Санкт-Петербургский Национальный Исследовательский Университет

Информационных Технологий, Механики и Оптики

Факультет инфокоммуникационных технологий

Лабораторная работа №4

Выполнил:

Конопля А. К.

Проверила:

Говорова М. М.

Санкт-Петербург

2023

Введение

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

Цель

Целью данной лабораторной работы является написание SQL запросов, включая запросы на получение и модификацию данных, изучить графическое представление запросов, а также изучение и составление индексных запросов с изучением временной затраты на каждый.

Задачи

1) Задание. Создать запросы и представления на выборку данных к базе данных PostgreSQL (согласно индивидуальному заданию, часть 2 и 3).

1. Составить список всех товаров, стоимостью, не превышающей, 1200.
2. Составить список всех сотрудников, оформлявших чеки в период с апреля по начало июня 2022 года.
3. Каков был месячный доход каждого магазина, в котором оформлялись покупки в период с апреля по начало июня 2022 года.
4. Составить список товаров, хранящихся в магазинах в количестве более 10 экземпляров.
5. Посчитать количество сотрудников, состоящий на должность “post2" или "post3"', не оформлявших заказы на протяжении периода с апреля по начало июня 2022 года.
6. Определить магазин с наибольшим числом товара на складе на текущие сутки.
7. Определить самый популярный товар на промежутке с апреля по начало июня 2022 года.

2) Задание 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.

Выполнение

Задание 1

Запрос 1

1. Составить список всех товаров, стоимостью, не превышающей, 1200.

	item_id [PK] integer	item_name character varying	price double precision	manufacturer_id integer	item_size integer
1	1	item1	1100.99	1	44
2	2	item2	1200.99	1	44
3	3	item3	1300.99	1	44
4	4	item4	1400.99	1	44
5	5	рубашка1	949.99	2	46
6	6	рубашка2	1049.99	2	47
7	7	пальто1	4149.99	3	51
8	8	пальто2	4249.99	3	52
9	9	пальто3	4349.99	3	53

Содержание таблицы items

```
1  SELECT * FROM items
2  WHERE price <= 1200
3  ORDER BY price DESC;
```

Код для запроса 1

	item_id [PK] integer	item_name character varying	price double precision	manufacturer_id integer	item_size integer
1	1	item1	1100.99	1	44
2	6	рубашка2	1049.99	2	47
3	5	рубашка1	949.99	2	46

Результат запроса 1

Запрос 2.

Составить список всех сотрудников, оформлявших чеки в период с апреля по начало июня 2022 года.

```sql
SELECT employees.employee_id, employees.employee_name
FROM employees
JOIN checks ON checks.employee_id = employees.employee_id
WHERE checks.check_date BETWEEN '2022-04-01' AND '2022-05-31';
```

Код для запроса 2

	employee_id [PK] integer	employee_name character varying
1	1	Alex1
2	4	Dima1

Вывод запроса 2

check_id [PK] integer	check_date date	employee_id integer
1	2022-07-19	4
2	2022-08-11	4
3	2022-09-26	1
4	2022-04-13	4
5	2022-05-20	1
6	2022-08-02	3
7	2022-07-24	1
8	2022-03-08	2

Содержание таблицы checks

Запрос 3

1. Каков был месячный доход каждого магазина, в котором оформлялись покупки в период с апреля по начало июня 2022 года.

```sql
SELECT stores.store_id, stores.store_address,
ROUND(SUM(items.price * (check_list.item_amount - check_list.returned_amount)) / 12)

FROM stores

JOIN employees emp ON emp.store_id = stores.store_id
JOIN checks ON checks.employee_id = emp.employee_id
JOIN check_list ON checks.check_id = check_list.check_id
JOIN items ON items.item_id = check_list.item_id

WHERE checks.check_date BETWEEN '2022-01-01' AND '2022-12-31'
GROUP BY stores.store_id;
```

Код для запроса 3

store_id [PK] integer	store_address character varying	round double precision
1	Lenina street,1	1952
2	Lenina street,2	2798

Результат запроса 3

Запрос 4

1. Составить список товаров, хранящихся в магазинах в количестве более 10 экземпляров.

item_id integer	item_name character varying	item_amount integer	store_id integer
2	item2	11	1
3	item3	11	1
4	item4	19	1
1	item1	17	2
2	item2	14	2

Результат запроса 4

```sql
SELECT i_strd.item_id, items.item_name, i_strd.item_amount, i_strd.store_id
FROM item_stored i_strd
JOIN items ON items.item_id = i_strd.item_id
WHERE item_amount > 10;
```

