# Scholarly Response: Riak – A key value noSQL database

**Rohan Purohit**

# College of Computer & Information Sciences

# Regis University

1. **Riak is a Key/value based database that doesn't lend itself to relational normalization well. Investigate and define bucket, bucket types, and key namespaces and the different ways to define them in a Riak database that lends itself to performant accesses of the database.**

Riak is key-value noSQL database that uses a simple hash table to maintain data in key value pairs. In comparison to a relational database instance a Riak database is composed of a cluster of nodes. These nodes can be each started individually from the Riak console via the command dev1/bin/riak start .Once the nodes are up and running in the cluster ring we can create data structures known as buckets and fill them with key value pairs.

There are three ways in which we can access Riak databases- via a third party programming language, erlang console or web browser. As we input data using curl commands into Riak we can specify a location URL’s along with a bucket and key name such as curl –v –X PUT url/bucket/key -H “Content-Type: application/json” \ -d ‘{“nickname” : “The Wonder Dog”,”breed “ : “German Shepard”}’

Once a write is successful we get a 2xx HTTP response code back from the cluster in a manner that modern API’s affirm payloads. This http friendly characteristic makes Riak the most web compliant database. Once value of a particular content type has been written to the bucket and key grouping of interest via a PUT call we can do a GET call to retrieve stored key value pairs.

Riak allows us to store values of any types which includes data as arrays, documents, and images. Different data structures and file formats can be stored within the value object of Riak which is basically encoded as a binary large object. (Sadalage, p.84) Riak data store allows modern computing ventures to store images, documents, videos in aggregations of namespaced buckets. These specialized groupings via the namespace allow make Riak especially suitable for analysis that requires efficient storing of session and profile data.

In Riak, we also have the concept of links. A link is metadata that determines relationships between different keys. (Redmond, p. 57) We can issue a link by specifying the –H “Link: </riak/bucket/key>; riaktag= \”contains\””\ -d ‘{“room”:101}’ command. Once a link is established we can lookup links via the curl command curl –I url/port/riak/bucket/key.

Riak also comes with several functions that a user can invoke on its data sets. The map reduce function in Riak allows us to map inputs to a reducer function of choice which sends the reduce request to nodes in parallel. Instead of grabbing data from database and running it on app server or client, the map reduce pattern passes the algorithm to the database nodes, which then each compute the algorithm to get the desired result. According to Redmond, each object on the server is mapped to some common key that groups the data together, and then all matching keys are reduced into some single value. (Redmond, p.64) In Riak it is faster to send the algorithm to the data then to send the data to the algorithm. (Redmond, p.65)

Riak is an AP database meaning that is highly available and partitioned. Riak server architecture manages single points of failures via a process known as hinted handoffs. Failing nodes are removed from the ring and updated by neighbor nodes. However Riak is eventually consistent meaning that data in all of its nodes is not the same at any given point of time. The masterless architecture within Riak is balanced via the formula N-W=R where N determines the replication factor, W determines the nodes required to mark a successful write and R determines the nodes required to mark a successful read operation. Riak server configuration is divided into partitions, there are 64 partitions by default. If you have a three node cluster each node gets 21 partitions. Each partition is also known as a virtual node.

Riaks strengths lie in its high availability, horizontal scaling and its focus in removing single points of failure. Riak has weaknesses where we need simple queryablity, complex data structures or rigid schemas. Riak is highly customizable in that CAP can be changed on a per bucket, per request basis. Ultimately Riak succeeds when we want to scale out and serve a lot of data on the Web. (Redmond, 92)

**References**

**Fowler, M & Sadalage, P. (2013).NoSQL Distilled: A brief guide to the Emerging World of Polyglot Persistence. Addison-Wesley. NJ Upper saddle.**

**Redmond, E & Wilson, J.R. (2012).Seven Databases in Seven Weeks: A Guide to Modern Databases and the NoSQL Movement. Pragmatic Programmers. Dallas Texas.**

**Basho Technical Writers. (2015) NoSQL databases explained. Retrieved Sept 25, 2016 from http://basho.com/resources/nosql-databases/**

**Basho Technical Writers. (2015) Relational to Riak. Retrieved Sept 25, 2016 from**

**http://info.basho.com/rs/721-DGT-611/images/Relational%20to%20RiakKV.PDF**

**Basho Technical Writers. (2015) Riak KV: Meeting Challenges of Unstructured Data. Retrieved Sept 25, 2016 from http://info.basho.com/rs/721-DGT 611/images/Challenges%20of%20Unstructured%20Data.PDF**