MONTANA
STATE UNIVERSITY

Redis

●●●

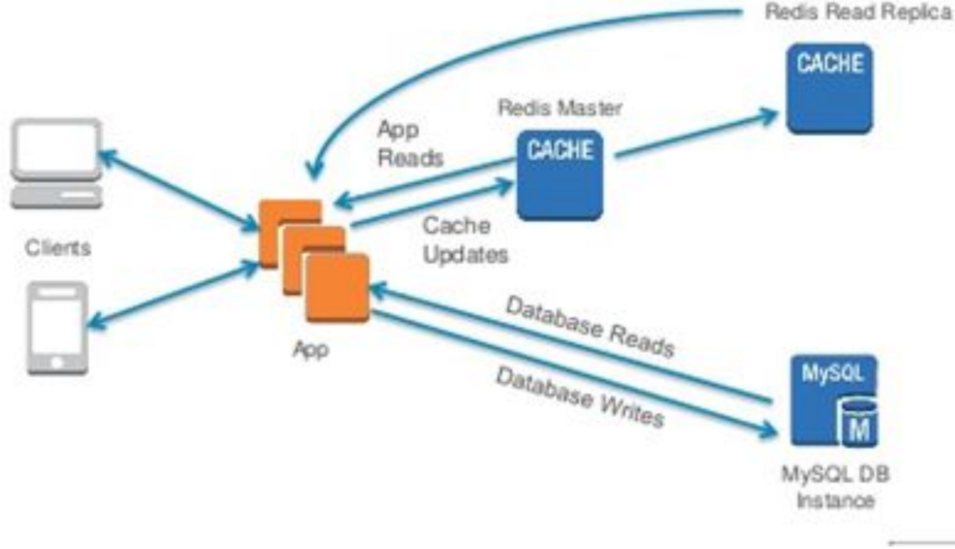
Beyond DBMS

Redis

- Redis is a *NoSQL* data store
- *Re*mote *Di*ctionary *S*ervice
- Core concept is that of key-value pairs
  - Not unlike a giant hash table
- *Widely* used in industry
  - Almost every major website you use has Redis somewhere

Redis

- Typically used as a *caching layer* for applications
- Here is a common cloud architecture layout
  - Application servers serve requests from users
    - First consult redis cache
    - Next, consult DBMS



Redis History

- Redis was created by Salvatore Sanfilippo, an italian software developer
  - AKA *antirez*
- Salvatore was having trouble scaling his startup with traditional DBMS systems
- Initially written in TCL, later ported to C

Redis History

- Open sourced & Announced on HackerNews in 2009
- Quickly adopted by startups
  - Github
  - Instagram
- Now the 4th most popular data store in the world
  - !!!

Redis

- Popularized the idea of keeping all data in memory
    - Data can be written to disk, but only for reconstructing in-memory state
    - Initially, no ACID guarantees
        - OK, but for instagram posts, who cares about ACID?

Redis

- Features
  - Speed - MUCH faster than traditional DBMS
  - Persistence - Sure, but not as expensive as normal DBMS
  - Data Structures - API support for common data structures
    - Vs. general SQL processing

Redis

- Features
  - Atomic Operations - operations on data structures are atomic, but no transaction complexity
  - Replication - Redis allows replication between multiple servers for failover, etc
  - Sharding - Supports sharding data amongst Redis instances
    - More on sharding later

Redis vs. DBMS

- Redis is awesome when
  - Data can fit in memory
  - Data doesn’t need to fit into the traditional DBMS model
- Redis is less awesome when
  - You are dealing with large amounts of data
  - You need extensive ACID guarantees around complex domain data

Using Redis - Data Types

- Redis data types
  - Strings
  - Hashes
  - Lists
  - Sets
  - Sorted Sets
- Internally, Redis uses strings as the core data type
  - E.g. a List is a List of Strings

Using Redis - Data Types

- Despite all primitive values being a strings Redis supports things like
    - Atomic Increment
    - Atomic Decrement
    - Atomic Arithmetic

Installing Redis

- Windows (Microsoft port) https://github.com/microsoftarchive/redis/releases/tag/win-3.0.504
- OSX
  - brew install redis
- Linux
  - You know what to do...

