

Санкт-Петербургский национальный исследовательский университет ИТМО

Факультет Инфокоммуникационных технологий

**Лабораторная работа №2 по теме**

**«Запросы на выборку и модификацию данных, представления и индексы в PostgreSQL»**

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

Выполнил:

студент 2 курса К32421 группы

Козлов Всеволод Денисович

Преподаватель:

Говорова Марина Михайловна

Санкт-Петербург

**Цель работы:** овладеть практическими навыками создания представлений и запросов на выборку данных к базе данных PostgreSQL, использования подзапросов при модификации данных и индексов.

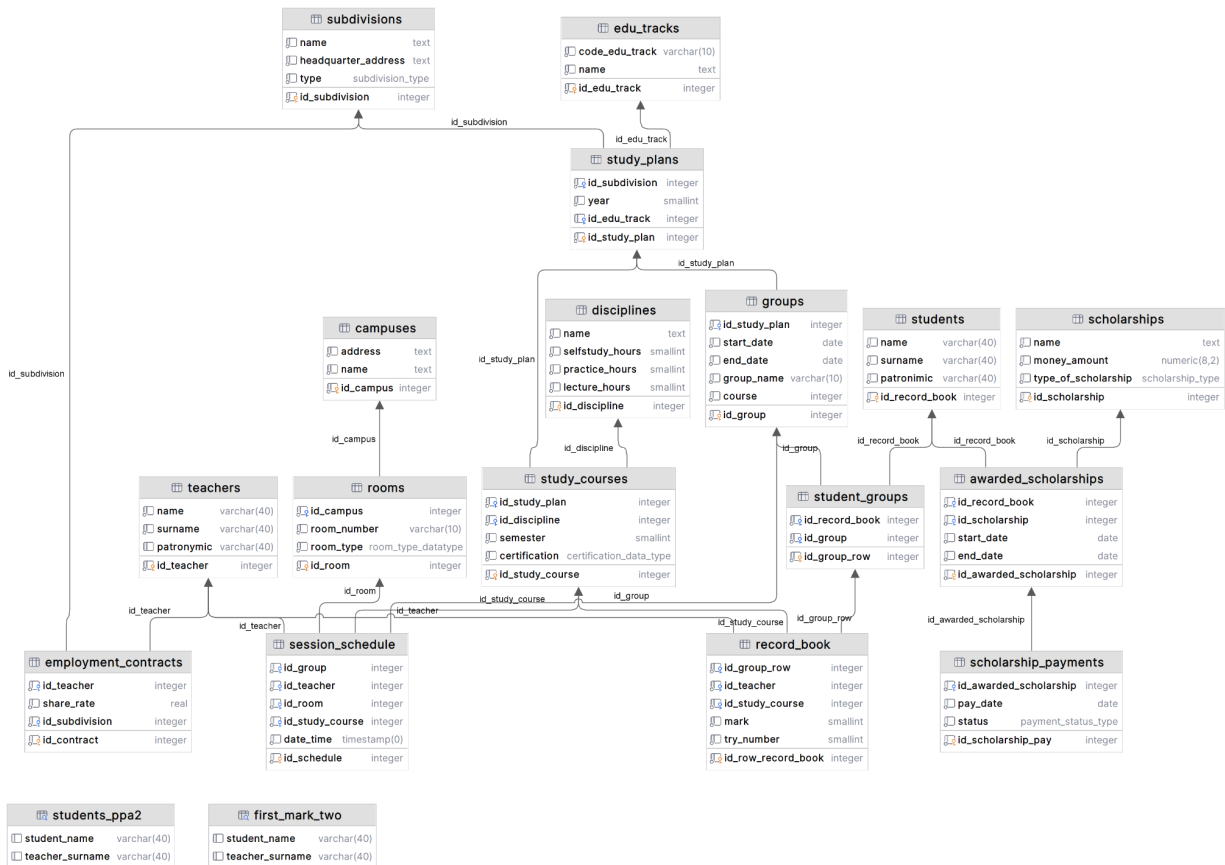
**Оборудование:** компьютерный класс.

**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.

## Схема БД



## Запросы согласно индивидуальному заданию

### Запрос №1

Формулировка запроса:

список дисциплин, которые должны быть сданы заданной группой с указанием дат сдачи и фамилий преподавателей.

Команда:

```

select date_time, surname
from session_schedule join teachers
    on session_schedule.id_teacher = teachers.id_teacher
where id_group = (select id_group from groups where
    group_name='K10');
  
```

Результат:

	📅 date_time	👤 surname
1	2012-01-11 10:00:00	teacher_surname0
2	2012-01-11 10:00:00	teacher_surname1
3	2012-01-04 10:00:00	teacher_surname2
4	2012-01-28 10:00:00	teacher_surname3
5	2012-01-23 10:00:00	teacher_surname4
6	2012-09-25 10:00:00	teacher_surname5
7	2012-09-15 10:00:00	teacher_surname6
8	2012-09-13 10:00:00	teacher_surname7
9	2012-09-15 10:00:00	teacher_surname8
10	2012-09-01 10:00:00	teacher_surname9

## Запрос №2

Формулировка запроса:

Список студентов, получивших двойки на первой попытке с указанием фамилии преподавателя, которым они должны пересдать экзамен.

