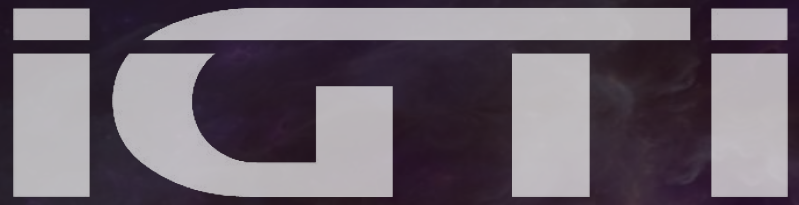


A aula interativa do **Bootcamp Online – Analista de Banco de Dados** começará em breve!

Atenção:

1) Acesse a aula com NOME COMPLETO, para que sua frequência seja computada.
2) Mantenha o microfone DESLIGADO, abrindo-o apenas em momentos de interatividade.
3) Mantenha seu vídeo sempre ATIVADO.

Banco de Dados NoSQL

PRIMEIRA AULA INTERATIVA

PROF. RICARDO BRITO ALVES

Nesta aula

- Apresentação do Professor.
- Trabalho Prático.
- Tópicos da Disciplina e temas interessantes.
- Desafio.

Apresentação do Professor

Apresentação do Professor

Ricardo Brito Alves

Formação Acadêmica:

- Graduado em Ciência da Computação pela Pontifícia Universidade Católica de Minas Gerais,1994.
- MBA em Gestão Estratégica de Projetos pela Una, 2009.
- Mestrado em Engenharia Elétrica pela Pontifícia Universidade Católica de Minas Gerais, 2018.
- Doutorando em Ciência da Computação pela Universidade Federal de Minas Gerais, 2024.

Experiência Profissional:

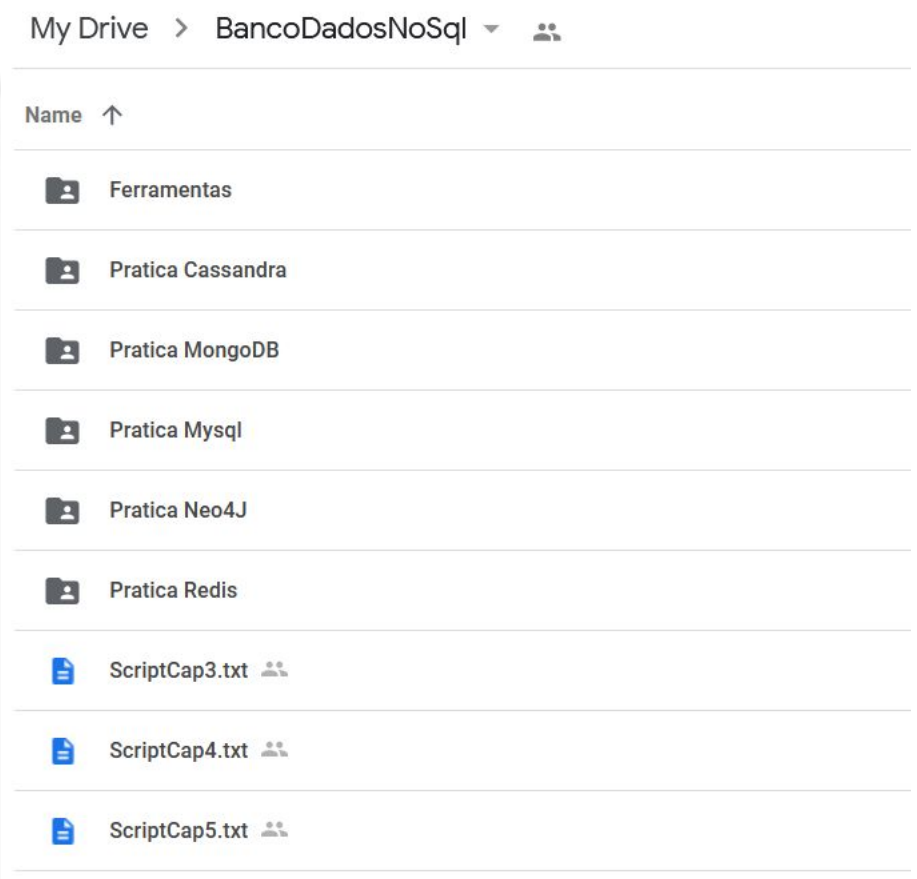
- Desde 2002 atua com projetos de Data Mining e BI.
- Atua há 7 anos na área de Inteligência Artificial.

Trabalho Prático

Tópicos da Disciplina e Temas Interessantes

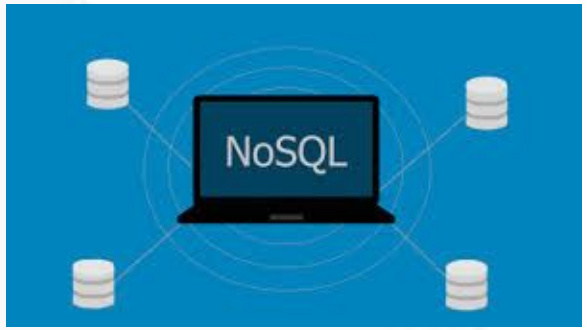
Material

https://drive.google.com/drive/u/1/folders/176_xlNV4obL2fbVeKFh9oKqpzv6_dfPp



NoSQL - Características

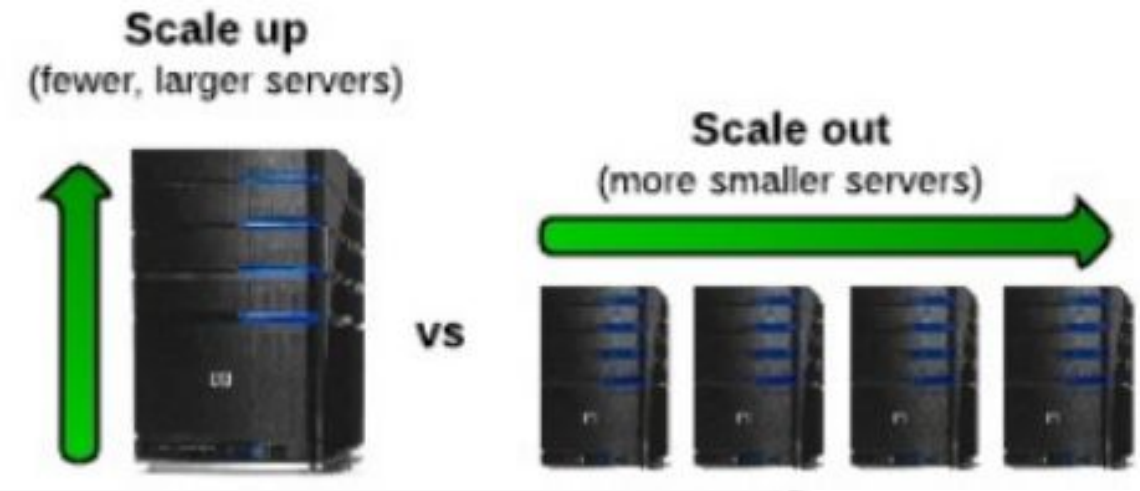
- Escalabilidade horizontal.
- Ausência de esquema ou esquema flexível.
- Suporte a replicação.
- API simples.
- Nem sempre prima pela consistência.



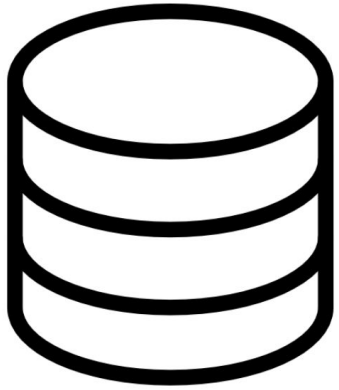
Escalabilidade Horizontal

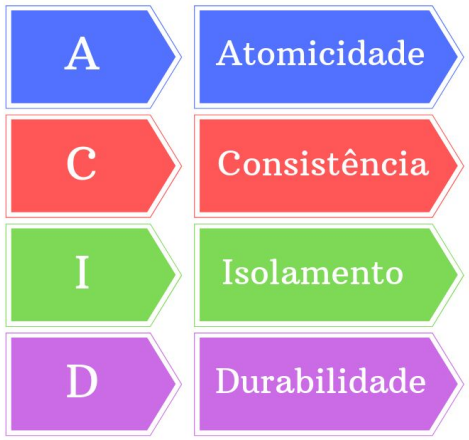
À medida em que o volume de dados cresce, aumenta-se a necessidade de escalabilidade e melhoria do desempenho.

Podemos escalar ***verticalmente*** (adicionar CPU, memória e disco) ou podemos escalar ***horizontalmente*** (adicionar mais nós).



