**Министерство науки и высшего образования Российской Федерации**

федеральное государственное автономное образовательное учреждение высшего образования

**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»**

**Отчет**

ПО ПРАКТИЧЕСКОМУ ЗАДАНИЮ  
 «Лабораторная работа №2»

Автор: Митурский Богдан Антонович

Факультет: ИКТ

Группа: K32392

Преподаватель: Говорова М. М.



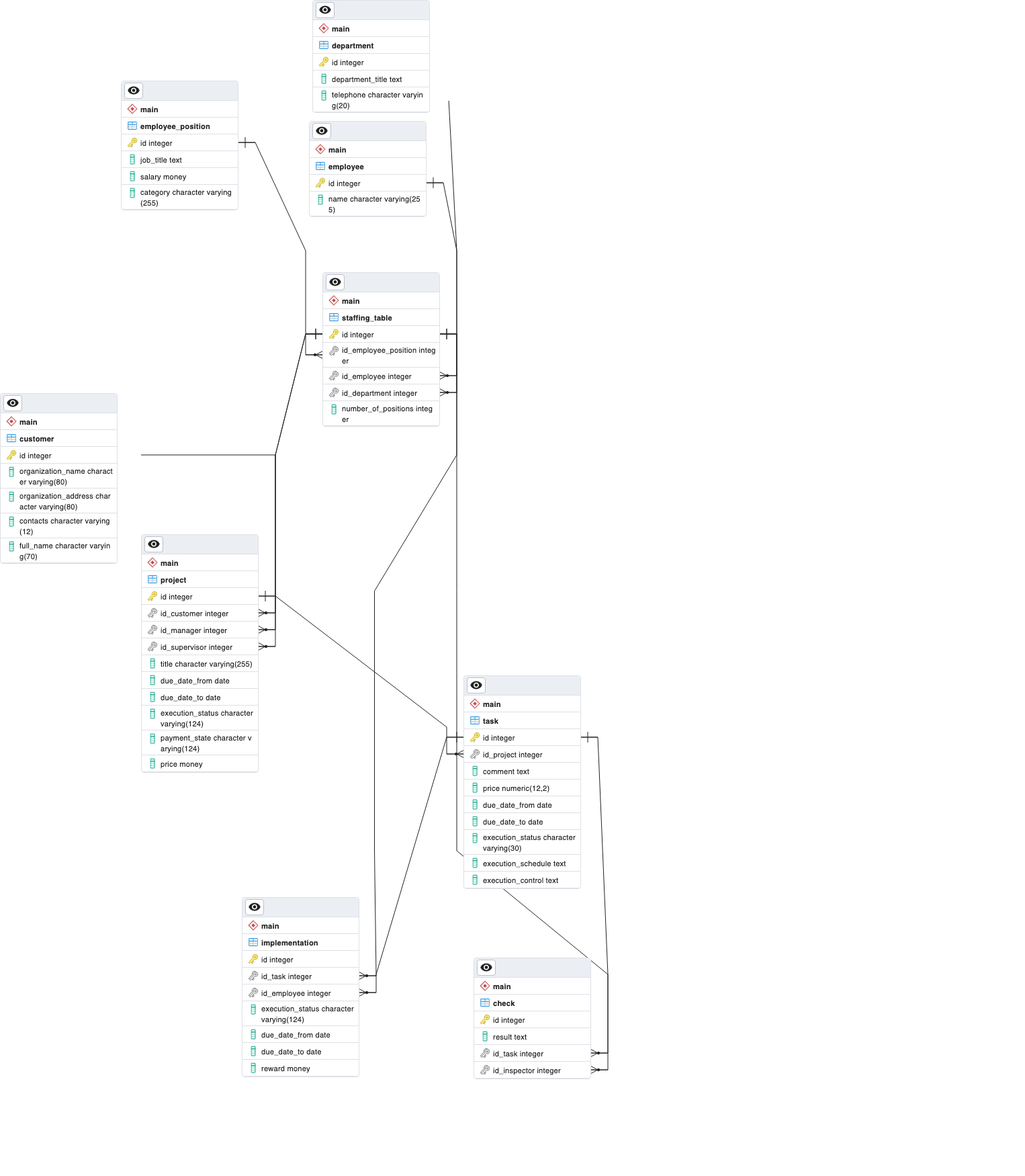
Санкт-Петербург 2023

**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 просмотреть 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.

