Cloud Scale Distributed Data Storage

Jürmo Mehine,
Pelle Jakovits

19 October 2016
Outline

• **Background**
  • Relational model
  • Database scaling
  • Keys, values and aggregates
  • The NoSQL landscape

• **Non-relational data models**
  • Key-value
  • Document-oriented
  • Column family
  • Graph

• **Five databases**
  • Riak
  • MongoDB
  • CouchDB
  • HBase
  • Cassandra
The Relational Model

- Data is stored in tables
- Tables can have strict relationship between each other
  - Defined as foreign key references between rows
- The expected format of the data is specified with a schema – which is very restrictive
- Data is accessed using Structured Query Language (SQL)
- ACID is a set of properties of database transactions.
  - Atomicity - All or nothing
  - Consistency - Transactions can not break consistency
  - Isolation - Transactions are isolated from each other
  - Durability - Commited transaction should survive failure
Database Scaling

• Vertical scaling – on one machine
• Horizontal scaling – across a cluster of machines
• Relational model does not scale well horizontally
  • Because there are too many dependancies in relational model
  • Database sharding is one approach
Keys, Values and Aggregates

- Non-relational data models are based on a simpler key-value structure.
- An aggregate is a collection of data that is treated as a unit.
  - E.g. a customer and all of his orders.
- In normalized relational databases aggregates are computed using GroupBy operations.
- Key-value, document-oriented and column family are said to be aggregate-oriented data models because they store aggregated data together.
- Keyed aggregates make for a natural unit of data sharding.
Benefits of the Key-value Model

- Horizontal scalability
  - Data with the same Key stored close to each other.
- Suitable for cloud computing
- Higher access speeds
- Flexible schemaless models suitable for unstructured data
- More suitable for Object Oriented languages
The NoSQL Movement

- NoSQL is a broad term with no clear boundaries
  - The term NoSQL itself is very misleading
- Emergence of persistence solutions using non-relational data models
- Schemaless data models
- Driven partly by the rise of cloud computing
- Simpler key-value based data models scale better horizontally than the relational model
- Tradeoff between data consistency and high availability
CAP Theorem

• It is impossible for a distributed computer system to simultaneously provide all three of the following:
  • **Consistency** - every read receives the most recent write or an error
  • **Availability** - every request receives a response
  • **Partition tolerance** - the system continues to operate despite arbitrary partition failures

• Usually the choice is between **consistency** or **availability**
• Relational databases choose **consistency**
• NoSQL solutions are more focused on **availability**
• When trying to aim for both **consistency** or **availability**
• Have to choose between latency and throughput (PACELC theorem)
The NoSQL Landscape

- Distributed key-value based data stores
- Embedded non-relational data stores (LevelDB, Tokyo Cabinet)
- In-memory data stores (Redis, Memcached)
- Hosted services (Amazon S3, DynamoDB)
- Data analysis and processing platforms/tools (Hadoop/HDFS, Pig, Hive)
- Object databases (ZODB, GemStone)
- NewSQL (Drizzle, SQLFire)
Four Non-relational Data Models
The Key-value Model

- Data stored as key-value pairs
- The value is an opaque blob to the database
- Examples: Dynamo, Riak
The Document-oriented Model

• Data stored as key-value pairs
• Values have further structure in the form of fields and values
• Examples: CouchDB, MongoDB
Example

- Aggregates described in JSON using map and array data structures
The Column Family Model

- Data stored in large tabular structures (sparse matrix)
- Columns grouped into column families
  - Column family is a meaningful group of columns
  - Similar concept as a table in relational database
- A record or row can be thought of as a two-level map
- Columns can be added at any time
- Solutions using it: BigTable, Cassandra, HBase

http://www.datastax.com/docs/1.1/ddl/column_family
Example

<table>
<thead>
<tr>
<th></th>
<th>names</th>
<th>contacts</th>
<th>messages</th>
</tr>
</thead>
<tbody>
<tr>
<td>_id</td>
<td>username</td>
<td>firstname</td>
<td>lastname</td>
</tr>
<tr>
<td>a001</td>
<td>jsmith01</td>
<td>John</td>
<td>Smith</td>
</tr>
<tr>
<td>b014</td>
<td>pauljones</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Aggregates as rows in a column family store
Indexing and Partitioning

- In Key-value model, Key acts as an index
- Secondary indexes can be created in some solutions.
- Implementations vary from partitioning to local distributed indexes.
- Data is partitioned between different locations
- It is basically sharding
- Usually data is partitioned by rows or/and column families
- Often users can specify partition key
- Gives control over how data is distributed
- Very important for optimising query speed
Graph Databases

• Data stored as nodes and edges of a graph
• The nodes and edges can have fields and values
• Aimed at graph traversal queries in connected data
• Examples: Neo4J, FlockDB

Non-relational Data Models

- In non-relational data stores data is denormalized
- It is common to also store JSON documents in key-value stores
- The key-value, document-oriented and column family models are aggregate oriented models
- The other models are based on the key-value model
- In reality the classification of databases into different models is not as straightforward
- Multiparadigm databases also exist (e.g. ArangoDB)
Five Databases
Riak

- Key-value model
- Based on Amazon's Dynamo specification
- Distributed decentralized persistent hash table
- Consistent Hashing
- Eventual Consistency
- Multiversion concurrency control (MVCC) using dotted version vectors

http://basho.com/products/riak-overview/
Riak

- RESTful query interface
  - Basic PUT, GET, POST, and DELETE functions
- Links and link walking
  - One way relationships between data objects
  - Turns Key-Value store into a simple Graph database
- Riak search is based on Apache Solr search engine
- MapReduce in Erlang and JavaScript
MongoDB

- Document oriented (BSON)
- Query language based on JavaScript
- GridFS for BLOB file storage
- Master-slave architecture
- Linking between documents
- Supports MapReduce in JavaScript

http://www.mongodb.org/
CouchDB

- Document oriented (JSON)
- RESTful query interface
- Built in web server
- Web based query front-end Futon

http://docs.couchdb.org/en/latest/
CouchDB

- MapReduce is the primary query method (JavaScript and Erlang)
- Materialized views as results of incremental MapReduce jobs
- CouchApps – javascript-heavy web applications built entirely on top of CouchDB without a separate web server for the logic layer
HBase

- Column family
- Master-slave architecture
- Built on top of Hadoop
- Data stored in HDFS
- A single table with several column families
- Supports Java MapReduce

http://hbase.apache.org/
Cassandra

- Column family model
- Data model from BigTable, distribution model from Dynamo (decentralized)
- Uses Cassandra Query Language (CQL)
  - SQL-like querying language

http://cassandra.apache.org/
Cassandra

- Provides AP from the CAP theorem
- Availability & Partition-Tolerance
- Static and dynamic column families
- Dynamic column families as materialized views on data
- The concept of super-columns
  - Family of column families
Polyglot Persistence

http://martinfowler.com/bliki/PolyglotPersistence.html
Conclusions

- In recent years there has been a rise in non-relational (NoSQL) data stores
- This is related to the rise of cloud computing – key-value based models offer better horizontal scalability across a cluster of computers
- The NoSQL landscape is extremely varied
- The four basic non-relational data models are:
  - The key-value model
  - The document-oriented model
  - The column family model
  - Graph databases
That is All

- Next week's practice session
  - Using NoSQL databases
- Exam times
  - Wednesday - 26 October 2016
  - Wednesday - 02 November 2016