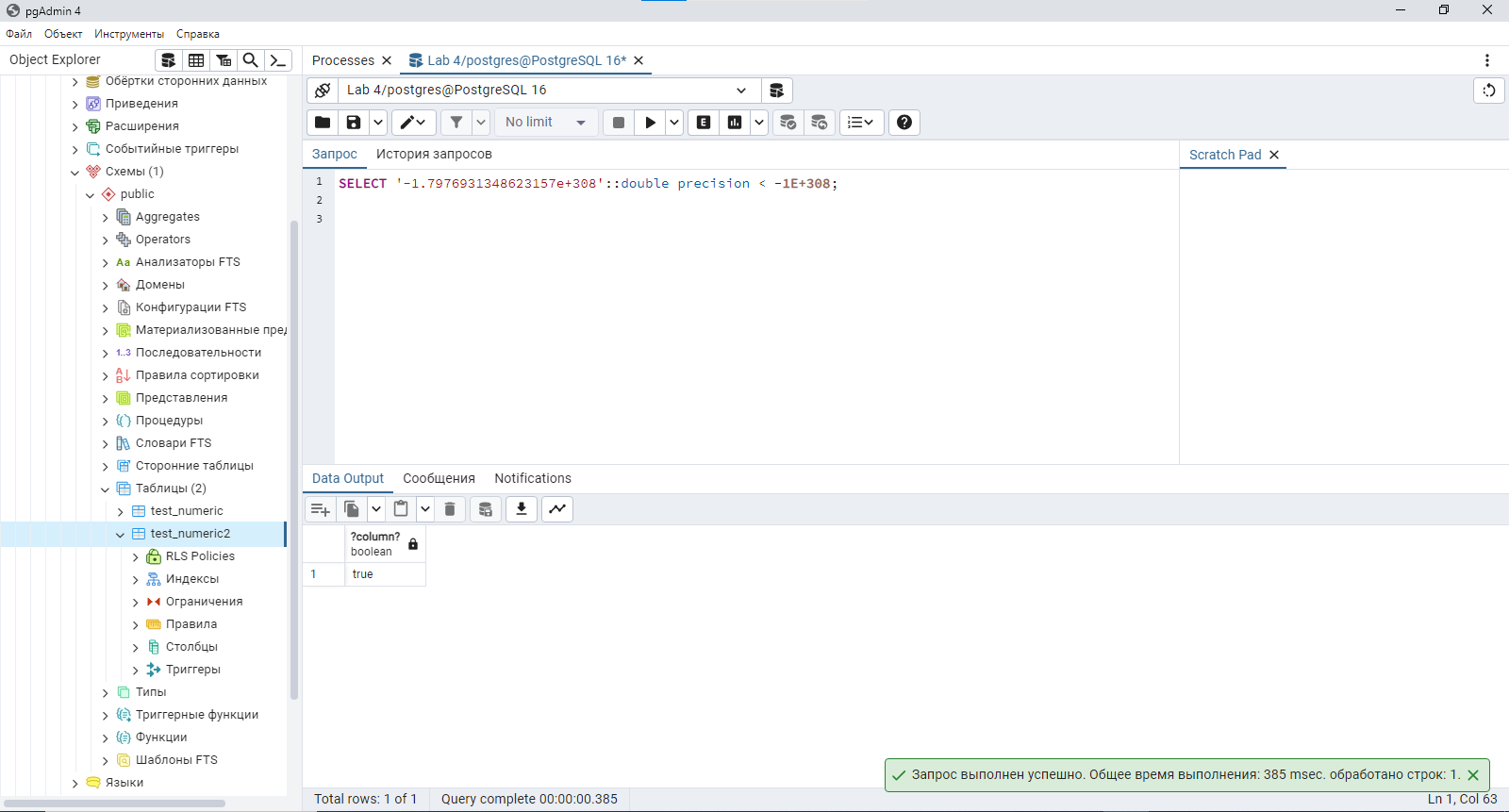
5.1. Проверьте с помощью команды SELECT ожидаемые свойства этих значений. Например, сравните Infinity с наибольшим значением, которое допускается для типа double precision (можно использовать сокращенное написание Inf):

SELECT ‘Inf’:: double precision > 1E+308;



5.2. Выполните аналогичный запрос для наименьшего возможного значения типа double precision.

SELECT '-1.7976931348623157e+308'::double precision < -1E+308;

