



Semantix

Redis

Aula 4

Quem sou eu?

Eu sou Rodrigo Augusto Rebouças.

Engenheiro de dados da Semantix
Instrutor do Semantix Mentoring Academy

Você pode me encontrar em:
rodrigo.augusto@semantix.com.br





Pub/Sub



Mensagens Pub/Sub

- Implementar o paradigma de mensagens **Publish/Subscribe** (Publicar/Assinar)
 - Mensagens dos remetente (editor) não são enviadas diretamente para um destinatário (assinante)
 - Mensagens são publicadas através de um canal
 - Sem o editor saber quem são os assinantes
- Sintaxe
 - Publicar mensagem
 - `publish <canal> <mensagem>`
 - Assinar um ou mais canais
 - `subscribe <canal1> ... <canalN>`

Mensagens Pub/Sub

- Ex.
 - Publicar/Assinar

```
127.0.0.1:6379> subscribe canal1
Reading messages... (press Ctrl-C to quit)
1) "subscribe"
2) "canal1"
3) (integer) 1
```

```
127.0.0.1:6379> subscribe canal1
Reading messages... (press Ctrl-C to quit)
1) "subscribe"
2) "canal1"
3) (integer) 1
1) "message"
2) "canal1"
3) "msg de teste"
```

```
127.0.0.1:6379> publish canal1 'msg de teste'
(integer) 1
127.0.0.1:6379>
```

Mensagens Pub/Sub pattern

- Assinar canais através de um padrão
 - Sintaxe
 - `psubscribe <padrão1> ... <padrãoN>`

```
127.0.0.1:6379> psubscribe canal*  
Reading messages... (press Ctrl-C to quit)  
1) "psubscribe"  
2) "canal*"  
3) (integer) 1
```

Mensagens Cancelamento

- Cancelar a assinatura dos canais
 - Sintaxe
 - Canais específicos
 - unsubscribe <canal1> ... <canalN>
 - Todos os canais
 - unsubscribe
- Cancelar a assinatura dos canais através de um padrão
 - Sintaxe
 - punsubscribe <padrao1> ... <padraoN>

```
127.0.0.1:6379> unsubscribe canal1
1) "unsubscribe"
2) "canal1"
3) (integer) 0
```

Exercícios – Pub/Sub

1. Criar um assinante para receber as mensagens do canal noticias:sp
2. Criar um editor para enviar as seguintes mensagens no canal noticias:sp
 - Msg 1
 - Msg 2
 - Msg 3
3. Cancelar o assinante do canal noticias:sp
4. Criar um assinante para receber as mensagens dos canais com o padrão noticias:*
5. Criar um editor para enviar as seguintes mensagens no canal noticias:rj
 - Msg 4
 - Msg 5
 - Msg 6



Configuração Básica



Configuração Básica

- Ler os parâmetros de configuração do servidor Redis em execução
- Os parâmetros podem ser configurados através do arquivo redis.conf
 - Versão do treinamento: <https://raw.githubusercontent.com/redis/redis/6.0/redis.conf>
 - Todas as versões: <https://redis.io/topics/config>
- Sintaxe
 - Ler um parâmetro: config get <Parâmetro>
 - Ler um padrão de parâmetros: config get <Parâmetro>*
 - Ler todos os parâmetros: config get *

```
127.0.0.1:6379> config get appendonly
1) "appendonly"
2) "yes"
```

```
127.0.0.1:6379> config get *
1) "rdbchecksum"
2) "yes"
3) "daemonize"
4) "no"
```

```
293) "bind"
294) ""
295) "requirepass"
296) ""
```

Configuração Básica - Cache

- Cache LRU (Least Recently Used)
 - Permitir que o Redis remova automaticamente os dados antigos
 - Conforme adicionar novos dados
- Principais configurações
 - Maxmemory: Configurar um limite de memória para o dataset
 - Maxmemory-policy: Configurar a política quando o limite de memória for atingido
 - noeviction - Retorna erros quando o limite de memória é atingido
 - allkeys-lru - Remove as chaves menos usadas recentemente
 - volatile-lru – Remove as chave menos usadas recentemente com expiração
 - allkeys-random – Remove as chaves aleatoriamente
 - volatile-random – Remove as chaves aleatoriamente com expiração
 - volatile-ttl – Remove as chave menos usadas recentemente com TTL menor
- Configuração de Cache LRU
<https://redis.io/topics/lru-cache>