Команда:

```
select s.surname as student_name, t.surname as teacher_surname
from record_book
join student_groups sg
  on sg.id_group_row = record_book.id_group_row
join students s
  on sg.id_record_book = s.id_record_book
join teachers t
  on record_book.id_teacher = t.id_teacher
where try_number=1 and mark=2;
```

Результат:

	student_name	teacher_surname
1	student_surname0	teacher_surname0
2	student_surname0	teacher_surname2
3	student_surname0	teacher_surname1
4	student_surname0	teacher_surname7
5	student_surname0	teacher_surname6
6	student_surname0	teacher_surname4
7	student_surname0	teacher_surname18
8	student_surname0	teacher_surname16
9	student_surname0	teacher_surname19
10	student_surname0	teacher_surname15
11	student_surname0	teacher_surname14
12	student_surname0	teacher_surname13
13	student_surname0	teacher_surname12

### Запрос №3

Формулировка запроса:

Фамилии студентов, получивших оценки по дисциплине, которые выше среднего балла по этой дисциплине.

Команда:

```
from record_book
join student_groups sg
    on sg.id_group_row = record_book.id_group_row
join students s
    on sg.id_record_book = s.id_record_book
join (select id_study_course, avg(mark) as avg_mark from
record_book where mark<>2 group by id_study_course) avg_mark
    on record_book.id_study_course = avg_mark.id_study_course
where record_book.mark > avg_mark.avg_mark;
```

Результат:

	stud_surname	id_study_course
1	student_surname0	7
2	student_surname0	0
3	student_surname0	5
4	student_surname0	3
5	student_surname0	8
6	student_surname0	6
7	student_surname0	15
8	student_surname0	19
9	student_surname0	11
10	student_surname1	2
11	student_surname1	5
12	student_surname1	6
13	student_surname1	10
14	student_surname1	12
15	student_surname1	15
16	student_surname1	16

#### Запрос №4

Формулировка запроса:

рейтинговый список групп по заданному направлению по результатам сдачи сессии, упорядочить его по убыванию.

Команда:

```
select sg.group_name, avg(mark) from record_book
join
  (select id_group_row, group_name from student_groups
  join groups g
    on g.id_group = student_groups.id_group
  where g.id_study_plan = 0) sg
on sg.id_group_row = record_book.id_group_row
where mark <> 2
group by sg.group_name
order by avg(mark) desc;
```

Результат:

	group_name	avg
1	K10	3.9504950495049505
2	K20	3.95

### Запрос №5

Формулировка запроса:

Списки студентов, упорядоченные по группам и фамилиям студентов, назначении на стипендии. Студент получает стипендию, если он сдал сессию без троек. Если студент не назначен на стипендию, указать 0, если назначен – 1.

Команда:

```
select s.surname, sg.group_name,
       case
         when min(mark) > 3 then 1
         else 0
       end as is_awarder
from record_book
join (select id_record_book, id_group_row, group_name
      from student_groups
      join groups g
      on g.id_group = student_groups.id_group) sg
on sg.id_group_row = record_book.id_group_row
join students s
on sg.id_record_book = s.id_record_book
group by (s.surname, sg.group_name)
order by surname, group_name;
```

Результат:

console 53 ms	ame	group_name	is_awarder
1	student_surname0	K10	0
2	student_surname0	K20	0
3	student_surname1	K10	0
4	student_surname1	K20	0
5	student_surname10	K11	0
6	student_surname10	K21	0
7	student_surname11	K11	0
8	student_surname11	K21	0
9	student_surname12	K11	0
10	student_surname12	K21	0

У меня ни один студент не получил стипендию, потому что оценки с равной вероятностью выбираются из 2, 3, 4, 5. Так как у каждого студента в группе по 10 предметов, то вероятность получить стипендию составляет  $1/(2 \cdot 10)$

#### Запрос №6

Формулировка запроса:

Список студентов, сдавших все положенные экзамены.

Команда:

```
select id_record_book from record_book
join (select id_group_row, id_study_course, max(try_number) as
last_try
      from record_book r2
      group by (id_group_row, id_study_course)) as trys
on record_book.try_number=last_try and
   record_book.id_group_row = trys.id_group_row and
   record_book.id_study_course = trys.id_study_course
join student_groups sg
on record_book.id_group_row = sg.id_group_row
group by sg.id_record_book
having min(mark) > 2;
```

Результат:



	id_record_book
1	0
2	1
3	2
4	3
5	4
6	5
7	6
8	7
9	8
10	9
11	10
12	11
13	12
14	13
15	14
16	15
17	16
18	17
19	18
20	19

У меня все студенты сдали экзамены, потому что если студент получает 2-ку, то для него обязательно генерирует 3/4/5 на одной из следующих попыток.

### Запрос №7

Формулировка запроса:

Список студентов, получивших максимальный средний балл в своей группе.