Кода запроса 4

stored_item_id [PK] integer	item_id integer	store_id integer	item_amount integer
1	1	1	1
2	2	1	11
3	3	1	11
4	4	1	19
5	1	2	17
6	2	2	14
7	3	2	10
8	4	2	9

Таблица хранящихся предметов

Запрос 5

Посчитать количество сотрудников, состоящий на должность “post2" или "post3"', не оформлявших заказы на протяжении периода с апреля по начало июня 2022 года.

```
SELECT emp.employee_id, emp.employee_name
FROM employees emp
JOIN checks ON emp.employee_id = checks.employee_id

WHERE emp.employee_post = 'post2' OR emp.employee_post = 'post3'
AND NOT EXISTS
(
    SELECT *
    FROM checks
    WHERE emp.employee_id = checks.employee_id
    AND checks.check_date BETWEEN '2022-04-01' AND '2022-5-31'
)
```

Код для запроса 5

employee_id [PK] integer	employee_name character varying
3	Alex3
2	Alex2
5	Dima2

Результат выполнения запроса

check_id [PK] integer	check_date date	employee_id integer
1	2022-07-19	4
2	2022-08-11	4
3	2022-09-26	1
4	2022-04-13	4
5	2022-05-20	1
6	2022-08-02	3
7	2022-07-24	1
8	2022-03-08	2
9	2022-05-18	5

Данные в таблице checks

```sql
SELECT i_strd.item_id, items.item_name, i_strd.item_amount
FROM item_stored i_strd
JOIN items ON i_strd.item_id = items.item_id
GROUP BY i_strd.item_id, items.item_name, i_strd.item_amount
ORDER BY i_strd.item_amount DESC
LIMIT 1;
```

Запрос 6

Определить магазин с наибольшим числом товара на складе на текущие сутки.

```sql
SELECT stores.store_address, SUM(item_stored.item_amount) AS total_stock
FROM stores
JOIN item_stored ON stores.store_id = item_stored.store_id
GROUP BY stores.store_address
HAVING SUM(item_stored.item_amount) = (
    SELECT MAX(total_stock)
    FROM (
        SELECT SUM(item_stored.item_amount) AS total_stock
        FROM stores
        JOIN item_stored ON stores.store_id = item_stored.store_id
        GROUP BY stores.store_address
    ) AS max_stocks
)
ORDER BY total_stock DESC;
```

Код запроса 6

	store_address character varying	total_stock bigint
1	Lenina street,2	50

Результат выполнения запроса

	stored_item_id [PK] integer	item_id integer	store_id integer	item_amount integer
1	1	1	1	1
2	2	2	1	11
3	3	3	1	11
4	4	4	1	19
5	5	1	2	17
6	6	2	2	14
7	7	3	2	10
8	8	4	2	9

Содержание таблицы с хранимыми предметами

Запрос 7

Определить самый популярный товар на промежутке 2022 года.

```
SELECT items.item_id, items.item_name, SUM(check_list.item_amount) AS total_amount FROM items
JOIN check_list ON items.item_id = check_list.item_id
JOIN checks ON check_list.check_id = checks.check_id
WHERE checks.check_date BETWEEN '2022-01-01' AND '2022-12-12'
GROUP BY items.item_id, items.item_name
HAVING SUM(check_list.item_amount) = (
    SELECT MAX(total_sum)
    FROM (
        SELECT SUM(check_list.item_amount) AS total_sum
        FROM items
        JOIN check_list ON items.item_id = check_list.item_id
        JOIN checks ON check_list.check_id = checks.check_id
        WHERE checks.check_date BETWEEN '2022-01-01' AND '2022-12-12'
        GROUP BY items.item_id
    ) AS max_sums
)
```

Код запроса 7

	item_id [PK] integer	item_name character varying	total_amount bigint
1	1	item1	28

Результат выполнения запроса

	check_id integer	check_date date	item_id integer	item_amount integer
1	1	2022-07-19	1	3
2	1	2022-07-19	2	4
3	1	2022-07-19	3	3
4	1	2022-07-19	4	3
5	1	2022-07-19	5	5
6	2	2022-08-11	1	5
7	2	2022-08-11	2	5
8	2	2022-08-11	1	2
9	3	2022-09-26	1	18
10	6	2022-08-02	2	4

Таблица, отражающая данные о проданных товарах

Задание 2

Запрос 1

1. Запрос на обновления цены в зависимости от адреса расположения производителя

```
UPDATE items
SET price = ROUND((price * 1.1):numeric), 2)
WHERE manufacturer_id IN (
    SELECT manufacturer_id FROM manufacturers
    WHERE manufacturer_adress IN ('manufacturers_street_1', 'manufacturers_street_5')
);
```