Playing with Redis

- Redis has a very nice online site for experimenting with it

  https://try.redis.io/

- You can use that if you want to follow along

Welcome to **Try Redis**, a demonstration of the Redis database!

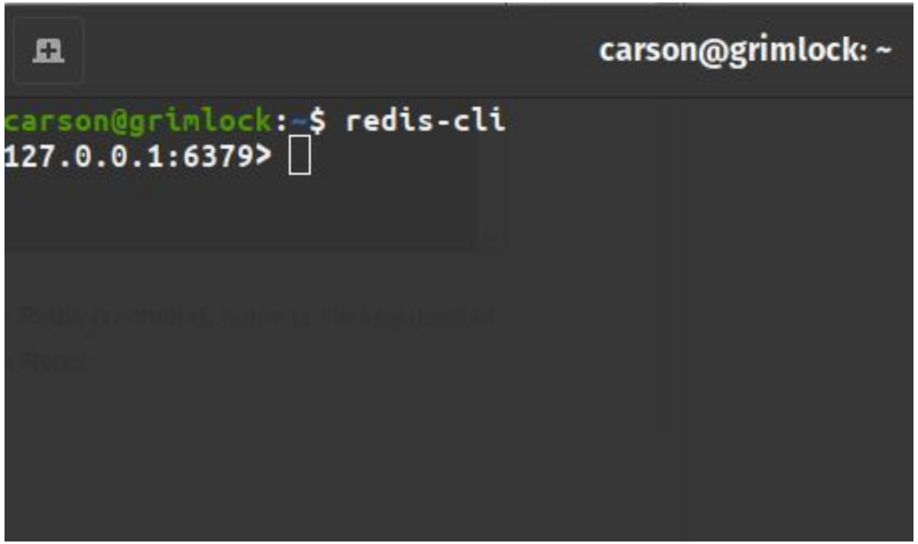
Please type `TUTORIAL` to begin a brief tutorial, `HELP` to see a list of supported commands, or any valid Redis command to play with the database.

Using Redis - CLI

- Using Redis is quite easy
- Fire up a terminal and type

  $ redis-cli

- Should bring up the redis command line, attached to redis server running on localhost



```
carson@grimlock: ~
carson@grimlock:~$ redis-cli
127.0.0.1:6379> 
```

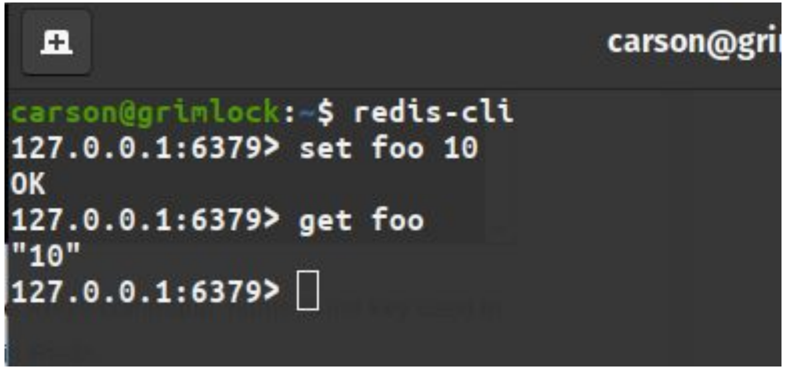
Basic Strings

- Setting a key value

  set <key> <value>

- Getting a value

  get <key>



carson@gri

```
carson@grimlock:~$ redis-cli
127.0.0.1:6379> set foo 10
OK
127.0.0.1:6379> get foo
"10"
127.0.0.1:6379> 
```