*Схема логической модели базы данных, сгенерированная в Generate ERD:*

**

**Выполнение (запросы):**

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

SELECT p.id as project\_id, p.title as project\_title, t.id as task\_id, c.organization\_name, ep.job\_title, e.name as employee\_name, d.department\_title

FROM main.project p

JOIN main.task t ON p.id = t.id\_project

JOIN main.customer c ON p.id\_customer = c.id

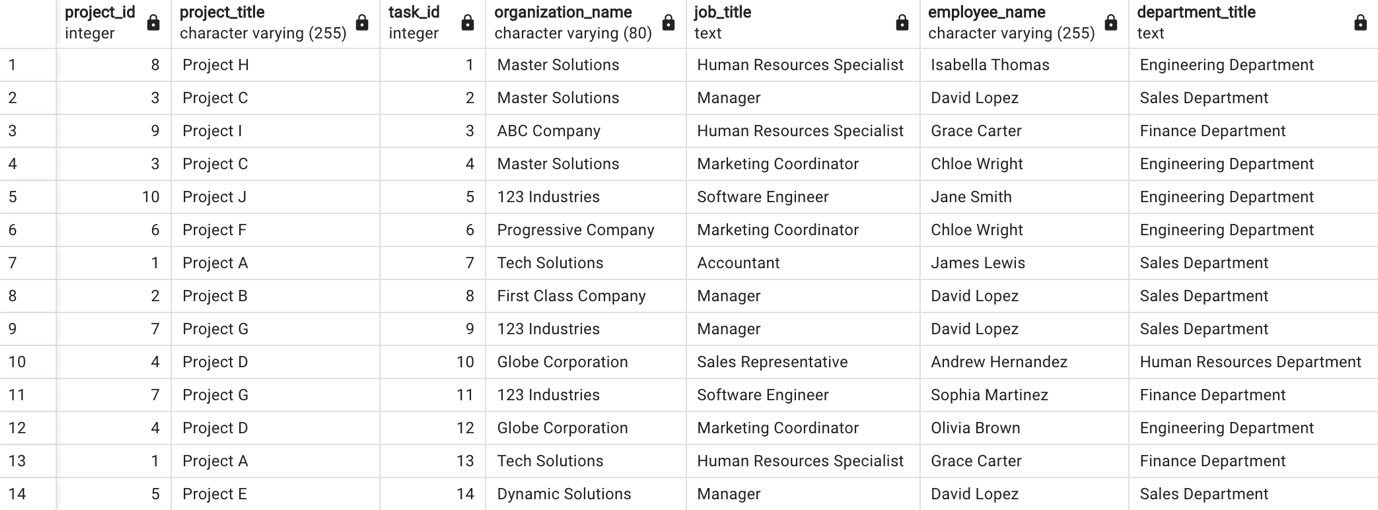
JOIN main.implementation i ON t.id = i.id\_task

JOIN main.employee e ON i.id\_employee = e.id

JOIN main.staffing\_table st ON e.id = st.id\_employee

JOIN main.employee\_position ep ON st.id\_employee\_position = ep.id

JOIN main.department d ON st.id\_department = d.id;

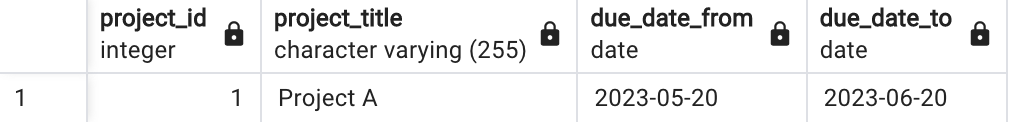


1. Составить список проектов, работа над которыми была начата больше месяца назад.

SELECT p.id as project\_id, p.title as project\_title, p.due\_date\_from, p.due\_date\_to

FROM main.project p

WHERE due\_date\_from < NOW() - INTERVAL '1 month';



