The comparative study of indexing techniques in different database systems

Student: Sokrat Bashirov

Instructors: Steve Kaisler and Jamaladdin Hasanov

Class: CSCI_6917_10: Guided Research Methods

Date: 8/10/2023

Project Objective

The primary objective of this project is to conduct a comparative study of query performance in MySQL and PostgreSQL databases with and without indexes. Aiming to evaluate the impact of indexing on query execution time using a real-world dataset and identify the performance differences between the two database systems.

Heilmeier questions:

What are you trying to do?

Comparing the query performance of MySQL and PostgreSQL databases with and without indexes to understand the influence of indexing on query execution.

How is it done today, and what are the limits of current practice?

The extent to which indexes influence query performance is not always well understood. My research aims to provide clear insights into the effectiveness of indexes and their limitations.

What is new in your approach, and why do you think it will be successful?

Approach involves systematically testing both databases without any indexes and then introducing indexes incrementally to study their impact.

Who cares?

Database administrators, developers, and researchers who seek to optimize database performance for their applications.

What are the risks and the payoffs?

Potential complexities in data collection, analysis, and accurately reflecting real-world workloads.

How much will it cost? How long will it take?

The cost involves computational resources, database hosting

What are the midterm and final exams to check for success?

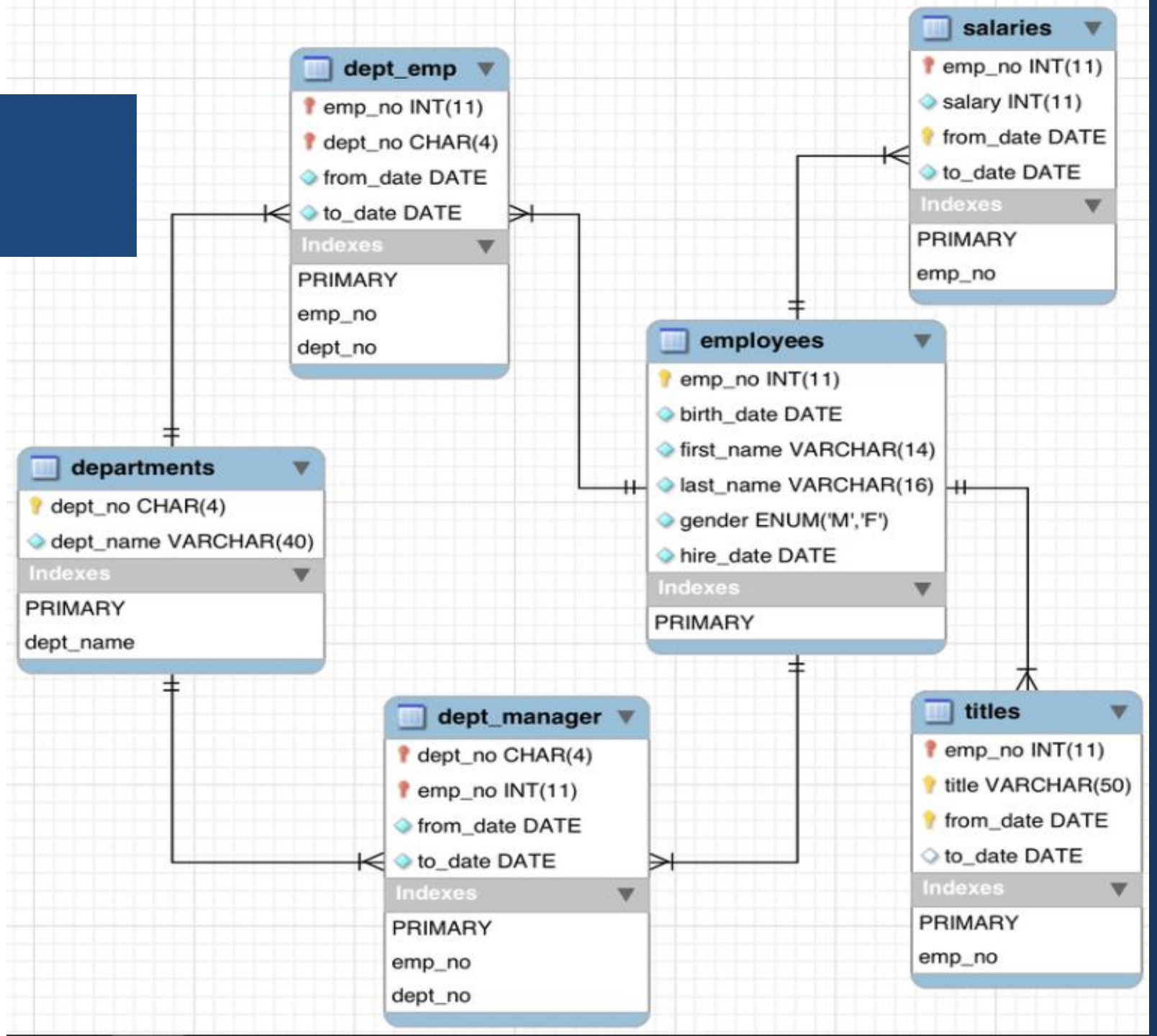
Successfully executing and comparing queries without indexes.
A comprehensive evaluation of queries with indexes added, supported by data analysis and visualization.