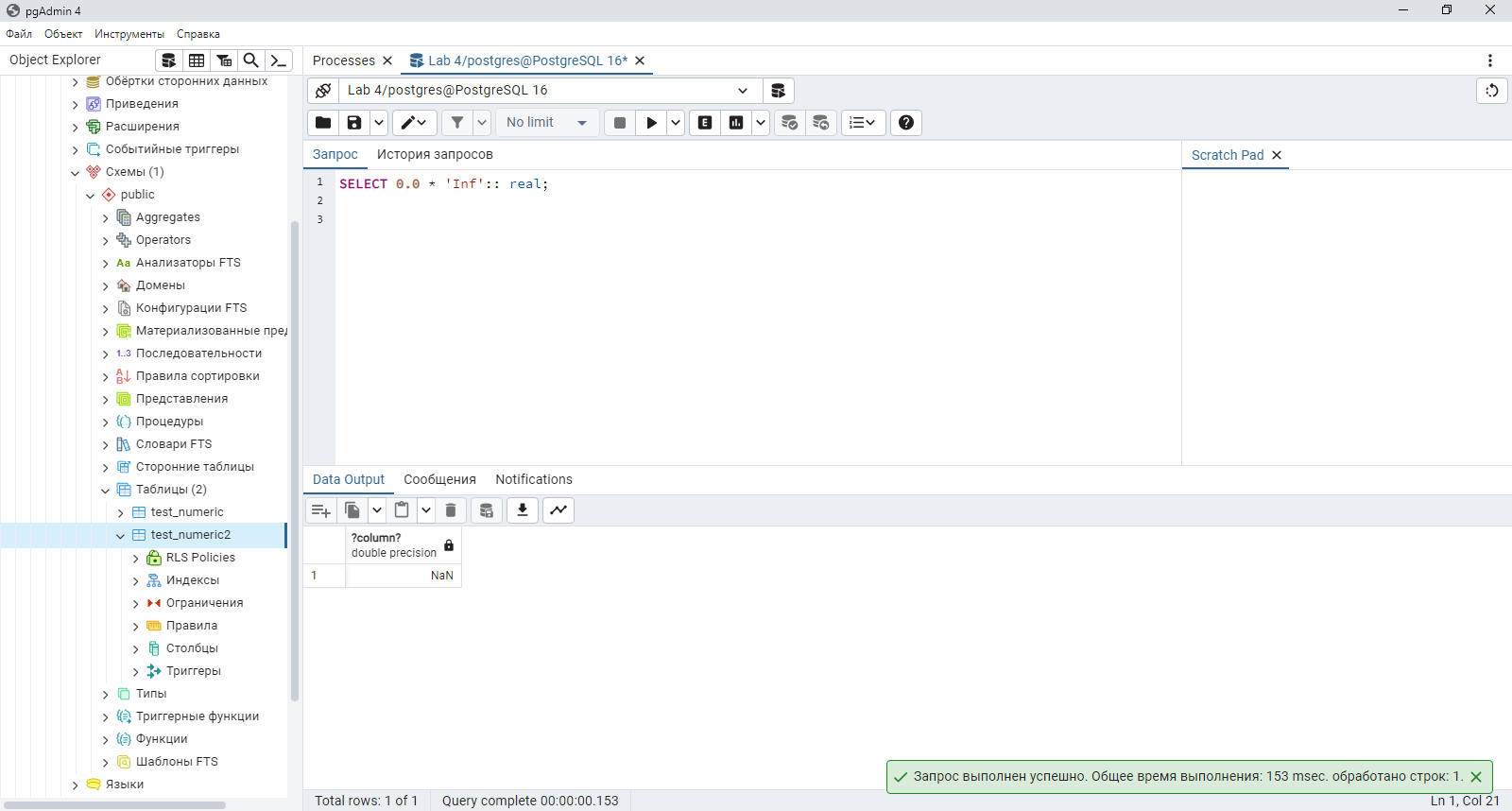


**Задание 6**. Типы данных real и double precision поддерживает специальное значение NaN (not a number, «не число»).

В математике существует такое понятие, как неопределенность. В качестве одного из ее вариантов служит результат операции умножения нуля на бесконечность.

6.1. Выполните команду и проанализируйте результат:

SELECT 0.0 \* ‘Inf’:; real;



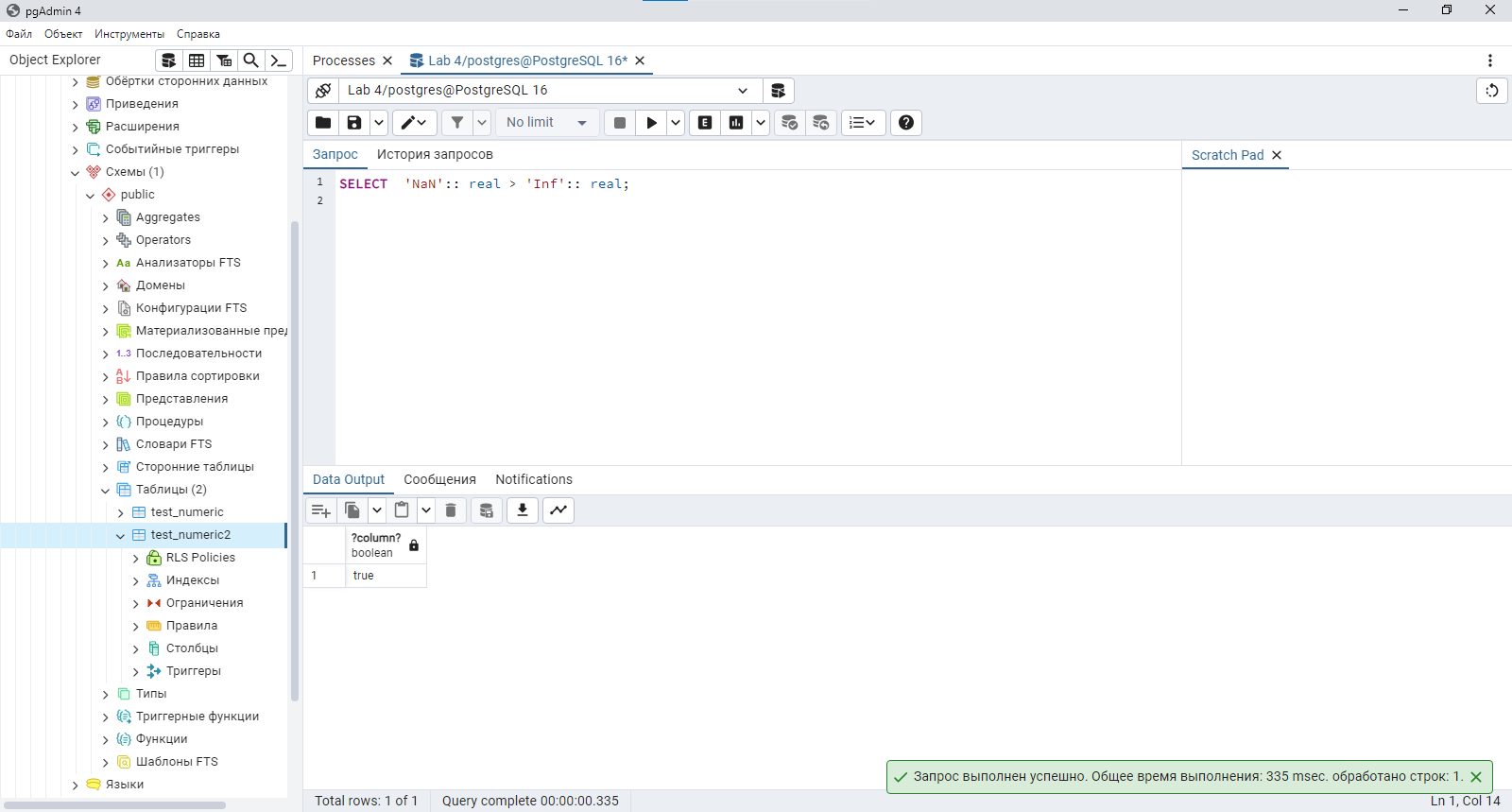
6.2. Значение NaN считается равным другому значению NaN.

