

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

Отчет

по лабораторной работе № 2
**«ЗАПРОСЫ ФНА ВЫБОРКУ И МОДИФИКАЦИЮ ДАННЫХ,
ПРЕДСТАВЛЕНИЯ И ИНДЕКСЫ В POSTGRESQL»**
по дисциплине **«Базы данных»**

Автор: Чаптыков Николай

Факультет: ИКТ

Группа: K32422

Преподаватель: Говорова М.М.



УНИВЕРСИТЕТ ИТМО

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
2. Составить три 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.

Описание варианта:

1) БД «Спортивный клуб»

2) Состав реквизитов сущностей

Спортсмен (ФИО, паспортные данные, телефон, ID спортсмена, квалификация спортсмена)

Тренер (ФИО, номер паспорта, телефон, ID тренера, квалификация тренера)

Тренировка (ID тренировки, время конца тренировки, время начала тренировки, дата проведения, место проведения)

Соревнование (ID соревнования, место проведения соревнования, дата начала проведения соревнования, вид соревнования, категория соревнования, вид спорта, дата конца проведения соревнования, название соревнования)

Квалификация тренера (код квалификации, С, По, код должности)

Должность (Код должности, оклад тренера, название должности)

Квалификация спортсмена (история квалификаций, рейтинг спортсмена, квалификация спортсмена)

Ход работы:

