Redis

Jeremias Kalebe
Debora Brito

Junho 2019

1 Introdução

O Redis é um open source(licenciado pelo BSD), armazenamento de estrutura de dados na memoria, usado como banco de dados, cache e message broker. Ele suportar diferentes estruturas de dados, como String, Streams,bitfilds, geoespatial index, bitmaps, hyperloglogs, hashes, sorted sets, set e lists. Redis significa REmote DIctionary Server. O Redis realiza o armazenamento de chave-valor, ou seja, os dados são armazenados como pares de chave-valor e recuperados por chaves.

1.1 Portatil

Escrito na linguagem C, mantido por uma comunidade de colaboradores disponibilizado no GitHub. E funciona na maioria dos sistemas POSIX, como Linux, BSD, Mac OS X, Solaris e assim por diante. Vale ressaltar que o Redis não suporta oficialmente Windows.

2 Conhecendo o Redis

2.1 Redis comparado a outros bancos de dados e softwares

Não é incomum ouvir o Redis comparado ao memcached, que é um servidor de cache de valor-chave de alto desempenho. Como o memcached, o Redis também pode armazenar um mapeamento de chaves para valores e pode até atingir níveis de desempenho semelhantes aos do memcached. Mas as semelhanças terminam rapidamente - o Redis suporta a gravação de seus dados no disco automaticamente de duas maneiras diferentes, e pode armazenar dados em quatro estruturas, além de teclas simples como o memcached. Essas e outras diferenças permitem que o Redis resolva uma variedade maior

de problemas e permita que o Redis seja usado como um banco de dados principal ou como um banco de dados auxiliar com outros sistemas de armazenamento.

3 Estrutura do Redis

O Redis possui cinco tipos diferentes de estrutura de dados; STRINGs, LISTs, SETs, HASHes, e ZSETs, e cada uma das cinco estruturas tem alguns comandos compartilhados (DEL, TYPE, RENAME e outros).

3.1 Strings

Em Redis, Strings são semelhantes às cadeias de caracteres que outros idiomas possuem, ou em outros armazenamentos de chave-valores. Geralmente, os diagramas têm o nome da chave e o tipo do valor ao longo da parte superior de uma caixa, com o valor dentro da caixa.

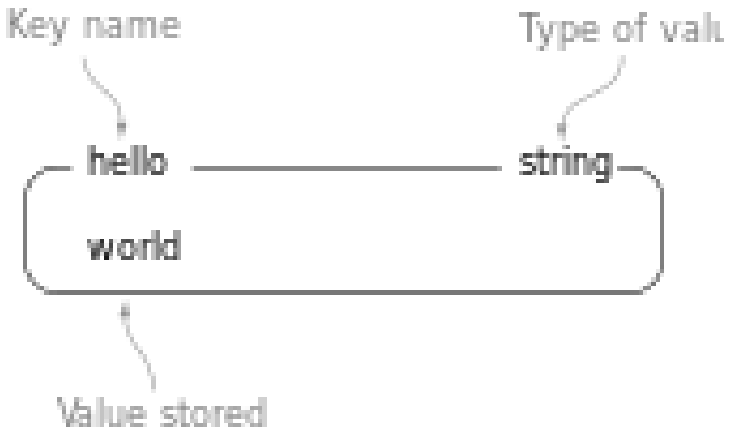

Figure 3.1: figura A

Strings possui alguns comandos como o GET (que permite buscar os dados armazenados na chave dada), o SET (que define o valor armazenado na chave dada) e o DEL (que exclui o valor armazenado na chave fornecida). Além dos comandos GET, SET e DEL, há um punhado de outros comandos para ler e escrever partes de STRINGs, e comandos que nos permitem tratar strings como números para in-

crementar / decrementar.

