# BANCO DE DADOS NOSQL

# BANCO DE DADOS RELACIONAL

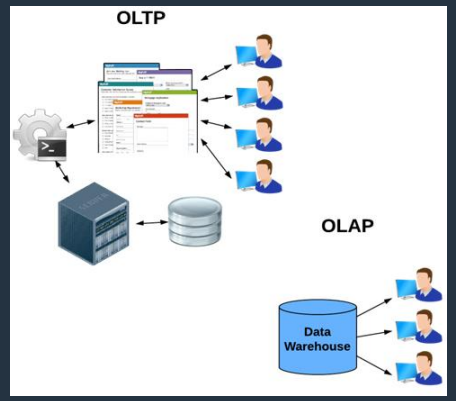
- Os dados são estruturados de acordo com o modelo relacional.
- SQL Server, Oracle, PostgreSQL, MySQL, DB2 e outros.
- Elementos básicos: Relações (tabelas) e Registros (tuplas).
- Características fundamentais:

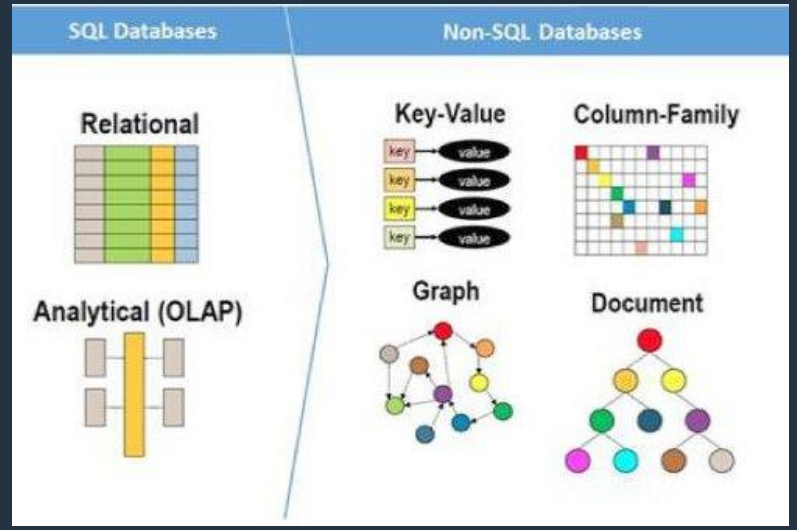
    *- Restrições de integridade*

        *- PK-primary key, FK-foreign key, UK-unique key, CK-check key, NN-not null.*

    *- Normalização (formas normais).*

    *- Linguagem SQL (Structured Query Language)*

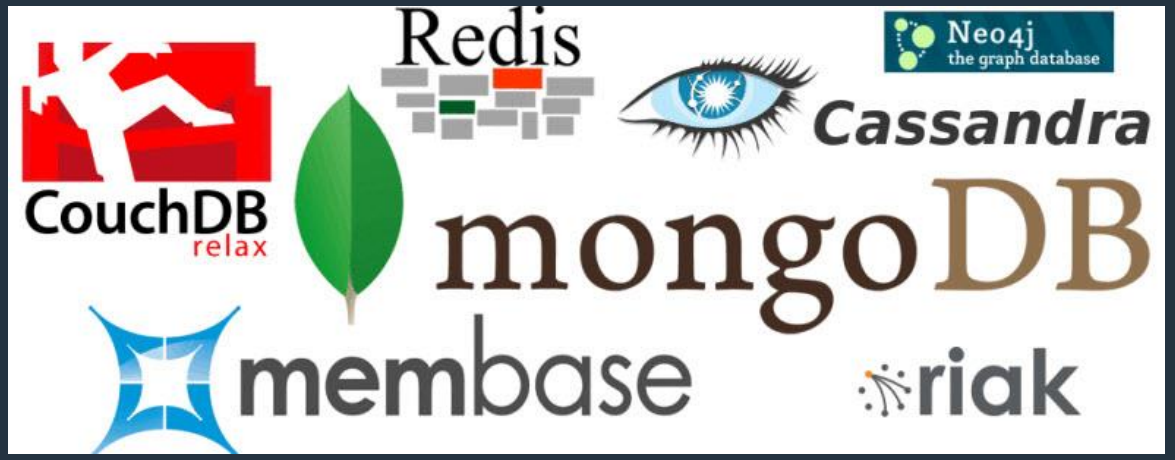


# 1. BANCO DE DADOS NOSQL

- NoSQL é um termo genérico que define bancos de dados não-relacionais.
- A tecnologia NoSQL foi iniciada por companhias líderes da internet como Google, Facebook, Amazon, e Linkedin, para superar as limitações de banco de dados relacional para aplicações web modernas.
- Características:

    - Escalabilidade horizontal (add mais nós)