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

SELECT e.id, e.name

FROM main.employee e

JOIN main.staffing\_table st ON e.id = st.id\_employee

JOIN main.employee\_position ep ON st.id\_employee\_position = ep.id

WHERE CAST(ep.salary AS NUMERIC) > (

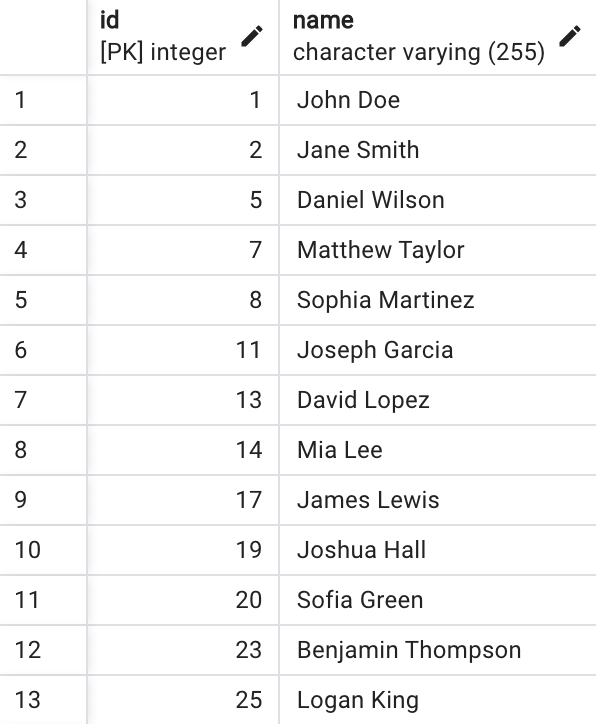
SELECT AVG(CAST(ep2.salary AS NUMERIC))

FROM main.employee\_position ep2

JOIN main.staffing\_table st2 ON ep2.id = st2.id\_employee\_position

WHERE st2.id\_department = st.id\_department

);



1. Найти отдел, работающий над максимальным количеством проектов.

SELECT d.id, d.department\_title, COUNT(DISTINCT p.id) as project\_count

FROM main.department d

JOIN main.staffing\_table st ON d.id = st.id\_department

JOIN main.employee e ON st.id\_employee = e.id

JOIN main.implementation i ON e.id = i.id\_employee

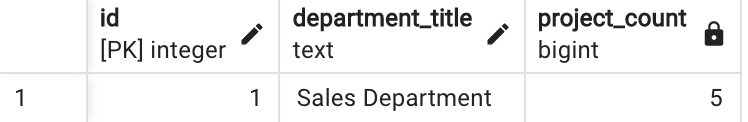
JOIN main.task t ON i.id\_task = t.id

JOIN main.project p ON t.id\_project = p.id

GROUP BY d.id, d.department\_title

ORDER BY project\_count DESC

LIMIT 1;



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

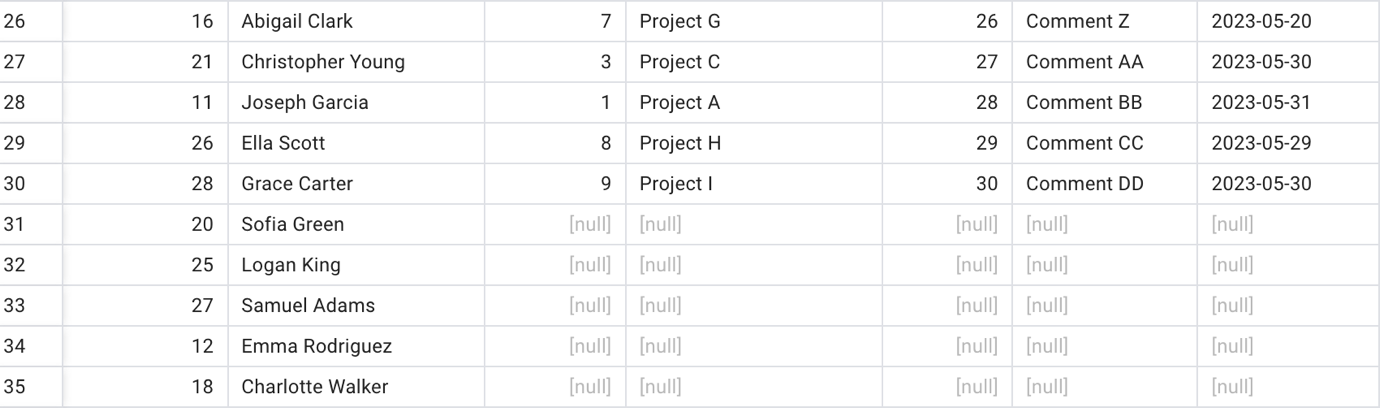
SELECT e.id as employee\_id, e.name as employee\_name, p.id as project\_id, p.title as project\_title, t.id as task\_id, t.comment as task\_comment, i.due\_date\_to as task\_due\_date

FROM main.employee e

LEFT JOIN main.implementation i ON e.id = i.id\_employee

LEFT JOIN main.task t ON i.id\_task = t.id

LEFT JOIN main.project p ON t.id\_project = p.id;



1. Составить список сотрудников, не выполнивших задания в срок с указанием проектов и заданий, которые они должны были выполнить и количества дней просрочки выполнения заданий.

