

# **Bancos de dados Não Convencionais**

Prof. Daniel Brandão

# Apresentação

## Daniel Brandão

- o Graduação em **Sistemas para Internet**
- o Esp. **Tecnologias para Web**
- o Mestrando em **Tecnologia da Informação**
  
- o Desenvolvedor web desde **2006**
- o Analista de sistemas desde **2010**
- o Professor desde **2012**
- o Analista de dados desde **2017**



**Subgerente de Sistemas SES/PB**  
**Consultor e Professor**



# CONTEÚDO PROGRAMÁTICO

## Parte 1

- Revisão sobre Banco de dados
- Introdução a bancos de dados não relacionais
- Modelos de BD NoSQL
- MongoDB
- Instalação do ambiente

## Parte 2

- Primeiros passos com MongoDB
- Comandos de **INSERÇÃO, ALTERAÇÃO, EXCLUSÃO**
- Consultas com MongoDB
- Importando dados no MongoDB

## Parte 3

- Consultas avançadas no MongoDB
- Projetos relacionados ao MongoDB
- Definição do Projeto
- Final



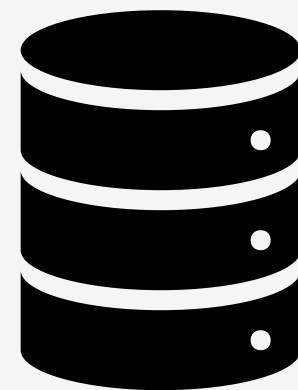
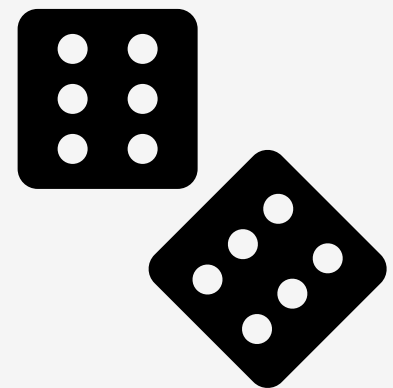
## *Objetivo do módulo*

---



Apresentar os conceitos por trás dos bancos de dados não relacionais, introduzindo ao modelo orientado a documentos através do MongoDB.

**Vamos falar  
de DADOS!**



# Dados...

As decisões estratégicas são baseadas em **informações** provenientes de **dados** coletados

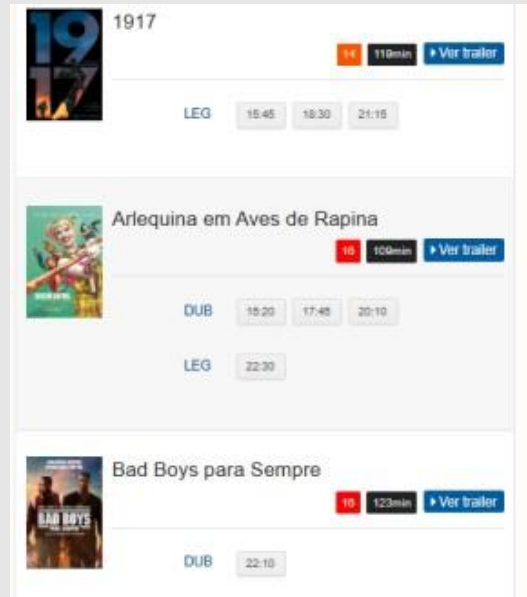
**Recursos** mais valiosos de uma empresa ou negócio

Não correm risco de extinção, **só aumentam...**

Várias fontes, em **ambientes diversos...**



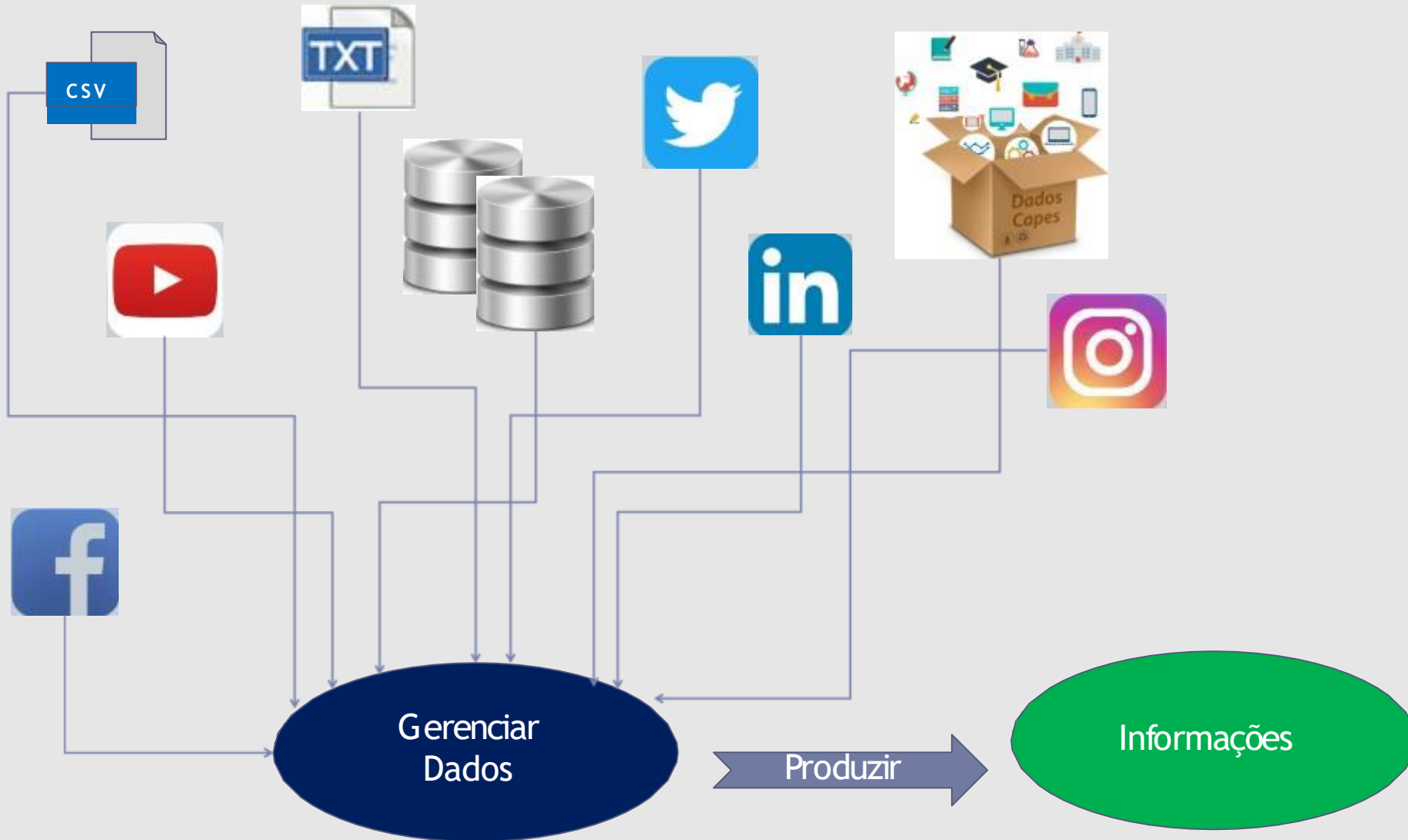
# Onde estão os dados???



```
<?xml version="1.0" encoding="ISO-8859-1" standalone="no"?><CURRICULO-VITAE
SISTEMA-ORIGEM-XML="LATTES_OFFLINE" DATA-ATUALIZACAO="17092015" HORA-ATUALIZACAO="190239"
NUMERO-IDENTIFICADOR="0454804076914929"><DADOS-GERAIS NOME-COMPLETO="Cândido José Ramos do Egypto"
NOME-EM-CITACOES-BIBLIOGRAFICAS="EGYPTO, C. J. R. do" NACIONALIDADE="B" PAIS-DE-NASCIMENTO="Brasil"
UF-NASCIMENTO="PB" CIDADE-NASCIMENTO="João Pessoa" PERMISSAO-DE-DIVULGACAO="NAO" DATA-FALECIMENTO=""
SIGLA-PAIS-NACIONALIDADE="BRA" PAIS-DE-NACIONALIDADE="Brasil"><RESUMO-CV TEXTO-RESUMO-CV-RH="Bacharel em
Ciência da Computação pela UFPB, especialista em Informática Educativa pelo CEFET/MG, mestre em Engenharia
Biomédica pela UFPB (Área: Informática em Saúde), doutorando em Computação pela UFPE (Área: Banco de
Dados). Atualmente é professor do IFPB, Campus João Pessoa, e Coordenador do CST em Redes de Computadores."
TEXTO-RESUMO-CV-RH-EN="graduate at Bacharelado Em Ciência da Computação from Universidade Federal da
Paraíba (1999) and master's at Biomedical Engineering from Universidade Federal da Paraíba (2001). Has
experience in Computer Science, focusing on Information Systems, acting on the following subjects:
informática em saúde, sistemas de informação, laboratório de bromatologia, vacinação and educação em
saúde."><OUTRAS-INFORMACOES-RELEVANTES OUTRAS-INFORMACOES-RELEVANTES=""><ENDERECO
FLAG-DE-PREFERENCIA="ENDERECO_RESIDENCIAL"><ENDERECO-PROFISSIONAL CODIGO-INSTITUICAO-EMPRESA="047000000007"
NOME-INSTITUICAO-EMPRESA="Instituto Federal de Educação, Ciência e Tecnologia da Paraíba" CODIGO-UNIDADE=""
NOME-UNIDADE="" CODIGO-ORGAO="047001000990" NOME-ORGAO="Diretoria de Ensino" PAIS="Brasil" UF="PB"
LOGRADOURO-COMPLEMENTO="Av. Primeiro de Maio, 720" BAIRRO="Jaguaripe" CIDADE="João Pessoa" CAIXA-POSTAL=""
CEP="58015360" DDD="83" TELEFONE="36121200" RAMAL="1391" FAX="" HOME-PAGE="http://www.ifpb.edu.br"/></
ENDERECO><FORMACAO-ACADEMICA-TITULACAO><GRADUACAO SEQUENCIA-FORMACAO="1" NIVEL="1"
```



# Como Gerenciar isso?





# Dados x Aplicações

Existem aplicações **sem** dados?

Os dados precisam ser **persistidos**?

# Dados x Aplicações

- Como persistir?
- Que **modelo de representação/persistência** de dados usar?