    - Ausência de esquema ou esquema flexível
    - Suporte e Replicação
    - API simples
    - Nem sempre prima pela consistência



# PROPRIEDADES DOS BANCOS DE DADOS

- ACID X BASE

Propriedades ACID:

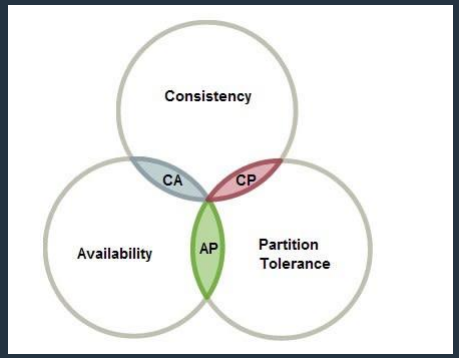
- Atomicidade
- Consistência
- Isolamento
- Durabilidade

Propriedades BASE:

- Basically Available
- Soft-State
- Eventually Consistent

- Teorema CAP:
  - Consistency
  - Availability
  - Partition Tolerance

De acordo com o teorema CAP, um sistema distribuído de BD somente pode operar com dois desses comportamentos ao mesmo tempo, mas jamais com três simultaneamente.



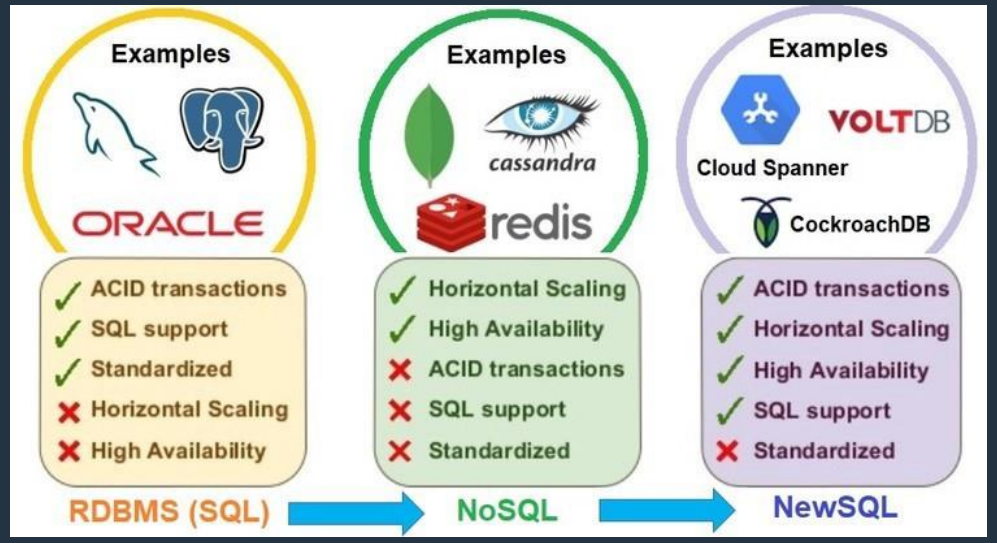
# VISÃO GERAL DE NEWSQL

- Definição e visão de NewSQL:

Pode ser definido como uma classe de SGBDs relacionais modernos que buscam fornecer o mesmo desempenho escalonável do NoSQL, para cargas de trabalho OLAP e, simultaneamente, garantir a conformidade ACID para transações como no RDBMS.

