01

O que o Redis resolve

Transcrição

No nosso dia-a-dia, quando estamos desenvolvendo uma aplicação *Web*, uma aplicação *Mobile*, ou até mesmo um jogo, sempre nos deparamos com situações em que se requer uma pesquisa. Por exemplo, em um jogo gostaríamos de saber quais amigos estão *online*, ou qual o *ranking* mundial. Num aplicativo *mobile* queremos saber quais amigos estão *online*. Num site da *Web*, podemos ter uma pesquisa que varre a base de dados inteira dele, como é no Alura.

O nosso site apresenta, logo na *homepage*, números que resumem toda a nossa aplicação:

Imagine que toda vez que um usuário acessa o site do Alura procurando uma aula, ele tenha que esperar a aplicação varrer toda a base de dados, para só depois ele conseguir visualizar a página e decidir qual aula deve assistir. Seria um processo muito lento.

É muito melhor que a aplicação retorne apenas um número que seja significativo, por exemplo, a quantidade de alunos, ou o número de aulas. A pesquisa deve estar focada naquilo que o usuário está procurando. É desnecessária uma pesquisa mais complexa.

É interessante conseguirmos armazenar uma chave - "Total de Cursos" - e um valor - "105". É interessante termos um sistema com uma base de dados que não necessariamente seja *sequel* ou *SQL*, mas que seja somente "*chave-valor*". Tal valor pode ser um número, uma palavra, ou uma lista de nomes.

Queremos, então, armazenar em algum lugar um conjunto de chaves e valores, de modo que a busca seja extremamente rápida, otimizada. Para isso, vamos utilizar o ***Redis***.

Vamos abrir o prompt de comando, entrar no diretório onde o *Redis* está instalado e mandar rodar o *Redis-server*. Funciona da mesma forma que um *SQL*, o servidor fica rodando e podemos ignorá-lo.

```
... cd src
./redis-server
```

Agora temos que nos conectar ao servidor como um cliente:

```
... cd src
./redis-cli
```

Agora estamos conectados. Mas vamos fazer um teste simples apenas por desencargo de consciência (se não estivéssemos, apareceria uma mensagem de erro): vamos pedir para que uma frase seja repetida, ecoada:

```
<ip> ECHO "estou no redis"
```

Se retornar tal frase, então está tudo certo.

Vamos enfim resolver o problema de busca do nosso site. Não queremos que ela varra todos os cursos e os contabilize. Queremos apenas que retorne o valor "105", que é o total de cursos do Alura.

Precisamos colocar - ou *setar* - o valor do total de cursos:

```
<ip> SET "total_de_cursos" 105
OK
```

Definimos a chave "total_de_cursos" com o valor "105". A partir de agora, qualquer cliente do *Redis* pode se conectar e buscar o total de cursos. Para testar, abrimos uma nova aba, conectamos como cliente no servidor e chamamos essa chave:

```
<ip> GET "total_de_cursos"
"105"
```

O valor foi armazenado no servidor através da chave. O problema foi resolvido. Não precisamos varrer tabelas ou buscar dados. A pergunta é direta e a resposta também. Podemos fazer isso para todas as chaves. Veja mais um exemplo:

```
<ip> SET "total_de_respostas" 1446014
OK

<ip> GET "total_de_respostas"
"1446014"
```

Repare que apesar de termos pedido para que o *Redis* armazenasse o valor como um número, ele o devolveu como uma *string* (por isso as aspas). O mais importante é a facilidade para armazenar esses dados.

Mas e se os valores mudarem, se acrescentarmos mais aulas, ou aparecerem mais respostas? Adianta mudar o valor da chave?

```
<ip> SET "total_de_respostas" 1446015
OK

<ip> GET "total_de_respostas"
"1446015
```

Sim, funciona da mesma forma! Em qualquer instante podemos trocar o valor atrelado a uma chave.

Lembre-se que o valor não precisa ser necessariamente um número, ele pode ser um nome:

```
<iP> SET "ultimo_usuario_que_se_logou" "guilherme silveira"
OK

<ip> GET "ultimo_usuario_que_se_logou"
"guilherme silveira"
```

Assim como conseguimos armazenar e buscar o valor de uma chave, também podemos exclui-lo:

```
<ip> DEL "ultimo_usuario_que_se_logou"
(integer) 1

<ip> GET "ultimo_usuario_que_se_logou"
(nil)
```

Se buscarmos o valor da chave, o servidor retornará "*nil*", ou seja, nulo, vazio.

Vimos, então, a primeira e mais comum utilização do *Redis*. Armazenar valores faz com que as buscas sejam muito mais rápidas.