

# Banco de Dados NoSQL

Igor Moraes, Lucas Fonseca, Marco Aurélio

Universidade Federal de Lavras

5 de outubro de 2018

# Índice

- 1 Introdução Banco de Dados NoSQL
- 2 Redes Sociais e Bancos de Dados NoSQL
- 3 Características dos Banco de Dados NoSQL
- 4 Por que usar Banco de Dados NoSQL?
- 5 Modelos de Dados:
  - Chave-Valor
    - Redis
  - Documento
    - MongoDB
  - Orientado à Coluna
    - Cassandra
  - Grafo
    - Neo4J
- 6 Comparativo dos Principais SGBD's NoSQL
- 7 Conclusão

**NoSQL**  
Not Only SQL

# Introdução Banco de Dados NoSQL

- O termo foi usado inicialmente como nome de um banco de dados não relacional de código aberto.
- É completamente distinto do modelo relacional e portanto deveria ser mais apropriadamente chamado "NoREL" ou algo que produzisse o mesmo efeito.
- Referência aos bancos de dados relacionais mais populares MySQL, Microsoft SQL Server, PostgreSQL.

*Not*      *Only*

**NoSQL**

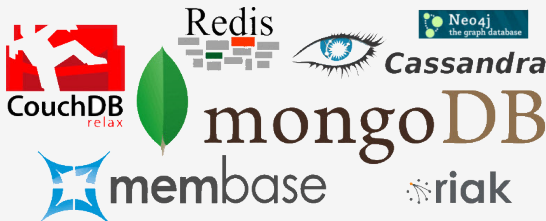
*Relational*

# Redes Sociais e Bancos de Dados NoSQL



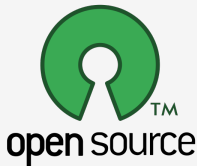
# Características dos Banco de Dados NoSQL

- Utilização do processamento paralelo para processamento das informações.
- Distribuição em escala global.
- Diversos tipos para diferentes aplicações.



# Por que usar Banco de Dados NoSQL?

- Flexibilidade
- Escalabilidade
- Disponibilidade
- Raízes Open Source
- Baixo Custo Operacional
- Funcionalidades Especiais



1

# Modelos de Dados NoSQL

## Key Value



Example:  
Riak, Tokyo Cabinet, Redis  
server, Memcached, Scalaris

## Document-Based



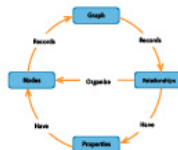
Example:  
MongoDB, CouchDB,  
OrientDB, RavenDB

## Column-Based



Example:  
BigTable, Cassandra, Hbase,  
Hypertable

## Graph-Based



Example:  
Neo4J, InfoGrid, Infinite  
Graph, Flock DB

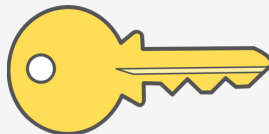
# Principais SGBD's do mercado devido à popularidade

Rank			DBMS	Database Model	Score		
Oct 2018	Sep 2018	Oct 2017			Oct 2018	Sep 2018	Oct 2017
1.	1.	1.	Oracle +	Relational DBMS	1319.27	+10.15	-29.54
2.	2.	2.	MySQL +	Relational DBMS	1178.12	-2.36	-120.71
3.	3.	3.	Microsoft SQL Server +	Relational DBMS	1058.33	+7.05	-151.99
4.	4.	4.	PostgreSQL +	Relational DBMS	419.39	+12.97	+46.12
5.	5.	5.	MongoDB +	Document store	363.19	+4.39	+33.79
6.	6.	6.	DB2 +	Relational DBMS	179.69	-1.38	-14.90
7.	↑ 8.	↑ 9.	Redis +	Key-value store	145.29	+4.35	+23.24
8.	↓ 7.	↑ 10.	Elasticsearch +	Search engine	142.33	-0.28	+22.09
9.	9.	↓ 7.	Microsoft Access	Relational DBMS	136.80	+3.41	+7.35
10.	10.	↓ 8.	Cassandra +	Wide column store	123.39	+3.83	-1.40
11.	11.	11.	SQLite +	Relational DBMS	116.74	+1.29	+4.76
12.	12.	12.	Teradata +	Relational DBMS	78.63	+1.25	-1.45
13.	13.	↑ 16.	Splunk	Search engine	76.90	+2.87	+12.54
14.	14.	↑ 18.	MariaDB +	Relational DBMS	73.13	+2.49	+16.73
15.	15.	↓ 13.	Solr	Search engine	61.31	+1.11	-9.82
16.	16.	↑ 19.	Hive +	Relational DBMS	61.10	+1.47	+9.66
17.	17.	↓ 15.	HBase +	Wide column store	60.67	+2.21	-3.72
18.	18.	↓ 14.	SAP Adaptive Server +	Relational DBMS	58.57	+0.52	-8.67
19.	19.	↓ 17.	FileMaker	Relational DBMS	56.04	+0.74	-5.02
20.	20.	↑ 22.	Amazon DynamoDB +	Multi-model	54.47	+1.12	+16.87
21.	21.	↓ 20.	SAP HANA +	Relational DBMS	54.37	+1.64	+4.28
22.	22.	↓ 21.	Neo4j +	Graph DBMS	42.65	+2.56	+4.71
23.	23.	23.	Couchbase +	Document store	35.92	+1.37	+2.55
24.	24.	↑ 25.	Memcached	Key-value store	30.56	-0.98	+2.92



