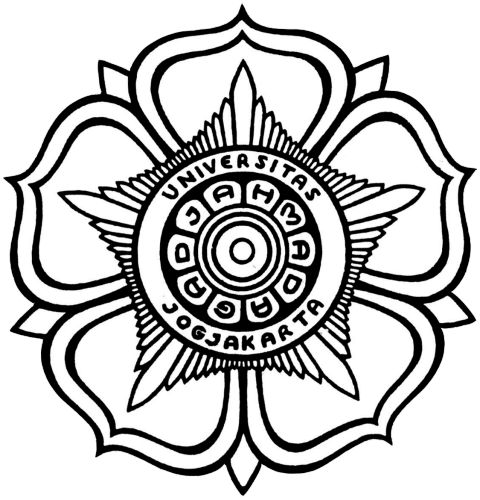
ASSIGNMENT REPORT
DATABASE TECHNOLOGY
Task 3 : DB Benchmarking



Advisor :

Dr. Guntur Dharma Putra, S.T., M.Sc.

Created by :

Jhon Samuel Kudadiri

(22/503772/TK/55066)

FACULTY OF ENGINEERING

UNIVERSITAS GADJAH MADA

YOGYAKARTA

2024

I. Introduction

In the realm of data management, efficiency and performance are significant. As organizations increasingly rely on data-driven decision-making, understanding the nuances of executing data queries and manipulations becomes critical. This short essay delves into a comparative analysis of execution times for Data Query Language (DQL) and Data Manipulation Language (DML) operations performed via Postman API and PostgreSQL. By benchmarking these operations, we aim to uncover insights that can guide developers and database administrators in optimizing their workflows and choosing the most efficient tools for their data management tasks.

II. Benchmarking

1. Data Query Language
    a. Query to show all book in the database

```
SELECT * FROM "Book"
```

Iteration	Postman API	PostgreSQL
1	247	285
2	327	169
3	317	218
4	311	149
5	352	206
6	400	154
7	403	143
8	315	223
9	392	201
10	319	151
Mean	338.3	189.9
Std Dev	49.03751398	45.24366131

    b. Query to show all book data with the author id

```
SELECT b."BookName", b."BookPrice", b."LanguageID", (SELECT
ROW_TO_JSON(BookAuthor_obj) FROM( SELECT "AuthorID", "BookID"
FROM "Book_Author" WHERE "BookID" = b."BookID")
BookAuthor_obj) AS "Book_Author" FROM "Book" b
```

Iteration	Postman API	PostgreSQL
1	357	174
2	364	138
3	344	126
4	361	137
5	327	163

6	350	117
7	329	155
8	324	171
9	326	163
10	306	129
Mean	338.8	147.3
Std Dev	19.16478252	20.34726299

c. Query to show all possible conditions of the book

```
SELECT * FROM "Condition"')
```

Iteration	Postman API	PostgreSQL
1	319	218
2	295	203
3	322	151
4	333	151
5	297	174
6	313	206
7	291	131
8	292	170
9	288	139
10	304	129
Mean	305.4	167.2
Std Dev	15.45746853	32.50914401

d. Query to show all wishlists

```
SELECT w."WishlistID", w."BookID",(SELECT
ROW_TO_JSON(WishlistAcc_obj) FROM (SELECT "CustomerID",
"Username","Name"FROM "CustomerAcc" WHERE "CustomerID" =
w."WishlistID") WishlistAcc_obj) AS "CustomerAcc" FROM
"Wishlist_Book" w
```

Iteration	Postman API	PostgreSQL
1	342	160
2	305	158
3	306	145

4	326	165
5	357	152
6	322	157
7	330	144
8	313	163
9	327	136
10	296	147
Mean	322.4	152.7
Std Dev	18.34968967	9.452219022

2. Data Manipulation Language
    a. Query to post a new condition

```
INSERT INTO "Condition" ("ConditionName") VALUES ($1)
RETURNING *
```

Iteration	Postman API	PostgreSQL
1	164	166
2	138	59
3	134	91
4	301	83
5	214	65
6	137	66
7	157	53
8	146	57
9	135	59
10	162	72
Mean	168.8	77.1
Std Dev	52.24046963	33.44464747

    b. Query to update information about a book

```
UPDATE "Book" SET "BookName" = $1 WHERE "BookID" = $2
RETURNING *
```