Why now?

The increasing reliance on databases for applications

Research plan

Database


dept_emp
emp_no INT(11)
dept_no CHAR(4)
from_date DATE
to_date DATE
Indexes
PRIMARY
emp_no
dept_no
salaries
emp_no INT(11)
salary INT(11)
from_date DATE
to_date DATE
Indexes
PRIMARY
emp_no
employees
emp_no INT(11)
birth_date DATE
first_name VARCHAR(14)
last_name VARCHAR(16)
gender ENUM('M','F')
hire_date DATE
Indexes
PRIMARY
departments
dept_no CHAR(4)
dept_name VARCHAR(40)
Indexes
PRIMARY
dept_name
dept_manager
dept_no CHAR(4)
emp_no INT(11)
from_date DATE
to_date DATE
Indexes
PRIMARY
emp_no
dept_no
titles
emp_no INT(11)
title VARCHAR(50)
from_date DATE
to_date DATE
Indexes
PRIMARY
emp_no

```
SELECT emp_no, COUNT(*) AS count FROM employees GROUP BY emp_no;
```

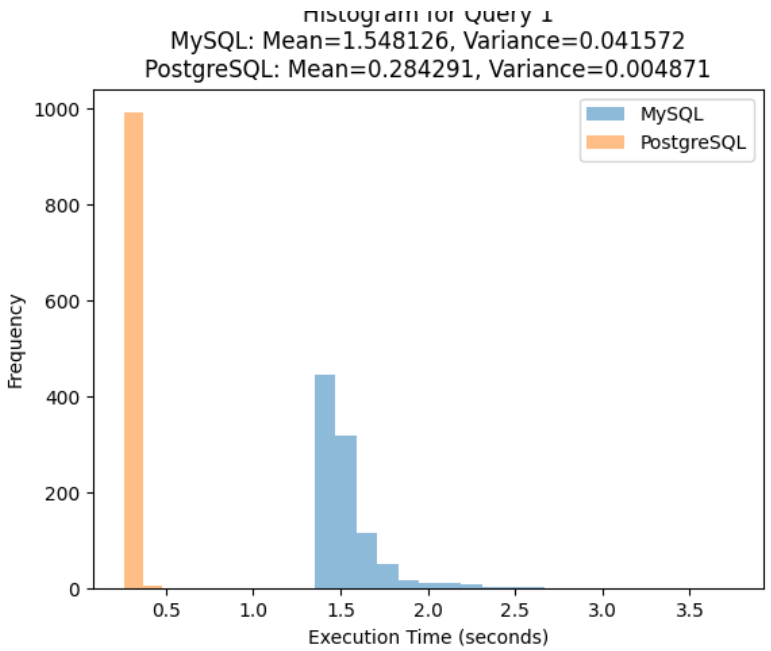
```
SELECT * FROM salaries WHERE salary = 94443 OR salary = 59571;
```