# Chave-Valor

- Armazena dados como um conjunto de pares de chave-valor.
- A chave funciona como um identificador exclusivo.
- A chave e os valores podem ser desde objetos simples até objetos compostos complexos.
- Principal SGBD: Redis.



# Redis

- **RE**mote **D**ictionary **S**erver
- Criado por Salvatore Sanfiippo;
- Escrito na Linguagem C;
- Open-source (Licença BSD);
- Ideia de Hash/Dicionário em LP's
- Dados armazenados na memória;
- Comandos atômicos;
- Single-threaded;
- Modelo cliente-servidor (TCP);
- Alta performance para gravação e/ou leitura de dados
- Interação através de comandos, não há linguagem de consulta semelhante ao SQL;



# Exemplo Redis

```
1 | redis 127.0.0.1:6379> set page registration
2 | OK
3 | redis 127.0.0.1:6379> keys *
4 | 1) "foo"
5 | 2) "page"
6 | redis 127.0.0.1:6379> get page
7 | "registration"
```

Figura: Comandos básicos do Redis

# Exemplo Redis

```
1 jedis.set("gone", "daddy, gone");
2 jedis.expire("gone", 10);
3 String there = jedis.get("gone");
4 assert there.equals("daddy, gone");
5
6 Thread.sleep(4500);
7
8 String notThere = jedis.get("gone");
9 assert notThere == null;
```

Figura: Um valor de Redis pode ser configurado para expirar

# Exemplo Redis

```
1 jedis.rpush("people", "Mary");
2 assert jedis.lindex("people", 0).equals("Mary");
3
4 jedis.rpush("people", "Mark");
5
6 assert jedis.llen("people") == 2;
7 assert jedis.lindex("people", 1).equals("Mark");
```

Figura: Listas do Redis

# Aplicações do Redis

- Aplicações Web e móveis;
- Jogos;
- Tecnologia de anúncios;
- IOT;

## Empresas que usam Redis:

- Twitter;
- GitHub;
- Pinterest;
- Snapchat;
- StackOverflow;
- Flickr;

# Documento

- Armazena os dados semiestruturados como documentos;
- Intuitivo para desenvolvedores por se assimilar ao JSON/XML;
- Cada documento é autodescritível;
- Os documentos são agrupados em “conjuntos”;
- Identificadores únicos universais (UUID);
- Possibilita a consulta de documentos através de métodos avançados de agrupamento e filtragem (MapReduce);



# MongoDB



mongoDB®

- Escrito na linguagem C++;
- Open-source (Licença GNU AGPL 3.0);
- Armazena dados em documentos flexíveis semelhantes a JSON.
- O modelo de documento é mapeado para os objetos no código do seu aplicativo , facilitando o trabalho com os dados;
- Distribuído em seu núcleo, de modo que a alta disponibilidade, o dimensionamento horizontal e a distribuição geográfica são integrados;



# Exemplo MongoDB

```
db.usuarios.insert( {  
    nome: "Higor Medeiros",  
    cidade: "Porto Alegre",  
    estado: "Rio Grande do Sul"  
}  
)
```

