Semantix

Redis

Aula 2

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

Estrutura de dados

Semantix

Redis Tipos de dados

- Servidor de estruturas de dados que oferece suporte a diferentes tipos de valores
  - Não é um armazenamento de chave-valor simples
    - String-string
  - Aceita estruturas de dados mais complexas
    - Strings
    - Listas
    - Sets
    - Sets Ordenados
    - Hash
    - HyperLogLogs
    - Streams

Redis Chave

- As chaves são *binary safe*
  - Sequência binária
  - Regras
    - Tamanho máximo 512 MB
      - Não indicado chaves > 1 MB
      - Gastar memória e largura de banda
    - Dicas
      - Instanciar “object-type:id”
        - “user:1000” ao invés de “u1000”

Strings

Semantix

Redis String

- Único tipo de dados *Memcached*
  - Cache para páginas Web
- Chaves também são string
  - Mapear string para outra string
- Sintaxe
  - Definir um valor de string: SET <chave> <valor>
  - Recuperar um valor de string: GET <chave>

```
root@2f3da0f8aeed:/data# redis-cli
127.0.0.1:6379> set minhaChave valorChave
OK
127.0.0.1:6379> get minhaChave
"valorChave"
```

Redis String

- Opções para a chave String
  - NX – Falhar se a chave existir
  - XX (Default) – Substituir o valor da chave

```
127.0.0.1:6379> set minhaChave novoValor nx
(nil)
127.0.0.1:6379> set minhaChave novoValor xx
OK
```

- Verificar o tamanho do valor
  - Sintaxe: strlen <chave>

```
127.0.0.1:6379> get minhaChave
"novoValor"
127.0.0.1:6379> strlen minhaChave
(integer) 9
```

Redis String

- String como um inteiro
  - Sintaxe para incrementos e decrementos do valor
    - incr <chave>
    - decr <chave>
    - incrby <chave> <incremento>
    - decrby <chave> <decremento>
- Incr atômico
  - Ex.
    - Ao mesmo tempo 2 clientes leem a chave 10 e ambos incrementem para 11
    - O valor final é 12
      - Não são executados os comandos ao mesmo tempo

```
127.0.0.1:6379> set contador 10
OK
127.0.0.1:6379> incr contador
(integer) 11
127.0.0.1:6379> incrby contador 5
(integer) 16
127.0.0.1:6379> decr contador
(integer) 15
127.0.0.1:6379> decrby contador 5
(integer) 10
127.0.0.1:6379> get contador
"10"
```

Redis String

- Definir e recuperar várias chaves em um comando
  - Reduzir latência
- Sintaxe
  - MSET <chave 1> <valor> <chave 2> <valor> ... <chave N> <valor>
  - MGET <chave 1> <chave 2>... <chave N>

```
127.0.0.1:6379>
127.0.0.1:6379> mset chave1 10 chave2 20 chave3 30
OK
127.0.0.1:6379> mget chave1 chave2 chave3
1) "10"
2) "20"
3) "30"
```

Opções Básicas com Chaves

Semantix

Redis Opções com chaves

- Verificar se a chave existe
  - Sintaxe
    - exists <chave>
- Deletar chave
  - Sintaxe
    - del <chave>
- Tipo da chave
  - Sintaxe
    - type <chave>

```
127.0.0.1:6379> set minhaChave teste
OK
127.0.0.1:6379> type minhaChave
string
127.0.0.1:6379> exists minhaChave
(integer) 1
127.0.0.1:6379> del minhaChave
(integer) 1
127.0.0.1:6379> exists minhaChave
(integer) 0
127.0.0.1:6379> type minhaChave
none
127.0.0.1:6379> get minhaChave
(nil)
```

Redis Persistência de Chaves

- Definir tempo para a chave expirar
  - Sintaxe
    - expire <chave> <tempo segundos>
    - pexpire <chave> <tempo milissegundos>
    - set <chave> <valor> ex <tempo segundos>
    - set <chave> <valor> px <tempo milissegundos >
- Verificar o tempo restante de vida da chave
  - Sintaxe
    - ttl <chave> (reposta em segundos)
    - pttl <chave> (reposta em milissegundos)

```
127.0.0.1:6379> set minhaChave teste
OK
127.0.0.1:6379> ttl minhaChave
(integer) -1
127.0.0.1:6379> expire minhaChave 20
(integer) 1
127.0.0.1:6379> get minhaChave
"teste"
127.0.0.1:6379> ttl minhaChave
(integer) 7
127.0.0.1:6379> ttl minhaChave
(integer) 3
127.0.0.1:6379> ttl minhaChave
(integer) -2
127.0.0.1:6379> get minhaChave
(nil)
```

Redis Persistência de Chaves

- Remover tempo para a chave expirar
  - Sintaxe
    - persist <chave>

```
127.0.0.1:6379> set minhaChave teste2 ex 50
OK
127.0.0.1:6379> ttl minhaChave
(integer) 46
127.0.0.1:6379> ttl minhaChave
(integer) 41
127.0.0.1:6379> persist minhaChave
(integer) 1
127.0.0.1:6379> ttl minhaChave
(integer) -1
127.0.0.1:6379> get minhaChave
"teste2"
```

