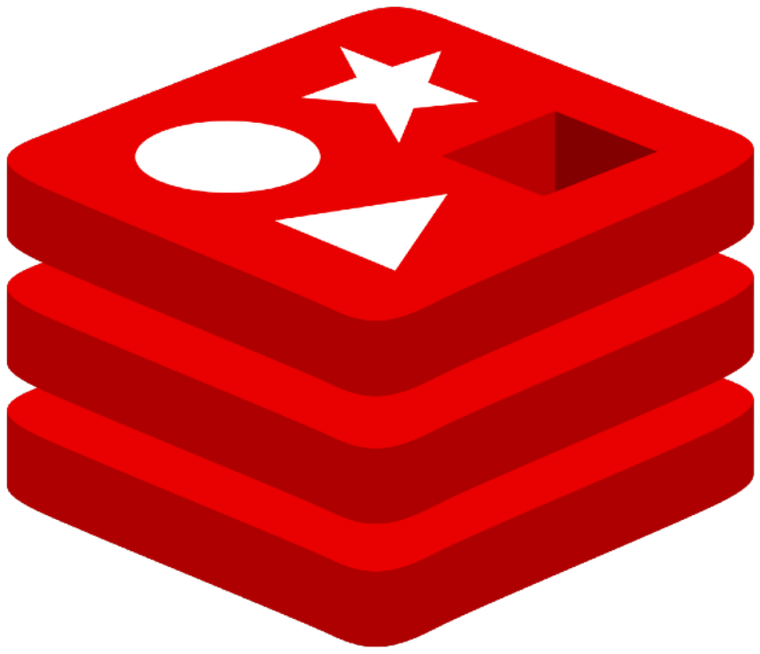


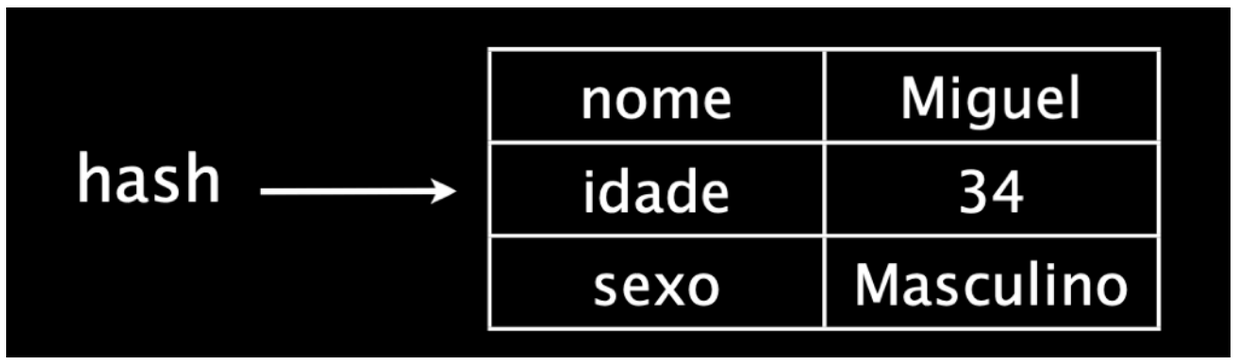
Banco de dados NoSQL - Chave/Valor com Redis

Hash, Listas e Pub/Sub

Prof. Gustavo Leitão

Utilizando Hashs

Armazenando Hashs



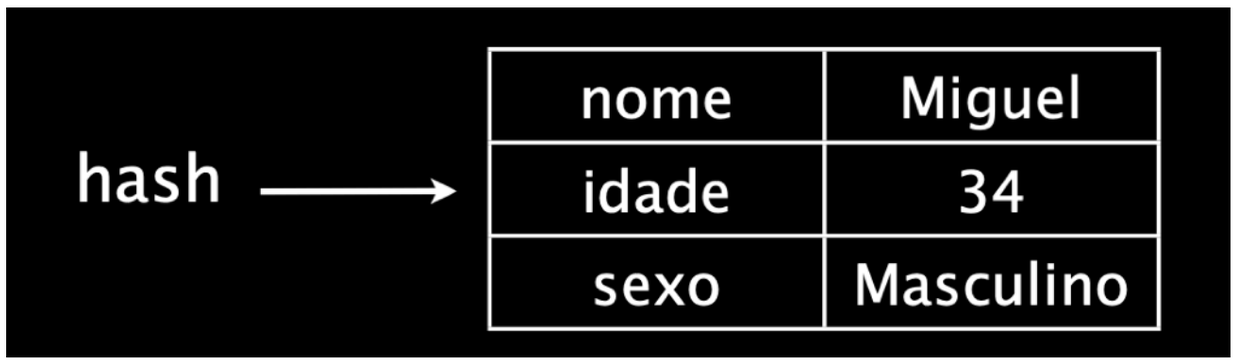

```
$ HSET my-key field “value"
```

Armazena o campo field na chave my-key

```
$ HGET my-key field
```

Recupera o valor do campo field da chave my-key

Armazenando Hashs




```
$ HSET user:10 nome “Miguel"
```

```
$ HSET user:10 idade 34
```

```
$ HSET user:10 sexo “Masculino"
```