Команда:



```
select sg.id_record_book, sg.id_group
from
  (select id_group_row, avg(mark) as avg_st
   from record_book
```

```

        where mark<>2
        group by id_group_row) avg_st
join
    student_groups sg
on
    sg.id_group_row = avg_st.id_group_row
join
    (select id_group, max(avg_st) as max_group_avg from
record_book as r1
    join student_groups sg on
        r1.id_group_row = sg.id_group_row
    join (select id_group_row, avg(mark) as avg_st
        from record_book
        where mark<>2
        group by id_group_row) s_av
    on r1.id_group_row = s_av.id_group_row
    group by id_group) as max_group_avg
on
    sg.id_group = max_group_avg.id_group
where avg_st=max_group_avg;

```

Результат:

	 id_record_book	 id_group
1	2	0
2	15	1
3	10	3
4	8	2

## Создание представлений согласно индивидуальному заданию

### Представление №

#### Формулировка:

Список студентов, получивших двойки на первой попытке с указанием фамилии преподавателя, которым они должны пересдать экзамен;

#### Команда:

```

create or replace view first_mark_two as
    select s.surname as student_name, t.surname as
teacher_surname

```

```

from record_book
join student_groups sg
    on sg.id_group_row = record_book.id_group_row
join students s
    on sg.id_record_book = s.id_record_book
join teachers t
    on record_book.id_teacher = t.id_teacher
where try_number=1 and mark=2;

```

### Результат:

Output session.public.first_mark_two		
275 rows		
	student_name	teacher_surname
1	student_surname0	teacher_surname0
2	student_surname0	teacher_surname2
3	student_surname0	teacher_surname1
4	student_surname0	teacher_surname7
5	student_surname0	teacher_surname6
6	student_surname0	teacher_surname4
7	student_surname0	teacher_surname18
8	student_surname0	teacher_surname16
9	student_surname0	teacher_surname19
10	student_surname0	teacher_surname15
11	student_surname0	teacher_surname14

## Представление №

### Формулировка:

Исходная: Данные о студентах при получении ими хотя бы одной оценки 2 (после 3-й попытки).

Проблема: У меня не нет ни одного с студента с 2 на 3-й попытке

Измененная формулировка: Так как у меня никто не отчислился из института, то есть нет ни одного с студента с 2 на 3-й попытке, то создам представление для студентов, побывавших на ппа2

### Команда:

```

create or replace view students_ppa2 as
    select s.surname as student_name, t.surname as
teacher_surname

```

```

from record_book
join student_groups sg
    on sg.id_group_row = record_book.id_group_row
join students s
    on sg.id_record_book = s.id_record_book
join teachers t
    on record_book.id_teacher = t.id_teacher
where try_number=2 and mark=2;

```

**Результат:**

	student_name	teacher_surname
1	student_surname0	teacher_surname7
2	student_surname0	teacher_surname6
3	student_surname0	teacher_surname2
4	student_surname0	teacher_surname12
5	student_surname0	teacher_surname11
6	student_surname0	teacher_surname13
7	student_surname0	teacher_surname16
8	student_surname0	teacher_surname14
9	student_surname0	teacher_surname19
10	student_surname0	teacher_surname17
11	student_surname0	teacher_surname10
12	student_surname1	teacher_surname4
13	student_surname1	teacher_surname3
14	student_surname1	teacher_surname13
15	student_surname1	teacher_surname12
16	student_surname1	teacher_surname10
17	student_surname1	teacher_surname11
18	student_surname1	teacher_surname17
19	student_surname2	teacher_surname8
20	student_surname2	teacher_surname0

Из-за особенности генерации данных у меня все студенты побывали на ППА2

## Запросы модификацию данных (INSERT, UPDATE, DELETE) с использованием подзапросов.

### Запрос №1 на update

Формулировку запроса:

Заменить название предмета на "nice discipline", по которому student\_surname4 получил больше всего 5

Команда:

```
create temporary table cnt_5_marks as
  select id_discipline, count(*) cnt_5 from record_book
  join study_courses sc
    on record_book.id_study_course = sc.id_study_course
  where
    id_group_row in
      (select id_group_row from student_groups where
        id_record_book in
          (select id_record_book from students where
            surname='student_surname4'))
    and
      mark=5
  group by id_discipline;

update disciplines
set name='nice discipline'
where id_discipline=(select id_discipline from cnt_5_marks
where cnt_5=(select max(cnt_5) from cnt_5_marks));
select * from disciplines where id_discipline=8;

drop table cnt_5_marks;
```

Скриншот до:

	id_discipline	name	selfstudy_h...	practice_hours	lecture_hours
1	8	name8	85	85	85

Скриншот после:

	id_discipline	name	selfstudy_h...	practice_hours	lecture_hours
1	8	nice discipline	85	85	85

### Запрос №2 на update

Формулировку запроса:

Увеличить количество selfstudy\_hours на 10 для дисциплин с самой низкой средней успеваемостью

Команда:

```
create temporary table discipline_avg as
select d.id_discipline, avg(mark) avg_mark from record_book
join study_courses sc
  on record_book.id_study_course = sc.id_study_course
join
  disciplines d on sc.id_discipline = d.id_discipline
where mark <> 2
group by d.id_discipline;

update disciplines
  set selfstudy_hours = selfstudy_hours + 10
where id_discipline in
  (select id_discipline from discipline_avg
   where avg_mark >= all(select avg_mark from discipline_avg)
   limit 1);

drop table discipline_avg;
```

Скриншот до:

	id_discipline	name	selfstudy_...	practice_h...	lecture_hours
1	0	name0	5	5	5
2	1	name1	15	15	15
3	2	name2	25	25	25
4	3	name3	35	35	35
5	4	name4	45	45	45
6	5	name5	55	55	55
7	6	name6	65	65	65
8	7	name7	75	75	75
9	9	name9	95	95	95
10	10	name10	105	105	105
11	11	name11	115	115	115
12	12	name12	125	125	125
13	13	name13	135	135	135
14	14	name14	145	145	145
15	15	name15	155	155	155
16	16	name16	165	165	165

