Департамент образования и науки города Москвы

Государственное автономное образовательное учреждение высшего образования города Москвы

«Московский городской педагогический университет»

Институт цифрового образования

Департамент информатики, управления и технологий

ДИСЦИПЛИНА:

Инструменты для хранения и обработки больших данных

**Лабораторная работа №4.1.**

**Тема:**

**«**Сравнение подходов хранения больших данных**»**

Выполнила: Сергеева А. И., группа: АДЭУ-211

Преподаватель: Босенко Т. М.

Москва

2024

**Цель работы:** сравнить производительность и эффективность различных подходов к хранению и обработке больших данных на примере реляционной базы данных PostgreSQL и документоориентированной базы данных MongoDB.

**Оборудование и программное обеспечение**

- Компьютер с операционной системой Ubuntu.

- PostgreSQL.

- MongoDB.

- Python 3.x.

- Библиотеки: psycopg2, pymongo, pandas, matplotlib.

**Ход работы**

**Вариант 12**

**Оценить производительность при работе с вложенными структурами данных на примере системы управления проектами**

С постоянным ростом объемов данных приходится задумываться об эффективных способах и инструментах их хранения. Данная лабораторная работа посвящена хранению данных в объектно-реляционной базе данных PostgreSQL и документоориентированной MongoDB, а также произведена оценка производительности при работе с загруженными в них данными.

MongoDB — это ориентированная на документы база данных NoSQL с открытым исходным кодом, которая использует для хранения структуру JSON. Вместо таблиц и строк, как в реляционных базах данных, в MongoDB коллекции и документы, которые состоят из пар «ключ — значение». Коллекция — это набор документов, эквивалент таблицы реляционной базы данных. Каждый документ может отличаться друг от друга размером, содержанием и количеством полей. MongoDB подходит для большого объёма полуструктурированных и неструктурированных данных.

PostgreSQL позволяет гибко управлять базами данных, создавать, модифицировать или удалять записи, отправлять транзакцию — набор из нескольких последовательных запросов на особом языке запросов SQL.

Хотя обе базы данных имеют свои сильные стороны, они также имеют свои ограничения. Например, MongoDB может иметь проблемы с транзакционностью и согласованностью данных, а PostgreSQL может иметь проблемы с хранением больших объемов данных и сложных документов.

Одной из ключевых проблем при работе с базами данных является хранение и обработка вложенных структур данных. Вложенные структуры - это данные, которые содержат другие данные внутри себя, такие как массивы, объекты или документы.

Для начала работы были импортированы необходимые библиотеки и сгенерированы тестовые данные для системы управления проектами (рисунок 1-2). В данных хранятся наименование проекта, менеджер, отвечающий за проект, начало и окончание проекта, его статус и бюджет, в рамках вложенных данных представлена информация о задачах по каждому из проектов.



Рисунок 1 – Импорт библиотек и подключение к MongoDB и PostgreSQL



Рисунок 2 – Генерация данных по проектам и задачам

Для работы с вложенными структурами в PostgreSQL я создала 2 таблицы: в первой хранится информация о проектах, во второй – о задачах и содержит внешний ключ, связывающий ее с таблицей о проектах (рисунок 3).

