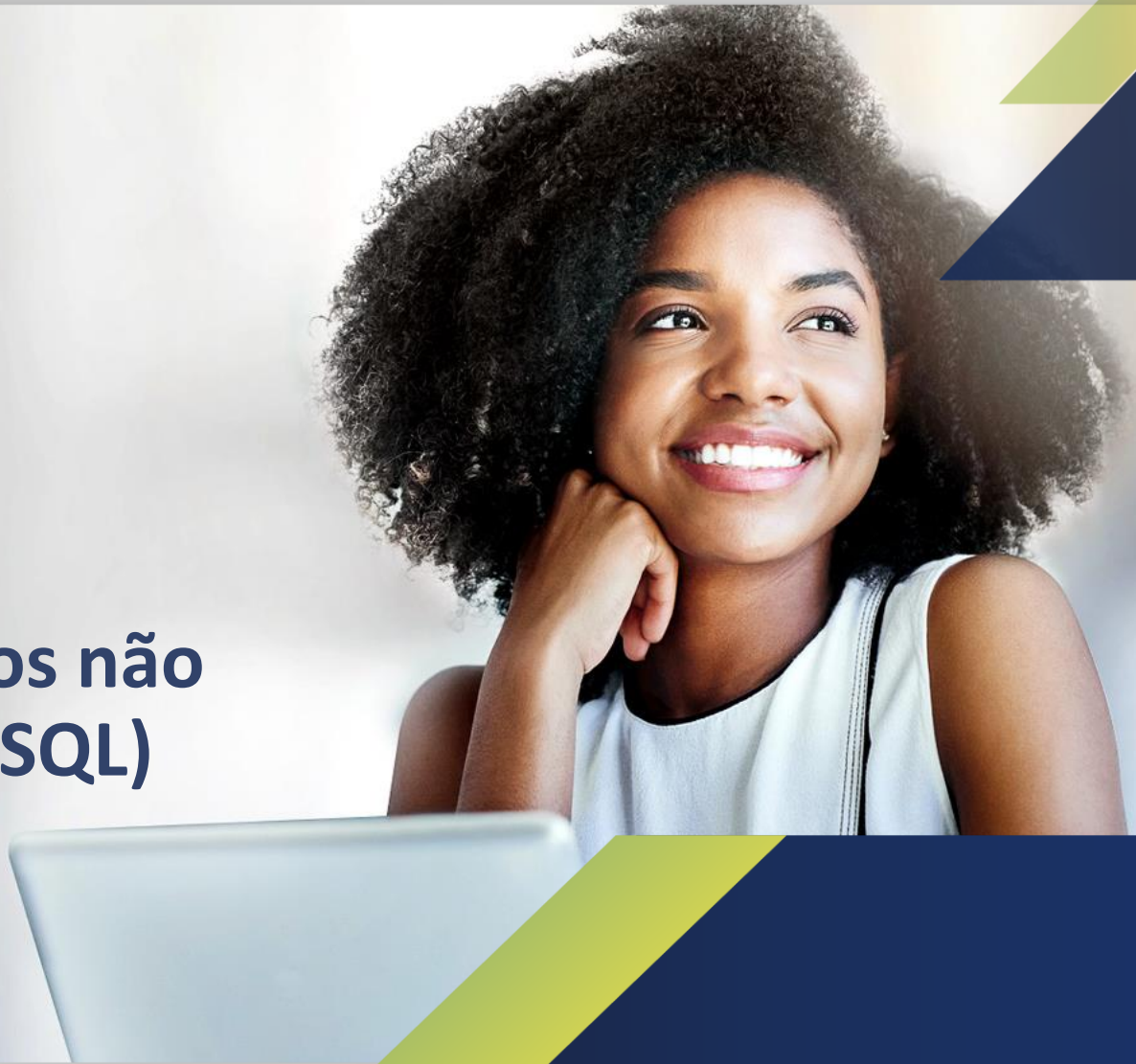


Banco de dados não relacional (NoSQL)



Modelo orientado a chave/ valor


Bloco 1

Marcelo Tavares de Lima





► Objetivos

- Compreender as diferenças entre bancos relacionais e não relacionais.
 - Compreender os conceitos de chave/ valor em bancos não relacionais.
 - Diferenciar as estruturas de armazenamento entre bancos relacionais em detrimento aos não relacionais.
- 




► O que veremos nesta aula?

