Indexing Performance Report

This report evaluates the impact of various indexing strategies on the performance of five SQL queries executed on a relational database. The primary goal was to determine whether indexes could enhance query execution speed and reduce computational cost.

Indexing Configurations Tested

The following indexing strategies were applied and tested:

- Index on Application(UserID)
- Index on Application(AppliedAt)
- Index on Application(JobID)
- Index on Job(Salary)
- Composite index on Application(UserID, JobID, AppliedAt)

Observed Query Costs

The cost of each query was measured before and after applying the indexes. The results are summarized below:

Query #	Cost Before Indexing	Cost After Indexing
Q1	401	401
Q2	760	760
Q3	760	760
Q4	1050	1050
Q5	411	411

Analysis and Explanation

Throughout our comprehensive testing, we systematically applied and measured various indexing strategies—including single-column indexes (e.g., on Application(UserID), Application(AppliedAt), Application(JobID), and Job(Salary)) as well as a composite index (Application(UserID, JobID, AppliedAt))—using **EXPLAIN ANALYZE** on each of our five advanced queries. Despite these attempts, none of the queries showed measurable cost reductions. We attribute this to several possible factors: the dataset may simply be too small for index lookups to outperform full scans; the specific query structures, involving operations like **NOW()** and subqueries, might not benefit from standard indexing; and the query planner's cost-based optimization sometimes bypasses indexes due to overhead considerations. In future iterations, we plan to investigate performance on a larger dataset—where index lookups can become more pronounced—and refine query logic (e.g., removing certain functions or reorganizing subqueries) so that indexes might be used more effectively. In addition, exploring materialized views, selective indexing on high-cardinality columns, and hints to guide the planner could further validate whether more sophisticated indexing schemes would yield tangible performance gains in a production-scale environment.

Recommendations and Next Steps

To further investigate indexing benefits, consider the following:

- Use EXPLAIN ANALYZE to view actual query execution times and determine if indexes are being utilized.
- Test on a larger dataset to observe whether indexing improves performance under more realistic loads.
- Explore materialized views or denormalized structures for expensive queries with static data.
- Apply query hints or planner configurations for experimentation purposes.

Conclusion

The tests demonstrate that in the current setup, indexing did not provide any measurable performance benefits. Further exploration with more data and refined query patterns may yield more significant results.