Basicamente esses sistemas desejam alcançar a escalabilidade do NOSQL sem ter que descartar o modelo relacional com SQL e suporte a transações do DBMS legado.

- Exemplos de banco de dados NewSQL:




Bancos NewSQL
- MemSQL
- VoltDB
- SQLFire
- MariaDB

| | Old SQL | NoSQL | NewSQL |
|---|---|---|---|
| Relational | Yes | No | Yes |
| SQL | Yes | No | Yes |
| ACID transactions | Yes | No | Yes |
| Horizontal scalability | No | Yes | Yes |
| Performance / big volume | No | Yes | Yes |
| Schema-less | No | Yes | No |

# TÉCNICAS EM BANCOS NOSQL

- Algumas técnicas utilizadas para a implementação de suas funcionalidades, entre elas estão:

  - Map/Reduce: Permite a manipulação de enormes volumes de dados ao longo de nós em uma rede.

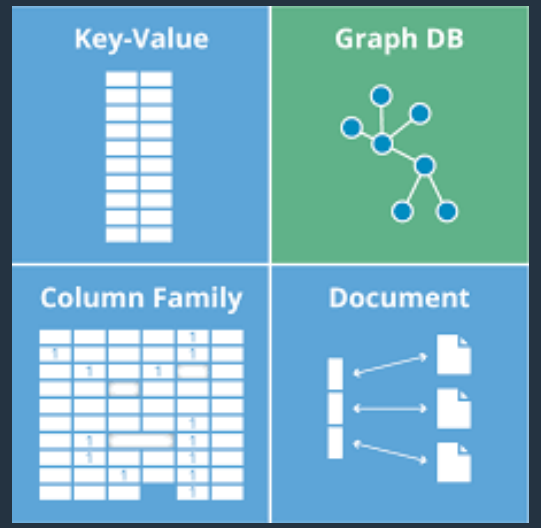
  - Consistent hashing: é o processo de mapear um dado, normalmente um objeto de tamanho arbitrário para outro dado de tamanho fixo.

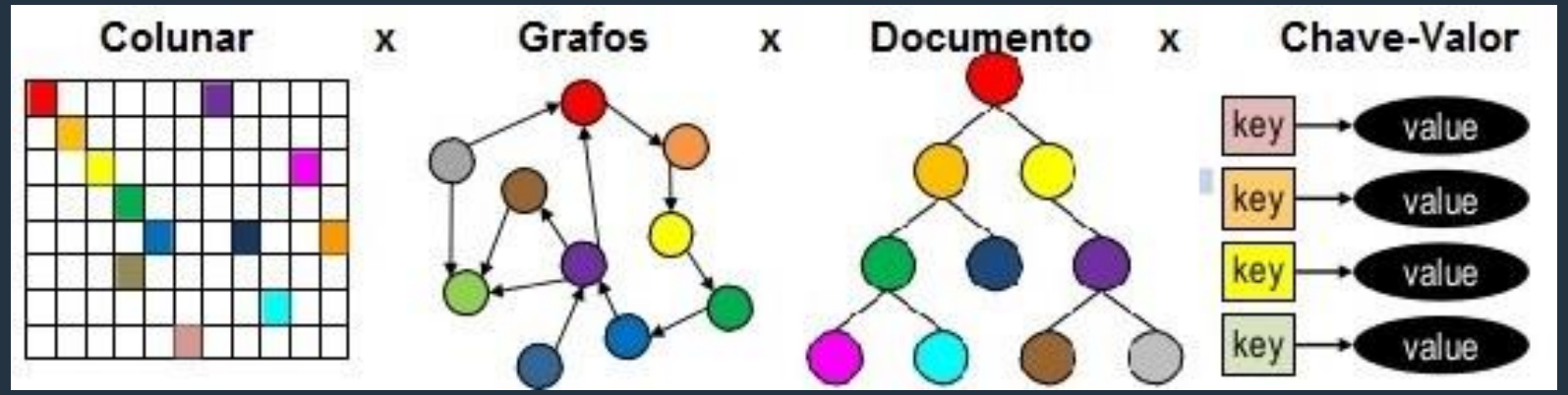
  - MVCC – Multiversion Concurrency Control: Oferece suporte a transações paralelas em bancos de dados.