- Algumas características de bancos de dados não relacionais.
- Comparativo entre bancos não relacionais e bancos relacionais.
- Quebra de paradigmas (padrões).



► Introdução

- Bancos não relacionais quebram os paradigmas pregados pelo SQL, onde a estrutura de armazenamento de dados é feita em tabelas (entidades) que se relacionam entre si.
 - O armazenamento de dados em bancos não relacionais é feito de diversas formas.
 - Nesta aula, veremos o armazenamento chave/valor.
- 

► Introdução

Quadro 1 - Diferenças entre SQL e NoSQL

SQL	NoSQL
Esquema baseado em tabelas com definição de tipos de dados.	A definição dos tipos de dados ocorre em tempo de execução (conceito <i>on the fly</i>).
Tem normalização de tabelas (formas normais), com foco na eliminação de redundância e economia de espaço.	Não há especificação para este critério.
Utilização de junção (<i>joins</i>) de dados.	Não possui junção de dados, considerando que não há tabelas para se relacionarem entre si – embora possam existir referências entre coleções, por exemplo.
Possui maior nível de segurança, mas ocasiona mais lentidão em execução.	Possui menor segurança, porém, é mais rápido.
Tem como uma de suas características a evolução vertical, que requer maior estrutura de <i>hardware</i> de acordo com a demanda de dados.	Possui evolução horizontal, permitindo a criação de ambientes de armazenamento baseados em <i>cluster</i> , particionando os dados em vários servidores distintos.
Sistemas que utilizam bancos relacionais, geralmente, possuem estruturas orientadas a dados, ou seja, focam seu escopo nos dados que podem ser recebidos, o que os torna menos adaptáveis a mudanças.	É mais versátil e tem maior adaptação a mudanças. É mais customizável e dinâmico para sistemas propensos a mudanças.

Fonte: elaborado pelo autor.




► Modelo chave/ valor

- A abordagem de um banco de dados baseado em chave/ valor é um dos modelos mais simples que podem existir para manipular dados de maneira estruturada, com baixo consumo de memória para armazenar e baixo custo de processamento em consultas.




► Modelo chave/ valor

- Todo dado armazenado está ligado a uma chave.
 - Cada chave possui seu valor: string, inteiro, booleano, imagens, objetos Json etc.
 - As chaves são únicas na extensão do banco de dados inteiro, ou seja, não permitem duplicação.
- 



► Modelo chave/ valor

- Exige-se que haja um atrelamento a uma forte camada de consistência de dados a ser realizada na aplicação que manipulará o banco de dados.
 - Pode soar estranho, no entanto, faz sentido quando entendemos que a função do banco é de armazenamento e não gerenciar regras.
- 



► Modelo chave/ valor

- Máxima na computação: quanto maior a velocidade de uma aplicação, menor sua complexidade e vice versa.
- Esta máxima é, de certa forma, superada pelos bancos não relacionais, por serem mais rígidos em suas estruturas.



► Modelo chave/ valor

- Embora não seja uma regra, bancos NoSQL têm mais aceitabilidade em ambientes que demandam maior fluxo de dados dinâmicos, como *e-commerce*.
- Também são muito utilizados em sistemas de grande porte e de alto fluxo, como as plataformas de redes sociais.

