**ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ**

**ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ**

**"ДОНЕЦКИЙ НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ"**

Лабораторная работа № 3

Тема: «Практическое применение документоориентированных баз данных»

Проверил: Выполнил:

Ногтев Е.А. ст. гр. ПИ-18б

Щедрин С.В. Моргунов А.Г.

\_\_\_\_.\_\_\_\_.2021г.

Донецк – 2021

**Задание**

Для предметной области, описанной в заданном варианте для курсовой работы, выполнить все необходимые требования, приведенные ниже.

**Требования** **к выполняемой работе**

* Необходимо продумать и написать пример структуры JSON исходя из задания;
* Создать таблицу для хранения данных формата JSON;
* Заполнить таблицу большим количеством псевдо-данными (не менее 100000). Глубина вложенности JSON данных не менее 4;
* Выборка из БД:
  + Создать запросы, с выводом конкретных полей и их значений, а также объектов или массивов JSON документа, с одним или несколькими условиями (не менее 3 штук);
  + Выборки (запросы) с использованием jsonpath (не менее 3 штук);
  + Выборки (запросы) с использованием функций обработки JSON (не менее 3 штук)
  + Обновление/добавление/удаление данных в JSON документе (не менее 6 штук);
  + Удаление (не менее 2 штук);
* Создать индекс (не менее 3 штук, 1 по полю в котором хранится JSON документ, и не менее 2 по определённым элементам ключей);
* Провести анализ созданных запросов (с помощью EXPLAIN и ANALYZE);
* Исходя из анализа созданных запросов оптимизировать их так чтобы они пользовались созданными индексами и при этом рассчитанная стоимость запуска и общая стоимость каждого узла плана не увеличилась (не менее 5 запросов оптимизировать).

**Структура JSON:**



Рисунок 1 – Структура JSON

**Простые запросы:**

Запрос с информацией о клиенте с именем “Client 50”:

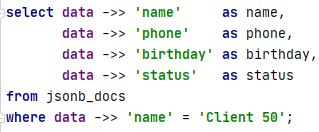


Рисунок 2 – Запрос



Рисунок 3 – Результат запроса

Запрос с информацией о клиентах, которые имеют 1 сданный товар и дату рождения меньше чем “01-01-2000”:

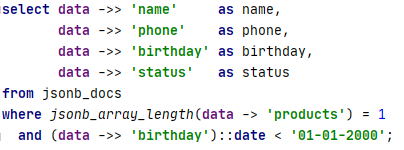


Рисунок 4 - Запрос

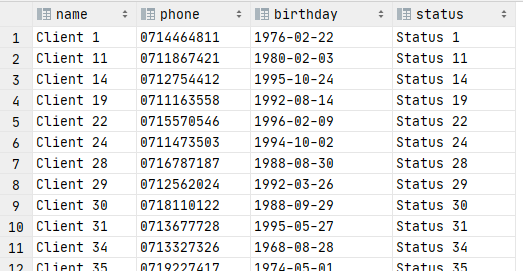


Рисунок 5 – Результат запроса

Запрос информации о товарах, которые были сданы клиентом с именем “Client 4”:

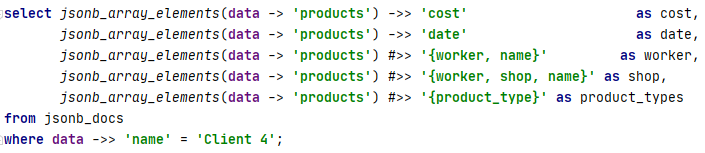


Рисунок 6 – Запрос

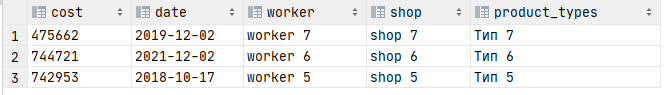


Рисунок 7 – Результат запроса

**Запросы с использованием jsonpath:**

Запрос, который выводит имя клиента и максимальную стоимость его товара, у клиентов, которые имеют товар, стоимостью больше 40000.

