МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ

(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)

Институт №8 «Компьютерные науки и прикладная математика»

Кафедра 806 «Вычислительная математика и программирование»

**Лабораторная работа №10**

**по курсу «Проектирование баз данных»**

Выполнила: Прудникова А. А.

Группа: М8О-114СВ-24

Преподаватель: Моргунов Е. П.

Москва, 2024

## Задание 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 -с "SELECT \* FROM Personnel"

psql -d ais -с "SELECT \* FROM Org\_chart"

psql -d ais -с "SELECT \* FROM Personnel\_org\_chart"

psql -d ais -с "SELECT \* FROM Create\_paths"

### Запрос

psql -h localhost -p 5432 -U postgres -d ais -c "SELECT

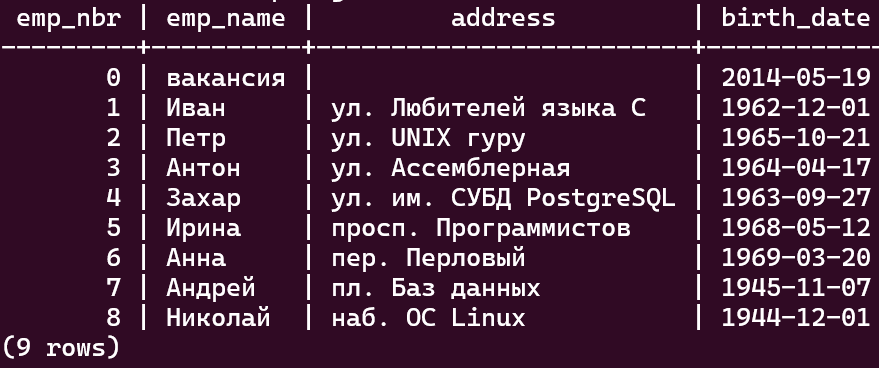
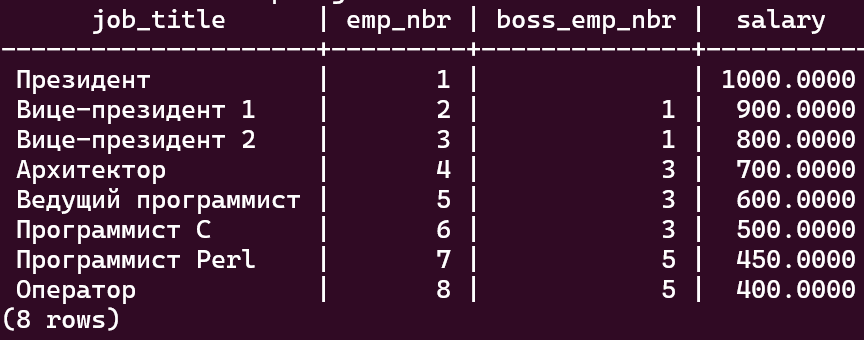
\* FROM Personnel"

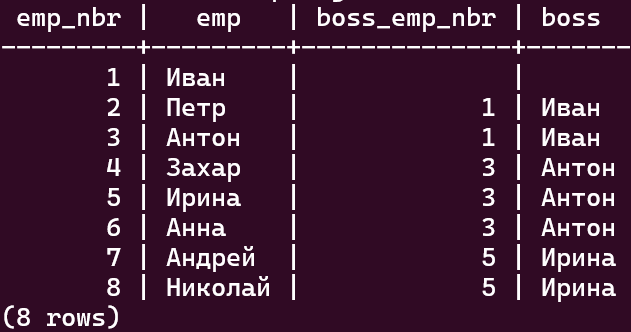
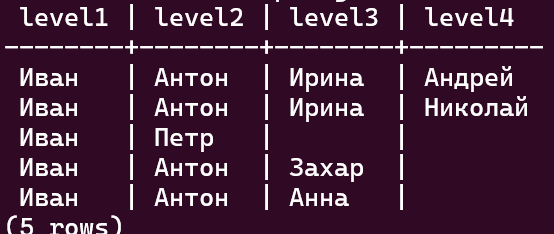
psql -h localhost -p 5432 -U postgres -d ais -c "SELECT \* FROM Org\_chart"

psql -h localhost -p 5432 -U postgres -d ais -c "SELECT \* FROM Personnel\_org\_chart"

psql -h localhost -p 5432 -U postgres -d ais -c "SELECT \* FROM Create\_paths"