  - Vector Clocks: Ordena eventos que ocorrem em um sistema.

# TIPOS DE BANCOS NOSQL

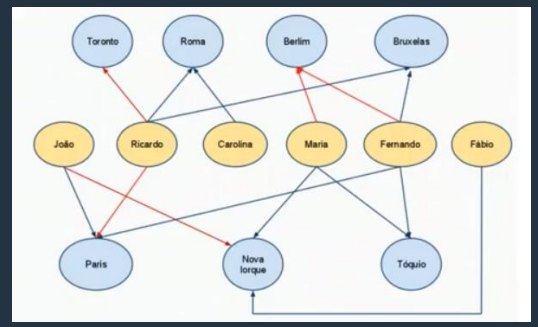
- Temos quatro categorias do NoSQL:
  - Chave-valor (key-Value)
  - Orientado a Grafos
  - Orientado a Coluna (Column Family)
  - Orientado a Documentos



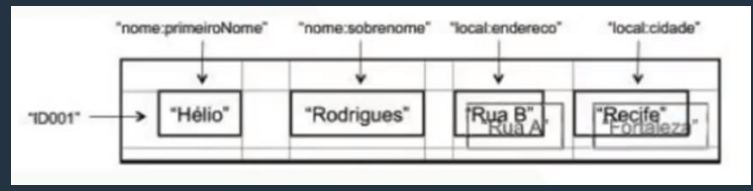




- Chave-valor (Key-Value)

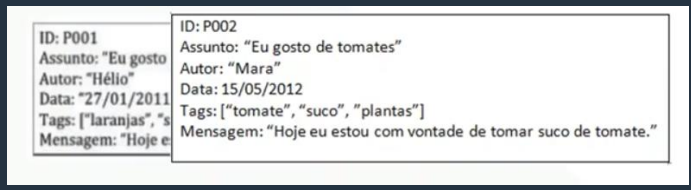

- Orientado a Grafos




- Orientado a Colunas




- Orientado a Documentos



# MOTIVAÇÕES NO USO DE NOSQL

Empresas estão adotando NoSQL para um número crescente de cenários, a escolha que é impulsionada por quatro tendências relacionadas:

- Big Users: Um grande número de usuários, combinados com a natureza dinâmica dos padrões de uso está demandando uma tecnologia de BD mais facilmente escalável.
- Big Data: é a área do conhecimento que estuda como tratar, analisar e obter informações a partir de conjuntos de dados demasiadamente grandes para serem analisados por sistemas tradicionais. Razões para usar Big Data:

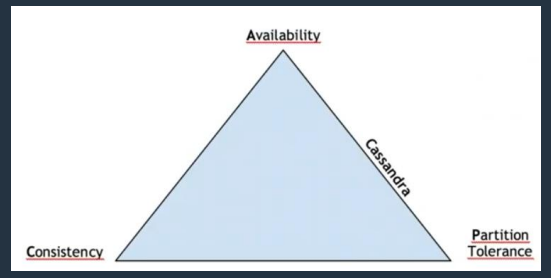
  - Entender padrões
  - Prever situações
  - Criar fronteiras
  - Informar coleções de dados
  - Estimar parâmetros escondidos
  - Calibrar


- Internet das Coisas: O termo foi usado pela primeira vez em 1999 para descrever um sistema onde os objetos poderiam ser conectados à internet. 32 bilhões de coisas vão estar conectadas a internet, 10% de todos os dados serão gerados por sistemas embarcados, 21% dos mais valiosos dados serão gerados por sistemas embarcados, e dados de telemetria (semi-estruturados e contínuos) representam um desafio para bancos de dados relacionais. Ex: Cidades Inteligentes: Poluição sonora; otimizar coleta de lixo; controle de tráfego; controle de distribuição de energia elétrica; segurança pública.
- Cloud Computing: é um termo coloquial para a disponibilidade sob demanda de recursos do sistema de computador, especialmente armazenamento de dados e capacidade de computação, sem o gerenciamento ativo direto do utilizador.