# Banco de Dados e SGBD

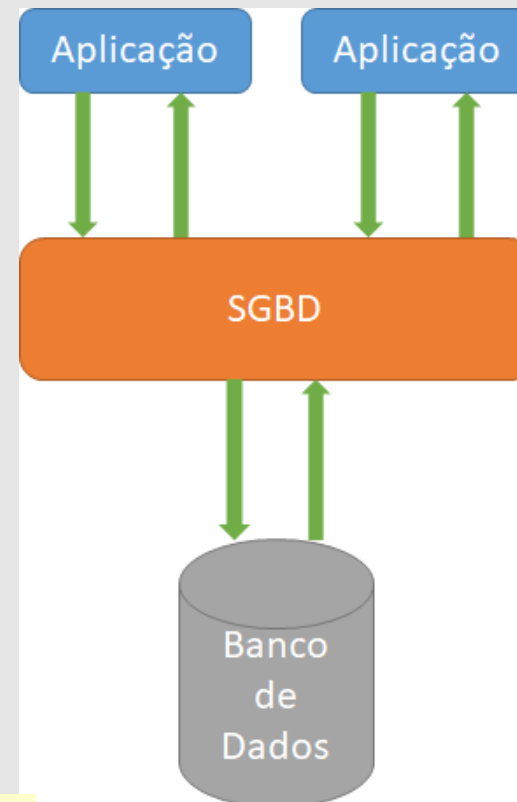
Um **BD** é uma coleção de dados **interrelacionados**

- Conjunto de dados organizados, compartilhados e integrados, que seguem uma **semântica** e um **conjunto de regras**

# Banco de Dados e SGBD

---

- Um **SGBD** é um componente de software projetado para garantir o armazenamento e o gerenciamento de bancos de dados
  - Armazenamento;
  - Acesso facilitado por meio de linguagem de consulta;
  - Controle de Concorrência e recuperação em caso de falhas



**BD e SGBD seguem um  
MODELO de DADOS**

# Modelos de Dados

---

- **Representação/persistência**

- Modelo **Relacional**
- Modelo Orientado a Objetos
- Modelo Objeto-Relacional
- Modelo baseado em XML
- Modelos **NoSQL**
  - Chave-Valor
  - Colunas
  - Documentos
  - Grafo

<https://db-engines.com/en/ranking>

# BD Modelo Relacional

---

- Baseados em Tabelas
  - Estrutura em colunas
  - Registros em linhas
  - SGBDs famosos no mercado



# BD Modelo Não Relacional

---

- Baseados em Diferentes Modelos

- Diferentes Estruturas
- Registros podem ser em colunas, em linhas, em nós, documentoc etc.
- SGBDs famosos no mercado





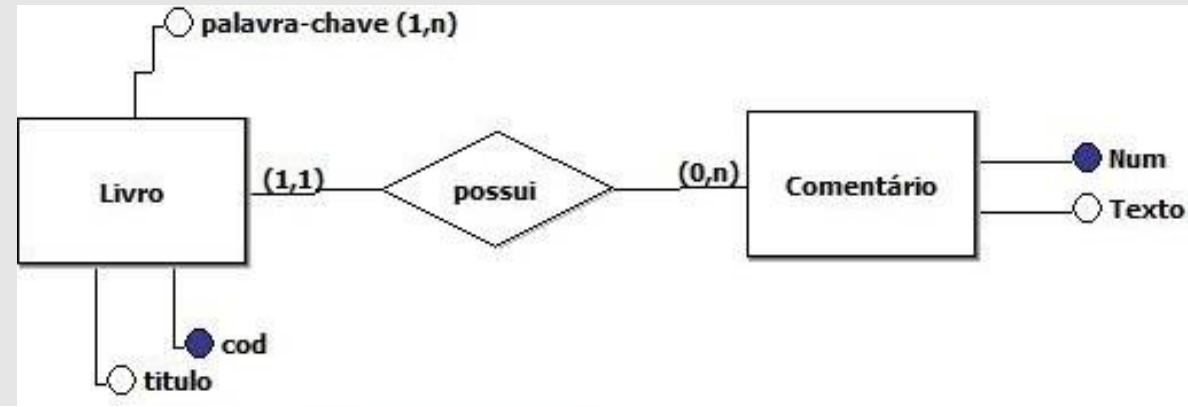
Mas, antes de implementar o banco

**O que é  
preciso?**

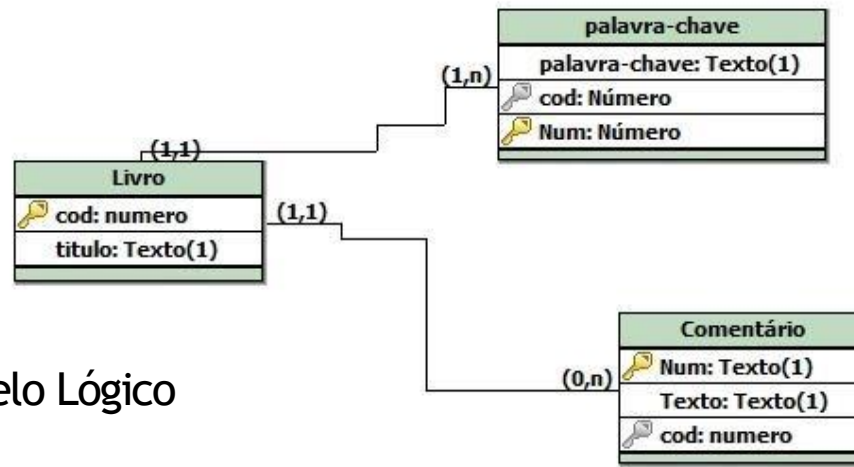


# Exemplo para Modelo Relacional

Modelo  
Conceitual



Modelo Lógico



⚡ COD	⚡ TITULO
1	Harry Potter

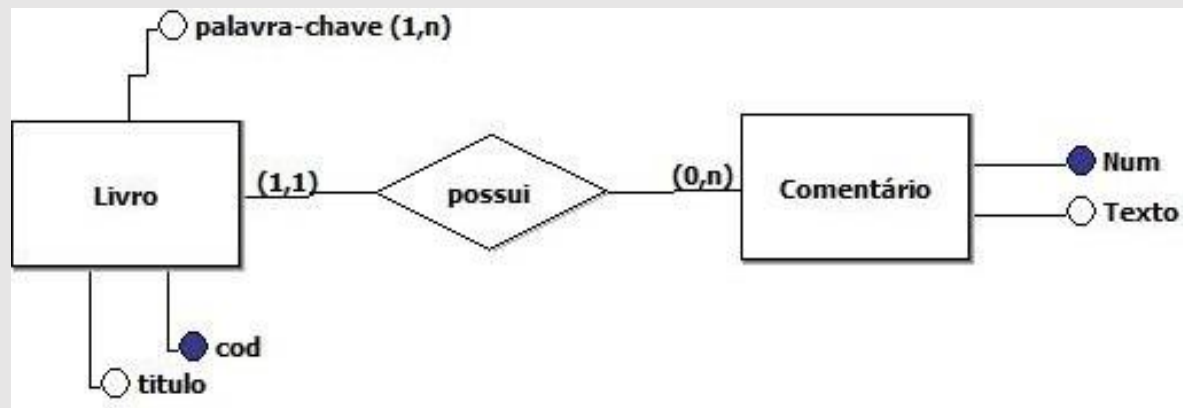
⚡ NUM	⚡ TEXTO	⚡ COD
1	Livro mágico	1

⚡ PALAVRA	⚡ COD	⚡ NUM
Ficção	1	1
Magia	1	2

Modelo Físico

# Exemplo para Modelo baseado em Documentos

Modelo  
Conceitual



Quais as principais entidades?  
Quais as entidades “folha”?

**Livro**  
Título  
[Cod]

Palavras-chave

Comentários

# Modelo Físico

```
{
  "_id" : ObjectId("5a7859d33c22785e9flaba64"),
  "title" : "MongoDB - Como?",
  "description" : "MongoDB - Como",
  "by" : "MongoDBExpert",
  "url" : "http://www.mongodbexpert.com",
  "tags" : [
    "mongodb",
    "database",
    "NoSQL",
    "Document"
  ],
  "likes" : 100.0,
  "comments" : [
    {
      "user" : "user1",
      "message" : "My comment",
      "dateCreated" : ISODate("2017-06-21T05:15:00.000Z"),
      "likes" : 0.0
    },
    {
      "user" : "user2",
      "message" : "My comments",
      "dateCreated" : ISODate("2017-06-21T10:45:00.000Z"),
      "likes" : 5.0
    }
  ]
}
```

---



Nem tudo são

**Flores**

# Tipos de dados



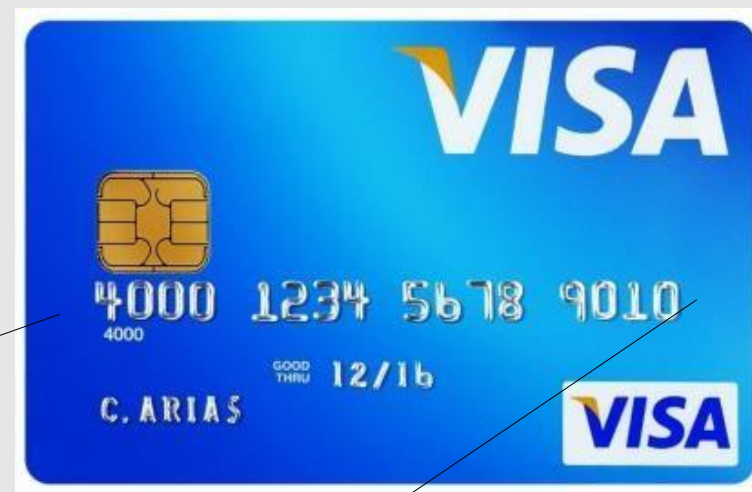
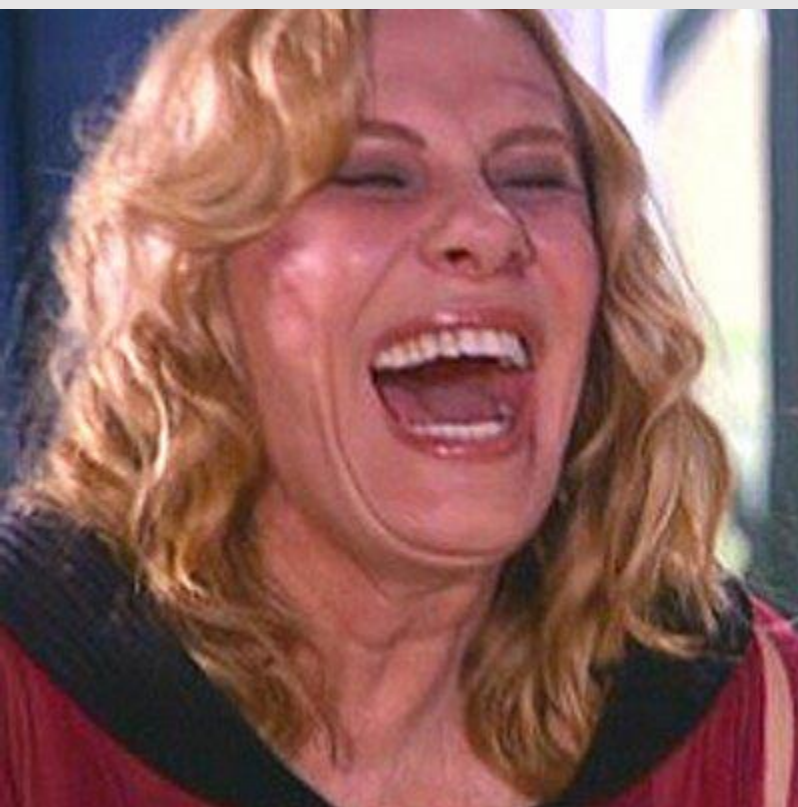


# 4000123456789010

---



Mas o que  
é isso?



