Условие:

Написать хранимую процедуру (ХП), которая возвращает следующее целое число в столбце. Для этого используется отдельная спец. таблица, в которой есть столбцы `id`, `имя таблицы`, `имя столбца` и `текущее максимальное значение`. Пользователь (программист) передаёт в функцию параметром имя таблицы и имя столбца. ХП ищет есть ли такая запись в спец. таблице. Если запись есть, то значение инкрементируется, после чего возвращается пользователю. Если такой записи нет - ХП сперва ищет максимальное число в столбце в запрашиваемой таблице, записывает новую строку, содержащую следующее за найденным число, в спец. таблицу и возвращает это значение пользователю. При отсутствии значений в запрашиваемой таблице, пользователю возвращается 1, и этот же результат записывается в спец. таблицу. Следующий идентификатор для новой строки в спец. таблице формируется рекурсивным вызовом разработанной ХП.

Л/Р выполнена при помощи процедурного расширения языка SQL – PL/pgSQL и СУБД PostgreSQL

Результаты работы скрипта, на примере из задания

1. Создание спец. таблицы.

CREATE TABLE

Query returned successfully in 63 msec.

2. Добавление в спец. таблицу записи (1, spec, id, 1). // информация о том, что максимальное число в столбце id спец. таблицы равно 1

INSERT 0 1
Query returned successfully in 77 msec.

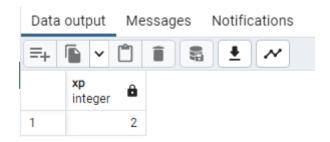
3. Создание хранимой процедуры (ХП).

Data output Messages Notifications

CREATE FUNCTION

Query returned successfully in 56 msec.

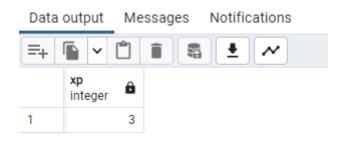
4. Вызов вашей ХП с параметрами 'spec', 'id'. Функция должна вернуть `2`.



5. Распечатка содержимого спец. таблицы. Должна быть 1 строка "(1, spec, id, 2)".



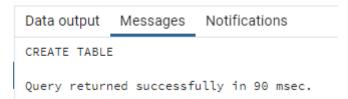
6. Вызов вашей ХП с параметрами 'spec', 'id'. Функция должна вернуть `3`.



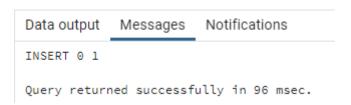
7. Распечатка содержимого спец. таблицы. Должна быть 1 строка "(1, spec, id, 3)".



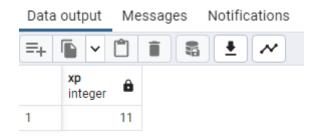
8. Создание новой таблицы с одним столбцом 'id'. Назовём её test.



9. Добавление в таблицу test записи (10).



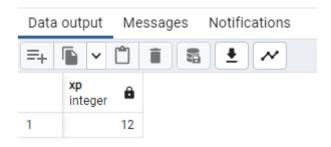
10. Вызов вашей ХП с параметрами 'test', 'id'. Функция должна вернуть `11`. // место для рекурсии



11. Распечатка содержимого спец. таблицы. Должно быть 2 строки "(1, spec, id, 4)" "(4, test, id, 11)".



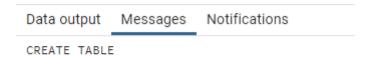
12. Вызов вашей ХП с параметрами 'test', 'id'. Функция должна вернуть `12`.



13. Распечатка содержимого спец. таблицы. Должно быть 2 строки "(1, spec, id, 4)" "(4, test, id, 12)".

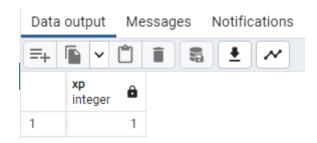


14. Создание таблицы 'test2' с столбцами 'numValue1', 'numValue2'.



Query returned successfully in 101 msec.

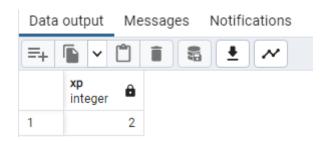
15. Вызов вашей ХП с параметрами 'test2', 'numValue1'. Функция должна вернуть `1`.



16. Распечатка содержимого спец. таблицы. Должно быть 3 строки "(1, spec, id, 5)" "(4, test, id, 12), (5, test2, numValue1, 1)".



17. Вызов вашей ХП с параметрами 'test2', 'numValue1'. Функция должна вернуть `2`.



18. Распечатка содержимого спец. таблицы. Должно быть 3 строки "(1, spec, id, 5)" "(4, test, id, 12), (5, test2, numValue1, 2)".