Код запроса

	item_id [PK] integer	item_name character varying	price double precision	manufacturer_id integer	item_size integer
1	1	item1	1100.99	1	44
2	2	item2	1200.99	1	44
3	3	item3	1300.99	1	44
4	4	item4	1400.99	1	44
5	5	рубашка1	949.99	2	46
6	6	рубашка2	1049.99	2	47
7	7	пальто1	4149.99	3	51
8	8	пальто2	4249.99	3	52
9	9	пальто3	4349.99	3	53

Данные до изменения

	item_id [PK] integer	item_name character varying	price double precision	manufacturer_id integer	item_size integer
1	1	item1	1211.09	1	44
2	2	item2	1321.09	1	44
3	3	item3	1431.09	1	44
4	4	item4	1541.09	1	44
5	5	рубашка1	949.99	2	46
6	6	рубашка2	1049.99	2	47
7	7	пальто1	4149.99	3	51
8	8	пальто2	4249.99	3	52
9	9	пальто3	4349.99	3	53

Данные после изменения

Запрос 2

2. Запрос на добавление данных в таблицу товаров, по адресу производителя

```
INSERT INTO items (item_name, price, manufacturer_id, item_size)
VALUES (
    'suit',
    39999.99,
    (
        SELECT manufacturer_id FROM manufacturers
        WHERE manufacturer_adress = 'manufacturers_street_1'
    ),
    48
)
```

Код запроса

	item_id [PK] integer	item_name character varying	price double precision	manufacturer_id integer	item_size integer
1	1	item1	1211.09	1	44
2	2	item2	1321.09	1	44
3	3	item3	1431.09	1	44
4	4	item4	1541.09	1	44
5	5	рубашка1	949.99	2	46
6	6	рубашка2	1049.99	2	47
7	7	пальто1	4149.99	3	51
8	8	пальто2	4249.99	3	52
9	9	пальто3	4349.99	3	53

Значения до изменения

	item_id [PK] integer	item_name character varying	price double precision	manufacturer_id integer	item_size integer
1	1	item1	1211.09	1	44
2	2	item2	1321.09	1	44
3	3	item3	1431.09	1	44
4	4	item4	1541.09	1	44
5	5	рубашка1	949.99	2	46
6	6	рубашка2	1049.99	2	47
7	7	пальто1	4149.99	3	51
8	8	пальто2	4249.99	3	52
9	9	пальто3	4349.99	3	53
10	10	suit	39999.99	1	48

Значения после изменения

Запрос 3

Запрос на удаления данных о производителях, товары от которых не хранятся на складах

	manufacturer_id [PK] integer	manufacturer_name character varying	manufacturer_adress character varying	email character varying
1	1	manufacturer1	manufacturers_street_1	manufacturer1@mail.com
2	2	manufacturer2	manufacturers_street_2	manufacturer2@mail.com
3	3	manufacturer3	manufacturers_street_3	manufacturer3@mail.com
4	4	manufacturer4	manufacturers_street_4	manufacturer4@mail.com
5	5	manufacturer5	manufacturers_street_5	manufacturer5@mail.com
6	6	manufacturer6	manufacturers_street_6	manufacturer6@mail.com
7	7	manufacturer7	manufacturers_street_7	manufacturer7@mail.com
8	8	manufacturer8	manufacturers_street_8	manufacturer8@mail.com
9	9	manufacturer9	manufacturers_street_9	manufacturer9@mail.com

Данные до изменения

	manufacturer_id [PK] integer	manufacturer_name character varying	manufacturer_adress character varying	email character varying
1	1	manufacturer1	manufacturers_street_1	manufacturer1@mail.com
2	2	manufacturer2	manufacturers_street_2	manufacturer2@mail.com
3	3	manufacturer3	manufacturers_street_3	manufacturer3@mail.com

Данные после изменения

```
DELETE FROM manufacturers
WHERE manufacturer_id NOT IN (
    SELECT manufacturer_id FROM items
)
```

Код запроса

Задание 3

	employee_id [PK] integer	store_id integer	employee_name character varying	employee_surname character varying	employee_post character varying	employee_email character varying
1	1	1	Alex1	Ulyanov1	post1	Alex1Ulyanov1@mail.ru
2	2	1	Alex2	Ulyanov2	post2	Alex2Ulyanov2@mail.ru
3	3	1	Alex3	Ulyanov3	post3	Alex3Ulyanov3@mail.ru
4	4	2	Dima1	Malkhov1	post1	Dima1Malkhov1@mail.ru
5	5	2	Dima2	Malkhov2	post2	Dima2Malkhov2@mail.ru
6	6	2	Dima3	Malkhov3	post3	Dima3Malkhov3@mail.ru