“4000123456789010”



# Dados Estruturados

---

- Dados de um mesmo grupo (classe, tabela) possuem as **mesmas descrições** (metadados - atributos)
  - Descrições de atributos de um grupo possuem o mesmo **formato (esquema)**
- Ou seja, possuem um **esquema rígido** e **claro** semanticamente
  - **Que foi previamente projetado**

# Dados Estruturados

---

**SGBD  
Relacional**

Table public.aluno	
Properties	Definition
Inherits	Like
Columns	Constraints
Auto-vacuum	Privileges
Column name	Definition
matricalu	integer NOT NULL DEFAULT nextval('aluno_matricalu_seq'::...
nomealu	character varying(40)
dataaniver	date DEFAULT '1995-01-01'::date
sexo	character(1)

# Dados Estruturados

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
nu_notific	tp_notific	co_cid	dt_notificacao	ds_semana_notificacao	notificacao_ano	co_uf_not	co_munic	id_region	co_unidad	dt_diagn	ds_semana	dt_nascim	nu_idade	tp_sexo	t
2586486	2	A90	01/10/2016	201602	2016	26	261160	1497	6530389	#####	201503	#####	4041	F	
2319522	2	A90	04/06/2016	201614	2016	26	261160	1497	6508960	#####	201514	#####	4008	M	
2293475	2	A90	02/01/2016	201605	2016	26	261160	1497	7775504	#####	201548	#####	4053	F	
2320326	2	A90	02/11/2016	201606	2016	26	261160	1497	28924	#####	201535	#####	4044	M	
2161354	2	A90	01/07/2016	201601	2016	26	261160	1497	6488315	#####	201549	#####	4051	M	
2126681	2	A90	01/11/2016	201602	2016	26	261160	1497	22411	#####	201549	#####	4041	F	
2444616	2	A90	09/09/2016	201636	2016	26	261160	1497	477	#####	201540	#####	4023	M	
2154534	2	A90	02/04/2016	201605	2016	26	261160	1497	6508561	#####	201549	#####	4054	F	
2161087	2	A90	01/08/2016	201601	2016	26	261160	1497	7775504	#####	201548	#####	4032	F	
2151888	2	A90	01/11/2016	201602	2016	26	261160	1497	28924	#####	201548	#####	4050	F	
2161089	2	A90	01/06/2016	201601	2016	26	261160	1497	7775504	#####	201549	#####	4032	F	
2134045	2	A90	01/02/2016	201552	2016	26	260960	1497	6443397	#####	201552	#####	4041	F	
2171068	2	A90	01/03/2016	201601	2016	26	261160	1497	20516	#####	201552	#####	4013	F	
261	2	A90	01/05/2016	201601	2016	26	261160	1497	20516	#####	201552	#####	4009	F	
335	2	A90	01/06/2016	201601	2016	26	261160	1497	20516	#####	201552	#####	4012	M	
2155667	2	A90	01/04/2016	201601	2016	26	261160	1497	6503640	#####	201552	#####	3004	M	
2291538	2	A90	01/11/2016	201602	2016	26	261160	1497	20516	#####	201552	#####	4036	F	
2586495	2	A90	03/10/2016	201610	2016	26	261160	1497	6530389	#####	201552	#####	4056	F	
2159462	2	A90	01/04/2016	201601	2016	26	261160	1497	6508960	#####	201552	#####	4048	F	

Planilha

# Dados Semiestruturados

---

- **Não existe um esquema padrão para os dados**
  - Coleções de dados são definidos de maneiras diferentes, contendo informações “incompletas”
  - Parte dos dados disponíveis podem ter uma estrutura
  - Definição de **esquema à posteriori**
    - Em geral, esquemas são definidos após a existência dos dados



# Dados Semiestruturados - XML

---

```
<musicartist>
  <artist>Michael Jackson</artist>
  <cd>
    <name>this is it</name>
    <genre>rock</genre>
    <track>Jam</track>
  </cd>
  <cd>
    <name>Michael Jackson:The Stripped Mixes</name>
    <genre>rock</genre>
    <track>Ben</track>
  </cd>
</musicartist>
```

# Dados Semiestruturados - JSON

---

```
{  
  "storyTitle": "Pablo Escobar - meu pai: As histórias que não deveríamos saber",  
  "appearance": "eBook",  
  "edition": "2nd Edition",  
  "language": "Portuguese",  
  "genre" : "Biograph",  
  "isbn" : "978-85-422-0597-8",  
  "object": "virtual"  
}
```

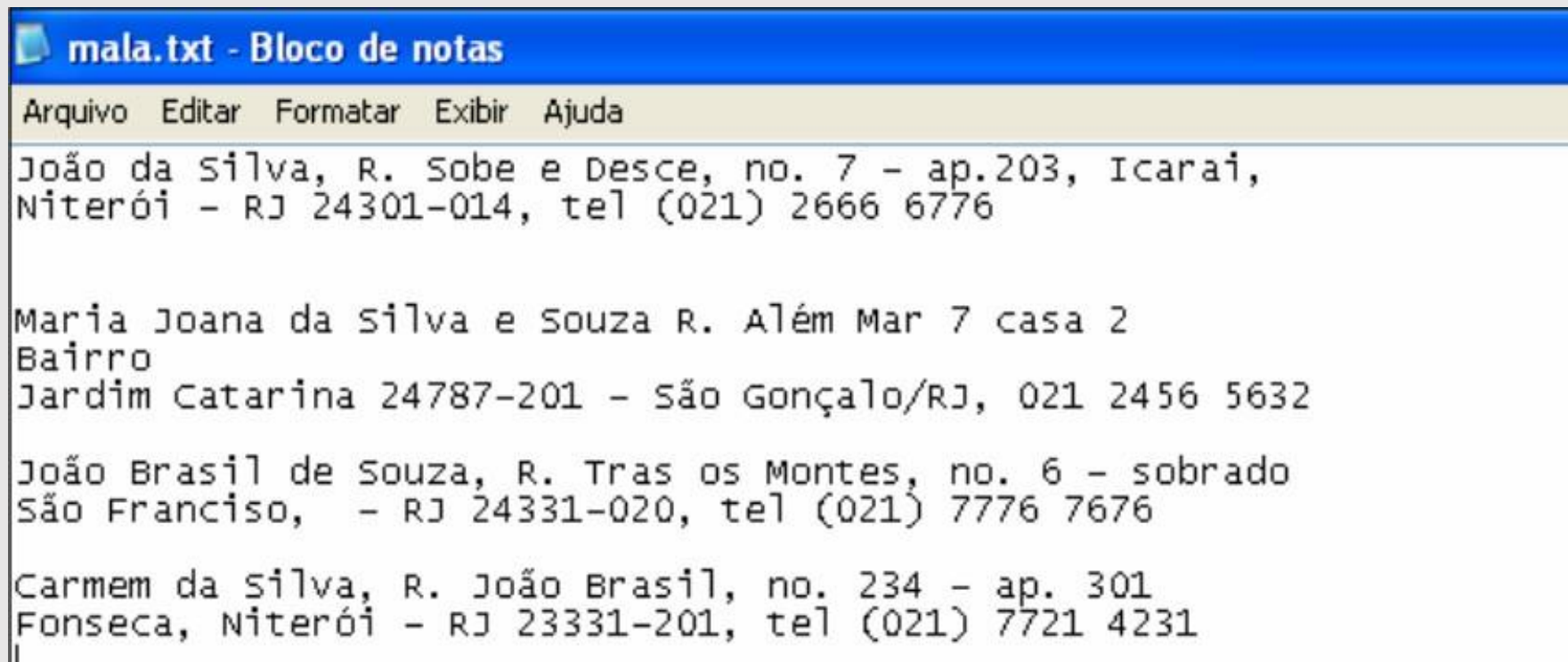
# Dados Não Estruturados

---

- São os dados que **não possuem uma estrutura definida**
- Não há preocupação com campos, restrições e limites.
- Normalmente caracterizados por documentos textos, imagens, vídeos, etc.
  - Dados de páginas web, emails, documentos (ex: PDF, TXT), dados de **sensores**