Configuração Básica - Cache

- Configuração de Cache
 - <https://redis.io/topics/lru-cache>
- Permitir que remova automaticamente os dados antigos à medida que você adiciona novos dados

```
127.0.0.1:6379> CONFIG GET maxmemory
1) "maxmemory"
2) "0"
127.0.0.1:6379> CONFIG GET maxmemory-policy
1) "maxmemory-policy"
2) "noeviction"
127.0.0.1:6379> CONFIG SET maxmemory 2mb
OK
127.0.0.1:6379> CONFIG SET maxmemory-policy allkeys-lru
OK
127.0.0.1:6379> CONFIG GET maxmemory*
1) "maxmemory-policy"
2) "allkeys-lru"
3) "maxmemory-samples"
4) "5"
5) "maxmemory"
6) "2097152"
```

Exercícios – Configuração

1. Visualizar todos os parâmetros de configuração
2. Verificar o parâmetro “appendonly”
3. Remover a persistência dos dados, alterando o parâmetro “appendonly” para “no”
4. Verificar o parâmetro “save”
5. Adicionar a persistência dos dados, para a cada 2 minutos (120 segundos) se pelo menos 500 chaves forem alteradas, adicionando o parâmetro “save” para “120 500”
6. Verificar os parâmetros “maxmemory*”
7. Permitir que o Redis remova automaticamente os dados antigos à medida que você adiciona novos dados, com uso da politica “allkeys-lru”, quando chegar a 1mb de memória.



