Министерство науки и высшего образования Российской Федерации
федеральное государственное автономное образовательное учреждение высшего образования
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»

Отчет

ПО ЛАБОРАТОРНОЙ РАБОТЕ №2

«ЗАПРОСЫ НА ВЫБОРКУ И МОДИФИКАЦИЮ ДАННЫХ, ПРЕДСТАВЛЕНИЯ И ИНДЕКСЫ В POSTGRESQL»

Автор: Кононов Степан Владимирович

Факультет: ИКТ

Группа: К32392

Преподаватель: Говорова М. М.

Дата: 02.05.2023

Санкт-Петербург 2023

Лабораторная №2

Цель работы:

Овладеть практическими навыками создания представлений и запросов на выборку данных к базе данных PostgreSQL, использования подзапросов при модификации данных и индексов.

Практическое задание:

1. Создать запросы и представления на выборку данных к базе данных PostgreSQL (согласно индивидуальному заданию, часть 2 и 3).
2. Составить 3 запроса на модификацию данных (INSERT, UPDATE, DELETE) с использованием подзапросов.
3. Изучить графическое представление запросов и посмотреть историю запросов.
4. Создать простой и составной индексы для двух произвольных запросов и сравнить время выполнения запросов без индексов и с индексами. Для получения плана запроса использовать команду EXPLAIN.

Индивидуальное практическое задание: «Учет выполнения заданий»

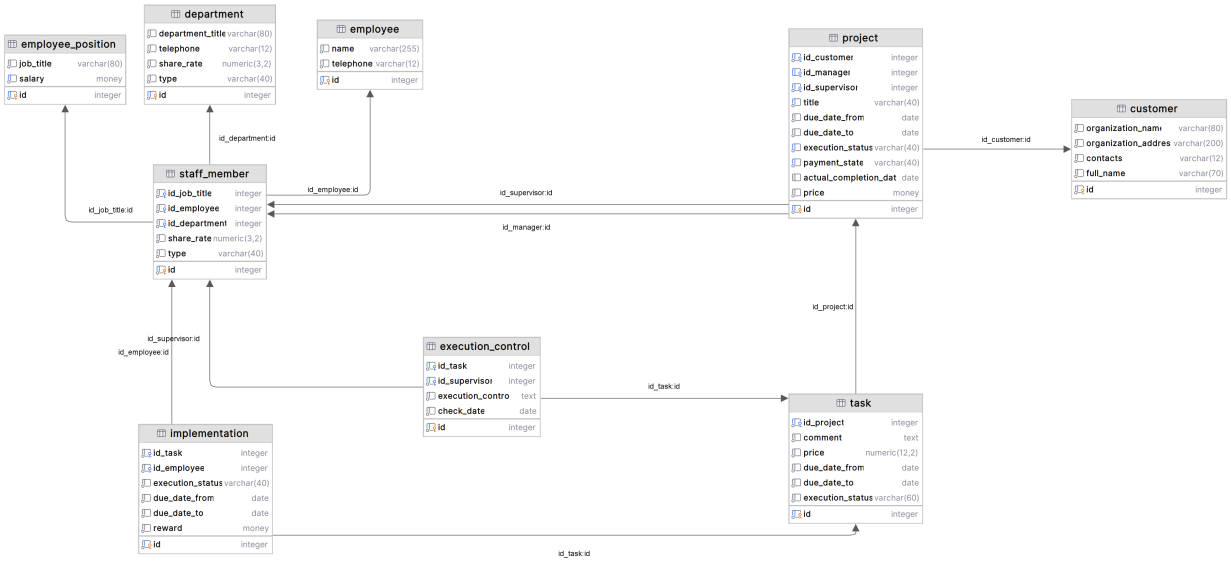
Запросы:

- Составить список всех заданий каждого проекта с указанием организаций, отделов и исполнителей, занятых в его выполнении.
- Составить список проектов, работа над которыми была начата больше месяца назад.
- Вывести список сотрудников, оклад которых превышает средний оклад сотрудников своего отдела.
- Найти отдел, работающий над максимальным количеством проектов.
- Составить список сотрудников, проектов, заданий, в выполнении которых они участвуют и дат предполагаемого выполнения ими заданий. Учесть сотрудников, не участвующих в проектах.
- Составить список сотрудников, не выполнивших задания в срок с указанием проектов и заданий, которые они должны были выполнить и количества дней просрочки выполнения заданий.
- Составить список проектов, в выполнении которого участвует более трех человек.

