

Домашняя работа №10

Попов Матвей, М8О-114СВ-24

Задание 12

12. Сделайте выборки данных из таблиц «Персонал» и «Организационная структура», а также реконструируйте организационную структуру с помощью двух представлений (view). Команды можно выполнять не только в среде интерактивного терминала psql, но также и из командной строки операционной системы. Выполните эти команды в командной строке операционной системы:

```
psql -d ais -c "SELECT * FROM Personnel"
psql -d ais -c "SELECT * FROM Org_chart"
psql -d ais -c "SELECT * FROM Personnel_org_chart"
psql -d ais -c "SELECT * FROM Create_paths"
```

Не забудьте, что если не указан параметр -U, то утилита psql подключается к базе данных от имени пользователя базы данных, имя которого совпадает с именем пользователя операционной системы. Поэтому возможно, что вам придется использовать параметр -U, если в базе данных не создана учетная запись такого пользователя.

Результат

```
papey08@papey08-Macbook ~ % psql -h localhost -p 5432 -U postgres -d ais -c "SELECT * FROM Personnel"
Password for user postgres:
 emp_nbr | emp_name | address | birth_date
-----+-----+-----+-----
      0 | вакансия |         | 2014-05-19
      1 | Иван    | ул. Любителей языка С | 1962-12-01
      2 | Петр    | ул. UNIX гуру | 1965-10-21
      3 | Антон   | ул. Ассемблерная | 1964-04-17
      4 | Захар   | ул. им. СУБД PostgreSQL | 1963-09-27
      5 | Ирина   | просп. Программистов | 1968-05-12
      6 | Анна    | пер. Перловый | 1969-03-20
      7 | Андрей  | пл. Баз данных | 1945-11-07
      8 | Николай | наб. ОС Linux | 1944-12-01
(9 rows)
```

```
papey08@papey08-Macbook ~ % psql -h localhost -p 5432 -U postgres -d ais -c "SELECT * FROM Org_chart"
Password for user postgres:
  job_title      | emp_nbr | boss_emp_nbr | salary
-----+-----+-----+-----
Президент       |      1 |              | 1000.0000
Вице-президент 1 |      2 |              | 900.0000
Вице-президент 2 |      3 |              | 800.0000
Архитектор       |      4 |              | 700.0000
Ведущий программист |      5 |              | 600.0000
Программист С    |      6 |              | 500.0000
Программист Perl |      7 |              | 450.0000
Оператор        |      8 |              | 400.0000
(8 rows)
```

```
papey08@papey08-Macbook ~ % psql -h localhost -p 5432 -U postgres -d ais -c "SELECT * FROM Personnel_org_chart"
Password for user postgres:
 emp_nbr | emp      | boss_emp_nbr | boss
-----+-----+-----+-----
      1 | Иван     |              |
      2 | Петр     |              |
      3 | Антон    |              |
      4 | Захар    |              |
      5 | Ирина    |              |
      6 | Анна     |              |
      7 | Андрей   |              |
      8 | Николай  |              |
(8 rows)
```

```
papey08@papey08-Macbook ~ % psql -h localhost -p 5432 -U postgres -d ais -c "SELECT * FROM Create_paths"
Password for user postgres:
 level1 | level2 | level3 | level4
-----+-----+-----+-----
Иван    | Антон  | Ирина  | Андрей
Иван    | Антон  | Ирина  | Николай
Иван    | Петр   |        |
Иван    | Антон  | Захар  |
Иван    | Антон  | Анна   |
(5 rows)
```

Задание 13

13. Выполните проверку структуры дерева на предмет отсутствия циклов с помощью функции `tree_test()`.

```
SELECT * FROM tree_test();
```

Если вы еще не вносили изменения в таблицу «Организационная структура», то функция покажет отсутствие нарушения структуры дерева. Теперь создайте в таблице «Организационная структура» сначала короткий цикл, а затем длинный цикл. Для каждого из указанных циклов выполните проверку с помощью функции `tree_test()`.