# 2. PRINCIPAIS BANCO DE DADOS NOSQL

# APACHE CASSANDRA

- Orientado a Coluna (Column Family)
- Do Teorema CAP:




Banco de Dados Relacional:

| sku_produto | nome_produto | preco_produto |
|---|---|---|
| 1 | Tênis | 100 |
| 2 | Bola | 50 |
| 3 | Camisa | 200 |
| 4 | Bike | 900 |

Banco de Dados Colunar:

sku_produto

| id | value |
|---|---|
| 0 | 1 |
| 1 | 2 |
| 2 | 3 |
| 3 | 4 |

nome_produto

| id | value |
|---|---|
| 0 | Tênis |
| 1 | Bola |
| 2 | Camisa |
| 3 | Bike |

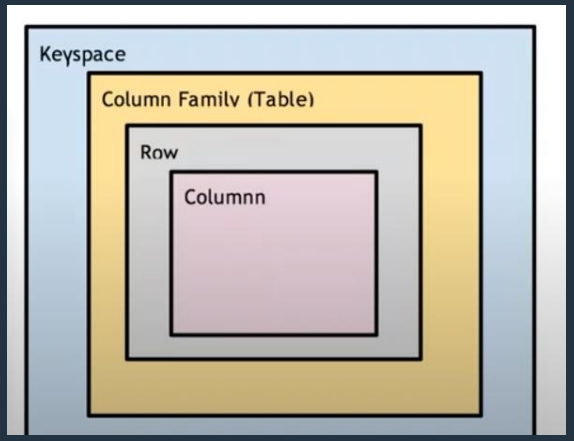
preco_produto

| id | value |
|---|---|
| 0 | 100 |
| 1 | 50 |
| 2 | 200 |
| 3 | 900 |

O Cassandra é um projeto de sistema de BD distribuído altamente escalável de segunda geração, que reúne a arquitetura do DynamoDB da Amazon Web Services e modelo de dados baseado no BigTable do Google.

Arquitetura do Cassandra: é segmentado em:

- Keyspace: corresponde a um banco de dados no relacional
- Tables (column families): equivalente à uma tabela no mundo relacional
- Rows: São compostas pela Primary key e um conj de columns
- Columns: Composto por Column key, Column value e Timestamp



# REDIS – REMOTE DICTIONARY SERVER

- Chave-valor (key-Value)
- São sistemas distribuídos, também conhecidos como tabelas de hash distribuídas, armazenam

objetos indexados por chaves, e possibilitam a busca por esses onjetos a partir de suas chaves.

