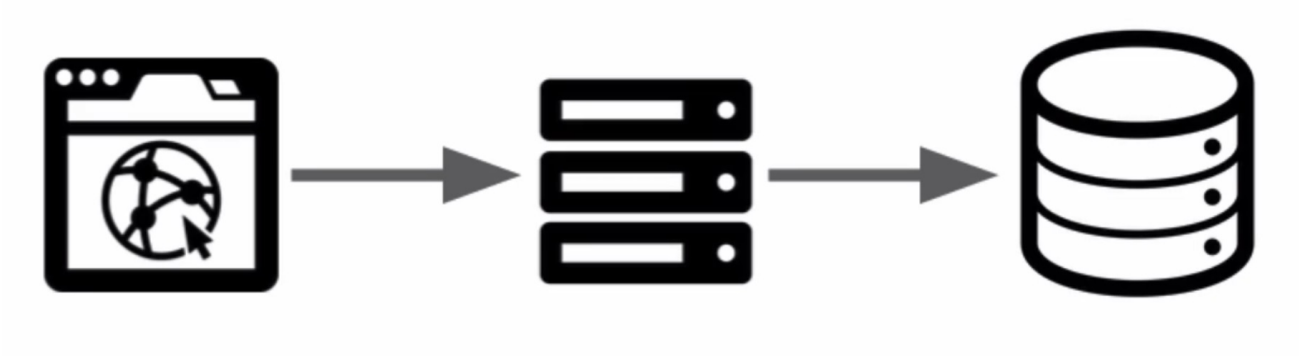
04

Sessão de um usuário web

Capítulo 6 - Sessão de um usuário web

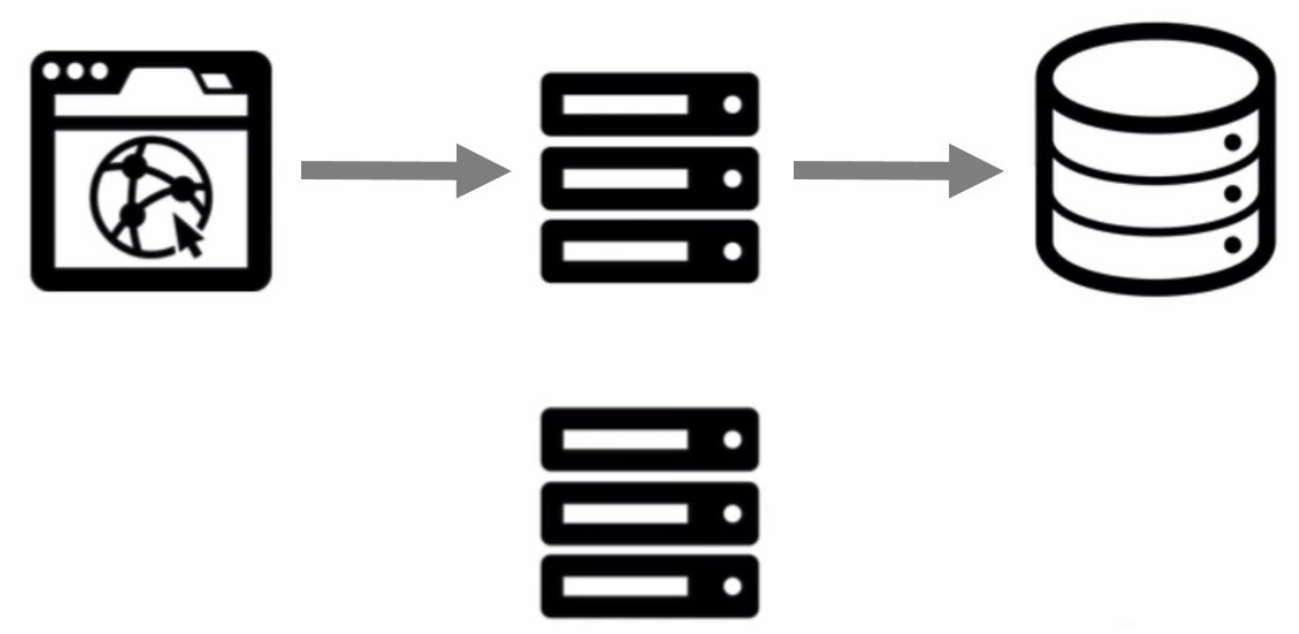
Vejamos um cenário muito comum na *Web*: um usuário no navegador que gostaria de acessar o *servidor web* de site. Por sua vez, o servidor web pode buscar as informações num servidor *Redis* como, por exemplo, quem foi o último usuário a se logar, qual a última notícia, qual tem sido a notícia mais comentada, etc.