Propriedades ACID





Atomicidade – estado em que as modificações no BD devem ser todas ou nenhuma feita. Cada transação é dita como “atômica”. Se uma parte desta transação falhar, toda transação falhará.

Consistência – estado que garante que todos os dados serão escritos no BD.

Isolamento – requer que múltiplas transações que estejam ocorrendo “ao mesmo tempo”, não interfiram nas outras.

Durabilidade – garante que toda transação submetida (commit) pelo BD não será perdida.

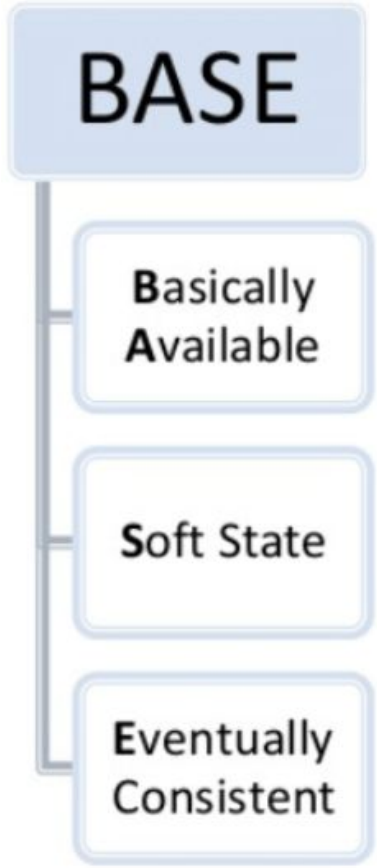
Propriedades BASE

Basically Available – Basicamente Disponível.

Soft-State – Estado Leve.

Eventually Consistent – Eventualmente Consistente.

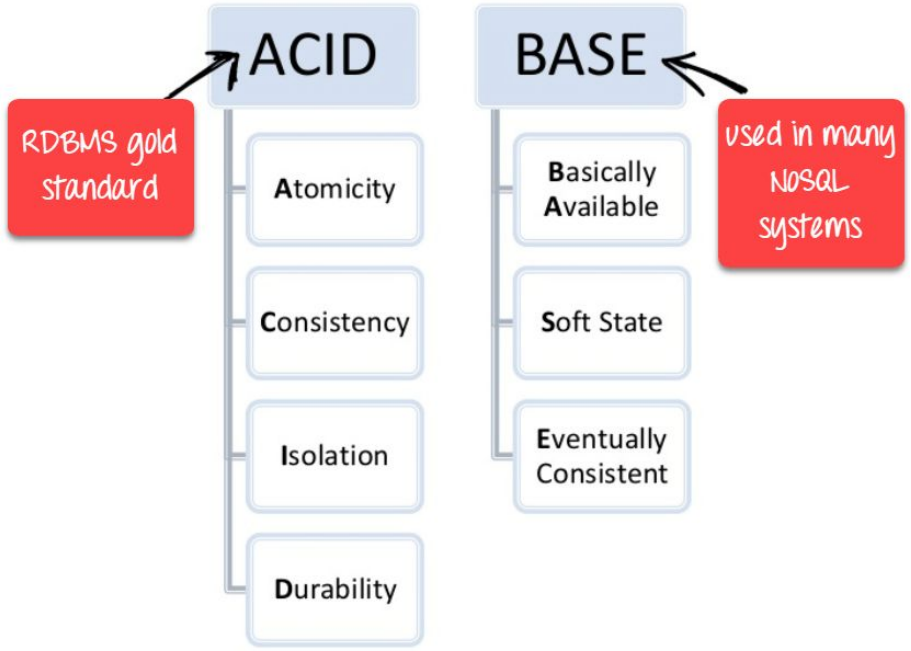
Uma aplicação funciona basicamente todo o tempo (Basicamente Disponível), não tem de ser consistente todo o tempo (Estado Leve) e o sistema torna-se consistente no momento devido (Eventualmente Consistente).



ACID vs BASE

BDs Relacionais trabalham com **ACID** (Atomicidade, Consistência, Isolabilidade, Durabilidade).

BDs NoSQL trabalham com **BASE** (Basicamente Disponível, Estado Leve, Eventualmente consistente).



CAP

Consistency – Consistência.

Availability – Disponibilidade.

Partition Tolerance – Tolerância ao Particionamento.

Consistência

Consistência (**C**onsistent):

As escritas são atômicas e todas as requisições de dados subsequentes obtêm o novo valor. Assim, é garantido o retorno do dado mais atualizado armazenado, logo após ter sido escrito. Isso independe de qual nó seja consultado pelo cliente – o dado retornado para a aplicação será igual.

Disponibilidade

Disponibilidade (**A**vailable):

O banco de dados sempre retornará um valor desde que ao menos um servidor esteja em execução – mesmo que seja um valor defasado.

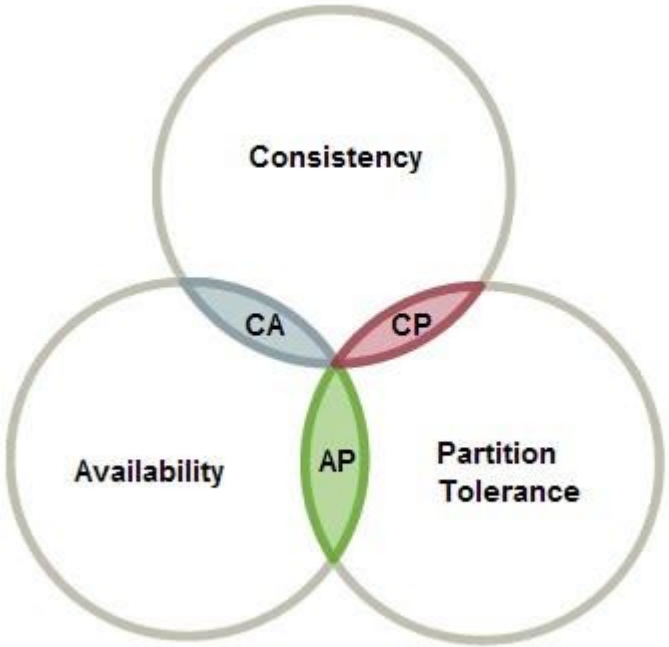
Tolerância ao Particionamento

Tolerância ao Particionamento (**P**artition Tolerant) ou Tolerante a Falhas.

O sistema ainda irá funcionar mesmo se a comunicação com o servidor for temporariamente perdida. Assim, se houver falha de comunicação com um nó da rede distribuída, os outros nós poderão responder às solicitações de consultas de dados dos clientes.

Teorema CAP

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

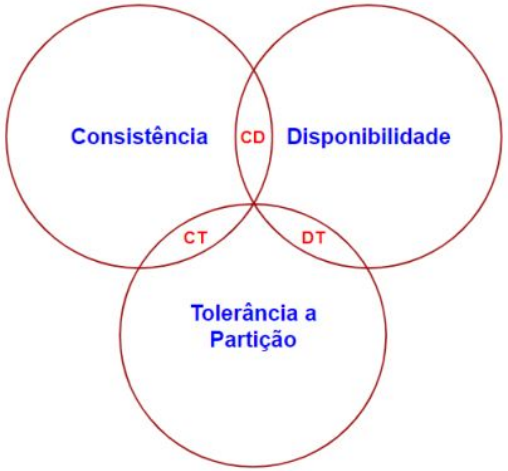


Consistência Eventual

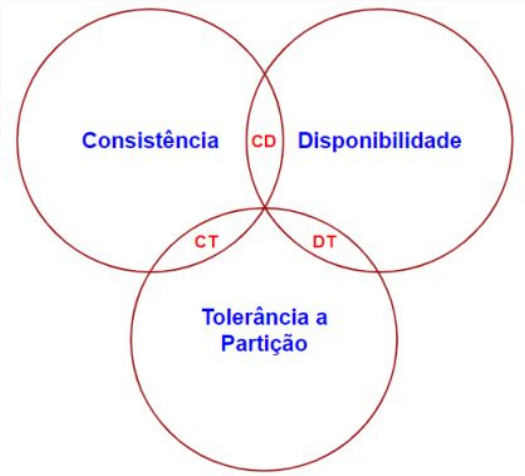
É um conceito interessante derivado do teorema CAP.

O sistema prioriza as escritas de dados (armazenamento), sendo o sincronismo entre os nós do servidor realizado em um momento posterior – o que causa um pequeno intervalo de tempo, no qual o sistema como um todo é inconsistente.