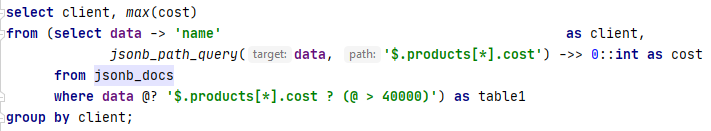


Рисунок 8 – Запрос

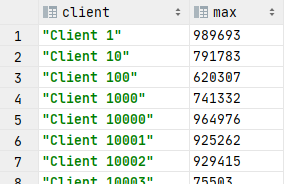


Рисунок 9 – Результат запроса

Запрос, показывающий имя и телефон клиента, а также количество продуктов, которые сдал этот клиент. Для клиентов, которые сдали как минимум 2 товара.

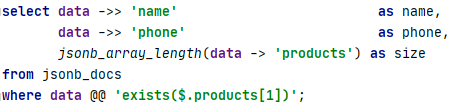


Рисунок 10 – Запрос

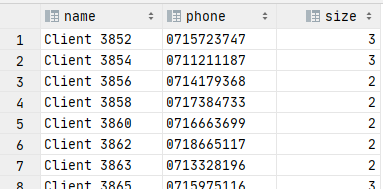


Рисунок 11 – Результат запроса

Запрос, который показывает название фирмы, имя и возраст владельца фирмы, для фирм, владельцы которых старше 40 лет.

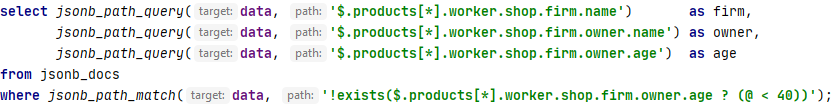


Рисунок 12 – Запрос



Рисунок 13 – Результат запроса

**Запросы с использованием функций обработки JSON:**

Запрос, который выводит имя работника и название города, для товаров клиента с именем «Client 16», которые находятся в магазине «shop 17».

**select name**, **shop** ->> **'city' as** city  
**from** *jsonb\_populate\_record*(**null**::*record*,  
 (**select** *jsonb\_path\_query*(**data**, **'$.products[\*].worker ? (@.shop.name == "shop 17")'**)  
 **from** jsonb\_docs  
 **where data** ->> **'name'** = **'Client 16'**)) **as** (**name text**, **shop jsonb**);

Рисунок 14 – Запрос



Рисунок 15 – Результат запроса

Запрос, который выводит стоимость, дату получения, тип продукта, имя работника, который принял продукт. Для продуктов, которые были сданы клиентом «Client 14».

**select cost**, **date**, **product\_type**, **worker** ->> **'name' as** worker  
**from** *jsonb\_populate\_recordset*(**null**::*record*, (**select data** -> **'products'  
 from** jsonb\_docs  
 **where data** ->> **'name'** = **'Client 14'**)) **as** (**cost numeric**, **date text**, **product\_type text**, **worker jsonb**);

Рисунок 16 - Запрос

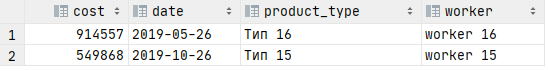


Рисунок 17 – Результат запроса

Запрос, который выводит информацию о фирме, для фирмы с названием «firm 13218», в которую были сданы товары клиентом «Client 13216».

**select** *jsonb\_each\_text*(*jsonb\_path\_query*(**data**, **'$.products[\*].worker.shop.firm ? (@.name == "firm 13218")'**))  
**from** jsonb\_docs  
**where data** ->> **'name'** = **'Client 13216'  
 and** *jsonb\_path\_exists*(**data**, **'$.products[\*].worker.shop.firm.name ? (@ == "firm 13218")'**);