Запрос

```
SELECT * FROM tree_test();
```

```
BEGIN;
```

```

UPDATE org_chart
SET boss_emp_nbr = 3
WHERE emp_nbr = 2;

UPDATE org_chart
SET boss_emp_nbr = 2
WHERE emp_nbr = 3;

SELECT * FROM tree_test();
ROLLBACK;

BEGIN;
UPDATE org_chart
SET boss_emp_nbr = 5
WHERE emp_nbr = 4;

UPDATE org_chart
SET boss_emp_nbr = 6
WHERE emp_nbr = 5;

UPDATE org_chart
SET boss_emp_nbr = 4
WHERE emp_nbr = 6;

SELECT * FROM tree_test();
ROLLBACK;

```

Результат

Первая проверка (нет циклов):

	tree_test ▾
1	Tree

Вторая проверка (короткий цикл):

	tree_test ▾
1	Cycles

Третья проверка (длинный цикл):

	tree_test ▾
1	Cycles

Задание 14

14. Выполните обход дерева организационной структуры снизу вверх, начиная с конкретного узла, можно с помощью функции `up_tree_traversal()` либо функции `up_tree_traversal2()`. Сначала сделайте это с помощью первой из функций:

```
SELECT * FROM up_tree_traversal( 6 );
```

Параметром этих функций является код работника. Измените код работника и повторите команду.

Теперь воспользуйтесь второй функцией. Учтите, что она возвращает SETOF RECORD, поэтому команда будет более сложной:

```
SELECT * FROM up_tree_traversal2( 6 ) AS (emp int, boss int);
```

Очевидно, что для использования числового кода работника нужно знать этот код. Удобнее иметь дело с именем работника. Поэтому можно в качестве параметра этих функций использовать подзапрос, возвращающий код работника в качестве своего результата. Не забудьте, что текст подзапроса заключается в скобки, поэтому появляются двойные скобки:

```
SELECT * FROM up_tree_traversal( ( SELECT ... FROM Personnel  
WHERE ... ) );
```

Завершите эту команду и выполните ее с различными именами работников.

Запрос

```
SELECT * FROM up_tree_traversal(6);
```

Результат

	emp_nbr	boss_emp_nbr
1	6	3
2	3	1
3	1	<null>

Запрос

```
SELECT * FROM up_tree_traversal(4);
```

Результат

	emp_nbr	boss_emp_nbr
1	4	3
2	3	1
3	1	<null>

Запрос

```
SELECT * FROM up_tree_traversal2(6) AS (emp int, boss int);
```

Результат

	emp	boss
1	6	3
2	3	1
3	1	<null>

Запрос

```
SELECT * FROM up_tree_traversal((SELECT emp_nbr FROM personnel  
WHERE emp_name = 'Иван'));
```

Результат

	emp_nbr	boss_emp_nbr
1	1	<null>

Запрос

```
SELECT * FROM up_tree_traversal((SELECT emp_nbr FROM personnel  
WHERE emp_name = 'Захар'));
```

Результат

	emp_nbr		boss_emp_nbr	
1		4		3
2		3		1
3		1		<null>

Задание 15

15. Выполните операцию удаления поддерева с помощью функции `delete_subtree()`. Параметром функции является код работника.

```
SELECT * FROM delete_subtree( 6 );
```

Аналогично работе с функцией `up_tree_traversal()` используйте подзапрос для получения кода работника по его имени. После удаления поддерева посмотрите, что стало с организационной структурой, с помощью двух представлений `Personnel_org_chart` и `Create_paths`.