Exercícios - Strings

1. Criar a chave "usuario:nome" e atribua o valor do seu nome
2. Criar a chave "usuario:sobrenome" e atribua o valor do seu sobrenome
3. Busque a chave "usuario:nome“
4. Verificar o tamanho da chave "usuario:nome“
5. Verificar o tipo da chave "usuario:sobrenome"
6. Criar a chave "views:qtd" e atribua o valor 100
7. Incremente o valor em 10 da chave "views:qtd"
8. Busque a chave "views:qtd"
9. Deletar a chave "usuario:sobrenome"
10. Verificar se a chave "usuario:sobrenome" existe
11. Definir um tempo de 3600 segundos para a chave "views:qtd" ser removida
12. Verificar quanto tempo falta para a chave "views:qtd" ser removida
13. Verificar a persistência da chave "usuario:nome"
14. Definir para a chave "views:qtd" ter persistência para sempre

Listas

Semantix

Redis Listas

- Lista é uma sequência de elementos ordenados
  - *Linked List* – Tempo constante de inserção
- Usos
  - O *Twitter* leva os últimos tweets postados por usuários nas listas do Redis
  - Lembrete de últimas atualizações postadas por usuários em uma rede social
  - Comunicação entre consumidor-produtor
    - Produtor inseri elementos em uma lista e o consumidor consome esses itens e executa as ações
  - Visualizar as fotos ou publicações mais recentes publicadas em uma rede social

Redis Listas

- Adicionar um novo elemento na lista

L	...	...	R
0	1	...	N

  - Sintaxe
    - No inicio (esquerda) da lista
      - LPUSH <chave> <valor>
    - No final (direita) da lista
      - RPUSH <chave> <valor>
- Extrair elementos em um intervalo na lista
  - Sintaxe
    - LRANGE <chave> <inicio> <fim>
  - Inicia a lista em 0
  - -1 último, -2 penúltimo, ...

```
127.0.0.1:6379> rpush lista primeiro
(integer) 1
127.0.0.1:6379> rpush lista segundo
(integer) 2
127.0.0.1:6379> lpush lista novoPrimeiro
(integer) 3
127.0.0.1:6379> rpush lista ultimo
(integer) 4
127.0.0.1:6379> lrange lista 0 -1
1) "novoPrimeiro"
2) "primeiro"
3) "segundo"
4) "ultimo"
127.0.0.1:6379> lrange lista 1 3
1) "primeiro"
2) "segundo"
3) "ultimo"
```

Redis Listas

- Recuperar um elemento e eliminá-lo da lista
  - Sintaxe
    - No inicio (esquerda) da lista
      - LPOP <chave>
    - No final (direita) da lista
      - RPOP <chave>
  - Retornar Null quando não existir elementos

```
127.0.0.1:6379> lrange lista 0 -1
1) "novoPrimeiro"
2) "primeiro"
3) "segundo"
4) "ultimo"
127.0.0.1:6379> lpop lista
"novoPrimeiro"
127.0.0.1:6379> lpop lista
"primeiro"
127.0.0.1:6379> rpop lista
"ultimo"
127.0.0.1:6379> lrange lista 0 -1
1) "segundo"
127.0.0.1:6379> lpop lista
"segundo"
127.0.0.1:6379> lpop lista
(nil)
```

Redis Listas

- Recuperar um elemento e eliminá-lo da lista
  - Bloquear se a lista estiver vazia até o tempo especificado
    - Evitar respostas Null
    - Implementar uma fila
      - lpush
      - rpop
  - Sintaxe
    - No inicio (esquerda) da lista
      - BLPOP <chave> <tempo>
    - No final (direita) da lista
      - BRPOP <chave> <tempo>
  - Retornar Null quando não existir elementos e o tempo se esgotar

```
127.0.0.1:6379> lpush teste 1 2 3
(integer) 3
127.0.0.1:6379> brpop teste 5
1) "teste"
2) "1"
127.0.0.1:6379> brpop teste 5
1) "teste"
2) "2"
127.0.0.1:6379> brpop teste 5
1) "teste"
2) "3"
127.0.0.1:6379> brpop teste 5
(nil)
(5.09s)
```

Redis Listas

- Definir um novo intervalo para a lista
  - Sintaxe
    - ltrim <chave> <novoInicio> <novoFim>
  - Removidos todos os elementos fora do intervalo

- Visualizar o tamanho da lista
  - Sintaxe
    - llen <chave>

```
127.0.0.1:6379> rpush letras A B C D E F
(integer) 6
127.0.0.1:6379> lrange letras 0 -1
1) "A"
2) "B"
3) "C"
4) "D"
5) "E"
6) "F"
127.0.0.1:6379> llen letras
(integer) 6
127.0.0.1:6379> ltrim letras 2 4
OK
127.0.0.1:6379> llen letras
(integer) 3
127.0.0.1:6379> lrange letras 0 -1
1) "C"
2) "D"
3) "E"
```

Exercícios - Listas

1. Criar a chave “views:ultimo_usuario" e insira nesta ordem os seguintes valores como lista:
   - Final da lista “Joao”
   - Final da lista “Ana”
   - Inicio da lista “Carlos”
   - Final da lista “Carol”
2. Visualizar todos os elementos da lista
3. Visualizar todos os elementos da lista, com exceção do último
4. Visualizar o tamanho da lista
5. Redefinir o tamanho da lista, removendo o primeiro elemento (Sem usar o pop)
6. Visualizar o tamanho da lista
7. Recuperar os elementos da lista da seguinte ordem:
   - Primeiro
   - Último
   - Primeiro com bloqueio de 5 segundos se a lista estiver vazia
   - Primeiro com bloqueio de 5 segundos se a lista estiver vazia

Semantix

Obrigado!

Alguma pergunta?

Você pode me encontrar em:

rodrigo.augusto@semantix.com.br

GET SMARTER