Рисунок 18 – Запрос

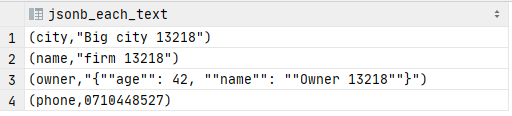


Рисунок 19 – Результат запроса

**Обновление/добавление данных:**



Рисунок 20 – Обновление данных и результат обновления



Рисунок 21 – Добавление данных

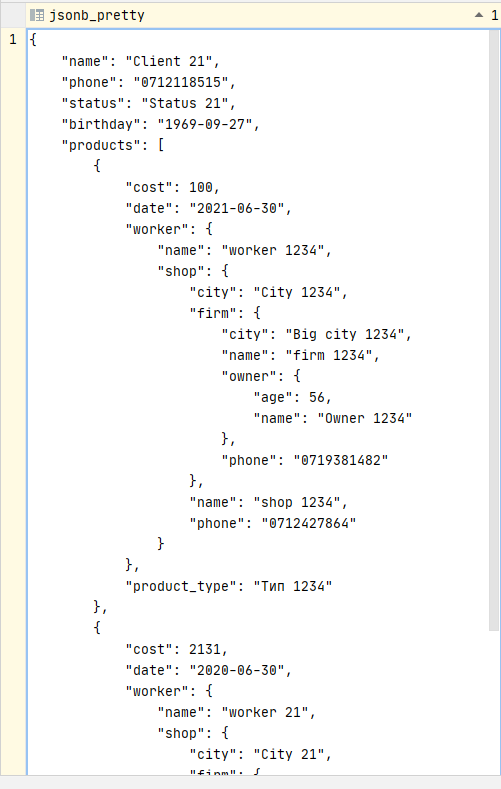


Рисунок 22 – Результат добавления

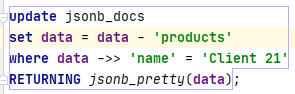


Рисунок 23 – Изменение данных

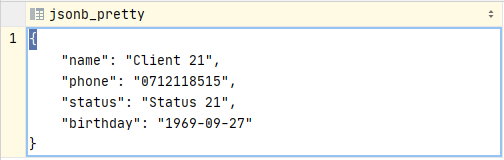


Рисунок 24 – Результат изменения

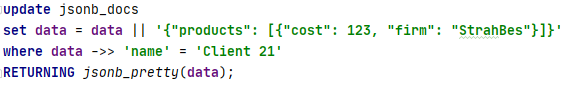


Рисунок 25 – Изменение данных

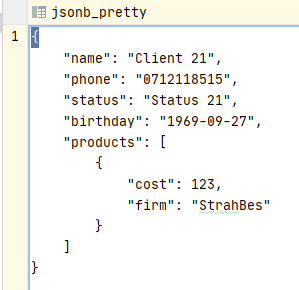


Рисунок 26 – Результат изменения

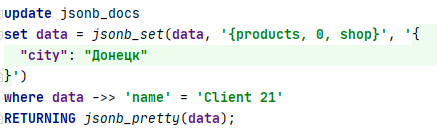


Рисунок 27 – Изменение данных

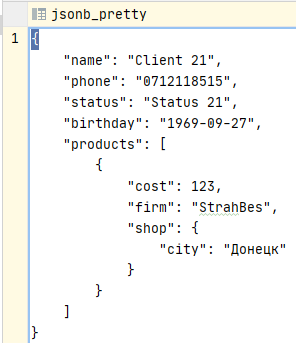


Рисунок 28 – Результат изменения

**Удаление:**

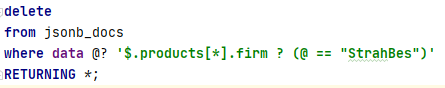


Рисунок 29 – Удаление данных

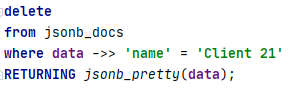


Рисунок 30 – Удаление данных

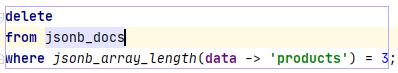


Рисунок 31 – Удаление данных

**Индексы:**

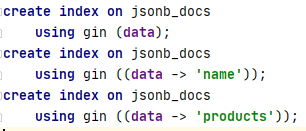
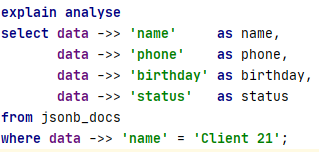
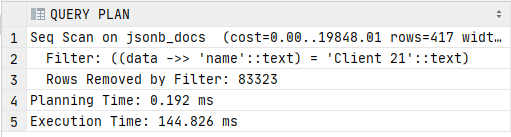