COMANDO	O QUE FAZ
GET	Busca os dados armazenados na chave dada
SET	Define o valor armazenado na chave dada
DEL	Exclui o valor armazenado na chave fornecida (funciona para todos os tipos)

Table 1: Comandos

3.2 Lists

No mundo das lojas de chave-valores, o Redis é único, pois suporta uma estrutura de lista vinculada. Listas no Redis armazenam uma sequência ordenada de strings. Os comandos que podem ser executadas em Listas são típicos do que se encontra em quase qualquer linguagem de programação.

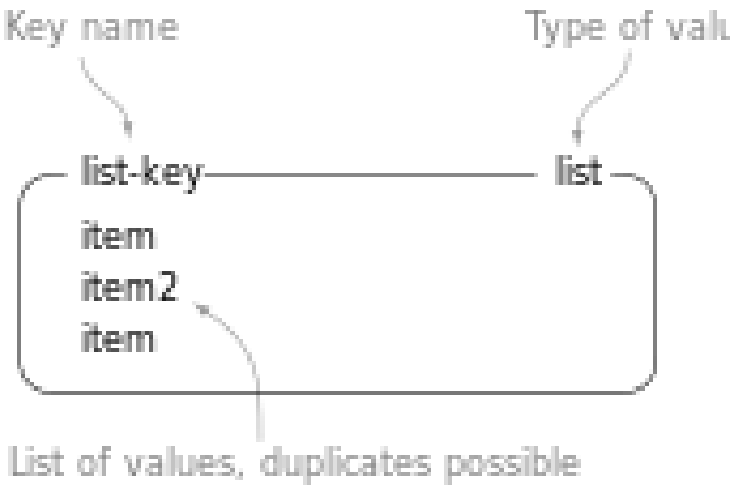

Figure 3.2: Lists

Alguns desses comandos são: RPUSH (que empurra o valor para o lado direito da lista), LRANGE (que busca um intervalo de valores da lista), LINDEX (que obtém um item em uma determinada posição na lista) e LPOP (Pops o valor da extremidade esquerda da lista e retorna). Também é possível remover itens, inserir itens no meio, aparar a lista para ter um tamanho específico (descartando itens de uma ou ambas as extremidades) e muito mais.

COMANDO	O QUE FAZ
RPUSH	Empurra o valor para o lado direito da lista
LRANGE	Busca um intervalo de valores da lista
LINDEX	Obtém um item em uma determinada posição na lista
LPOP	Pops o valor da extremidade esquerda da lista e retorna

Table 2: Comandos

3.3 SETs

Em Redis, SETs são semelhantes a LISTs em que são uma seqüência de caracteres, mas ao contrário LISITs, Redis SETs usa uma tabela de hash para manter todas as strings exclusivas (embora não haja valores associados).

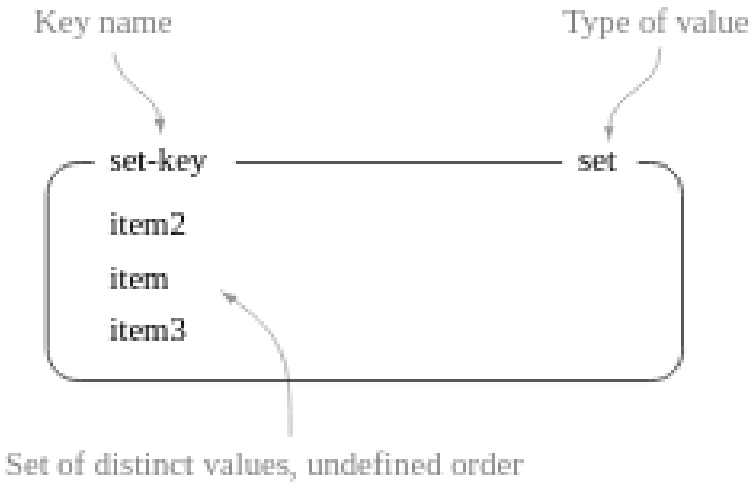

Figure 3.3: Sets

SETs também possui alguns comandos como SADD (que adiciona o item ao conjunto), SMEMBERS (que retorna o conjunto inteiro de itens), SISMEMBER (que verifica se um item está no conjunto) e SREM (que remove o item do conjunto, se existir). Além dessas operações, SETs possui também comandos de incluem intersecção, união e diferença (SINTER, SUNIONe SDIFF, respectivamente).