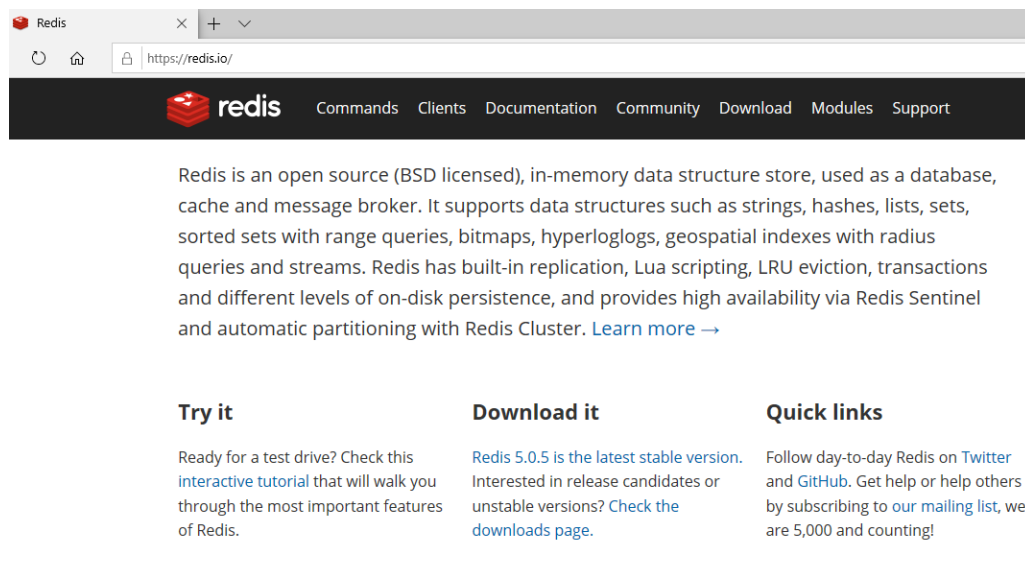


► Conhecendo o Redis

- É uma estrutura de armazenamento de dados de chave-valor do servidor, que possui suporte a diversos tipos de dados: strings, listas, conjuntos, *hash*, *bitmap*, dentre outros.
- Dada a simplicidade, é uma das estruturas mais utilizadas em NoSQL.

► Conhecendo o Redis

Figura 1 - Página do Redis



Redis News

Fonte: elaborado pelo autor.

Modelo orientado a chave/ valor

Bloco 2

Marcelo Tavares de Lima



► Usando o Redis

Figura 2 - Modelo de chave/ valor em NoSQL

CHAVE	VALOR
usuario	José
usuario2	Pedro
idade	38
casado	true

Fonte: elaborado pelo autor.

- Não há uma menção dos tipos de dados envolvidos para cada entrada.
- Existe somente a chave e seu respectivo valor.
- Esta é uma característica do modelo NoSQL: a dispensa explícita de tipos de dados, o que torna o manuseio da estrutura muito mais dinâmico.



► Usando o Redis

- Os dados são reconhecidos previamente pela linguagem.
- Se forem inseridos números nos campos, serão considerados como campos numéricos, permitindo, assim, operações incrementais.
- Outros tipos de campos (string, booleano etc.) também são reconhecidos previamente.



► Usando o Redis

- Para iniciar o Redis, executar o comando `redis-cli.exe`.
- O prompt de comando será aberto.

► Usando o Redis

Figura 3 - Uso de comandos no Redis



```
127.0.0.1:6379> SET usuario Jose  
OK  
127.0.0.1:6379> GET usuario  
"Jose"  
127.0.0.1:6379>
```

Fonte: elaborado pelo autor.

► Usando o Redis

Figura 4 - Uso de comandos no Redis

 C:\Users\mtav\Downloads\Redis-x64-3.2.100\redis-cli.exe

```
127.0.0.1:6379> SET Idade 48
OK
127.0.0.1:6379> GET Idade
"48"
127.0.0.1:6379>
```

Fonte: elaborado pelo autor.