### Результат

## Задание 13

Выполните проверку структуры дерева на предмет отсутствия циклов с помощью функции 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();

-- Иван — начальник Петра

UPDATE Org\_chart

SET boss\_emp\_nbr = 1

WHERE emp\_nbr = 2;

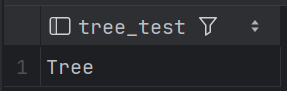
-- Петр — начальник Ивана

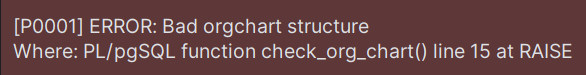
UPDATE Org\_chart

SET boss\_emp\_nbr = 2

WHERE emp\_nbr = 1;

### Результат





### Решение

Сработал триггер. Отключим его и посмотрим, что будет со структурами.

### Запрос

ALTER TABLE Org\_chart DISABLE TRIGGER check\_org\_chart;

-- Иван — начальник Петра

UPDATE Org\_chart

SET boss\_emp\_nbr = 1

WHERE emp\_nbr = 2;

-- Петр — начальник Ивана

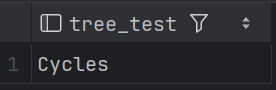
UPDATE Org\_chart

SET boss\_emp\_nbr = 2

WHERE emp\_nbr = 1;

SELECT \* FROM tree\_test();

### Результат



### Решение

Проделаем теперь то же самое для длинного цикла.

### Запрос

UPDATE Org\_chart

SET boss\_emp\_nbr = 1

WHERE emp\_nbr = 2;

UPDATE Org\_chart

SET boss\_emp\_nbr = 2

WHERE emp\_nbr = 3;

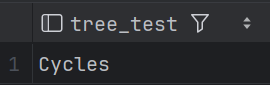
UPDATE Org\_chart

SET boss\_emp\_nbr = 3

WHERE emp\_nbr = 1;

SELECT \* FROM tree\_test();

### Результат



### Решение

Вернем структуру дерева для дальнейшей работы.

### Запрос

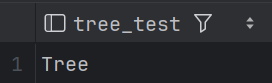
UPDATE Org\_chart

SET boss\_emp\_nbr = NULL

WHERE emp\_nbr = 1;

SELECT \* FROM tree\_test();

### Результат



## Задание 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);

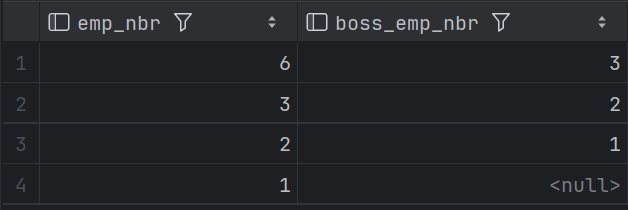
SELECT \* FROM up\_tree\_traversal(2);

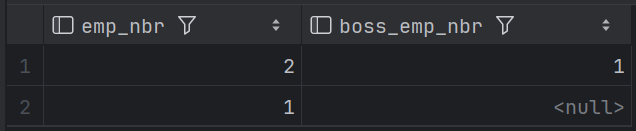
SELECT \* FROM up\_tree\_traversal2(6) AS (emp int, boss int);

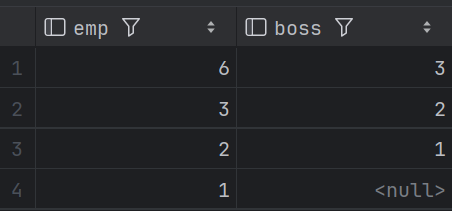
SELECT \* FROM up\_tree\_traversal((SELECT emp\_nbr FROM Personnel WHERE emp\_name = 'Анна'));

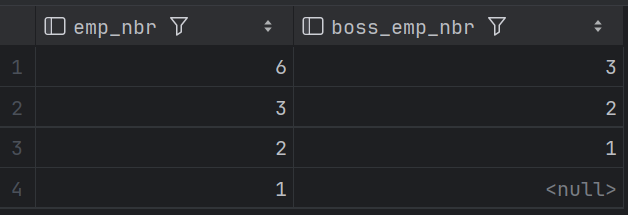
SELECT \* FROM up\_tree\_traversal((SELECT emp\_nbr FROM Personnel WHERE emp\_name = 'Иван'));