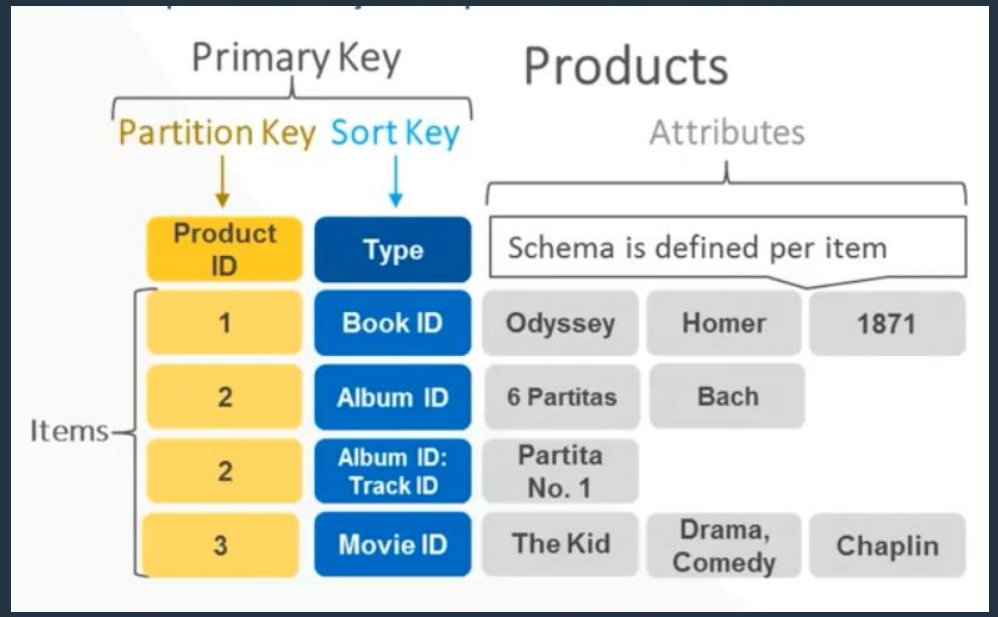
- O Redis é um banco de dados de memória e  de código aberto que é usado como  cache e como intermediário de mensagens. Também é conhecido como um servidor de dados estruturados.
- Oficialmente o Redis não tem uma distribuição para

Windows, porém é possível encontrar uma distribuição

paralela no GitHub. Características:

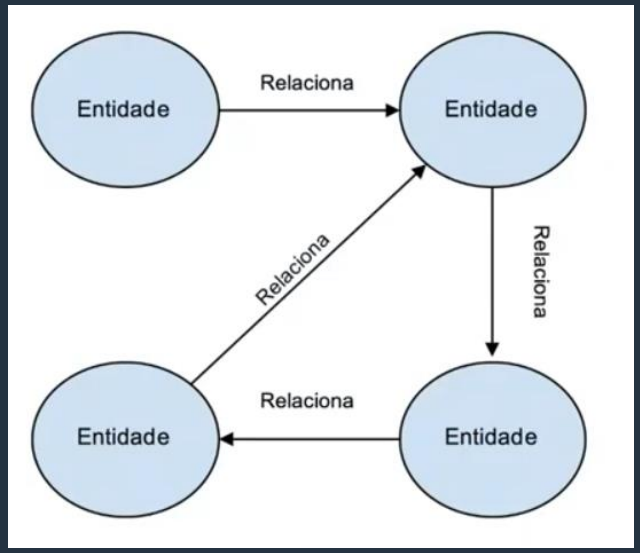
- Desempenho muito rápido
- Estruturas de dados n memória
- Versatilidade e facilidade de uso
- Replicação e persistência
- Compatibilidade com as linguagens Java, Python, PHP, C, C++, C#,

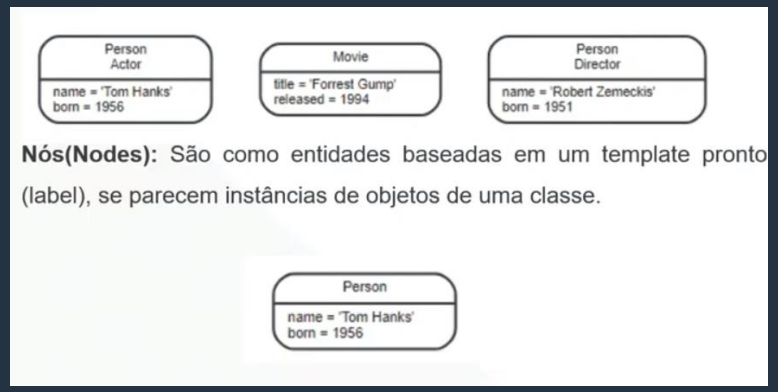
JavaScript, Node.js, Ruby, R, Go e muitas outras.