# Dados Não Estruturados - TXT

---



```
mala.txt - Bloco de notas
Arquivo  Editar  Formatar  Exibir  Ajuda

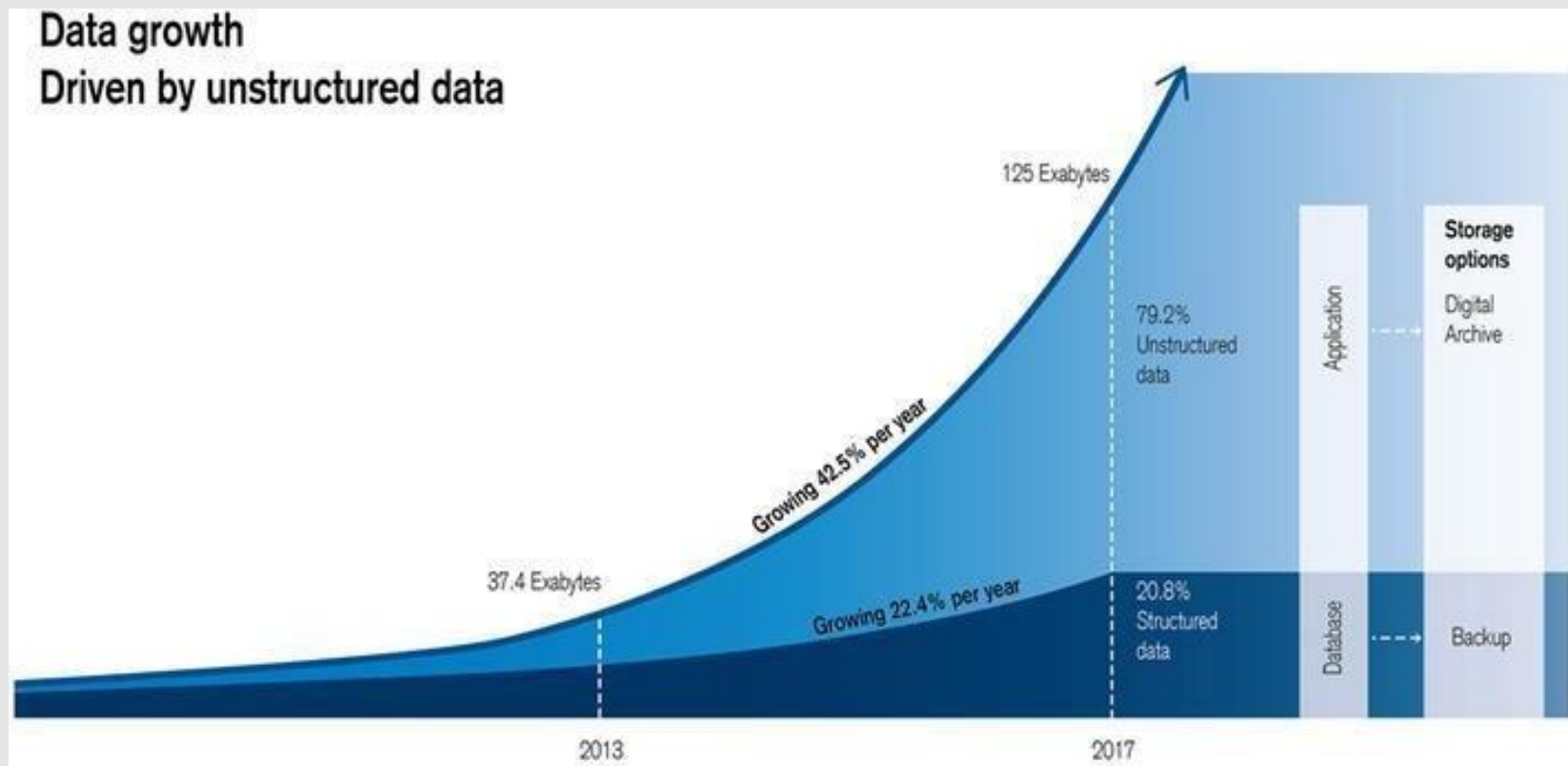
João da Silva, R. Sobe e Desce, no. 7 - ap.203, Icaraí,
Niterói - RJ 24301-014, tel (021) 2666 6776

Maria Joana da Silva e Souza R. Além Mar 7 casa 2
Bairro
Jardim Catarina 24787-201 - São Gonçalo/RJ, 021 2456 5632

João Brasil de Souza, R. Tras os Montes, no. 6 - sobrado
São Francisco, - RJ 24331-020, tel (021) 7776 7676

Carmem da Silva, R. João Brasil, no. 234 - ap. 301
Fonseca, Niterói - RJ 23331-201, tel (021) 7721 4231
|
```

# Dados Estruturados x Não Estruturados



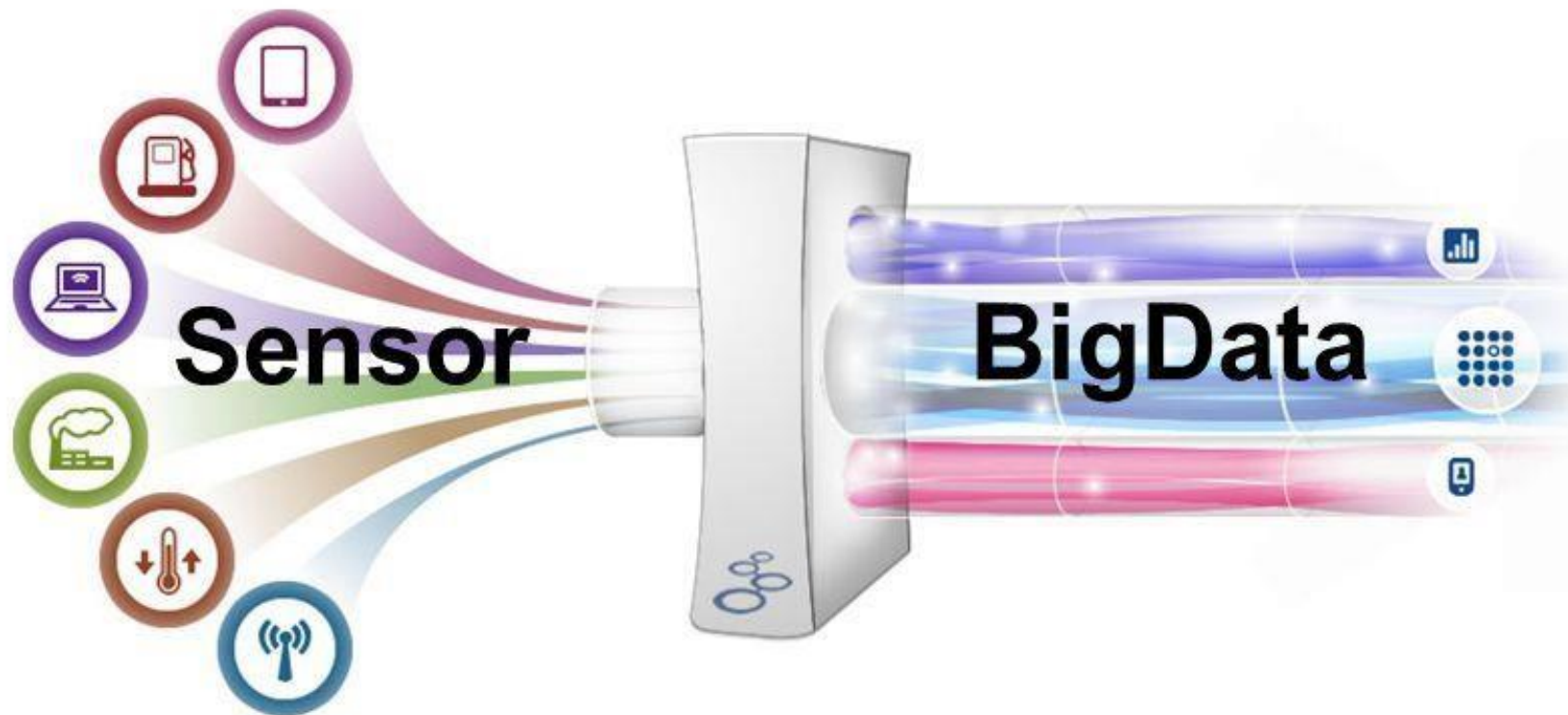
<https://towardsdatascience.com/everything-a-data-scientist-should-know-about-data-management-6877788c6a42>

# Desafios do mundo BIG DATA



# Desafios do mundo BIG DATA

---





# Desafios do mundo BIG DATA

---



**DADO É O NOVO PETRÓLEO!**  
Precisamos encontrá-lo, extraí-lo,  
refiná-lo, distribuí-lo e monetizá-  
lo!

David Buckingham



**Fonte:** <https://c2ti.com.br/blog/big-data-entenda-o-que-e-e-para-que-serve-esta-tecnologia-revolucionaria-inovacao>

# Grandes Dados Grandes Desafios



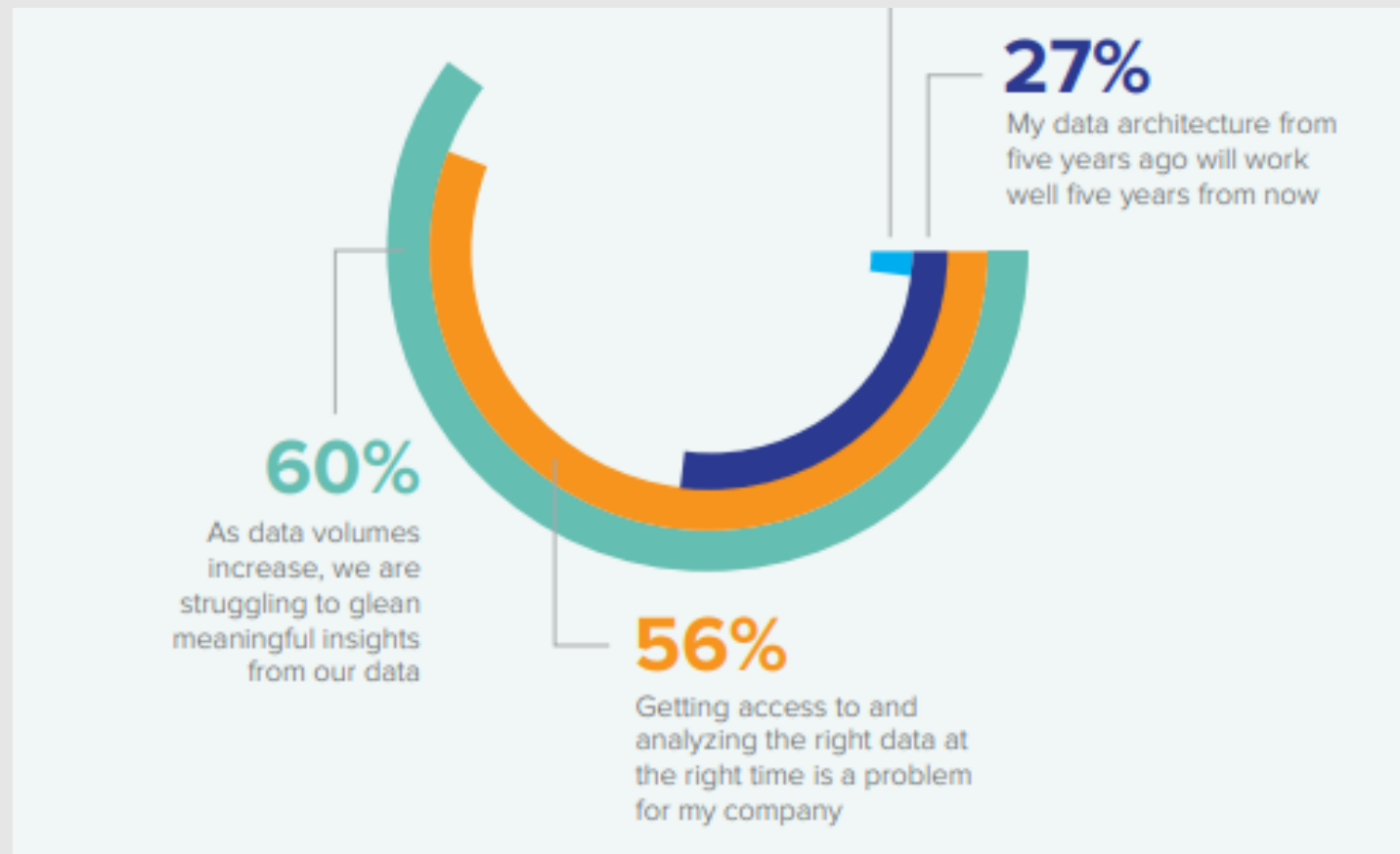


# Grandes Dados, Grandes Desafios

- As empresas modernas **coletam** mais dados do que nunca, de uma ampla variedade de fontes e em uma ampla variedade de formatos.
- Em uma pesquisa recente (**Researchscape**), os executivos entrevistados relataram que os **volumes de dados tem aumentado** e é desafiador obter *insights* significativos de seus dados (60%), e que obter **acesso** e **analisar** os dados certos no momento certo é um problema (56%).



