Scaling Cassandra Database under Heavy User Loads

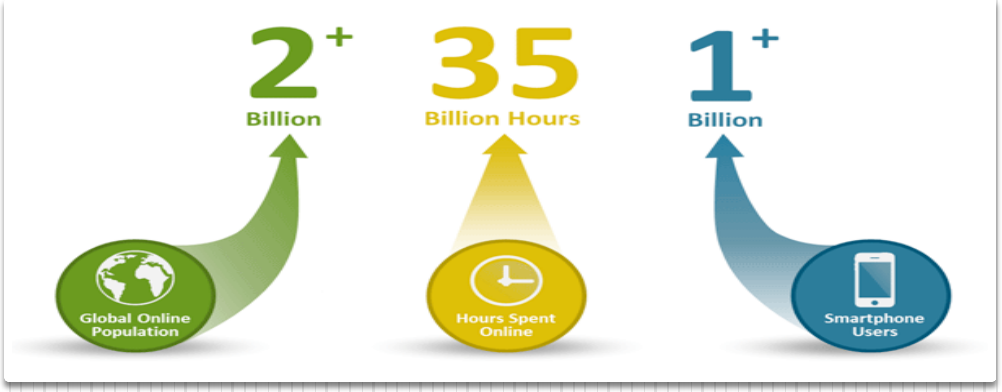
SYSC 5703
Integrated Database Systems

Nikhil Nayyar (100 941 327)
nikhil.nayyar@carleton.ca

Ferhan Jamal (100 953 487)
ferhan.jamal@carleton.ca

Introduction

- Why NoSQL?
  - Big Data
  - Cloud Computing



- What is Cassandra?
  - Distributed…High Performance…Fault Tolerant (i.e. no single point of failure)…non relational DBMS
  - Data model inspired by Google Bigtable
  - Distribution model and replication mechanism inspired by Amazon Dynamo
  - Open source by Facebook in 2008
  - Highly Scalable
    - Supports linear scalability
  - Used by Facebook,Twitter,eBay,Netflix,Cisco and many others

Cassandra Architecture

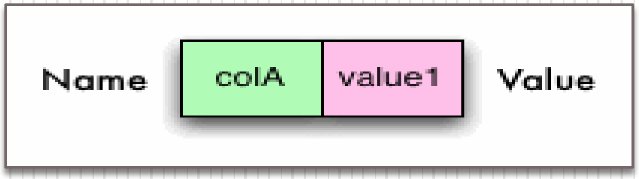
- Cassandra uses a ring or cluster architecture
- Peer to Peer distributed system(master-less)
  - Uses Gossip protocol to maintain and keep in synch of list of dead and alive nodes
- No single point of failure
- Data partitioned among all the nodes in the cluster
- Custom data replication to ensure fault tolerance
- Read/Write anywhere design
  - A commit log is used on each node to capture write activity to ensure data durability
  - Data is also written in-memory structure (memtable) and then written to a disk (SS table)
- Tunable Consistency(Brewer CAP theorem)
  - Cassandra focuses on high availability and partition tolerance with the option of eventual consistency

Data Model

- Schema free
  - No need to define the structure of data in advance
  - Does not support relationships between column families by using joins
- Row oriented column structure
- A keyspace is similar to a database in the RDBMS world
- A column family is similar to a RDBMS table but is more flexible and dynamic
- A column family is a set of key value pairs i.e. an ordered collection of columns
- Each column consist of a name, a value and a timestamp