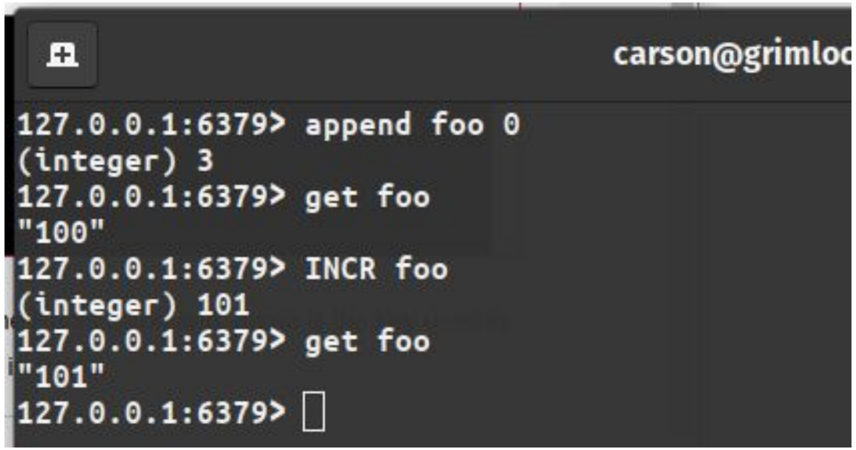
Basic Strings

- Appending to a key value

  append <key> <value>

- Incrementing a value

  incr <key>



carson@grimloc

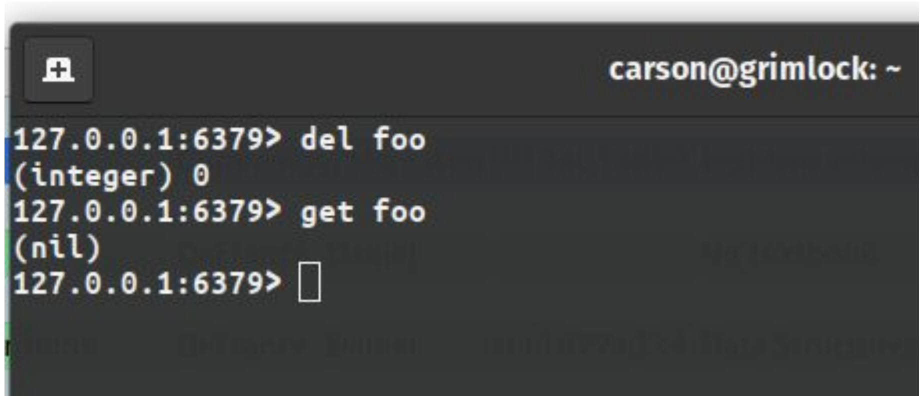
```
127.0.0.1:6379> append foo 0
(integer) 3
127.0.0.1:6379> get foo
"100"
127.0.0.1:6379> INCR foo
(integer) 101
127.0.0.1:6379> get foo
"101"
127.0.0.1:6379> 
```

Deleting A Value

- Use the delete command:

  del <key>

- Value is no longer available in the redis server



```
127.0.0.1:6379> del foo
(integer) 0
127.0.0.1:6379> get foo
(nil)
127.0.0.1:6379> 
```

