Riak

-Priyanka Goyal

What is Riak?

- Distributed Key-value database
  - bucket
    - key-value
      - key
      - value
- Can store plain text, JSON, XML, images, videos
- Fault-tolerant
- Simple REST API
- Best suitable for Amazon

http://en.wikipedia.org/wiki/File:Riak_distributed_NoSQL_key-value_data_store_logo.png

Features

- **SCALING:** Value of key defines on which node key is stored

- **CONSISTENCY:** Eventual but how to resolve update conflicts?
  - 2 ways
    - newest write wins, older write loose
    - return both values to client and get it resolved

- **TRANSACTIONS:** Uses Quorum concept

Quorums - The Riak Way!

- n = The number of Replicas (Replication Factor)
  - varies per bucket
- r = The number of Replicas needed for successful read
  - varies per operation
- w = The number of Replicas needed for successful write
  - varies per operation

Quorums - The Riak Way! (Contd..)

n-r = read fault tolerance

n-w = write fault tolerance

Example: 6-node Riak cluster, n=4,r=1, w=2

- 4-1 = 3 hosts can be down but Riak will still perform read operation
- 4-2 = 2 hosts can be down but Riak will still perform write operation

Map/Reduce

Map() : Convert list of data into another type of list

Reduce() : convert the new list to one/more scalar values

The Riak Way!

Each object on the server is "mapped" to a common key that groups data together and then all matching keys are "reduced" into a single value.

Erlang Map Phase

```
map_object_value(obj, _keyData, _Arg)->[riak_object:
                    get_value(obj)]
```

- ➔ obj: Riak object retrieved from bucket/key
- ➔ keyData: Static argument specified with the bucket/key
- ➔ Arg: Static argument specified with the job

Built-in Functions:

- `riak_mapreduce:map_object_value/3`
- `riak_mapreduce:map_object_value_list/3`

Erlang Reduce Phase

Built-in Functions:

- `riak_mapreduce:reduce_set_union/2`
- `riak_mapreduce:reduce_sum/2`
- `riak_mapreduce:reduce_sort/2`

➔ Riak explores MapReduce using REST API
➔ Riak submits job via POST
➔ Default URL is `/mapred`

Linking Objects & Link Walking

- Object links (Link Header) store pointers/links to other objects

```
</riak/demo/test1>;riaktag=”userinfo”
```

- Link Walking - Collect object along the link and return

Examples:

```
/riak/demo/test1/_,_,1
/riak/demo/test1/demo,_,1
/riak/demo/test1/_,_,0/_,_,1
/riak/demo/test1/_,child,0/_,_,1
```

When to use

- Storing Session Information
  - web-sessions with their unique sessionId. Use PUT/GET
- User Profiles, Preferences
- Shopping Cart Data
  - e-commerce websites

When not to use

- Relationship among Data
  - correlate data b/w different sets of keys
- Multiops Transactions
  - multiple keys with failure to save even one key
- Query by Data
  - search keys based on values
- Operations on multiple keys

Wrap up!

riak

...flexible storage engine...

...with REST API...

...and Map/Reduce capability...

...designed to be fault-tolerant...

...distributed...

...operations friendly...