Запрос

```
SELECT * FROM personnel_org_chart;  
SELECT * FROM create_paths;
```

Результат

	emp_nbr		emp		boss_emp_nbr		boss	
1		1	Иван			<null>	<null>	
2		2	Петр			1	Иван	
3		3	Антон			1	Иван	
4		4	Захар			3	Антон	
5		5	Ирина			3	Антон	
6		6	Анна			3	Антон	
7		7	Андрей			5	Ирина	
8		8	Николай			5	Ирина	

	level1 ↕	level2 ↕	level3 ↕	level4 ↕
1	Иван	Антон	Ирина	Андрей
2	Иван	Антон	Ирина	Николай
3	Иван	Петр	<null>	<null>
4	Иван	Антон	Захар	<null>
5	Иван	Антон	Анна	<null>

Запрос

```
BEGIN;
SELECT * FROM delete_subtree(6);
SELECT * FROM personnel_org_chart;
SELECT * FROM create_paths;
ROLLBACK;
```

Результат

	emp_nbr ↕	emp ↕	boss_emp_nbr ↕	boss ↕
1	1	Иван	<null>	<null>
2	2	Петр	1	Иван
3	3	Антон	1	Иван
4	4	Захар	3	Антон
5	5	Ирина	3	Антон
6	7	Андрей	5	Ирина
7	8	Николай	5	Ирина

	level1 ↕	level2 ↕	level3 ↕	level4 ↕
1	Иван	Антон	Ирина	Андрей
2	Иван	Антон	Ирина	Николай
3	Иван	Петр	<null>	<null>
4	Иван	Антон	Захар	<null>

Запрос

```
BEGIN;
SELECT * FROM delete_subtree((SELECT emp_nbr FROM personnel
WHERE emp_name = 'Захар'));
SELECT * FROM personnel_org_chart;
SELECT * FROM create_paths;
ROLLBACK;
```

Результат

	emp_nbr	emp	boss_emp_nbr	boss
1	1	Иван	<null>	<null>
2	2	Петр	1	Иван
3	3	Антон	1	Иван
4	5	Ирина	3	Антон
5	6	Анна	3	Антон
6	7	Андрей	5	Ирина
7	8	Николай	5	Ирина

	level1	level2	level3	level4
1	Иван	Антон	Ирина	Андрей
2	Иван	Антон	Ирина	Николай
3	Иван	Петр	<null>	<null>
4	Иван	Антон	Анна	<null>

Задание 16

16. Если в таблице «Организационная структура» осталось мало данных, то дополните ее данными и выполните удаление элемента иерархии и продвижение дочерних элементов на один уровень вверх (т. е. к «бабушке»).

```
SELECT * FROM delete_and_promote_subtree( 5 );
```

Аналогично работе с функцией `up_tree_traversal()` используйте подзапрос для получения кода работника по его имени.

После удаления элемента иерархии посмотрите, что стало с организационной структурой, с помощью двух представлений `Personnel_org_chart` и `Create_paths`.

Запрос

```
BEGIN;  
SELECT * FROM delete_and_promote_subtree(5);  
SELECT * FROM personnel_org_chart;  
SELECT * FROM create_paths;  
ROLLBACK;
```


Результат

	emp_nbr ↕	emp ↕	boss_emp_nbr ↕	boss ↕
1	1	Иван	<null>	<null>
2	2	Петр	1	Иван
3	3	Антон	1	Иван
4	4	Захар	3	Антон
5	6	Анна	3	Антон
6	7	Андрей	3	Антон
7	8	Николай	3	Антон

	level1 ↕	level2 ↕	level3 ↕	level4 ↕
1	Иван	Петр	<null>	<null>
2	Иван	Антон	Захар	<null>
3	Иван	Антон	Николай	<null>
4	Иван	Антон	Анна	<null>
5	Иван	Антон	Андрей	<null>

Запрос

```
BEGIN;  
SELECT * FROM delete_and_promote_subtree((SELECT emp_nbr FROM  
personnel WHERE emp_name = 'Анна'));  
SELECT * FROM personnel_org_chart;  
SELECT * FROM create_paths;  
ROLLBACK;
```