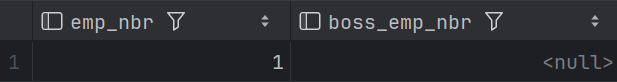
### Результат











## Задание 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

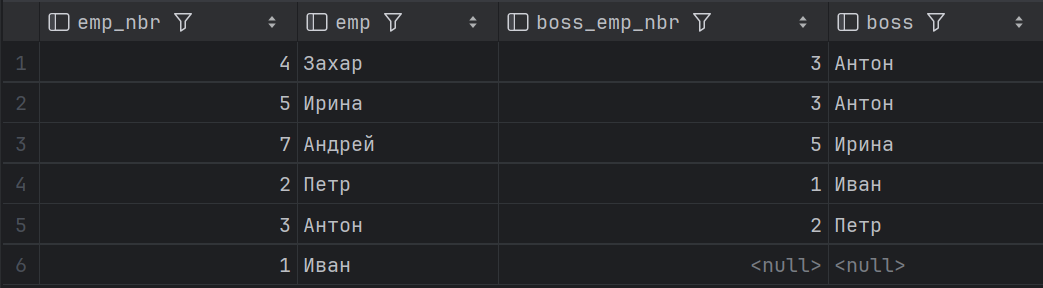
SELECT \* FROM delete\_subtree(6);

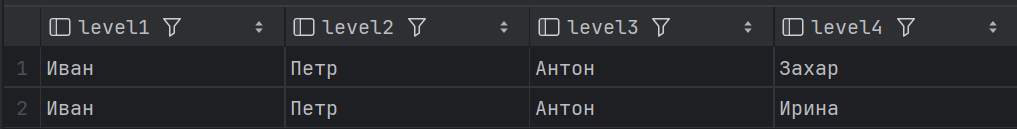
SELECT \* FROM delete\_subtree((SELECT emp\_nbr FROM Personnel WHERE emp\_name = 'Николай'));

SELECT \* FROM Personnel\_org\_chart;

SELECT \* FROM Create\_paths;

### Результат





## Задание 16

Если в таблице «Организационная структура» осталось мало данных, дополните ее и выполните удаление элемента иерархии с продвижением 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 проверьте, что произошло с организационной структурой, с помощью двух представлений: Personnel\_org\_chart и Create\_paths.

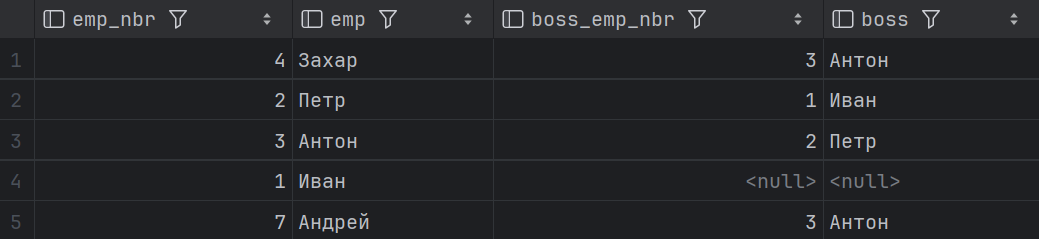
### Запрос

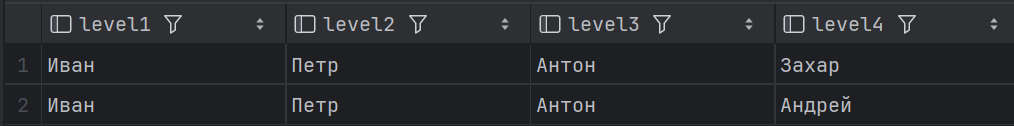
SELECT \* FROM delete\_and\_promote\_subtree((SELECT emp\_nbr FROM Personnel WHERE emp\_name = 'Ирина'));

SELECT \* FROM Personnel\_org\_chart;

SELECT \* FROM Create\_paths;

### Результат





## Задание 17

Представление Create\_paths позволяет отобразить только четыре уровня иерархии. Модифицируйте его так, чтобы оно могло работать с пятью уровнями иерархии.

### Решение

Для начала дополним таблицу данными.