Armazenando Hashs

hash →

nome	Miguel
idade	34
sexo	Masculino

```
$ HMSET user:10 nome “Miguel” idade 34 sexo

“Masculino”
```

Armazenando Hashs

```
$ HGETALL my-key
```

Recupera todos os campos da chave my-key

```
$ HSETNX my-key field "Hello"
```

Armazena o valor “Hello" no campo field apenas se o campo não estiver definido.

```
redis> HSETNX myhash field "Hello"
(integer) 1
redis> HSETNX myhash field "World"
(integer) 0
redis> HGET myhash field
"Hello"
redis>
```

Armazenando Hashs

```
$ HEXISTS my-key field
```

Verifica se determinado campo existe

```
$ HDEL my-key field
```

Remove o campo field da chave my-key

Utilizando Listas

Armazenando Listas

As listas permitem armazenar um conjunto de valores em uma mesma chave.
Exemplos: Ultimas 10 notícias de um site; Ranking de pontuação de um jogo.

```
$ LPUSH -> Insere elementos à esquerda (início)
```

```
$ RPUSH -> Insere elementos à direita (fim)
```

```
$ LPOP -> Remove e retorna um elemento à esquerda (início)
```

```
$ RPOP -> Remove e retorna um elemento à direita (início)
```

[]

["1@gmail.com"]

["2@gmail.com","1@gmail.com"]

["3@gmail.com", 2@gmail.com","1@gmail.com"]

["4@gmail.com, "3@gmail.com", 2@gmail.com","1@gmail.com"]

FIFO - First in First Out

Armazenando Listas

As listas permitem armazenar um conjunto de valores em uma mesma chave.
Exemplos: Ultimas 10 notícias de um site; Ranking de pontuação de um jogo.

```
$ LSET -> Insere elemento em determinada posição
```

```
$ LINDEX -> Retorna o elemento de determinado índice da lista
```

```
$ LLEN -> Recupera o tamanho da lista
```

```
$ LREM -> Remove N elementos à esquerda
```

```
$ RREM -> Remove N elementos à direita
```

Atenção: O primeiro elemento da lista possui índice zero.

Armazenando Listas

As listas permitem armazenar um conjunto de valores em uma mesma chave.
Exemplos: Ultimas 10 notícias de um site; Ranking de pontuação de um jogo.

```
$ LRANGE –> Recupera elementos em um dado range
```

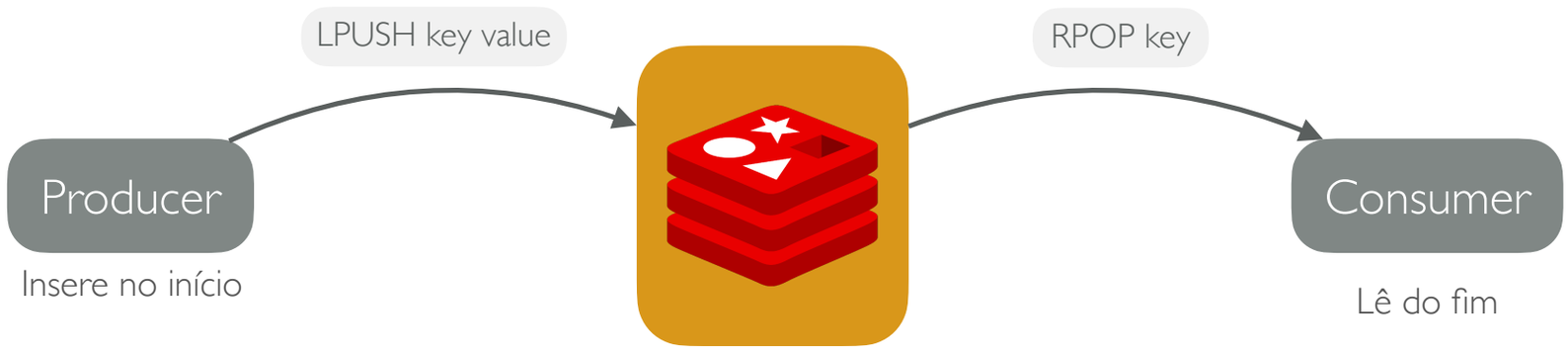
```
$ LRANGE key 0 -1 (Recupera todos elementos da lista)
```

