





Modelo Chave-Valor

- Base simples baseada em Tabela Hash.
 - Uma tabela de duas colunas: Key, Value
 - Key: String, Value: Blob
- Modelo mais simples dos NoSQL.
- Chaves são tudo, valores são nada. – Karl Seguin

Ranking Chave-Valor

43 systems in ranking, October 2014

Rank	Last Month	DBMS	Database Model	Score	Changes
1.	1.	Redis	Key-value store	79.38	+4.78
2.	2.	Memcached	Key-value store	32.61	+1.02
3.	3.	Riak	Key-value store	12.72	+0.23
4.	4.	DynamoDB	Key-value store	10.58	-0.09
5.	5.	Ehcache	Key-value store	7.40	+0.35
6.	6.	Hazelcast	Key-value store	4.90	+0.33
7.		8. Berkeley DB	Key-value store	2.99	-0.09
8.		7. SimpleDB	Key-value store	2.91	-0.23
9.		9. Coherence	Key-value store	2.74	+0.07

Fonte: db-engines.com

REmote DIctionary Server

- Redis é um armazenamento avançado de chave-valor com fonte aberto e licença BSD.
- Ele é muitas vezes referido como um servidor de estrutura de dados já que as valores podem conter strings, hashes, listas, conjuntos, conjuntos ordenados, bitmaps e hyperloglogs.
- <http://redis.io>

Features

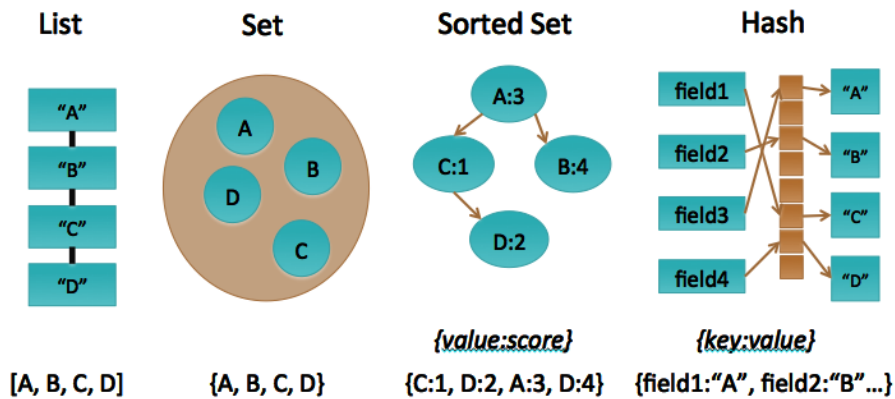
- Dados em memória
- Faz dump em eventos pré-determinados
- Possui estruturas de dados
- Executa scripts em Lua.
- Chave
 - binário. Pode ser desde uma string à uma imagem JPEG
 - Até 512Mb
 - O comum é usar uma string separada por ":"
 - comment:1234:reply.to

Tipos de dados

- String
 - Contém string, inteiro ou ponto-flutuante
- List
 - Lista encadeada de strings
- Set
 - Coleção não ordenada de strings únicas
- Hash
 - Tabela hash desordenada de chaves para valores
- Zset
 - Mapeamento ordenado de strings para scores de ponto-flutuante

Fonte: Carlson, 2013

Estruturas no Redis



Fonte: Internet

Instalação

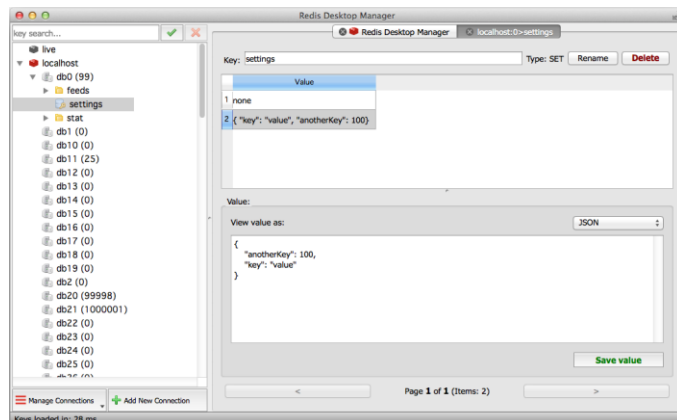
- A distribuição padrão é por código fonte que é facilmente compilada em *nix e OSX através de make install
 - <http://redis.io>
- A Microsoft mantém um projeto no github que porta e compila o Redis para Windows:
 - <https://github.com/Microsoft/redis>