Значение NaN считается большим любого другого «нормального» значения.

Проверьте это утверждение с помощью команды SELECT.

Сравните значения NaN и Infinity:

SELECT ‘NaN’:: real > ‘Inf’:: real;



**Задание 7**. Тип serial может применяться для столбцов, содержащих числовые значения, которые должны быть уникальными в пределах таблицы, например, идентификаторы каких-то объектов.

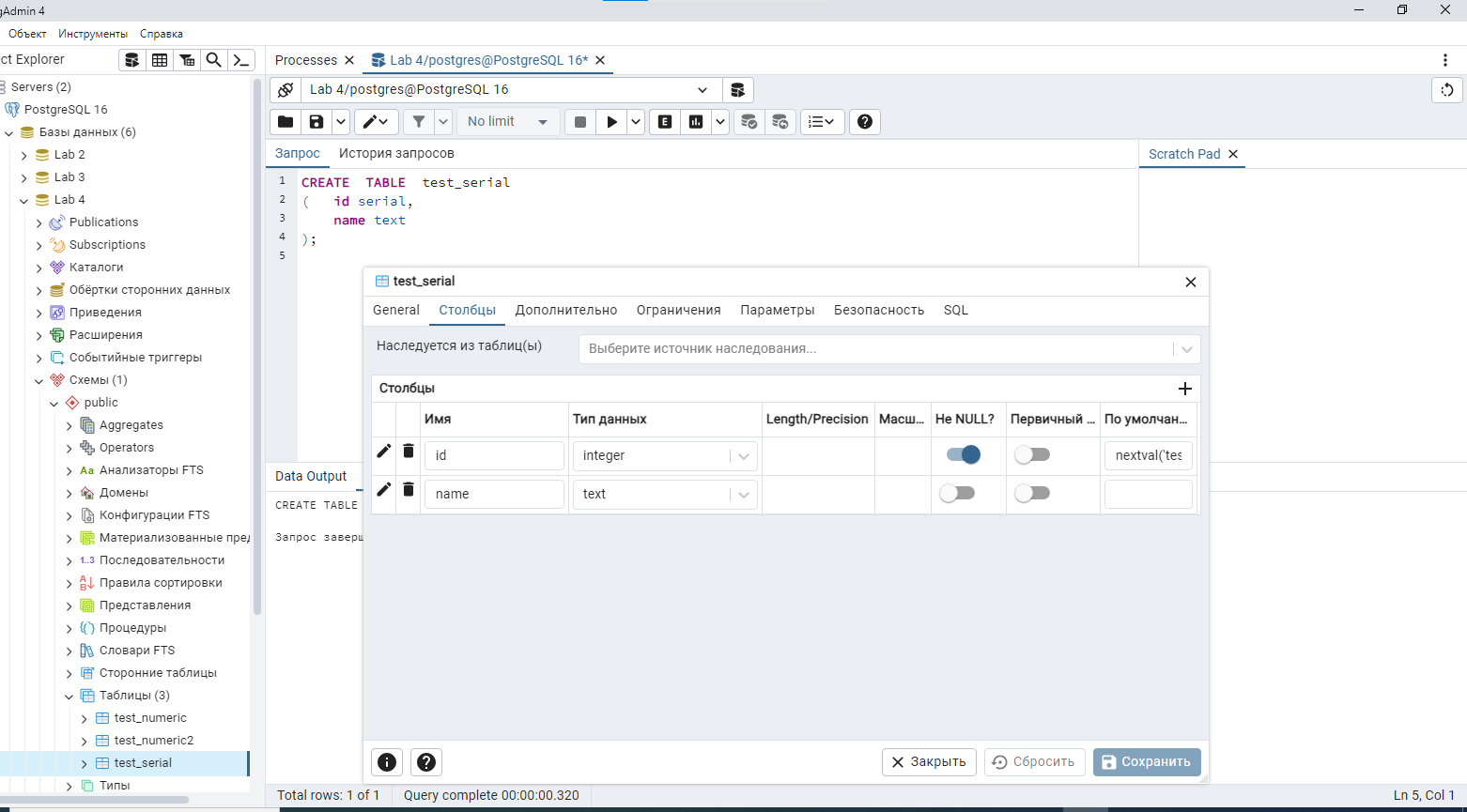
7.1. В качестве примера применения типа serial создайте таблицу test\_serial, содержащую наименования улиц и площадей:

CREATE TABLE test\_serial

( id\_serial,

name text

);



7.2. Введите несколько строк.

Обратите внимание, что значение для столбца id указывать не нужно!