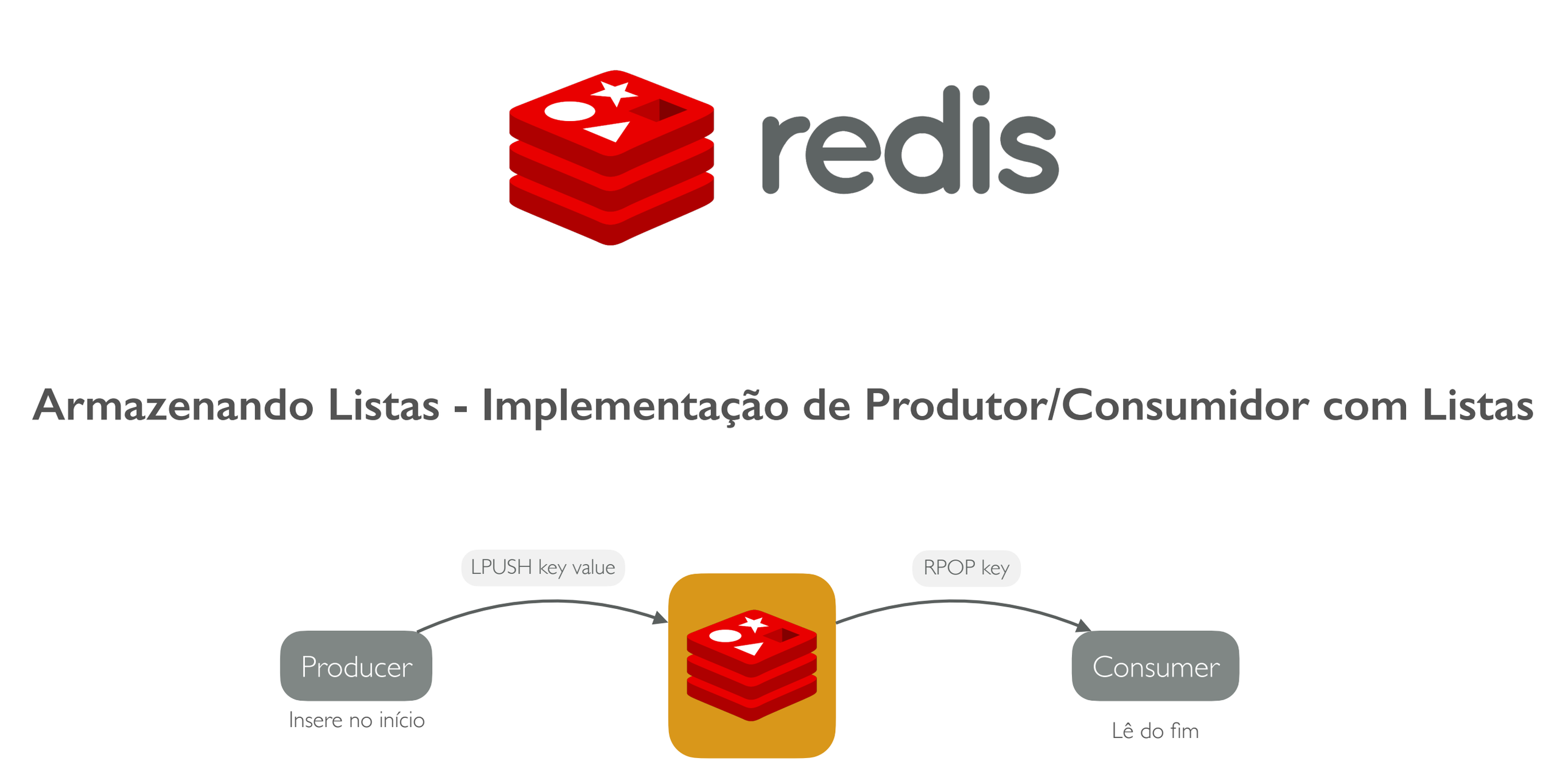
-1 significa ultimo elemento, -2 penúltimo, -3 antepenúltimo…

```
$ LTRIM –> Remove elementos em um dado range
```

Atenção: O primeiro elemento da lista possui índice zero.