Задание 1

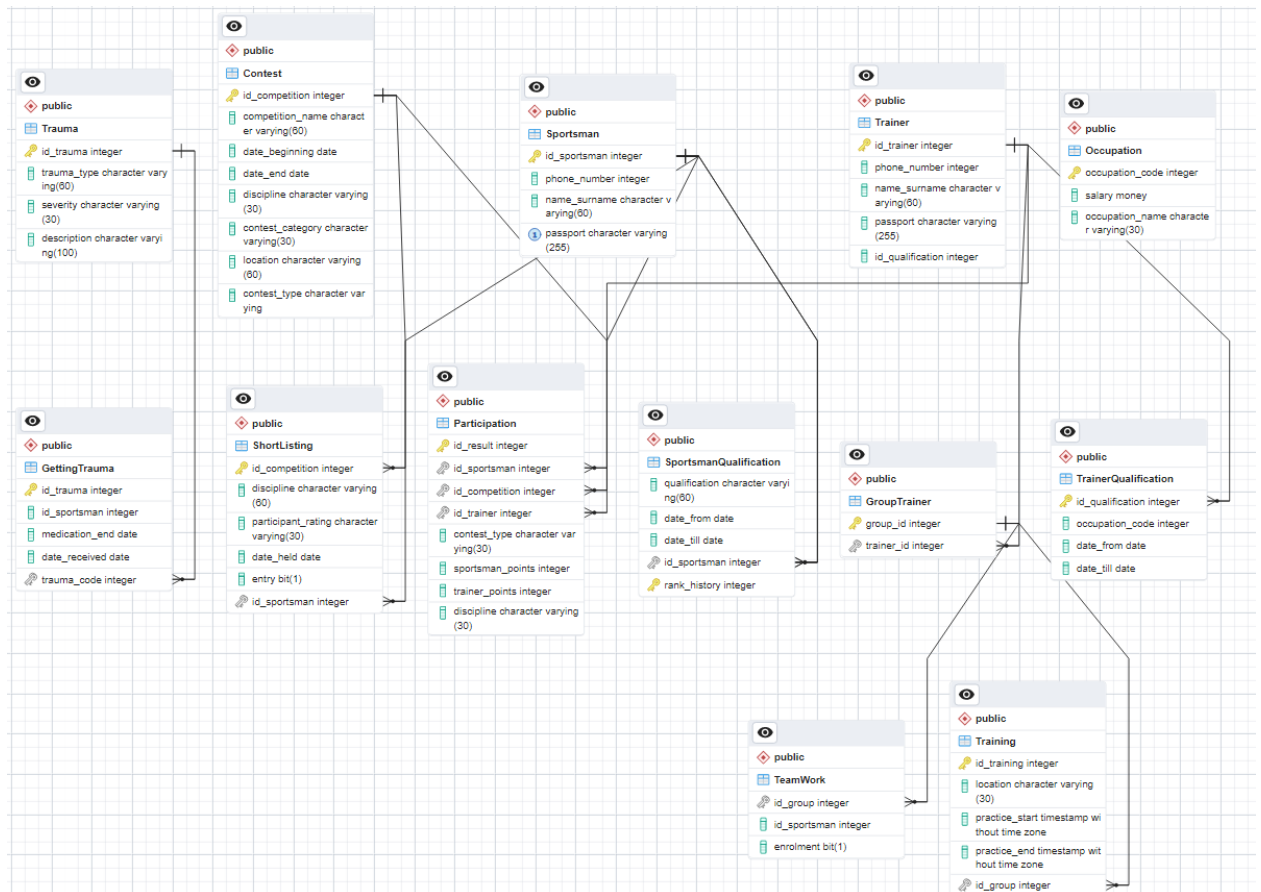


Рисунок 1 - ERD схема базы данных

Задание 2

С каким количеством спортсменов работает каждый тренер:

```

1 SELECT "Trainer".id_trainer, COUNT("TeamWork".id_sportsman) as cnt from "Trainer"
2 LEFT JOIN "GroupTrainer" ON "Trainer".id_trainer = "GroupTrainer".trainer_id
3 LEFT JOIN "TeamWork" ON "TeamWork".id_group = "GroupTrainer".group_id
4 GROUP BY "Trainer".id_trainer
5 ORDER BY cnt DESC
6

```

	id_trainer [PK] integer	cnt bigint	
1	5	10	
2	3	10	
3	17	10	
4	11	10	
5	7	9	
6	35	0	
7	45	0	
8	6	0	
9	39	0	
10	36	0	
11	31	0	

Найти тренеров, чьи спортсмены не имеют травм:

```
1 select DISTINCT "GroupTrainer".trainer_id from "GroupTrainer"
2 INNER JOIN "TeamWork" ON "TeamWork".id_group = "GroupTrainer".group_id
3 WHERE not "TeamWork".id_sportsman = ANY (SELECT "GettingTrauma".id_sportsman FROM "GettingTrauma")
4
```

	trainer_id integer
1	3
2	5
3	7
4	11
5	17

Найти тренера, получающего минимальную зарплату:

```
1 SELECT "Trainer".id_trainer, "Occupation".salary FROM "Trainer"
2 LEFT JOIN "TrainerQualification" ON "Trainer".id_qualification = "TrainerQualification".id_qualification
3 INNER JOIN "Occupation" ON "Occupation".occupation_code = "TrainerQualification".occupation_code
4 WHERE "Occupation".salary = (SELECT MIN(salary) FROM "Occupation")
5 GROUP BY "Occupation".salary, "Trainer".id_trainer
6 ORDER BY "Occupation".salary ASC
7
```

	id_trainer integer	salary money
1	19	5 731,00 ?

Определить количество соревнований каждой категории:

Query Query History

```
1 SELECT DISTINCT "Contest".contest_category, COUNT("Contest".id_competition) FROM "Contest"
2 GROUP BY "Contest".contest_category
```

	contest_category character varying (30)	count bigint
1	Мировой	2
2	Региональный	5
3	Галактический	3
4	Школьный	5
5	Муниципальный	4

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

```
1 SELECT "GroupTrainer".trainer_id,
2       to_timestamp(avg(extract(epoch from "Sportsman".birthdate))) as avg_date FROM "GroupTrainer"
3 -- avg от date почему-то не хотел работать
4 INNER JOIN "TeamWork" ON "GroupTrainer".group_id = "TeamWork".id_group
5 INNER JOIN "Sportsman" ON "TeamWork".id_sportsman = "Sportsman".id_sportsman
6 GROUP BY "GroupTrainer".trainer_id
7 ORDER BY avg_date DESC
```

	trainer_id integer	avg_date timestamp with time zone
1	3	2002-10-27 07:48:00+03

Сколько спортсменов участвует в соревнованиях каждой категории в заданный период:

```
1 SELECT DISTINCT "Participation".contest_type, COUNT("Participation".id_sportsman) FROM "Participation"
2 INNER JOIN "Contest" ON "Contest".id_competition = "Participation".id_competition
3 WHERE ("Contest".date_beginning <= '2020-12-20') and ('2020-12-20' <= "Contest".date_end)
4 GROUP BY "Participation".contest_type
```

	contest_type character varying (30)	count bigint
1	Турнир-Десятиборье	1
2	Турнир-Пятиборье	1
3	Турнир-Семиборье	2
4	Командный турнир	2