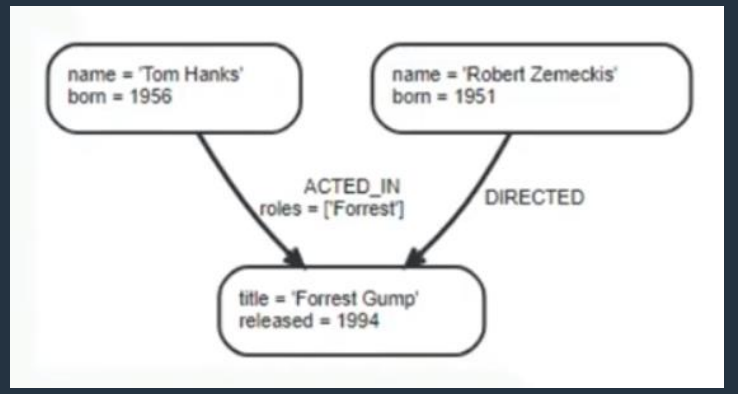


# NEO4J

- Os dados são persistidos em um esquema de grafo, onde um registro "aponta" para o próximo.
- O Neo4j é um sistema SGBD gráfico, descrito como um BD transacional compatível com ACID com armazenamento e processamento de gráfico nativo, está disponível um uma "edição da comunidade" de código aberto.
- Implementado em Java e acessível a partir de softwares escritos em outras linguagens usando  a linguagem de consulta Cypher Query através de um endpoint HTTP transacional ou através do protocolo binário "bolt"




Palavras chaves do Cypher:
- MATCH
- WHERE
- RETURN
- CREATE
- MERGE

# MONGO DB


- Orientado a Documentos
- Armazenam chave/valor.
- O valor é um documento estruturado e indexado,

com metadados.

- Valor (Documento), pode ser consultado.
- JSON: JavaScript Object Notation.

  - Feito para trocas de dados

  - Mais compacto e legível que XML

- Código aberto e multiplataforma, escrito em C++, escalável, não possui

Schema fixo, não tem integridade referencial, permite documentos aninhados.


```
{"clientes":[
  { "nome" "Fernando", "sobrenome","Amaral" },
  { "nome":"Anna", "sobrenome": "Ebling" },
  { "nome": "Pedro", "sobrenome": "Soares" }
]}
```

| Relacional | MongoDB |
|---|---|
| Banco de Dados | Banco de Dados |
| Tabela | Coleção |
| Linha | Documento |
| Coluna | Campo |

