Национальный исследовательский университет ИТМО

Лабораторная работа №2

«ЗАПРОСЫ НА ВЫБОРКУ И МОДИФИКАЦИЮ ДАННЫХ, ПРЕДСТАВЛЕНИЯ И ИНДЕКСЫ В POSTGRESQL»

По дисциплине
«Проектирование и реализация баз данных»

Выполнил:
Кривцов П.А.
Группа:
K3240
Преподаватель:
Говорова М.М.

Санкт-Петербург
2022 г

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 (ВАРИАНТ 9 – Оптовая база)

Задание 1. Создайте запросы:

- Вывести список поставщиков, которые поставляют все товары.
- Определить поставщика, который поставляет каждый из товаров по самой низкой цене.
- Вывести названия товаров, цены на которые у всех поставщиков одинаковы.
- Чему равен общий суточный доход оптового склада за прошедший день?
- Вычислить общее количество каждого вида товара, находящегося на базе.
- В какой день было вывезено минимальное количество товара?
- Сколько различных видов товара имеется на базе?

Задание 2. Создайте представления:

- Количество заказов фирм-покупателей за прошедший год;
- Доход базы за конкретный период.

Задание 3.

Составить 3 запроса на модификацию данных (INSERT, UPDATE, DELETE) с использованием подзапросов.

Задание 4.

Изучить графическое представление запросов и просмотреть историю запросов

Задание 5.

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 БАЗЫ ДАННЫХ

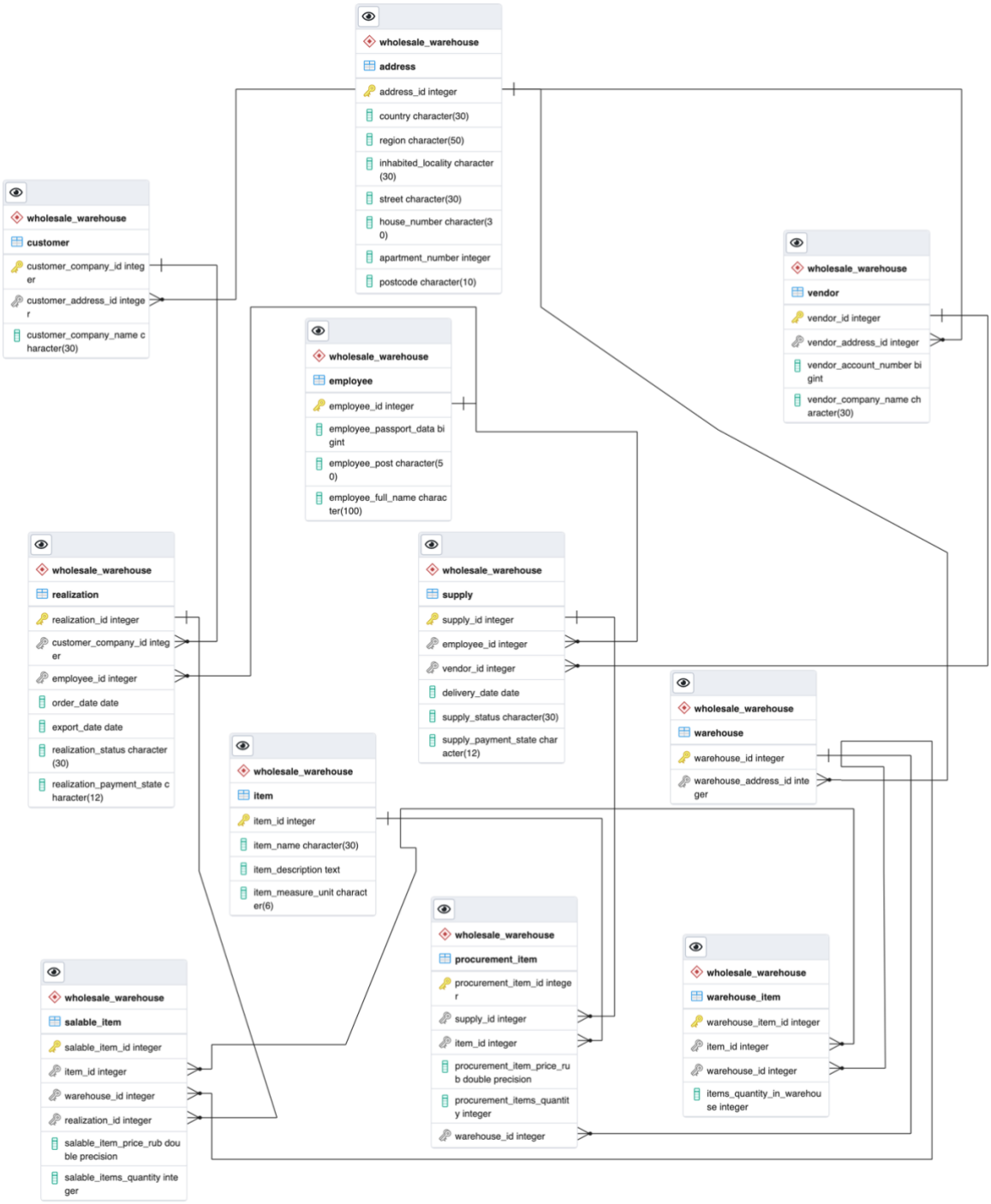

Рисунок 1 - Схема логической модели данных

ВЫПОЛНЕНИЕ

Запросы к базе данных

1. Вывести список поставщиков, которые поставляют все товары

```
SELECT s_supply.item_name, vendor.vendor_company_name
FROM
    (SELECT supply.vendor_id, p_item.item_name
    FROM (
        SELECT item.item_name, procurement_item.supply_id
        FROM wholesale_warehouse.item
        JOIN wholesale_warehouse.procurement_item
        ON item.item_id = procurement_item.item_id
    ) AS p_item
    JOIN wholesale_warehouse.supply
    ON p_item.supply_id = supply.supply_id) AS s_supply
JOIN wholesale_warehouse.vendor
ON s_supply.vendor_id = vendor.vendor_id
```