Для всех спортсменов определить количество соревнований, в которых они участвовали:

```
1 SELECT DISTINCT "Participation".contest_type, COUNT("Participation".id_sportsman) FROM "Participation"
2 INNER JOIN "Contest" ON "Contest".id_competition = "Participation".id_competition
3 WHERE ("Contest".date_beginning <= '2020-12-20') and ('2020-12-20' <= "Contest".date_end)
4 GROUP BY "Participation".contest_type
```

	id_sportsman [PK] integer	cnt bigint
1	0	1
2	1	2
3	2	1
4	5	1
5	7	2
6	8	1
7	9	2
8	10	2
9	12	3
10	14	1
11	15	3

Задание 3

Необходимо составить следующие представления:

1. Представление, содержащее сведения обо всех тренерах, соревнованиях, в которых участвовали их спортсмены и местах, которые они заняли:

```

1 SELECT DISTINCT "Trainer".name_surname, "Trainer".id_trainer,
2     "Sportsman".id_sportsman, "Sportsman".name_surname, "Contest".competition_name,
3     "Participation".sportsman_points FROM "Trainer"
4 INNER JOIN "GroupTrainer" ON "GroupTrainer".trainer_id = "Trainer".id_trainer
5 INNER JOIN "TeamWork" ON "GroupTrainer".group_id = "TeamWork".id_group
6 INNER JOIN "Sportsman" ON "TeamWork".id_sportsman = "Sportsman".id_sportsman
7 INNER JOIN "Participation" ON "Sportsman".id_sportsman = "Participation".id_sportsman
8 INNER JOIN "Contest" ON "Participation".id_competition = "Contest".id_competition
9 GROUP BY "Trainer".name_surname, "Trainer".id_trainer,
10     "Sportsman".id_sportsman, "Sportsman".name_surname,
11     "Participation".sportsman_points, "Contest".competition_name
12 ORDER BY "Sportsman".name_surname DESC

```

1 SELECT * FROM ContestResults

	name_surname character varying (60)	id_trainer integer	id_sportsman integer	name_surname character varying (60)	competition_name character varying (60)	sportsman_p integer
1	Нурлан Любимов	3	21	Станислав Цой	Галактический Турнир имени Дзержинского	31
2	Нурлан Любимов	3	21	Станислав Цой	Региональный Турнир-Десятиборье имени Дзержинского	2
3	Нурлан Любимов	3	21	Станислав Цой	Региональный Турнир-Десятиборье имени Дзержинского	34
4	Нурлан Любимов	3	21	Станислав Цой	Школьный Турнир-Семиборье имени Пола Пота	42
5	Нурлан Любимов	3	23	Станислав Цой	Муниципальный Турнир имени Дзержинского	74
6	Нурлан Любимов	3	23	Станислав Цой	Региональный Турнир имени Циолковского	52
7	Аяз Любимов	11	7	Станислав Бондаренко	Мировой Турнир-Пятиборье имени Пола Пота	82
8	Аяз Любимов	11	7	Станислав Бондаренко	Региональный Турнир-Пятиборье имени Королёва	10
9	Нурлан Любимов	3	29	Сергей Хёнгю	Муниципальный Турнир-Пятиборье имени Ленина	31
10	Аяз Любимов	11	1	Сергей Вольных	Галактический Турнир имени Дзержинского	16
11	Аяз Любимов	11	1	Сергей Вольных	Региональный Турнир-Десятиборье имени Дзержинского	83

2. Найти самую распространенную травму:

```

3 CREATE VIEW FrequentTrauma AS
4 SELECT DISTINCT "Trauma".trauma_type, "Trauma".severity, COUNT("GettingTrauma".trauma_code) as cnt FROM "Trauma"
5 INNER JOIN "GettingTrauma" ON "GettingTrauma".trauma_code = "Trauma".id_trauma
6 GROUP BY "Trauma".trauma_type, "Trauma".severity
7 ORDER BY cnt DESC

```

1 SELECT * FROM FrequentTrauma

	trauma_type character varying (60)	severity character varying (30)	cnt bigint
1	Перелом	Незначительная	8
2	Перелом	Тяжелая	7
3	Растяжение	Несовместимая с жизнью	2
4	Ожог	Несовместимая с жизнью	1
5	Травма головы	Легкая	1

Задание 4

Выполнение INSERT:

```

1  --Создадим новую группу для тренера--
2  INSERT INTO "GroupTrainer" (group_id, trainer_id)
3  VALUES (420, (SELECT id_trainer FROM "Trainer" WHERE id_trainer = 15))

