

РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ
Факультет физико-математических и естественных наук
Кафедра информационных технологий

ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ № 2

дисциплина: Системы управления базами данных

Студент: Шутенко Виктория Михайловна

Группа: НФИ-бд-02-19

Студбилет 1032196705

МОСКВА

2022 г.

Задание:

Агрегация

- Вывести количество мужчин и женщин в каждой группе
- Вывести количество студентов в каждой специальности упорядоченно по убыванию

Индексы и их производительность:

- Создать таблицу index_test(id serial, some_value integer)
- Заполнить таблицу index_test случайными значениями (несколько миллионов записей)
- Для этого использовать select ... from generate_series(1, 100000) – для генерации серии записей и floor(random()*2147483647) для генерации случайного значения для some_value
- Убедиться, что выборка из index_test в некотором диапазоне some_value, например между 100000 и 101000 занимает заметное время >1с
- Построить индекс по полю some_value
- Сравнить скорость работы выборки после построения индекса

Ход работы:

В прошлой лабораторной работе я создала таблицу student, содержащую основные данные студентов.

The screenshot shows the pgAdmin interface with the 'student' table selected. The left sidebar lists various database objects like FTS Configuration, FTS Dictionaries, FTS Parsers, etc. The 'Tables (5)' section is expanded, showing 'group_stud', 'speciality', and 'student'. The 'student' table is selected and its details are shown in the main pane. The table has columns: id [PK] integer, name character varying, stip integer, birthday date, gender character varying, and group_id integer. The data pane displays 21 rows of student information. At the bottom, it says 'Total rows: 21 of 21' and 'Query complete 00:00:00.683'.

	id [PK] integer	name character varying	stip integer	birthday date	gender character varying	group_id integer
1	1350135689	Будашева Анна	0	2000-09-21	f	4
2	1351567840	Полькин Максим	0	2000-01-28	m	2
3	1352310913	Лисин Иван	2000	1999-08-22	m	2
4	1352311234	Жарова Ксения	2000	2000-09-16	f	1
5	1352311295	Яковлев Степан	5000	2000-08-17	m	4
6	1352311367	Ванюшкина Татьяна	3000	1995-10-14	f	1
7	1352314376	Король Илья	5000	1999-05-12	m	4
8	1352314818	Костина Юлия	0	1995-06-29	f	1
9	1352314845	Мартынов Владимир	3000	1992-05-27	m	2
10	1352314863	Павлова Наталья	3000	1999-08-12	f	4
11	1352314892	Толкалов Артем	5000	2000-02-17	m	3
12	1352314894	Тимофеев Эйвен	0	1996-11-24	m	1
13	1352316705	Ефимов Андрей	0	2002-11-25	m	2
14	1352316790	Александрова Анастасия	0	2000-01-25	f	3
15	1352317680	Сысоева Зоя	0	1995-03-08	f	2
16	1352654893	Бофт Светлана	2000	2000-03-13	f	2
17	1352655483	Афонин Андрей	2000	2000-01-15	m	4
18	1356451252	Липухин Игорь	3000	2000-12-30	m	1
19	1356451796	Гончарова Ирина	5000	1996-08-27	f	3
20	1356521910	Корандашева Ксения	0	1996-12-23	f	2
21	1356542180	Крылова Анастасия	0	2001-03-27	f	4

Рисунок 1. Таблица student.

Сначала я вывела всех женщин, а потом мужчин.

```
SELECT * FROM public.student  
WHERE gender='f'  
ORDER BY "id" ASC
```

```
SELECT * FROM public.student  
WHERE gender='m'  
ORDER BY "id" ASC
```

The screenshot shows the pgAdmin interface with a query editor and a results grid. The query is:

```
1 SELECT * FROM public.student  
2 WHERE gender='f'  
3 ORDER BY "id" ASC  
4
```

The results grid displays 11 rows of student data where gender is 'f'. The columns are: id, name, stip, birthday, gender, and group_id. The data includes names like Будашева Анна, Жарова Ксения, Ванюшкина Татьяна, etc., with group_ids ranging from 1 to 4.

	id [PK] integer	name character varying	stip integer	birthday date	gender character varying	group_id integer
1	1350135689	Будашева Анна	0	2000-09-21	f	4
2	1352311234	Жарова Ксения	2000	2000-09-16	f	1
3	1352311367	Ванюшкина Татьяна	3000	1995-10-14	f	1
4	1352314818	Костина Юлия	0	1995-06-29	f	1
5	1352314863	Павлова Наталья	3000	1999-08-12	f	4
6	1352316790	Александрова Анастасия	0	2000-01-25	f	3
7	1352317680	Сысоева Зоя	0	1995-03-08	f	2
8	1352654893	Бофт Светлана	2000	2000-03-13	f	2
9	1356451796	Гончарова Ирина	5000	1996-08-27	f	3
10	1356521910	Корандашева Ксения	0	1996-12-23	f	2
11	1356542180	Крылова Анастасия	0	2001-03-27	f	4

Рисунок 2. Запрос, выводящий список всех женщин.