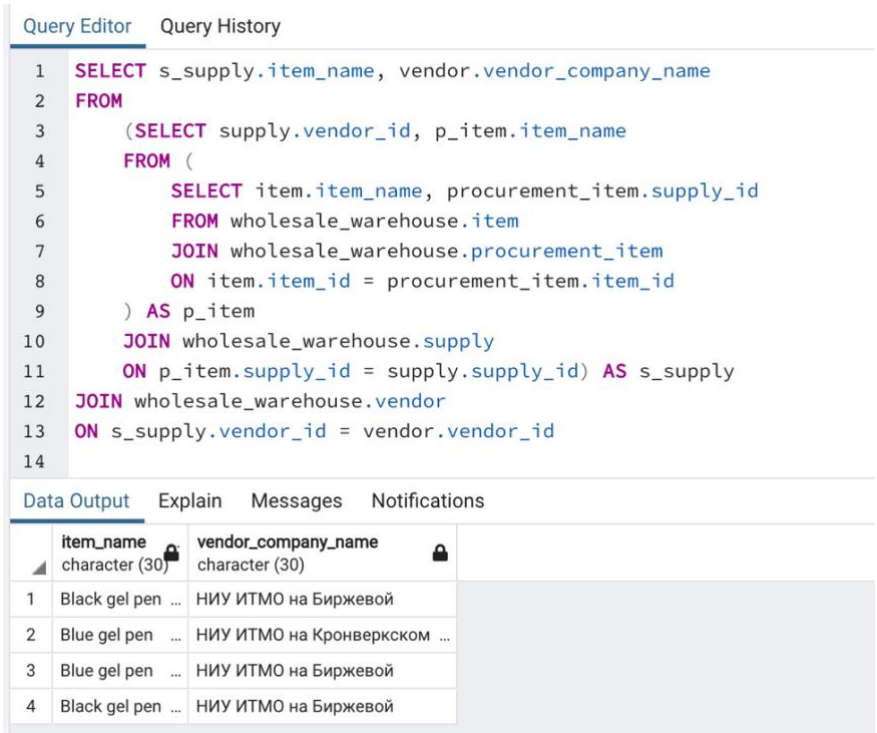

Рисунок 2

2. Определить поставщика, который поставляет каждый из товаров по самой низкой цене.

```
SELECT s_supply.item_name, s_supply.price, vendor.vendor_company_name
FROM
	(SELECT supply.vendor_id, p_item.price, p_item.item_name
	FROM (
		SELECT item.item_id, item.item_name,
procurement_item.procurement_item_price_rub as price, procurement_item.supply_id
		FROM wholesale_warehouse.item
		JOIN wholesale_warehouse.procurement_item
		ON item.item_id = procurement_item.item_id
	) AS p_item
	JOIN wholesale_warehouse.supply
	ON p_item.supply_id = supply.supply_id
	WHERE p_item.price = (
		SELECT MIN(procurement_item_price_rub)
		FROM wholesale_warehouse.procurement_item
		WHERE item_id = p_item.item_id
	)) AS s_supply
JOIN wholesale_warehouse.vendor
ON s_supply.vendor_id = vendor.vendor_id
```