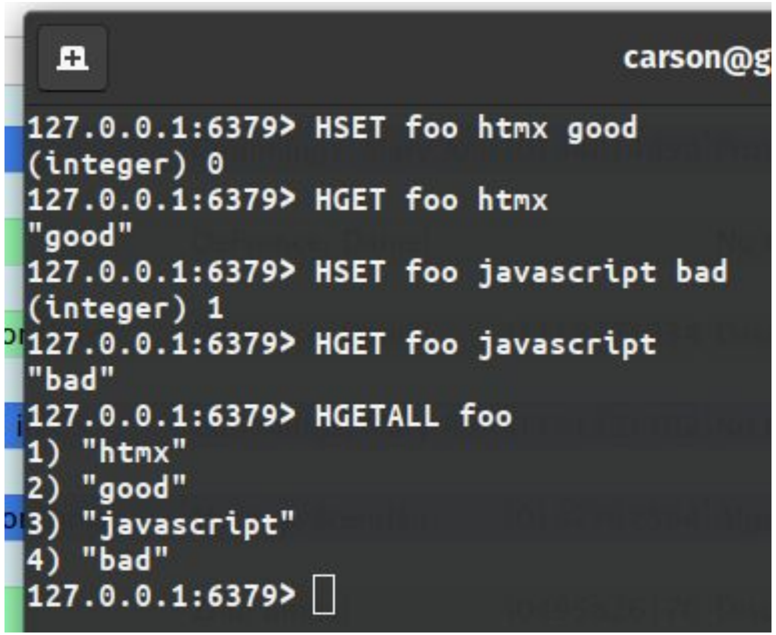
Hashes

- Create a hash with the HSET command

  hset <key> <key> <value>

- Retrieve with HGET

  hget <key> <key>



```
127.0.0.1:6379> HSET foo htmx good
(integer) 0
127.0.0.1:6379> HGET foo htmx
"good"
127.0.0.1:6379> HSET foo javascript bad
(integer) 1
127.0.0.1:6379> HGET foo javascript
"bad"
127.0.0.1:6379> HGETALL foo
1) "htmx"
2) "good"
3) "javascript"
4) "bad"
127.0.0.1:6379> 
```

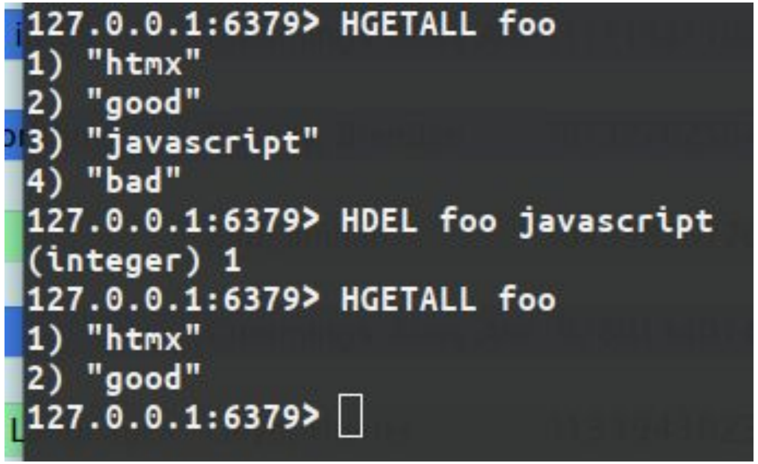
Hashes

- Show all values with the HGETALL command

  hgetall <key>

- Remove with HDEL

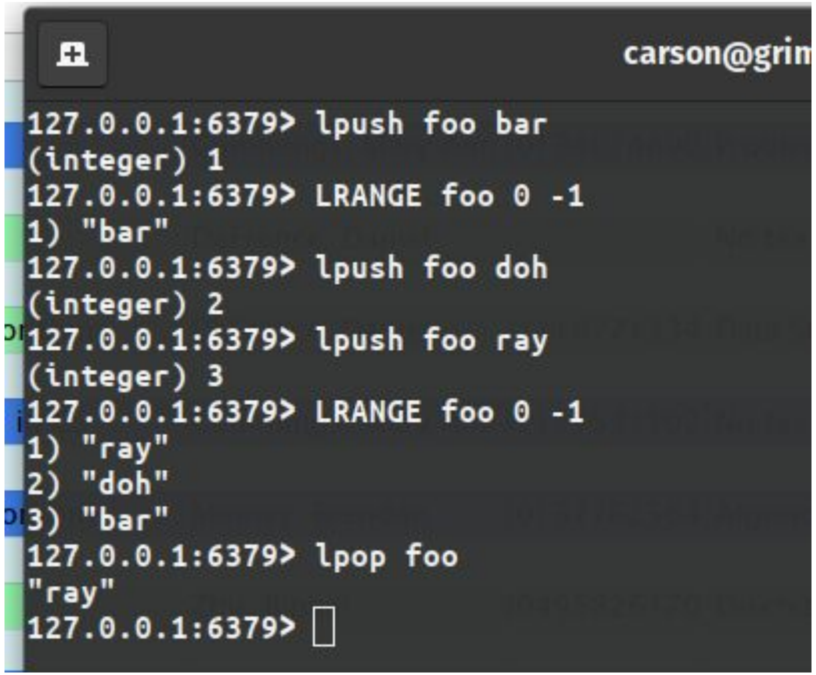
  hdel <key> <key>



```
127.0.0.1:6379> HGETALL foo
1) "htmx"
2) "good"
3) "javascript"
4) "bad"
127.0.0.1:6379> HDEL foo javascript
(integer) 1
127.0.0.1:6379> HGETALL foo
1) "htmx"
2) "good"
127.0.0.1:6379> 
```