Para isso, ***são implementadas as propriedades Disponibilidade e Tolerância a Partição.***



Consistência Eventual



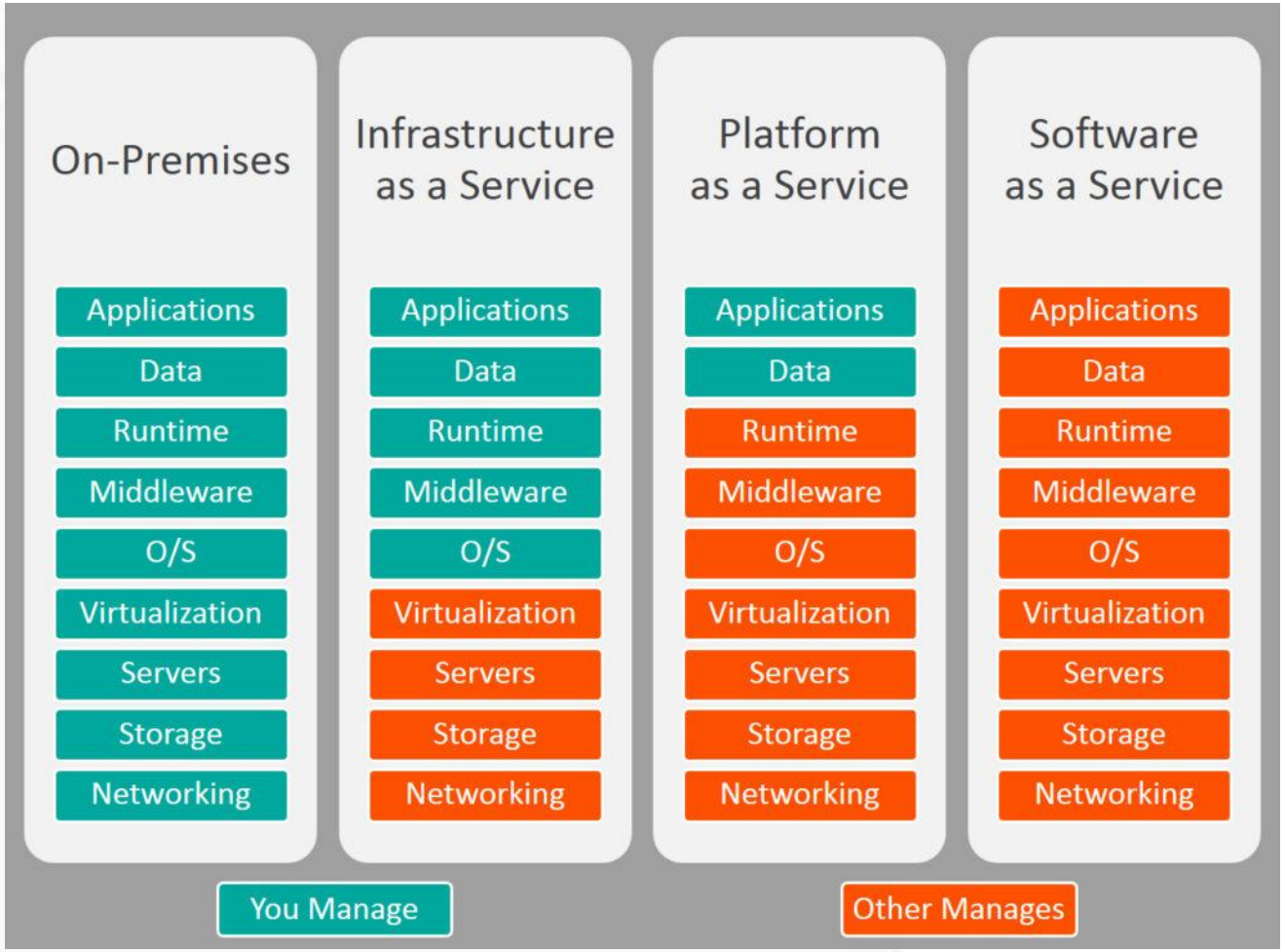
Exemplos de sistemas de bancos de dados que implementam a consistência eventual são o ***MongoDB, Cassandra e RavenDB (bancos NoSQL),*** entre outros.

Em Bancos Relacionais é muito comum implementar as propriedades ***Consistência*** e ***Disponibilidade***. Como exemplos, citamos os SGBDRs ***Oracle, MySQL, PostgreSQL, SQL Server*** e outros.

Ao criar um banco de dados distribuído é importante ***ter em mente o teorema CAP***.

Você terá de decidir se o banco será consistente ou disponível, pois bancos de dados distribuídos são sempre tolerantes à partição.

Computação em Nuvem



Arquiteturas Monolíticas

Com as arquiteturas monolíticas, todos os processos são altamente acoplados e executam como um único serviço.

Isso significa que se um processo do aplicativo apresentar um pico de demanda, toda a arquitetura deverá ser escalada. A complexidade da adição ou do aprimoramento de recursos de aplicativos monolíticos, aumenta com o crescimento da base de código. Essa complexidade limita a experimentação e dificulta a implementação de novas ideias.

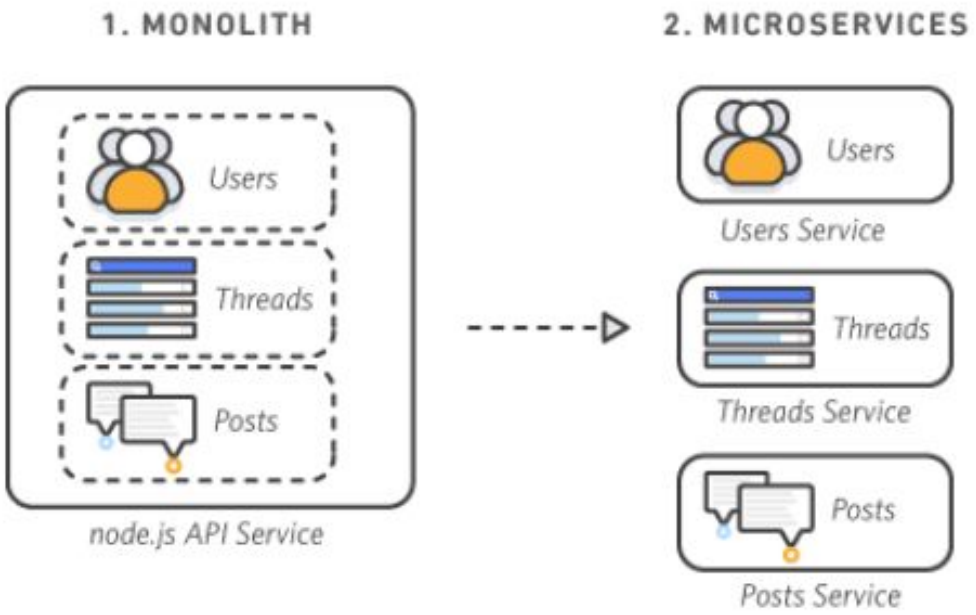
As arquiteturas monolíticas aumentam o risco de disponibilidade de aplicativos, pois muitos processos dependentes e altamente acoplados aumentam o impacto da falha de um único processo.

Arquitetura de Microsserviços

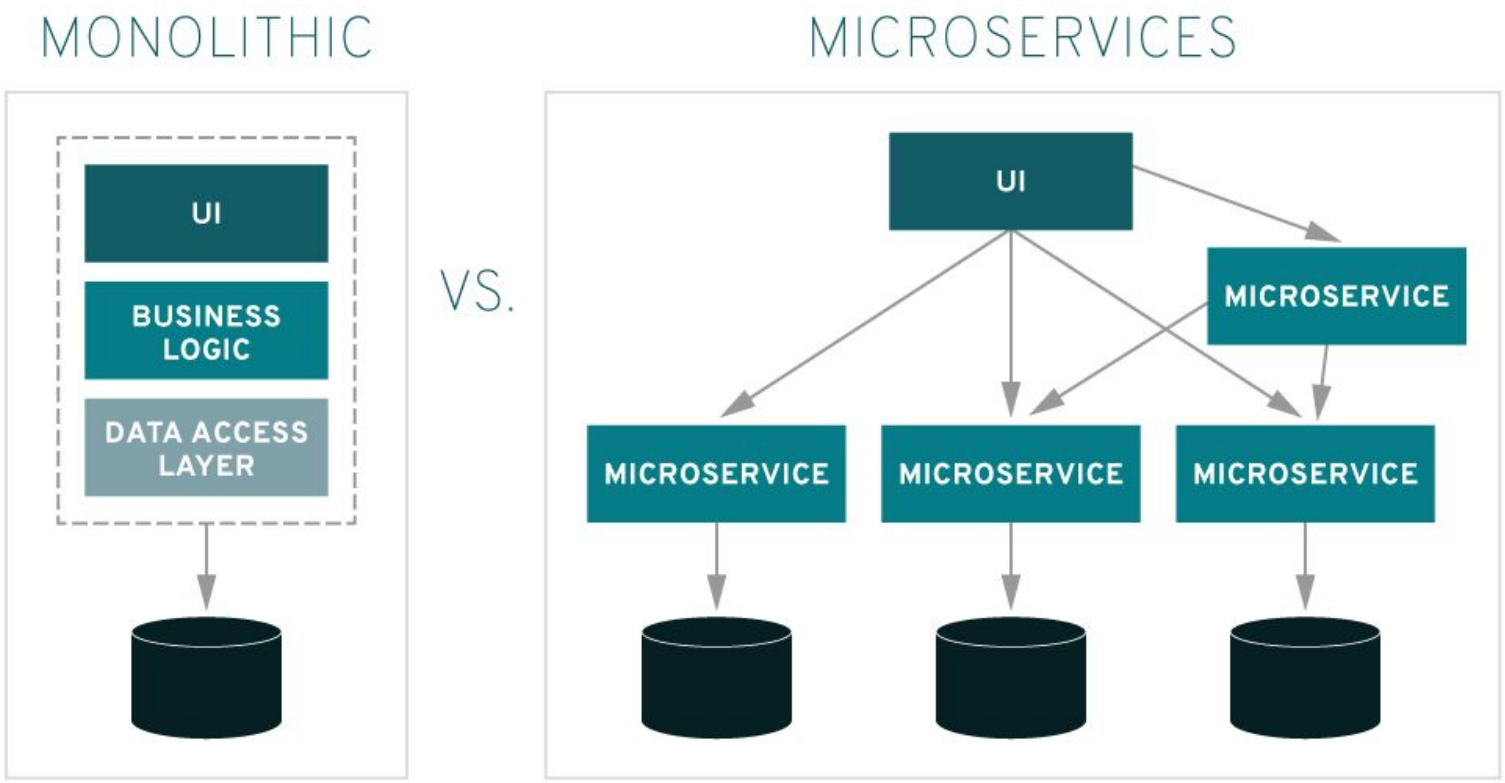
Com uma arquitetura de microsserviços, um aplicativo é criado como componentes independentes que executam cada processo do aplicativo como um serviço.