► Usando o Redis

Figura 5 - Uso de comandos no Redis

 C:\Users\mtav\Downloads\Redis-x64-3.2.100\redis-cli.exe

```
127.0.0.1:6379> SET Idade 48
OK
127.0.0.1:6379> GET Idade
"48"
127.0.0.1:6379> SET Idade 55
OK
127.0.0.1:6379> GET Idade
"55"
127.0.0.1:6379> DEL Idade
(integer) 1
127.0.0.1:6379> GET Idade
(nil)
127.0.0.1:6379>
```

Fonte: elaborado pelo autor.

► Usando o Redis

Figura 6 - Uso de comandos no Redis

```
127.0.0.1:6379> MSET nome Jose sobrenome Antonio idade 48 sexo M casado true
OK
127.0.0.1:6379> MGET nome sobrenome idade sexo casado
1) "Jose"
2) "Antonio"
3) "48"
4) "M"
5) "true"
127.0.0.1:6379>
```

Fonte: elaborado pelo autor.

PÓS-GRADUAÇÃO

Teoria em prática

Bloco 3

Marcelo Tavares de Lima





► Usando o Redis

- Suponha que você seja um *Data Base Administrator (DBA)*, precisou utilizar uma estrutura NoSQL para armazenar os dados de uma rede social, optou por utilizar o Redis como plataforma de gerenciamento e usou o Modelo chave/ valor.



► Usando o Redis

- Considerando a versatilidade desse modelo de banco de dados, elabore um escopo para armazenar os seguintes dados: nome e sobrenome de usuário; telefone; e-mail; sexo; idade; data de nascimento; estado civil; time de futebol; preferências culinárias; e partido político.



► Usando o Redis

- Após concluir o armazenamento de pelo menos três usuários, liste todos os dados na tela em forma sequencial. Uma dica é recorrer à documentação do Redis para ter acesso a todas as sintaxes disponíveis.

Dica do professor

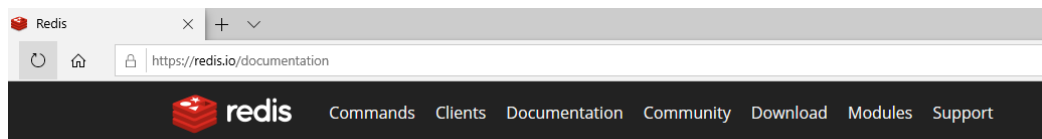
Bloco 4

Marcelo Tavares de Lima



► Documentação Redis

Figura 7 - Documentação do Redis



Documentation

Note: The Redis Documentation is also available in raw (computer friendly) format in the [redis-doc github repository](#). The Redis Documentation is released under the [Creative Commons Attribution-ShareAlike 4.0 International license](#).

Programming with Redis

- [The full list of commands](#) implemented by Redis, along with thorough documentation for each of them.
- [Pipelining](#): Learn how to send multiple commands at once, saving on round trip time.
- [Redis Pub/Sub](#): Redis is a fast and stable Publish/Subscribe messaging system! Check it out.
- [Redis Lua scripting](#): Redis Lua scripting feature documentation.
- [Debugging Lua scripts](#): Redis 3.2 introduces a native Lua debugger for Redis scripts.
- [Memory optimization](#): Understand how Redis uses RAM and learn some tricks to use less of it.
- [Expires](#): Redis allows to set a time to live different for every key so that the key will be automatically removed from the server when it expires.
- [Redis as an LRU cache](#): How to configure and use Redis as a cache with a fixed amount of memory and auto eviction of keys.
- [Redis transactions](#): It is possible to group commands together so that they are executed as a single transaction.
- [Mass insertion of data](#): How to add a big amount of pre existing or generated data to a Redis instance in a short time.

Fonte: elaborado pelo autor.



► Referências

TOTH, Renato Molina. **Abordagem NoSQL – uma real alternativa**. [s.l.]: Universidade Federal de São Carlos – Campus Sorocaba, 2011. Disponível em: <https://dcomp.sor.ufscar.br/verdi/topicosCloud/nosql_artigo.pdf>. Acesso em: 24 out. 2019.