Lists

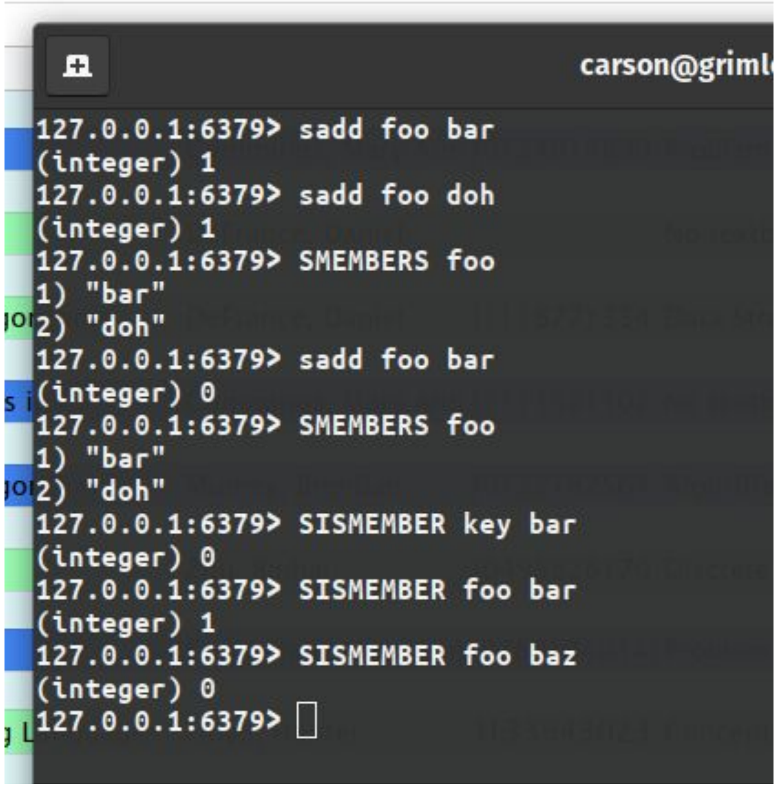
- List Commands
  - lpush - push a value on to front of a list
  - lpop - pop the first value off a list
  - lrange - show the values for a given range
  - rpush - append a value to a list



```
127.0.0.1:6379> lpush foo bar
(integer) 1
127.0.0.1:6379> LRANGE foo 0 -1
1) "bar"
127.0.0.1:6379> lpush foo doh
(integer) 2
127.0.0.1:6379> lpush foo ray
(integer) 3
127.0.0.1:6379> LRANGE foo 0 -1
1) "ray"
2) "doh"
3) "bar"
127.0.0.1:6379> lpop foo
"ray"
127.0.0.1:6379> 
```