Esses serviços se comunicam por meio de uma interface bem definida usando APIs leves. Os serviços são criados para recursos empresariais e cada serviço realiza uma única função. Como são executados de forma independente, cada serviço pode ser atualizado, implantado e escalado para atender a demanda de funções específicas de um aplicativo.

Arquitetura de Microsserviços



Arquitetura



Estratégia

Cientista de Dados

Cientistas de Dados ***recebem uma enorme massa de dados (estruturados e não estruturados)*** e usam suas habilidades em Matemática, Estatística, Ciência da Computação e Programação para ***limpar, tratar e organizar os dados.***

Em seguida, eles aplicam suas capacidades analíticas – Machine Learning, Inteligência Artificial, conhecimento de negócio, ceticismo de suposições existentes – ***para descobrir soluções para os desafios de negócios.***

Engenheiro de Dados

Um Engenheiro de Dados é o profissional dedicado ao ***desenvolvimento, construção, teste e manutenção de arquiteturas, como um sistema de processamento em grande escala.***

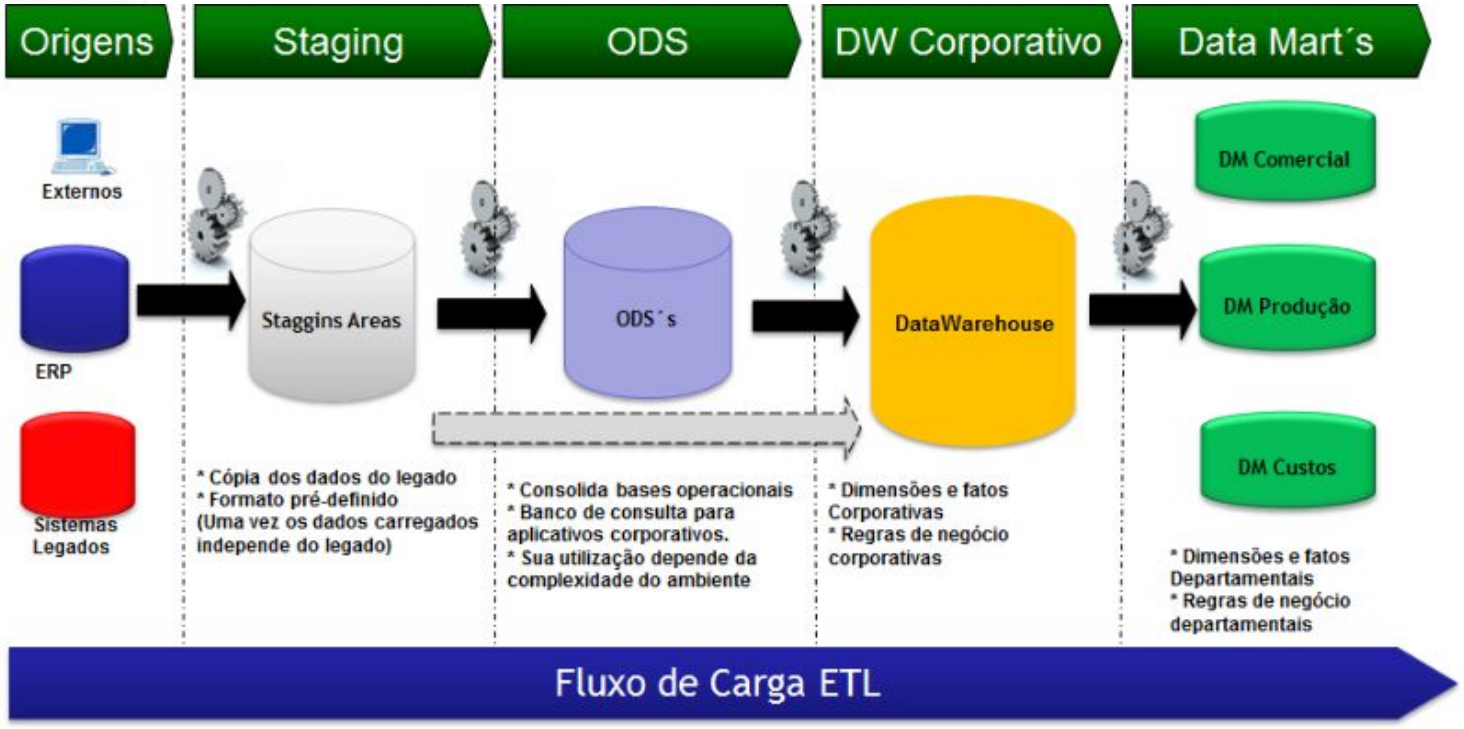
O Engenheiro de Dados é responsável por criar o pipeline dos dados, ***desde a coleta, até a entrega*** para análise ou para alimentar um produto ou serviço baseado em análise preditiva já em produção.

Engenheiro de Dados x DBA

As atividades desempenhadas pelo Engenheiro de Dados englobam principalmente a já conhecida rotina de um DBA, porém acrescidas de muitas outras tarefas, tais como: ***manutenção de sistemas de banco de dados relacionais e não-relacionais, ETL (Extração, Transformação e Carga), soluções de Data-Warehouse e Datamart, modelagem de dados e armazenamento em nuvem.***

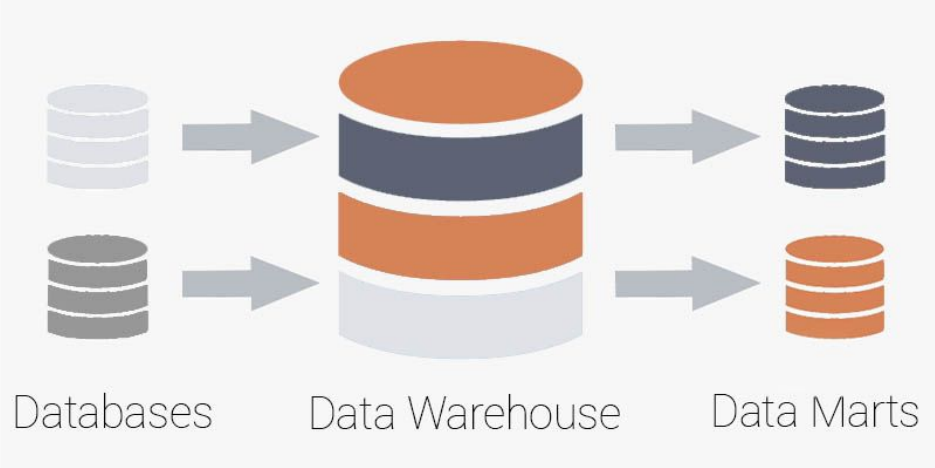
Além disso, diversas tecnologias fazem parte do dia a dia desse profissional, tais como: Oracle, MSSQL, MySQL, PostgreSQL, Neo4J, MongoDB, Cassandra, Sqoop, HDFS, Hive e muitas outras.