Data Model ..cont'd

A single column

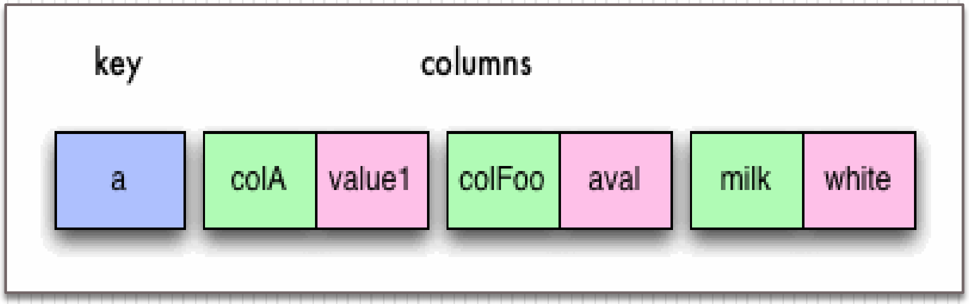


Name

colA | value1

Value

A single row



key

columns

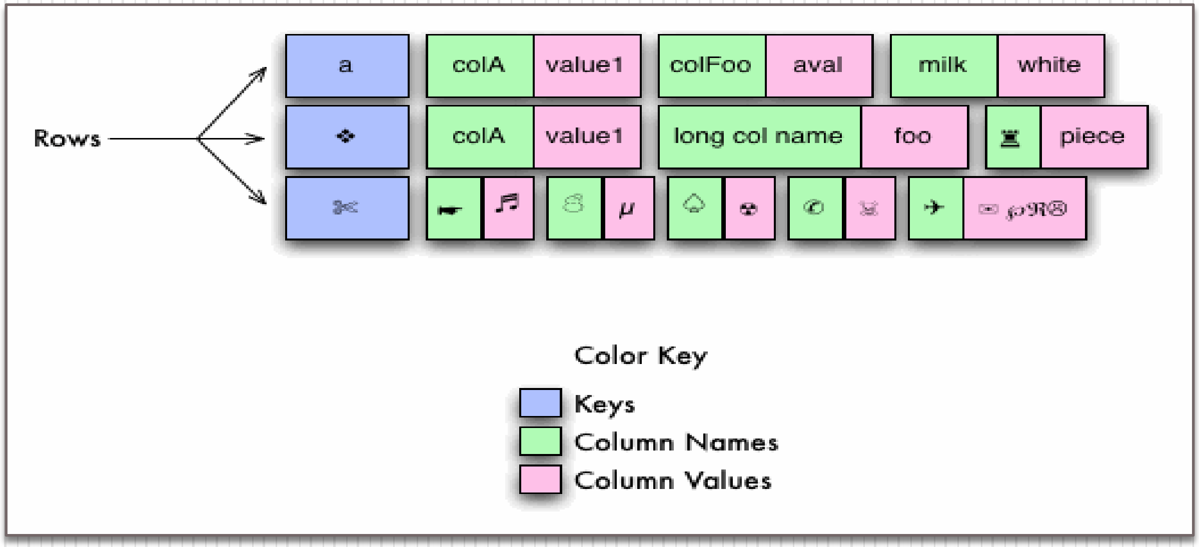
a

colA | value1

colFoo | aval

milk | white

Column family



Rows

a — colA | value1 — colFoo | aval — milk | white

❖ — colA | value1 — long col name | foo — ♜ | piece

✂ — ☛ | ♬ — ☃ | μ — ♤ | ☢ — ✆ | ☠ — ✈ | ✉ ℘ℜ®

Color Key

🟦 Keys

🟩 Column Names

🟪 Column Values

Goals

- Scaling Cassandra database under heavy user loads
- Host Cassandra cluster on few machines
- Wire on the traffic to Cassandra cluster
- Setup OPSCENTER with our existing Cassandra cluster
- Monitor Cassandra cluster through OPSCENTER
- Horizontally Scaling Cassandra database

Literature Review

- How to do scalability in Cassandra when there are more read and write request
- Write scaling
- How to ensure no single point of failure in Cassandra without compromising with performance
- Setting up few nodes of Cassandra cluster and generate large amount of traffic against it
- Monitor how Cassandra is be having under high user loads

Work Done so far…

- Nikhil Nayyar
  - Setup Single node Cassandra Cluster
  - Created Data Model by using CQL
- Ferhan Jamal
  - Write a Single threaded Java program by using Datastax Java driver to wire on the traffic against Cassandra database
  - Run the program for couple of hours with some load
  - Setup OPSCENTER for Cassandra to see how Cassandra is behaving

- Next Steps
  - Setup multinode Cassandra cluster
  - Make the program multithreaded
  - Single node cluster vs multinode cluster performance

Challenges

- Lot of libraries are available for Cassandra
  - Netflix Astyanax Client
  - Datastax Java driver
  - Pelops Client
  - Hector Client

- CQL vs Thrift
- Setting up single node Cassandra cluster

Conclusion

- Cassandra is based on sound design principles
- Data model is incredibly powerful specially CQL
- Very easy to setup
- Datastax Batch Writes/Asynchronous feature is pretty powerful
- How easily Cassandra can be scale-out as the throughput keeps on increasing without impacting the client application directly

References

- The Little Engine(s) That Could: Scaling OnlineSocial Networks: Josep M. Pujol, Vijay Erramilli, Member, IEEE, GeorgosSiganos, Xiaoyuan Yang,NikolaosLaoutaris, Member, IEEE, ParminderChhabra, and Pablo Rodriguez, Member, IEEE.

- A request skew aware heterogeneous distributed storage system based on Cassandra : Zhen Ye, Shanping Li

- Issues in Handling Complex Data Structures with NoSQLdatabases : Santo Lombardo, Elisabetta Di Nitto and DaniloArdagna

- A Study Into the Capabilities of NoSQLDatabases in Handling a Highly Heterogeneous Tree:DileepaJayathilake, CharithSooriaarachchi, ThilokGunawardena, BuddhikaKulasuriya and ThusithaDayaratne

- Cattell, Rick. "Scalable SQL and NoSQL data stores." ACM SIGMOD Record39.4 (2011): 12-27.

Questions?