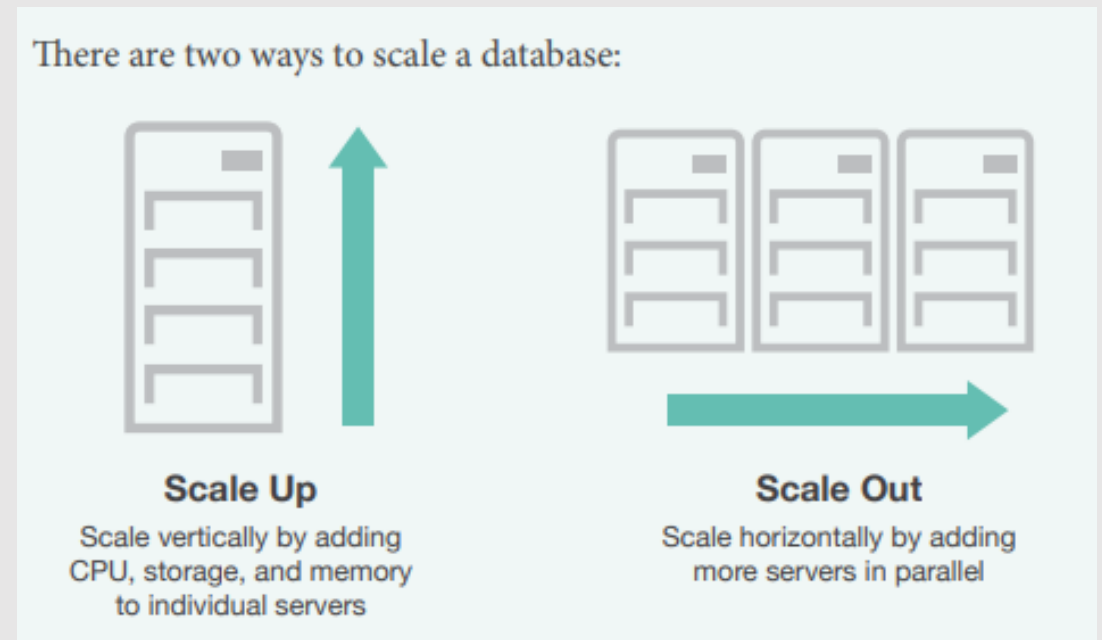
- Apenas um quarto dos executivos acreditava que sua arquitetura de dados de cinco anos atrás funcionará bem daqui a cinco anos (27%).



# Grandes Dados, Grandes Desafios

---

- Volume de dados vem crescendo em ritmo acelerado
- Milhares de usuários acessando dados simultaneamente



=> Demanda por **escalabilidade** é cada vez maior

# Como prover escalabilidade no Relacional?

---

## ➤ Estratégia #1: Escalonamento vertical

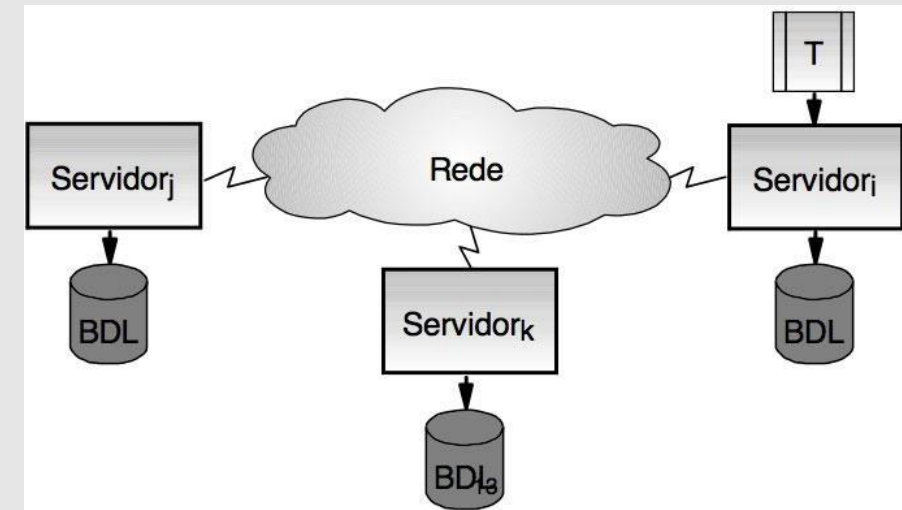
- Aumento da capacidade do servidor
- Mais capacidade de processamento
- Mais memória
- Mais espaço em disco

Abordagem **limitada** e normalmente **cara!!**

# Como prover escalabilidade no Relacional?

---

- Estratégia #2: Escalonamento horizontal
  - Aumentar o número de servidores
  - Distribuir os dados e o processamento em muitos servidores
  - Configurar para que a **distribuição**, **sincronização** e **consistência** sejam mantidos



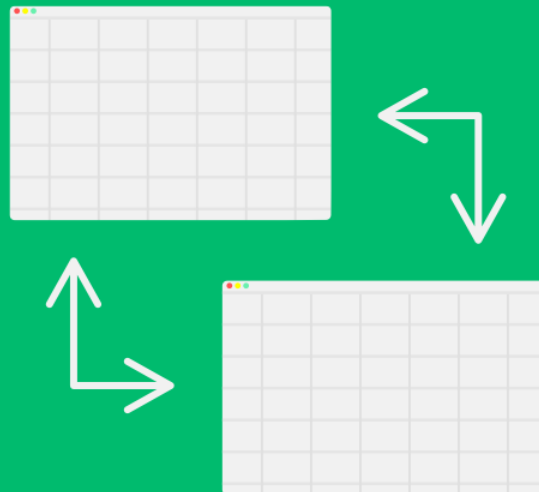
A manutenção da **consistência** pode ser mais difícil!!





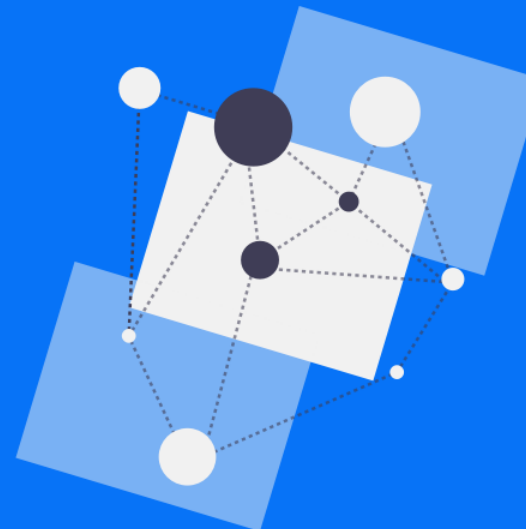
**E agora,  
quem poderá  
nos defender?**

# SQL



vs

# NoSQL



!!

*NoSQL = Not Only SQL*

!!

Não apenas SQL ou Não  
Relacional

# BD NoSQL

---

- Usar NoSQL não envolve necessariamente **descartar** seu BD Relacional.
- NoSQL deve ser pensado como um ferramenta que pode ser usada para **resolver** os novos tipos de **desafios** associados ao big data, de forma eficiente e mais econômica, com um Banco de dados **NoSQL**.

# BD NoSQL

---

- Podem haver processos de negócios que devem continuar a ser abordado de forma eficaz com **BD Relacional**.
- Mas, com o novos desafios apresentados pelo **big data**, você provavelmente enfrentará novos problemas que podem ser resolvidos de forma mais eficiente e econômica com um **BD NoSQL**.

# Consistência Eventual

---

- Princípio: teorema **CAP** (*Consistency, Availability e Partition tolerance*)
- **Consistência**: significa que uma **leitura** de um **item** após **uma escrita** desse item **deve retornar o novo valor**.
- **Disponibilidade**: propriedade de um **sistema** **responder** a todas **as requisições que chegam** a um nó.

# Consistência Eventual

---

- Princípio: teorema **CAP** (*Consistency, Availability e Partition tolerance*)
- **Tolerância à partição**: propriedade de um sistema continuar funcionando mesmo quando um problema ocorre na rede dividindo o sistema em duas ou mais partições, quando nós de uma partição não podem se comunicar com os demais
  - Em sistemas tolerantes à partição, clientes acessando uma partição conseguem ser atendidos

# Classificações NOSQL





# Banco de dados NoSQL

Classificado pelos modelos:

- Chave-valor
- Documentos
- Colunas
- Grafos



# Chave-Valor

---

- Modelo **mais simples**
- Sua estrutura constitui-se de uma lista de pares de valores compostos por uma chave e um valor
  - Assemelha-se a uma **estrutura de Tabela Hash**
- Algumas soluções desse modelo permitem, além dos tipos básicos de dados como numerais e strings, a utilização de **listas** e conjuntos de valores dos tipos básicos
- Todo o acesso é feito por meio das **chaves** de busca e, apenas com a chave, é possível se ter acesso ao **valor**

# Exemplo

---

Estrutura tabular com duas colunas

---

Chave	Valor
CPFUser1	88456707830
NomeUser1	Webber
TelefoneUser1	999999999
CarroUser1	9BWHE21JX24060960
ModeloCarroUser1	Gol 2013
ValorCarroUser1	28.000,00
CPFUser2	76052657278
NomeUser2	Sidarta
TelefoneUser2	[999572110, 988081046]

# Chave-Valor



**Aplicações:** gerenciamento de sessões; paginas que exibem totalizadores ou um ranking (de games, por exemplo); dashboard ou tela que mostra os totalizadores de itens vendidos ou o valor total de vendas do dia anterior,

# Documentos

---

- Cada **documento** é uma **coleção de chaves e valores** que está relacionado a uma instância de dados
- Os dados são agrupados em documentos que podem seguir a codificação XML ou **JSON**
- Os vários documentos pertencentes a um mesmo domínio são armazenados em uma **coleção de documentos**

# Documentos

---

- Um **documento**, em geral, é um **objeto** com um identificador único e um conjunto de campos, que podem ser **strings**, **listas** ou **documentos aninhados**
- Não depende de um esquema rígido
  - **não exige** uma **estrutura fixa** de campos
  - é possível que ocorra uma **atualização** na estrutura do documento, com a **adição** de novos campos, sem **causar problemas**

## Exemplo - JSON

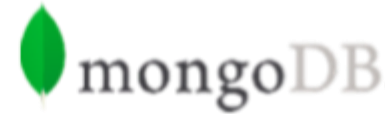
---

```
{
  _id: "88456707830",
  Nome: "William",
  Telefone: "999999999"
}
{
  _id: "76052657278",
  Nome: "Sidartha",
  Telefones: [
    {telephone: "999957211"},
    {telephone: "988081046"}
  ]
}
```

