

Министерство науки и высшего образования Российской Федерации  
Федеральное государственное автономное образовательное учреждение  
высшего образования  
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»  
Факультет инфокоммуникационных технологий

**ОТЧЕТ**  
**О ЛАБОРАТОРНОЙ РАБОТЕ № 4**

по теме:

**ЗАПРОСЫ НА ВЫБОРКУ И МОДИФИКАЦИЮ ДАННЫХ. ПРЕДСТАВЛЕНИЯ. РАБОТА С  
ИНДЕКСАМИ**

по дисциплине: **Проектирование и реализация баз данных**

Специальность:

**09.03.03 Мобильные и сетевые технологии**

Проверила:

Говорова М.М.

Дата: \_\_\_\_\_ 2023 г.

Оценка

Выполнил:

студента группы К3239

Прокопец С. Р.

Санкт-Петербург 2023/2024

**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.

**Предметная область:** Вариант 16. БД "Спортивный клуб"

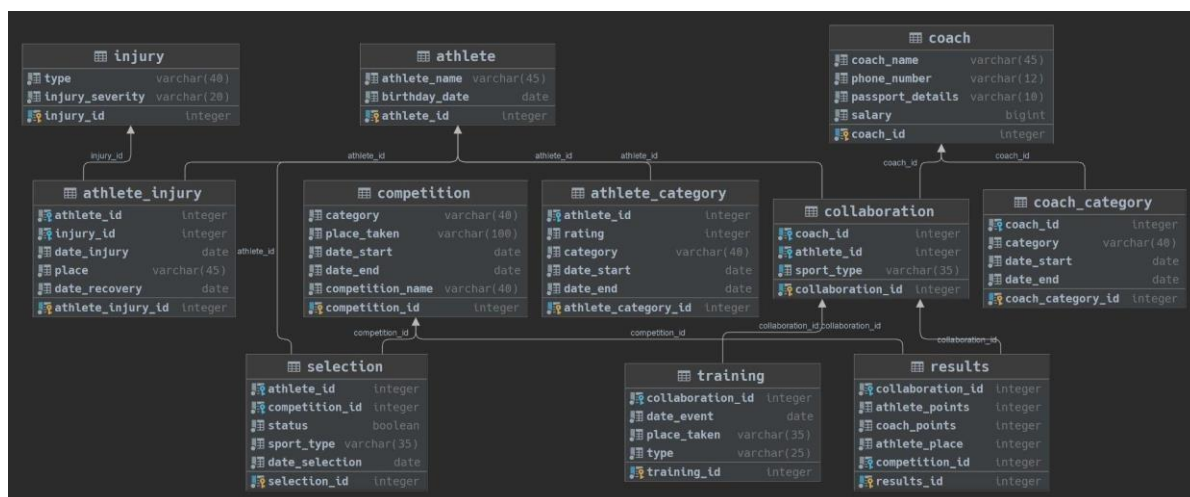


Рисунок 1 – ERD базы данных

**Выполнение работы:**

Запросы на выборку:

Найти тренера с минимальной зарплатой:

```
select coach_name AS Имя_тренера, salary AS Зарплата  
From coach c  
where salary = (  
    select MIN(c2.salary)  
    From coach c2  
);
```

Output Result 7 x

1 row v | | | |

	Имя_тренера	Зарплата
1	Белогаев Даниил	1

2. С каким количеством спортсменов работает тренер

```
select coach_name Имя_тренера, count(athlete_id) количество_спортсменов
From coach c
      left join collaboration c2 on c.coach_id = c2.coach_id
group by c.coach_id, coach_name
order by 2 desc, 1;
```

Output		Result 8	
6 rows		CSV	
Имя_тренера		количество_спортсменов	
1	Караваев Владимир	3	
2	Матченя Иван	1	
3	Родин Михаил	1	
4	Белогаев Даниил	0	

3. найти тренеров ,у который спортсмены не имеют травм

```
select coach_name as имя_тренера
from coach c where not exists(
    select 1
    from collaboration c2
        JOIN public.athlete a on c2.athlete_id = a.athlete_id
        join athlete_injury ai on a.athlete_id = ai.athlete_id
    where c.coach_id = c2.coach_id
)
```

