NOSQL DATABASE PERFORMANCE TESTING

Jayashree Panchapakesan
Agenda

- Why Database?
- What is SQL
- What is NOSQL?
- Characteristics of NOSQL and SQL
- Why NOSQL?
- How is NOSQL performance testing different?
- Research Papers on NOSQL database performance testing
- Issues in NOSQL
- Overall Conclusion
What is SQL?

- A Structured Query language for RDBMS
- Enables unrestrained queries against normalized data
- Geared towards joins, filters, and aggregations.

Why Database?

- Database is important part of Big data
- Traditional Database RDBMS - SQL
- Big data - NOSQL.
What is NOSQL?

From www.nosql-database.org:

- Stands for Not Only SQL
- Next Generation Databases mostly addressing some of the points: being non-relational, distributed, open-source and horizontal scalable. The original intention has been modern web-scale databases. The movement began early 2009 and is growing rapidly. Often more characteristics apply as: schema-free, easy replication support, simple API, eventually consistent / BASE (not ACID), a huge data amount, and more.
- The NOSQL concept tree illustrates the variety of concepts related to NOSQL.
- Usually do no use require a fixed table schema nor they use the concept of joins
# RDBMS vs NOSQL

<table>
<thead>
<tr>
<th>SQL Database</th>
<th>Non-SQL Database</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Relational</strong></td>
<td><strong>Key-Value</strong></td>
</tr>
<tr>
<td><img src="image" alt="Relational Diagram" /></td>
<td><img src="image" alt="Key-Value Diagram" /></td>
</tr>
<tr>
<td><strong>Analitics (OLAP)</strong></td>
<td><strong>Column-Family</strong></td>
</tr>
<tr>
<td><img src="image" alt="Analitics Diagram" /></td>
<td><img src="image" alt="Column-Family Diagram" /></td>
</tr>
<tr>
<td><strong>Graph</strong></td>
<td><strong>Document</strong></td>
</tr>
</tbody>
</table>

**RDBMS**
- Looks at parts
- Structured
- Relational
- Consistent
- Rigid
- Mature
- Stable

**NoSQL**
- Looks at wholes
- Semi-structured
- Object-oriented
- Eventually Consistent
- Flexible
- Emerging
- Scalable
ACID Transactions in SQL

- **Main Focus in ACID**
  - **Atomicity** - Each transaction is atomic. If one part fails, the entire transaction fails.
  - **Consistency** - Every transaction is bound to have consistent set of rules (triggers, constraints, cascades)
  - **Isolation** - No transaction should interfere with another transaction.
  - **Durability** - Once a transaction is committed, it remains committed

BASE Transactions in NOSQL

- **It is considered to be opposite to the ACID**
  - **Basically Available**
  - **Soft State**
  - **Eventually Consistent**

Characteristics

- Availability first
- Simpler and faster
- Weak Consistency - stale data OK
- Aggressive and Approximate

*(Gudivada, V., Dhana Rao, and Vijay Raghavan)*
CAP Theorem

- Three properties of a system: consistency, availability and partitions

- You can have at most two of these three properties for any shared-data system

- To scale out, you have to partition. That leaves either consistency or availability to choose from

  - In almost all cases, you would choose availability over consistency

(Gudivada, V., Dhana Rao, and Vijay Raghavan)
Why NOSQL?

- Fits well for Volatile data.
- It’s faster than SQL in execution and other operations.
- Scales very well.
- High read and write throughput
Why is NOSQL performance Testing Different than SQL testing

NOSQL has different database types:

- Column Store
- Key Value store
- Document Store
- Graph store
Column Store

- Each storage block contains data from only one column and its more efficient than row
- Multiple row/record/documents are inserted at the same time so updates of column blocks can be aggregated
- Databases: Hadoop, Cassandra


- **Document Store/XML**
  - A key that points to serialized object.
  - DB can operate against values in document
  - Databases: MongoDB, CouchDB,

*(Gudivada, V., Dhana Rao, and Vijay Raghavan)*
- **Key-Value store**
  - Hash tables of Keys
  - Values stored with Keys
  - Fast access to small data values
  - Example – MemCacheDB
    http://memcachedb.org/

- **Graph Store**
  - Focus on modeling the structure of data-interconnectivity
  - Scales the complexity of the data
  - Example – Neo4j, Allegro graph

*(Gudivada, V., Dhana Rao, and Vijay Raghavan)*
- **Scaling**
  NOSQL does lend itself to massively parallel processing.

