**Министерство науки и высшего образования Российской Федерации**

федеральное государственное автономное образовательное учреждение высшего образования

**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»**

**Отчет**

по лабораторной работе №4 «Запросы на выборку и модификацию данных. Представления. Работа с индексами**»**

по дисциплине **«Проектирование и реализация баз данных»**

Автор: Таипов Т. А.

Факультет: ИКТ

Группа: К3241

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.

**Оборудование:** компьютерный класс.

**Программное обеспечение:** СУБД PostgreSQL, pgadmin 4.

**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.

# Запросы к базе данных

* Вывести все номера групп и программы, где количество слушателей меньше 10.

SELECT "group".id, current\_members, curriculum.id AS curriculum\_id, program.id AS program\_id

FROM "group"

JOIN curriculum ON curriculum.id = "group".curriculum\_id

JOIN program ON program.id = curriculum.program\_id

Where current\_members < 10;

Изображение выглядит как текст, снимок экрана, Мультимедийное программное обеспечение, программное обеспечение

Автоматически созданное описание

* Вывести список преподавателей с указанием количества программ, где они преподавали за истекший год.

SELECT

l.id AS lecturer\_id,

l.full\_name AS lecturer\_name,

COUNT(DISTINCT p.id) AS number\_of\_programs\_taught

FROM

public.lecturer l

JOIN

public.schedule s ON l.id = s.lecturer\_id

JOIN

public.discipline\_to\_curriculum dc ON s.discipline\_in\_curriculum\_id = dc.id

JOIN

public.curriculum c ON dc.curriculum\_id = c.id

JOIN

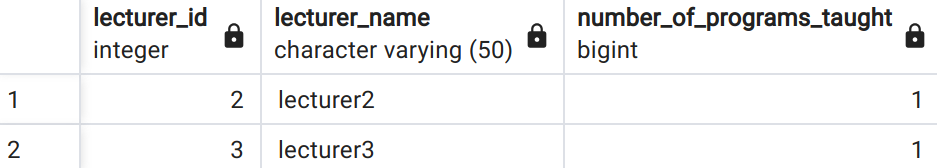
public.program p ON c.program\_id = p.id

WHERE

EXTRACT(YEAR FROM s.date) = EXTRACT(YEAR FROM CURRENT\_DATE) - 1

GROUP BY

l.id, l.full\_name;



* Вычислить среднюю загруженность компьютерных классов в неделю за последний месяц (в часах).

SELECT

AVG(weekly\_load) / 4.41 AS average\_weekly\_load

FROM (

SELECT

SUM(90) AS weekly\_load

FROM

public.schedule s

WHERE

s.date >= CURRENT\_DATE - INTERVAL '1 month'

AND s.audience\_id IN (

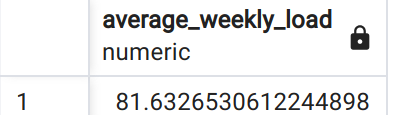
SELECT id

FROM public.audience

WHERE type = 'computer\_lab'

)

) AS weekly\_subquery;



* Вывести список преподавателей, которые не проводят занятия на третьей паре ни в один из дней недели.

SELECT s.id, time, lecturer.id AS lecturer\_id, lecturer.full\_name

FROM schedule s

JOIN lecturer ON lecturer.id = s.lecturer\_id

WHERE time != '12:00:00'

Изображение выглядит как текст, снимок экрана, Шрифт, Мультимедийное программное обеспечение

Автоматически созданное описание

* Вывести список свободных лекционных аудиторий на ближайший понедельник.

SELECT

a.id AS audience\_id,

a."number" AS audience\_number

FROM

public.audience a

WHERE

a.type = 'lecture'

AND a.id NOT IN (

SELECT DISTINCT

s.audience\_id

FROM

public.schedule s

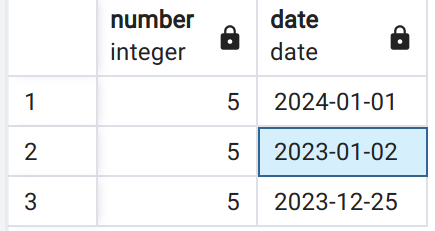
WHERE

EXTRACT(DOW FROM s.date) = 1

AND s.date >= CURRENT\_DATE

AND s.date < CURRENT\_DATE + INTERVAL '7 days'

);



* Вычислить общее количество обучающихся по каждой программе за последний год.