Figura: Inserindo os dados

# Exemplo MongoDB

```
db.meudb.save(MeusDados)
```

Figura: Salvando os dados

# Exemplo MongoDB

```
MeusDados = {  
    nome: "Higor Medeiros",  
    cidade: "Porto Alegre",  
    estado: "Rio Grande do Sul"  
}
```

Figura: Conteúdo de MeusDados

# Exemplo MongoDB

```
MeusDados = {  
  nome: "Higor Medeiros",  
  endereco: {cidade: "Porto Alegre", estado: "Rio Grande do Sul"}  
}
```

Figura: Conteúdo de MeusDados em outra armazenagem

# Exemplo MongoDB

```
db.usuarios.remove( { estado: "Rio Grande do Sul" } )
```

Figura: Removendo dados

# Exemplo MongoDB

```
db.usuarios.update( { cidade: { $eq:"Rio de Janeiro" } },  
                    $set:  { estado: "Rio de Janeiro" } },  
                    { multi: true }  
)
```

Figura: Atualizando dados

# Exemplo MongoDB

```
db.usuarios.find(  
    { estado: { $eq: "Rio de Janeiro" } }  
)
```

Figura: Consultando dados

# Aplicações do MongoDB

- Aplicativos de gerenciamento de conteúdo, como blogs e plataformas de vídeo;
- Armazenamento de informações de catálogo, por exemplo: Comércio eletrônico;
- Aplicações em que precisa-se gerencia diversos atributos sem afetar outros;

Empresas que usam MongoDB:

- Globo.com;
- SourceForge;
- FourSquare;
- MailBox(e-mail do Dropbox);
- LinkedIn;



# Orientado à Coluna

- Criado para trabalhar com grande quantidade de dados;
- Trabalha apenas com colunas, cada coluna é um arquivo diferente no disco;
- Buscar por dados é muito mais rápido quando se tem uma grande quantidade de colunas na tabela;
- Compressão de dados maior do que em linha;
- Busca é feita antes da criação das tuplas;



# Linha vs. Coluna

people_id	people_name	people_age
101	Mary	54
102	Jhon	35
103	Paul	22

people\_id

id	value
0	101
1	102
2	103

people\_name

id	value
0	Mary
1	Jhon
2	Paul

people\_age

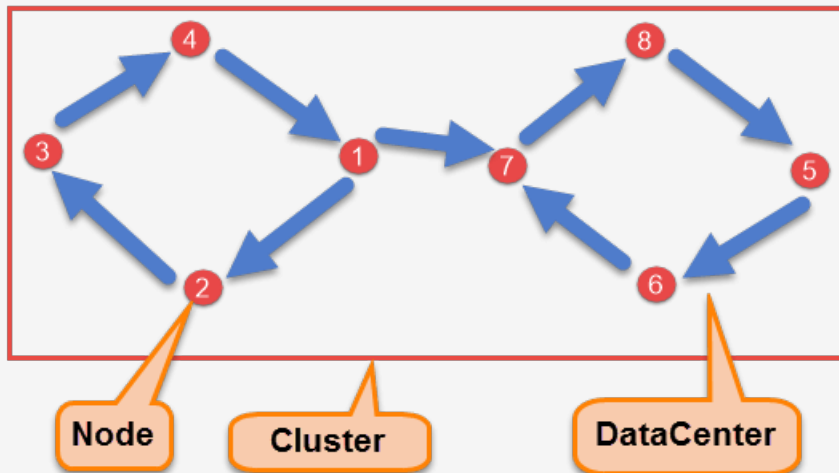
id	value
0	54
1	35
2	22

# Cassandra

- Inicialmente criado no facebook e hoje é mantido pela Apache;
- Velocidade de gravação de até 360MB/s, mais rápido que as buscas;
- Armazenamento distribuído por hash em nós;
- Nós são replicados;
- Linguagem: CQL (Cassandra Query Language)



# Arquitetura do Cassandra



# Exemplo Cassandra

```
CREATE TABLE "my_keyspace"."contacts" (  
    contact_id uuid,  
    first_name text,  
    last_name text,  
    phone_number text,  
    PRIMARY KEY(contact_id)  
);
```