Business Intelligence



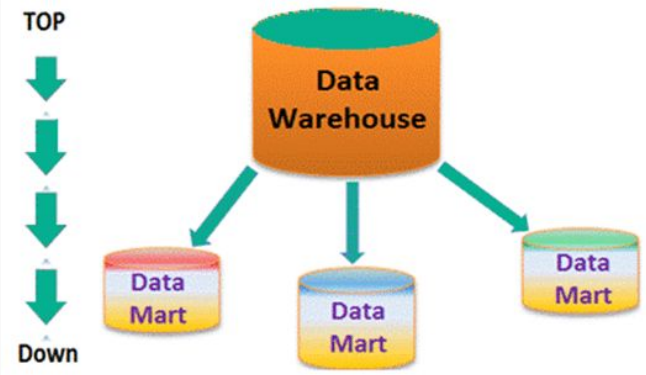
Data Warehouse (DW)

- ✔ Data Warehouse é um ***depósito de dados digitais que serve para armazenar informações detalhadas relativamente a uma empresa, criando e organizando relatórios através de históricos,*** que são depois usados pela empresa para ajudar a tomar decisões importantes com base nos fatos apresentados.



Data Mart

- ✔ Um Data Mart é uma ***subdivisão ou subconjunto de um DW.*** Os Data Marts são como pequenas fatias que armazenam subconjuntos de dados, normalmente organizados para um departamento ou um processo de negócio.
- ✔ Normalmente, o Data Mart ***é direcionado para uma linha de negócios*** ou equipe, sendo que a sua informação costuma pertencer a um único departamento.



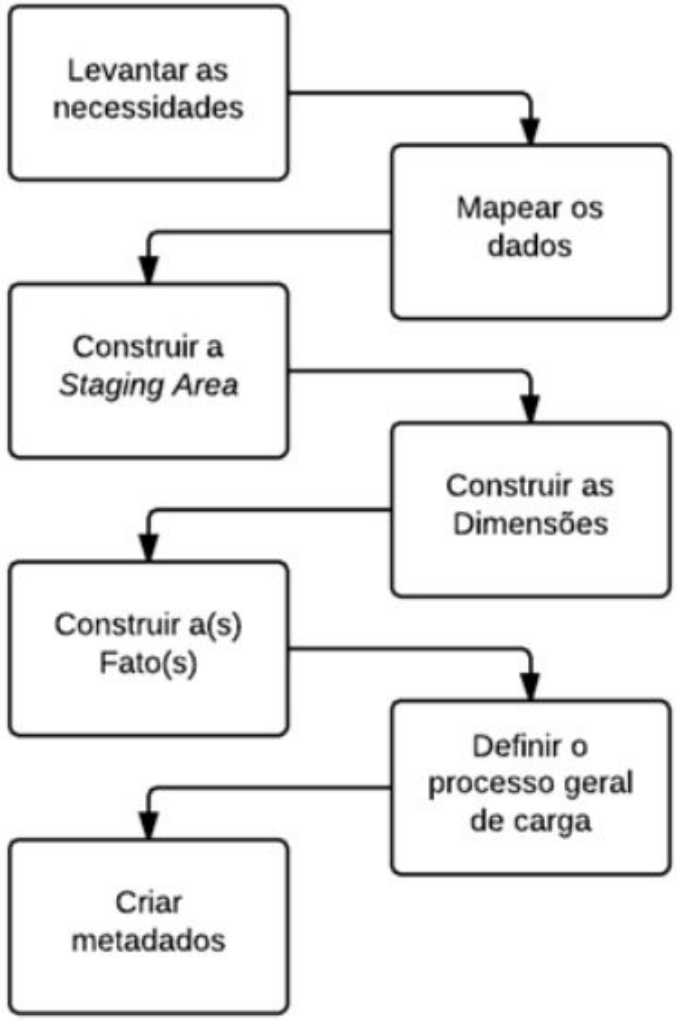
OLAP

Online Analytical Processing

OLAP (Online Analytical Processing ou Processo Analítico em Tempo Real) é uma das ferramentas mais usadas para a exploração de um Data Warehouse. O OLAP possibilita alterar e analisar grandes quantidades de dados em várias perspectivas diferentes. As funções básicas do OLAP são:

- ✔ Visualização multidimensional dos dados.
- ✔ Exploração.
- ✔ Rotação.
- ✔ Vários modos de visualização.

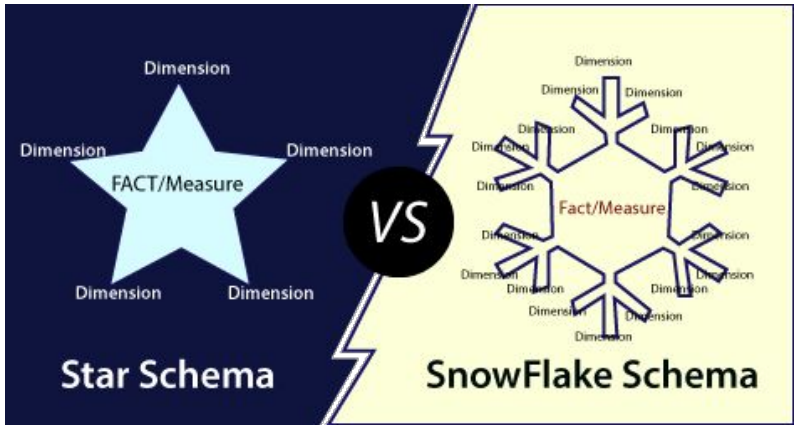
Etapas na Construção de um DW



Considerações sobre os Modelos Dimensionais

✔ **Modelos Star Schema *(mais usado)*:**

- *Dimensões Desnormalizadas.*
- Voltado para acessos com performance.
- Hierarquias achatadas.
- Mais simples e mais fácil navegação.
- Utiliza mais espaço repetindo as mesmas descrições ao logo de toda a tabela.



✔ **Modelo Snowflake:**

- *Normalizado.*
- Hierarquias mantidas.
- Muitas tabelas Muitas junções – 1:N..
- *Reduz o espaço de armazenamento dos dados dimensionais, mas acrescenta várias tabelas, deixando-o mais complexo.*
- **Acesso mais lento que no StarSchema.**

Definição ETL

Extract – Extrair.

Transform – Transformar.

Load – Carregar.

Definição ELT

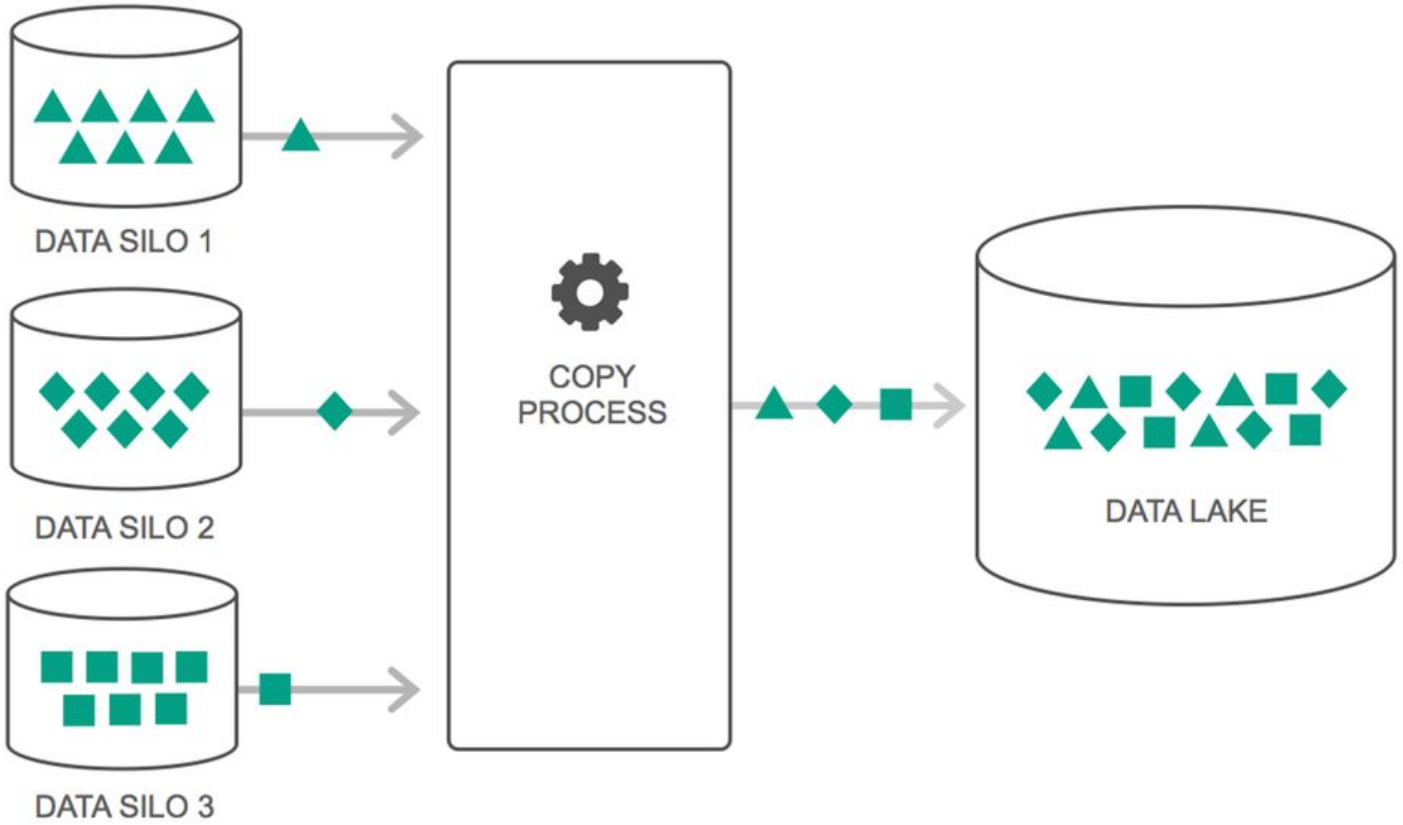
Extract – Extrair.