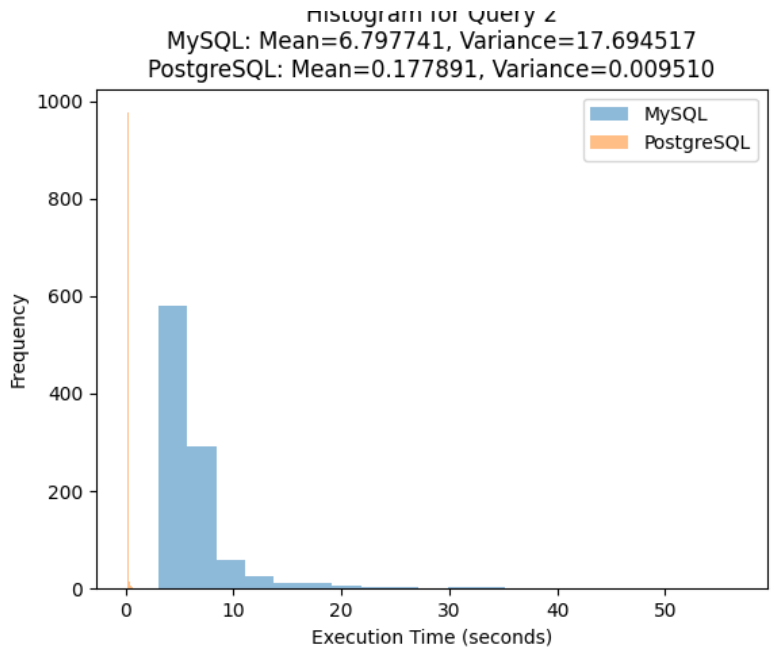
```
SELECT E.*, S.* FROM employees E JOIN salaries S ON E.emp_no = S.emp_no
WHERE E.first_name = 'Duangkaew';
```

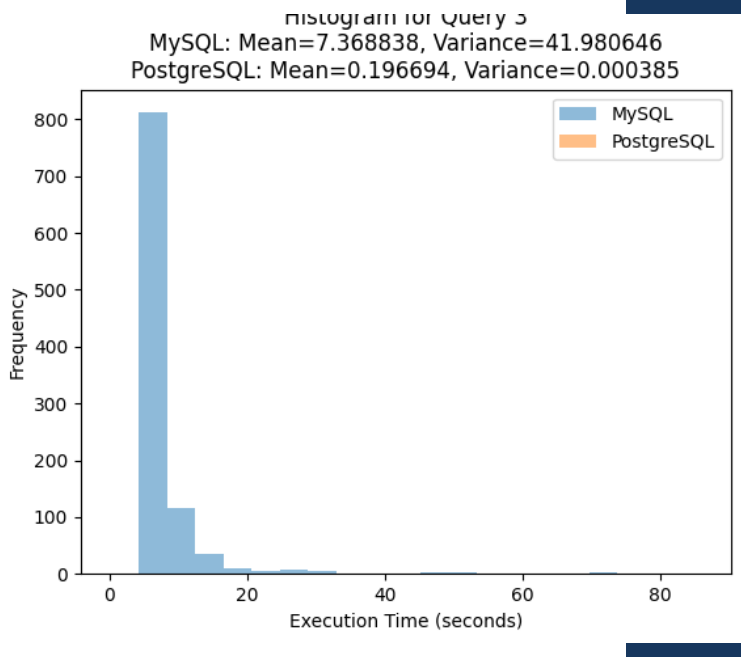
```
SELECT * FROM titles WHERE title LIKE 'senior%';
```

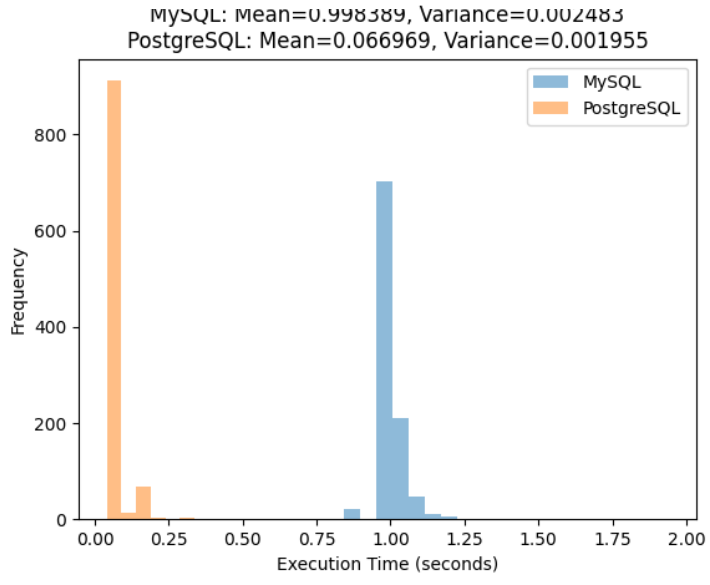
```
SELECT E.*, T.* FROM employees E JOIN titles T ON E.emp_no = T.emp_no
WHERE E.first_name = 'Duangkaew';
```