Sets

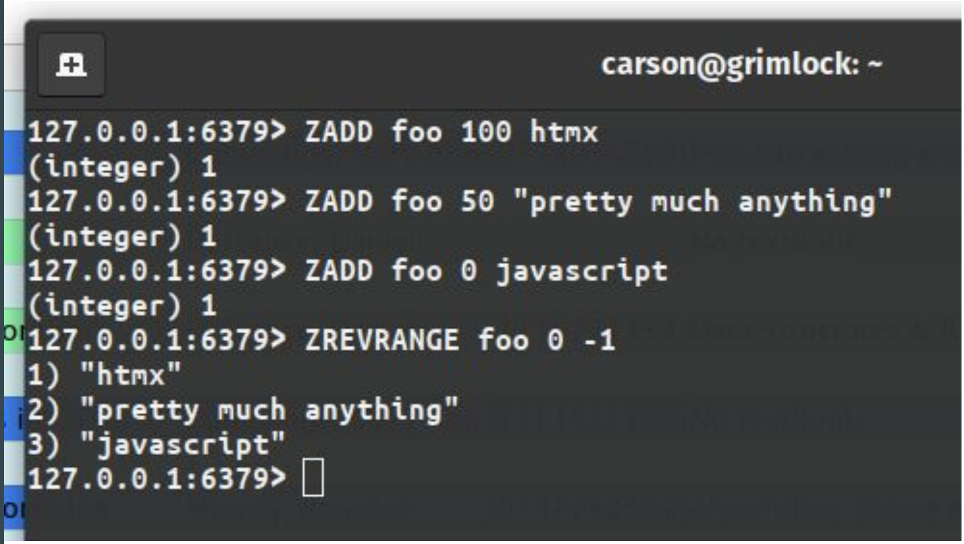
- Set Commands
  - sadd - add a value to a set
  - smembers - show all members of a set
  - sismember - 1 if element is a member of the set, 0 otherwise
  - Also supports many set operations
    - sunion
    - sdiff
    - etc.



```
127.0.0.1:6379> sadd foo bar
(integer) 1
127.0.0.1:6379> sadd foo doh
(integer) 1
127.0.0.1:6379> SMEMBERS foo
1) "bar"
2) "doh"
127.0.0.1:6379> sadd foo bar
(integer) 0
127.0.0.1:6379> SMEMBERS foo
1) "bar"
2) "doh"
127.0.0.1:6379> SISMEMBER key bar
(integer) 0
127.0.0.1:6379> SISMEMBER foo bar
(integer) 1
127.0.0.1:6379> SISMEMBER foo baz
(integer) 0
127.0.0.1:6379>
```

Sorted Sets

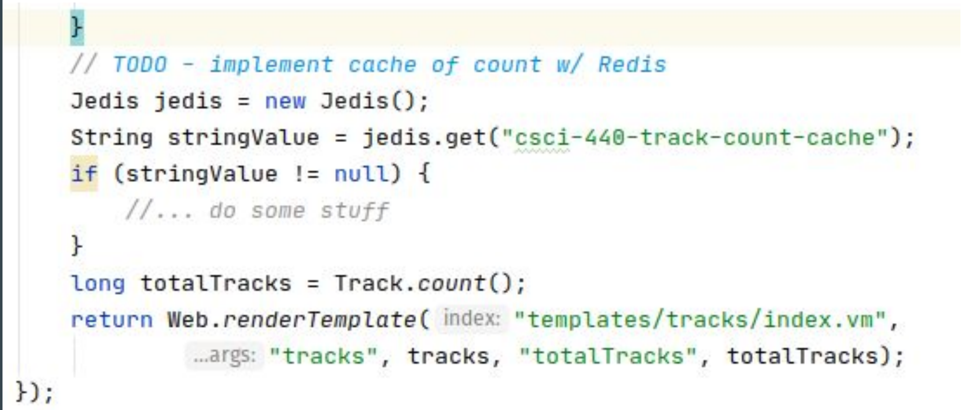
- Sorted Set Commands
  - zadd - add a value to a sorted set
  - zrange - show all members of a sorted set, ordered low to high
  - zrevrange - show all members of a sorted set, ordered high to low



```
127.0.0.1:6379> ZADD foo 100 htmx
(integer) 1
127.0.0.1:6379> ZADD foo 50 "pretty much anything"
(integer) 1
127.0.0.1:6379> ZADD foo 0 javascript
(integer) 1
127.0.0.1:6379> ZREVRANGE foo 0 -1
1) "htmx"
2) "pretty much anything"
3) "javascript"
127.0.0.1:6379> 
```

Java ←→ Redis

- To access the locally running Redis server, we will be using the Jedis client
- Should already be imported for your application
- Jedis has methods for Redis commands
    - Note that you are typically using Strings!