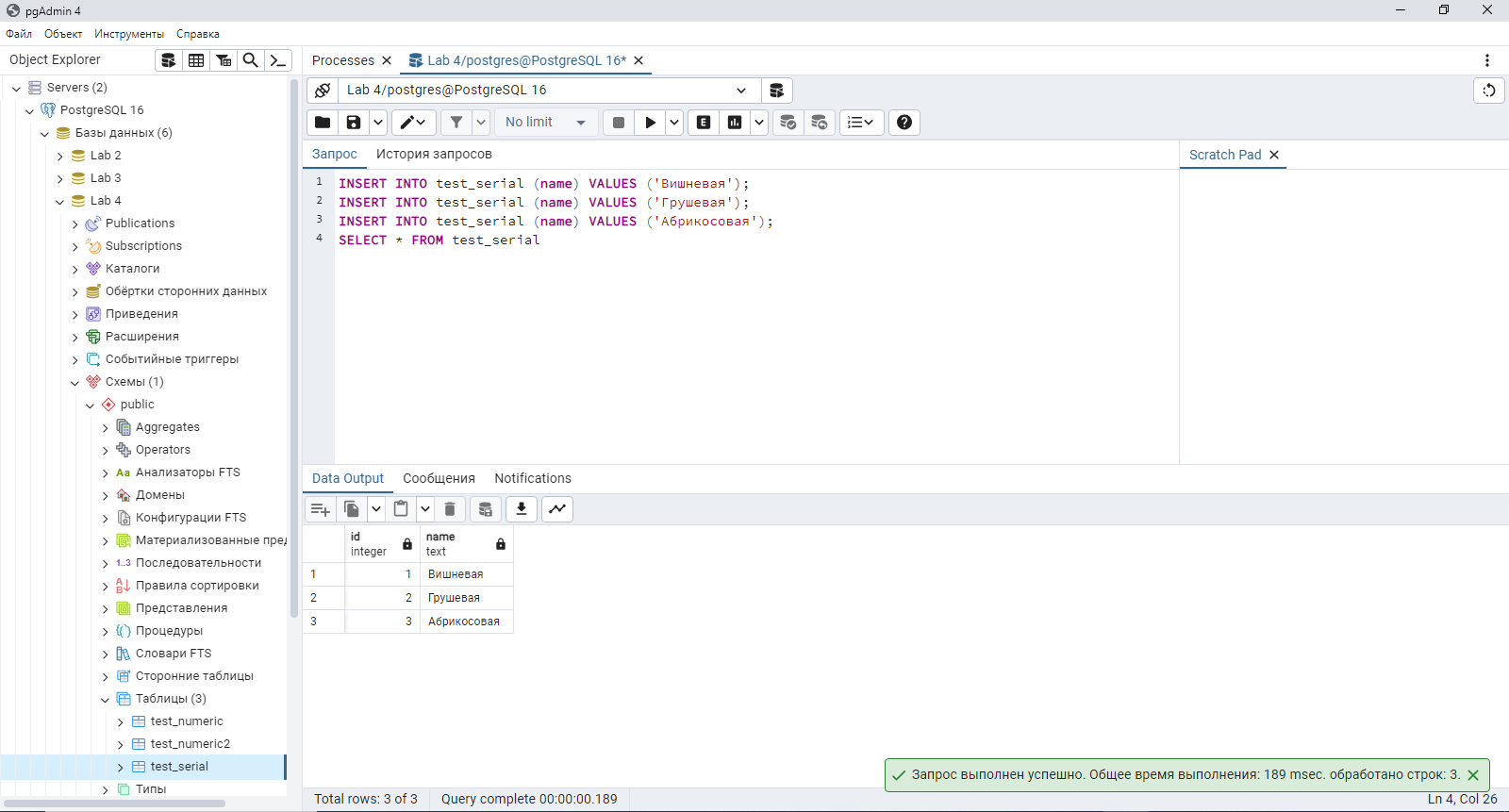
Но так как мы задаем значения не всех столбцов таблицы, то должны указать в команде INSERT не только список значений, но и список столбцов (в данном примере всего один столбец).

INSERT INTO test\_serial (name) VALUES (‘Вишневая’);

INSERT INTO test\_serial (name) VALUES (‘Грушевая’);

INSERT INTO test\_serial (name) VALUES (‘Абрикосовая’);

7.3. Выполните выборку данных из таблицы командой SELECT.