Load – Carregar.

Transform – Transformar.

O ***ELT*** é um processo de dados usado para replicar dados de uma fonte para um banco de dados de destino, sendo uma evolução ETL, pois torna o processo de replicação de dados muito menos complexo, uma vez que o passo de transformação é realizado após os dados estarem no destino.

Data Lake



Data Lake

	Data Warehouse	Data Lake
Dados	· Estruturados · Processados	· Estruturados / Semi-estruturados / Não estruturados · Não processados (em estado bruto)
Processamento	· Esquema de dados gerado no momento da escrita	· Esquema de dados gerado no momento da leitura
Armazenamento	· Alto custo para alto volume de dados	· Criado para ser de baixo custo, independente do volume de dados
Agilidade	· Pouco ágil, configuração fixa	· Bastante ágil, pode ser configurado e reconfigurado conforme necessário
Segurança	· Estratégias de segurança bastante maduras	· Ainda precisa aperfeiçoar o modelo de segurança e acesso aos dados
Usuários	· Analistas de Negócios	· Cientistas e Analistas de Dados

Data Lake no NoSQL

Um Data Lake pode residir em Hadoop, NoSQL, Amazon Simple Storage Service, Banco de Dados Relacional, ou combinações diferentes deles.

Alimentado por fluxos de dados (data streams).

Data Lake tem muitos tipos de elementos de dados, estruturas de dados e metadados no HDFS sem levar em conta a importância, IDs ou resumos e agregados.

Importante entender a natureza variada dos dados do Data Lake em relação ao metamodelo do banco de dados perspectiva, NoSQL:

- Semiestruturado.
- Chave: valor (principalmente) com sua estrutura hierárquica.
- A chave e o nome da coluna são partes essenciais da maioria do NoSQL.

Mais frequentemente um Data Lake é mantido no Hadoop e alimentado de ou para NoSQL:

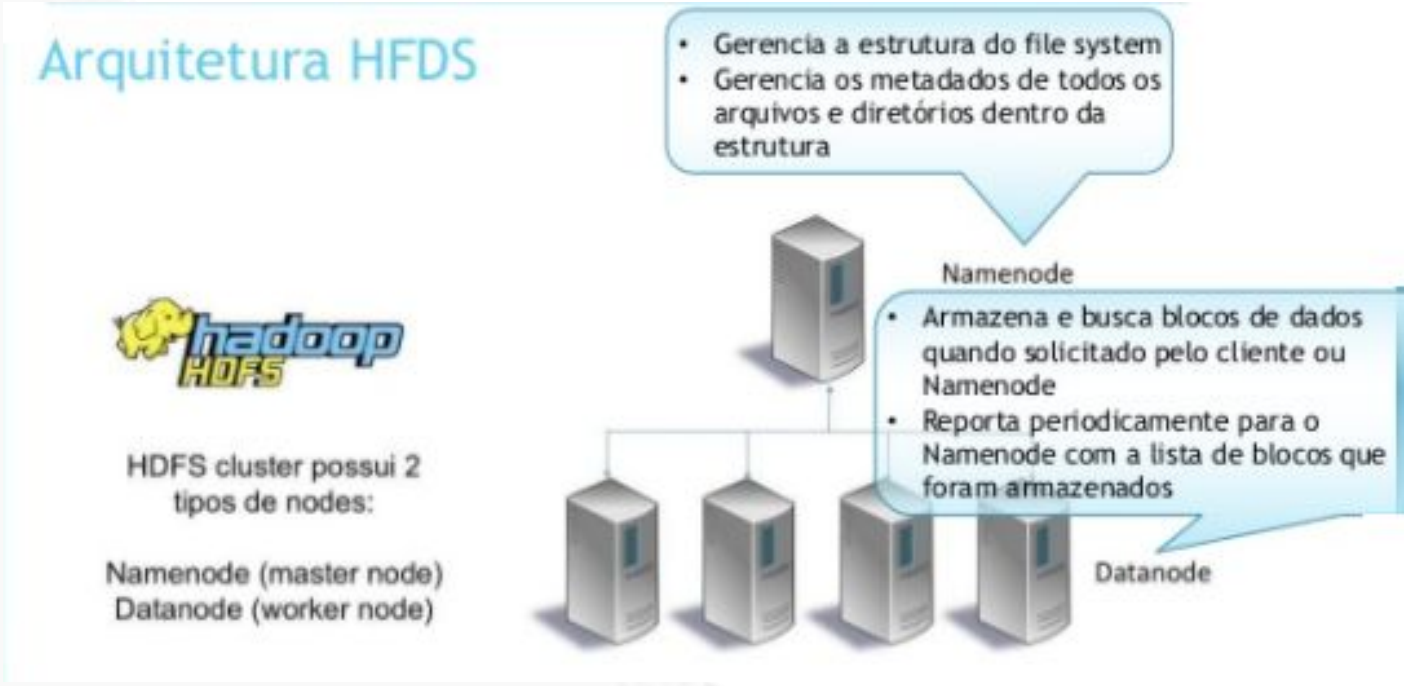
- NoSQL é um armazenamento de dados operacional, não analítico.

NoSQL e BI

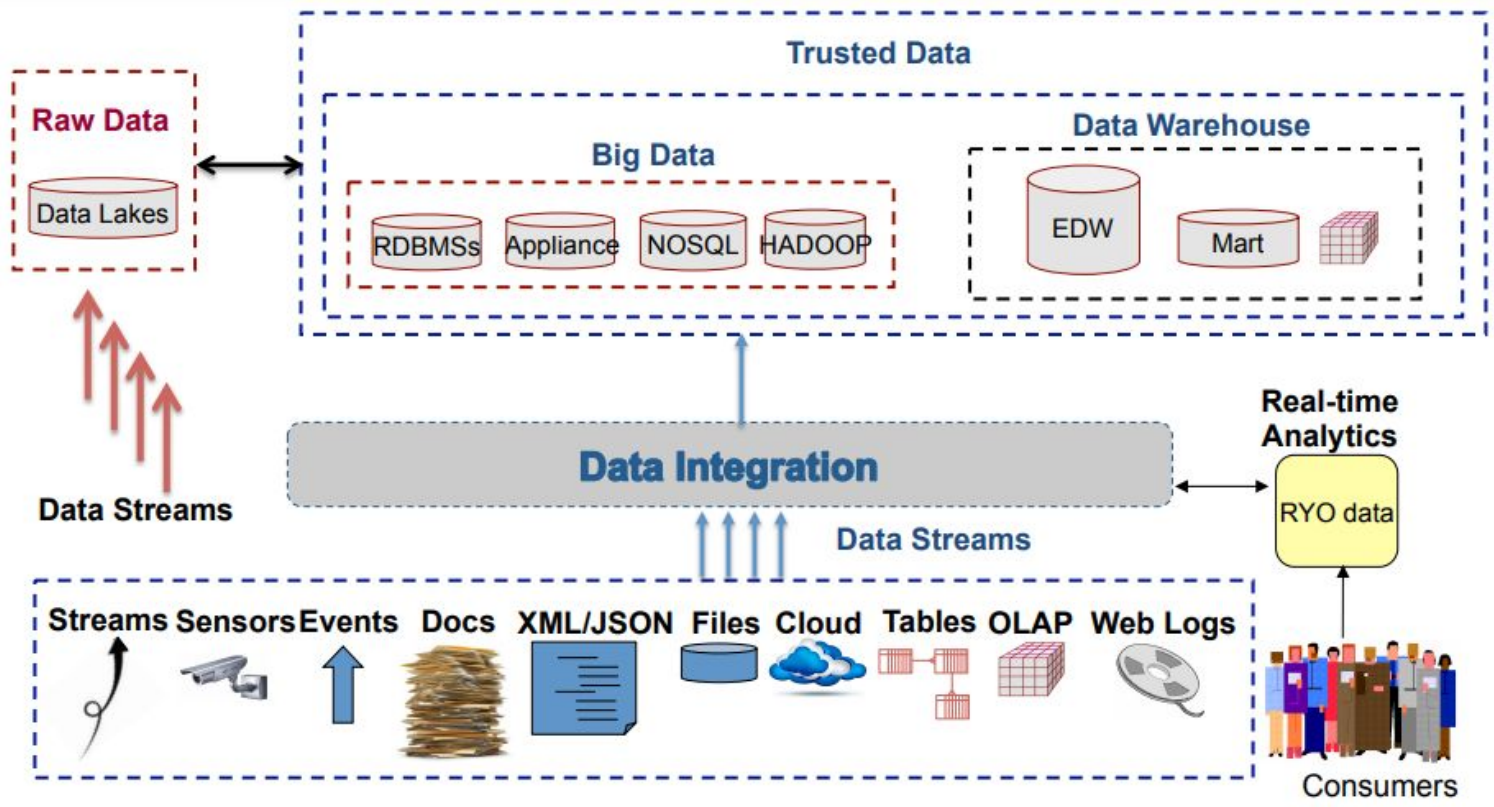
Os bancos NoSQL ainda não oferecem ferramentas para BI.