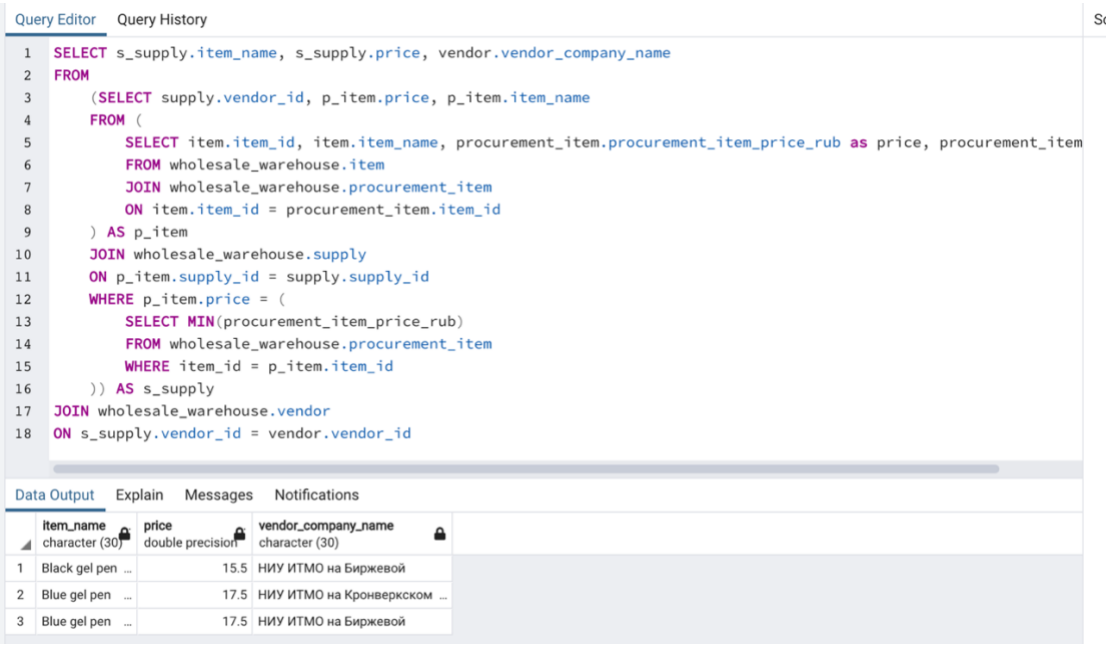

Рисунок 3

3. Вывести названия товаров, цены на которые у всех поставщиков одинаковы.

SELECT item.item_name FROM

(SELECT item_id, COUNT(DISTINCT procurement_item_price_rub) as number_of_distinct

FROM wholesale_warehouse.procurement_item

GROUP BY item_id) AS f_result,

wholesale_warehouse.item AS item

WHERE item.item_id = f_result.item_id AND f_result.number_of_distinct = 1

Рисунок 4

4. Чему равен общий суточный доход оптового склада за прошедший день?

SELECT sum(salable_item.salable_item_price_rub *

salable_item.salable_items_quantity)

FROM wholesale_warehouse.realization

JOIN wholesale_warehouse.salable_item

ON realization.realization_id = salable_item.realization_id

WHERE realization.order_date = TIMESTAMP 'yesterday' AND

realization.realization_payment_state = 'оплачено'

Рисунок 5

5. Вычислить общее количество каждого вида товара, находящегося на базе.

SELECT item.item_name, warehouse_item.items_quantity_in_warehouse

FROM wholesale_warehouse.warehouse_item

JOIN wholesale_warehouse.item

ON warehouse_item.item_id = item.item_id

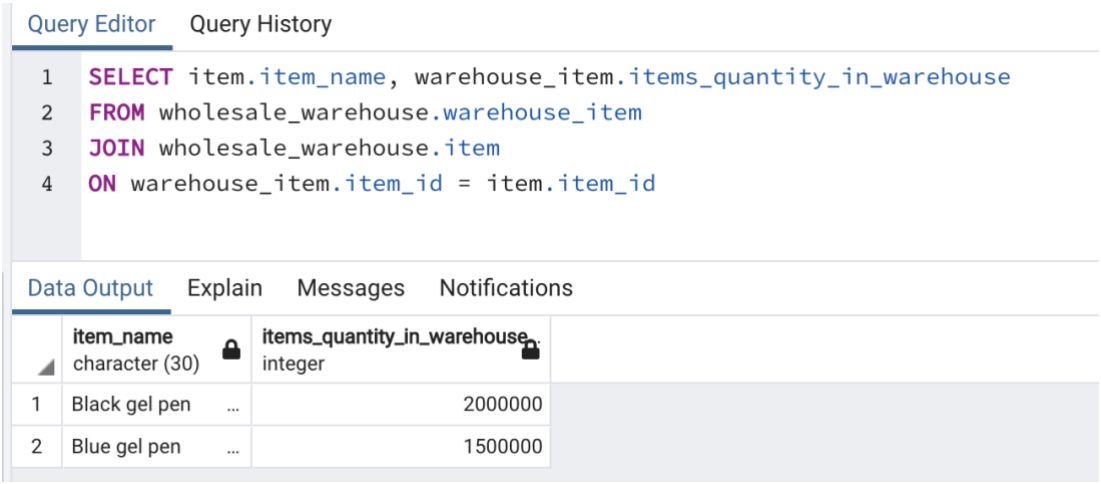

Рисунок 6

6. В какой день было вывезено минимальное количество товара?

SELECT realization.export_date, SUM(salable_item.salable_items_quantity) as f_items_sum

FROM wholesale_warehouse.realization

JOIN wholesale_warehouse.salable_item

ON realization.realization_id = salable_item.realization_id

GROUP BY realization.export_date

HAVING SUM(salable_item.salable_items_quantity) = (SELECT MIN(f_result.items_sum) FROM

(SELECT SUM(salable_item.salable_items_quantity) as items_sum
FROM wholesale_warehouse.realization
JOIN wholesale_warehouse.salable_item
ON realization.realization_id = salable_item.realization_id
GROUP BY realization.export_date) as f_result

)