The screenshot shows the pgAdmin interface with a query editor and a results grid. The query is:

```
1 SELECT * FROM public.student  
2 WHERE gender='m'  
3 ORDER BY "id" ASC  
4
```

The results grid displays 10 rows of student data where gender is 'm'. The columns are: id, name, stip, birthday, gender, and group_id. The data includes names like Полькин Максим, Лисин Иван, Яковлев Степан, etc., with group_ids ranging from 1 to 4.

	id [PK] integer	name character varying	stip integer	birthday date	gender character varying	group_id integer
1	1351567840	Полькин Максим	0	2000-01-28	m	2
2	1352310913	Лисин Иван	2000	1999-08-22	m	2
3	1352311295	Яковлев Степан	5000	2000-08-17	m	4
4	1352314376	Король Илья	5000	1999-05-12	m	4
5	1352314803	Тимофей Эйвен	0	1996-11-24	m	1
6	1352314845	Мартынов Владимир	3000	1992-05-27	m	2
7	1352314892	Толкалов Артем	5000	2000-02-17	m	3
8	1352316705	Ефимов Андрей	0	2002-11-25	m	2
9	1352655483	Афонин Андрей	2000	2000-01-15	m	4
10	1356451252	Липухин Игорь	3000	2000-12-30	m	1

Рисунок 3. Запрос, выводящий список всех мужчин.

Потом я посчитала общее количество мужчин и женщин:

```
SELECT
count(*)
FROM public.student
WHERE gender='f'
```

```
SELECT
count(*)
FROM public.student
WHERE gender='m'
```

The screenshot shows a database interface with a toolbar at the top and two tabs: 'Query' and 'Query History'. The 'Query' tab is selected. Below it is a code editor with the following SQL query:

```
1 SELECT
2 count(*)
3 FROM public.student
4 WHERE gender='f'
5
6
```

Below the code editor is a 'Data Output' tab with a table showing the result:

	count	bigint
1	11	

Рисунок 4. Запрос, выводящий количество всех женщин.

The screenshot shows a database interface with a toolbar at the top and two tabs: 'Query' and 'Query History'. The 'Query' tab is selected. Below it is a code editor with the following SQL query:

```
1 SELECT
2 count(*)
3 FROM public.student
4 WHERE gender='m'
5
```

Below the code editor is a 'Data Output' tab with a table showing the result:

	count	bigint
1	10	

Рисунок 5. Запрос, выводящий количество всех мужчин.

Далее я посчитала количество человек в каждой группе:

```
SELECT group_stud.name,
COUNT(student.group_id)
```

```

FROM student JOIN group_stud
ON group_stud.id = student.group_id
GROUP BY group_stud.name,student.group_id

```

The screenshot shows a PostgreSQL query editor interface. At the top, there are tabs for 'Query' (which is selected) and 'Query History'. Below the tabs is a code area containing the following SQL query:

```

1  SELECT group_stud.name,
2  COUNT(student.group_id)
3  FROM student JOIN group_stud
4  ON group_stud.id = student.group_id
5  GROUP BY group_stud.name,student.group_id
6

```

Below the code area is a toolbar with icons for file operations like new, open, save, and download. Underneath the toolbar is a table with the following data:

	name character varying	count bigint
1	npi0320	3
2	nkn0121	7
3	nbi0216	6
4	nfi0219	5

Рисунок 6. Запрос, выводящий количество студентов в каждой группе.

После того, как я разобралась, как делать запросы, то я перешла к выполнению 1 задания. Так я вывела количество женщин и мужчин в каждой группе:

```

SELECT group_stud.name,
count(*) FILTER (WHERE student.gender='f') AS f,
count(*) FILTER (WHERE student.gender='m') AS m
FROM student JOIN group_stud
ON group_stud.id = student.group_id
GROUP BY group_stud.name,student.group_id

```

The screenshot shows a PostgreSQL query editor interface. At the top, there are tabs for 'Query' (selected) and 'Query History'. Below the tabs is a code area containing the following SQL query:

```

1  SELECT group_stud.name, student.group_id,
2  COUNT(*) FILTER (WHERE student.gender='f') AS f,
3  COUNT(*) FILTER (WHERE student.gender='m') AS m
4  FROM student JOIN group_stud
5  ON group_stud.id = student.group_id
6  GROUP BY group_stud.name,student.group_id

```

Below the code area is a toolbar with icons for file operations like new, open, save, and download. Underneath the toolbar is a table with the following data:

	name character varying	group_id integer	f bigint	m bigint
1	npi0320	3	2	1
2	nkn0121	2	3	4
3	nbi0216	4	3	3
4	nfi0219	1	3	2

Рисунок 7. Запрос, выводящий количество женщин (f) и мужчин (m) в каждой группе.

Во 2 задании нужно вывести количество студентов в каждой специальности упорядочено по убыванию.