Значения атрибутов в оригинальной таблице employees

```
1  CREATE VIEW view_employees AS
2  SELECT * FROM employees
```

Data Output | Сообщения | Notifications

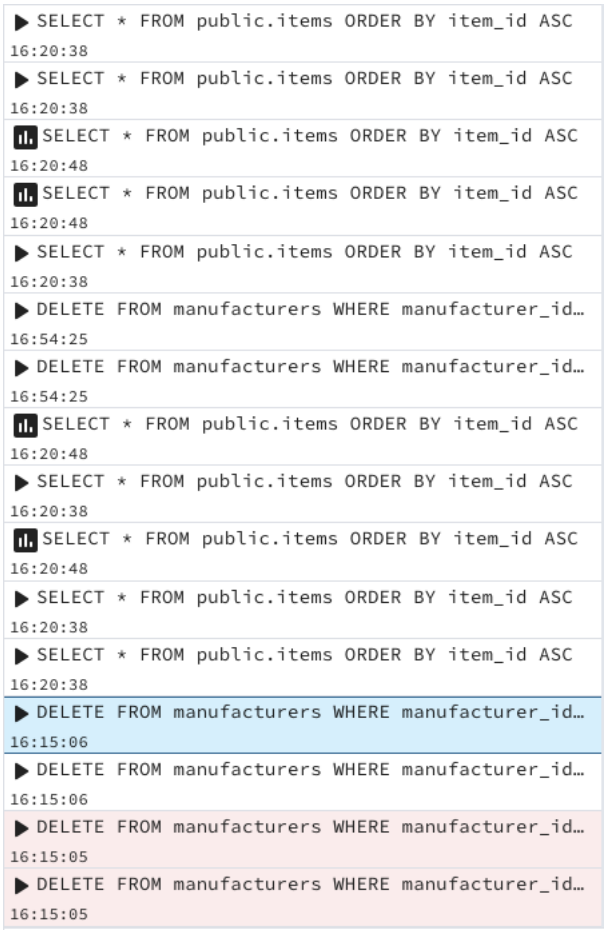
```
CREATE VIEW
```

Запрос для создания графического представления

	employee_id integer	store_id integer	employee_name character varying	employee_surname character varying	employee_post character varying	employee_email character varying
1	1	1	Alex1	Ulyanov1	post1	Alex1Ulyanov1@mail.ru
2	2	1	Alex2	Ulyanov2	post2	Alex2Ulyanov2@mail.ru
3	3	1	Alex3	Ulyanov3	post3	Alex3Ulyanov3@mail.ru
4	4	2	Dima1	Malkhov1	post1	Dima1Malkhov1@mail.ru
5	5	2	Dima2	Malkhov2	post2	Dima2Malkhov2@mail.ru
6	6	2	Dima3	Malkhov3	post3	Dima3Malkhov3@mail.ru

Значение в таблице представления view_employees

Представления позволяют обезопасить данные, так как представление может быть сделано не для всей таблицы, а только для отдельных атрибутов.



История выполнения запросов, включающая в себя не выполнившееся запросы, появившееся при тестировании составленных запросов, а также запросы на выбор данных для проверки корректности запросов.

Задание 4

Запрос 1

```
SET enable_seqscan = on;
EXPLAIN ANALYZE
SELECT item_name FROM items
WHERE item_name = 'suit' AND item_size = 48
```

Код текстового запроса

	QUERY PLAN text
1	Index Scan using idx_item_name on items (cost=0.14..8.15 rows=1 width=32) (actual time=0.024..0.027 rows=2 loop…
2	Index Cond: ((item_name)::text = 'suit'::text)
3	Filter: (item_size = 48)
4	Planning Time: 0.162 ms
5	Execution Time: 0.050 ms

План выполнения запроса без индекса

	QUERY PLAN text
1	Seq Scan on items (cost=0.00..1.17 rows=1 width=32) (actual time=0.023..0.025 rows=2 loops...
2	Filter: (((item_name)::text = 'suit'::text) AND (item_size = 48))
3	Rows Removed by Filter: 9
4	Planning Time: 0.145 ms
5	Execution Time: 0.044 ms

План выполнения с простым индексом

	QUERY PLAN text
1	Seq Scan on items (cost=0.00..1.17 rows=1 width=32) (actual time=0.014..0.016 rows=2 loops...
2	Filter: (((item_name)::text = 'suit'::text) AND (item_size = 48))
3	Rows Removed by Filter: 9
4	Planning Time: 0.158 ms
5	Execution Time: 0.033 ms