### Запрос

INSERT INTO personnel (emp\_nbr, emp\_name, address, birth\_date)

VALUES

(10, 'Вадим', 'ул. Программистов', '1980-05-15'),

(11, 'Максим', 'ул. Кодеров', '1982-06-17'),

(12, 'Анастасия', 'пер. Фронта', '1985-03-10'),

(13, 'Матвей', 'ул. Бэкенда', '1987-09-22');

INSERT INTO Org\_chart (emp\_nbr, job\_title, boss\_emp\_nbr, salary)

VALUES

(10, 'Программист', 7, 120.00),

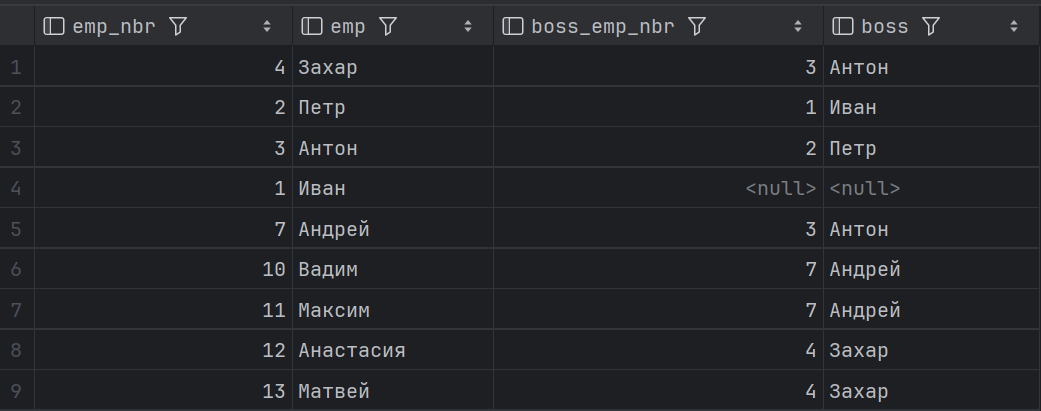
(11, 'HR', 7, 300.00),

(12, 'Дизайнер интерфейса', 4, 350.00),

(13, 'Тестировщик', 4, 500.00);

SELECT \* FROM personnel\_org\_chart;

### Результат



### Запрос

DROP VIEW IF EXISTS Create\_paths;

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

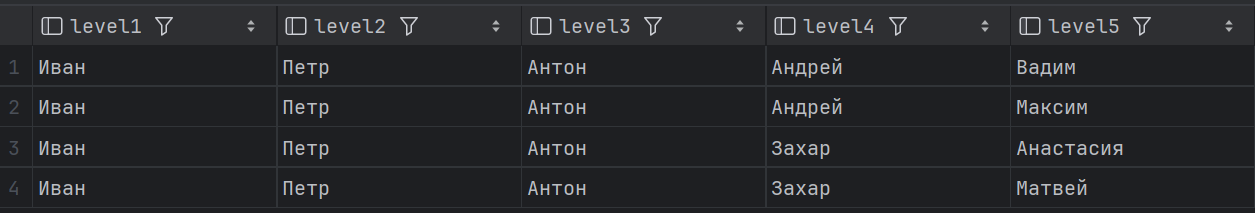
LEFT OUTER JOIN Personnel\_org\_chart AS O5 -- Изменено с O4 на O5

ON O4.emp = O5.boss

WHERE O1.emp = 'Иван';

SELECT \* FROM Create\_paths;

### Результат



## Задание 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 process\_salaries()

RETURNS VOID AS $$

DECLARE

e\_name TEXT;

emp\_salary NUMERIC;

cursor\_salary CURSOR FOR

SELECT (SELECT emp\_name FROM personnel WHERE personnel.emp\_nbr = org\_chart.emp\_nbr), salary

FROM org\_chart

WHERE salary > 500;

BEGIN

OPEN cursor\_salary;

LOOP

FETCH cursor\_salary INTO e\_name, emp\_salary;

EXIT WHEN NOT FOUND;

RAISE NOTICE 'Employee: %, Salary: %', e\_name, emp\_salary;

END LOOP;

CLOSE cursor\_salary;

END;

$$ LANGUAGE plpgsql;

SELECT \* FROM process\_salaries();

### Результат