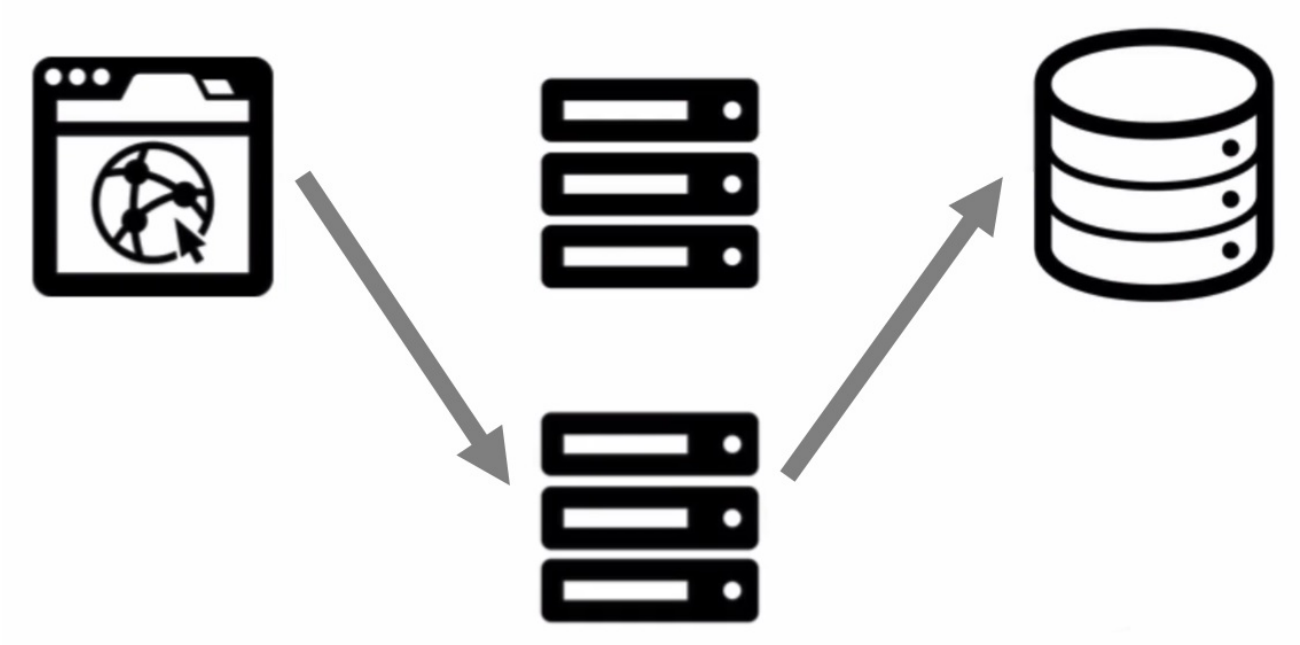


Em determinadas situações, o usuário tem características atreladas somente a ele, como seu carrinho de compras, e queremos guardar essas informações para sempre ou por um tempo limitado, dependendo da decisão de negócio do site. Tais informações que serão guardadas por tempo limitado devem permanecer em um lugar específico.

Quando existem dois ou mais servidores web, no caso de um site com muitas informações, existe a chance do usuário transitar entre eles. Se os dados atrelados a esse usuário estiverem alocados em um servidor web, a partir do momento em que ele acessa o outro, eles ficarão invisíveis. Deveria haver uma replicação de informações para que isso não ocorresse.

Nós queremos que haja um compartilhamento das informações com os servidores, logo, elas devem ficar alocadas no servidor de dados, no banco de dados. Nesse caso usaremos o *Redis* para armazenar os dados da sessão atrelados ao usuário. Dessa forma não importa em qual servidor web o usuário está conectado.





Jogando as informações em um servidor *Redis*, não existe a necessidade de haver replicação de dados nos servidores web.

Mas agora temos um outro problema a ser resolvido: no caso de informações temporárias, como fazemos para que elas sumam da base de dados automaticamente depois de um determinado tempo?

Temos uma situação onde um cliente se conecta com o seu navegador em um ou mais servidores web e se loga num site de compras. Queremos armazenar informações da sessão desse usuário num servidor *Redis*. Informações estas que queremos apagar depois de, por exemplo, 30 minutos.

Depois de estarmos conectados no *Redis*, vamos colocar informações do usuário que está logado. Vimos que podemos usar o *hash*:

```
<ip> HSET "sessao:usuario:1675" "nome" "guilherme"
(integer) 1

<ip> HSET "sessao:usuario:1675" "total_de_produtos" "3"
(integer) 1
```

Aprendemos em aulas passadas o *hash*, que é um dicionário de dados, e o *Multiple Set*, que armazena mais de uma informação. Agora queremos armazenar mais de uma informação dentro de um dicionário, então usamos o *Multiple Hash Set*, ou "*MHSET*":

```
<ip> HMSET "sessao:usuario:1675" "nome" "guilherme" "total_de_produtos" "3" "sobrenome" "silvei
OK
```

Se buscarmos um dos valores da chave:

```
<ip> HGET "sessao:usuario:1675" "nome"
"guilherme"

<ip> HGET "sessao:usuario:1675" "sobrenome"
"silveira"
```

Agora vamos ver como fazer para que eles sejam apagados automaticamente. Para isso usamos o comando "*EXPIRE*"

```
<ip> EXPIRE "sessao:usuario:1675" 1800
(integer) 1
```

Isso significa que daqui 1800 segundos, ou 30 minutos, os dados do usuário serão apagados, expirados. Para sabermos quanto tempo falta para que essas informações sejam apagadas usamos o comando *Time Left* ou "*TTL*":

```
<ip> TTL "sessao:usuario:1675"
(integer) 1680
```

O que significa que se passaram 2 minutos desde o começo da contagem regressiva, faltam ainda 28 minutos para os dados expiarem.

Passados os 30 minutos, se tentarmos pegar um dos valores da chave, o *Redis* retornará *nulo*:

```
<ip> HGET "sessao:usuario:1675" "nome"
(nil)
```

É muito comum em um site da web que toda vez que há uma requisição por parte do usuário, esse contador (EXPIRE) é ativado. Ou seja, para cada clique ou página visitada, nós postergamos o *logout* do usuário.