SELECT e.id as employee\_id, e.name as employee\_name, p.id as project\_id, p.title as project\_title, t.id as task\_id, t.comment as task\_comment, t.due\_date\_to as task\_due\_date,

CURRENT\_DATE - t.due\_date\_to as overdue\_days

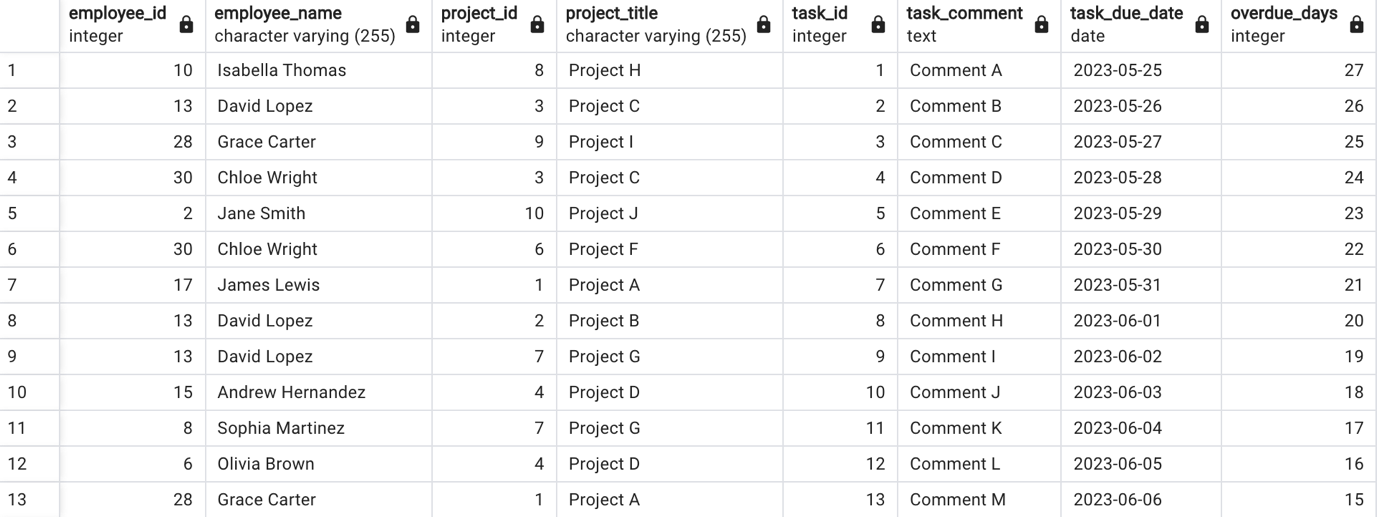
FROM main.employee e

JOIN main.implementation i ON e.id = i.id\_employee

JOIN main.task t ON i.id\_task = t.id

JOIN main.project p ON t.id\_project = p.id

WHERE t.due\_date\_to < CURRENT\_DATE AND i.execution\_status != 'completed';



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

SELECT p.id as project\_id, p.title as project\_title, COUNT(DISTINCT e.id) as employee\_count

FROM main.project p

JOIN main.task t ON p.id = t.id\_project

JOIN main.implementation i ON t.id = i.id\_task

JOIN main.employee e ON i.id\_employee = e.id

GROUP BY p.id, p.title

HAVING COUNT(DISTINCT e.id) > 3;



**Выполнение (представления):**

1. Для руководителей проектов, содержащее сведения об исполнителях, отделах, сроках выполнения заданий, включенных в проект.