Figura: Comando CREATE TABLE

# Exemplo Cassandra

```
INSERT INTO
"my_keyspace"."contacts" (contact_id, first_name, last_name, phone_number)
VALUES (uuid(), 'John', 'Doe', '559912341234');
INSERT INTO
"my_keyspace"."contacts" (contact_id, first_name, last_name)
VALUES (uuid(), 'Mary', 'Jane');
INSERT INTO
"my_keyspace"."contacts" (contact_id, first_name, phone_number)
VALUES (uuid(), 'Will', '441239876543');
```

Figura: Comando INSERT

# Exemplo Cassandra

```
SELECT * FROM "my_keyspace"."contacts";
```

contact_id	first_name	last_name	phone_number
910ac22b-538e-42d2-a5cc-7163cdd5a0c3	Will	null	441239876543
e7beb210-b9b6-4b08-9afb-950d5918ba2e	John	Doe	559912341234
7ef9a9af-1332-46bb-abb4-2deb9b335588	Mary	Jane	null

Figura: Comando SELECT

# Exemplo Cassandra

```
SELECT first_name, last_name FROM my_keyspace.contacts  
WHERE first_name = 'Mary';
```

```
InvalidRequest: code=2200 [Invalid query] message="Cannot execute this query as it  
might involve data filtering and thus may have unpredictable performance.  
If you want to execute this query despite the performance unpredictability,  
use ALLOW FILTERING"
```

Figura: SELECT sem chave na cláusula WHERE



# Aplicações do Cassandra

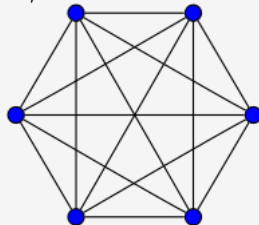
- Catálogo de produtos;
- Redes sociais;
- Detecção de fraudes;
- Aplicações analíticas;

Empresas que usam Cassandra:

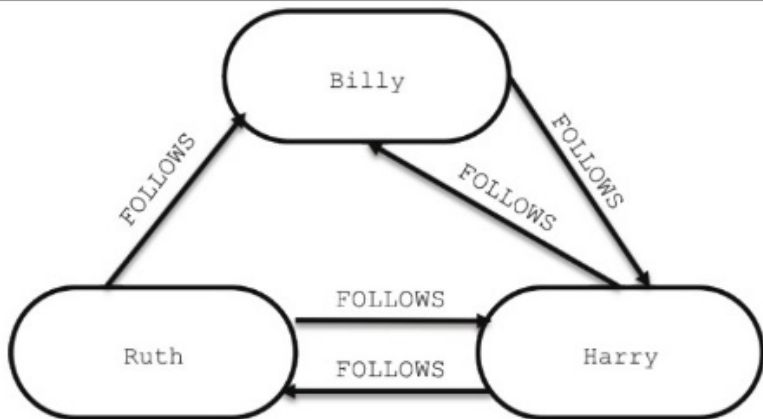
- Amazon;
- eBay;
- Netflix;
- Facebook;
- Microsoft;
- Instagram;
- NASA;

# Grafo

- Motivado pela grande quantidade de dados sendo geradas diariamente (Big Data);
- Expressa de forma explícita os relacionamentos através de:
  - Relacionamentos = arestas;
  - Entidades = Vértices;
- Modelagem mais simples, linguagem mais natural;
- Usado para consultas complexas;
- Relacionamentos podem possuir dados;



# Exemplo Twitter



# Neo4J

- Desenvolvido em Java;
- Banco de dados em grafo mais popular;
- Open Source;
- Linguagem: Cypher



# Exemplo Neo4J

## CREATE

```
( _1 { name: "Radiohead" } ),  
( _2 { name: "The Black Keys" } ),  
( _3 { name: "Joy Division" } ),  
( _4 { name: "Los hermanos" } ),  
( _5 { name: "Oasis" } ),  
( _6 { name: "Daft Punk" } ),  
( _7 { name: "Felguk" } ),  
( _8 { name: "Deadmau5" } ),  
  
( eder { name: "Eder Ignatowicz", age: 22 } ),  
( tiago { name: "Tiago Bento", age: 19 } ),
```

Figura: Criação de bandas e fãs (Cypher)