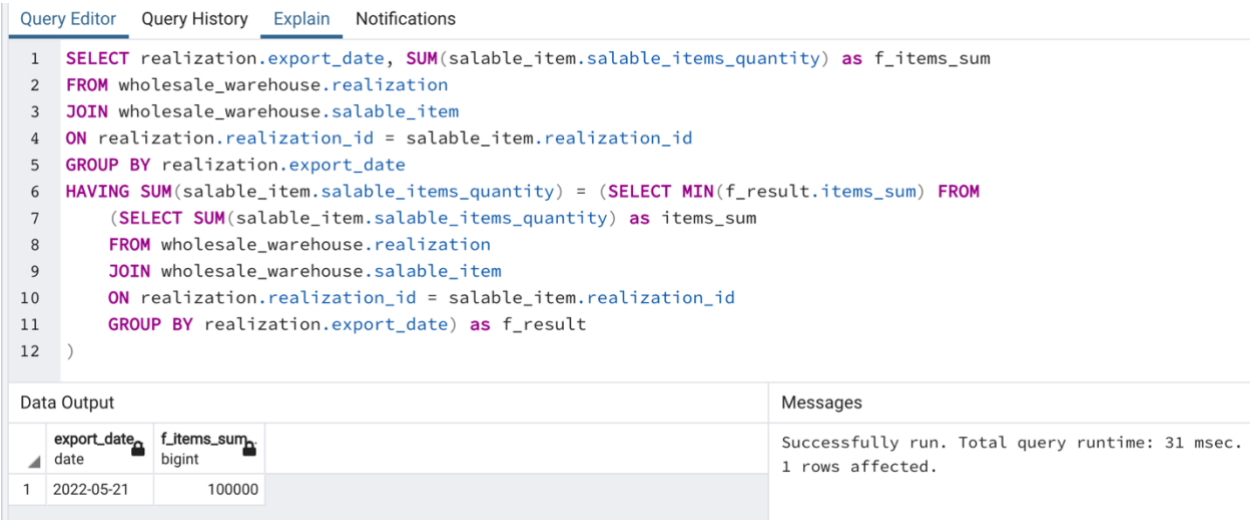

Рисунок 7

7. Сколько различных видов товара имеется на базе?

SELECT DISTINCT item.item_name
FROM wholesale_warehouse.warehouse_item
JOIN wholesale_warehouse.item
ON warehouse_item.item_id = item.item_id

Query Editor | Query History | Explain | Notifications

```
1  SELECT DISTINCT item.item_name
2  FROM wholesale_warehouse.warehouse_item
3  JOIN wholesale_warehouse.item
4  ON warehouse_item.item_id = item.item_id
```

Data Output

	item_name character (30)
1	Black gel pen
2	Blue gel pen

Рисунок 8

Представления

1. Количество заказов фирм-покупателей за прошедший год

```
CREATE VIEW ly_orders_number AS
    SELECT COUNT(realization_id)
    FROM wholesale_warehouse.realization
    WHERE order_date BETWEEN '2021-01-01' AND '2021-12-31'
```

Query Editor | Query History | Explain | Notifications

```
1  CREATE VIEW ly_orders_number AS
2      SELECT COUNT(realization_id)
3      FROM wholesale_warehouse.realization
4      WHERE order_date BETWEEN '2021-01-01' AND '2021-12-31'
```

Data Output

	count bigint
1	2

Messages

```
CREATE VIEW

Query returned successfully in 59 msec.
```

Рисунок 9

Вид:

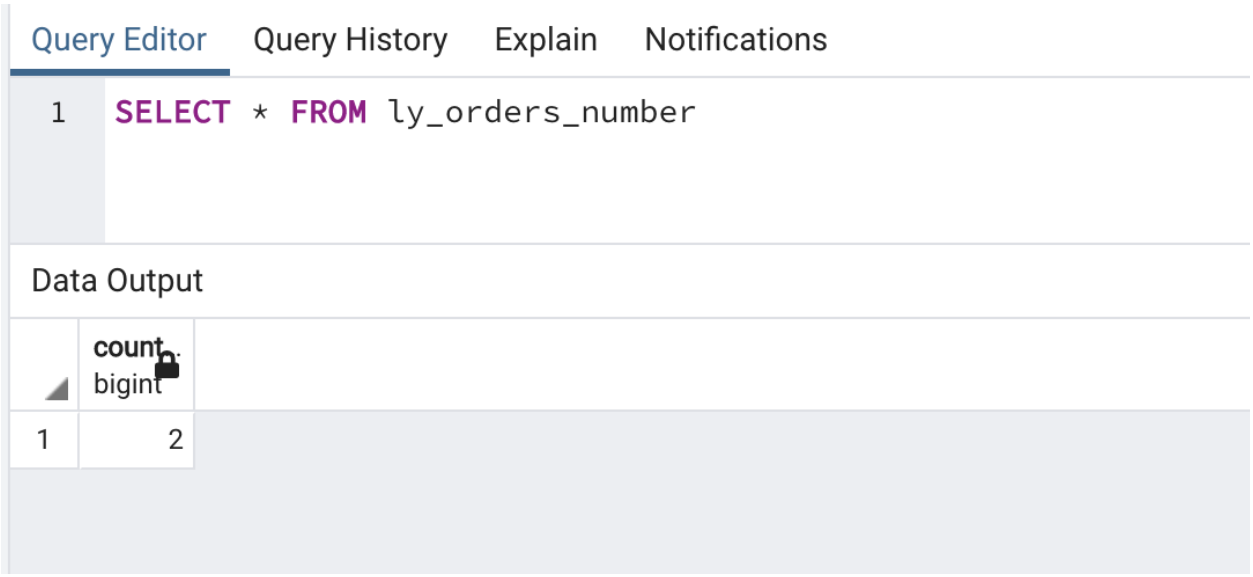

Рисунок 10

2. Доход базы за конкретный период

CREATE VIEW ty_income AS

SELECT sum(salable_item.salable_item_price_rub * salable_item.salable_items_quantity)

FROM wholesale_warehouse.realization

JOIN wholesale_warehouse.salable_item

ON realization.realization_id = salable_item.realization_id

WHERE realization.order_date BETWEEN '2022-01-01' AND '2022-12-31'

AND realization.realization_payment_state = 'оплачено'