Output имя\_тренера:varchar(45) ×

5 rows

имя_тренера
1 Родин Михаил
2 Федукин Александр
3 Матченя Иван
4 Белогаев Даниил
5 Карякинин Сергей

4. Сколько спортсменов участвуют в каждой категории?

```
select
    category as категория,
    count(a.athlete_id) as количество_спортсменов
from athlete a
join selection s on a.athlete_id = s.athlete_id
join competition c on s.competition_id = c.competition_id
where status is true
and date_selection between '2023-12-1' and '2000-01-31'
group by category
order by 2 desc , 1;
```

⌕ | +

athlete  
athlete  
console\_9  
console\_11 49 ms

Output Result 11 ×

< < 0 rows > > | ↺ ⌚ ■ | 📌 CSV ▾ | ⬇ ⬆ ⬇ ⬇ | 🔍

категория ⌵

количество\_спортсменов ⌵

5 для спортсменов определить количество соревнований в которых они участвовали.

```
select
  a.athlete_name as Имя_спортсмена,
  count(s.competition_id) as количество_соревнований
  from athlete a
left join selection s on a.athlete_id = s.athlete_id
where status is true
group by a.athlete_id, a.athlete_name
order by 2 desc , 1;
```

во\_соревнований

Output Result 16 x

10 rows v | | | |

	Имя_спортсмена	количество_соревнований
1	Бахилкин Никита	1
2	Беликова Алиса	1
3	Коник Анастасия	1
4	Курмышов Владимир	1
5	Мищенко Кира	1
6	Смалденская Кира	1

6. Найти тренера, работающего с самыми молодыми спортсменами (средний возраст спортсменов минимален)

```
1 with t as (  
2     select  
3         c.coach_name as Имя_тренера,  
4         round(avg(extract(year from age(CURRENT_DATE, a.birthday_date)))) as Средний_возраст_спортсменов  
5     from coach c  
6     join collaboration c2 on c.coach_id = c2.coach_id  
7     join public.athlete a on c2.athlete_id = a.athlete_id  
8     group by c.coach_id, c.coach_name  
9  
10  
11 )  
12 select Имя_тренера, Средний_возраст_спортсменов  
13 from t  
14 where Средний_возраст_спортсменов = (select min(Средний_возраст_спортсменов) from t)
```