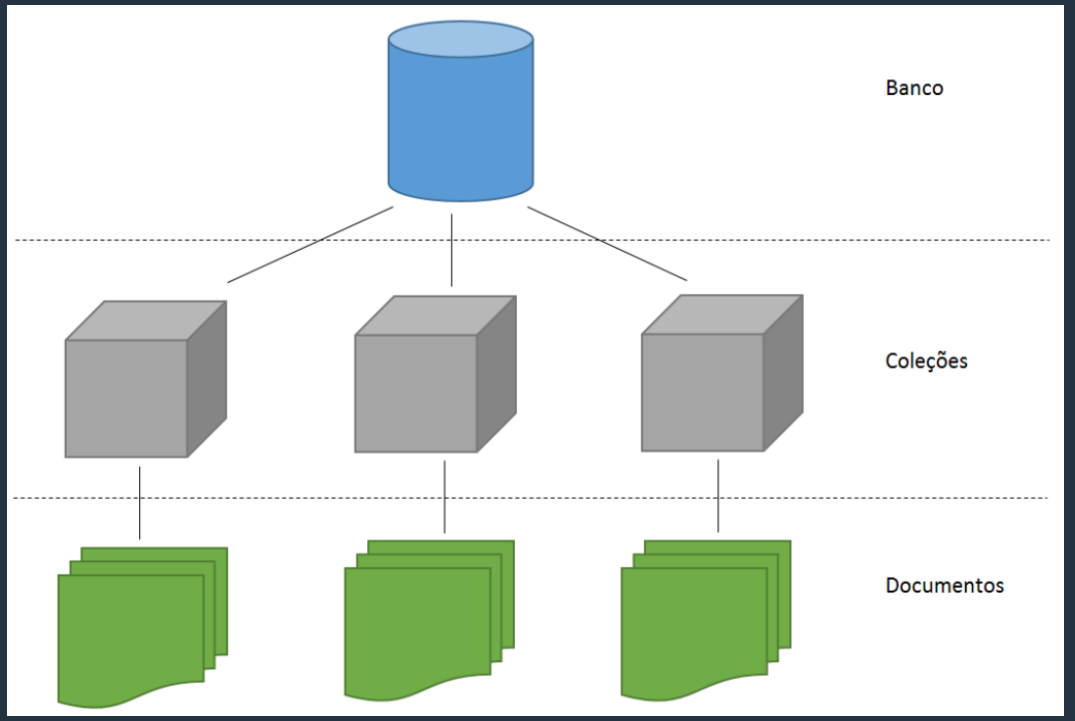
# MONGO DB

## Document

- Um documento é um conjunto de pares de valores-chave.
- Documentos têm esquema dinâmico. Esquema dinâmico significa que os documentos na mesma coleção não precisam ter o mesmo conjunto de campos ou estrutura e campos comuns em dumentos de uma coleção podem conter diferentes tipos de dados.

| Relacional | MongoDB |
|---|---|
| Banco de Dados | Banco de Dados |
| Tabela | Coleção |
| Linha | Documento |
| Coluna | Campo |

```
db.users.insertOne(          <-- collection
   {
      name: "sue",           <-- field: value
      age: 26,               <-- field: value
      status: "pending"      <-- field: value
   }                         } document
)
```



# MONGO DB – COMANDOS

https://docs.mongodb.com/manual/reference/command/

```
db.posts.insert([
{nome:"André",postagem:"Produtos caros", data:"12-01-2019",idade:25},
{nome:"Ricardo",postagem:"Produtos caros", data:"14-07-2019", idade:12}])

#idade menor ou igual a 12
> db.posts.find({postagem:"Produtos caros",idade: {$lte: 12}})
{ "_id" : ObjectId("5d0911600ee1100c307004da"), "nome" : "Ricardo", "postagem"
: "Produtos caros", "data" : "14-07-2019", "idade" : 12 }
```

- $lt: menor que
- $lte: menor ou igual que
- $ne: diferente de
- $in: contém
- $nin: não contém

- Comando "use" acessa banco
- Acessar banco inexistente cria o banco
  - É necessário inserir dados para persistir o banco de dados

```
> use dbmidias
switched to db dbmidias
```

- insert
  - Insere um único documento na coleção posts

```
>db.posts.insert({nome:"José",postagem:"Bons Produtos!", data:"31-06-2019"})
WriteResult({ "nInserted" : 1 })
```

Acima a coleção é criada implicitamente

- Para criar a coleção primeiro:

```
>db.createCollection("clientes")
```

```
> db.posts.find()
{ "_id" : ObjectId("5d090bc10ee1100c307004d4"), "nome" : "José", "postagem" : "Bons Produtos!", "data" : "31-06-2019" }
{ "_id" : ObjectId("5d090cd10ee1100c307004d5"), "nome" : "Antonio", "postagem" : "Minha bike quebrou", "data" : "26-05-2019" }
{ "_id" : ObjectId("5d090cd10ee1100c307004d6"), "nome" : "Maria Silva", "postagem" : "Encontrei tudo que procurava", "data" : "14-06-2019" }
{ "_id" : ObjectId("5d090cd10ee1100c307004d7"), "nome" : "Lucas Andrade", "postagem" : "Ótimo atendimento!", "data" : "12-04-2019" }
```

= (Select *)

# 3. CRUD NO MONBO DB

- Mongo DB

# 4. MODELAGEM E RELACIONAMENTO MONGO DB

# 5. SCHEMA E VALIDATION

# 6. ARMAZENAMENTO DE DADOS EM NUVEM E SISTEMAS DE ARQUIVOS