Скриншот после:

	id_discip... ^ 1	name	selfstudy_... ÷	practice_h... ÷	lecture_hours ÷
1	0	name0	5	5	5
2	1	name1	15	15	15
3	2	name2	25	25	25
4	3	name3	35	35	35
5	4	name4	55	45	45
6	5	name5	55	55	55
7	6	name6	65	65	65
8	7	name7	75	75	75
9	8	nice discipline	85	85	85
10	9	name9	95	95	95
11	10	name10	105	105	105
12	11	name11	115	115	115
13	12	name12	125	125	125
14	13	name13	135	135	135
15	14	name14	145	145	145
16	15	name15	155	155	155

### Запрос №1 на удаление

Формулировку запроса:

Удалить подразделение, в котором не обучается ни один студент и не трудоустроен ни один преподаватель

Команда:

```
delete from subdivisions
where id_subdivision not in
    (select distinct id_subdivision from study_plans
    where id_study_plan in
        (select distinct id_study_plan from groups
        where id_group in
            (select distinct id_group from student_groups))

union

select distinct id_subdivision from employment_contracts);
```

Скриншот до:

	🔍 id_subdivi... ▾	🔍 name ▾	🔍 headquarte... ▾	🔍 type ▾
1	0	name 0	address 0	основной
2	1	name 1	address 1	основной
3	2	name 2	address 2	основной
4	3	name 3	address 3	основной
5	4	name 4	address 4	основной
6	5	name 5	address 5	филиал
7	6	name 6	address 6	филиал
8	7	name 7	address 7	филиал
9	8	name 8	address 8	филиал
10	9	name 9	address 9	филиал

Скриншот после:

	🔍 id_subdivision ▾	🔍 name ▾	🔍 headquarter... ▾	🔍 type ▾
1	0	name 0	address 0	основной
2	1	name 1	address 1	основной
3	2	name 2	address 2	основной
4	3	name 3	address 3	основной
5	4	name 4	address 4	основной
6	5	name 5	address 5	филиал
7	6	name 6	address 6	филиал
8	7	name 7	address 7	филиал
9	8	name 8	address 8	филиал
10	9	name 9	address 9	филиал

## Запрос №2 на удаление

Формулировку запроса:

Удалить учебные планы, которые были созданы позже всего и не реализует ни одной дисциплины





Команда:

```
delete from study_plans
where id_study_plan in
```







```
(select id_study_plan from study_plans
where id_study_plan not in
      (select distinct id_study_plan from study_courses)
and year>=all(select year from study_plans));
```

Скриншот до:

	 id_study_plan ▾	 id_subdivi... ▾	 year ▾	 id_edu_track ▾
1	0	0	2010	0
2	1	0	2010	0
3	2	0	2011	0
4	3	0	2011	1
5	4	0	2012	1
6	5	1	2012	1
7	6	1	2013	2
8	7	1	2013	2
9	8	1	2014	2
10	9	1	2014	3

Скриншот после:

	 id_study_plan ▾	 id_subdivision ▾	 year ▾	 id_edu_track ▾
1	0	0	2010	0
2	1	0	2010	0
3	2	0	2011	0
4	3	0	2011	1
5	4	0	2012	1
6	5	1	2012	1
7	6	1	2013	2
8	7	1	2013	2

### Запрос №1 на вставку

Формулировка запроса:

Для студента с id\_record\_book=2 во время его обучения в группе с id\_group=0. Если по дисциплине была получена 3 на 1-й попытке, то добавить 4 со второй попытки

Команда:

```
insert into record_book
```

```

(id_group_row, id_teacher, id_study_course, mark, try_number)
select id_group_row, id_teacher, id_study_course, 4, 2 from
record_book
where
    id_group_row =
    (select id_group_row from student_groups
    where id_group=0 and id_record_book=2)
and
    mark=3
and
    try_number=1;

```

Скриншот до:

	id_row_record_book	id_group_row	id_teacher	id_study_course	mark	try_number
7	46	4	6	6	3	1
8	47	4	7	7	5	1
9	48	4	8	8	5	3
10	49	4	9	9	5	1
11	448	4	0	0	2	1
12	449	4	0	0	2	2
13	450	4	1	1	2	1
14	451	4	3	3	2	1
15	452	4	5	5	2	1
16	453	4	8	8	2	1
17	454	4	8	8	2	2

Скриншот после:

	id_row_record_book	id_group_row	id_teacher	id_study_course	mark	try_number
1	40	4	0	0	3	3
2	41	4	1	1	5	2
3	42	4	2	2	5	1
4	43	4	3	3	5	2
5	44	4	4	4	5	1
6	45	4	5	5	3	2
7	46	4	6	6	3	1
8	47	4	7	7	5	1
9	48	4	8	8	5	3
10	49	4	9	9	5	1
11	817	4	6	6	4	2

## Создание индексов.

### Запрос с составным индексом

#### Формулировка запроса:

Список студентов, получивших двойки на первой попытке с указанием фамилии преподавателя, которым они должны передать экзамен.

