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

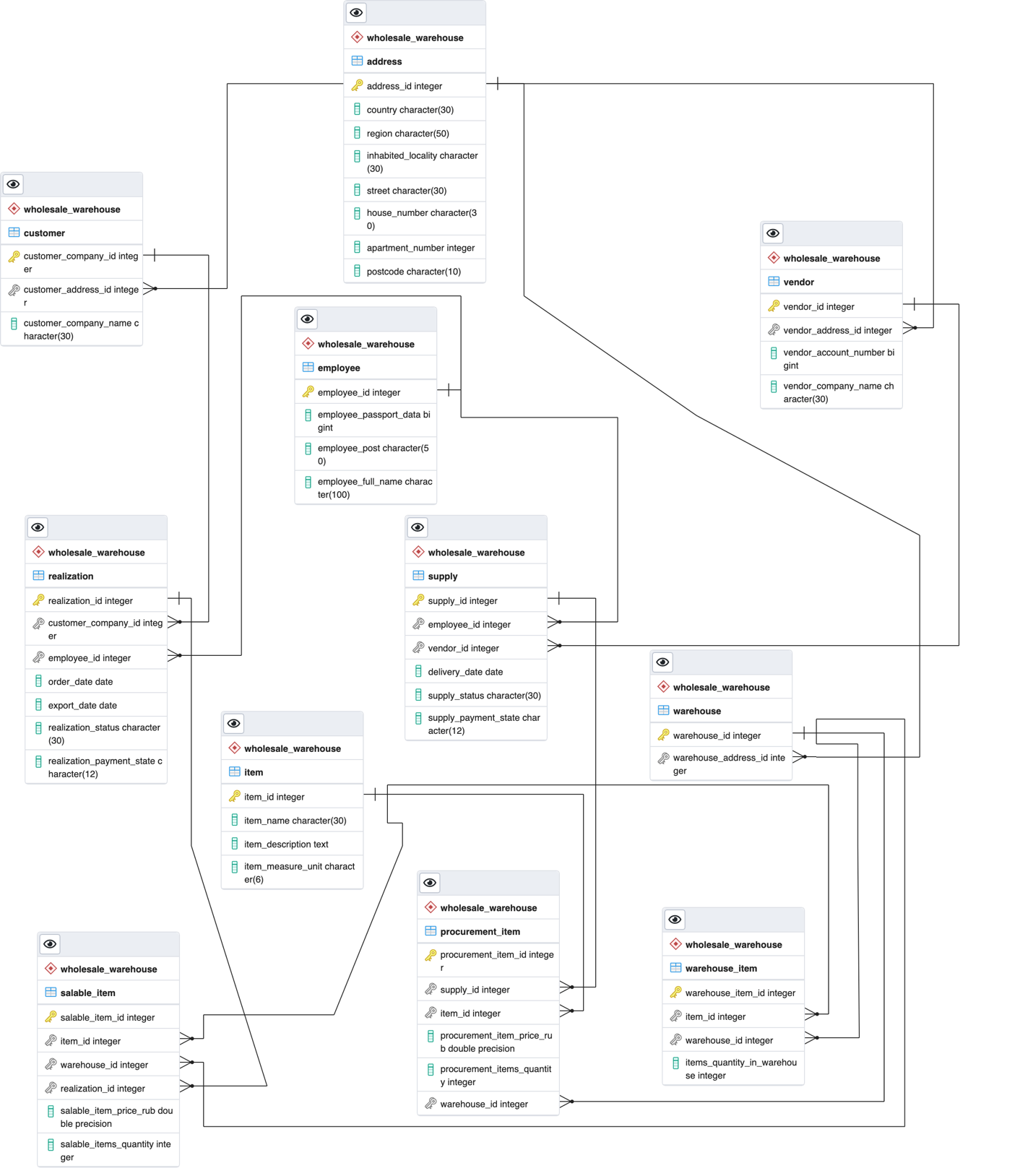


Рисунок 1 - Схема логической модели данных

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

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 2

1. 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

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 3

1. 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

Изображение выглядит как текст

Автоматически созданное описание

*Рисунок 4*

1. 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'

Изображение выглядит как текст

Автоматически созданное описание

*Рисунок 5*

1. 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

Изображение выглядит как текст

Автоматически созданное описание

*Рисунок 6*

1. 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

)