Представления:

- для руководителей проектов, содержащее сведения об исполнителях, отделах, сроках выполнения заданий, включенных в проект.
- список проектов, срок выполнения которых истекает сегодня и которые включают больше трех невыполненных заданий.

Схема базы данных:



Запросы

- Составить список всех заданий каждого проекта с указанием организаций, отделов и исполнителей, занятых в его выполнении.

```
select task.id as task_id, e.name as employee_name, d.department_title
from task
         join implementation i on task.id = i.id_task
         join staff_member sm on i.id_employee = sm.id
         join department d on d.id = sm.id_department
         join employee e on e.id = sm.id_employee
order by task_id
```

	task_id	employee_name	department_title
1	5	Beatrice Strickland	Department 9
2	5	Tammy Lindley	Department 3
3	6	Charles Fox	Department 8
4	8	Bryan Rooney	Department 3
5	8	William Hairr	Department 1

- Составить список проектов, работа над которыми была начата больше месяца назад.

```
SELECT project.id, project.title, project.due_date_to
FROM project
WHERE due_date_to <= NOW() - INTERVAL '10 year'
```

	id	title	due_date_to
1	2	Labore quisquam tempora aliquam consecte	2007-07-24
2	5	Dolor ipsum aliquam ut non quiquia.	2008-08-03
3	8	Est porro dolor neque dolore quaerat.	2013-04-18
4	10	Neque quaerat sed magnam est ipsum eius	2009-05-10
5	12	Voluptatem eius est quaerat dolor dolor	2007-02-21

- Вывести список сотрудников, оклад которых превышает средний оклад сотрудников своего отдела.

```
SELECT e.name, ep.salary, d.department_title AS department_name
FROM staff_member AS sm
         JOIN employee AS e ON sm.id_employee = e.id
         JOIN employee_position AS ep ON sm.id_job_title = ep.id
         JOIN department AS d ON sm.id_department = d.id
WHERE ep.salary::numeric > (SELECT AVG(ep2.salary::numeric)
                            FROM staff_member AS sm2
                                     JOIN employee_position AS ep2 ON sm2.id_job_title = ep2.id
                            WHERE sm2.id_department = sm.id_department)::numeric
ORDER BY d.department_title, e.name;
```

	name	salary	department_name
1	Charles Clifton	4797.58	Department 1
2	Charles Roberts	1586.56	Department 1
3	Charlotte Caspers	4478.78	Department 1
4	Cory Smith	3527.1	Department 1
5	Danielle Guidry	4738.48	Department 1
6	David Smith	4788.17	Department 1
7	Dorothy Crain	3938.1	Department 1

- Найти отдел, работающий над максимальным количеством проектов.

```
select department.id, department.department_title, count(p.id) as project_count
from department
         join staff_member sm on department.id = sm.id_department
         join implementation i on sm.id = i.id_employee
         join task t on i.id_task = t.id
         join project p on p.id = t.id_project
group by department.id
order by count(p.id) desc
limit 1
```

	id	department_title	project_count
1	269	Department 6	10

```
SELECT d.department_title, d.id
FROM department d
         JOIN
     (SELECT d.id, count(DISTINCT p.id) as projects_count
      FROM staff_member s
```

```
              INNER JOIN department d ON s.id_department = d.id
              INNER JOIN implementation i ON s.id = i.id_employee
              INNER JOIN task t ON i.id_task = t.id
              INNER JOIN project p ON t.id_project = p.id
      GROUP BY d.id) as department_projects ON d.id = department_projects.id
WHERE department_projects.projects_count = (SELECT MAX(projects_count) as max_projects_count
                                            FROM (SELECT d.id, count(DISTINCT p.id) as projects_count
                                                  FROM staff_member s
                                                           INNER JOIN department d ON s.id_department = d.id
                                                           INNER JOIN implementation i ON s.id = i.id_employee
                                                           INNER JOIN task t ON i.id_task = t.id
                                                           INNER JOIN project p ON t.id_project = p.id
                                                  GROUP BY d.id) as department_projects)
```