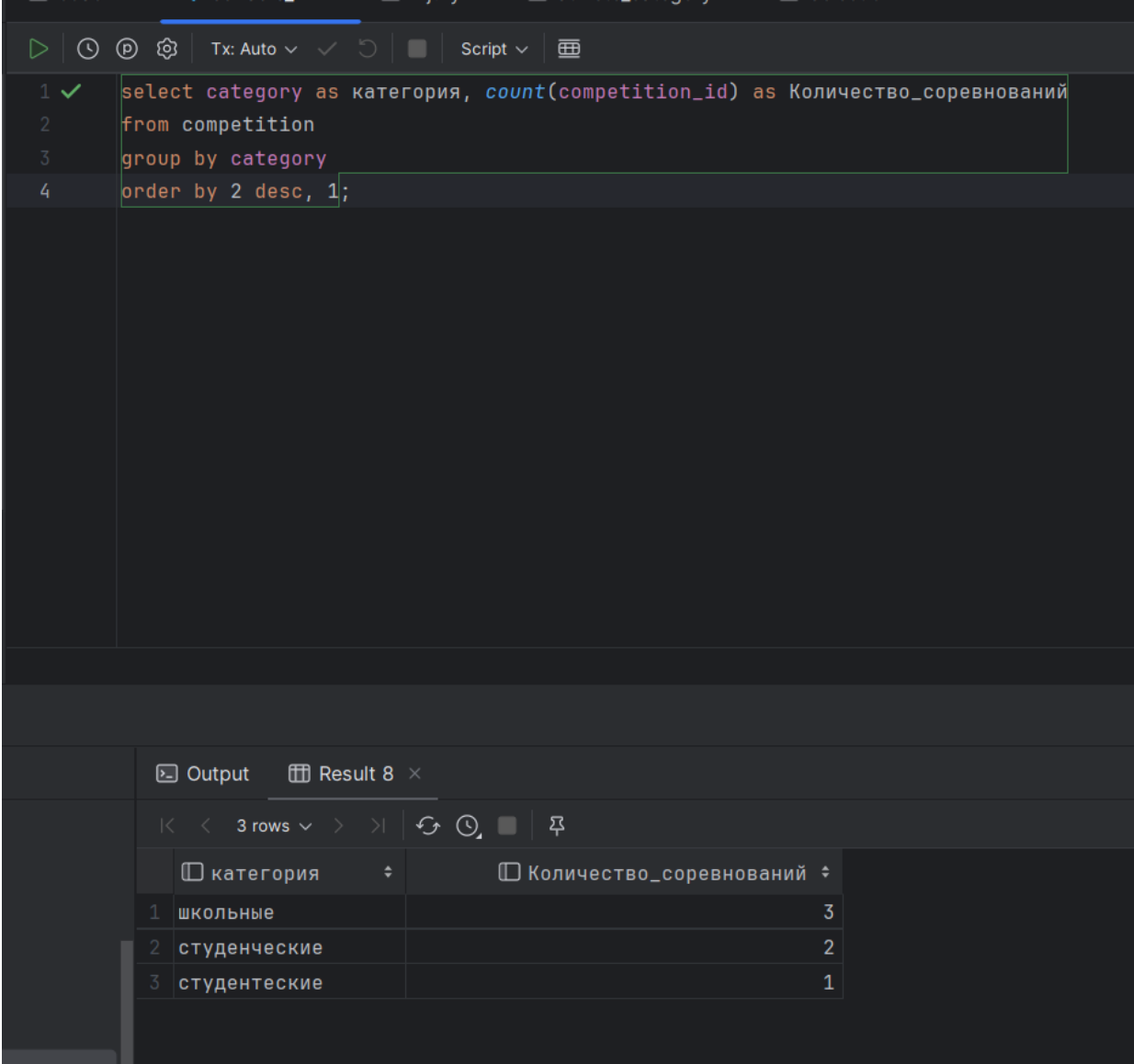
Output Result 7 x

1 row v < > | ↺ ⌚ ⌛ ⚙

	Имя_тренера	Средний_возраст_спортсменов
1	Родин Михаил	18



7. Определить количество соревнований каждой категории.



The screenshot shows a SQL IDE interface. The top toolbar includes icons for running, refreshing, and saving, along with a dropdown menu set to 'Tx: Auto'. The main editor area contains a SQL query with line numbers 1 through 4. The query is: `select category as категория, count(competition_id) as Количество_соревнований from competition group by category order by 2 desc, 1;`. Below the editor, the 'Output' pane is active, displaying 'Result 8'. It shows a table with 3 rows. The table has two columns: 'категория' and 'Количество\_соревнований'. The data rows are: 1. школьные (3), 2. студенческие (2), 3. студенческие (1).

```
1 ✓ select category as категория, count(competition_id) as Количество_соревнований
2   from competition
3   group by category
4   order by 2 desc, 1;
```

Output Result 8 ×

3 rows

	категория	Количество_соревнований
1	школьные	3
2	студенческие	2
3	студенческие	1

## Создание представлений

```
create view coach_achievements as (  
select  
    c.coach_name as имя_тренера,  
    a.athlete_name as имя_спортсмена,  
    competition_name as название_соревнования,  
    c2.sport_type_id as дисциплина,  
  
    r.athlete_points as баллы_спортсмена,  
    r.athlete_place as занятое_место  
from coach c  
    join collaboration c2 on c.coach_id = c2.coach_id  
    join athlete a on a.athlete_id = c2.athlete_id  
    join results r on c2.collaboration_id = r.collaboration_id  
    join competition c3 on c3.competition_id = r.competition_id  
order by занятое_место desc , название_соревнования, имя_спортсмена
```

имя_тренера	имя_спортсмена	название_соревнования	дисциплина	баллы_спортсмена	занятое_место
Караваев	ямалетдинова	первенство округа	chess	10	1
Караваев	станкевич	первенство округа	chess	10	2
Караваев	джаватов	первенство округа	chess	10	3
Федукин	курмышов	первенство округа	chess	5	4
Карякин	беликова	первенство округа	chess	5	5
chess	белов	первенство округа	chess	1	6
Карякин	якунин	первенство округа	chess	1	7