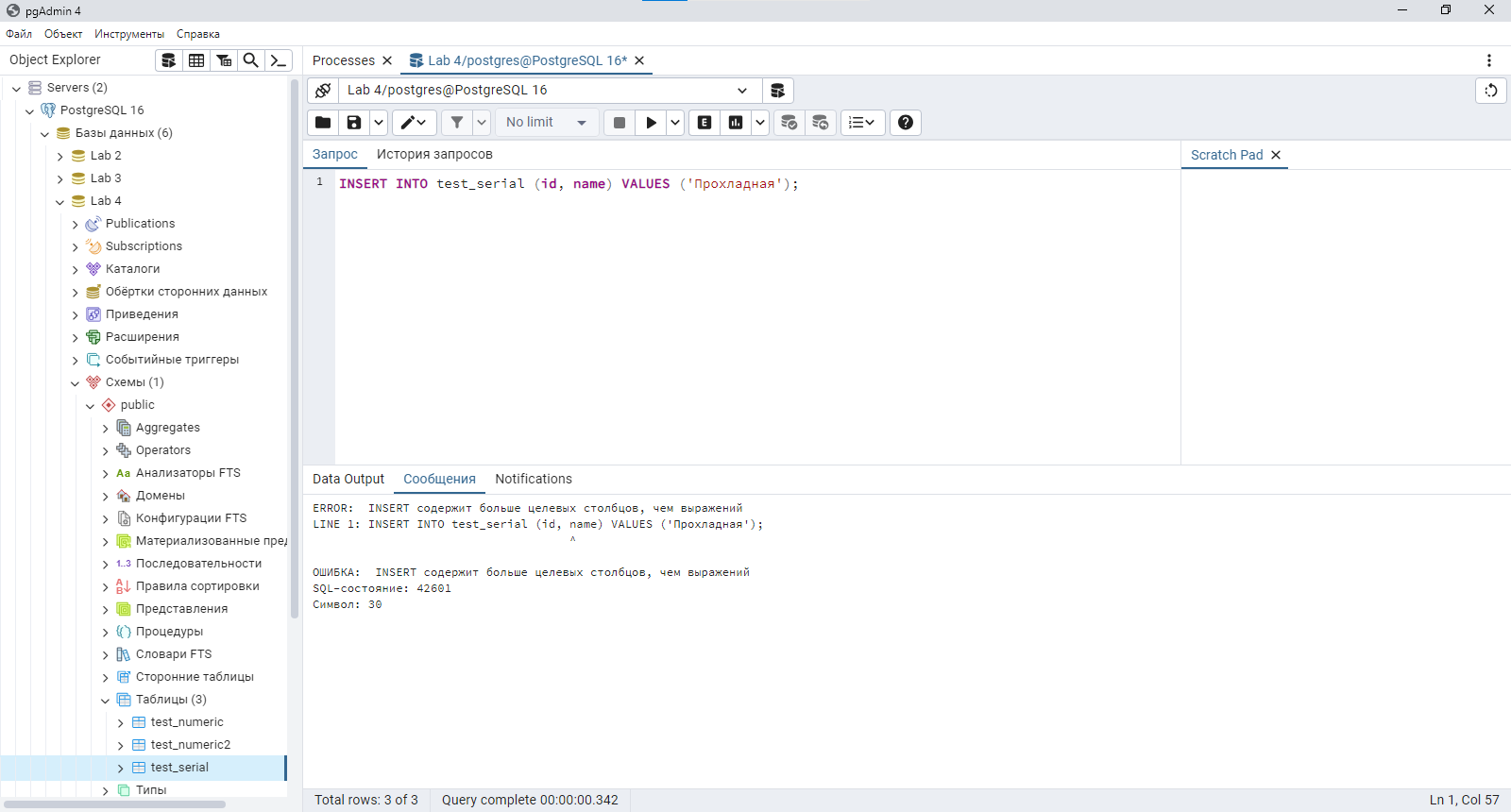


Убедитесь в том, что значения столбца id имеют последовательные значения, начиная с единицы.

7.4. Проведите эксперимент со столбцом id.

Выполните команду INSERT, в которой укажите явное значение столбца id:

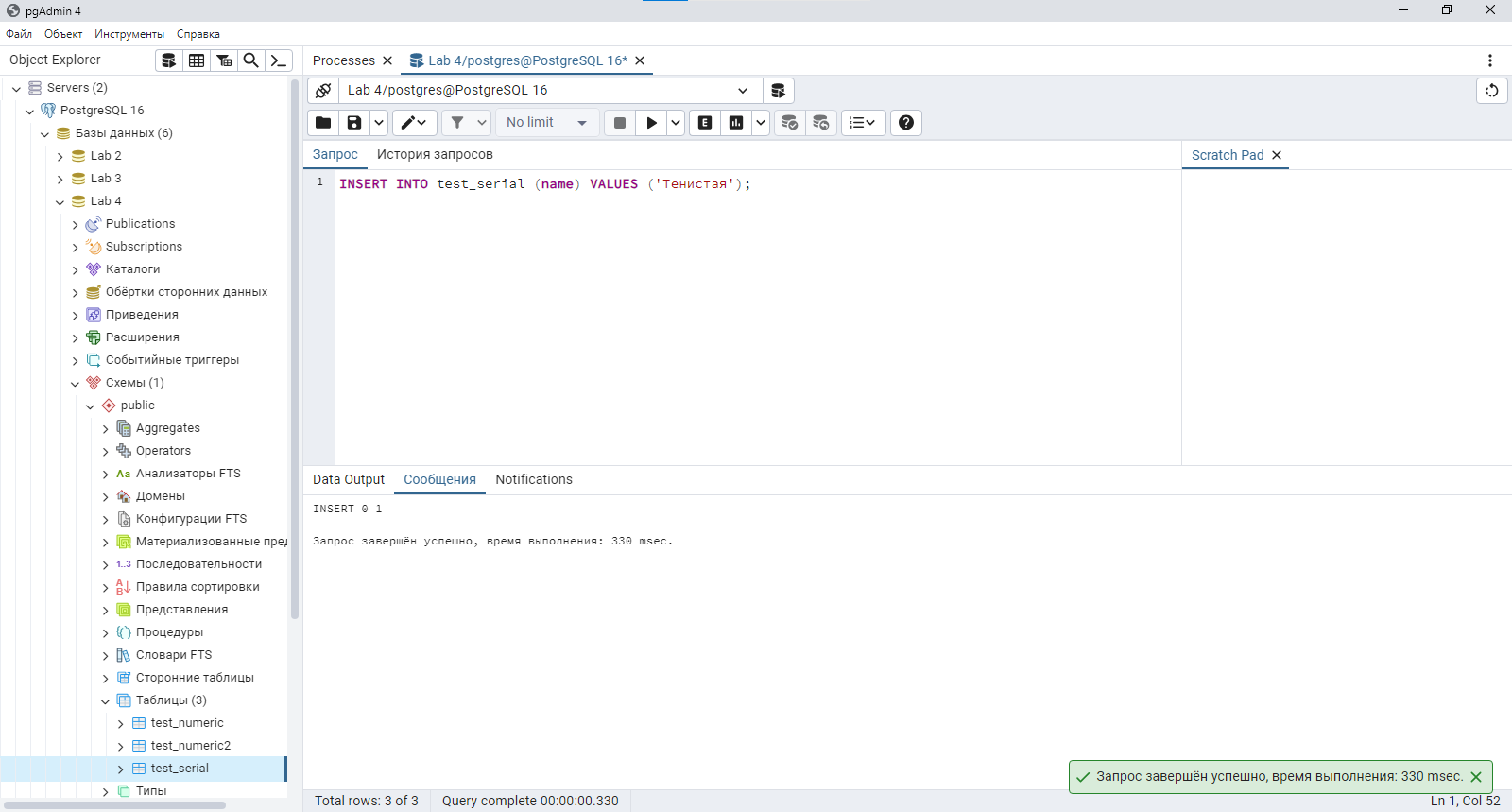
INSERT INTO test\_serial (id, name) VALUES (‘Прохладная’);



7.5. Добавьте еще одну строку, но уже не указывая явно значение для столбца id (как это было в предыдущих командах):

INSERT INTO test\_serial (name) VALUES (‘Тенистая’);

Убедитесь в том, что явное задание значения для столбца id не влияет на автоматическое генерирование значений этого столбца.