```

	group_id [PK] integer	trainer_id integer
1	7	28
2	8	5
3	9	3
4	10	17
5	11	7
6	12	13

	group_id [PK] integer	trainer_id integer
1	7	28
2	8	5
3	9	3
4	10	17
5	11	7
6	12	13
7	420	15

Рисунок 2 – до/после INSERT

Выполнение UPDATE:

```

1  --Сменим тренера для группы Аяза Поклонского--
2  UPDATE "GroupTrainer" SET trainer_id = 15
3  WHERE "GroupTrainer".group_id = (SELECT trainer_id FROM "Trainer" WHERE name_surname = "Аяз Поклонский")

```

	group_id [PK] integer	trainer_id integer
1	7	28

	group_id [PK] integer	trainer_id integer
1	7	15

Рисунок 3 – до/после выполнения UPDATE

Выполнение DELETE:

```

1  --Удалим старую группу нового тренера--
2  DELETE FROM "GroupTrainer"
3  WHERE "trainer_id" = (SELECT id_trainer FROM "Trainer" WHERE "Trainer".name_surname = 'Василий Любимов')
4  and "group_id" = 420
5

```

	group_id [PK] integer	trainer_id integer
1	7	15
2	8	5
3	9	3
4	10	17
5	11	7
6	12	13
7	420	15

	group_id [PK] integer	trainer_id integer
1	7	15
2	8	5
3	9	3
4	10	17
5	11	7
6	12	13

Рисунок 4 – до/после выполнения DELETE

Задание 4

```
1 EXPLAIN ANALYZE SELECT id_sportsman, sportsman_points
2 FROM "Participation"
3 WHERE sportsman_points > 80
```

Data Output Messages Notifications

QUERY PLAN
text

1	Seq Scan on "Participation" (cost=0.00..1.63 rows=6 width=8) (actual time=0.052..0.066 rows=7 loops...
2	Filter: (sportsman_points > 80)
3	Rows Removed by Filter: 43
4	Planning time: 0.713 ms
5	Execution time: 0.472 ms

Рисунок 5 – выполнение запроса до создания индексов

```
1 create index col_a on "Participation"(id_sportsman);
2 create index col_b on "Participation"(sportsman_points);
```


Рисунок 6 - создание индексов

```

1
2 EXPLAIN ANALYZE SELECT id_sportsman, sportsman_points
3 FROM "Participation"
4 WHERE sportsman_points > 80

```

Data Output Messages Notifications

QUERY PLAN text 

1	Seq Scan on "Participation" (cost=0.00..1.63 rows=6 width=8) (actual time=0.023..0.033 rows=7 loops...
2	Filter: (sportsman_points > 80)
3	Rows Removed by Filter: 43
4	Planning time: 0.941 ms
5	Execution time: 0.063 ms

Рисунок 7 – выполнение запроса после создания индексов

```

1 CREATE INDEX bogus ON "Sportsman"(id_sportsman, birthdate);

```

Рисунок 8 - создание составного индекса

	QUERY PLAN text 
1	Seq Scan on "Sportsman" (cost=0.00..1.75 rows=6 width=8) (actual time=0.024..0.036 rows=22 loops...
2	Filter: ((birthdate > '2000-05-18'::date) AND (id_sportsman > 2))
3	Rows Removed by Filter: 28
4	Planning time: 0.406 ms
5	Execution time: 0.072 ms

Рисунок 9 - до создания индексов


	QUERY PLAN	
	text	
1	Seq Scan on "Sportsman" (cost=0.00..1.75 rows=6 width=8) (actual time=0.029..0.042 rows=22 loops...	
2	Filter: ((birthdate > '2000-05-18'::date) AND (id_sportsman > 2))	
3	Rows Removed by Filter: 28	
4	Planning time: 0.190 ms	
5	Execution time: 0.070 ms	

Рисунок 10 - после создания индексов

Запросы выполняются значительно быстрее после создания индексов. Целесообразно создавать индексы при фильтрации и сортировке. С другой стороны, редактирование данных таблицы может усилить нагрузку на систему потому, что индексированные данные тоже необходимо изменить.

Выводы:

Глубже ознакомился с программой Pgadmin. Научился делать представления и запросы на выборку данных в среде PostgreSQL, использовать индексы. На практике эти навыки позволяют провести качественную оптимизацию работы БД, что прекрасно.