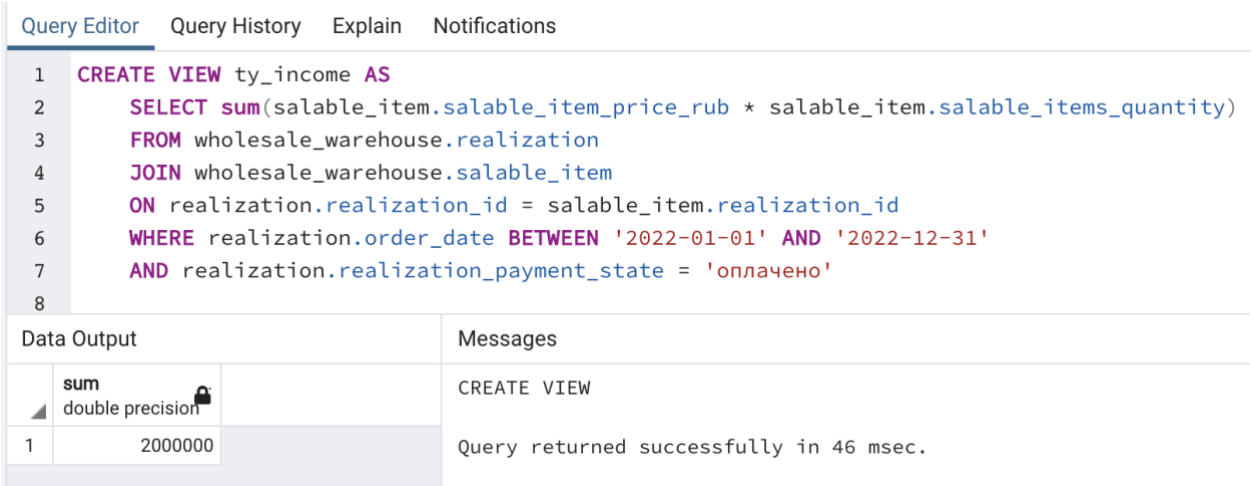

Рисунок 11

Вид:

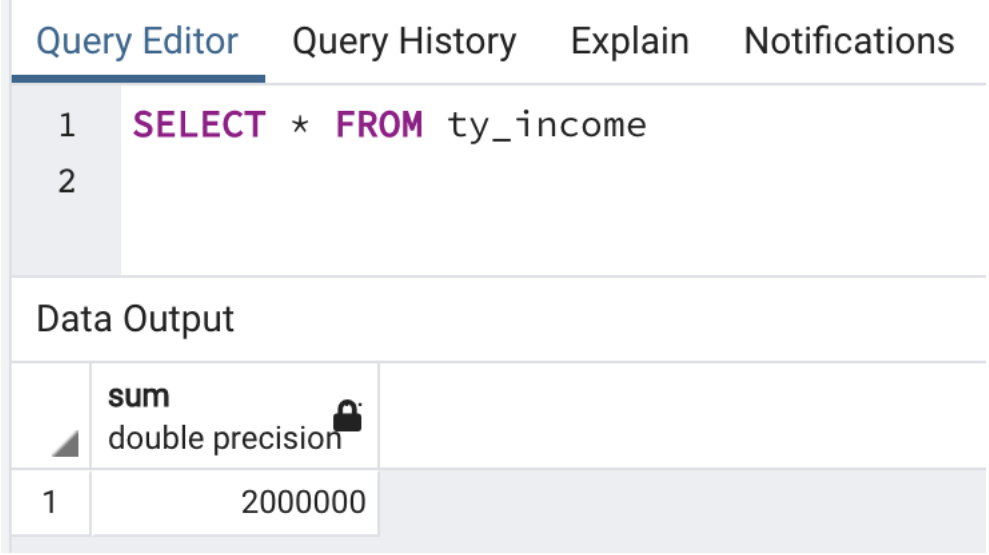

Рисунок 12

Запросы на модификацию данных

1. INSERT

Зададим информацию о продаваемом товаре, который называется “Black gel pen”. Информация об id этого товара находится в другой сущности.

```
INSERT INTO wholesale_warehouse.salable_item (
        item_id,
        warehouse_id,
        realization_id,
        salable_item_price_rub,
        salable_items_quantity
) SELECT item_id, 1, 2, 19, 25000
FROM wholesale_warehouse.item
WHERE item_name = 'Black gel pen'
```