Iteration	Postman API	PostgreSQL
1	321	152
2	372	149

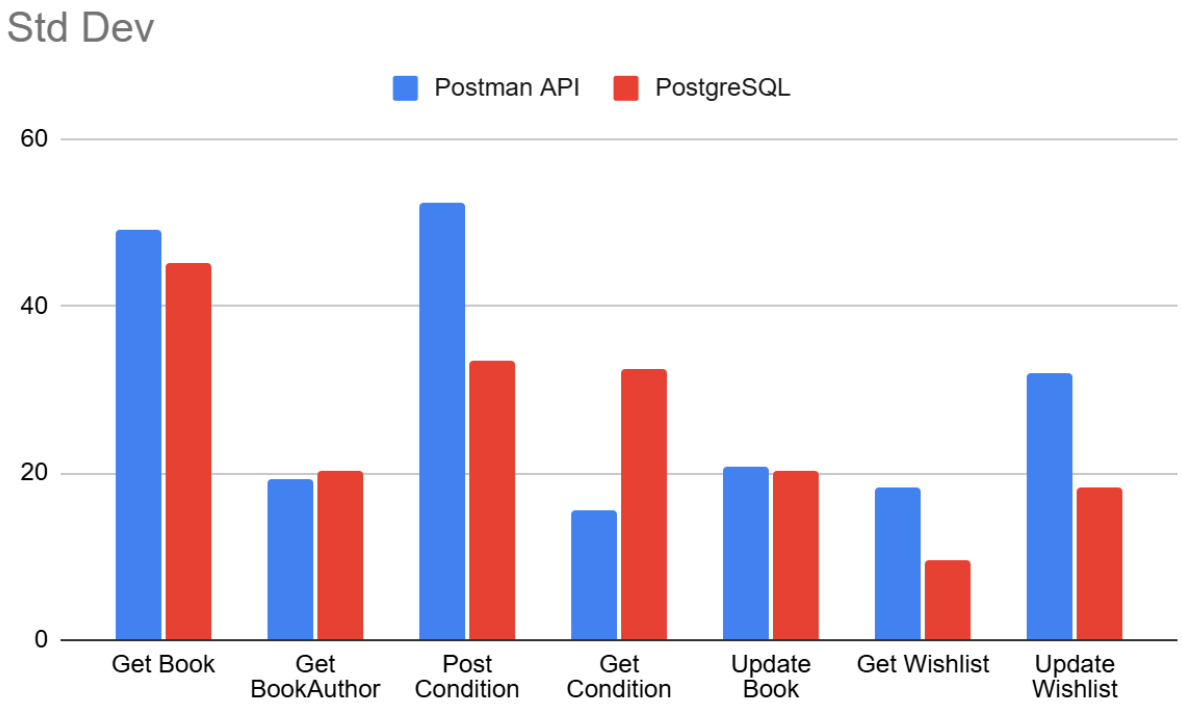
3	322	132
4	354	195
5	326	134
6	317	162
7	316	135
8	359	150
9	339	130
10	313	130
Mean	333.9	146.9
Std Dev	20.88300745	20.22896713