Armazenando Listas - Implementação de Produtor/Consumidor com Listas



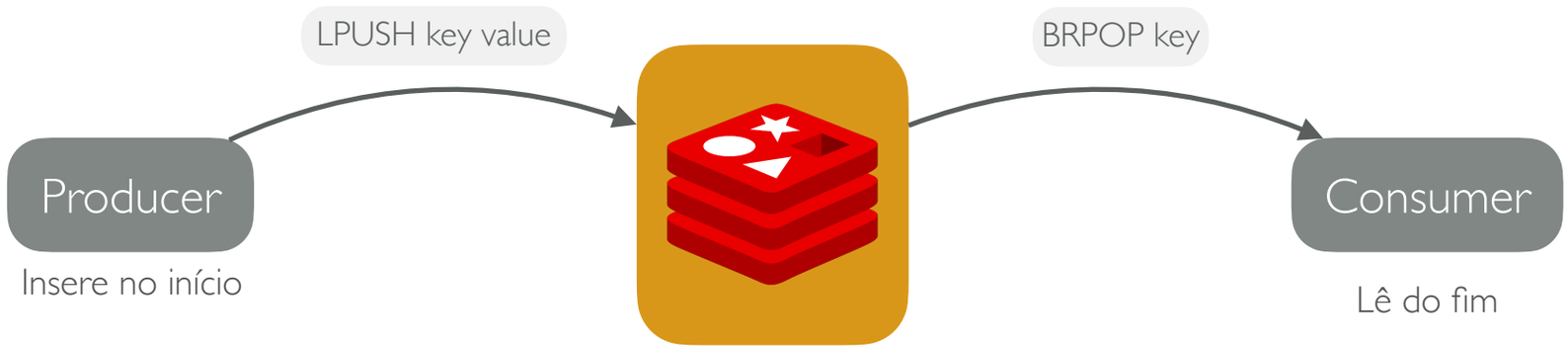


Armazenando Listas - Implementação de Produtor/Consumidor com Listas

O que é retornado para o consumer caso a lista esteja vazia?

É retornado `nil`. Ou seja, o consumer precisa ficar checando continuamente se há novos elementos. Esse processo chama-se **espera ocupada.**

Armazenando Listas - Implementação de Produtor/Consumidor com Listas




```
$ BLPOP key timeout –> Remove e retorna um elemento à
esquerda (início) e aguarda caso não haja elementos.
```

```
$ BRPOP key timeout –> Remove e retorna um elemento à
direita (início) e aguarda caso não haja elementos.
```

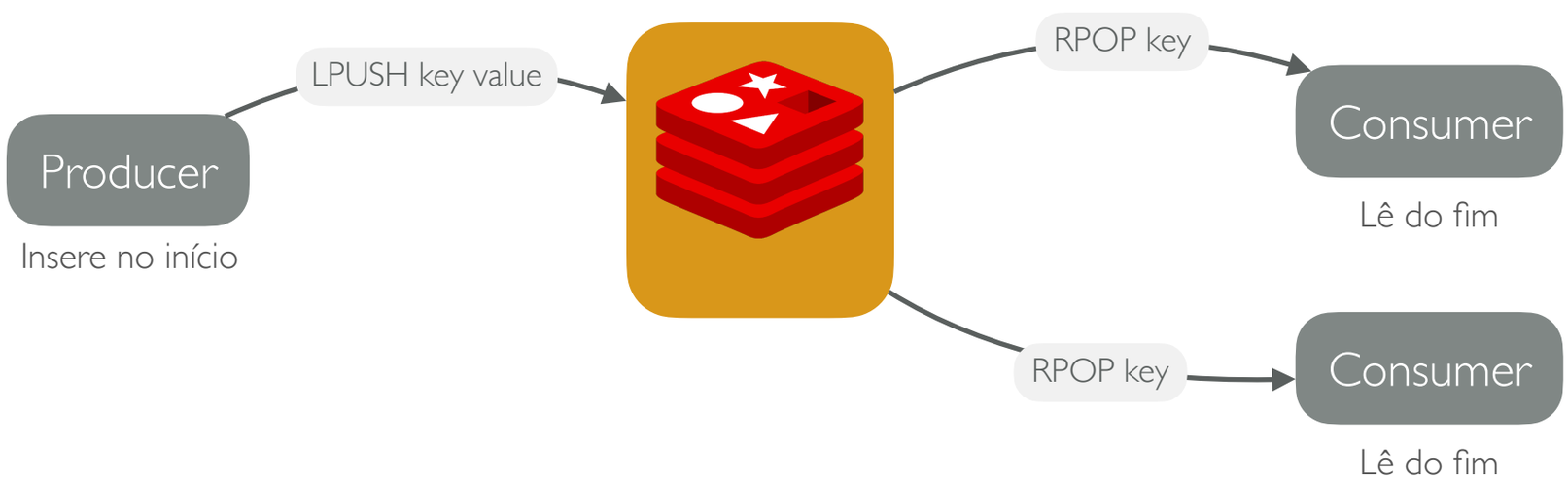
Atenção: Timeout zero significa esperar indefinidamente.