CREATE VIEW main.project\_manager\_view AS

SELECT p.id as project\_id, p.title as project\_title, e.id as employee\_id, e.name as employee\_name, d.id as department\_id, d.department\_title, t.due\_date\_from as task\_start\_date, t.due\_date\_to as task\_end\_date

FROM main.project p

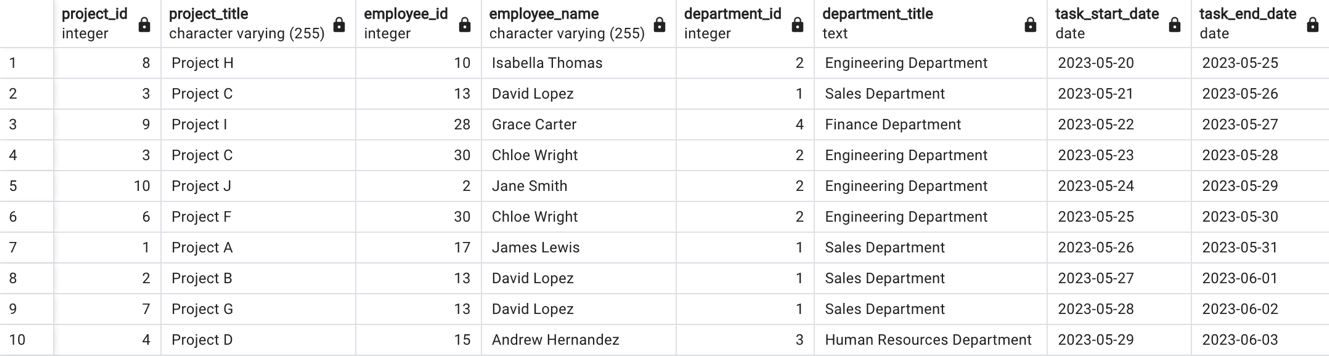
JOIN main.task t ON p.id = t.id\_project

JOIN main.implementation i ON t.id = i.id\_task

JOIN main.employee e ON i.id\_employee = e.id

JOIN main.staffing\_table st ON e.id = st.id\_employee

JOIN main.department d ON st.id\_department = d.id;



1. Cписок проектов, срок выполнения которых истекает сегодня и которые включают больше трех невыполненных заданий.

CREATE VIEW main.projects\_due\_today\_with\_uncompleted\_tasks AS

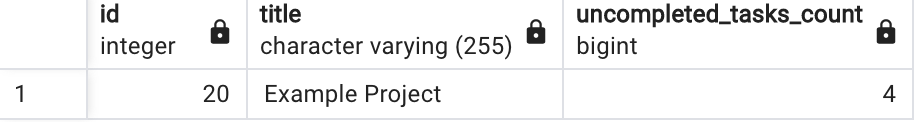
SELECT p.id, p.title, COUNT(t.id) AS uncompleted\_tasks\_count

FROM main.project p

JOIN main.task t ON t.id\_project = p.id AND t.execution\_status != 'Completed'

WHERE p.due\_date\_to = CURRENT\_DATE

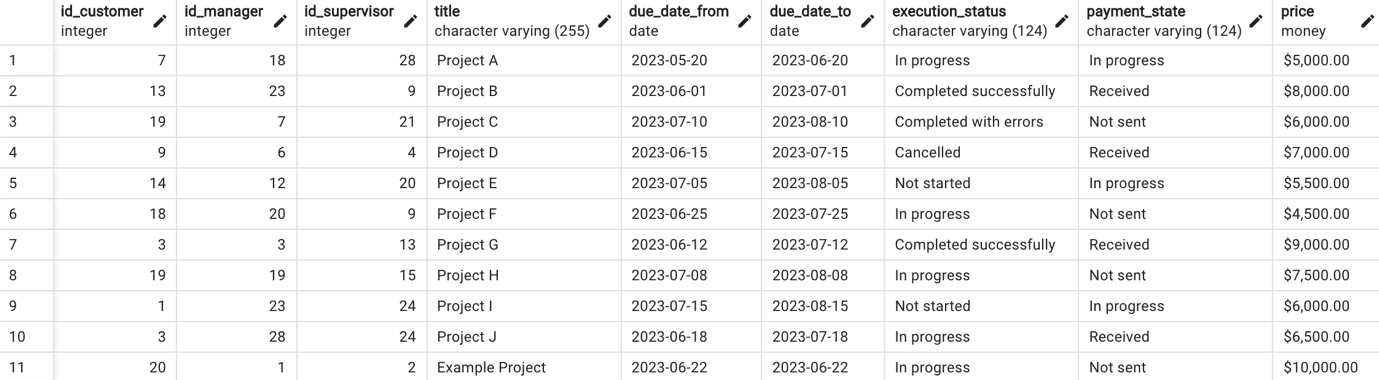
GROUP BY p.id, p.title

HAVING COUNT(t.id) > 3;