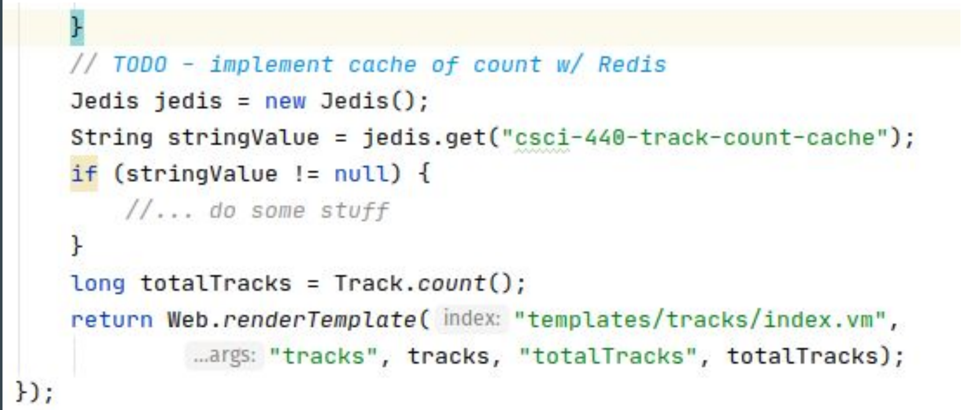
```
    }
    // TODO - implement cache of count w/ Redis
    Jedis jedis = new Jedis();
    String stringValue = jedis.get("csci-440-track-count-cache");
    if (stringValue != null) {
        //... do some stuff
    }
    long totalTracks = Track.count();
    return Web.renderTemplate( index: "templates/tracks/index.vm",
            ...args: "tracks", tracks, "totalTracks", totalTracks);
});
```

Java ⟷ Redis

- NB: You will need to read from and also write to this cache

  “There are only two hard things in Computer Science: cache invalidation and naming things.”

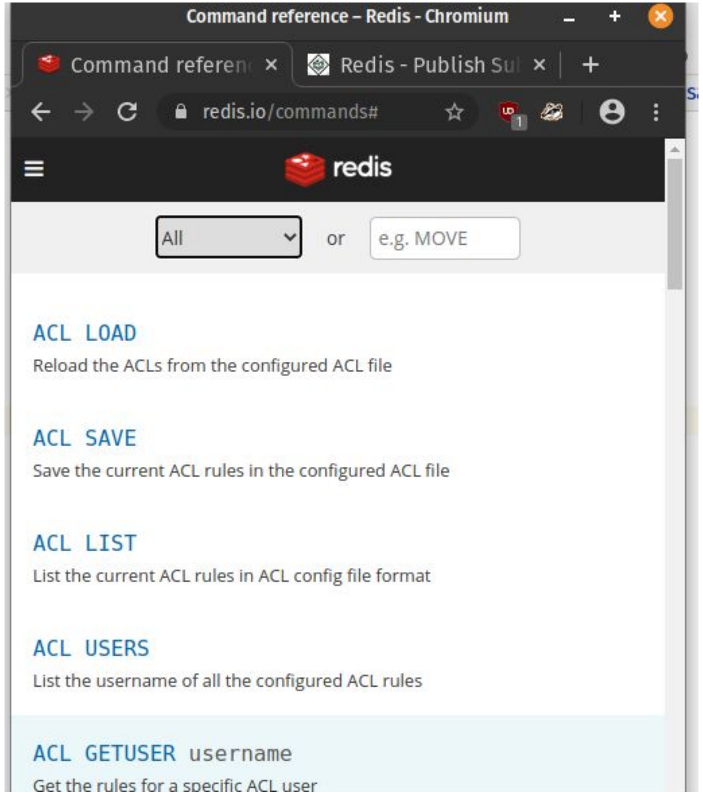
  -- Phil Karlton



```
    }
    // TODO - implement cache of count w/ Redis
    Jedis jedis = new Jedis();
    String stringValue = jedis.get("csci-440-track-count-cache");
    if (stringValue != null) {
        //... do some stuff
    }
    long totalTracks = Track.count();
    return Web.renderTemplate( index: "templates/tracks/index.vm",
            ...args: "tracks", tracks, "totalTracks", totalTracks);
});
```

Redis - Other

- Redis has a *TON* of functionality
    - Can be used for synchronizing processes & threads with *wait commands*
    - HyperLogLog - cheap set count
    - Pub/Sub commands for more client synchronization
- Really is an awesome piece of technology