**Задание 8**. Усложним определение таблицы из предыдущего задания.

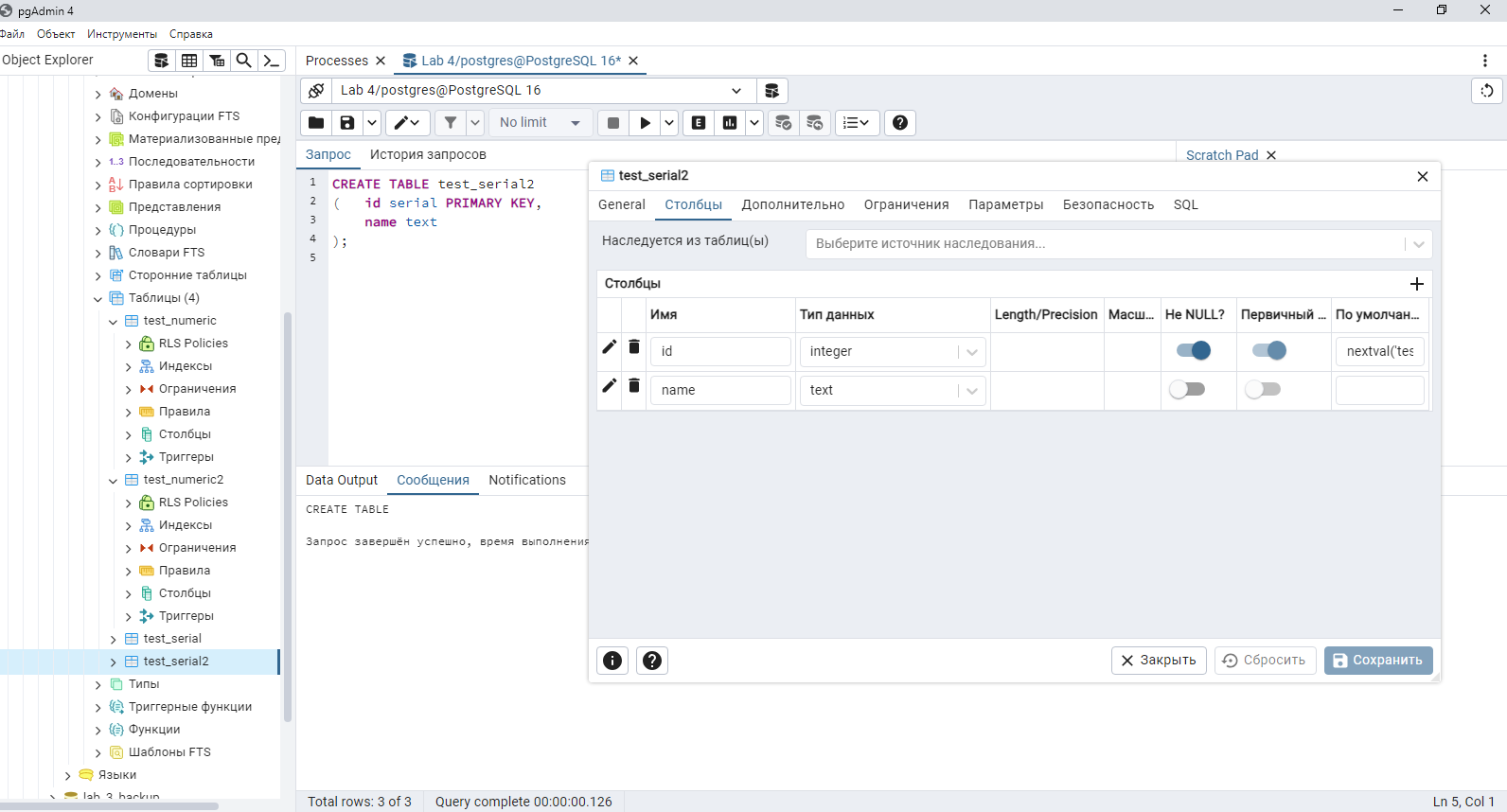
8.1. Пусть столбец id будет первичным ключом этой таблицы:

CREATE TABLE test\_serial2

( id serial PRIMARY KEY,

name text

);



8.2. Выполните следующие команды для добавления строк в таблицу и удаления одной строки из нее.

Для пошагового управления этим процессом выполняйте выборку данных из таблицы с помощью команды SELECT после каждой команды вставки или удаления.

INSERT INTO test\_serial2 (name) VALUES (‘Вишневая’);

Явно зададим значение столбца id:

INSERT INTO test\_serial2 (id, name) VALUES (2, ‘Прохладная’);

Почему СУБД не выдало сообщение об ошибке?

Выполним эту же команду еще раз.

INSERT INTO test\_serial2 (id, name) VALUES (2, ‘Прохладная’);

При выполнении этой команды СУБД выдаст сообщение об ошибке. Почему?

INSERT INTO test\_serial2 (name) VALUES (‘Грушевая’);

Повторим эту же команду.

INSERT INTO test\_serial2 (name) VALUES (‘Грушевая’);

Все в порядке. Почему?

А теперь удалим строку с улицей «Грушевая», которую ввели дважды.

DELETE FROM test\_serial2 WHERE id = 4;

Добавим еще одну строк.

INSERT INTO test\_serial2 (name) VALUES (‘Абрикосовая’);

Добавим еще строку.

INSERT INTO test\_serial2 (name) VALUES (‘Тенистая’);

Теперь выполните выборку командой:

SELECT \* FROM test\_serial2;

Убедитесь в том, что в нумерации образовалась «дыра», пропал номер 4 (рис. 1).