**Выполнение (Модификация данных):**

1. INSERT - Создать проект цена которого равна продолжительности самого длинного проекта умноженного на 10000.

*До:*

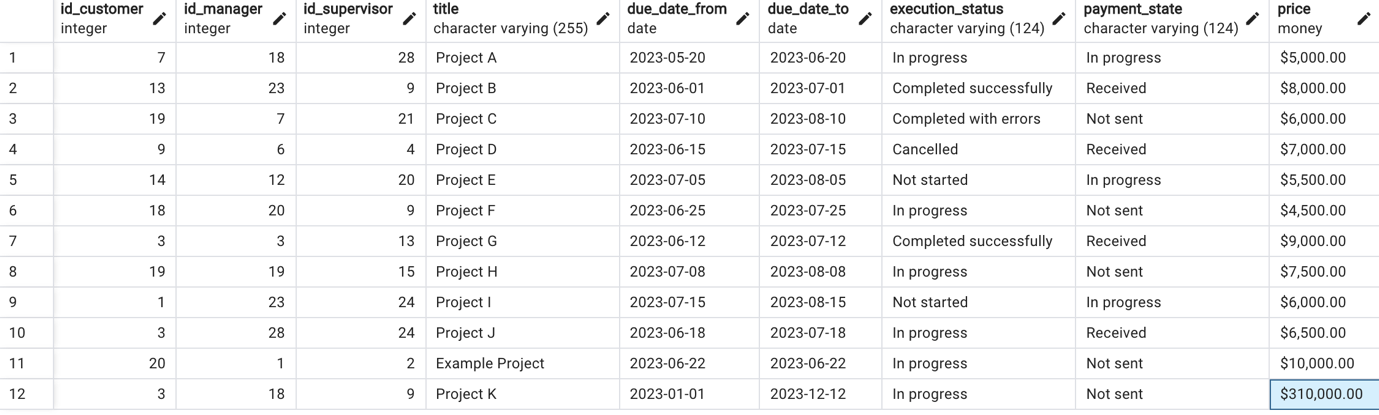


INSERT INTO main.project (id\_customer, id\_manager, id\_supervisor, title, due\_date\_from, due\_date\_to, execution\_status, payment\_state, price)

VALUES (3, 18, 9, 'Project K', '2023-01-01', '2023-12-12', 'In progress', 'Not sent',

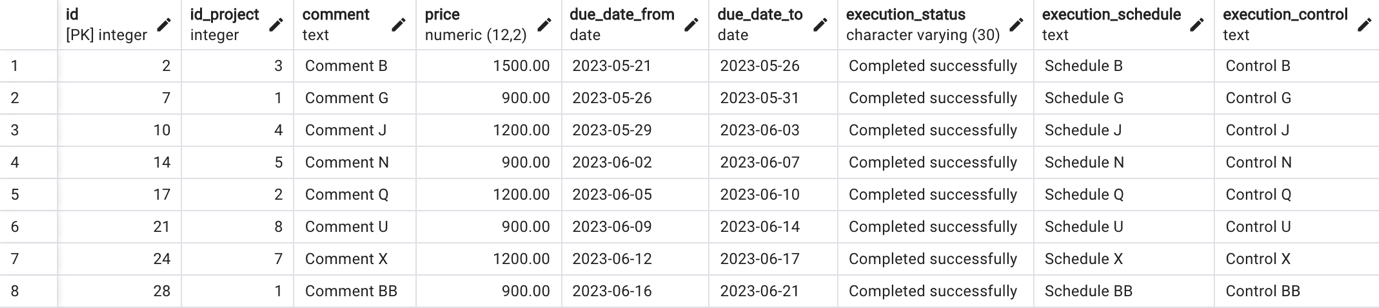
(SELECT (due\_date\_to - due\_date\_from) \* 10000 FROM main.project ORDER BY (due\_date\_to - due\_date\_from) DESC LIMIT 1));

*После:*

**

1. UPDATE - Изменить статус выполнения на 'Completed' для всех задач, которые начались более месяца назад и еще не завершены.