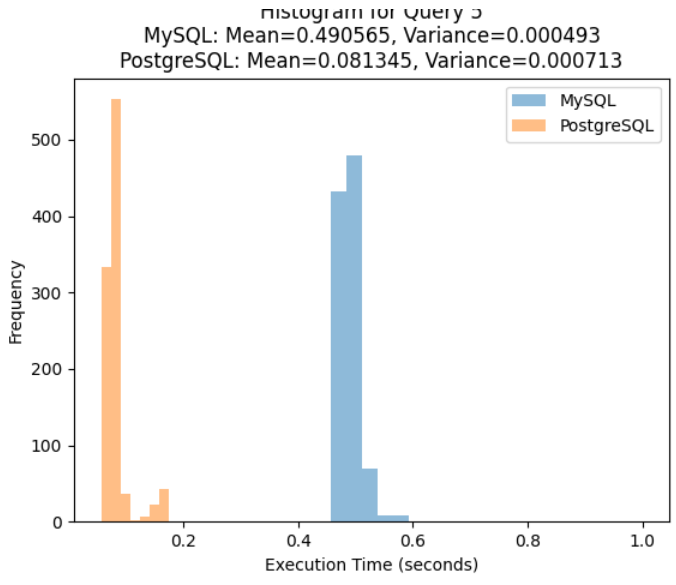
Query run without indexes



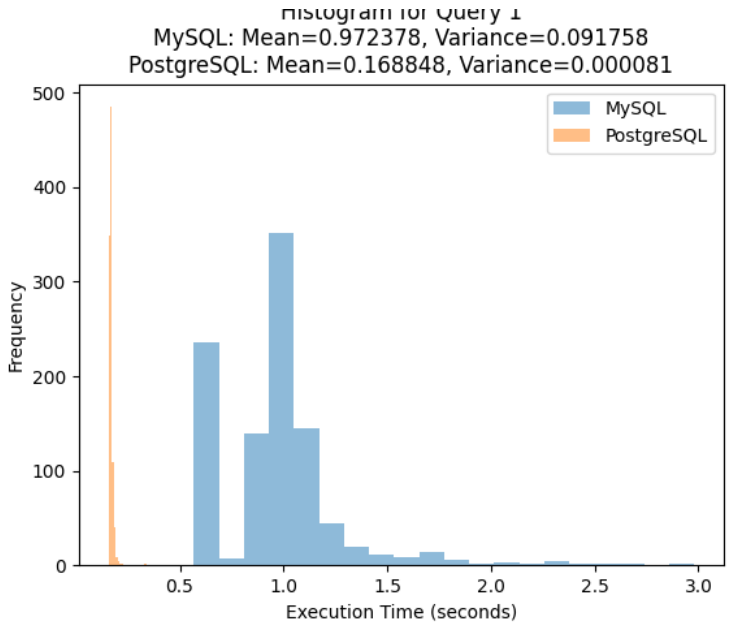


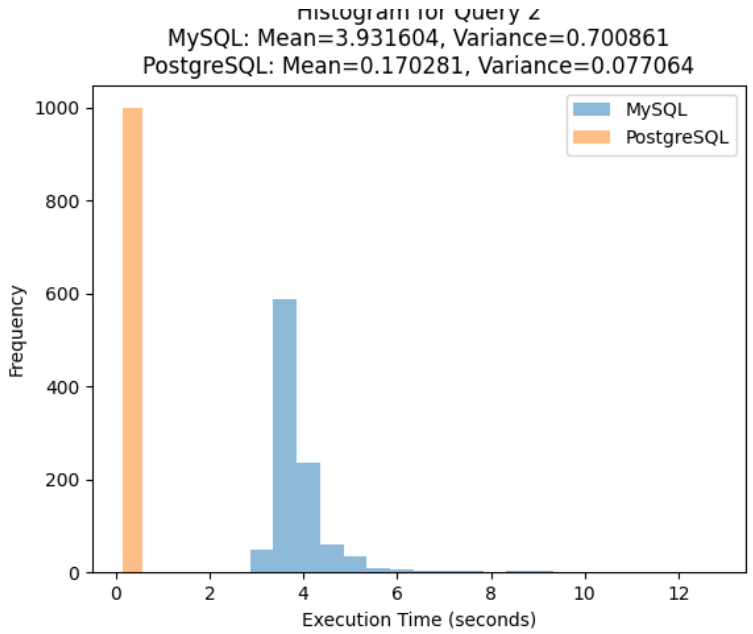


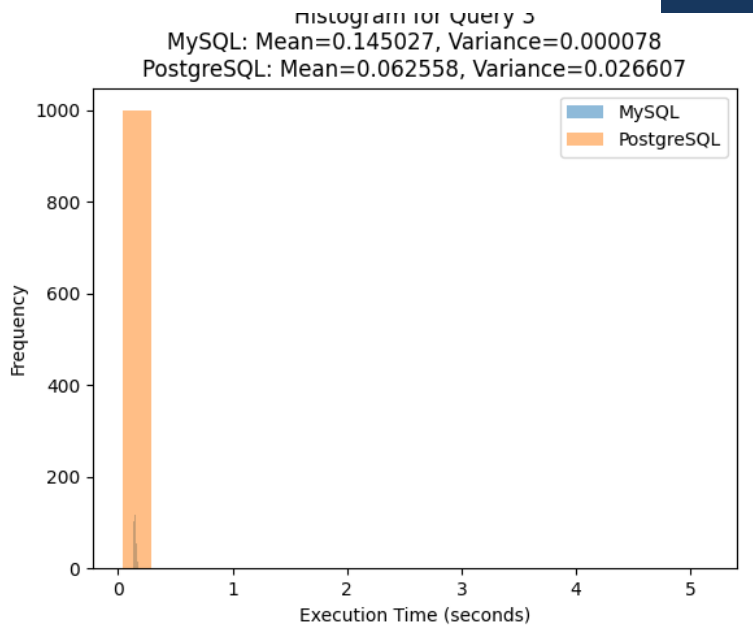


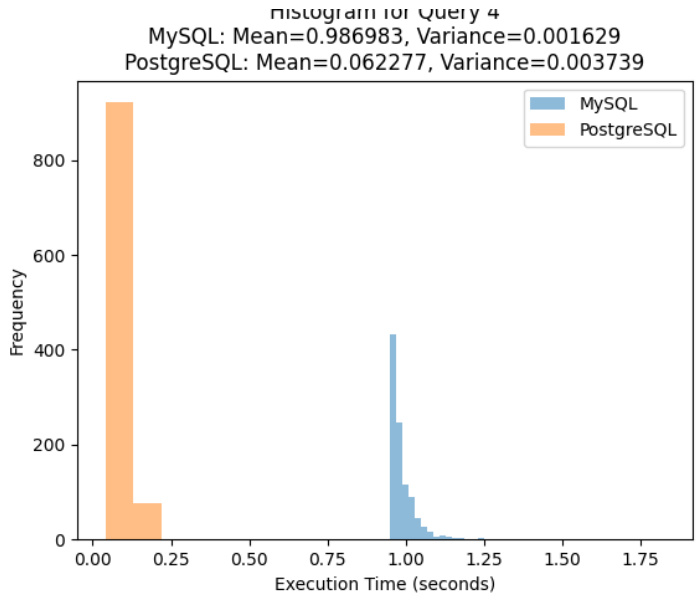


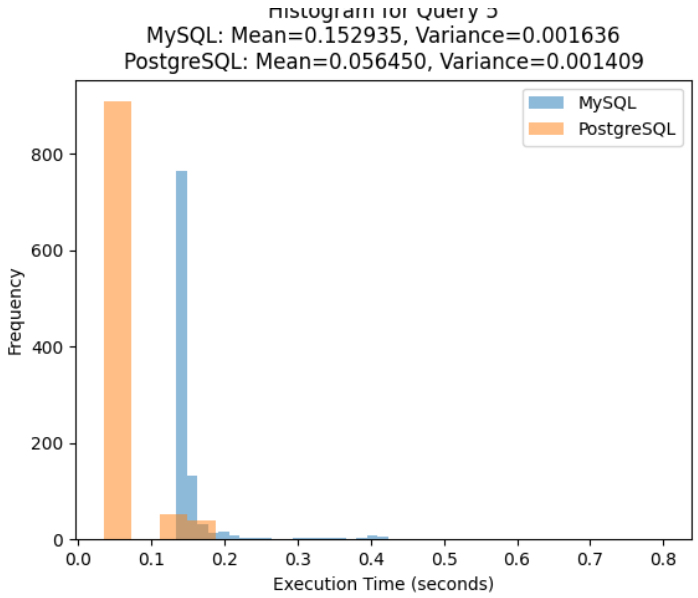
Query run with PK, FK



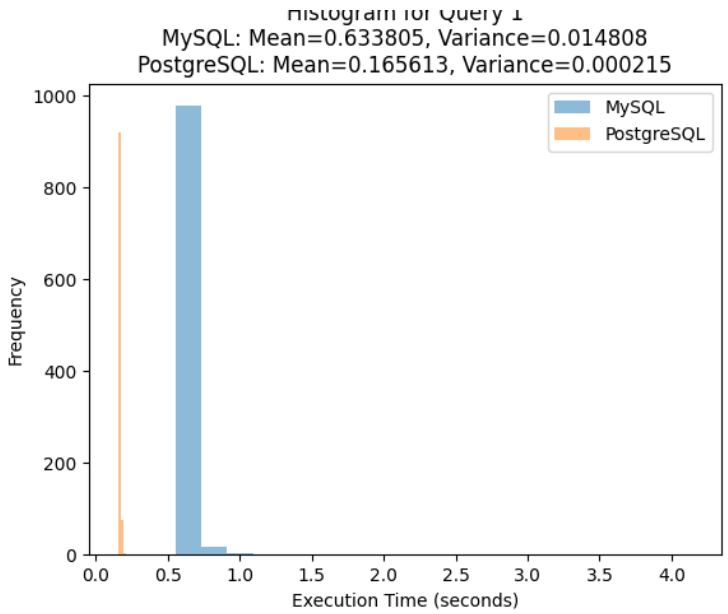


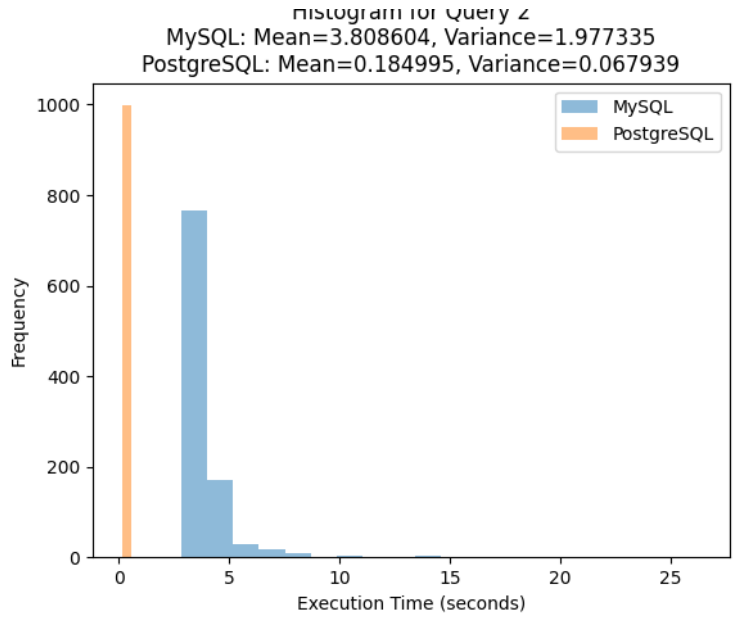


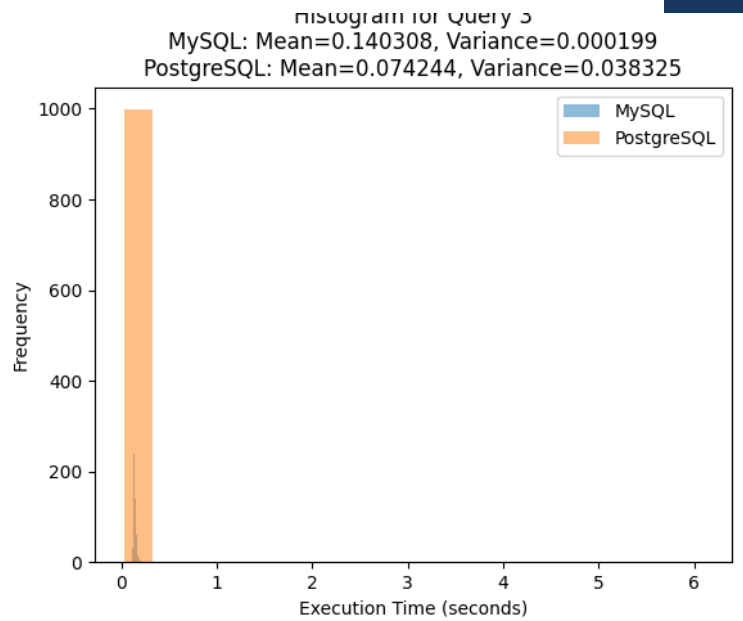


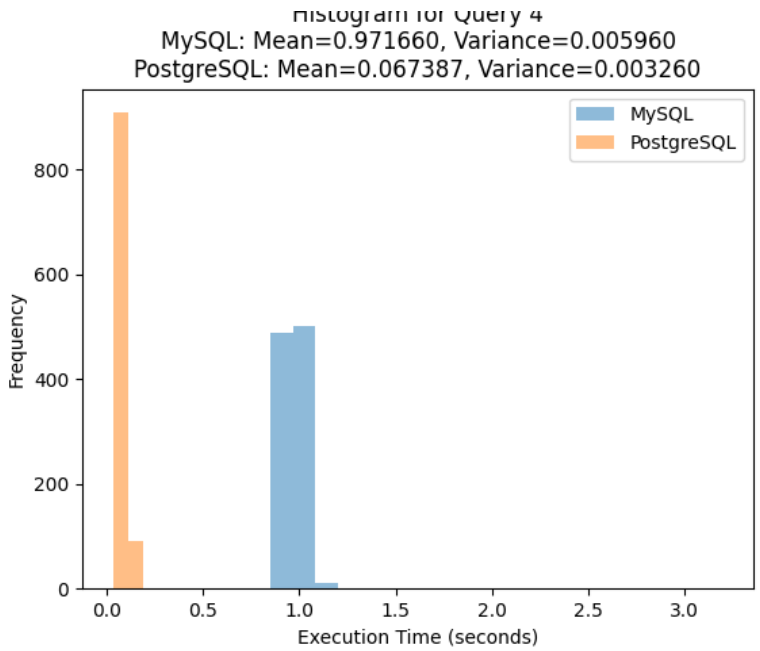


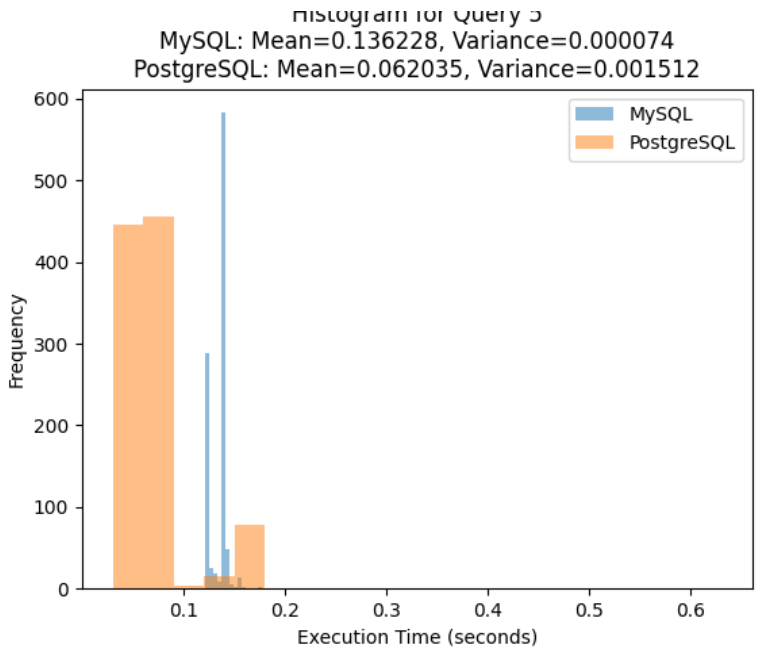
Query run with index











Results

	No Index		PK, FK		Index	
	MySQL	PostgreSQL	MySQL	PostgreSQL	MySQL	PostgreSQL