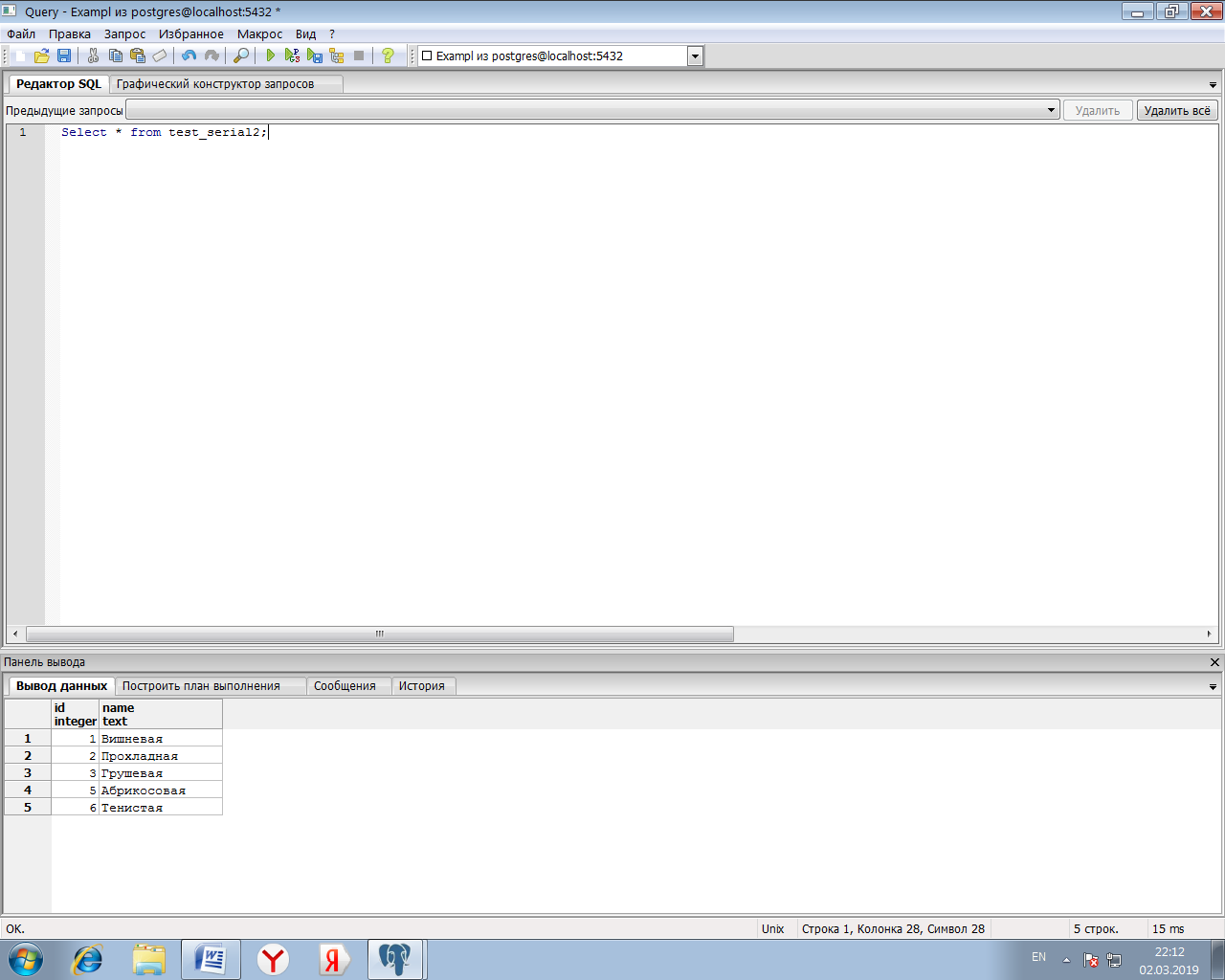
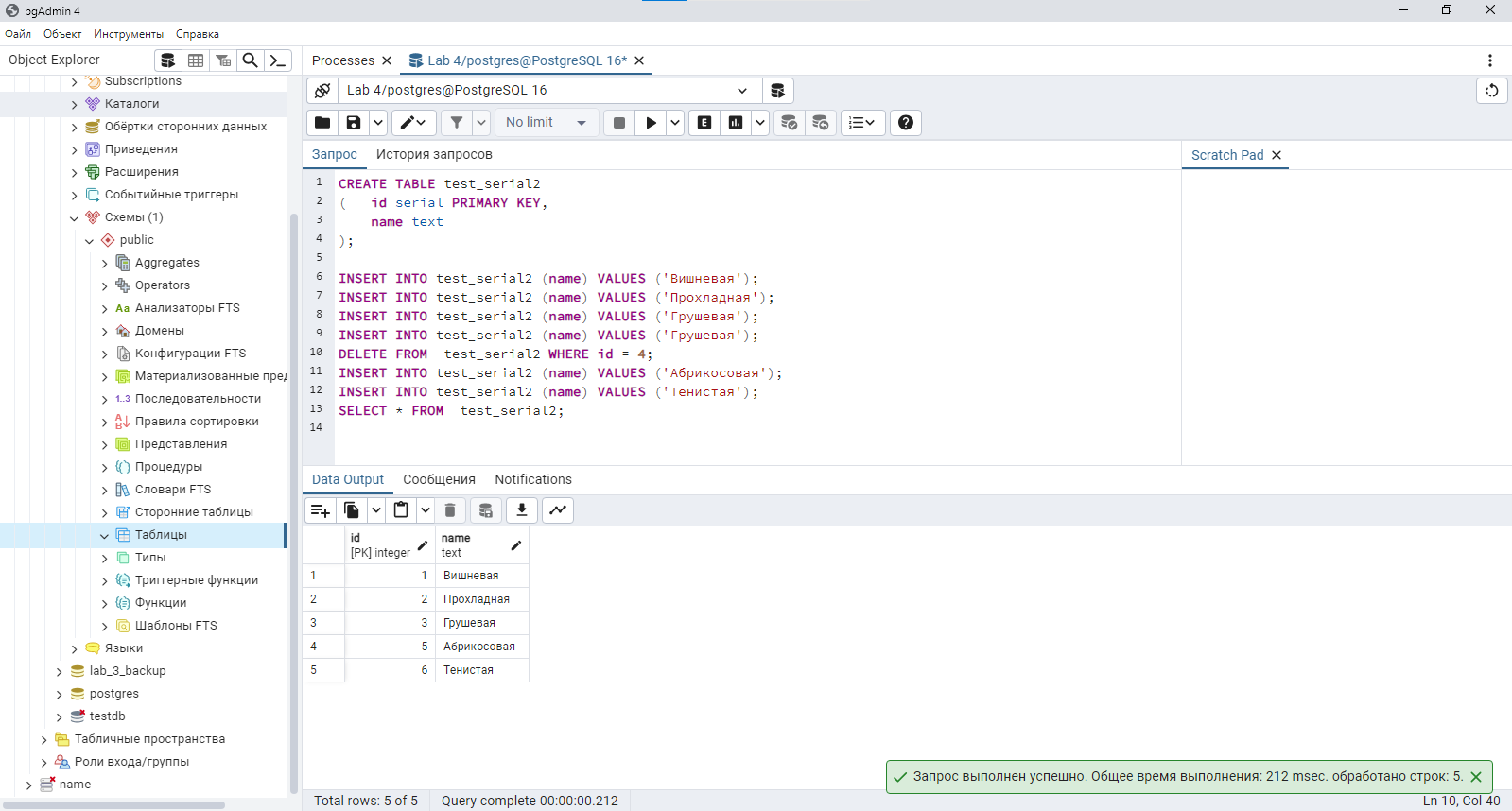