# Exemplo Neo4J

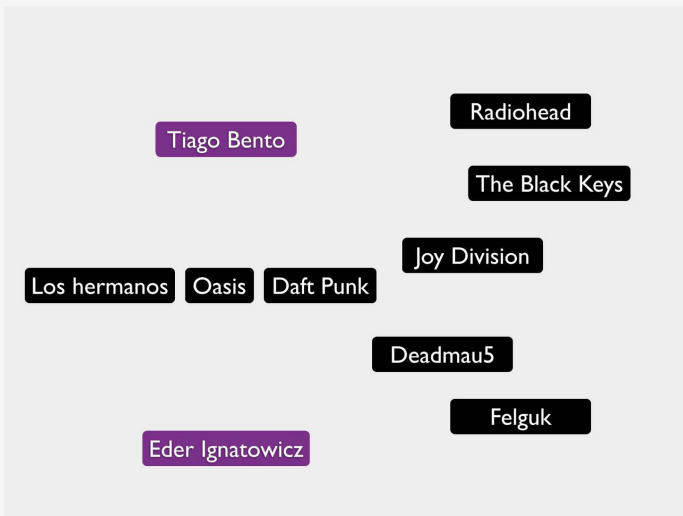


Figura: Grafo

## Exemplo Neo4J

```
eder-[:LIKES]->_8,  
eder-[:LIKES]->_7,  
eder-[:LIKES]->_6,  
eder-[:LIKES]->_5,  
eder-[:LIKES]->_4,  
  
tiago-[:LIKES]->_6,  
tiago-[:LIKES]->_5,  
tiago-[:LIKES]->_4,  
tiago-[:LIKES]->_3,  
tiago-[:LIKES]->_2,  
tiago-[:LIKES]->_1
```

**RETURN \***

Figura: Criação dos relacionamentos entre fãs e bandas (Cypher)

# Exemplo Neo4J

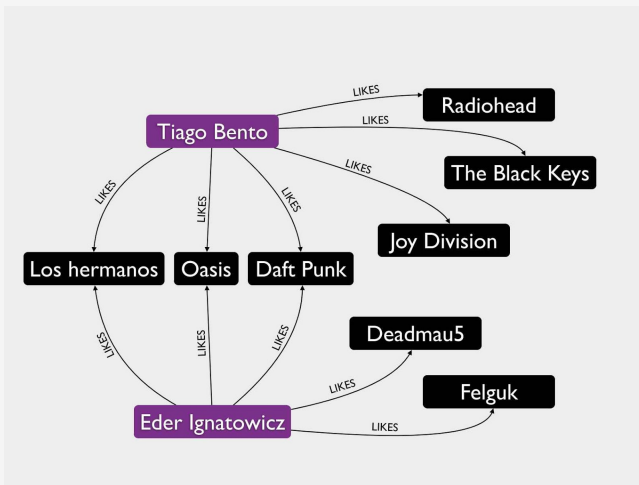


Figura: Grafo



# Exemplo Neo4J

```
START    ed=node(9)
MATCH    (ed)-[:LIKES]->(a)
RETURN   collect(a.name);
```



```
[ "Deadmau5", "Felguk", "Daft Punk", "Oasis", "Los hermanos" ]
```

Figura: Quais são as bandas que Eder curte? (Cypher)