COMANDO	O QUE FAZ
SADD	Adiciona o item ao conjunto
SMEMBERS	Retorna o conjunto inteiro de itens
SISMEMBER	Verifica se um item está no set
SREM	Remove o item do conjunto, se existir

Table 3: Comandos

3.4 Hashes

Enquanto LISTs e SETs em Redis possui uma seqüências de itens, Redis Hashes armazenam um mapeamento de chaves para valores. Os valores que podem ser armazenados em Hashes são os mesmos que podem ser armazenados em STRINGs: strings próprias, ou se um valor puder ser interpretado como um número, esse valor pode ser incrementado ou decrementado.

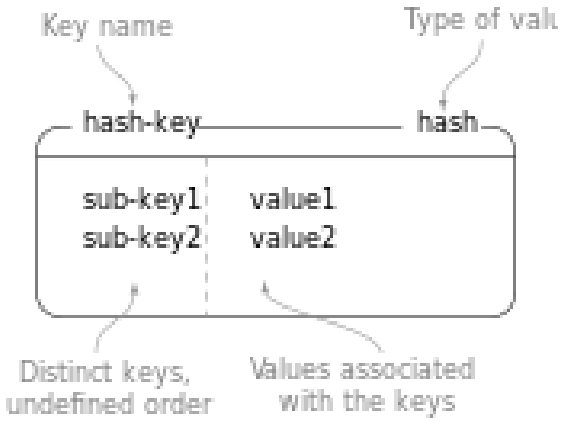

Figure 3.4: Hashes

Em HASHEs existe comandos para inserir, buscar e remover itens. Esses comandos são HSET (que armazena o valor na chave no hash), HGET (que busca o valor na chave de hash especificada), HGETALL (que busca o hash inteiro) e o HDEL (que remove uma chave do hash, se existir).

COMANDO	O QUE FAZ
HSET	Armazena o valor na chave no hash
HGET	Busca o valor na chave de hash
HGETALL	Busca o hash inteiro
HDEL	Remover uma chave do hash, se existir

Table 4: Comandos

3.5 ZSET (Sorted)

ZSETs também possui um tipo de chave e valor. As chaves (chamadas membros) são exclusivas e os valores (chamados pontuações) são limitados a números de ponto flutuante.ZSETs tem a propriedade única no Redis de poder ser acessado pelo membro (como um HASHE), mas os itens também podem ser acessados pela ordem classificada e pelos valores das pontuações. Figura 3.5 mostra um exemplo ZSET com dois itens.

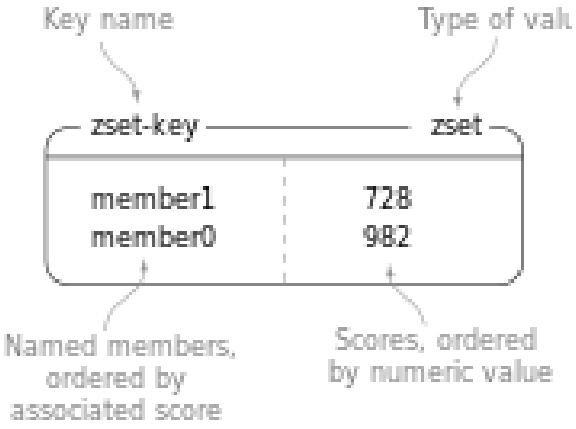

Figure 3.5: Sorted

Como é o caso de todas as outras estruturas, precisamos adicionar, remover e buscar itens de ZSETs. A Listagem 1.5 oferece comandos add, remove e fetching paraZSETé semelhante àquelas das outras estruturas, e a tabela 1.7 descreve os comandos que usaremos.