# Documentos



```
BaseDocument baseDocument = new BaseDocument();  
baseDocument.addAttribute(name, value);
```



```
Document document = new Document();  
document.append(name, value);
```



```
JsonObject jsonObject = JsonObject.create();  
jsonObject.put(name, value);
```



```
ODocument document = new ODocument("collection");  
document.field(name, value);
```

- **Aplicações típicas** – apresentação de catálogos de produtos; gerenciamento de conteúdo; listagem de dados em grande volume



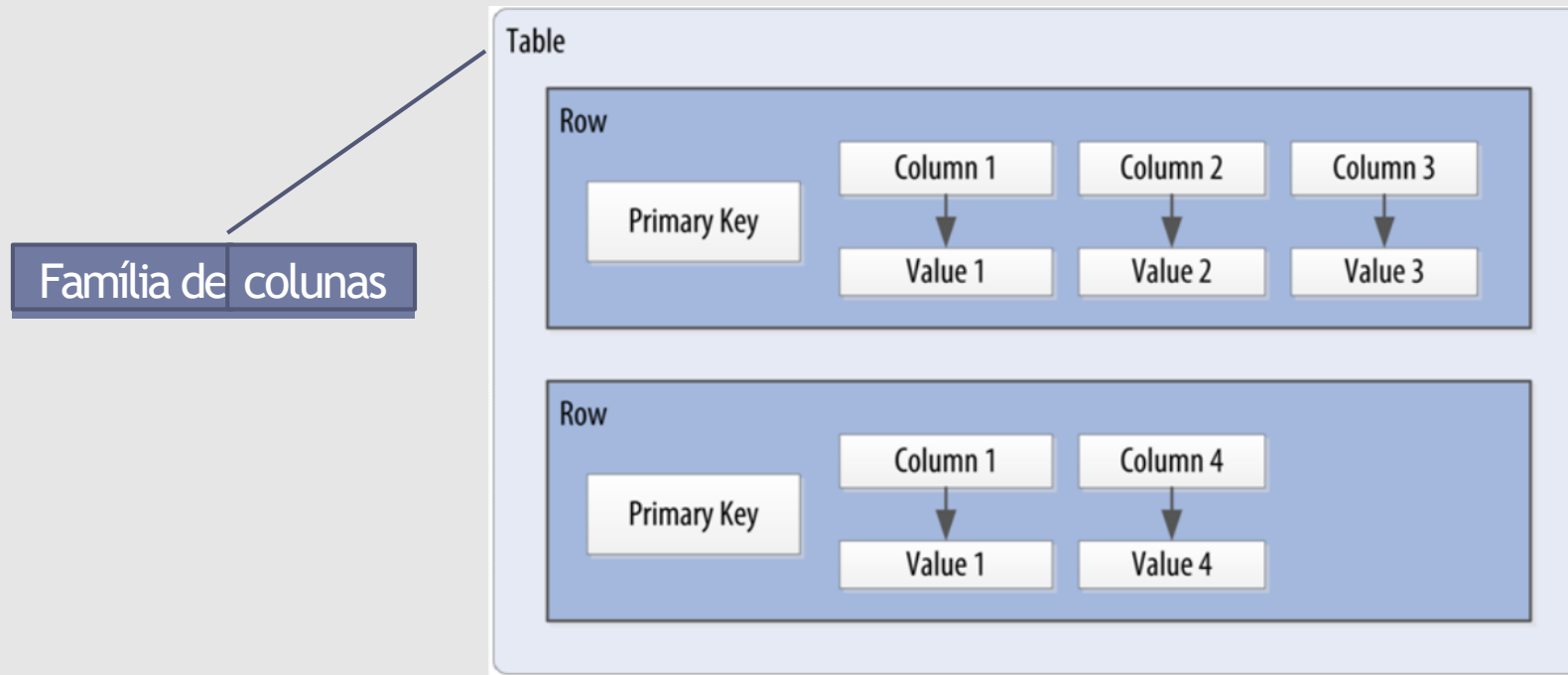
# Colunas

---

- Neste modelo, muda-se o paradigma de orientação a registros ou tuplas (relacional) para orientação a atributos ou colunas
- Os dados são indexados por uma tripla = **linha, coluna e timestamp**
  - Linhas e colunas são identificadas por chaves
  - O timestamp permite diferenciar múltiplas versões de um mesmo dado.
- Família de colunas (*column family*)
  - Usado com o intuito de agrupar colunas que armazenam dados pertencentes a um mesmo tipo (por exemplo, dados de um cliente)

# Colunas

- Inversão na orientação de armazenamento dos dados do modelo Relacional para o modelo em colunas
  - Linhas não precisam ter todos os espaços armazenados.



# Exemplo

---

row key	columns ...			
jbellis	name	email	address	state
	jonathan	jb@ds.com	123 main	TX
dhutch	name	email	address	state
	daria	dh@ds.com	45 2 <sup>nd</sup> St.	CA
egilmore	name	email		
	eric	eg@ds.com		

- Em uma família de colunas, é possível a existência de atributos não atômicos (por exemplo, listas)
- É possível definir colunas como conjunto de subcolunas (*super column family*)
  - Manter dados relacionados juntos

# Exemplo

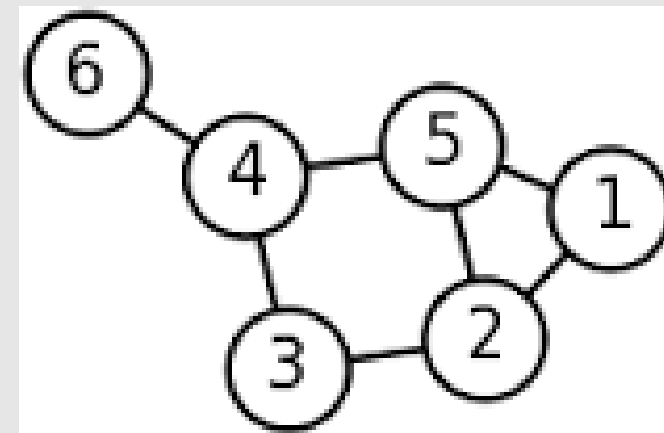
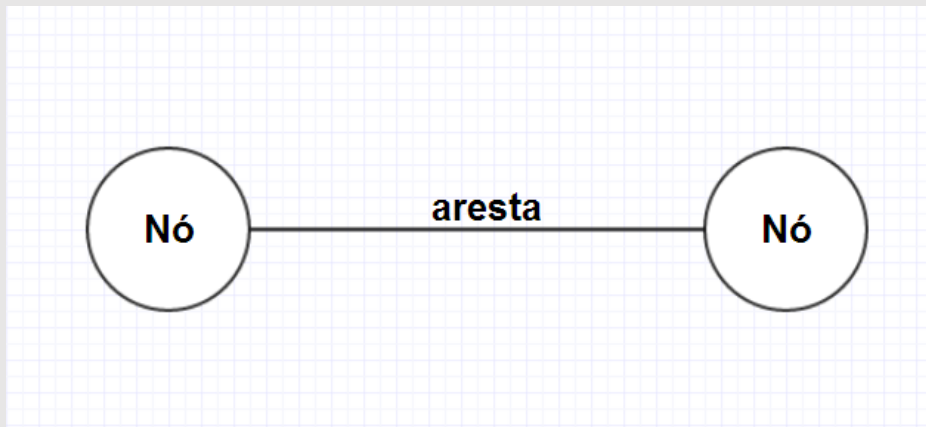
---



- **Aplicação:** sistemas que precisam ler e realizar operações de consultas em grandes volumes de dados (OLAP), para sistemas do tipo OLTP as operações de escrita podem não ser tão vantajosas; mecanismos de recomendação, detecção de fraudes ...

# Grafos

- Enquanto as outras abordagens têm seu foco no armazenamento dos dados, esse modelo tem como **destaque principal os relacionamentos que ocorrem entre as entidades de sua base**
- Estrutura de grafos



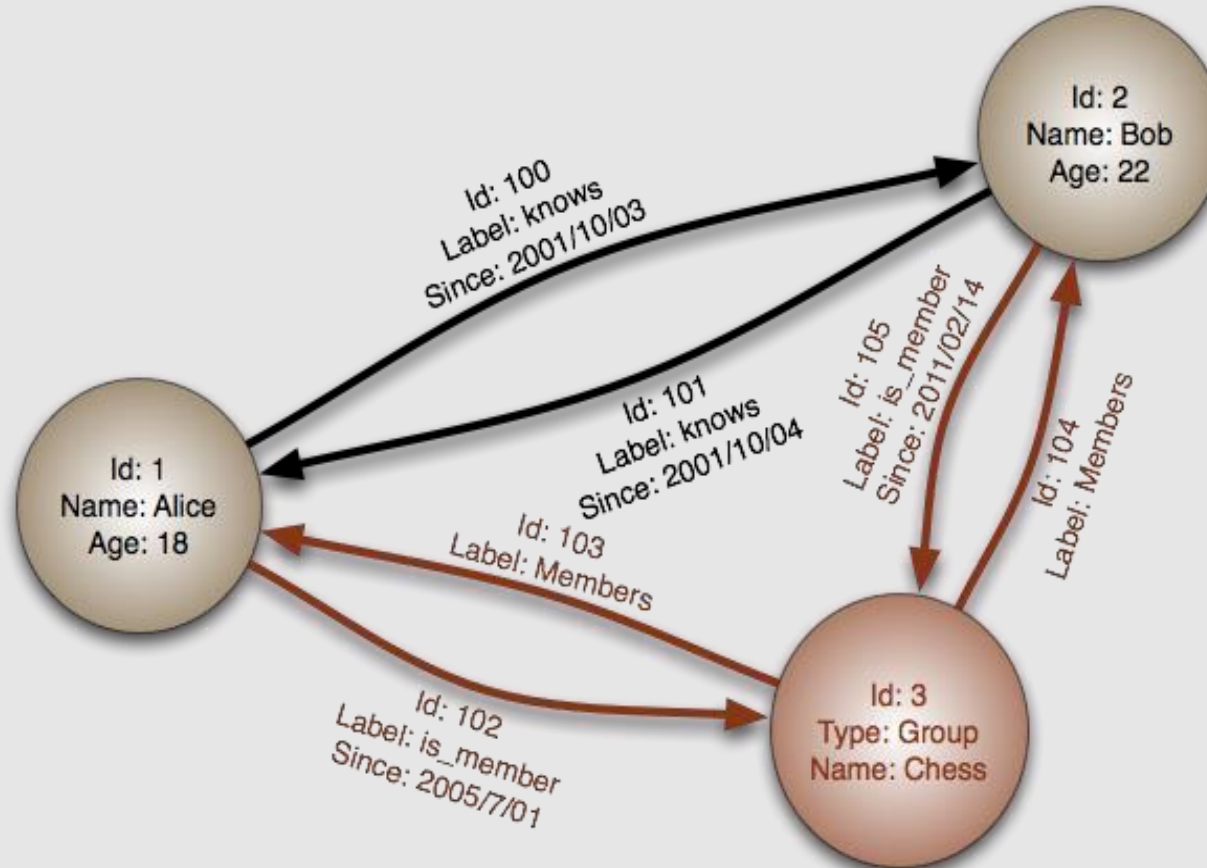
# Grafos

---

- Três tipos de informações:
- **Nós**: instâncias de dados
  - **Arestas**: relacionamentos mantidos entre as instâncias
  - **Propriedades**: atributos com valores de dados
    - podem assumir valores como booleanos, inteiros, caracteres e conjuntos de valores
- Os **nós e arestas** podem conter **rótulos** (espécie de nomenclatura) que os classificam em grupos mais específicos