The screenshot shows two tables in a database interface:

	id [PK] integer	name character varying	level character varying	faculty character varying
1	101	Mathematics	master	Physico-Mathematical Faculty
2	102	Informatics	bachelor	Physico-Mathematical Faculty

	id [PK] integer	name character varying	first_year integer	speciality_id integer
1	1	nfi0219	2019	102
2	2	nkn0121	2021	101
3	3	npi0320	2020	101
4	4	nbi0216	2016	102

Рисунок 8. Таблицы speciality и group_stud.

Я уже почти пришла к этому решению (Рисунок 6), оставалось только распределить студентов по специальности. Тогда я подключила таблицу speciality и сделала подсчет по полю id.

```
SELECT speciality.name,
COUNT(speciality.id)
FROM student
inner JOIN group_stud ON group_stud.id = student.group_id
inner JOIN speciality ON speciality.id = group_stud.speciality_id
GROUP BY speciality.name
```

The screenshot shows the results of the query in a table:

	name character varying	count bigint
1	Informatics	11
2	Mathematics	10

Рисунок 9. Количество студентов в каждой специальности

Далее выполняла задания, посвященные пункту «Индексы и их производительность».

Создала таблицу

```
CREATE TABLE IF NOT EXISTS public.index_test
(
    id serial,
    some_value integer
)
```

The screenshot shows the pgAdmin 4 interface with a query editor window. The title bar says 'study/postgres@PostgreSQL 15.1'. The toolbar has icons for file, database, table, edit, search, and others. A dropdown menu shows 'No limit'. The tabs at the bottom are 'Query' (selected), 'Query History', 'Data Output', 'Messages' (selected), and 'Notifications'. The query text is:

```
1 CREATE TABLE IF NOT EXISTS public.index_test
2 (
3     id serial,
4     some_value integer
5 )
```

The 'Messages' tab shows the output:

```
CREATE TABLE
Query returned successfully in 43 msec.
```

Рисунок 10. Создание таблицы

Добавление полей:

```
insert into index_test(id)
select * from generate_series(1, 100000);
insert into index_test(some_value)
select floor(random()*2147483647)
```

The screenshot shows the pgAdmin 4 interface with a query editor window. The title bar says 'study/postgres@PostgreSQL 15.1'. The toolbar has icons for file, database, table, edit, search, and others. A dropdown menu shows 'No limit'. The tabs at the bottom are 'Query' (selected), 'Query History', 'Data Output', 'Messages' (selected), and 'Notifications'. The query text is:

```
1 DROP TABLE IF EXISTS public.index_test;
2
3 CREATE TABLE public.index_test
4 (
5     id serial,
6     some_value integer
7 );
8
9 insert into index_test(id)
10 select * from generate_series(1, 100000);
11 insert into index_test(some_value)
12 select floor(random()*2147483647)
```

The 'Messages' tab shows the output:

```
INSERT 0 1
Query returned successfully in 289 msec.
```

Рисунок 11. Добавление полей.

```
select some_value from index_test
```

The screenshot shows a PostgreSQL query editor interface. The main pane displays the following SQL code:

```

1 DROP TABLE IF EXISTS public.index_test
2
3 CREATE TABLE public.index_test
4 (
5     id serial,
6     some_value integer
7 );
8
9 insert into index_test(id)
10 select * from generate_series(1, 100000);
11 insert into index_test(some_value)
12 select floor(random()*2147483647);
13
14 select some_value from index_test
15 where some_value in (100000, 101000);

```

The line `14 select some_value from index_test` is highlighted with a light blue background. Below the editor is a data output pane showing a single row:

some_value	integer

A green status bar at the bottom right indicates: "Successfully run. Total query runtime: 45 msec. 0 rows affected.".

Рисунок 12. Выборка

`create index some_value_idx ON index_test(some_value);`

The screenshot shows a PostgreSQL query editor interface. The main pane displays the following SQL code:

```

5     id serial,
6     some_value integer
7 );
8
9 insert into index_test(id)
10 select * from generate_series(1, 100000);
11 insert into index_test(some_value)
12 select floor(random()*2147483647);
13
14 select some_value from index_test
15 where some_value in (100000, 101000);
16
17 create index some_value_idx ON index_test(some_value);
18
19 select some_value from index_test
20 where some_value in (100000, 101000);

```

The line `17 create index some_value_idx ON index_test(some_value);` is highlighted with a light blue background. Below the editor is a data output pane showing a single row:

some_value	integer

A green status bar at the bottom right indicates: "Successfully run. Total query runtime: 250 msec. 0 rows affected.".

Рисунок 13. Создание индекса

Полный код:

```

DROP TABLE IF EXISTS public.index_test;
CREATE TABLE public.index_test
(
    id serial,
    some_value integer
);
insert into index_test(id)
select * from generate_series(1, 100000);
insert into index_test(some_value)
select floor(random()*2147483647);
create index some_value_idx ON index_test(some_value);
select some_value from index_test
where some_value in (100000, 101000);

```