# Exemplo Neo4J

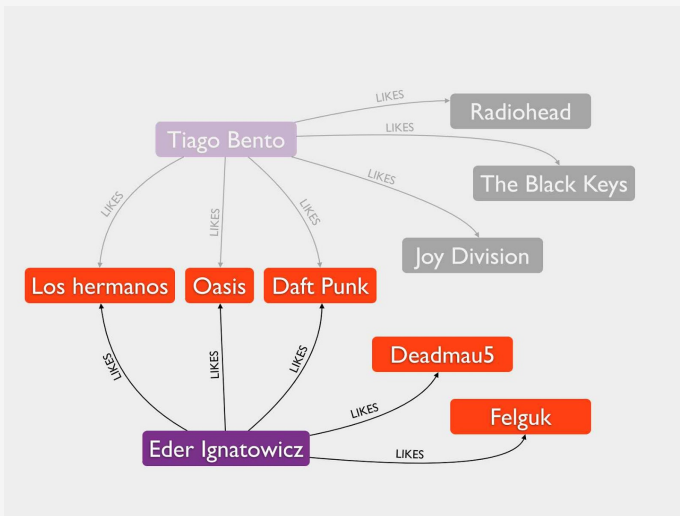


Figura: Grafo

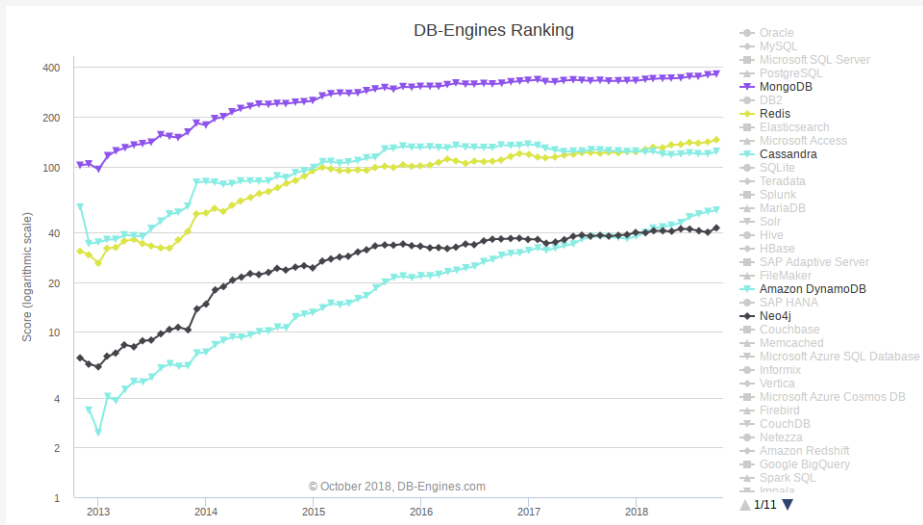
# Aplicações do Neo4J

- Detecção de fraude e solução de análise;
- Ferramenta de recomendação e recomendação de produto;
- Mídias sociais e Redes Sociais
- Inteligência artificial e Machine Learning;

Empresas que usam Neo4J:

- eBay;
- Walmart;
- Medium
- LinkedIn;
- Gamesys;

# Comparativo dos Principais SGBD's NoSQL



# Conclusão

Com o grande crescimento do volume de dados em determinadas organizações, os bancos de dados NoSQL tem se tornado uma grande alternativa quando nos referimos a escalabilidade e disponibilidade, fatores estes que se tornam imprescindíveis em algumas aplicações Web.



# Referências Bibliográficas

- <https://imasters.com.br/banco-de-dados/introducao-ao-redis-o-nosql-chave-valor-mais-famoso>
- <http://desenvolvedor.ninja/redis-o-que-e-e-para-que-serve/>
- <https://aws.amazon.com/pt/elasticache/what-is-redis/>
- <http://blog.clubedocodigo.com.br/crie-aplicacoes-mais-rapidas-utilizando-redis/>
- <https://diogobemfica.com.br/comecando-com-o-redis/>
- <https://www.ibm.com/developerworks/br/library/j-javadev2-22/index.html>
- <https://aws.amazon.com/pt/nosql/document/>

# Referências Bibliográficas

- <https://www.devmedia.com.br/introducao-ao-mongodb/30792>
- <https://www.mongodb.com/what-is-mongodb>
- <https://www.devmedia.com.br/banco-de-dados-nosql-um-novo-paradigma-revista-sql-magazine-102/25918>
- <https://aws.amazon.com/pt/nosql/columnar/>
- <https://www.slideshare.net/gilesaugusto/nosql-familia-de-colunas-monografia>
- <https://www.devmedia.com.br/introducao-ao-cassandra/38377>
- <https://www.infoq.com/br/presentations/neo4j-visao-pratica>
- <https://imasters.com.br/banco-de-dados/graphdb-series-o-que-e-um-banco-de-dados-de-grafos>



# Referências Bibliográficas

- <https://www.slideshare.net/MayogaX/banco-de-dados-de-grafos>
- <https://neo4j.com/use-cases/>
- <https://www.devmedia.com.br/introducao-aos-bancos-de-dados-nosql/26044>
- <https://dicasdeprogramacao.com.br/6-motivos-para-usar-bancos-de-dados-nosql/>
- <http://www.cienciaedados.com/top-6-nosql-databases/>
- <https://imasters.com.br/banco-de-dados/o-top-5-das-plataformas-nosql-no-mercado-atual>
- <https://aws.amazon.com/pt/nosql/key-value/>
- <https://db-engines.com/en/ranking>
- <https://www.devmedia.com.br/introducao-ao-mongodb/30792>
- <https://www.couchbase.com/resources/why-nosql>