#### Команда:

```

select s.surname as student_name, t.surname as teacher_surname
from record_book
join student_groups sg

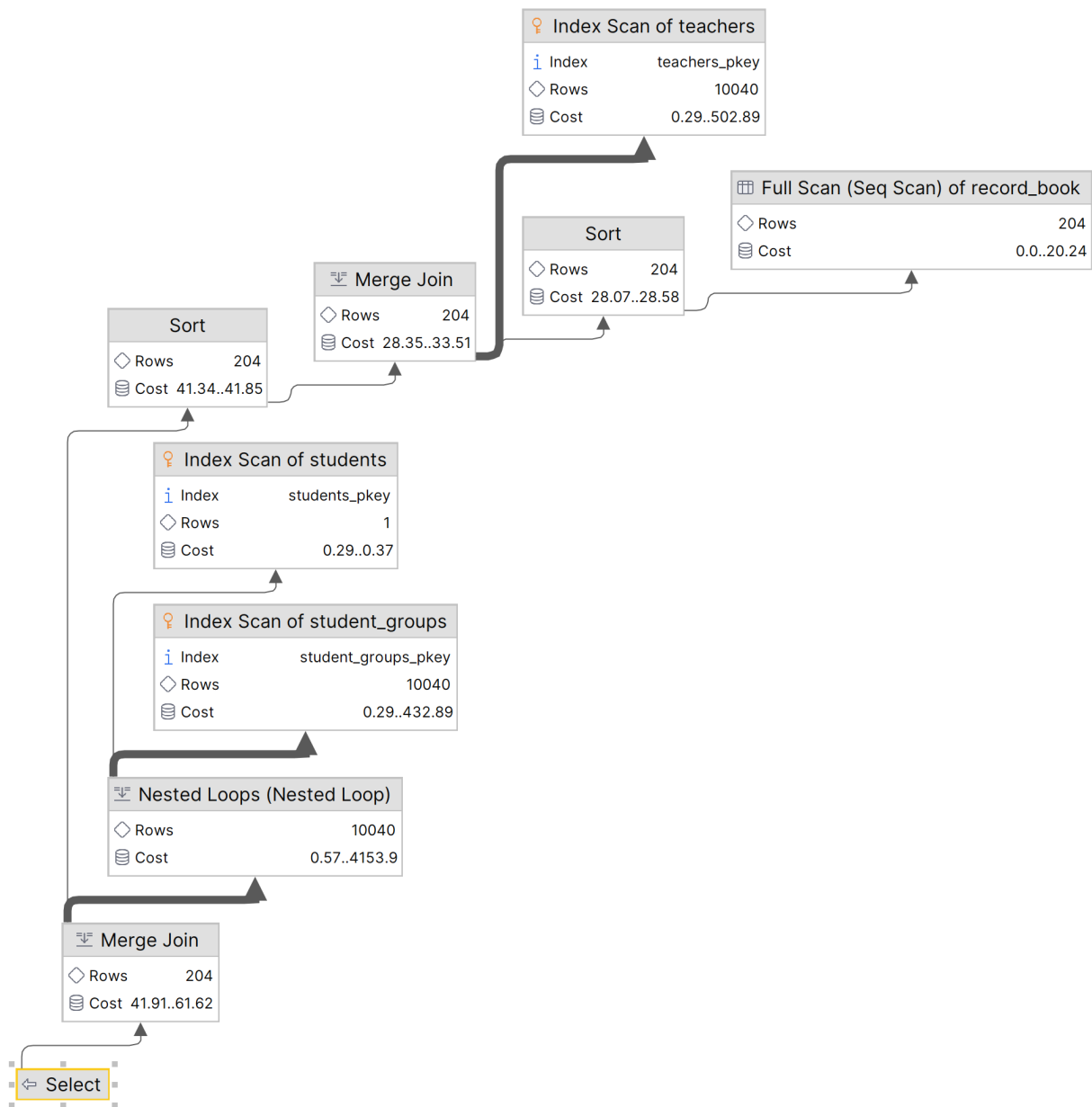
```

```

on sg.id_group_row = record_book.id_group_row
join students s
on sg.id_record_book = s.id_record_book
join teachers t
on record_book.id_teacher = t.id_teacher
where try_number=1 and mark=2;