План выполнения запроса с составным индексом

```
CREATE INDEX cmplx_idx_items ON items(item_name, item_size)
```

Задание составного индекса

Запрос 2

```
EXPLAIN ANALYZE
SELECT item_name FROM items
JOIN manufacturers ON manufacturers.manufacturer_id = items.manufacturer_id
WHERE manufacturer_adress = 'manufacturers_street_1'
```

Текстовый запрос

	QUERY PLAN text
1	Hash Join (cost=1.25..19.15 rows=1 width=32) (actual time=0.050..0.056 rows=6 loops=1)
2	Hash Cond: (manufacturers.manufacturer_id = items.manufacturer_id)
3	-> Seq Scan on manufacturers (cost=0.00..17.88 rows=3 width=4) (actual time=0.017..0.019 rows=1 lo...
4	Filter: ((manufacturer_adress)::text = 'manufacturers_street_1'::text)
5	Rows Removed by Filter: 2
6	-> Hash (cost=1.11..1.11 rows=11 width=36) (actual time=0.022..0.023 rows=11 loops=1)
7	Buckets: 1024 Batches: 1 Memory Usage: 9kB
8	-> Seq Scan on items (cost=0.00..1.11 rows=11 width=36) (actual time=0.008..0.012 rows=11 loops...
9	Planning Time: 0.311 ms
10	Execution Time: 0.086 ms

План выполнения без запроса без индекса

```
CREATE INDEX idx_items_by_man_adrs ON manufacturers(manufacturer_adress);
CREATE INDEX idx_items_item_name ON items(item_name);
```

Задание простого индекса

	QUERY PLAN text
1	Hash Join (cost=1.05..2.21 rows=4 width=32) (actual time=0.039..0.042 rows=6 loops=1)
2	Hash Cond: (items.manufacturer_id = manufacturers.manufacturer_id)
3	-> Seq Scan on items (cost=0.00..1.11 rows=11 width=36) (actual time=0.009..0.010 rows=11 loops=1)
4	-> Hash (cost=1.04..1.04 rows=1 width=4) (actual time=0.011..0.011 rows=1 loops=1)
5	Buckets: 1024 Batches: 1 Memory Usage: 9kB
6	-> Seq Scan on manufacturers (cost=0.00..1.04 rows=1 width=4) (actual time=0.006..0.007 rows=1 loops...
7	Filter: ((manufacturer_adress)::text = 'manufacturers_street_1'::text)
8	Rows Removed by Filter: 2
9	Planning Time: 0.194 ms
10	Execution Time: 0.060 ms

План выполнения запроса с простым индексом

```
CREATE INDEX idx_cmplx_item_name_manufacturer_id ON items(item_name, manufacturer_id);
CREATE INDEX idx_cmplx_manufacturer_id_manufacturer_adress ON manufacturers(manufacturer_id, manufacturer_adress);
```

Задание составных индексов

	QUERY PLAN text
1	Hash Join (cost=1.05..2.21 rows=4 width=32) (actual time=0.023..0.026 rows=6 loops=1)
2	Hash Cond: (items.manufacturer_id = manufacturers.manufacturer_id)
3	-> Seq Scan on items (cost=0.00..1.11 rows=11 width=36) (actual time=0.006..0.007 rows=11 loops=1)
4	-> Hash (cost=1.04..1.04 rows=1 width=4) (actual time=0.009..0.010 rows=1 loops=1)
5	Buckets: 1024 Batches: 1 Memory Usage: 9kB
6	-> Seq Scan on manufacturers (cost=0.00..1.04 rows=1 width=4) (actual time=0.006..0.006 rows=1 loops...
7	Filter: ((manufacturer_adress)::text = 'manufacturers_street_1'::text)
8	Rows Removed by Filter: 2
9	Planning Time: 0.201 ms
10	Execution Time: 0.046 ms

План выполнения запроса с составным индексом

Вывод

В ходе выполнения данной лабораторной работы были изучены и написаны запросы к базе данных, а также изучен фукнционал аналитики pgadmin 4. Также были изучены простые и составные индексы для запросов и была оценена временная затрата на выполнение запросов.

Список источников:

1. Документация PostgreSQL [Электронный ресурс] // Официальный сайт PostgreSQL. 1996-2023. URL: https://www.postgresql.org/docs/13/index.html (дата обращения: 11.02.2023).
2. Документация pgAdmin 4 PostgreSQL [Электронный ресурс] // Официальный сайт pgAdmin. URL: https://www.pgadmin.org/docs/pgadmin4/latest/ (дата обращения: 11.02.2023)