## Найти самую распространенную травму.

```
CREATE VIEW most_common_injury AS (  
    WITH t AS (  
        SELECT  
            type AS Травма,  
            injury_severity AS Тяжесть,  
            COUNT(DISTINCT athlete_id) AS Количество_спортсменов_с_данной_травмой  
        FROM injury i  
            JOIN athlete_injury ai on i.injury_id = ai.injury_id  
        GROUP BY i.injury_id  
    )  
  
    SELECT *  
    FROM t  
    WHERE Количество_спортсменов_с_данной_травмой = (SELECT MAX(Количество_спортсменов_с_данной_травмой) FROM t)  
    ORDER BY  
        CASE Тяжесть  
            WHEN 'очень тяжелая' THEN 1  
            WHEN 'тяжелая' THEN 2  
            WHEN 'средняя' THEN 3  
        END,  
        Травма  
)
```

	Травма	Тяжесть	Количество_спортсменов_с_данной_т
1	вывих пальца	очень тяжелая	7
2	перелом ребра	очень тяжелая	7
3	разрыв связки голеностопа	очень тяжелая	7
4	травма седалищного нерва	очень тяжелая	7
5	травма двуглавой мышцы	тяжелая	7
6	ушиб груди	средняя	7

Сравнение запросов с использованием индексирования

## 1. Простой индекс Время без индекса:

```

QUERY PLAN
1 Seq Scan on athlete (cost=0.00..23.50 rows=1 width=57) (actual time=0.243..0.244 rows=1 loops=1)
2   Filter: ((athlete_name)::text = 'Николаева Ольга Фёдоровна'::text)
3   Rows Removed by Filter: 999
4 Planning Time: 0.093 ms
5 Execution Time: 0.262 ms

```

Время с индексом:

```

QUERY PLAN
1 Index Scan using idx_athlete_name on athlete (cost=0.28..8.29 rows=1 width=57) (actual time=0.019..0.020 rows=1 loops=1)
2   Index Cond: ((athlete_name)::text = 'Николаева Ольга Фёдоровна'::text)
3 Planning Time: 0.053 ms
4 Execution Time: 0.031 ms

```

Создание индекса:

```
CREATE INDEX idx_athlete_name ON athlete(athlete_name);
```

## 2. Составной индекс

Время без индекса:

```

QUERY PLAN
1 Seq Scan on athlete (cost=0.00..26.00 rows=1 width=57) (actual time=0.266..0.267 rows=1 loops=1)
2   Filter: (((athlete_name)::text = 'Николаева Ольга Фёдоровна'::text) AND (birthday_date = '1998-03-02'::date))
3   Rows Removed by Filter: 999
4 Planning Time: 0.119 ms
5 Execution Time: 0.289 ms

```

Время с индексом:

```

QUERY PLAN
1 Index Scan using idx_athlete_name_birthday on athlete (cost=0.28..8.29 rows=1 width=57) (actual time=0.023..0.023 rows=1 loops=1)
2   Index Cond: (((athlete_name)::text = 'Николаева Ольга Фёдоровна'::text) AND (birthday_date = '1998-03-02'::date))
3 Planning Time: 0.057 ms
4 Execution Time: 0.034 ms

```

Как мы видим, индексы очень сильно сокращают время выполнения запроса.

Вывод:

В рамках данной лабораторной работы были разработаны и выполнены запросы на выборку данных, а также были созданы представления для базы данных PostgreSQL в соответствии с поставленной индивидуальной задачей. Кроме того, мы успешно реализовали разнообразные запросы на модификацию данных.

Особое внимание уделялось анализу графического представления всех запросов. Мы провели создание как простых, так и составных индексов, а также проанализировали время выполнения запросов при их использовании. Этот процесс позволил нам оптимизировать производительность базы данных и улучшить время выполнения запросов.

В итоге данной работы мы получили практические навыки работы с базой данных PostgreSQL, а также глубокое понимание важности создания эффективных индексов для повышения производительности системы. Эти знания окажутся ценными при проектировании и оптимизации баз данных.