```

**План операции до создания индексов:**



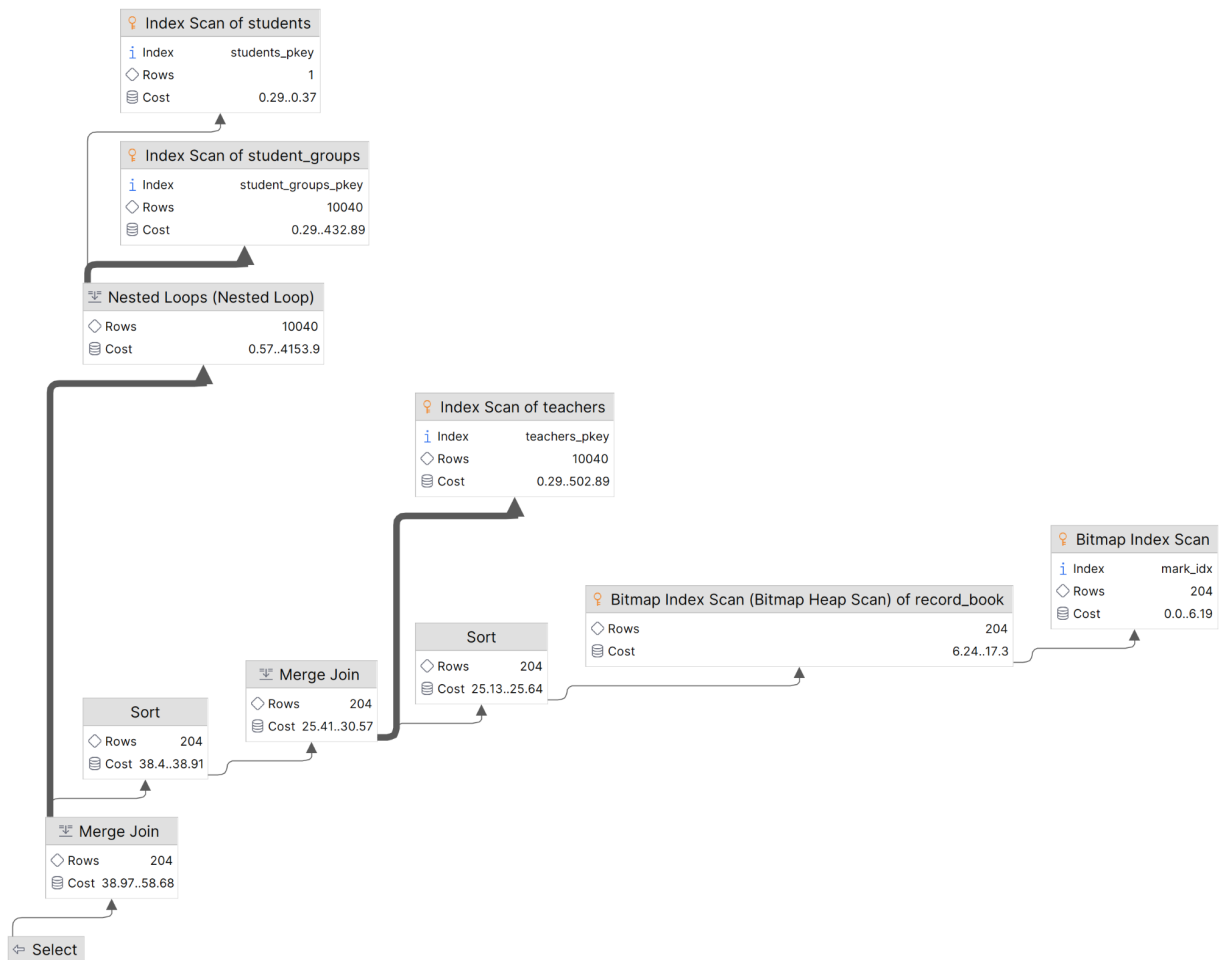
**Время до создания индексов:**

437ms

### Добавление индексов:

```
create index try_mark_idx on record_book(try_number, mark);  
create index record_book_idteacher_idx on  
record_book(id_teacher);  
create index record_book_group_row_idx on  
record_book(id_group_row);  
create index group_recordbook_idx on  
student_groups(id_record_book);
```

### План операции после создания индексов:



**Вывод:** видим, что использовался только mark\_idx. Это связано с тем, индексы, которые я добавил на foreign keys на самом деле бесполезные в данном запросе, ибо мы не отфильтруем значения по ним. У нас уже есть значения, которые мы получили из предыдущей таблицы и по ним мы ищем primary key в другой таблице.

Индексы для FK полезны только при подзапросах, а не объединении. Пример будет далее по тексту.