ACL LOAD
Reload the ACLs from the configured ACL file

ACL SAVE
Save the current ACL rules in the configured ACL file

ACL LIST
List the current ACL rules in ACL config file format

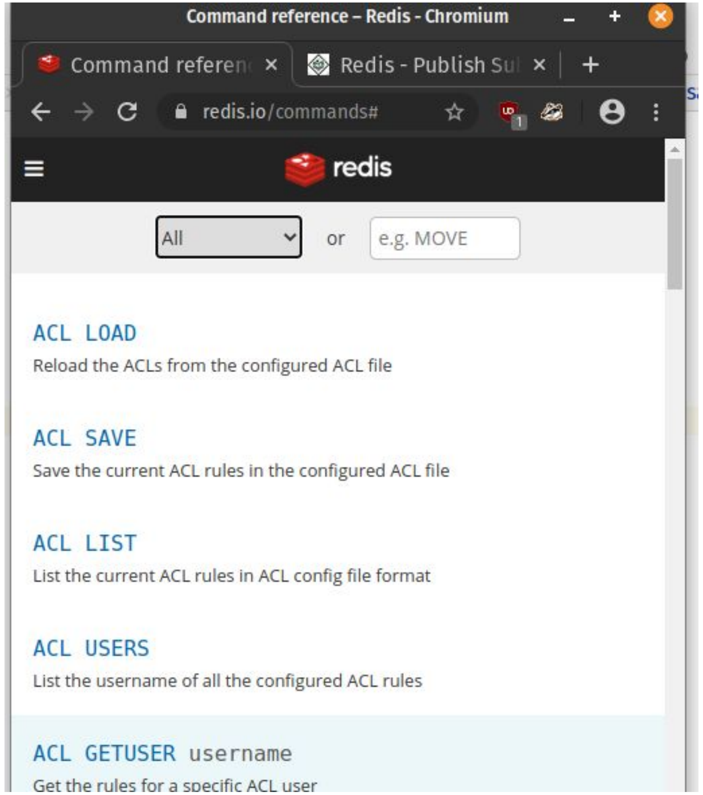
ACL USERS
List the username of all the configured ACL rules

ACL GETUSER username
Get the rules for a specific ACL user

Redis - Other

- One way I like to think of Redis:

  Online, in memory data structures



ACL LOAD
Reload the ACLs from the configured ACL file

ACL SAVE
Save the current ACL rules in the configured ACL file

ACL LIST
List the current ACL rules in ACL config file format

ACL USERS
List the username of all the configured ACL rules

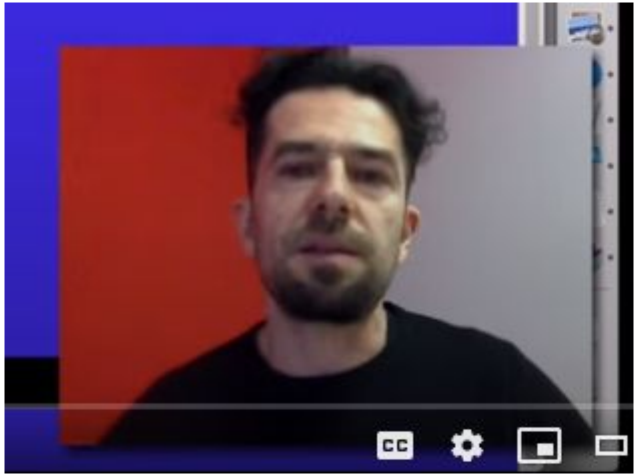
ACL GETUSER username
Get the rules for a specific ACL user

AntiRez

- Excellent youtube series on writing system software

  https://www.youtube.com/watch?v=VBrnmciV9fM

- *“To eeehh show ow tings work…”*



Redis

- A widely used NoSQL datastore
- Uses the key value paradigm
  - Everything is stored in memory
- Supports various types of data
  - Strings
  - Lists
  - Sets
  - Ordered Sets
- Very, very fast
- antirez is an absolute king

MONTANA
STATE UNIVERSITY