SELECT p.name AS program\_name, COUNT(stg.id) AS total\_students

FROM public.program p

JOIN public.curriculum c ON p.id = c.program\_id

JOIN public."group" g ON c.id = g.curriculum\_id

JOIN public.student\_to\_group stg ON g.id = stg.group\_id

WHERE g.date\_of\_creation BETWEEN CURRENT\_DATE - INTERVAL '1' YEAR AND CURRENT\_DATE

GROUP BY p.name

ORDER BY total\_students DESC;

Изображение выглядит как текст, снимок экрана, Мультимедийное программное обеспечение, программное обеспечение

Автоматически созданное описание

* Найти самые популярные программы за последние 3 года.

WITH ProgramsWithRanks AS (

SELECT

p.id AS program\_id,

p.name AS program\_name,

COUNT(sg.student\_id) AS total\_students,

RANK() OVER (ORDER BY COUNT(sg.student\_id) DESC) AS rnk

FROM

public.program p

JOIN

public.curriculum c ON p.id = c.program\_id

JOIN

public."group" g ON c.id = g.curriculum\_id

JOIN

public.student\_to\_group sg ON g.id = sg.group\_id

WHERE

g.date\_of\_creation >= (CURRENT\_DATE - INTERVAL '3 years')

GROUP BY

p.id, p.name

)

SELECT

program\_id,

program\_name,

total\_students

FROM

ProgramsWithRanks

WHERE

rnk = 1;

Изображение выглядит как текст, снимок экрана, Мультимедийное программное обеспечение, программное обеспечение

Автоматически созданное описание

# Представления

* Для потенциальных слушателей, содержащее перечень специальностей, изучаемых на них дисциплин и количество часов;

CREATE OR REPLACE VIEW public.potential\_students\_view AS

SELECT

p.id AS program\_id,

p.name AS program\_name,

d.id AS discipline\_id,

d.name AS discipline\_name,

d.duration AS hours\_per\_week

FROM

public.program p

JOIN

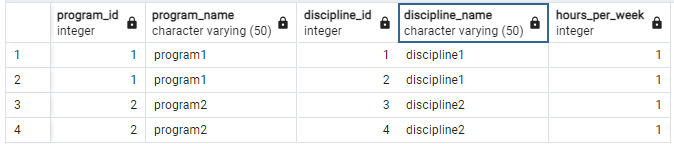
public.curriculum c ON p.id = c.program\_id

JOIN

public.discipline\_to\_curriculum dc ON c.id = dc.curriculum\_id

JOIN

public.discipline d ON dc.discipline\_id = d.id;



* общих доход по каждой программе за последний год.

CREATE OR REPLACE VIEW public.program\_revenue

AS

SELECT p.id AS program\_id,

p.name AS program\_name,

sum(c.contract\_price) AS total\_revenue

FROM program p

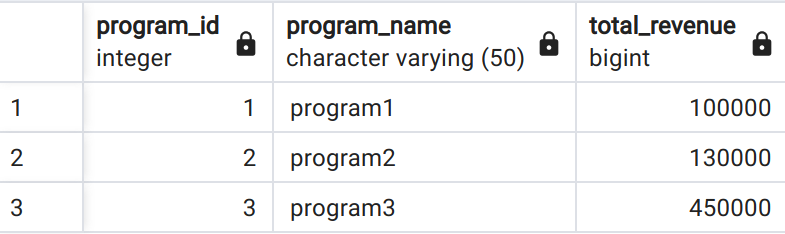
JOIN curriculum c ON p.id = c.program\_id

JOIN "group" g ON g.curriculum\_id = c.id

JOIN student\_to\_group sg ON g.id = sg.group\_id

WHERE g.date\_of\_creation >= (CURRENT\_DATE - '1 year'::interval year) AND sg.type = 'contract'::studenttogrouptype

GROUP BY p.id, p.name;