Revisão



Redis Revisão

- Tutorial Redis online
 - <http://try.redis.io>

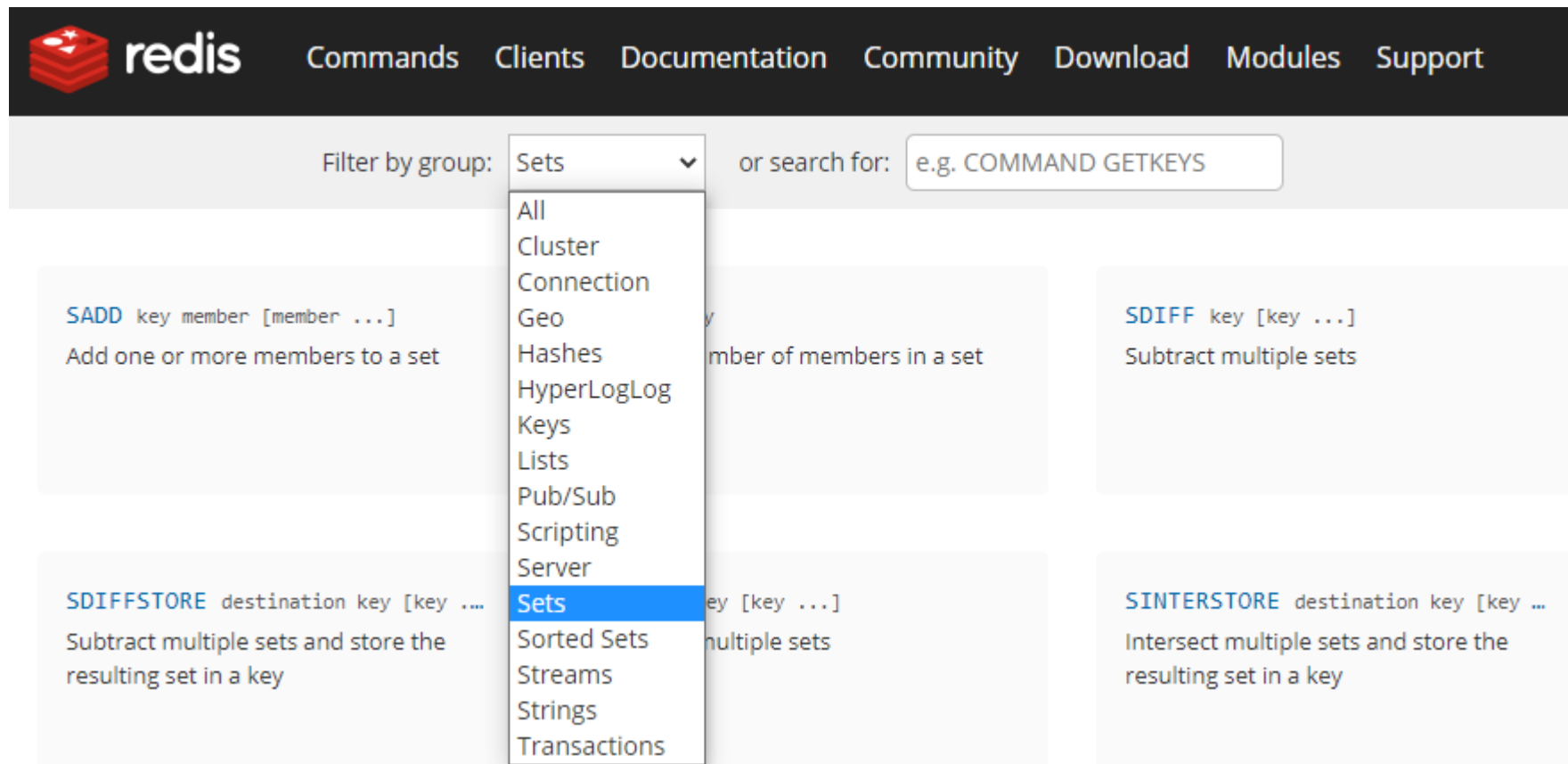


The screenshot shows the 'Try Redis' web application. At the top, there is a header with the text '* TRY REDIS *' in a stylized, hand-drawn font. Below the header, a dark gray box contains the following text: 'Welcome to **Try Redis**, a demonstration of the Redis database!' and 'Please type **TUTORIAL** to begin a brief tutorial, **HELP** to see a list of supported commands, or any valid Redis command to play with the database.' Below this, a light yellow box contains the text '> TUTORIAL'. Underneath, a dark gray box contains the following text: 'Redis is in the family of databases called key-value stores.', 'The essence of a key-value store is the ability to store some data, called a value, inside a key. This data can later be retrieved only if we know the exact key used to store it.', 'Often Redis it is called a data structure server because it has outer key-value shell, but each value can contain a complex data structure, such as a string, a list, a hashes, or ordered data structures called sorted sets as well as probabilistic data structures like hyperloglog.', and 'As a first example, we can use the command **SET** to store the value "fido" at key "server:name":'. At the bottom of this dark gray box, the command 'SET server:name "fido"' is displayed. At the very bottom of the interface, there is a light yellow box with a large '>' character and a vertical line, indicating a command prompt.

Redis Revisão

- Lista completa de comandos do Redis

- <https://redis.io/commands>



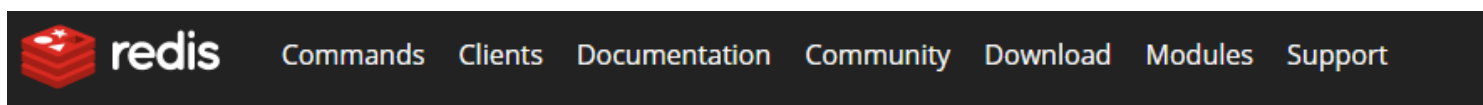
The screenshot shows the Redis.io website's command list interface. At the top, there is a navigation bar with the Redis logo and links for Commands, Clients, Documentation, Community, Download, Modules, and Support. Below the navigation bar, there is a search bar with the text "Filter by group:" and a dropdown menu currently set to "Sets". To the right of the dropdown is a search input field with the placeholder text "or search for: e.g. COMMAND GETKEYS". The main content area displays a grid of command cards. Visible cards include:

- SADD** key member [member ...]: Add one or more members to a set.
- SDIFF** key [key ...]: Subtract multiple sets.
- SDIFFSTORE** destination key [key ...]: Subtract multiple sets and store the resulting set in a key.
- SINTERSTORE** destination key [key ...]: Intersect multiple sets and store the resulting set in a key.

 A dropdown menu is open over the "Sets" filter, listing various data types and structures: All, Cluster, Connection, Geo, Hashes, HyperLogLog, Keys, Lists, Pub/Sub, Scripting, Server, **Sets** (highlighted), Sorted Sets, Streams, Strings, and Transactions.

○ Documentação oficial do Redis

- <https://redis.io/documentation>



Documentation

Note: The Redis Documentation is also available in raw (computer friendly) format in the [redis-doc github repository](#). The Redis Documentation is released under the [Creative Commons Attribution-ShareAlike 4.0 International license](#).

Programming with Redis

- The full list of [commands](#) implemented by Redis, along with thorough documentation for each of them.
- [Pipelining](#): Learn how to send multiple commands at once, saving on round trip time.
- [Redis Pub/Sub](#): Redis is a fast and stable Publish/Subscribe messaging system! Check it out.
- [Redis Lua scripting](#): Redis Lua scripting feature documentation.
- [Debugging Lua scripts](#): Redis 3.2 introduces a native Lua debugger for Redis scripts.
- [Memory optimization](#): Understand how Redis uses RAM and learn some tricks to use less of it.
- [Expires](#): Redis allows to set a time to live different for every key so that the key will be automatically removed from the server when it expires.
- [Redis as an LRU cache](#): How to configure and use Redis as a cache with a fixed amount of memory and auto eviction of keys.



Semantix

Obrigado!

Alguma pergunta?



Você pode me encontrar em:
rodrigo.augusto@semantix.com.br

GET SMARTER