- Составить список сотрудников, проектов, заданий, в выполнении которых они участвуют и дат предполагаемого выполнения ими заданий. Учесть сотрудников, не участвующих в проектах.

```
select employee.id, employee.name, p.id as project_id, t.id as task_id, t.due_date_from
from employee
         join staff_member sm on employee.id = sm.id_employee
         join implementation i on sm.id = i.id_employee
         join task t on i.id_task = t.id
         join project p on p.id = t.id_project
order by project_id
```

	id	name	project_id	task_id	due_date_from
1	798	Kyle Harris	2	301	2008-06-29
2	151	Irene Whitenack	2	401	2008-02-12
3	530	Deanna Cooper	2	401	2008-02-12
4	360	Kyle Flores	5	803	2010-04-10
5	162	Edward Landaker	6	629	2013-04-10
6	487	Casey Dejong	6	629	2013-04-10
7	310	Paul Bryant	7	844	2013-07-31

- Составить список руководителей, не сдавших проект в срок с указанием проектов , которые они должны были выполнить и количества дней просрочки сдачи проектов.

```
select employee.id,
       employee.name,
       p.due_date_to,
       p.actual_completion_date,
      EXTRACT(DAY FROM (DATE_TRUNC('day', p.actual_completion_date) - DATE_TRUNC('day', p.due_date_to))) AS days_dif
from employee
         join staff_member sm on employee.id = sm.id_employee
         join project p on sm.id = p.id_supervisor
where p.due_date_to <= p.actual_completion_date
```

	id	name	due_date_to	actual_completion_date	days_diff
1	810	Jennifer Cuzco	2012-05-30	2013-03-03	277

- Составить список проектов, в выполнении которого участвует более трех человек.

```
SELECT p.id as project_id, p.title, COUNT(DISTINCT s.id) as employee_number
FROM project p
         JOIN task t ON t.id_project = p.id
         JOIN implementation i ON i.id_task = t.id
         JOIN staff_member s ON s.id = i.id_employee
GROUP BY p.id
HAVING COUNT(DISTINCT s.id) > 3;
```

	project_id	title	employee_number
1	38	Dolorem voluptatem aliquam sed.	4
2	49	Sit sit quaerat dolor.	6
3	54	Adipisci amet dolor porro quisquam ipsum	4
4	62	Sed consectetur adipisci sit etincidunt	4
5	72	Tempora quiquia adipisci eius quisquam.	4
6	142	Velit amet quaerat etincidunt quisquam q	6

Создание представлений

- для руководителей проектов, содержащее сведения об исполнителях, отделах, сроках выполнения заданий, включенных в проект.

```
CREATE OR REPLACE VIEW supervisor_info AS
select employee.id as supervisor_id,
       employee.name,
       p.id        as project_id,
       t.id        as task_id,
       t.due_date_to,
       t.due_date_from,
       i.id_employee,
       d.department_title
from employee
         join staff_member sm on employee.id = sm.id_employee
         join project p on sm.id = p.id_supervisor
         join task t on p.id = t.id_project
         join implementation i on t.id = i.id_task
         join staff_member sm2 on i.id_employee = sm2.id_employee
         join department d on sm2.id_department = d.id
order by  employee.id
```

- список проектов, срок выполнения которых истекает сегодня и которые включают больше трех невыполненных заданий.

```
CREATE VIEW overdue_projects AS
SELECT p.id, p.title, p.due_date_to, COUNT(t.id) as task_count
FROM project p
         JOIN task t ON t.id_project = p.id
WHERE t.execution_status IN ('Not started', 'In progress')
  AND p.due_date_to <= CURRENT_DATE
GROUP BY p.id, p.title, p.due_date_to
HAVING COUNT(t.id) > 3;
```

Вставка значений:

Создать проект цена которого равна продолжительности последнего проекта умноженного на 1000

```
INSERT INTO project (id_customer, id_manager, id_supervisor, title, due_date_from, due_date_to, execution_status,
                    payment_state, actual_completion_date, price)
VALUES (3, 45, 22, 'New Project', '2022-01-01', '2023-01-01', 'In progress', 'pending payment', NULL,
        (SELECT (due_date_to - due_date_from) * 1000 FROM project WHERE id = (SELECT max(id) from project)));
```

До вставки:

	id	due_date_to	due_date_from	price
1	1003	2012-05-30	2006-02-23	20000.22
2	1000	2005-12-08	2002-03-04	26670
3	999	2017-06-26	2008-01-13	46506