Рисунок 32 – Создание индексов

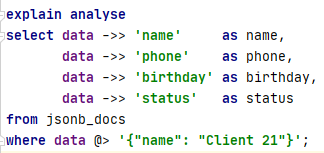
**Оптимизация запросов:**

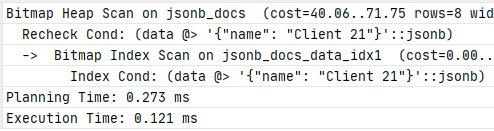
Неоптимизированный запрос





Оптимизированный запрос

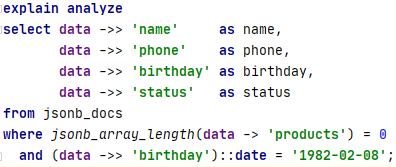


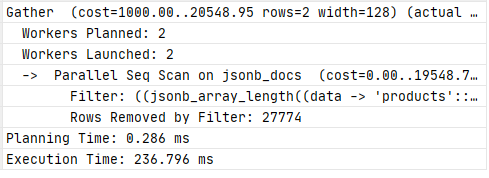


По результатам видим, что:

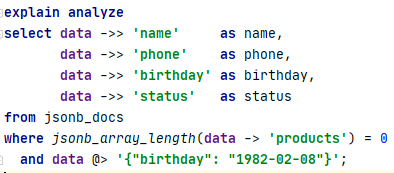
* оценка затратности операции (cost) уменьшилась значительно, было – 19848.01, стало – 71.75;
* по затраченному времени (Execution Time) на запрос показатель уменьшился, было – 144.826 ms, стало – 0.121 ms;
* последовательное сканирование (Seq Scan) сменилось на Bitmap Index Scan (используется индекс jsonb\_docs\_data\_idx1 для определения нужных нам записей).

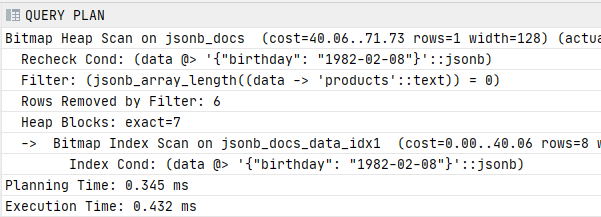
Неоптимизированный запрос





Оптимизированный запрос

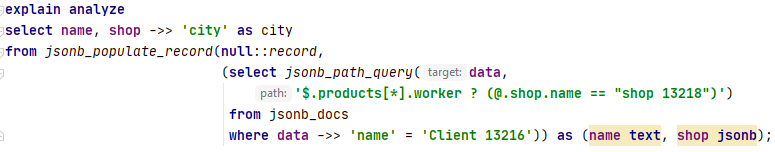


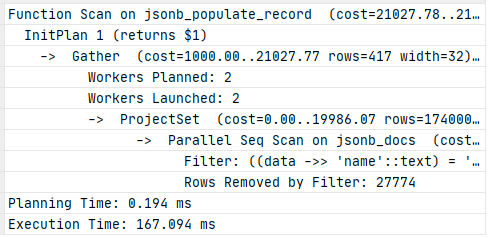


По результатам видим, что:

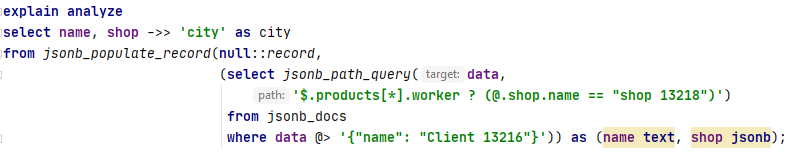
* оценка затратности операции (cost) уменьшилась значительно, было – 20548.95, стало – 71.73;
* по затраченному времени (Execution Time) на запрос показатель уменьшился, было – 236.796 ms, стало – 0.432 ms;
* последовательное сканирование (Seq Scan) сменилось на Bitmap Index Scan (используется индекс jsonb\_docs\_data\_idx1 для определения нужных нам записей) а затем PostgreSQL лезет в саму таблицу: (Bitmap Heap Scan), чтобы убедиться, что эти записи на самом деле существуют.)