COMANDO	O QUE FAZ
ZADD	Adiciona um membro com a pontuação dada ao ZSET
ZRANGE	Busca os itens no ZSET de suas posições na ordem de classificação
ZRANGEBYSCORE	Busca itens no ZSET com base em uma série de pontuações
ZREM	Remove o item do ZSET, se existir

Table 5: Comandos

4 Popularidade

Como o Redis é open-source, podemos encontrar uma vasta documentação, exemplos e muita colaboração de foruns.
Nos últimos anos, muitas APIs foram desenvolvidas para uma variedade incrivelmente ampla de linguagens de programação, tornando o Redis uma escolha fácil para desenvolvedores.
O site Atlantic.net mostra alguns motivos da popularidade do Redis:

- Forças básicas de Redis
- Construído para velocidade e idiomas
- A multidão diz sim
- Processamento em memória para mídia
- Entrega da tabela de classificação
- Servidores Cloud Redis-Ready

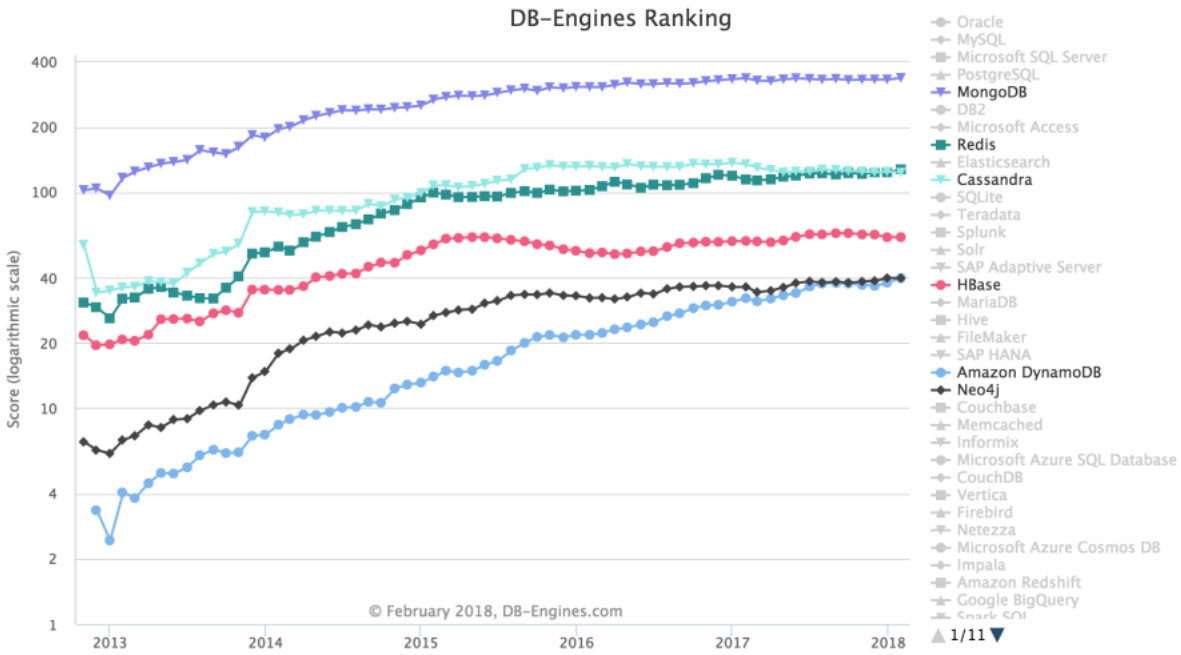

Figure 4.1: Ranking Banco de Dados

Como mostrado na figura 4.1, o Redis tem sido o segundo mais usado pelos desenvolvedores.