После:

	id	due_date_to	due_date_from	price
1	1004	2023-01-01	2022-01-01	2288000
2	1003	2012-05-30	2006-02-23	20000.22

Изменение значений:

Увеличить зарплату на 30% пятерым сотрудникам которые управляют наибольшим количеством проектов

```
SELECT
    e.name,
    sm.id_employee,
    COUNT(p.id) AS project_count,
    ep.salary::numeric
FROM
    staff_member sm
    INNER JOIN employee e ON sm.id_employee = e.id
    INNER JOIN employee_position ep ON sm.id_job_title = ep.id
    INNER JOIN project p ON sm.id = p.id_supervisor
GROUP BY
    e.name,
    sm.id_employee,
    ep.salary
ORDER BY
    project_count DESC
LIMIT 5;
```

	name	id_employee	project_count	salary
1	Christopher Yi	184	5	1124.4
2	Justin Parker	454	5	2917.99
3	Mike Rader	804	5	3009.84
4	Eddie Paradis	89	5	3291.41
5	Kari Avent	519	5	4848.23

```
UPDATE employee_position
SET salary = salary * 1.3
WHERE id IN (SELECT ep.id
             FROM staff_member sm
                      INNER JOIN employee e ON sm.id_employee = e.id
                      INNER JOIN employee_position ep ON sm.id_job_title = ep.id
                      INNER JOIN project p ON sm.id = p.id_supervisor
             GROUP BY e.name,
                      sm.id_employee,
                      ep.salary,
                      ep.id
             ORDER BY COUNT(p.id) DESC
             LIMIT 5);
```

	name	id_employee	project_count	salary
1	Christopher Yi	184	5	1461.72
2	Justin Parker	454	5	3793.39
3	Mike Rader	804	5	3912.79
4	Eddie Paradis	89	5	4278.83
5	Kari Avent	519	5	6302.7

Удаление элементов

Удалить execution_control за 2003 год

```
SELECT count(execution_control.id)
FROM execution_control
WHERE date_part('year', check_date) = 2003;
```

	count
1	39

```
DELETE
FROM execution_control
WHERE date_part('year', check_date) = 2003;
```

	count
1	0

Индексы

Запрос

```
SELECT employee.name, ep.salary::numeric
FROM employee
         join staff_member sm on employee.id = sm.id_employee
```

```
         join employee_position ep on ep.id = sm.id_job_title
where ep.salary::numeric > ANY (select salary::numeric from employee_position);
```

- 500 rows retrieved starting from 1 in 30 ms (execution: 12 ms, fetching: 18 ms)13 rows retrieved starting from 1 in 55 ms (execution: 10 ms, fetching: 45 ms)

```
EXPLAIN
SELECT *
FROM project
WHERE date_part('year', due_date_to) = 2003;
```

	QUERY PLAN
1	Nested Loop (cost=29.77..13543.49 rows=333 width=46)
2	-> Nested Loop Semi Join (cost=29.50..13427.94 rows=333 width=12)
3	Join Filter: ((ep.salary)::numeric > (employee_position.salary)::numeric)
4	-> Hash Join (cost=29.50..55.14 rows=1000 width=12)
5	Hash Cond: (sm.id_job_title = ep.id)
6	-> Seq Scan on staff_member sm (cost=0.00..23.00 rows=1000 width=8)
7	-> Hash (cost=17.00..17.00 rows=1000 width=12)
8	-> Seq Scan on employee_position ep (cost=0.00..17.00 rows=1000 width=12)
9	-> Materialize (cost=0.00..22.00 rows=1000 width=8)
10	-> Seq Scan on employee_position (cost=0.00..17.00 rows=1000 width=8)
11	-> Index Scan using employee_pkey on employee (cost=0.28..0.34 rows=1 width=18)
12	Index Cond: (id = sm.id_employee)

Создание индекса

```
CREATE INDEX salary_idx ON employee_position (salary);
```

После создание индекса

- 500 rows retrieved starting from 1 in 21 ms (execution: 10 ms, fetching: 11 ms)

Выводы

В процессе выполнения лабораторной работы было освоено составление запросов INSERT, UPDATE и DELETE, а также изучено графическое представление запросов. Кроме того, были созданы индексы, что привело к уменьшению количества этапов при выполнении запросов и ускорению их выполнения.