*До:*



UPDATE main.task

SET execution\_status = 'Completed successfully'

WHERE id IN (

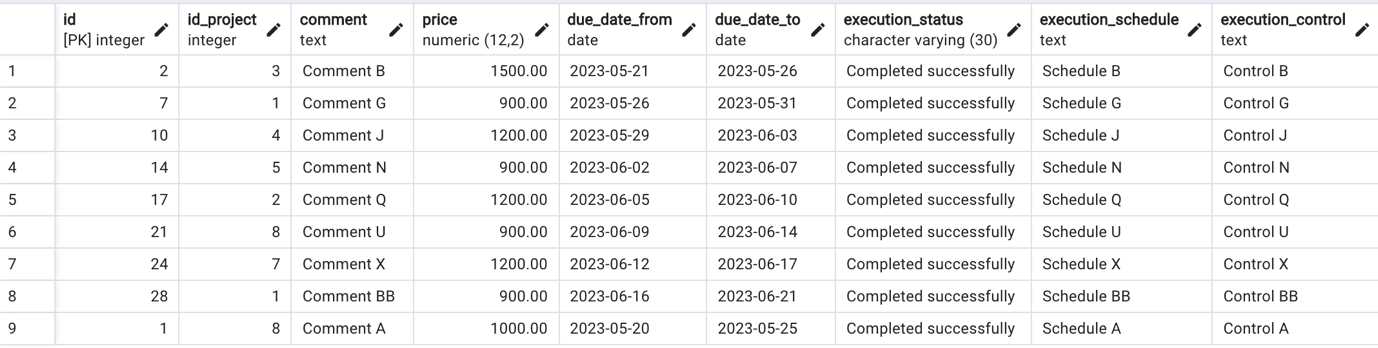
SELECT id

FROM main.task

WHERE execution\_status NOT LIKE '%Completed%' AND due\_date\_from < CURRENT\_DATE - INTERVAL '1 month'

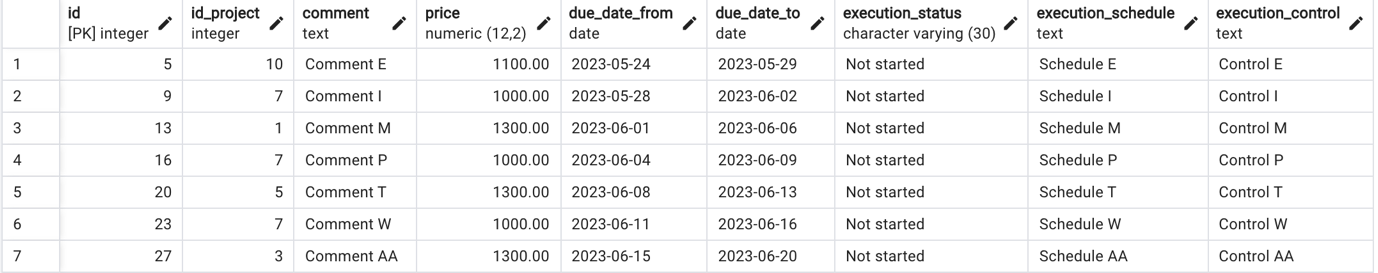
);

*После:*



1. DELETE – Удалить все задачи которые не были начаты и срок их выполнения уже истек.

*До:*



-- Удаляем записи из таблицы "implementation", т.к. они зависят от task и просто удалить task не получится

DELETE FROM main.implementation

WHERE id\_task IN (

SELECT id

FROM main.task

WHERE execution\_status = 'Not started' AND due\_date\_to < CURRENT\_DATE

);

-- Удаляем записи из таблицы "task"

DELETE FROM main.task

WHERE execution\_status = 'Not started' AND due\_date\_to < CURRENT\_DATE;

*После:*



**Выполнение (индексы):**

*До индексов:*

EXPLAIN ANALYZE

SELECT st.id\_employee, pr.title, t.id

FROM main.project AS pr

JOIN main.task AS t ON pr.id = t.id\_project

JOIN main.implementation AS i ON t.id = i.id\_task

JOIN main.staffing\_table AS st ON st.id\_employee = i.id\_employee

JOIN main.department AS d ON d.id = st.id\_department

WHERE d.id > 1;

*Planing Time: 17ms*

*Excution Time 0.380ms*

CREATE INDEX idx\_task\_project\_id ON main.task(id\_project);

CREATE INDEX idx\_implementation\_task\_id ON main.implementation(id\_task);

CREATE INDEX idx\_implementation\_employee\_id ON main.implementation(id\_employee);

*После индексов:*

*Planing Time: 1ms*

*Excution Time 2.060ms*

**Выводы:**

В процессе выполнения лабораторной работы было освоено составление запросов INSERT, UPDATE и DELETE. Также, были созданы индексы, что привело к ускорению их выполнения.