- **Modeling**
  NOSQL does not require pre defined model.
NOSQL Databases

- **Mongo DB**: [https://www.mongodb.com/](https://www.mongodb.com/)
- **HBase**: [https://hbase.apache.org/](https://hbase.apache.org/)
- **Couch DB**: [http://couchdb.apache.org/](http://couchdb.apache.org/)
- **Elastic Search**: [https://www.elastic.co/](https://www.elastic.co/)
## Companies involved in Big Data

<table>
<thead>
<tr>
<th>Company Name</th>
<th>NoSQL Name</th>
<th>NoSQL Storage Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adobe</td>
<td>HBase</td>
<td>Column</td>
</tr>
<tr>
<td>Amazon</td>
<td>Dynamo</td>
<td>Key-Value</td>
</tr>
<tr>
<td>BestBuy</td>
<td>Riak</td>
<td>Key-Value</td>
</tr>
<tr>
<td>eBay</td>
<td>Cassandra</td>
<td>Column</td>
</tr>
<tr>
<td>Facebook</td>
<td>Cassandra</td>
<td>Neo4j</td>
</tr>
<tr>
<td>Google</td>
<td>BigTable</td>
<td>Column</td>
</tr>
<tr>
<td>LinkedIn</td>
<td>Voldemort</td>
<td>Key-Value</td>
</tr>
<tr>
<td>LotsOfWords</td>
<td>CouchDB</td>
<td>Document</td>
</tr>
<tr>
<td>MongoDB</td>
<td>MongoDB</td>
<td>Document</td>
</tr>
<tr>
<td>Mozilla</td>
<td>HBase</td>
<td>Riak</td>
</tr>
<tr>
<td>Netflix</td>
<td>SimpleDB</td>
<td>HBase</td>
</tr>
<tr>
<td>Twitter</td>
<td>Cassandra</td>
<td>Column</td>
</tr>
</tbody>
</table>
Research Papers on NOSQL and few approaches followed to perform the Testing
NOSQL database Performance Testing

- The performance testing was conducted for geospatial data
- The usability in NOSQL (MongoDB, CouchBase) and SQL (PostgreSQL) and comparing their results for their performance.

**Experimental setup:**
- 10GB RAM, 8CPU 2.5 GHz, Microsoft windows server 2008 R2
- The data for NOSQL is document stored data
- Format of data will be JSON, YAML or XML
- The queries used were the typical attribute of object and calculate the spatial data
- Apache Jmeter used for processing the requests.

(Schmid, Stephan, Eszter Galicz, and Wolfgang Reinhardt)
Conclusion:

- PostgreSQL response time increases with the size of the dataset where as MongoDB and Couch DB is faster and takes few seconds.

- For small database the PostgreSQL performs better.

- MongoDB keeps the performance even with large dataset.

- NOSQL is superior and fast compared to RDBMS.

(Schmid, Stephan, Eszter Galicz, and Wolfgang Reinhardt)
• The performance testing was conducted for different database for NOSQL for storing and retrieving data.

• They tested 10 database with yahoo cloud serving benchmark.

• To check how to performance is affected by non relational database.

• They tested the performance of database based on read, update operation execution time.

• Each test scenario is called workload and defined by certain features like % of read and update, total number of transaction, number of records used.

• There are totally 8 workloads.

(Abramova, Veronika, Jorge Bernardino, and Pedro Furtado)
Experimental setup:

• Ubuntu server 32bit and 2GB RAM in windows 7

• Yahoo cloud serving benchmark.

• Key value store, document store, Column store were the data format.

• Cassandra, Hbase, Oracle NoSQL, Redis, Scalaris, Tarantool, Voldemort, Elasticsearch, MongoDB, Orient DB

(Abramova, Veronika, Jorge Bernardino, and Pedro Furtado)
Conclusion.

Data Loading:
- Comparing the results of key, column, document
- Column store – Hbase was faster than Cassandra
- Document store- MongoDB was fastest than elastic search
- Key Value- Redis and Tarantool failed to meet the expectations.

Overall Execution Time:
- The execution time increased with the increased use of disk operations by database
- Lowest time was taken by Tarantool
- Highest time taken by Orient DB

(Abramova, Veronika, Jorge Bernardino, and Pedro Furtado)
In this approach NOSQL Mongo DB is compared to standard SQL.

In terms of runtime, of these two database for a modest sized structure database.

**Experimental setup.**

- Mongo DB and Microsoft SQL server
- Intel i7 quad core , 3.4GHz processor with 8GB of DDR3 RAM
- Four separate tests consisted off 100 runs for each test.
- It tested 3 types of updates, 2 types of select, Insert ,simple queries, aggregating queries

(Parker, Zachary, Scott Poe, and Susan V. Vrbsky)
Conclusion:

- Mongo DB has better runtime performance for inserts, updates and simple queries than SQL

- SQL has better performance for aggregate functions as Mongo DB required additional effort in implementation.

(Parker, Zachary, Scott Poe, and Susan V. Vrbsky)
Issues with NOSQL database

- No common standards. Each database does things differently.
- Querying data does not involve familiar SQL model to find records.
- NoSQL databases are relatively immature and constantly evolving.
- NoSQL database avoids ACID (Atomicity, Consistency, Isolation, Durability) model, there is no guarantee that all of the data will be successfully written.
Overall Conclusions

- NoSQL database fits exactly your specific needs.
- NoSQL database could be a very simple one in terms of architecture.
- It is easier to find a cluster database among NoSQL ones rather than among SQL ones.
Thank You