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.

```
ais=# SELECT * FROM personnel;
 emp_nbr | emp_name |         address          | birth_date
---------+----------+--------------------------+------------
       0 | вакансия |                          | 2014-05-19
       1 | Иван     | ул. Любителей языка C    | 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)
```

```
ais=# SELECT * FROM create_paths;
 level1 | level2 | level3 | level4
--------+--------+--------+---------
 Иван   | Антон  | Ирина  | Андрей
 Иван   | Антон  | Ирина  | Николай
 Иван   | Петр   |        |
 Иван   | Антон  | Захар  |
 Иван   | Антон  | Анна   |
(5 rows)
```

```
ais=# SELECT * FROM org_chart;
      job_title      | emp_nbr | boss_emp_nbr |  salary
---------------------+---------+--------------+-----------
 Президент           |       1 |              | 1000.0000
 Вице-президент 1    |       2 |            1 |  900.0000
 Вице-президент 2    |       3 |            1 |  800.0000
 Архитектор          |       4 |            3 |  700.0000
 Ведущий программист |       5 |            3 |  600.0000
 Программист C       |       6 |            3 |  500.0000
 Программист Perl    |       7 |            5 |  450.0000
 Оператор            |       8 |            5 |  400.0000
(8 rows)
```

```
ais=# SELECT * FROM personnel_org_chart;
 emp_nbr |   emp    | boss_emp_nbr | boss
---------+----------+--------------+-------
       1 | Иван     |              |
       2 | Петр     |            1 | Иван
       3 | Антон    |            1 | Иван
       4 | Захар    |            3 | Антон
       5 | Ирина    |            3 | Антон
       6 | Анна     |            3 | Антон
       7 | Андрей   |            5 | Ирина
       8 | Николай  |            5 | Ирина
(8 rows)
```

```
ais=# CREATE VIEW names AS SELECT emp_name FROM personnel WHERE emp_name LIKE 'A%';
CREATE VIEW
ais=# SELECT * FROM names;
 emp_name
----------
 Антон
 Анна
 Андрей
(3 rows)
```

```
ais=# CREATE VIEW job_with_p AS SELECT job_title FROM org_chart WHERE job_title LIKE 'П%';
CREATE VIEW
ais=# SELECT * FROM job_with_p;
    job_title
-----------------
 Президент
 Программист C
 Программист Perl
(3 rows)
```

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().

```
ais=# UPDATE org_chart SET boss_emp_nbr=4 WHERE emp_nbr=3;
UPDATE 1
ais=# SELECT * FROM tree_test();
 tree_test
-----------
 Cycles
(1 row)
```

```
ais=# UPDATE org_chart SET boss_emp_nbr=7 WHERE emp_nbr=3;
UPDATE 1
ais=# SELECT * FROM tree_test();
 tree_test
-----------
 Cycles
(1 row)
```

циклов выполните проверку с помощью функции tree_test().

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.

```
ais=# SELECT * FROM up_tree_traversal(6);
 emp_nbr | boss_emp_nbr
---------+--------------
       6 |            3
       3 |            1
       1 |
(3 rows)
```

```
ais=# SELECT * FROM up_tree_traversal(7);
 emp_nbr | boss_emp_nbr
---------+--------------
       7 |            5
       5 |            3
       3 |            1
       1 |
(4 rows)
```

```
ais=# SELECT * FROM up_tree_traversal2( 6 ) AS (emp int, boss int);
 emp | boss
-----+------
   6 |    3
   3 |    1
   1 |
(3 rows)
```

```
ais=# SELECT * FROM up_tree_traversal( ( SELECT emp_nbr FROM Personnel
WHERE emp_name='Анна') );
 emp_nbr | boss_emp_nbr
---------+--------------
       6 |            3
       3 |            1
       1 |
(3 rows)
```

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.

```
ais=# SELECT * FROM create_paths;
 level1 | level2 | level3 | level4
--------+--------+--------+---------
 Иван   | Антон  | Ирина  | Андрей
 Иван   | Антон  | Ирина  | Николай
 Иван   | Петр   |        |
 Иван   | Антон  | Захар  |
(4 rows)

ais=# SELECT * FROM personnel_org_chart;
 emp_nbr |   emp   | boss_emp_nbr | boss
---------+---------+--------------+-------
       1 | Иван    |              |
       4 | Захар   |            3 | Антон
       5 | Ирина   |            3 | Антон
       7 | Андрей  |            5 | Ирина
       8 | Николай |            5 | Ирина
       2 | Петр    |            1 | Иван
       3 | Антон   |            1 | Иван
(7 rows)

ais=# SELECT * FROM delete_subtree(4);
 delete_subtree
```

```
ais=# SELECT * FROM create_paths;
 level1 | level2 | level3 | level4
--------+--------+--------+---------
 Иван   | Антон  | Ирина  | Андрей
 Иван   | Антон  | Ирина  | Николай
 Иван   | Петр   |        |
(3 rows)

ais=# SELECT * FROM personnel_org_chart;
 emp_nbr |   emp   | boss_emp_nbr | boss
---------+---------+--------------+-------
       1 | Иван    |              |
       5 | Ирина   |            3 | Антон
       7 | Андрей  |            5 | Ирина
       8 | Николай |            5 | Ирина
       2 | Петр    |            1 | Иван
       3 | Антон   |            1 | Иван
(6 rows)
```

```
ais=# SELECT * FROM delete_subtree((SELECT emp_nbr FROM personnel WHERE emp_name='Антон'));
 delete_subtree
----------------

(1 row)
```

```
ais=# SELECT * FROM create_paths;
 level1 | level2 | level3 | level4
--------+--------+--------+--------
 Иван   | Петр   |        |
(1 row)

ais=# SELECT * FROM personnel_org_chart;
 emp_nbr | emp  | boss_emp_nbr | boss
---------+------+--------------+------
       1 | Иван |              |
       2 | Петр |            1 | Иван
(2 rows)
```

