Logo, company name

Description automatically generated

Group: Shrey Chaudhary (2863973)

Omi Patel (2835823)

Mail ID: [2863973@vikes.csuohio.edu](mailto:2863973@vikes.csuohio.edu)

[2835823@vikes.csuohio.edu](mailto:2835823@vikes.csuohio.edu)

Subject: CIS 611

Supervised by Prof. - SS Chung

**Project Proposal**

**Building a Simple Query Optimizer with Performance Evaluation Experiment on Query Rewrite Optimization**

**Goals:**

* Create a basic optimizer to analyze and suggest optimized versions of SQL queries..
* Conduct systematic tests to measure the impact of query rewrite on query performance
* Investigate and implement various optimization strategies within the query optimizer.
* Measure performance metrics, compare against baseline, document implementation, and draw conclusions for future improvements.

**Programming language:**  JAVA, JAVASCRIPT or PYTHON

**Platform:** Microsoft SQL SERVER, SQL Server Management Studio, Visual Studio Code

**Specification**

* When it comes to computer systems, query optimization is very challenging because of how complicated it is.
* Code Optimization is one of the most important ways to improve the speed of code.
* This optimizer carefully looks over SQL lines and figures out the fastest way to get to the data that is being sought.
* This in-depth study showed that optimizing the schema is not only a good idea, but also necessary to get and keep high speed in the constantly changing world of database operations.

**Query Basics**

* Enhance access to data
* You should check to see if your app is getting more data than it needs. This usually means it's getting to too many rows and columns.
* Examine the database to see if the Microsoft SQL server is looking at too many rows.

**Rebuild the queries**

* The goal is to find other ways to get the result we want, knowing that this might not always mean getting the same set of results from Microsoft SQL Server.
* By looking into how to change queries into similar forms, we hope to find ways to make things run faster.
* This lets us get better efficiency without having to stick to the original set of results. The main goal is to come up with new ways to write queries so that they run faster in Microsoft SQL Server environments.

## **Advantages**

 The optimized code has the following advantages-

## **Enhanced Performance**

## **Resource Utilization**

## **Faster Execution**

## **Improved User Experience**

## **Cost Efficiency**

**Methodology**

* A simple optimizer that tests both the original query and its optimized version for speed.
* We plan to find the most efficient method by looking at the processing costs of all the possible ways to run both the original and rewritten queries.
* The final goal is to find the best plan that gives the best performance between the original query and its improved version, which will make query processing more efficient overall.

Here, Q1 is given

SELECT T1.x1, SUM (T2.x2)

FROM T1, T2

WHERE T2.x2 = (SELECT SUM (T3.x3)

FROM T3

WHERE T1.x1 = T3.x3)

GROUP BY T1.x1;

**First try to rewrite this Q1 to RQ1.**

**Given Input of System Parameters and Table Info**

Given a page size of 4096 bytes and a block size of 100 pages, let's consider table sizes:

For T1 with tuples of 20 bytes, there are 1000 pages.

For T2 with tuples of 40 bytes, there are 500 pages.

For T3 with tuples of 100 bytes, there are 2000 pages.

Considering predicate selectivity and join conditions, the query processing cost is calculated using the formula:

Query Processing Cost = Number of Disk I/O \* (Average Seek Time + Average Latency).

The result, representing join cost in milliseconds, is then converted into hh:mm:ss format for readability.

Additionally, projection rates are factored in, with 70% of projections being retained. Assuming a data transfer rate of 45 MB/sec, this analysis provides a comprehensive understanding of the query processing cost, taking into account table sizes, join conditions, and projection rates.