Результат

	emp_nbr ↕	emp ↕	boss_emp_nbr ↕	boss ↕
1	1	Иван	<null>	<null>
2	2	Петр	1	Иван
3	3	Антон	1	Иван
4	4	Захар	3	Антон
5	5	Ирина	3	Антон
6	7	Андрей	5	Ирина
7	8	Николай	5	Ирина

	level1	level2	level3	level4
1	Иван	Антон	Ирина	Андрей
2	Иван	Антон	Ирина	Николай
3	Иван	Петр	<null>	<null>
4	Иван	Антон	Захар	<null>

Задание 17

17. Представление `Create_paths` позволяет отобразить только четыре уровня иерархии. Модифицируйте его так, чтобы оно могло работать с пятью уровнями иерархии.

Запрос

```
CREATE VIEW Create_paths ( level1, level2, level3, level4,
level5 ) AS
  SELECT O1.emp AS e1, O2.emp AS e2, O3.emp AS e3, O4.emp AS e4,
O5.emp AS e5
  FROM Personnel_org_chart AS O1
 LEFT OUTER JOIN Personnel_org_chart AS O2 ON O1.emp = O2.boss
 LEFT OUTER JOIN Personnel_org_chart AS O3 ON O2.emp = O3.boss
 LEFT OUTER JOIN Personnel_org_chart AS O4 ON O3.emp = O4.boss
 LEFT OUTER JOIN Personnel_org_chart AS O5 ON O4.emp = O5.boss
 WHERE O1.emp = 'Иван';
```

Результат

	level1	level2	level3	level4	level5
1	Иван	Антон	Анна	<null>	<null>
2	Иван	Антон	Захар	<null>	<null>
3	Иван	Петр	<null>	<null>	<null>
4	Иван	Антон	Ирина	Николай	<null>
5	Иван	Антон	Ирина	Андрей	<null>

Запрос

```
--- добавляем сотрудника в иерархию
INSERT INTO personnel (emp_nbr, emp_name, address, birth_date)
VALUES (9, 'Владислав', 'просп. Реляционных СУБД', '2000-01-
01');

INSERT INTO org_chart (job_title, emp_nbr, boss_emp_nbr, salary)
VALUES ('Бэкендер', 9, 8, 300);
```

```
SELECT * FROM create_paths;
```

Результат

	level1 ▾	level2 ▾	level3 ▾	level4 ▾	level5 ▾
1	Иван	Антон	Ирина	Николай	Владислав
2	Иван	Антон	Анна	<null>	<null>
3	Иван	Антон	Захар	<null>	<null>
4	Иван	Петр	<null>	<null>	<null>
5	Иван	Антон	Ирина	Андрей	<null>

Задание 18

18. Самостоятельно ознакомьтесь с таким средством работы с таблицами базы данных, как курсоры (cursors). Воспользуйтесь технической документацией на PostgreSQL, глава «PL/pgSQL – SQL Procedural Language». Напишите небольшую функцию с применением курсора.

Запрос

```
CREATE OR REPLACE FUNCTION get_addresses()
RETURNS VOID AS $$
    DECLARE cursor CURSOR FOR
        SELECT emp_name, address FROM personnel;
    record RECORD;
    BEGIN
        OPEN cursor;
        LOOP
            FETCH cursor INTO record;
            EXIT WHEN NOT FOUND;
            RAISE NOTICE 'name: %, address: %', record.emp_name,
record.address;
        END LOOP;
        CLOSE cursor;
    END;
$$ LANGUAGE plpgsql;

SELECT get_addresses();
```

Результат

```
ais.public> SELECT get_addresses()
name: вакансия, address:
name: Иван, address: ул. Любителей языка C
name: Петр, address: ул. UNIX гуру
name: Антон, address: ул. Ассемблерная
name: Захар, address: ул. им. СУБД PostgreSQL
name: Ирина, address: просп. Программистов
name: Анна, address: пер. Перловый
name: Андрей, address: пл. Баз данных
name: Николай, address: наб. ОС Linux
name: Владислав, address: просп. Реляционных СУБД
```