**Время после создания индексов:**

58ms

**Запрос без составных индексов**

**Формулировка запроса:**

Получить количество 5-к по дисциплинам у студента с surname=`student_surname4`

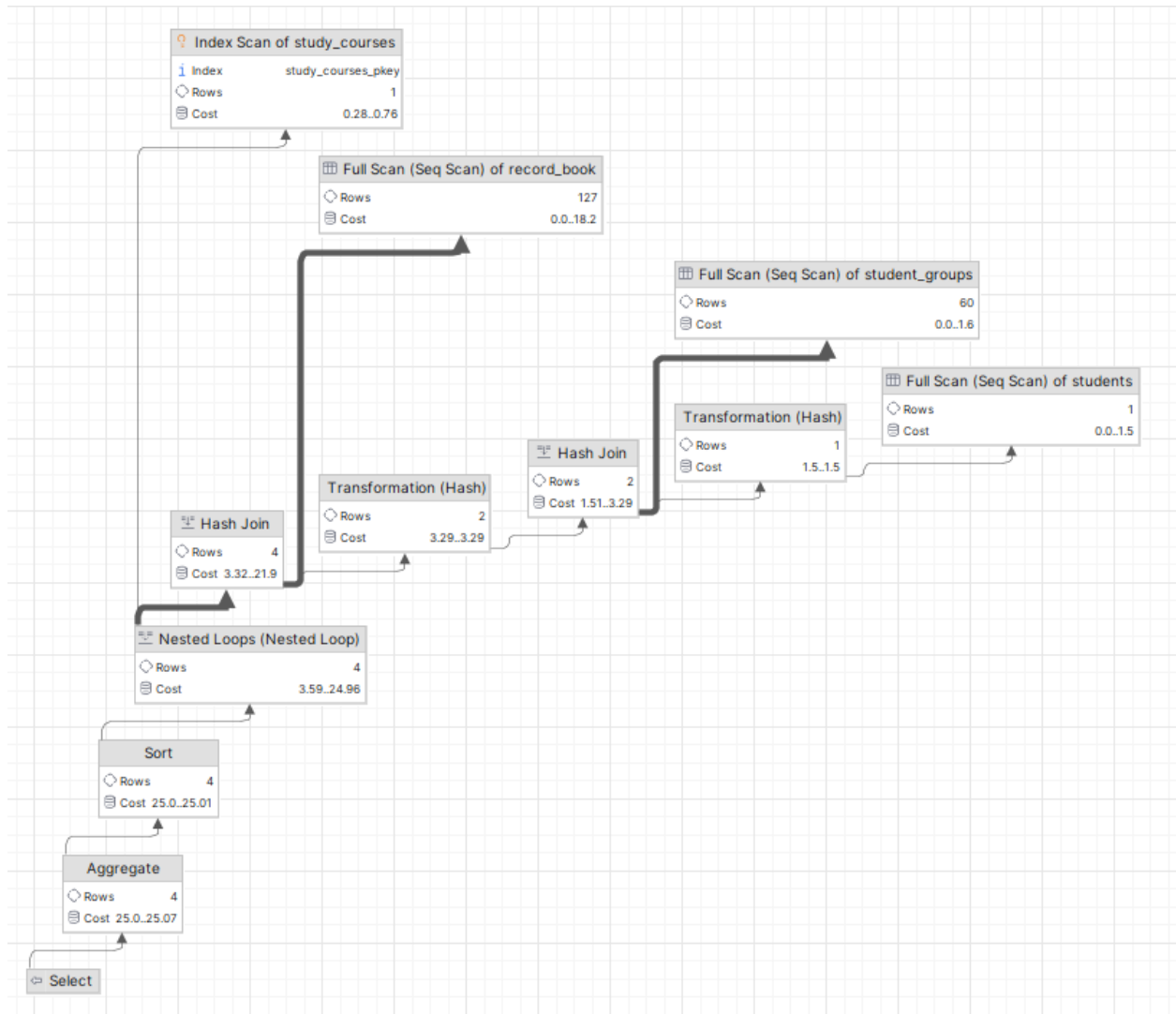
**Команда:**

```
select id_discipline, count(*) cnt_5 from record_book
join study_courses sc
  on record_book.id_study_course = sc.id_study_course
where
  id_group_row in
    (select id_group_row from student_groups where id_record_book
in
      (select id_record_book from students where
surname='student_surname4'))
  and
  mark=5
group by id_discipline;
```

**Время до создания индексов:**

68ms

**План операции до создания индексов:**



### Добавление индексов:

```

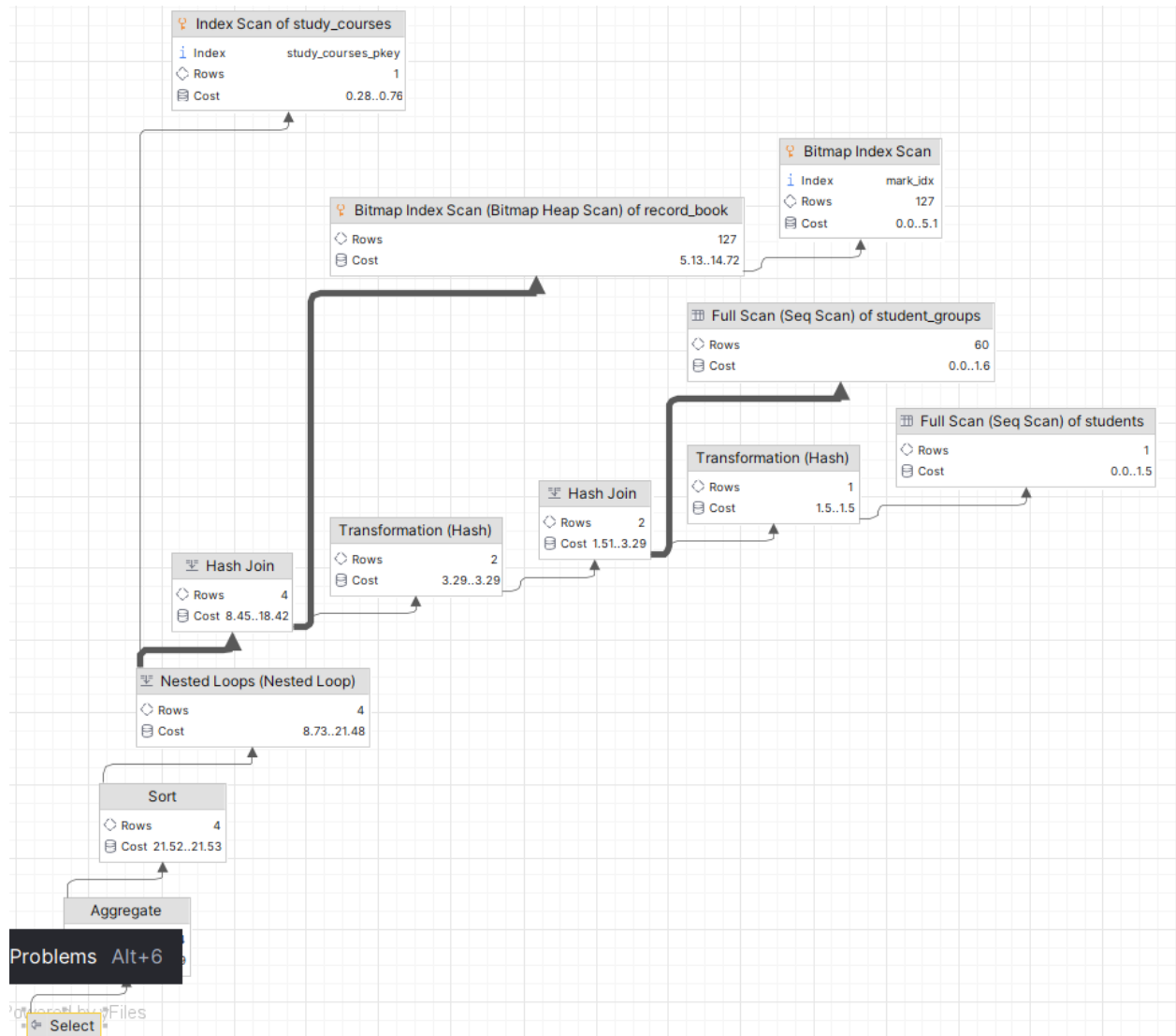
create index mark_idx on record_book(mark);
create index student_surname_idx on students(surname);
create index group_recordbook_idx on student_groups(id_record_book);
  