Рисунок 1- Результаты выполнения задания 8.2

Пропал номер 4 из-за того, что при формировании нового значения из последовательности поиск максимального значения, уже имеющегося в столбце, не выполнятся.

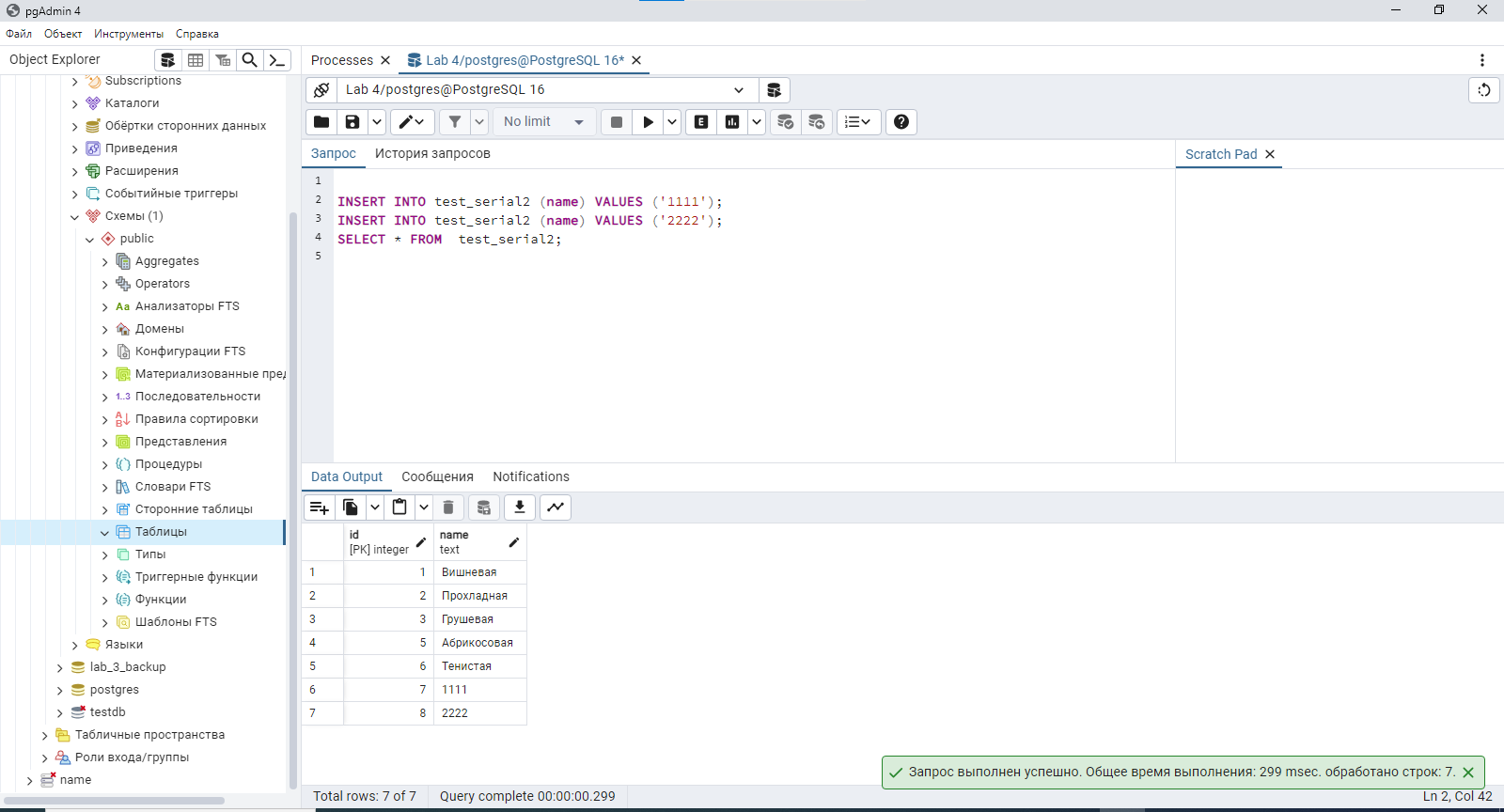


9. **Добавьте 2 названия новых улиц**. **Выведите новый список улиц.**

INSERT INTO test\_serial2 (name) VALUES (1111');

INSERT INTO test\_serial2 (name) VALUES ('2222');

SELECT \* FROM test\_serial2;



Выводы: в ходе лабораторной работы я получил практические навыки в использовании различных типов данных в СУБД PostgreSQL.