Existem algumas abordagens para BI e NoSQL.

Hadoop

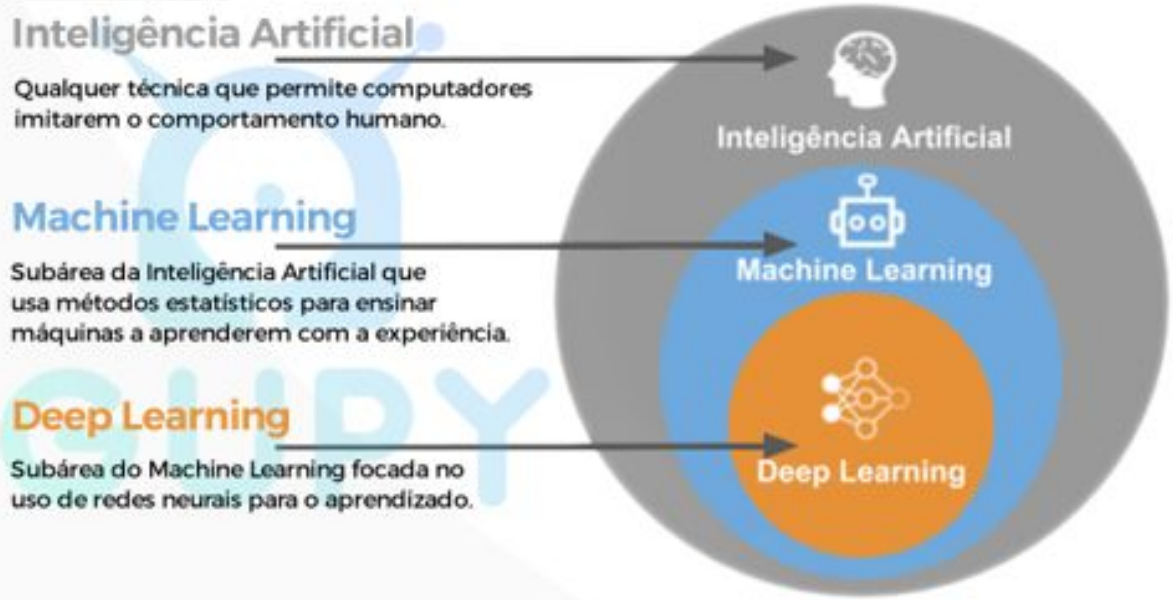

Arquitetura HFDS
• Gerencia a estrutura do file system
• Gerencia os metadados de todos os arquivos e diretórios dentro da estrutura
Namenode
• Armazena e busca blocos de dados quando solicitado pelo cliente ou Namenode
• Reporta periodicamente para o Namenode com a lista de blocos que foram armazenados
Datanode
HDFS cluster possui 2 tipos de nodes:
Namenode (master node)
Datanode (worker node)

NoSQL e BI

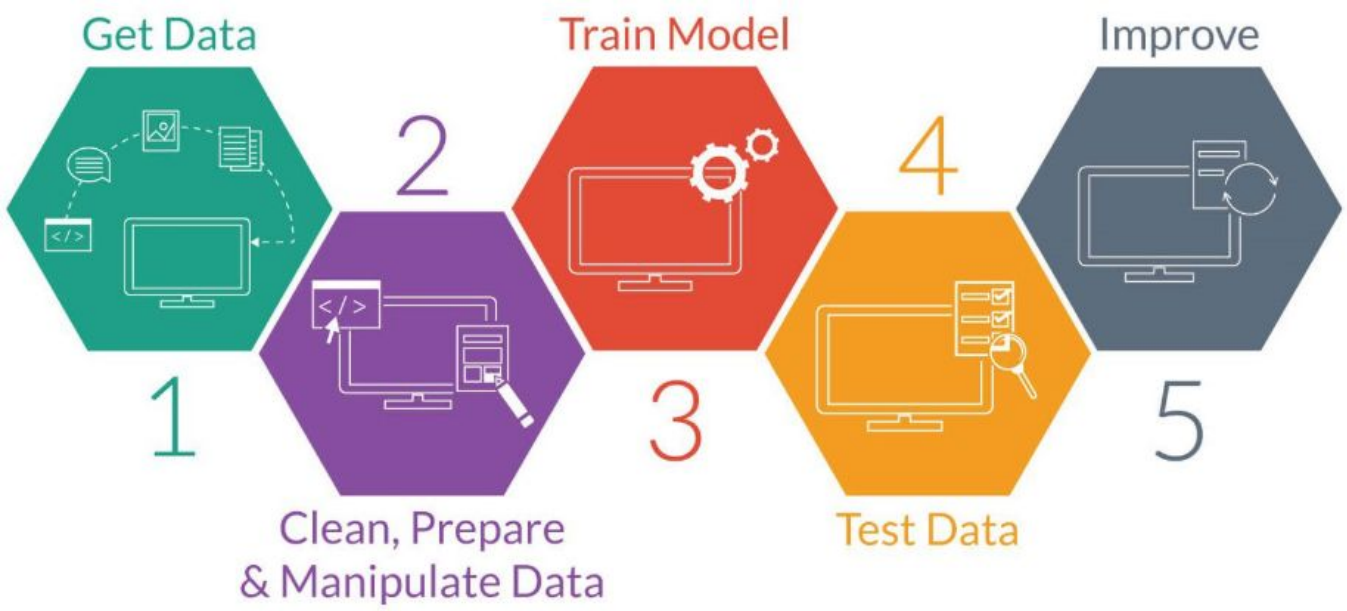

Trusted Data
Raw Data
Data Lakes
Big Data
RDBMSs
Appliance
NOSQL
HADOOP
Data Warehouse
EDW
Mart
Data Streams
Data Integration
Real-time Analytics
RYO data
Data Streams
Streams
Sensors
Events
Docs
XML/JSON
Files
Cloud
Tables
OLAP
Web Logs
Consumers

Machine Learning x Deep Learning

O aprendizado de máquina e o aprendizado profundo *começam com dados de treinamento e teste e um modelo,* e passam por um processo de otimização para encontrar os pesos que tornam o modelo o melhor para os dados.



Dados


Get Data
Train Model
Improve
2
4
1
3
5
Clean, Prepare
& Manipulate Data
Test Data