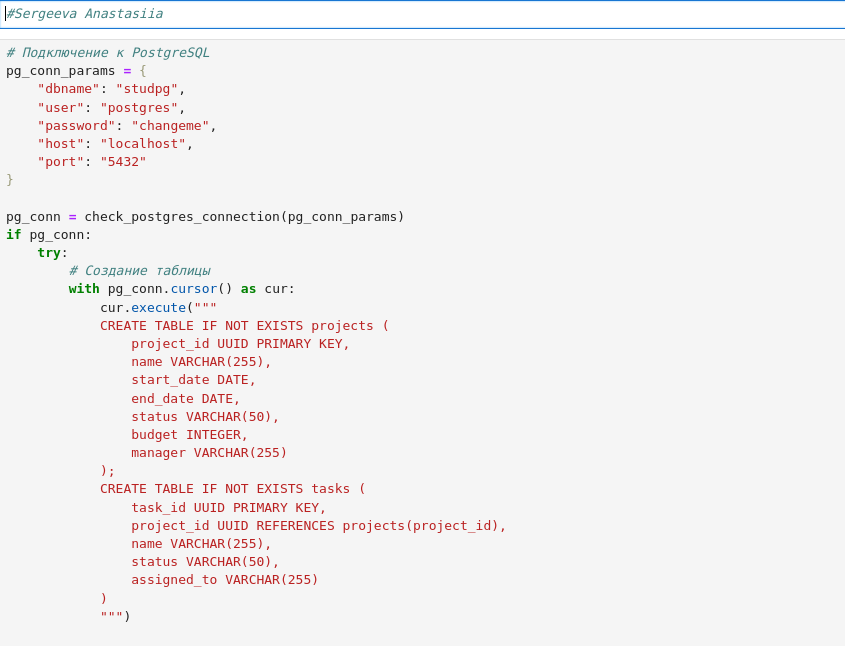


Рисунок 3 – Создание таблиц по проектам и задачам

Далее данные таблицы были заполнены данными и выполнен запрос, объединяющий 2 таблицы и выводящий наименование проекта для каждой задачи и статус задачи. Также было посчитано врем, затраченное на вставку данных и выполнение запроса (рисунок 4). С результатами запроса можно ознакомиться на рисунке 5.