Exercício:

Implemente um programa que envie notificações a clientes através do uso de listas do REDIS bloqueantes. Experimente utilizar mais de um consumidor (cliente) para uma lista. O que ocorre nesse caso?

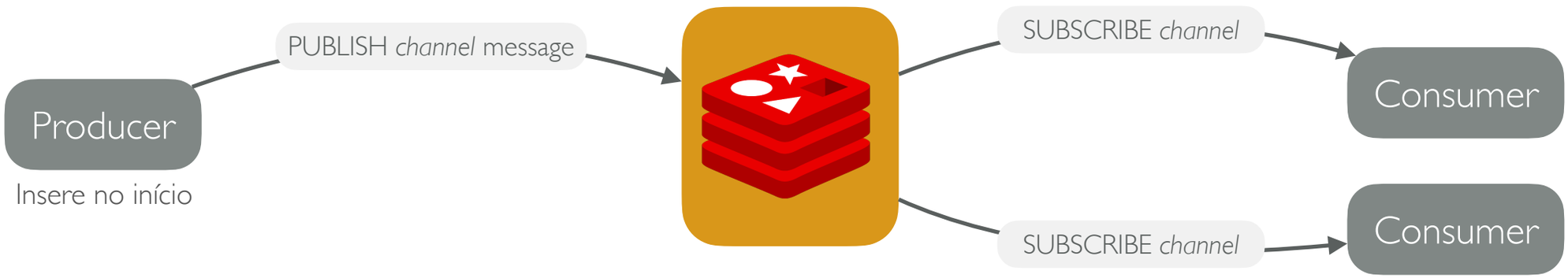
Pub/Sub

Implementação de Publish/Subscribe



Neste cenário apenas um dos consumidores receberá a mensagem. Pois o primeiro a ler, também remove o dado da lista. **E se quisermos implementar um sistema de notificação onde todos os clientes sejam avisados?**

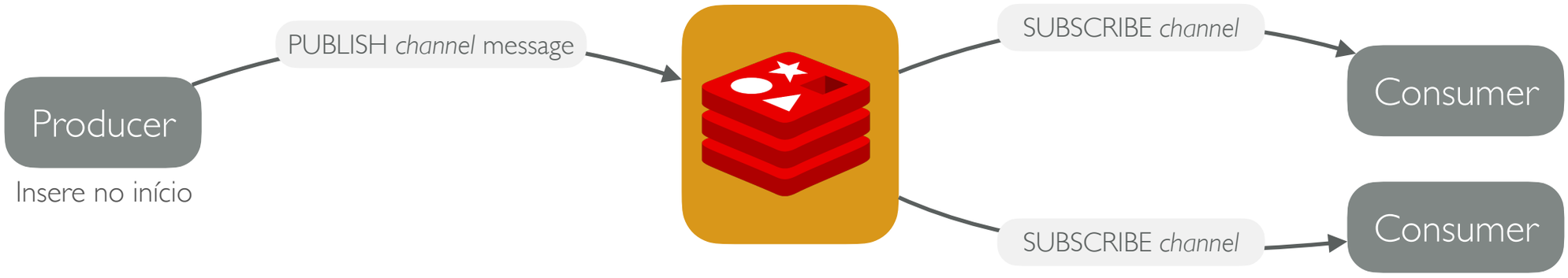
Implementação de Publish/Subscribe



O Redis possui um conjunto de instruções para implementação do padrão PUB/SUB. Nesse caso, todos os clientes são notificados de uma nova mensagem.

redis

Implementação de Publish/Subscribe




```
$ PUBLISH channel message -> Publica uma mensagem em um canal
```

```
$ SUBSCRIBE channel [channel2] -> Assina um ou mais canais para ser
notificado, em caso de mensagem.
```

```
$ PSUBSCRIBE channel* -> Assina todos os canais que satisfazem um padrão.
```

Exercício:

Implemente um sistema de notificações com o Redis que satisfaça os seguintes critérios.

A) Publique mensagens nos canais "eventos-restrito" e "eventos-gerais"
B) Crie um cliente que consegue ler todas as notificações de eventos e imprima na tela
C) Crie outro cliente que só consegue notificações restritas

Obrigado!

Prof. Gustavo Leitão