Project Proposal

Part 1: Group Info

- *Group Members:*
    - Aishwarya Krishnamurthy (ak73043@usc.edu)
    - Muskan Moti Hassanandani (hassanan@usc.edu)
    - Nisarg Shihora (shihora@usc.edu)

Part 2: Project Idea

Open-source database system: **Neo4j**

Current limitations: Neo4j utilizes a hash join algorithm to handle join operations. However, for complex queries involving multiple joins, particularly with larger datasets or queries involving frequent lookups, the hash join algorithm can lead to performance bottlenecks.

Proposed idea and justification: We propose implementing two alternative join algorithms: Index Nested Loop Join and Sort-Merge Join, to address these limitations. These algorithms are designed to optimize join performance, particularly in scenarios where indexed data or sorted datasets are involved. By allowing the Neo4j query planner to select the optimal join method depending on query characteristics, we aim to reduce execution time and resource usage, enhancing overall query efficiency.

Part 3: Proposed Solution

System component and extension: We will modify the Neo4j query planner and query execution engine. Based on the query structure and available indexes, the query planner will be updated to detect queries that could benefit from Index Nested Loop Joins or Sort-Merge Joins.
Design/architecture:

1. **Index Nested Loop Join:** This algorithm will leverage indexes to minimize scan times for join operations. The outer dataset will iterate through each row, while the inner dataset will retrieve matching rows using index lookups.
2. **Sort-Merge Join:** In this algorithm, input datasets will be sorted initially, followed by a merging process. Duplicate values and varying row counts will be managed through a robust merging mechanism that handles differing dataset sizes effectively.
   Illustrative Diagram:

Part 4: Proposed Evaluation

Evaluation Plan:

- *Datasets:* We will use mock datasets that simulate real-world graph data patterns, along with publicly available graph datasets (e.g., DBLP, Twitter).
- *Workloads:* The evaluation will involve complex queries with multiple joins and simple joins to compare performance gains.
- *Benchmarks and metrics:* Primary metrics include execution time, throughput, and speedup ratio. We will compare the current hash join method with the new join algorithms regarding these metrics across varied dataset sizes and complexity levels.

Part 5: Proposed Timeline

1. *Design Phase:* Finalize algorithm design, including query planner modifications and execution engine integration.
2. *Implementation Phase:*
    - Week 3: Implement Index Nested Loop Join
    - Week 4: Implement Sort-Merge Join
    - Week 5-6: Integrate algorithms with the Neo4j query planner and execution engine
3. *Evaluation Phase:* Conduct performance evaluations using selected datasets and metrics.
4. *Reporting/Documentation Phase:* Prepare project documentation, evaluation reports, and finalize code review.