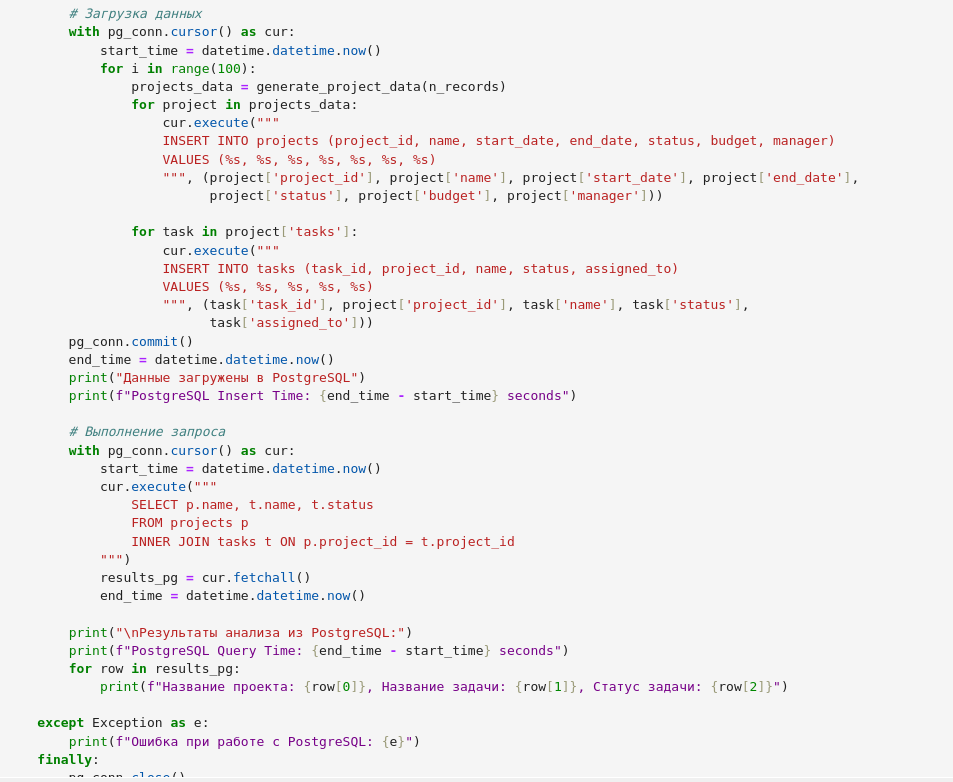


Рисунок 4 – Вставка данных и пример запроса

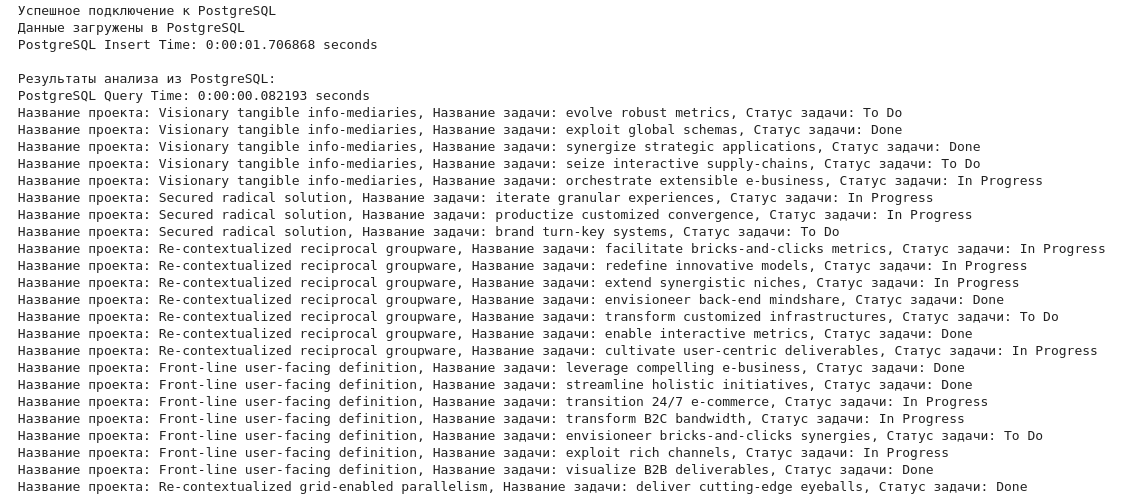


Рисунок 5 – Результат 1 запроса в PostgreSQL

Далее был выполнен запрос, использующий подзапрос и осуществлено сравнение времени выполнения запроса с индексацией и без (рисунки 6-7).