Изображение выглядит как текст

Автоматически созданное описание

*Рисунок 7*

1. 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

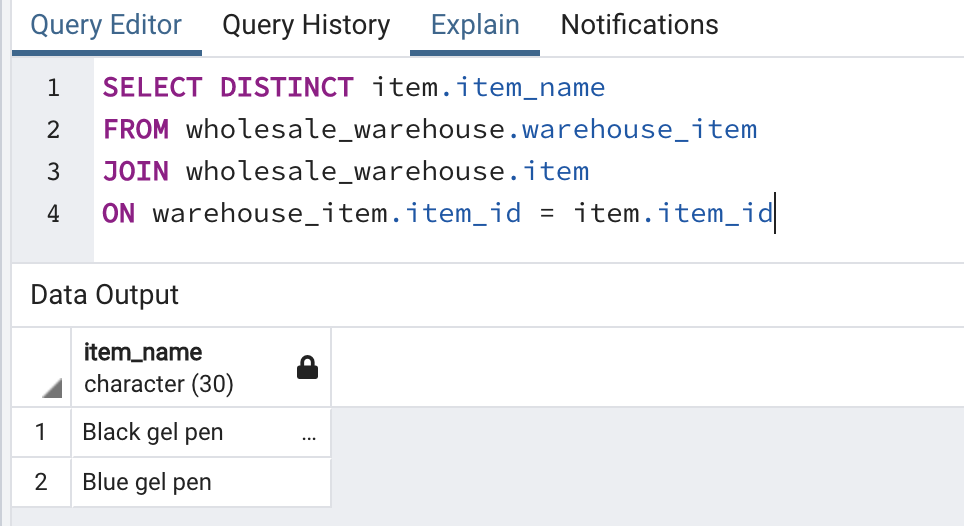


Рисунок 8

**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'

Изображение выглядит как текст

Автоматически созданное описание

*Рисунок 9*

Вид:

Изображение выглядит как текст

Автоматически созданное описание

*Рисунок 10*

1. 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'

Изображение выглядит как текст

Автоматически созданное описание

*Рисунок 11*

Вид:

Изображение выглядит как текст

Автоматически созданное описание

*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'

Изображение выглядит как текст

Автоматически созданное описание

*Рисунок 13*

1. 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

Изображение выглядит как текст

Автоматически созданное описание

*Рисунок 14*

1. 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)

Изображение выглядит как текст

Автоматически созданное описание

*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

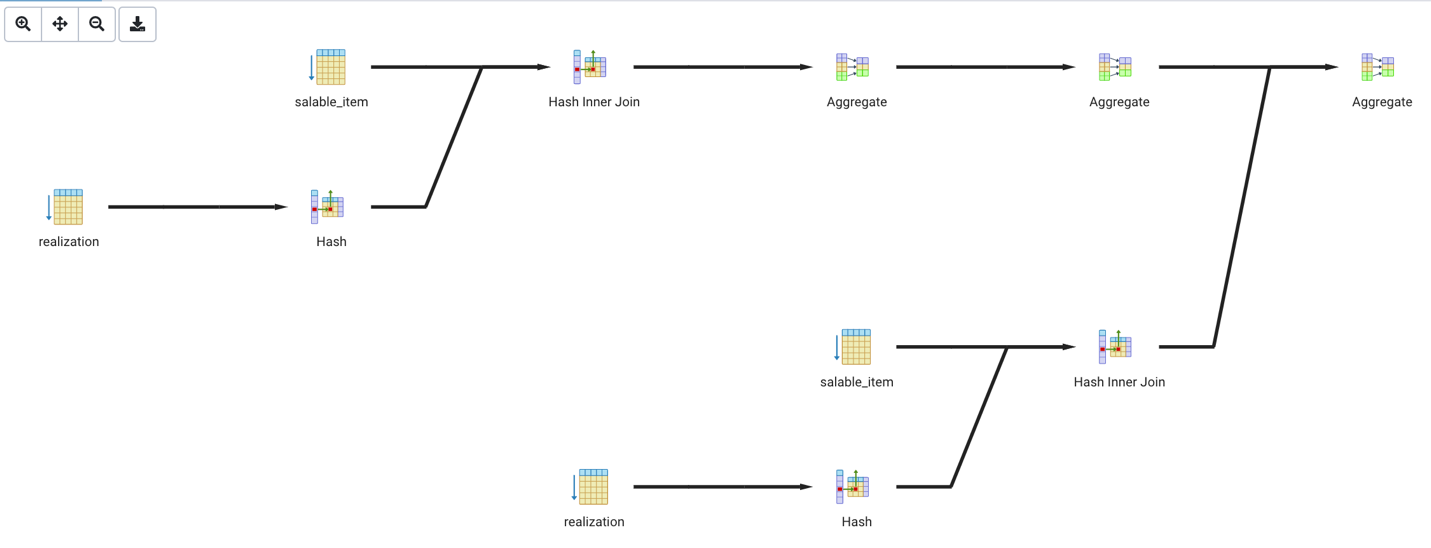
FROM wholesale\_warehouse.realization

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*

Изображение выглядит как стол

Автоматически созданное описание

*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)

Изображение выглядит как текст

Автоматически созданное описание

*Рисунок 18 – Создание простого индекса*

Теперь запрос выполнился за 34 миллисекунды:

Изображение выглядит как стол

Автоматически созданное описание

*Рисунок 19 – Аналитика запроса с простым индексом*

1. Второй запрос:

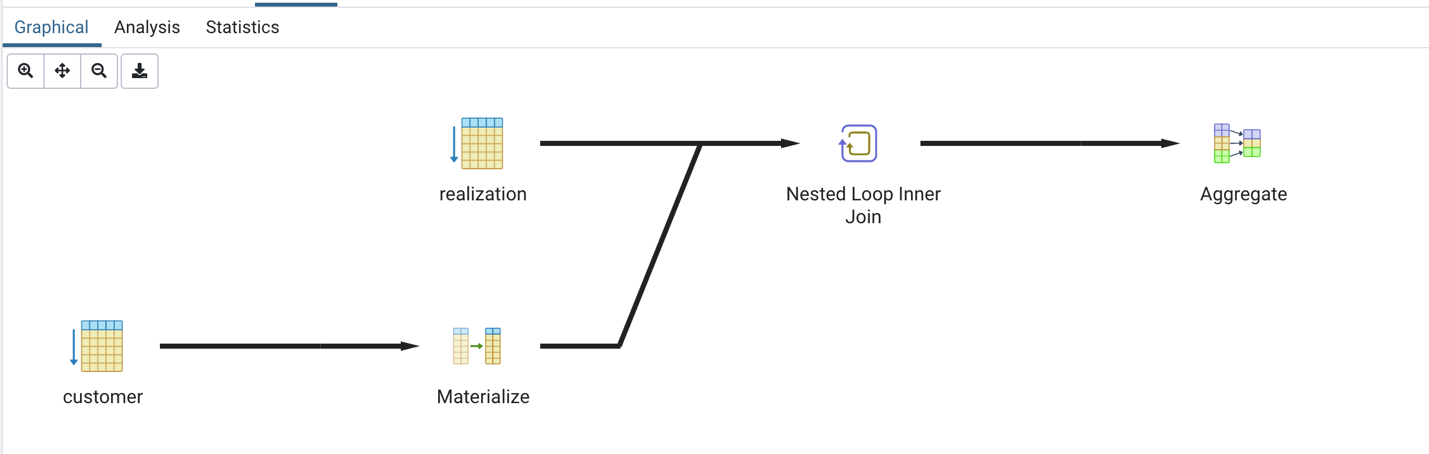
SELECT MIN(realization.export\_date - realization.order\_date)

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*

Изображение выглядит как стол

Автоматически созданное описание

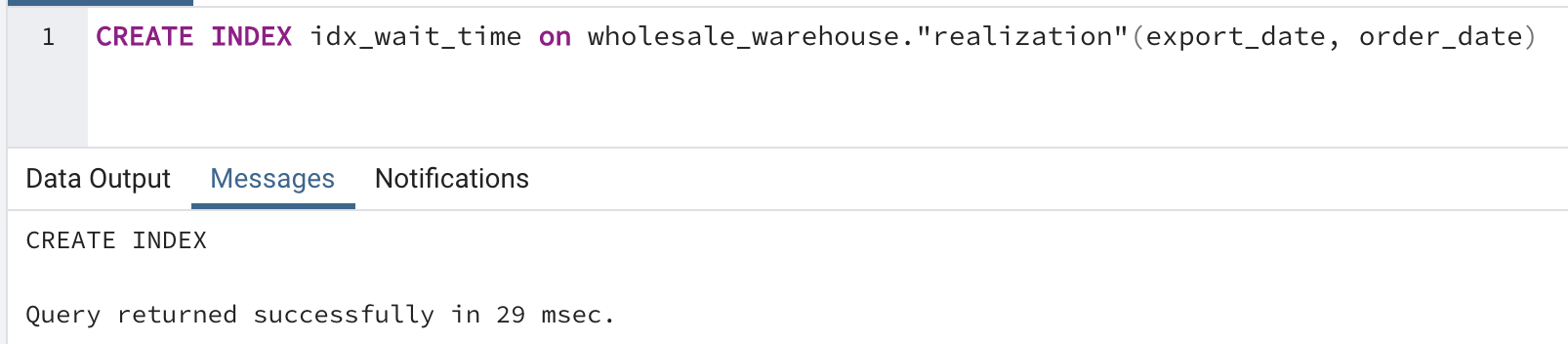
*Рисунок 21 – Аналитика второго запроса без индексов*

Итого запрос выполнен за 47 миллисекунд

**Составной индекс:**

CREATE INDEX idx\_wait\_time

ON wholesale\_warehouse."realization"(export\_date, order\_date)



*Рисунок 22 – Создание составного индекса*

Теперь время запроса составляет 33 миллисекунды:

*Изображение выглядит как стол

Автоматически созданное описание*

*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.