```

### Время после создания индексов:

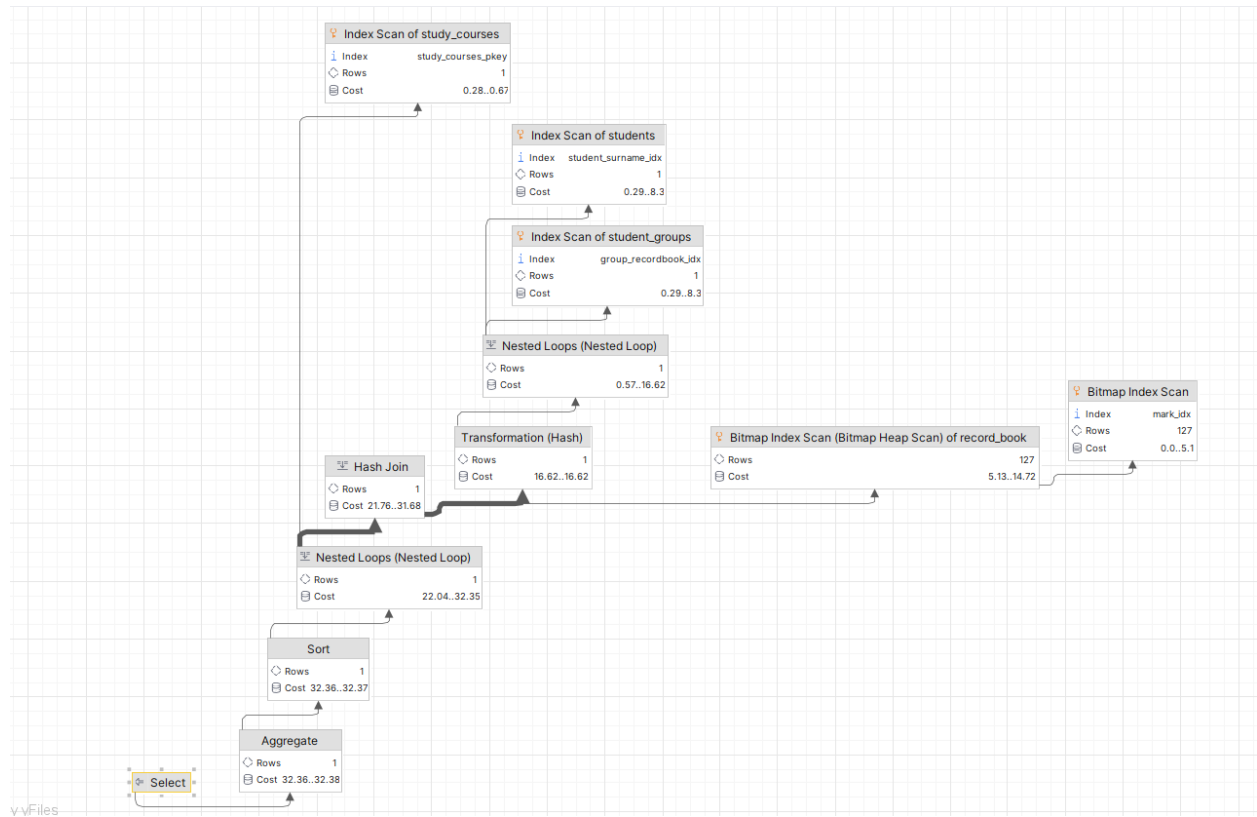
48ms

### План операции после создания индексов:

План до добавления дополнительных студентов



План после добавления 10000 студентов:



**Вывод:** Такое различие обусловлено тем, что при небольшом количестве записей не имеет смысла использовать индексы. Гораздо проще перебрать все значения

## Выводы

В ходе проделанной лабораторной работы я:

- Написал запросы select, delete, update, inset
- Создал представления
- Создал индексы для оптимизации работы БД