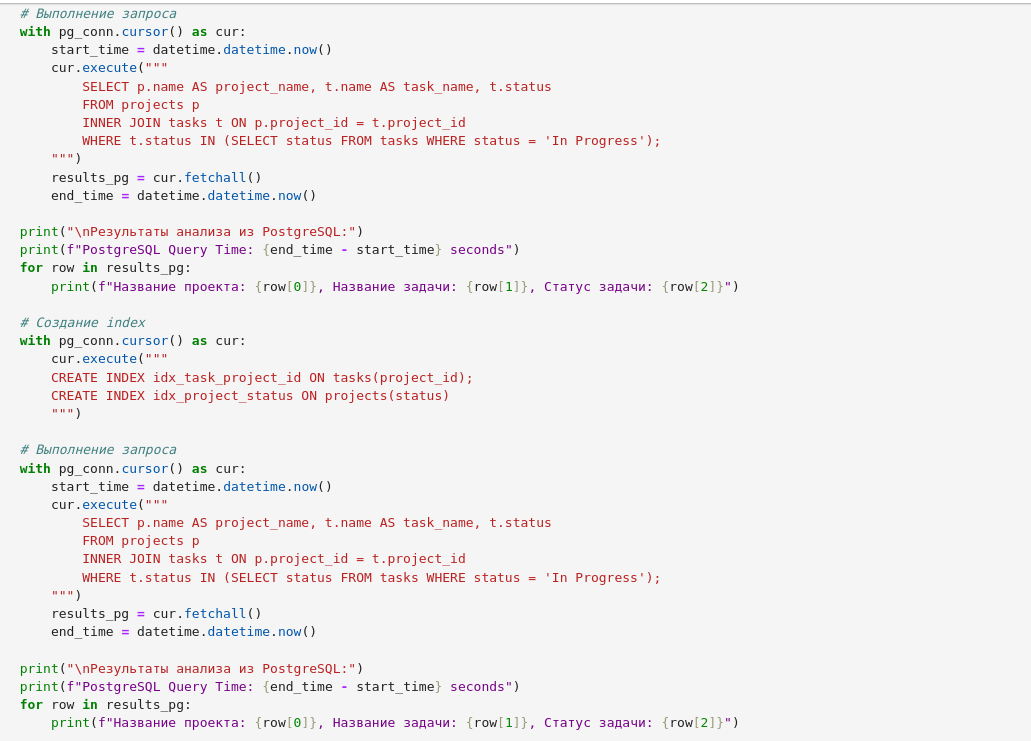


Рисунок 6 – Выполнение 2 запроса с индексацией и без

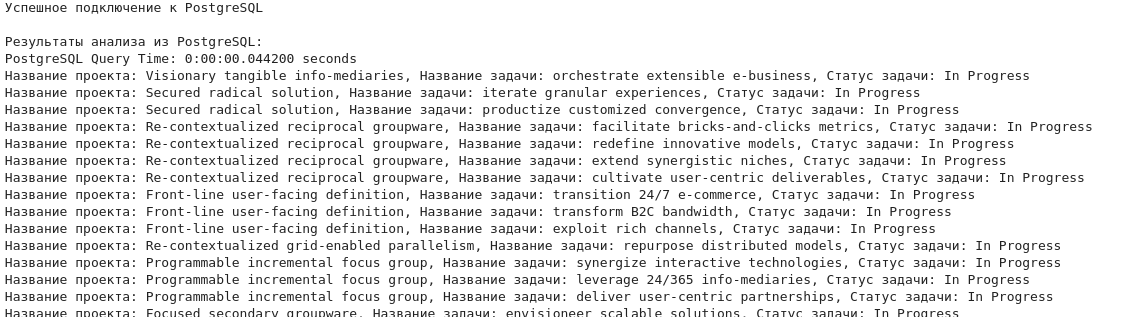


Рисунок 7 – Результат выполнения 2 запроса

Также была применена специальная функция для анализа производительности запросов, применяемая в PosgreSQL (EXPLAIN (ANALYZE)) (рисунок 8). Запрос выполняется быстро, за 0,684 миллисекунды. Планирование запроса занимает относительно много времени, 1,514 миллисекунды. Основная часть времени тратится на выполнение операций агрегации и объединения.



Рисунок 8 – Оценка производительности запроса в PostgreSQL

Далее анализировалась производительность при работе с данными в MongoDB. Для этого генерировались новые данные, загружались, а также считалось время выполнения запроса, выводящего информацию о наименованиях проекта, задачи и ее статуса (рисунок 9).