Executáveis

- redis-server
- redis-cli

Ferramenta gráfica free

- Redis Desktop Manager
- <http://redisdesktop.com>



Strings

- SET
 - `set hello world`
 - `set hello city EX 90`
 - `set hello house PX 1000`
 - `set hello cat NX`
 - `set hello dog XX`
- GET
 - `get hello`
- DEL
 - `del hello`

Strings com inteiros

- SET
 - `set hits 1`
- INCR/DECR
 - `incr hits`
 - `decr hits`
- INCRBY/DECRBY
 - `incrby hits 2`

Strings com ponto flutuante

- SET
 - `set temp 23.5`
- INCRBYFLOAT
 - `incrbyfloat temp 1.7`

List

- R PUSH/L PUSH
 - Adiciona no inicio ou fim da lista
 - `rpush key item1`
 - `lpush listkey item2`
- L RANGE
 - Retorna um grupo de valores
 - `lrange key 0 -1`
- L LEN
 - Retorna o tamanho da lista
 - `llen key`
- R POP/L POP
 - Retorna um item da esquerda ou direita
 - `lpop key`

Coleções Set

- **SADD**
 - Adiciona itens a coleção
 - `sadd key item`
- **SMEMBERS**
 - Retorna os itens da coleção
 - `smembers key`
- **SISMEMBER**
 - Verifica se existe um item
 - `sismember key item`
- **SREM**
 - Remove um item
 - `srem key item`

Sorted Set

- **ZADD**
 - Adiciona com score
 - `zadd key 123 item`
- **ZRANGE**
 - Retorna itens pela posição
 - `zrange key 0 -1 withscores`
- **ZRANGEBYSCORE**
 - Retorna itens por score
 - `zrangebyscore key 100 200 withscores`
- **ZREM**
 - Remove item
 - `zrem key item`

Hash

- HSET
 - Define o valor de uma chave e sub-chave
 - `hset key subkey value`
- HGET
 - Pega um valor
 - `hget key subkey`
- HGETALL
 - Pega todos os valores de uma chave
 - `hgetall key`
- HDEL
 - Exclui uma ou mais subchaves
 - `hdel key subkey [subkey...1]`

Transações (!) no Redis

- É possível criar um lote de operações:
 - MULTI
 - Inicia um lote
 - EXEC
 - Executa o lote
 - DISCARD
 - Descarta o lote
- Não há Rollback

Scripts em Lua

- Suporta Lua 5.1
- EVAL
 - Executa um script
 - `eval "return 1" 0`
 - `eval "return redis.call('get','x')" 0`

Redis em Java

- Jedis
 - <https://github.com/xetorthio/jedis>
- Ex:

```
Jedis jedis = new Jedis("127.0.0.1");
jedis.set("foo", "bar");
String value = jedis.get("foo");
```

Classe Produto

```
class Produto implements Serializable {  
    int id;  
    String descricao;  
    double preco;  
}
```

Serializando um objeto

```
Jedis jedis = new Jedis("127.0.0.1");  
  
Produto p = new Produto();  
p.id = 1;  
p.descricao = "Computador 486 DX2";  
p.preco = 1999.99;  
  
ByteArrayOutputStream baos = new ByteArrayOutputStream();  
ObjectOutputStream oos = new ObjectOutputStream(baos);  
oos.writeObject(p);  
  
String key = "produto:" + p.id;  
jedis.set(key.getBytes(), baos.toByteArray());  
jedis.close();
```

Deserializando um objeto

```
Jedis jedis = new Jedis("127.0.0.1");

String key = "produto:" + 1;

byte[] bs = jedis.get(key.getBytes());
ByteArrayInputStream bais = new ByteArrayInputStream(bs);
ObjectInputStream bis = new ObjectInputStream(bais);

Produto p = (Produto) bis.readObject();
System.out.println(p.descricao);

jedis.close();
```

Referências

- CARLSON, Josiah L. **Redis in Action**. Manning Publications Co., 2013.
- SEGUIN, Karl. **The little Redis Book**. 2014.