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.

```
ais=# SELECT * FROM delete_and_promote_subtree( 5 );
 delete_and_promote_subtree
----------------------------

(1 row)

ais=# SELECT * FROM personnel_org_chart;
 emp_nbr |   emp    | boss_emp_nbr | boss
---------+----------+--------------+-------
       1 | Иван     |              |
       2 | Петр     |            1 | Иван
       3 | Антон    |            1 | Иван
       4 | Захар    |            3 | Антон
       6 | Анна     |            3 | Антон
       7 | Андрей   |            3 | Антон
       8 | Николай  |            3 | Антон
(7 rows)

ais=# SELECT * FROM create_paths;
 level1 | level2 | level3  | level4
--------+--------+---------+--------
 Иван   | Петр   |         |
 Иван   | Антон  | Захар   |
 Иван   | Антон  | Николай |
 Иван   | Антон  | Анна    |
 Иван   | Антон  | Андрей  |
(5 rows)
```

```
ais=# SELECT * FROM delete_and_promote_subtree( (SELECT emp_nbr FROM personnel WHERE emp_name='Анна') );
 delete_and_promote_subtree
----------------------------

(1 row)

ais=# SELECT * FROM personnel_org_chart;
 emp_nbr |   emp    | boss_emp_nbr | boss
---------+----------+--------------+-------
       1 | Иван     |              |
       2 | Петр     |            1 | Иван
       3 | Антон    |            1 | Иван
       4 | Захар    |            3 | Антон
       7 | Андрей   |            3 | Антон
       8 | Николай  |            3 | Антон
(6 rows)

ais=# SELECT * FROM create_paths;
 level1 | level2 | level3  | level4
--------+--------+---------+--------
 Иван   | Петр   |         |
 Иван   | Антон  | Захар   |
 Иван   | Антон  | Николай |
 Иван   | Антон  | Андрей  |
(4 rows)
```

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

```
ais=# CREATE VIEW create_paths_5 (level1, level2, level3, level4, level5) AS
ELECais-# SELECT a1.emp AS e1, a2.emp AS e2, a3.emp AS e3, a4.emp AS e4, a5.emp AS e5
ais-# FROM personnel_org_chart as a1
ais-#   LEFT OUTER JOIN personnel_org_chart as a2
ais-#                ON a1.emp = a2.boss
ais-#   LEFT OUTER JOIN personnel_org_chart as a3
ais-#                ON a2.emp = a3.boss
ais-#   LEFT OUTER JOIN  personnel_org_chart AS a4
ais-#                ON a3.emp = a4.boss
ais-#   LEFT OUTER JOIN  personnel_org_chart as a5
ais-#                ON a4.emp = a5.boss
ais-#   WHERE a1.emp = 'Иван';
CREATE VIEW
ais=# SELECT * FROM create_paths_5;
 level1 | level2 | level3 | level4  | level5
--------+--------+--------+---------+--------
 Иван   | Антон  | Анна   |         |
 Иван   | Антон  | Захар  |         |
 Иван   | Петр   |        |         |
 Иван   | Антон  | Ирина  | Николай |
 Иван   | Антон  | Ирина  | Андрей  |
(5 rows)
```

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.

```
ais=# CREATE TEMP TABLE tmp AS SELECT * FROM personnel AS DATA;
SELECT 9
```

```
ais=# ALTER TABLE tmp ADD COLUMN check text;

LINE 1: ALTER TABLE tmp ADD COLUMN check text;
                                    ^


ALTER TABLE
```

```
ais=# CREATE OR replace FUNCTION test() RETURNS void AS $$ DECLARE curs CURSOR FOR SELECT * FROM tmp;
ais$# BEGIN
ais$# OPEN curs;
ais$# MOVE curs;
ais$# WHILE FOUND LOOP
ais$# UPDATE tmp SET check_emp='yes' WHERE current of curs;
ais$# MOVE curs;
ais$# END loop;
ais$# CLOSE curs;
ais$#
ais$# END
ais$# $$
ais-# LANGUAGE plpgsql;
CREATE FUNCTION
ais=# SELECT * FROM test();
 test
------

(1 row)
```

```
ais=# SELECT * FROM tmp;
 emp_nbr | emp_name |         address          | birth_date | check_emp
---------+----------+--------------------------+------------+-----------
       0 | вакансия |                          | 2014-05-19 | yes
       1 | Иван     | ул. Любителей языка C    | 1962-12-01 | yes
       2 | Петр     | ул. UNIX гуру            | 1965-10-21 | yes
       3 | Антон    | ул. Ассемблерная         | 1964-04-17 | yes
       4 | Захар    | ул. им. СУБД PostgreSQL  | 1963-09-27 | yes
       5 | Ирина    | просп. Программистов     | 1968-05-12 | yes
       6 | Анна     | пер. Перловый            | 1969-03-20 | yes
       7 | Андрей   | пл. Баз данных           | 1945-11-07 | yes
       8 | Николай  | наб. ОС Linux            | 1944-12-01 | yes
(9 rows)

ais=#
```