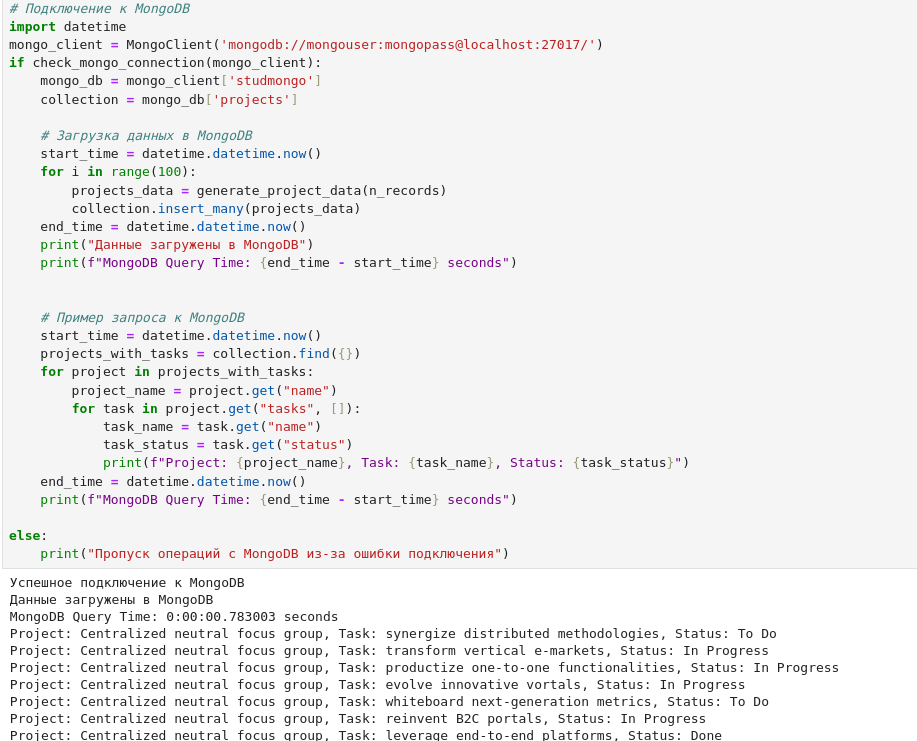


Рисунок 9 – Загрузка данных в MongoDB и выполнение 1 запроса

Далее та же методика индексации была применена и для данных в MongoDB. Был проведен сравнительный анализ времени выполнения запроса, выводящего проект, менеджера и задачи для проекта со статусом «In Progress» (рисунок 10).



Рисунок 10 – Сравнительный анализ времени выполнения запроса с индексацией и без для MongoDB

И в рамках заключительного этапа анализа были построены сравнительные графики и сделаны выводы.

Данные через MongoDB загрузились намного быстрее (рисунок 11). Данные в виде JSON-документов легко загружаются и обрабатываются. Также не нужно создавать много таблиц и настраивать связи между ними, как в PostgreSQL. Все данные загружаются в рамках одного документа, а не по разным таблицам.

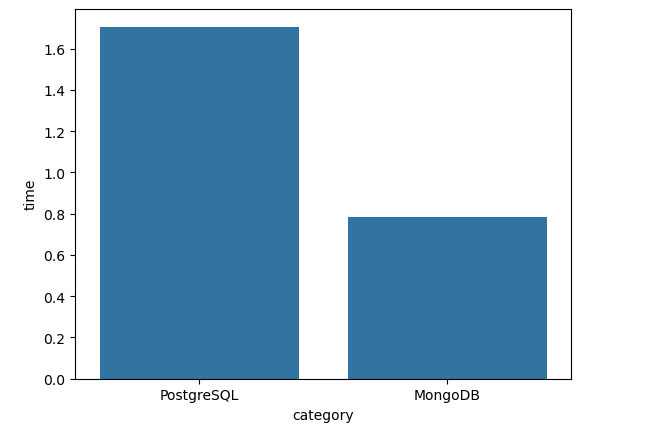


Рисунок 11 – Сравнительный анализ загрузки данных

При этом запросы выполняются быстрее в PostgreSQL (рисунок 12). Сложные запросы легко обрабатываются с помощью языка SQL, а также у PostgreSQL есть сложный оптимизатор запросов. При этом, если бы пришлось работать с большим количеством таблиц, то MongoDB мог бы отрабатывать быстрее, ведь данные дезорганизованы и не требуют таких соединений.