5 Vantagens

Extremamente rápido, tanto para escrita como para leitura, devido o armazenamento dos dados ser na memoria, como por a memoria RAM. Possui um servidor TCP, baseado em um modelo cliente-servidor. Os comandos são atômicos, devido usar single-threaded. Ou seja, enquanto um comando é executado, outros não podem operar.

6 Desvantagens

- Vale ressaltar que o conjunto de dados inteiro sempre reside na RAM com isso, deve ser bem pensado seu uso.

- As soluções de clustering para Redis devem ser implementadas internamente e requerem um esforço considerável. O Redis Labs fornece uma solução Redis As Service baseada em nuvem.
- A persistência afeta o desempenho. O Redis usa o despejo de memória para criar um instantâneo de persistência (o que é uma ótima ideia). Mas isso requer alguns ajustes no kernel do Linux para evitar a degradação do desempenho enquanto o processo do servidor Redis está se forking.
- Problemas de fragmentação de memória. Escrever e excluir grandes quantidades de dados pode resultar em degradação do desempenho.
- O gerenciamento de chaves requer um esforço considerável. Não existe uma maneira fácil de executar algo como SELECT COUNT (*) FROM REDIS WHERE KEY LIKE '* key *'. As versões mais recentes do Redis são equipadas com um comando SCAN que facilita a vida.
- O gerenciamento de TTL requer uma disciplina no nível do aplicativo. Periodicamente, você pode acabar com o servidor Redis sobrecarregado pelos dados obsoletos porque algumas chaves não têm TTL.

7 Aplicações

O redis é empregado em muitos usos práticos, como:

- Analise em tempo real
- Armazenamento de sessão do Usuário
- Injeção de alta velocidade
- Job e Queue Management
- Dados de series temporais
- Analise estatísticas complexa
- Notificações

- Bloqueio distribuído
- Cache de conteúdo
- Dados
- Streaming data
- Aprendizado de Maquina

8 Desempenho

De acordo com o site Octalmind, o Redis fornece um desempenho melhor que o Memcached e o MongoDB para a operação de leitura e também quando se trata de excluir chaves. **?** Dentre os muitos desempenho do Redis, destacaremos cinco:

- Desempenho muito rápido
  Todos os dados do Redis reside na memoria principal do seu servidor. Com isso, evitam atraso de tempo de busca e pode acessar os dados com algoritmos mais simples que usam menos instruções da CPU
- Estruturas de dados na memória
  Praticamente qualquer tipo de dados, como foi abordado na seção 3, é suportado pelo Redis.
- Versatilidade e facilidade de uso
  O Redis emprega uma arquitetura no estilo mestre/subordinado e é compatível com a replicação assíncrona em que os dados podem ser replicados para vários servidores subordinados. Isso pode disponibilizar desempenho de leitura melhorado (à medida que as solicitações podem ser divididas entre os servidores) e recuperação quando o servidor primário passar por uma interrupção.

  Para disponibilizar durabilidade, o Redis oferece compatibilidade com snapshots point-in-time (copiando o conjunto de dados do Redis no disco) e criando um Append Only File (AOF) para armazenar cada alteração de dados no disco conforme elas vão sendo gravadas. Os dois métodos permitem a restauração rápida dos dados do Redis no caso de uma interrupção.

- Replicação e persistência
  O Redis é disponibilizado com várias ferramentas que tornam o desenvolvimento e as operações mais rápidas e fáceis, inclusive o PUB/SUB para publicar mensagens nos canais que são entregues para os assinantes, o que é ótimo para sistemas de mensagens e chat

- Compatibilidade com a sua linguagem de desenvolvimento preferencial
  Mais de cem clientes de código aberto estão disponíveis para os desenvolvedores do Redis. As linguagens compatíveis incluem Java, Python, PHP, C, C++, C#, JavaScript, Node.js, Ruby, R, Go e muitas outras.

References