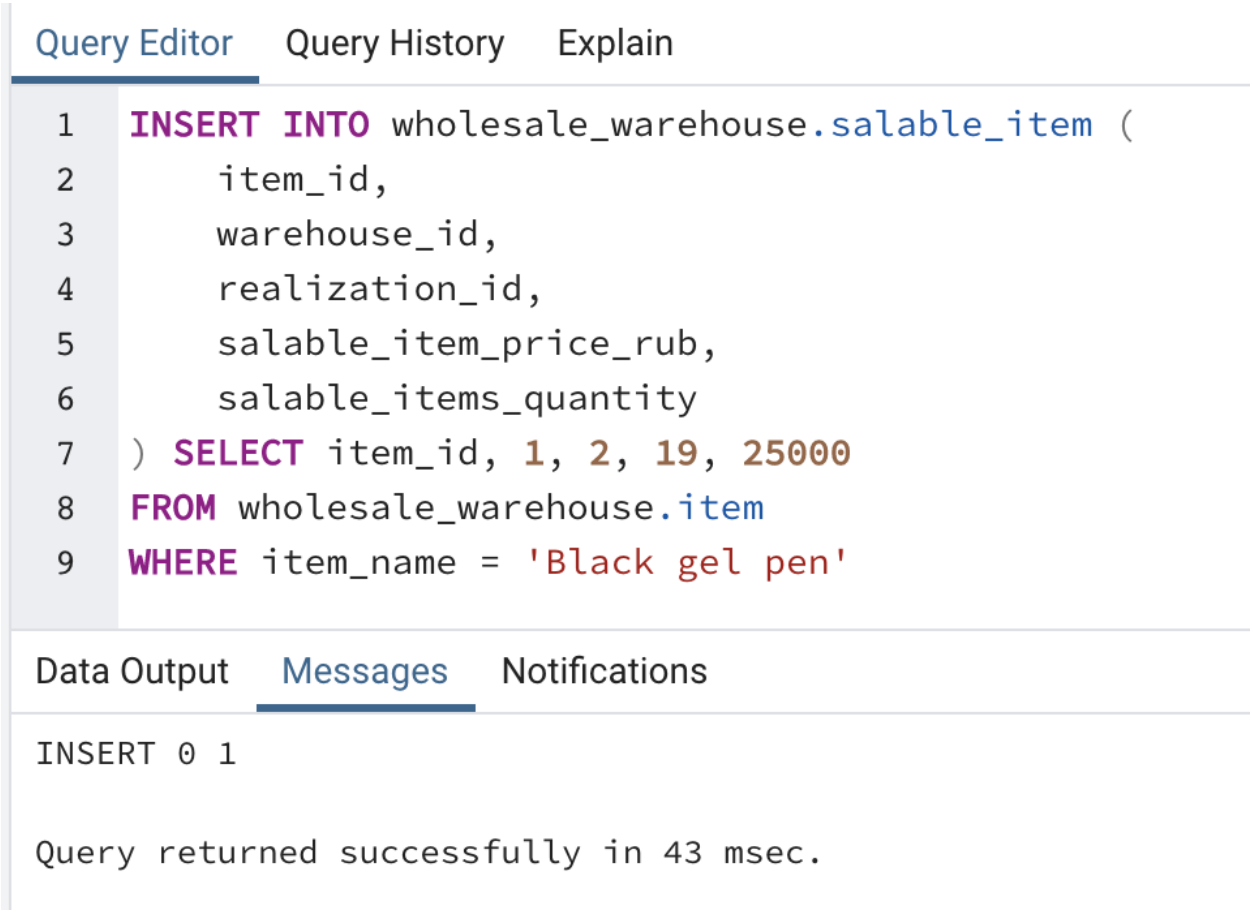

Рисунок 13

2. UPDATE

Увеличим цену продаваемого товара с id равным 3 вдвое

```
UPDATE wholesale_warehouse.salable_item
SET salable_item_price_rub = (
        SELECT salable_item_price_rub * 2
        FROM wholesale_warehouse.salable_item
        WHERE salable_item_id = 3
)
WHERE salable_item_id = 3
```

Query Editor | Query History | Explain

```
1  UPDATE wholesale_warehouse.salable_item
2  SET salable_item_price_rub = (
3      SELECT salable_item_price_rub * 2
4      FROM wholesale_warehouse.salable_item
5      WHERE salable_item_id = 3
6  )
7  WHERE salable_item_id = 3
```

Data Output | Messages | Notifications

```
UPDATE 1

Query returned successfully in 34 msec.
```

Рисунок 14

3. DELETE

Удалить товары из списка продажи, если доход с их общей продажи меньше 3000000.

```
DELETE FROM wholesale_warehouse.salable_item
WHERE realization_id IN
    (SELECT f_result.realization_id
    FROM (
        SELECT salable_item.realization_id,
            sum(salable_item.salable_item_price_rub *
salable_item.salable_items_quantity) AS profit
        FROM wholesale_warehouse.salable_item
        GROUP BY salable_item.realization_id) AS f_result
    WHERE f_result.profit < 3000000)
```

Рисунок 15

Создание индексов

1. Первый запрос:

```
SELECT realization.export_date, SUM(salable_item.salable_items_quantity) as
f_items_sum
FROM wholesale_warehouse.realization
JOIN wholesale_warehouse.salable_item
ON realization.realization_id = salable_item.realization_id
GROUP BY realization.export_date
HAVING SUM(salable_item.salable_items_quantity) = (SELECT
MIN(f_result.items_sum) FROM
        (SELECT SUM(salable_item.salable_items_quantity) as items_sum
        FROM wholesale_warehouse.realization
        JOIN wholesale_warehouse.salable_item
        ON realization.realization_id = salable_item.realization_id
        GROUP BY realization.export_date) as f_result
)
```