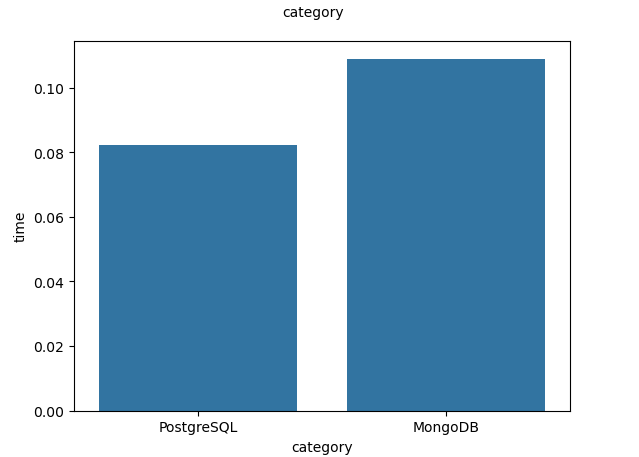


Рисунок 12 – Сравнительный анализ обработки запросов

С индексацией запросы в PostgreSQL срабатывают намного быстрее (рисунок 14).

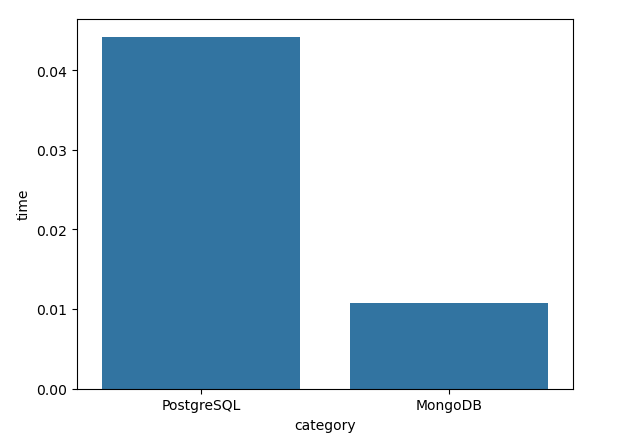


Рисунок 13 – Сравнительный анализ обработки по индексации для PostgreSQL

Но при этом введение индексации в MongoDB только ухудшило ситуацию (рисунок 14). Скорее всего структура индекса не подходит для запроса.

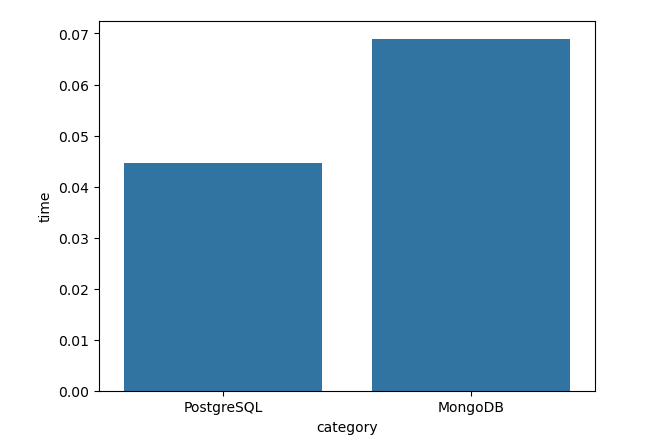


Рисунок 14- Сравнительный анализ обработки по индексации для MongoDB

**Выводы**

1. Для хранения данных больше подойдет MongoDB, т. к. имеет документоориентированную модель данных, которая позволяет хранить вложенные структуры данных в виде JSON-документов. Это делает MongoDB подходящим выбором для систем управления проектами, где данные проектов имеют сложную структуру, в них легче вносить изменения и не приходится работать с большим количеством таблиц.
2. PostgresSQL больше подходит для простых данных, хотя при этом сложные запросы на нем выполнять проще и быстрее.
3. Также MongoDB является лучшим вариантом, т. к. обращения, связанные с частым объединением таблиц, выполняются быстрее, чем в PostgresSQL, нужные данные в одном документе.
4. MongoDB имеет более высокую скорость чтения и записи: MongoDB имеет более высокую скорость чтения и записи, чем PostgresSQL, особенно при работе с вложенными структурами данных. Это делает MongoDB подходящим выбором для систем управления проектами, где требуется высокая скорость чтения и записи данных.
5. PostgresSQL имеет более высокую безопасность: PostgresSQL имеет более высокую безопасность, чем MongoDB, особенно при работе с конфиденциальными данными.