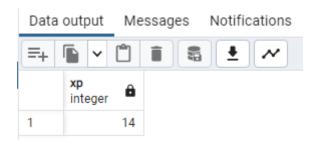


19. Добавление в таблицу 'test2'(numValue1, numValue2) записи (2, 13).



Query returned successfully in 67 msec.

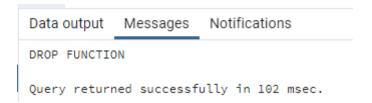
20. Вызов вашей ХП с параметрами 'test2', 'numValue2'. Функция должна вернуть `14`.



21. Распечатка содержимого спец. таблицы. Должно быть 4 строки "(1, spec, id, 6)" "(4, test, id, 12), (5, test2, numValue1, 2), (6, test2, numValue2, 14)".



22. Удаление ХП.



23. Удаление таблиц.



Все тесты пошли проверку, функция и таблицы корректно удалены. Подробные комментарии по реализации функции описаны в теле функции.

Листинг скрипта:

```
-- 1
CREATE TABLE spec
  id integer NOT NULL,
  table_name character varying(30) NOT NULL,
  column_name character varying(30) NOT NULL,
  cur_max_value integer NOT NULL
);
--2
INSERT INTO spec VALUES (1, 'spec', 'id', 1);
--3
-- реализация
CREATE OR REPLACE FUNCTION xp (_table_name text, _column_name text)
RETURNS integer
AS $$
DECLARE
 maxValue integer := 0; -- понадобится для поиска макс. знач. в столбце запрашиваемой таблицы
BEGIN
-- проверяем есть ли в таблице spec запись таблицы и столбца, которые пришли в качестве параметров
 ΙF
       (SELECT COUNT(*)
  FROM spec
       WHERE column_name = _column_name AND table_name = _table_name) > 0
 THEN
 -- если такая запись нашлась, то увеличиваем макс. значение на 1 и возвращаем это же значение
  UPDATE spec
       SET cur_max_value = cur_max_value + 1
       WHERE column_name = _column_name AND table_name = _table_name;
       return cur_max_value from spec
                 WHERE column_name = _column_name AND table_name = _table_name;
```

-- иначе ищем максимальное значение в таблице, название которой пришло параметром, в столбце, -- который так же пришел параметром EXECUTE format('SELECT MAX(%s) FROM %s ', _column_name, _table_name) INTO maxValue; -- если приходит null -> в таблице нет записей -> присваиваем maxValue значение = 1 -- если же maxValue не null -> инкрементируем maxValue IF maxValue IS null THEN maxValue := 1; ELSE maxValue := maxValue + 1; END IF; -- далле нам нужно сделать новую запись в таблицу spec. id - это значение, -- которое вернула наша функция с параметрами xp('spec', 'id'). -- следующие два параметра - это название таблицы и столбца, которые пришли при первом вызове функции -- записывать строковые данные в PL/pgSQL нужно в одинарных ковычках. Так как мы используем экранирование -- ковычки в функции дублируются. Поэтому второй и третий пишем определяем в двойных ковычках явно -- в качестве макс. значение передаем уже найденное значение из переменной maxValue EXECUTE format('INSERT INTO spec VALUES (%s, "%s", "%s", %s)', (SELECT xp('spec', 'id')), _table_name, _column_name, maxValue); -- И возвращаем это макс. значение. Т.е. если в таблице уже была запись в рассматриваемом столбце, -- то на выходе из функции будет это значене из столбца, увеличенное на 1. Если не было, -- то вернется просто 1. RETURN maxValue; END IF; END; \$\$ LANGUAGE plpgsql; SELECT xp('spec', 'id')

SELECT * FROM spec

--6

```
SELECT xp('spec', 'id')
--7
SELECT * FROM spec
--8
CREATE TABLE test
(
  id integer NOT NULL
);
--9
INSERT INTO test VALUES (10);
--10
SELECT xp('test', 'id')
--11
SELECT * FROM spec
--12
SELECT xp('test', 'id')
--13
SELECT * FROM spec
--14
CREATE TABLE test2
  numValue1 integer NOT NULL,
        numValue2 integer NOT NULL
);
--15
SELECT xp('test2', 'numValue1')
--16
SELECT * FROM spec
--17
SELECT xp('test2', 'numValue1')
--18
SELECT * FROM spec
--19
INSERT INTO test2 VALUES (2, 13);
```

```
--20

SELECT xp('test2', 'numValue2')
--21

SELECT * FROM spec
--22

DROP FUNCTION xp(_table_name text, _column_name text)
--23

DROP TABLE spec;

DROP TABLE test;

DROP TABLE test2
```

Дата сдачи 17.10.2022