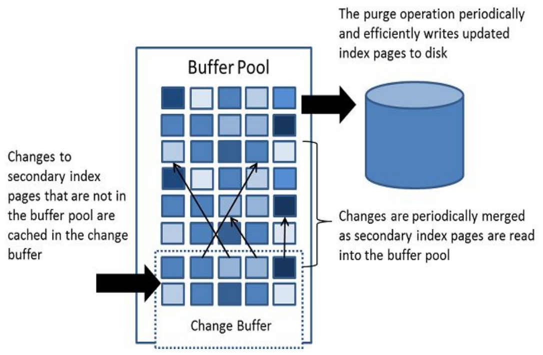
Database Benchmarking Project - Part II

MySQL

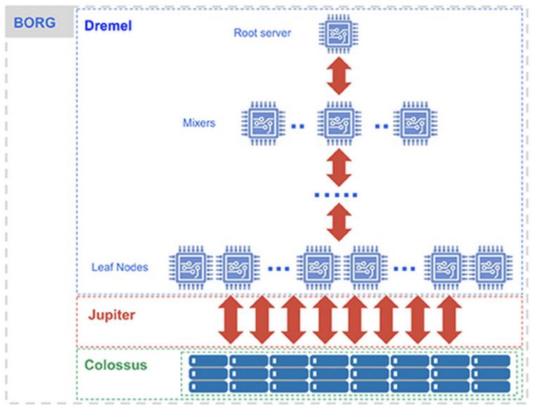
- Index Types:
  - B-tree
  - Hash
  - R-trees
  - Inverted lists
- Join Types:
  - Nested loop join
  - Block nested loop join
- Buffer Pool:
  - *InnoDB buffer pool is the memory space that holds many in-memory data structures of InnoDB, buffers, caches, indexes and even row-data.*
  - For systems running with less than 1GB of RAM, it is better to go with the MySQL default configuration value of 128MB for InnoDB buffer pool size.


The purge operation periodically and efficiently writes updated index pages to disk
Buffer Pool
Changes to secondary index pages that are not in the buffer pool are cached in the change buffer
Changes are periodically merged as secondary index pages are read into the buffer pool
Change Buffer

- Performance Tracking:
  - MySQL supports a profiler which provides statistics for query execution time

BigQuery

- Index Type(s):
  - N/A
- Built on the Google DREMEL query system
- Storage is handled through the Google Colossus system and computing is handled by the Google Borg system, so storage and computing scale independently
- Colossus allows splitting of the data into multiple partitions to enable parallel read
- Structure:



- Performance Tracking:
  - Query jobs in include diagnostic speed and query plan data that can be accessed through the BQ UI or through the jobs.get method using the BQ command line interface.
- Uses columnar storage
- Because data can be shuffled between nodes, it is important to prune data early in the join

Carter Wilson
Parker Moore
CS487P

Why MySQL & BigQuery?

We chose MySQL and BigQuery as our database management systems (DBMS) for two reasons. First, they are both widely used in industry today and will be relevant experience when entering the workforce. And secondly, we are interested in comparing these two as they have inherently different system architectures and am curious as to the performance of querying an indexed (MySQL) versus a non-indexed (BigQuery) DBMS. Both systems will be hosted on the cloud, with BigQuery working on its native Google Cloud platform and MySQL hosted on Amazon's RDS platform. Depending on the results of our experiments, this may also give us insight into the relative performance differences between the two platforms.

Performance Experiments

1. Baseline performance check of each system
   a. Experiment Specifications
      i. Test performance time of each system using using a select(*) command to determine baseline performance. Based on this we can isolate the performance effect of the functionality used in the rest of our experiments.
      ii. Use 10K tuple relation
      iii. SELECT * FROM relation
      iv. N/A
      v. The query time should be about the same in theory, but most likely will not be due to system differences. This should give us a baseline idea of how much difference in speed we can expect as a factor in other queries.

2. Compare performance of comparison-based query and range-based query in each system to measure a quantifiable impact of indexes in MySQL vs. no index in BigQuery
   a. Experiment Specifications
      i. Test performance time of each system using a 'where' clause in one query to find a specific integer and a 'where' clause in another query to find a range.
      ii. Use 10k tuple relation
      iii. SELECT * FROM relation WHERE unique2 = 150 / SELECT * FROM relation WHERE unique2 BETWEEN 5000 AND 6000
      iv. Perform range-based portion of experiment with clustered and nonclustered index in MySQL
      v. With a comparison-based query or a range query with a clustered index, MySQL should go faster. With a non-clustered index, BigQuery should perform faster. This can also give us a baseline difference due to indexes that we can use to interpret the results of question 5.

3. Compare left join across systems to find the impact of joins on the difference in the systems' performance.
   a. Experiment Specifications
      i. Test the performance time of each system against a single left join query and a multiple left join query.
      ii. Use two relations to test a single join, one with size 10k and one with size 1k. For multiple relations, use two 10k relations and 1k relations where the intermediate relation is the join between the two 10k relations.
      iii. Use Wisconsin benchmark join queries 10 and 11 for each system.
      iv. Since BigQuery does not allow for configuration of join algorithms, we think it is best here to allow each systems' optimizer to pick the join algorithm. We can then at least see the join performance with what is most likely their fastest choice.
      v. On a non-indexed system it should take longer to compare and find matches than on an indexed system. Therefore, MySQL should outperform BigQuery.

4. Compare performance of different types of aggregation to determine the effects of aggregation on the performance difference between the two systems.
   a. Experiment Specifications
      i. Test the performance time of each system with count, avg, or sum aggregation included in a query
      ii. Use 1k tuple relation and test against two attributes on each system
      iii. Select count(x) from table, select avg(x) from table, select sum(x) from table
      iv. N/A
      v. Count < sum < avg. Count just counts the tuples while sum has to access memory of a record in an attribute and sum it together with a total. Avg does the same as count but ends with dividing the sum by n where n is the total number of numbers summed.

5. Sizeup
   a. Experiment Specifications
      i. Explores how the two systems respond to different sizes of relations to isolate how size affects the performance of each system when trying to find a specific tuple.
      ii. Use 1k, 10k, 100k, 1000k relations
      iii. SELECT * FROM relation WHERE unique2 = 100
      iv. N/A
      v. There should be a linear relationship between relation size and delta time up to an extent. However, since BigQuery does not use indexes, we may see that the indexing ability of MySQL makes its performance superior at higher volumes.

Lessons Learned

1. BigQuery built in a different way than many other systems. Though an excellent white paper explaining the Dremel query system can be found here: https://static.googleusercontent.com/media/research.google.com/en//pubs/archive/36632.pdf, there is not much other technical data and implementation details vs. what can be found for MySQL (ex. Join algorithms, buffer pool information). So, we designed the experiment to isolate some of these vague areas as variables to make it easier to identify performance differences.