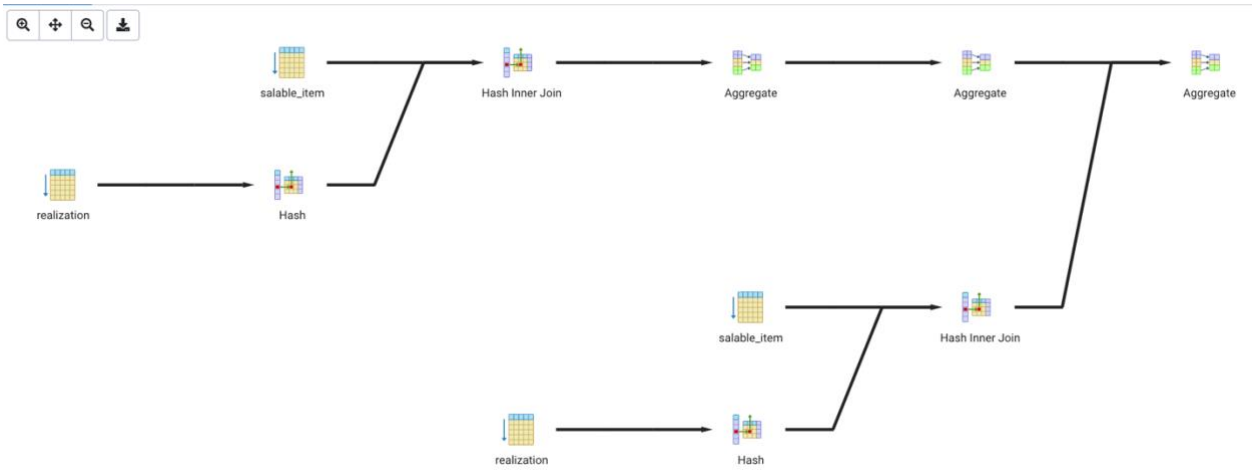

Рисунок 16 - Графический вид второго запроса без индексов

#	Node	Timings		Rows	
		Exclusive	Inclusive	Actual	Loops
1.	→ Aggregate (actual=0.068..0.07 rows=1 loops=1) Filter: (sum(salable_item.salable_items_quantity) = $0) Rows Removed by Filter: 0 Buckets: Batches: Memory Usage: 24 kB	0.009 ms	0.07 ms	1	1
2.	→ Aggregate (actual=0.024..0.026 rows=1 loops=1)	0.003 ms	0.026 ms	1	1
3.	→ Aggregate (actual=0.022..0.023 rows=1 loops=1) Buckets: Batches: Memory Usage: 24 kB	0.003 ms	0.023 ms	1	1
4.	→ Hash Inner Join (actual=0.018..0.02 rows=3 loops=1) Hash Cond: (salable_item_1.realization_id = realization_1.realization_id)	0.012 ms	0.02 ms	3	1
5.	→ Seq Scan on salable_item as salable_item_1 (actual=0.00...	0.003 ms	0.003 ms	3	1
6.	→ Hash (actual=0.004..0.005 rows=4 loops=1) Buckets: 1024 Batches: 1 Memory Usage: 9 kB	0.002 ms	0.005 ms	4	1
7.	→ Seq Scan on realization as realization_1 (actual=0.00...	0.003 ms	0.003 ms	4	1
8.	→ Hash Inner Join (actual=0.033..0.036 rows=3 loops=1) Hash Cond: (salable_item.realization_id = realization.realization_id)	0.014 ms	0.036 ms	3	1
9.	→ Seq Scan on salable_item as salable_item (actual=0.011..0.012 row...	0.012 ms	0.012 ms	3	1
10.	→ Hash (actual=0.01..0.01 rows=4 loops=1) Buckets: 1024 Batches: 1 Memory Usage: 9 kB	0.003 ms	0.01 ms	4	1
11.	→ Seq Scan on realization as realization (actual=0.005..0.007 row...	0.007 ms	0.007 ms	4	1

Рисунок 17 – Аналитика первого запроса без индексов

Итого, запрос выполнился за 67 миллисекунд

Простой индекс:

CREATE INDEX idx_exportdate on wholesale_warehouse."realization"(export_date)

```
1  CREATE INDEX idx_exportdate on wholesale_warehouse."realization"(export_date)
```

Data Output | Messages | Notifications

```
CREATE INDEX

Query returned successfully in 43 msec.
```

Рисунок 18 – Создание простого индекса

Теперь запрос выполнился за 34 миллисекунды:

Graphical | Analysis | Statistics

#	Node	Timings		Rows	
		Exclusive	Inclusive	Actual	Loops
1.	→ Aggregate (actual=0.041..0.043 rows=1 loops=1) Filter: (sum(salable_item.salable_items_quantity) = $0) Rows Removed by Filter: 0 Buckets: Batches: Memory Usage: 24 kB	0.006 ms	0.043 ms	1	1
2.	→ Aggregate (actual=0.013..0.014 rows=1 loops=1)	0.002 ms	0.014 ms	1	1
3.	→ Aggregate (actual=0.011..0.012 rows=1 loops=1) Buckets: Batches: Memory Usage: 24 kB	0.004 ms	0.012 ms	1	1
4.	→ Hash Inner Join (actual=0.008..0.009 rows=3 loops=1) Hash Cond: (salable_item_1.realization_id = realization_1.realization_id)	0.004 ms	0.009 ms	3	1
5.	→ Seq Scan on salable_item as salable_item_1 (actual=0.00...	0.002 ms	0.002 ms	3	1
6.	→ Hash (actual=0.003..0.003 rows=4 loops=1) Buckets: 1024 Batches: 1 Memory Usage: 9 kB	0.001 ms	0.003 ms	4	1
7.	→ Seq Scan on realization as realization_1 (actual=0.00...	0.002 ms	0.002 ms	4	1
8.	→ Hash Inner Join (actual=0.021..0.023 rows=3 loops=1) Hash Cond: (salable_item.realization_id = realization.realization_id)	0.007 ms	0.023 ms	3	1
9.	→ Seq Scan on salable_item as salable_item (actual=0.009..0.009 row...	0.009 ms	0.009 ms	3	1
10.	→ Hash (actual=0.007..0.007 rows=4 loops=1) Buckets: 1024 Batches: 1 Memory Usage: 9 kB	0.002 ms	0.007 ms	4	1
11.	→ Seq Scan on realization as realization (actual=0.004..0.005 row...	0.005 ms	0.005 ms	4	1