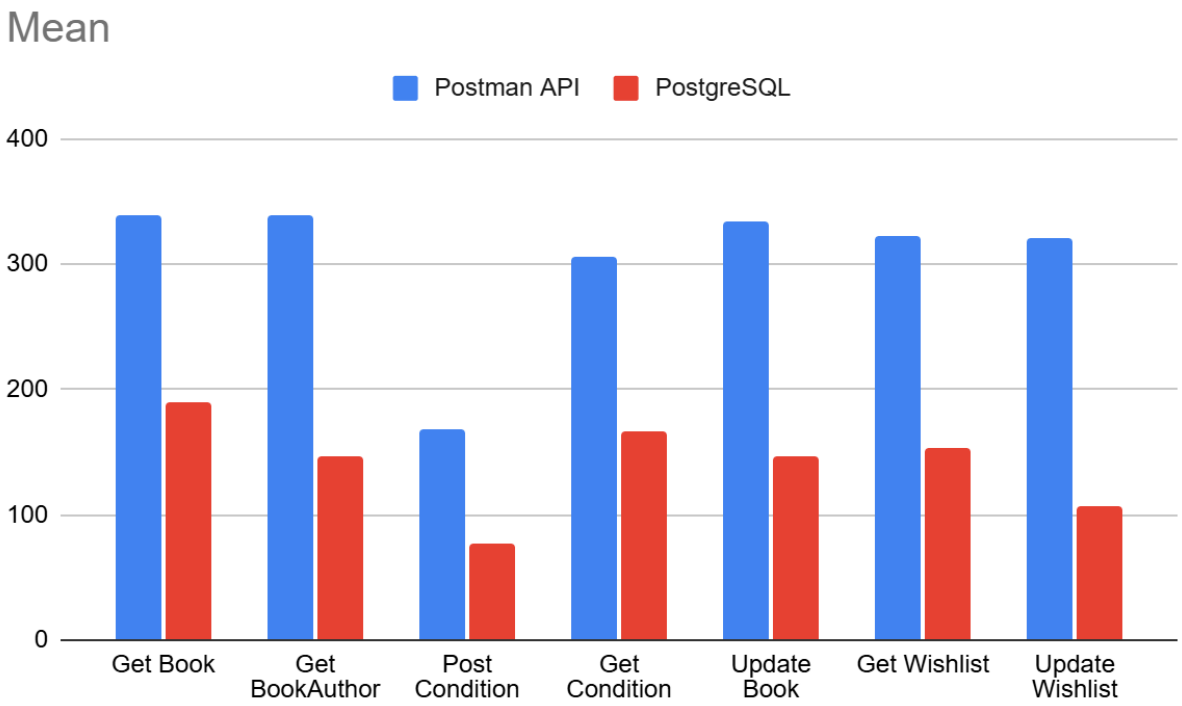
c. Query to add a book to a wishlist

```
    INSERT INTO "Wishlist_Book"  ("WishlistID", "BookID") VALUES
($1, $2)
```

Iteration	Postman API	PostgreSQL
1	249	100
2	314	91
3	316	138
4	312	108
5	325	84
6	321	102
7	374	100
8	353	101
9	319	105
10	324	140
Mean	320.7	106.9
Std Dev	31.97238392	18.25407108

III. Statistics





The full sheet :
https://docs.google.com/spreadsheets/d/1KxjfsM8juKqrzUD8SLjymxKqX9T8bUMsAmNM-nEB4Jg/edit?usp=sharing

IV. Discussion

In this experiment, I tested the execution process of queries using manual script execution in PostgreSQL and API requests in Postman. The database is created through PostgreSQL and API is built through Express while utilizing routes, controller, services, and an index.js file. PostgreSQL is an open-source object-relational database management system (ORDBMS). PostgreSQL supports SQL (relational) and JSON (non-relational) querying. Therefore, it is a versatile choice for various types of applications. On the other hand, Postman is a widely used API (Application Programming Interface) client that allows developers to create, share, test, and document API testing by making HTTP requests to RESTful APIs, ensuring that they work as expected.
For this experiment, there are several methods that I tested. They are 'get book','get bookauthor', 'post condition', 'get condition', '(put) update book', 'get wishlist', and 'update wishlist'. Based on the execution time data that I collected, it can be seen that querying through pgAdmin PostgreSQL is faster than Postman API testing. There are several reasons:

- pgAdmin is directly connected to the PostgreSQL database while API access needs to involve additional network latency and use HTTP/HTTPS protocol which introduces overhead for request/response headers and additional data encapsulation
- pgAdmin typically maintains a persistent connection to the database, reducing the overhead of repeatedly establishing and tearing down connections while API access might establish a new connection, perform the request, and then close the connection, adding the time required for each connection
- pgAdmin needs not to serialize large volumes of data into JSON or other formats, making it more efficient while API access needs to serialize large datasets into formats like JSON, which can be bulky and slow to transfer

V. Conclusion

In conclusion, the comparative benchmarking of DQL and DML operations between Postman API and PostgreSQL reveals significant insights into their performance dynamics. The findings underscore the importance of selecting the appropriate tool based on the specific requirements of data querying and manipulation tasks. As data continues to be a critical asset in the digital age, optimizing these operations not only enhances efficiency but also empowers organizations to leverage their data more effectively. This study provides a foundational understanding that can drive better performance tuning and tool selection in the ever-evolving landscape of data management.