Query 1	M=1.548126 V=0.041572	M=0.284291 V=0.004871	M=0.972378 V=0.091758	M=0.168848 V=0.000081	M=0.633805 V=0.014808	M=0.165613 V=0.000215
Query 2	M=6.797741 V=17.694517	M=0.177891 V=0.009510	M=3.931604 V=0.700861	M=0.170281 V=0.077064	M=3.808604 V=1.977335	M=0.184995 V=0.067939
Query 3	M=7.368838 V=41.980646	M=0.196694 V=0.000385	M=0.145027 V=0.000078	M=0.062558 V=0.026607	M=0.140308 V=0.000199	M=0.074244 V=0.038325
Query 4	M=0.998389 V=0.002483	M=0.066969 V=0.001955	M=0.986983 V=0.001629	M=0.062277 V=0.003739	M=0.971660 V=0.005960	M=0.067387 V=0.003260
Query 5	M=0.490565 V=0.000493	M=0.081345 V=0.000713	M=0.152935 V=0.001636	M=0.056450 V=0.001409	M=0.136228 V=0.000074	M=0.062035 V=0.001512

Key Findings:

- Without Indexes: PostgreSQL consistently outperformed MySQL in query execution time, indicating its inherent optimization and advanced query processing capabilities.

- With Indexes: The introduction of indexes in both databases led to significant improvements in query execution time, reducing the overall response time for queries.

Performance advantages

- **Parallel Query Execution**: While both databases support parallelism, PostgreSQL's implementation of parallel query execution allows queries to be split into smaller tasks that are executed concurrently, potentially speeding up query performance for certain workloads.