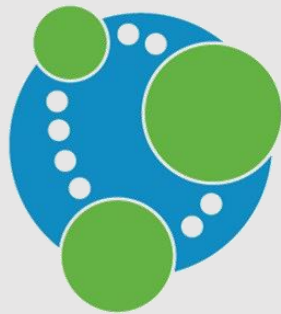
# Exemplo

---



# Exemplo

---



neo4j



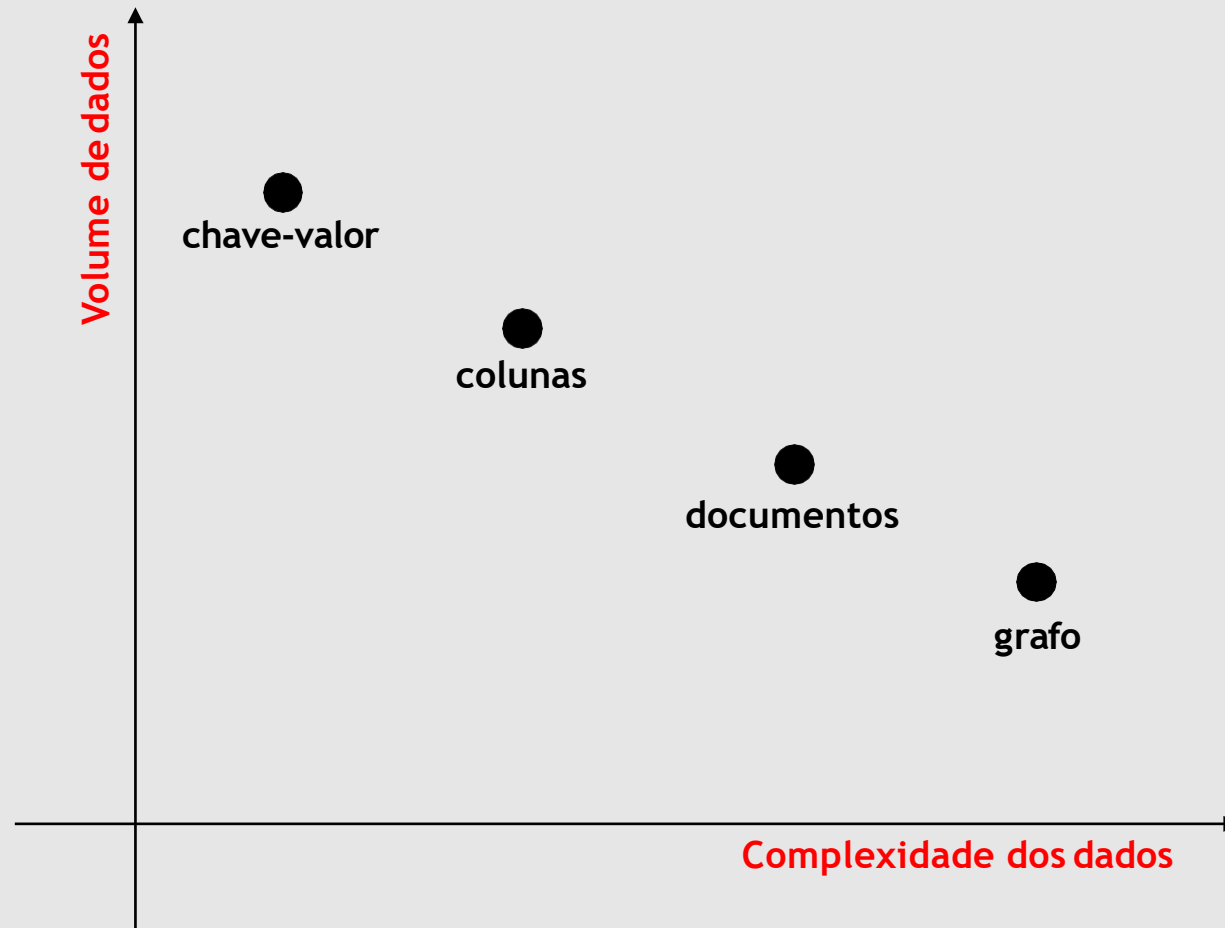
INFINITEGRAPH®

**Aplicações:** redes sociais; recomendações;  
mapear relacionamentos, como sistemas de  
reservas ou gerenciamento de relacionamento com  
o cliente

