CMPSC 441
Distributed Systems

Dr. Aravind Mohan
Allegheny College

March 6, 2018
Lecture 08 - Introduction to NoSQL
What is NoSQL?

- A NoSQL database environment is, simply put, a non-relational and largely distributed database system that enables rapid, ad-hoc organization and analysis of extremely high-volume, disparate data types.

- NoSQL databases are sometimes referred to as cloud databases, non-relational databases, Big Data databases and a myriad of other terms and were developed in response to the sheer volume of data being generated, stored and analyzed by modern users (user-generated data) and their applications (machine-generated data).
Taxonomy of NoSQL

- Key-value
  - Redis
  - Riak

- Graph database
  - Neo4j
  - HyperGraphDB

- Document-oriented
  - MongoDB
  - CouchDB

- Column family
  - Cassandra
  - HBase
NoSQL Features

- Next Generation Databases
- Distributed Architecture
- Not Only SQL
- Open Source
- Non-Relational
- Horizontally Scalable
NoSQL Features (contd)

- **Schema – Free!**
  - Can manage huge amount of data

- **Easy – Replication**
  - Can be implemented on commodity hardware's

- **Simple API**
  - ~150 NoSQL databases are there in the market

- **Request**
  - Your APP

- **DATA**
  - Their APP
Why NoSQL?
### NoSQL Vs SQL Comparison

<table>
<thead>
<tr>
<th>Entity</th>
<th>SQL Databases</th>
<th>NoSQL Databases</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>One Type (SQL) with Minor Variation</td>
<td>Many Types (Document, Key-Value, Tabular, Graph)</td>
</tr>
<tr>
<td>Development</td>
<td>1970</td>
<td>2000</td>
</tr>
<tr>
<td>Examples</td>
<td>Oracle, MSSQL, DB2 etc.</td>
<td>MongoDB, Cassandra, Hbase, Neo4J</td>
</tr>
<tr>
<td>Schemas</td>
<td>Fixed</td>
<td>Dynamic</td>
</tr>
<tr>
<td>Scaling</td>
<td>Vertical</td>
<td>Horizontal</td>
</tr>
<tr>
<td>Dev Model</td>
<td>Mix</td>
<td>Open Source</td>
</tr>
<tr>
<td>Consistency</td>
<td>Follow ACID</td>
<td>Follow BASE</td>
</tr>
</tbody>
</table>
How does NoSQL vary from RDBMS?

- Looser schema definition
- Applications written to deal with specific documents/data
  - Applications aware of the schema definition as opposed to the data
- Designed to handle distributed, large databases
- Trade offs:
  - No strong support for ad hoc queries but designed for speed and growth of database Query language through the API
  - Relaxation of the ACID properties
Benefits of NoSQL

Flexible data models

- Change management to schema for RDMS have to be carefully managed
- NoSQL databases more relaxed in structure of data
  - Database schema changes do not have to be managed as one complicated change unit
  - Application already written to address an amorphous schema
Benefits of NoSQL (contd)

Economics

- RDMS rely on expensive proprietary servers to manage data
- No SQL: clusters of cheap commodity servers to manage the data and transaction volumes
- Cost per gigabyte or transaction/second for NoSQL can be lower than the cost for a RDBMS
Benefits of NoSQL (contd)

- Large volumes of Structured, Semi-structured and Unstructured Data

![Diagram of Big Data with Volume, Velocity, and Variety](image)

- **Volume**: Petabytes per day/week
- **Velocity**: Real-time capture and Real-time analytics
- **Variety**: Unstructured data, web logs, audio, video, image
Benefits of NoSQL (contd)

- Agile development, quick changes and frequent code pushes
Drawbacks of NoSQL

Support

- RDBMS vendors provide a high level of support to clients
  - Stellar reputation
- NoSQL - are open source projects with startups supporting them
  - Reputation not yet established
Drawbacks of NoSQL (contd)

Maturity

- RDMS mature product: means stable and dependable
  - Also means old no longer cutting edge nor interesting
- NoSQL are still implementing their basic feature set
Drawbacks of NoSQL (contd)

Administration
- RDMS administrator well defined role
- No SQL’s goal: no administrator necessary however NO SQL still requires effort to maintain

Lack of Expertise
- Whole workforce of trained and seasoned RDMS developers
- Still recruiting developers to the NoSQL camp
Drawbacks of NoSQL (contd)

Analytics of business
- RDMS designed to address this niche
- NoSQL designed to meet the needs of an Web 2.0 application - not designed for ad hoc query of the data. Tools are being developed to address this need
SQL Database ACID Properties

A - Atomicity
   All operation in a transaction succeed or every operation is rolled back

C - Consistency
   Transactions never observe or result in inconsistent data

I - Isolation
   Transactions do not contend with each other, each transaction behaves as if it is the only operation being performed

D - Durability
   Upon completion of the transaction, the operation is not be reversed (i.e. committed transactions are permanent)
CAP Theorem

It is impossible for a distributed computer system to simultaneously provide all three of the following guarantees:

- Consistency (all nodes see the same data at the same time)
- Availability (a guarantee that every request/query receives a response about whether it succeeded or failed)
- Partition tolerance (the system continues to operate despite arbitrary partitioning due to network failures)

A distributed system can satisfy any two of these guarantees at the same time, but not all three.
CAP Theorem (contd)

- **CP = NoSQL/Column**
  - Hadoop
  - Big Table
  - HBase
  - MemCacheDB (Graph?)

- **CA = SQL/RDBMS**
  - SQL Server/SQL Azure
  - Oracle
  - MySQL

- **AP = NoSQL/Document or Key/Value**
  - DynamoDB
  - CouchDB
  - Cassandra
  - Voldemort

Consistency

Partitioning

Availability
Class Activity (Project Discussion)

Discuss with your team regarding your project. Some questions, you should keep in mind while writing your proposal is:

The course project counts towards 30% of your final grade for this course. You should choose a good project, which has some novelty to it.

1. Track 01 - What is the data that you are collecting? How much is the expected data collection going to be?
2. Track 01 - What analytics are you going to perform on the data? Does this analytics create any impact to the common man in the society?
3. Track 01 - Do you need the Cloud and Distributed System to develop your project? Why can’t you do your project in just one machine.
Class Activity (Project Discussion)

1. Track 02 - Why is the scheduling problem so important to solve? Why do you need to solve the scheduling problem in a Cloud based framework?

2. Track 02 - What existing scheduling algorithms do you plan to implement? What are the pros and cons of those algorithms?

3. Track 02 - Do you plan to invent a new scheduling algorithm? If so what is your idea, Why do think your idea is so innovative?
Questions