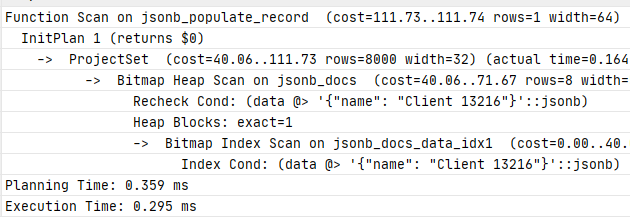
Неоптимизированный запрос





Оптимизированный запрос

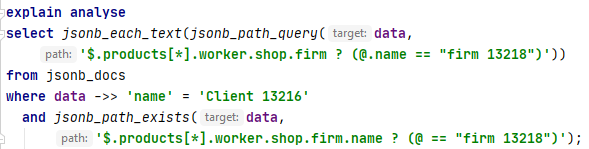


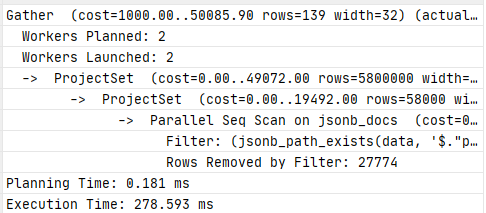


По результатам видим, что:

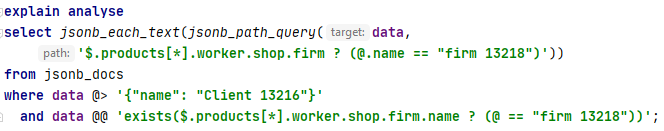
* оценка затратности операции (cost) уменьшилась значительно, было – 21027.78, стало – 111.74;
* по затраченному времени (Execution Time) на запрос показатель уменьшился, было – 167.094 ms, стало – 0.295 ms;
* последовательное сканирование (Seq Scan) сменилось на Bitmap Index Scan (используется индекс jsonb\_docs\_data\_idx1 для определения нужных нам записей) а затем PostgreSQL лезет в саму таблицу: (Bitmap Heap Scan), чтобы убедиться, что эти записи на самом деле существуют.)

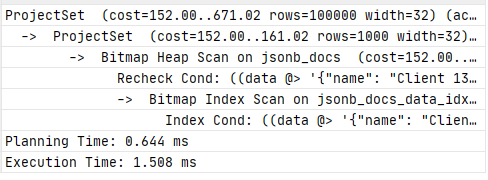
Неоптимизированный запрос





Оптимизированный запрос

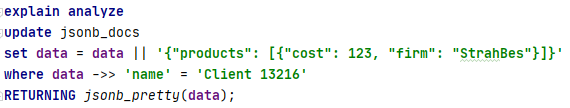


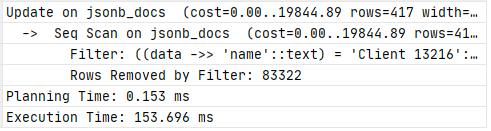


По результатам видим, что:

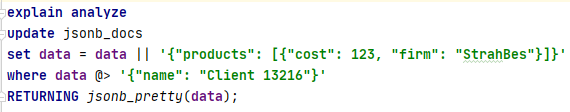
* оценка затратности операции (cost) уменьшилась значительно, было – 50085.90, стало – 671.02;
* по затраченному времени (Execution Time) на запрос показатель уменьшился, было – 278.593 ms, стало – 1.508 ms;
* последовательное сканирование (Seq Scan) сменилось на Bitmap Index Scan (используется индекс jsonb\_docs\_data\_idx1 для определения нужных нам записей) а затем PostgreSQL лезет в саму таблицу: (Bitmap Heap Scan), чтобы убедиться, что эти записи на самом деле существуют.)

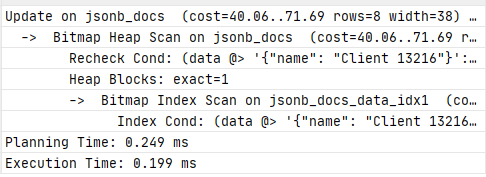
Неоптимизированный запрос





Оптимизированный запрос





По результатам видим, что:

* оценка затратности операции (cost) уменьшилась значительно, было – 19844.89, стало – 71.69;
* по затраченному времени (Execution Time) на запрос показатель уменьшился, было – 153.696 ms, стало – 0.199 ms;
* последовательное сканирование (Seq Scan) сменилось на Bitmap Index Scan (используется индекс jsonb\_docs\_data\_idx1 для определения нужных нам записей) а затем PostgreSQL лезет в саму таблицу: (Bitmap Heap Scan), чтобы убедиться, что эти записи на самом деле существуют.)