# Запросы на модификацию (с подзапросами)

* Добавить нового студента в группу

INSERT INTO public.student\_to\_group (student\_id, group\_id, start\_date, end\_date, status)

VALUES (

(SELECT id FROM public.student WHERE full\_name = 'Pupin Vasilis Vasiliev'),

(SELECT id FROM public."group" WHERE name = 'group4'),

'2023-01-01',

NULL,

'studying');

Изображение выглядит как снимок экрана, текст, Мультимедийное программное обеспечение, программное обеспечение

Автоматически созданное описание

* Изменить дату окончания у студента в группе

UPDATE public.student\_to\_group

SET end\_date = '2023-12-31', status = 'graduated'

WHERE

student\_id = (SELECT id FROM public.student WHERE full\_name = 'Pupin Vasilis Vasiliev')

AND group\_id = (SELECT id FROM public."group" WHERE name = 'group4');

Изображение выглядит как снимок экрана, текст, Мультимедийное программное обеспечение, Графическое программное обеспечение

Автоматически созданное описание

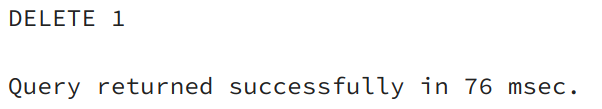
* Удалить все записи о студенте из группы:

DELETE FROM public.certification

WHERE student\_id = (SELECT id FROM public.student WHERE full\_name = 'Petrov Petr Petrovich');

DELETE FROM public.student\_to\_group

WHERE student\_id = (SELECT id FROM public.student WHERE full\_name = 'Petrov Petr Petrovich');



# Индексы

Тестовый запрос:

EXPLAIN ANALYZE

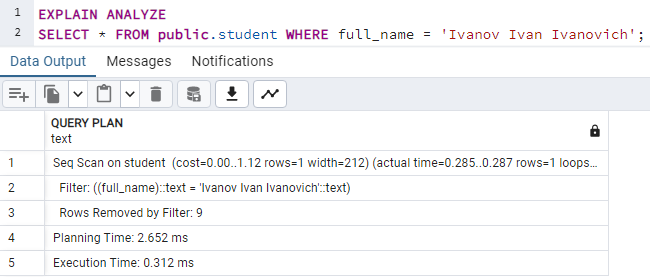
SELECT \*

FROM public.student WHERE full\_name = 'Ivanov Ivan Ivanovich';

EXPLAIN ANALYZE

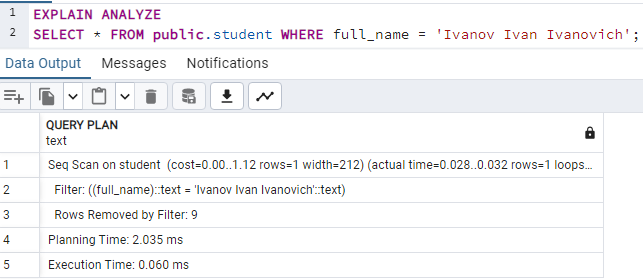
SELECT \* FROM public.student WHERE full\_name = 'Ivanov Ivan Ivanovich' AND phone\_number='+77529123456';

Без индекса:



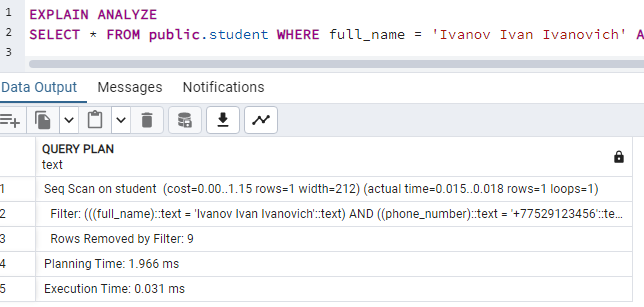
С простым индексом

CREATE INDEX idx\_student\_name ON public.student (full\_name);:



С составным индексом

CREATE INDEX idx\_student\_name\_phone ON public.student (full\_name, phone\_number);:



# Вывод

В ходе лабораторной работы я освоил практические навыки по выполнению select, insert, delete и update запросов. Также, научился делать представления и индексы. В ходе анализа в первом запросе индексы позволили сократить время выполнения запроса.