- **Advanced Indexing Techniques**: PostgreSQL offers features like Generalized Inverted Indexes (GIN) and Generalized Search Tree (GiST) indexes, which provide specialized indexing methods for complex data types like arrays, full-text search, and geometric data. These indexing options can be particularly useful for applications that require advanced search capabilities.
- **Materialized Views:** PostgreSQL supports materialized views that store precomputed results, reducing the need for repeated complex calculations. This feature accelerates query performance by retrieving data directly from the materialized view.
- **Concurrency Control:** PostgreSQL's advanced concurrency control mechanisms, such as Multi-Version Concurrency Control (MVCC), mitigate locking and contention issues, enabling efficient multi-user access. Effective concurrency control mechanisms to ensure efficient multi-user access without sacrificing performance.

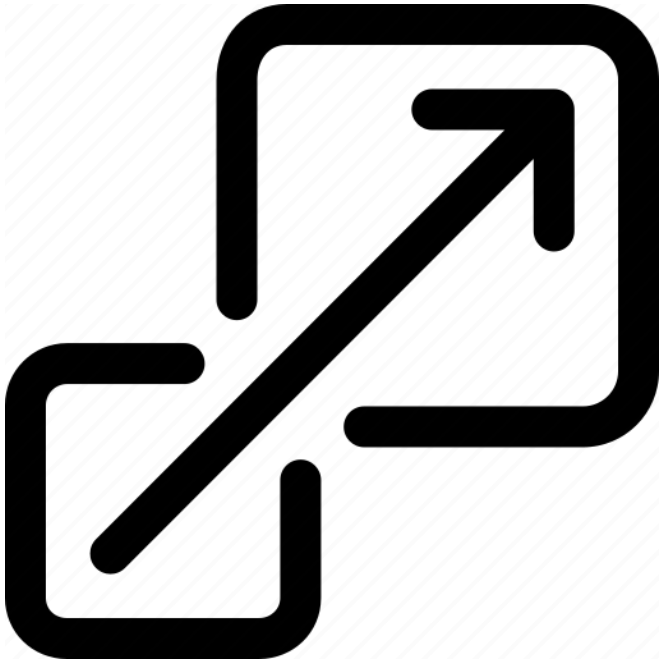
Conclusion

The comparative study of indexing techniques in MySQL and PostgreSQL databases has provided valuable insights into the impact of indexing on query performance. I have executed a set of representative queries 1000 times in both databases without indexes and with indexes added, enabling a thorough evaluation of their respective performances.

Future Work

Fine-tuning Indexing Strategies

Benchmarking with Other Databases



Scale Testing

Thank you for your attention!