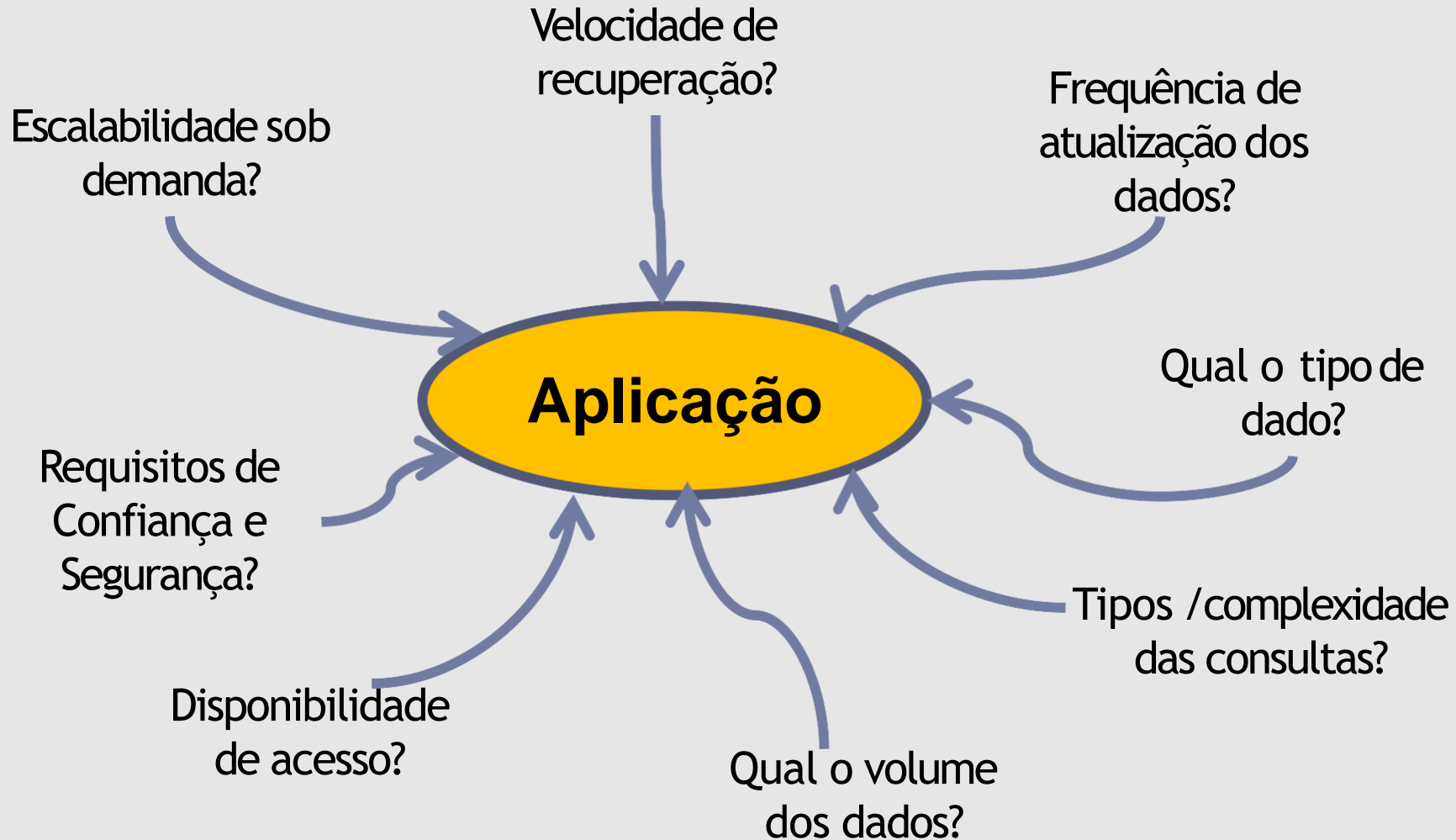


# NoSQL

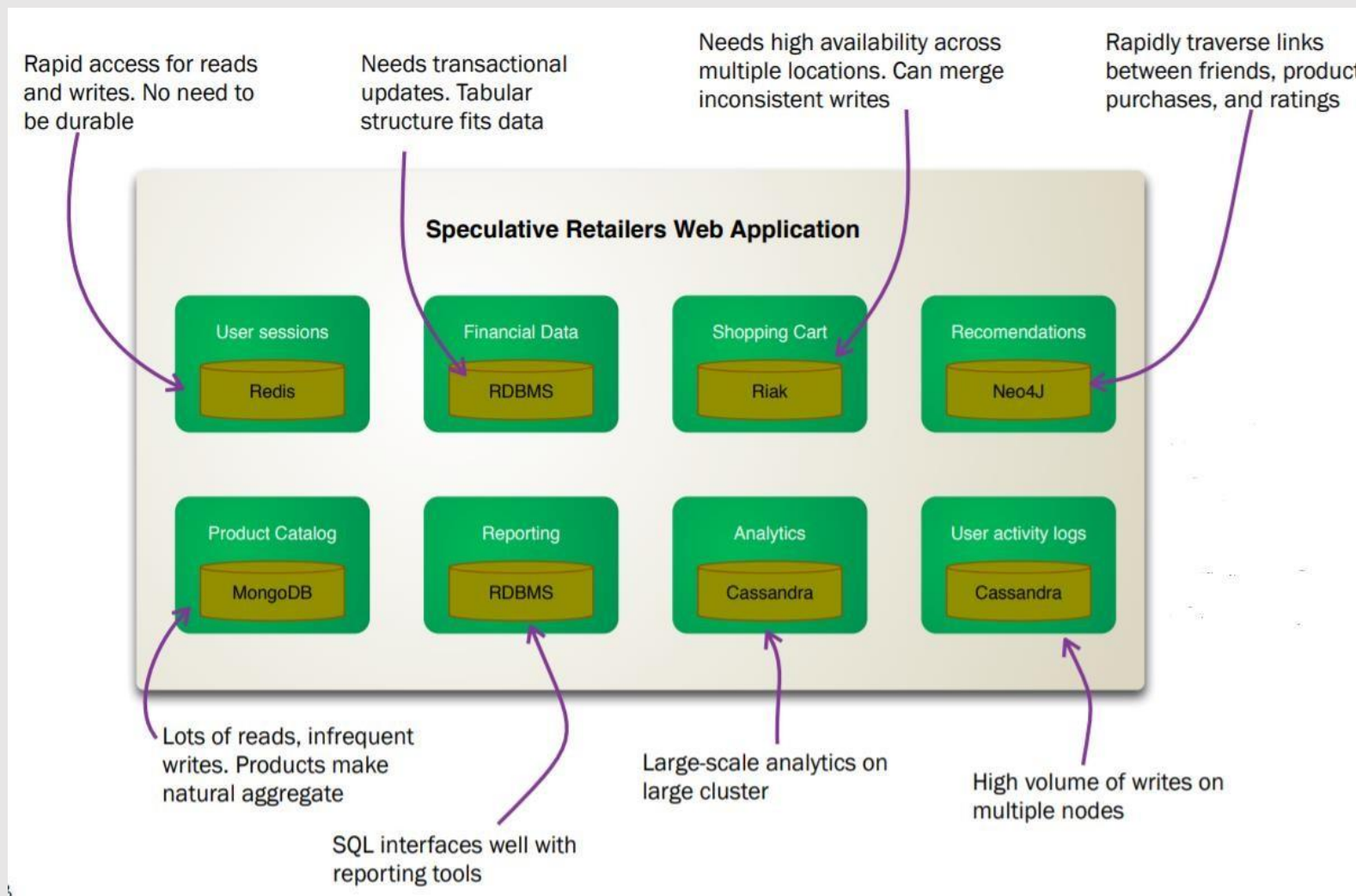
---



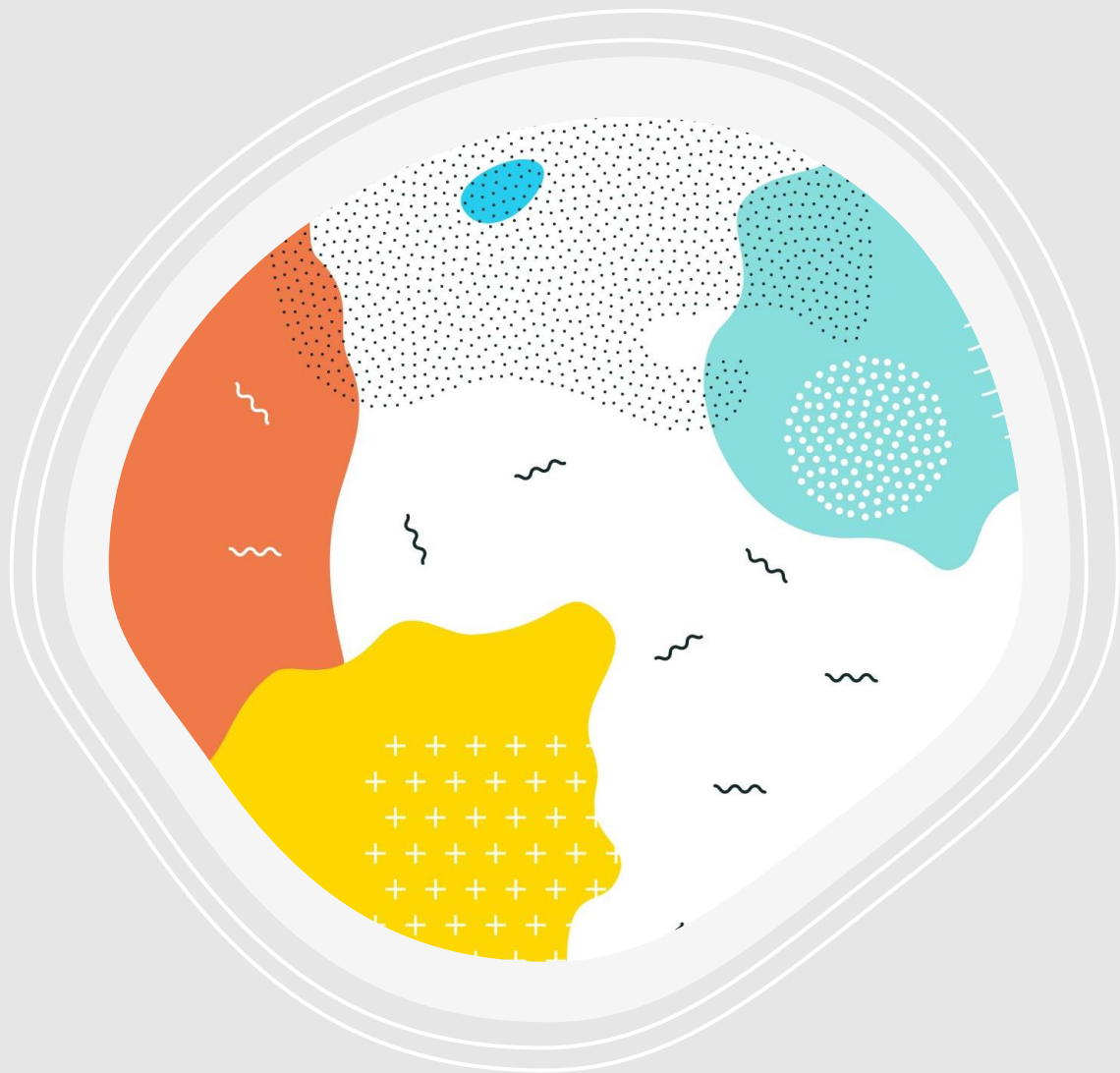
# É preciso analisar a aplicação...



# Solução híbrida - Persistência Poliglota



(Martin Fowler, 2012 - "Persistência Poliglota")

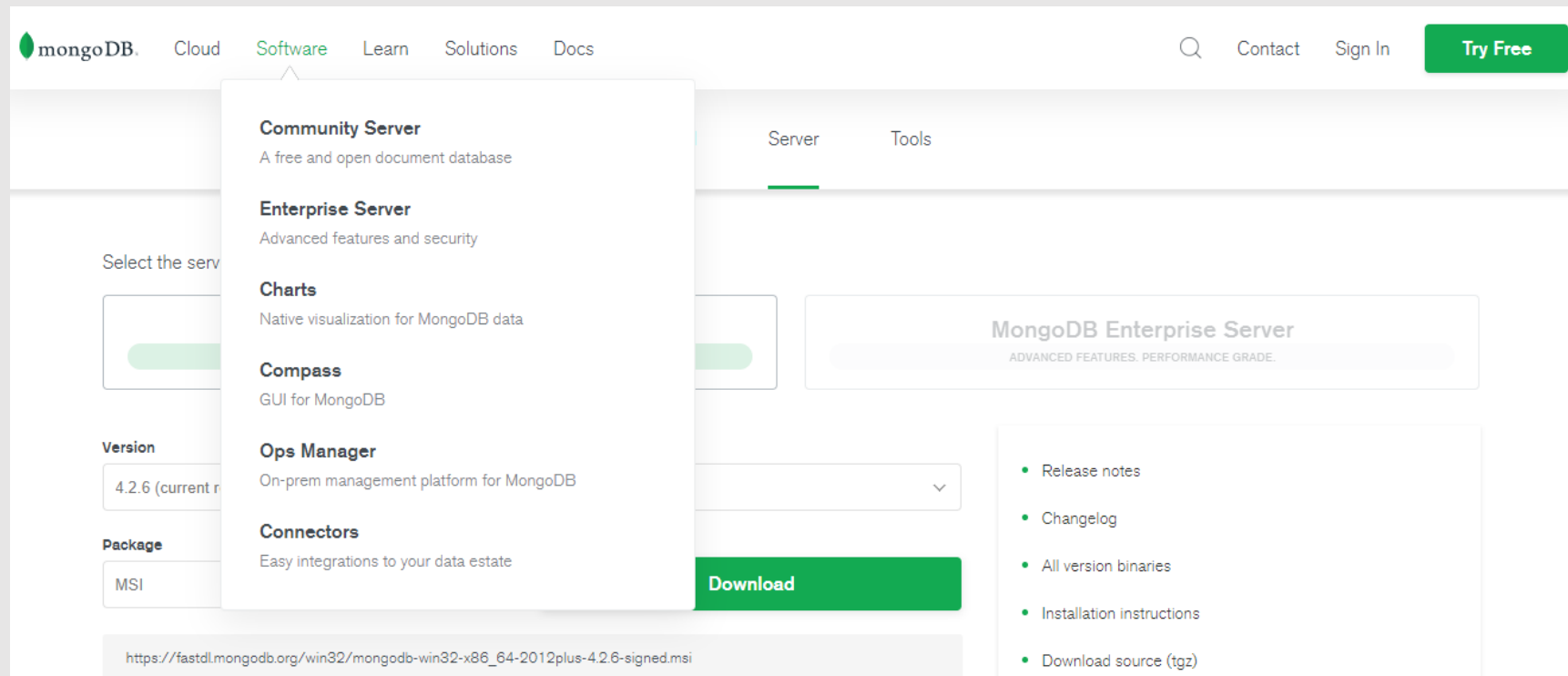


**Por onde  
começaremos?**

# MongoDB

Vamos utilizar o MongoDB e suas tecnologias.

- Faça o download do **banco MongoDB** e do **COMPASS** (IDE)



# MongoDB

## Passos de **CONFIGURAÇÃO**:

Configurando **PATH**: C:\Program Files\MongoDB\Server\4.2\bin

- Computador > Propriedades > Propriedades do Sistema > Avançado > Variáveis de Ambiente > Path
- Criar pasta **DATA > DB (no disco C)**
- Comando **mongod** no **CMD**

# MongoDB

Vamos explorar o serviço online ATLAS:

- <https://www.mongodb.com/pt-br>
- **Crie conta gratuita**
- Vamos criar um **Cluster** e uma instância de **Banco de dados online**

# MongoDB

Vamos explorar na prática, conectando a uma aplicação web local

**Continuaremos isso no próximo encontro!**



# Contatos

[Professordanielbrandao@gmail.com](mailto:Professordanielbrandao@gmail.com)

@profdanielbrandao (Instagram)

**<http://bit.ly/arquivosnosql20>**

# Referências

---

GRUS, Joel. **Data Science do Zero**. São Paulo: Alta Books, 2017.

BRAGHITTONI, Ronaldo. **Business Intelligence - Implementar do jeito certo e a custo zero**. São Paulo: Casa do Código, 2017.

Arquivos disponíveis em: **<http://bit.ly/arquivosnosql20>**