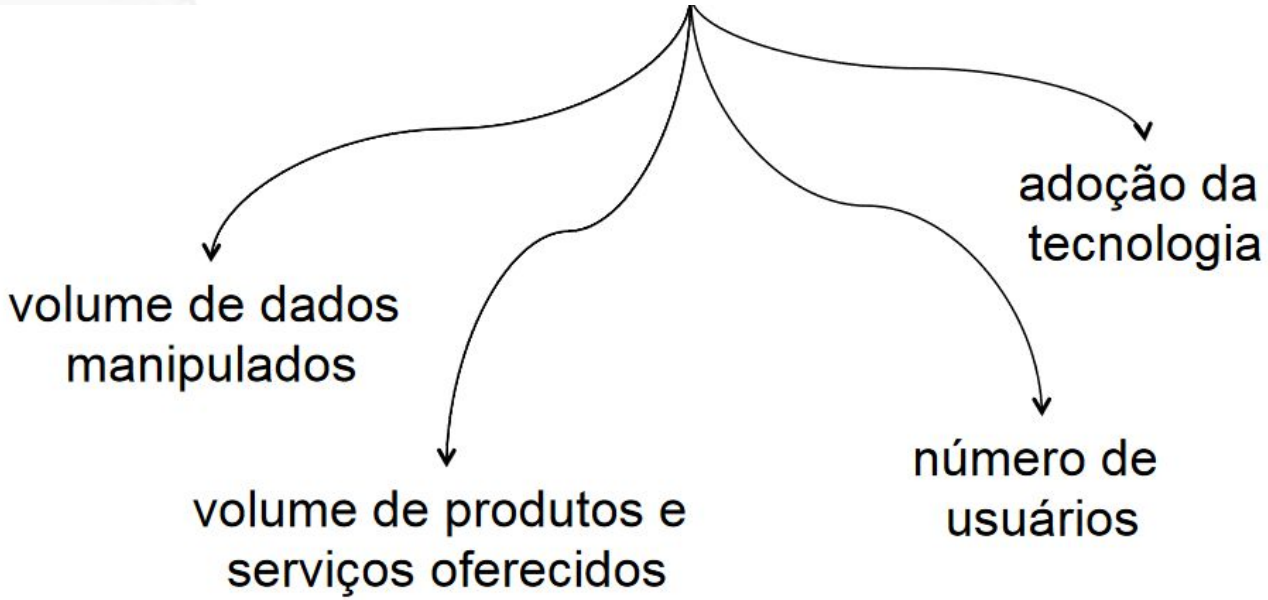
Preparação dos Dados

Não existem dados limpos na natureza. Por exemplo, observe os dados e exclua quaisquer colunas que tenham muitos dados ausentes.

1. Selecione as colunas mais importantes para sua previsão (CFS).
2. Exclua quaisquer linhas que ainda tenham dados ausentes nas colunas restantes.
3. Corrija erros de digitação óbvios e mescle respostas equivalentes. Por exemplo, Brasil, Brazil e BR devem ser mesclados em uma única categoria.
4. Exclua linhas que tenham dados fora do intervalo. Por exemplo, se você estiver analisando dados para pessoas de avançada idade (idosos) talvez não valha a pena manter dados fora dessa faixa.

Conceitos Básicos

Arquitetura



Tuning em Banco de Dados

Em TI, Tuning refere-se basicamente ao conceito de propor e aplicar mudanças *visando otimizar o desempenho na recuperação ou atualização de dados.*

Em curtas palavras, *Tuning é sinônimo de otimização.*

Para fazer um bom trabalho de Tuning, é necessário executar criteriosamente os seguintes processos:

- Entender o problema.
- Elaborar o diagnóstico.
- Aplicar as dicas e técnicas de otimização (que se aplicam ao diagnóstico elaborado).

Tuning em Banco de Dados

1. **Planejamento de performance:**
   Definição e configuração do ambiente em que o BD será instalado, considerando-se os seguintes itens: Hardware, Software, Sistema Operacional e Infraestrutura de rede.

2. **Tuning de instância e BD:**
   Ajuste de parâmetros e configurações do BD (atividades que fazem parte do trabalho de um DBA).

3. **SQL Tuning:**
   Otimização de instruções SQL.

SQL Tuning: otimização de instruções SQL

- Identificar uma query lenta ou “pesada” no seu banco de dados.
- Identificar as consultas que utilizam um determinado índice através de alguma ferramenta – rebuild de índices.
- Uso de comandos como order by, colunas calculadas, delete x truncate etc.

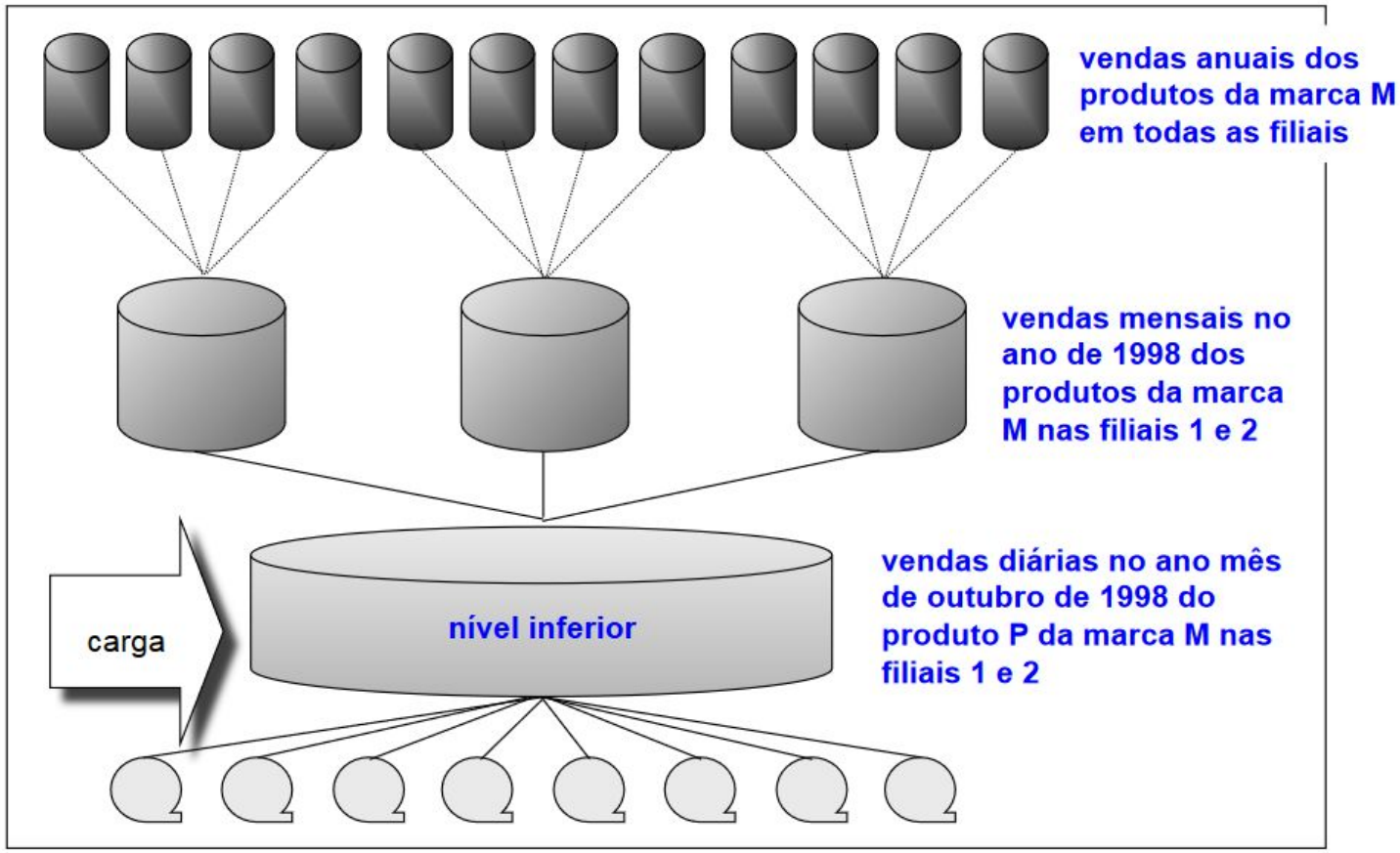
Granularidade

Grau de detalhamento em que os dados são armazenados em um nível.

Questão de projeto muito importante, pois:

- Impacta no volume de dados armazenado.
- Afeta as consultas que podem ser respondidas.

Características dos Dados

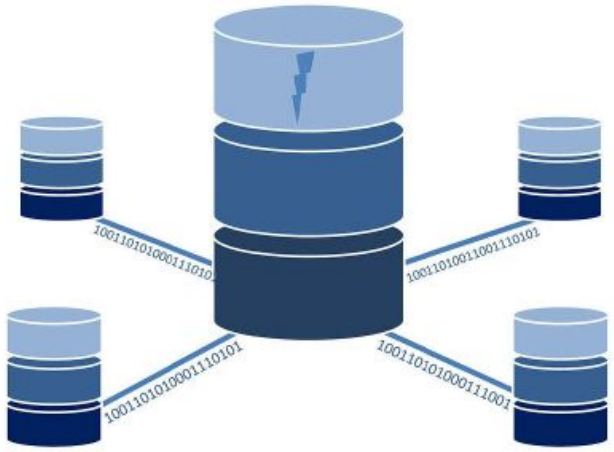


NoSQL

- ✔ NoSQL (originalmente se referindo a "no SQL": "não SQL" ou "não relacional", posteriormente estendido para ***Not Only SQL*** - Não Somente SQL) é um termo genérico que representa os bancos de dados não relacionais.
- ✔ Bancos de dados NoSQL são cada vez mais usados em ***Big Data*** e ***aplicações web de tempo real.***

NoSQL – Banco de Dados de 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 troca de dados.
  - ✔ Mais compacto e legível que XML.



MongoDB

mongoDB.

- ✔ Open source.
- ✔ Multiplataforma.
- ✔ Escalável.

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

MongoDB – Comandos

```
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

MongoDB – Comandos

```
> 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" }
```

Desafio