Рисунок 19 – Аналитика запроса с простым индексом

2. Второй запрос:

```
SELECT MIN(realization.export_date - realization.order_date)
FROM wholesale_warehouse.realization
JOIN wholesale_warehouse.customer
ON realization.customer_company_id = customer.customer_company_id
WHERE customer.customer_company_name = 'Oxford University'
```

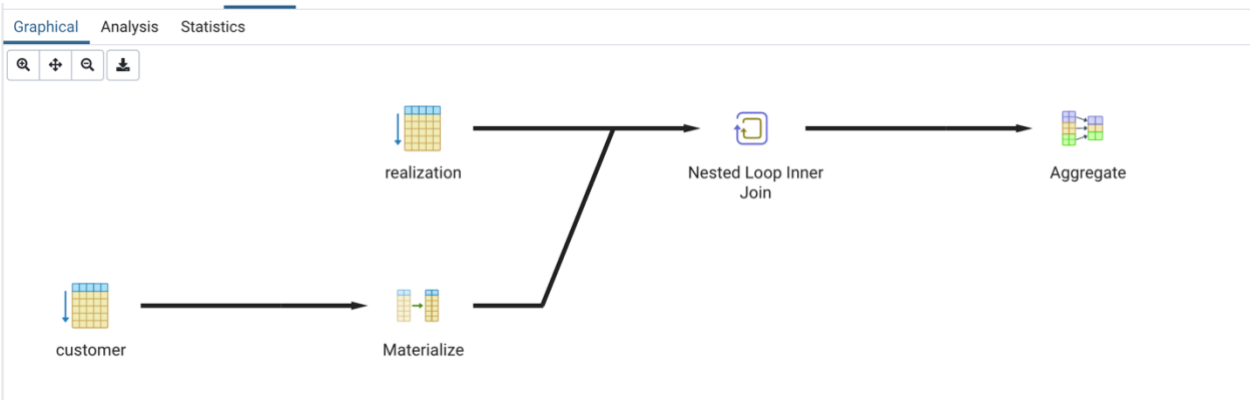

Рисунок 20 – Графический вид второго запроса без индексов

Graphical | Analysis | Statistics

#	Node	Timings		Rows	
		Exclusive	Inclusive	Actual	Loops
1.	→ Aggregate (actual=0.049..0.051 rows=1 loops=1)	0.007 ms	0.051 ms	1	1
2.	→ Nested Loop Inner Join (actual=0.038..0.044 rows=2 loops=1) Join Filter: (realization.customer_company_id = customer.customer_company_id)	0.023 ms	0.044 ms	2	1
3.	→ Seq Scan on realization as realization (actual=0.015..0.016 rows=4 loo...	0.016 ms	0.016 ms	4	1
4.	→ Materialize (actual=0.005..0.005 rows=1 loops=4)	0.002 ms	0.005 ms	1	4
5.	→ Seq Scan on customer as customer (actual=0.01..0.011 rows=1 l... Filter: (customer_company_name = 'Oxford University'::bpchar) Rows Removed by Filter: 1	0.003 ms	0.003 ms	1	1

Рисунок 21 – Аналитика второго запроса без индексов

Итого запрос выполнен за 47 миллисекунд

Составной индекс:

CREATE INDEX idx_wait_time

ON wholesale_warehouse."realization"(export_date, order_date)

```
1  CREATE INDEX idx_wait_time on wholesale_warehouse."realization"(export_date, order_date)
```

Data Output | Messages | Notifications

```
CREATE INDEX

Query returned successfully in 29 msec.
```

Рисунок 22 – Создание составного индекса

Теперь время запроса составляет 33 миллисекунды:

Graphical | Analysis | Statistics

#	Node	Timings		Rows	
		Exclusive	Inclusive	Actual	Loops
1.	→ Aggregate (actual=0.047..0.048 rows=1 loops=1)	0.006 ms	0.048 ms	1	1
2.	→ Nested Loop Inner Join (actual=0.04..0.043 rows=2 loops=1) Join Filter: (realization.customer_company_id = customer.customer_company_id)	0.017 ms	0.043 ms	2	1
3.	→ Seq Scan on realization as realization (actual=0.02..0.021 rows=4 loop...	0.021 ms	0.021 ms	4	1
4.	→ Materialize (actual=0.004..0.005 rows=1 loops=4)	0.003 ms	0.005 ms	1	4
5.	→ Seq Scan on customer as customer (actual=0.007..0.008 rows=1 ... Filter: (customer_company_name = 'Oxford University'::bpchar) Rows Removed by Filter: 1	0.002 ms	0.002 ms	1	1

Рисунок 23 – Аналитика запроса с составным индексом

Итого:

Время первого запроса после добавления простого индекса сократилось почти в 2 раза, время второго запроса после добавления составного индекса сократилось на 30%.

Удаление индексов:

DROP INDEX idx_exportdate, idx_wait_time

ВЫВОДЫ

Созданы запросы на выборку данных к базе данных PostgreSQL, составлены запросы на модификацию данных (INSERT, UPDATE, DELETE) с использованием подзапросов. Созданы простой и составной индексы для двух запросов, которые, как показало сравнение аналитических сводок по запросам, ускорили выполнение.